VICTORIA UNIVERSITY OF TECHNOLOGY

DEPARTMENT OF COMPUTER AND MATHEMATICAL SCIENCES

Implementation Options for Reliable
Satellite Broadcast Transmission (RSBT)

Ivan Z. Jutrisa, Nalin Sharda and Pietro Cerone

(63 COMP 21)

December, 1995

(AMS : 94A99)

TECHNICAL REPORT

VICTORIA UNIVERSITY OF TECHNOLOGY
DEPARTMENT OF COMPUTER AND MATHEMATICAL SCIENCES
P O BOX 14428
MCMC
MELBOURNE, VICTORIA 8001
AUSTRALIA

TELEPHONE (03) 9688 4492
FACSIMILE (03) 9688 4050

Footscray Campus

Implementation Options for Reliable Satellite Broadcast Transmission (RSBT)

Ivan Z. Jutrisa, Nalin Sharda and Pietro Cerone
{ivan, nalin, pc}@matilda.vut.edu.au
Department of Computer and Mathematical Sciences
Victoria University of Technology
P.O. Box 14428, MCMC, Melbourne, Vic, 3000, Australia

Abstract: This paper discusses implementation options for a new protocol called "Reliable Satellite Broadcast Transmission" (RSBT), being developed for the Omnicast Digital service offered by Optus Communications. The sending satellite interface will receive data from the LAN as per FTP. Then, using RSBT, it will broadcast the data to receiving satellite interfaces, which will transmit the data on their LANs using FTP. Omnicast Digital is a one-way service, thus, separate return links are required to facilitate reverse error control, for which several options are discussed in detail. XMelba, the case tool used to model RSBT, is also outlined.

1. Introduction

Many large organisations disseminate information from their head office to regional offices. Existing transmission media and services provide these organisations with options that best suit their needs. In most cases, the selected media or service is sufficient for the requirements. In some cases though, tight constraints (time, quality, etc.), and variables such as file size, may render existing systems inefficient.

The need for Reliable Satellite Broadcast Transmission (RSBT) was mooted by the Bureau of Meteorology. The Bureau collects weather data from all over the country and processes it at its head office in Melbourne. The resulting data (several types) are then compressed, and disseminated to all Regional Offices (one in each state capital city) via terrestrial links [1].

The usefulness of weather data and its derivatives deteriorates rapidly with time. Therefore, data processed centrally must be transmitted without delay for the Regions to take full advantage of the information. Delays in transmission will be inevitable due to the large amount of data currently available and the projected increase in weather data over the next five years [2]. Hence, a new method of disseminating date is required.

A potential alternative method for broadcasting data to Regional Offices is a new satellite service offered by Optus Communications called Omnicast Digital. Omnicast Digital is one-way broadcasting service which transmits data at rates between 64 Kbps and 2 Mbps in a continuous flow [3]. Preliminary analysis based on information from the Bureau of Meteorology and Optus Communications, suggest that Omnicast Digital could prove to be more efficient, reliable and cheaper than the currently used terrestrial links.

A communication protocol for the effective use of the Omnicast Digital service is not available from Optus or any other vendor. To explore the possibilities of using the Omnicast Digital service, the Bureau of Meteorology would like to develop a suitable protocol (RSBT) that will include return links for messages and acknowledgments. The successful development of RSBT would satisfy the Bureau's requirements, and provide it with a viable alternative for disseminating data. This paper discusses the various options available for implementing such a protocol.

2. Purpose

RSBT is to be used for reliable broadcast transmission over a satellite link at a speed of up to 2 Mbps in a continuous flow. The type of files to be broadcast range from plain text to compressed image data. These files will be broadcast from one site only; all other sites will only ever transmit acknowledgments.(hereinafter meaning ACK/NAK), and possibly control messages (implementation dependent).

RSBT is to be used in a real-time system, where files are generated by the sender and disseminated to all receivers for further processing. The process will begin with automatic transmission of centrally processed data to the satellite interface (described in section 3.1), using FTP/TCP (FTP at the application layer and TCP at the transport layer). RSBT will then broadcast the data across a permanent satellite link to each receiving satellite interface, which will temporarily store the received data on disk. And finally, FTP/TCP will transmit the data to its final destination on the LAN. This three step transmission is also used by Fairhurst [4], where the Packet Oriented Protocol packets are converted to HDLC and back.

The Bureau's current daily transmission load is 221 Mbytes of compressed data. The peak hourly load is 27.3 Mbytes (compressed) [2], and the maximum file size is 3 Mbytes. As mentioned in section 1, data must be broadcast without delay for the regions to take full advantage of the information. This is particularly important with satellite data, which is given high priority, and should be available in Regional Offices within two to three minutes of the end of central processing [2].

3. Assumptions About the Environment

RSBT is designed to function in WANs where all information is disseminated from one site (head office). To complete a file transfer, several media will be used, and RSBT will have to interface with the application protocols used on the LANs (such as FTP).

3.1. System Configuration

The system configuration, shown in figure 1, will consist of:

1. a LAN at the sending and receiving stations,
2. a digital modem at each station,
3. a satellite receiver at each receiving station,
4. a satellite interface at each station.

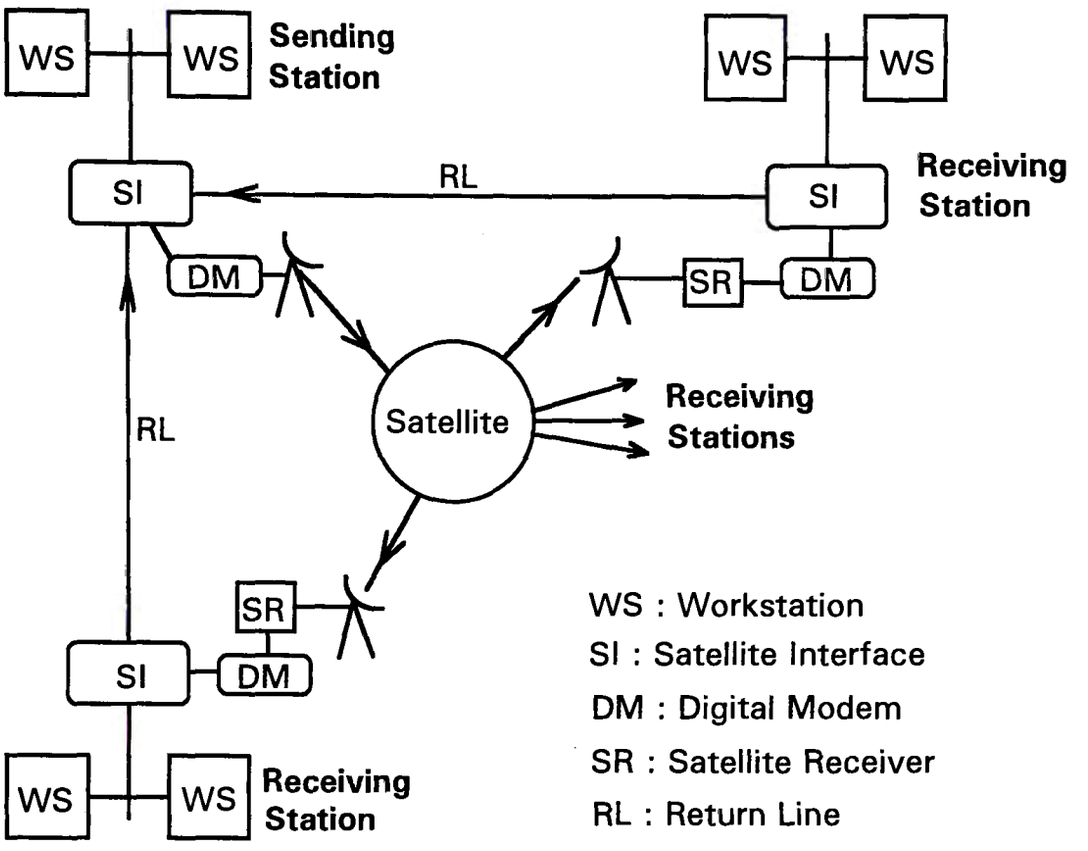

Figure 1. *System configuration and links.*

Each satellite interface will be a dedicated UNIX box, with its own hard disk for temporary storage of data. As with the Network Block Transfer (NETBLT) protocol [5], each receiving satellite interface will need multiple buffers, which will be used in a cyclic manner to cope with the continuous incoming data stream (see section 3.2).

The RSBT protocol will reside in each satellite interface, which will be connected to its respective LAN in the same way that other workstations are connected. Thus, it will have its own IP address, allowing it the same internet connections that can be used as return links.

3.2. Service Used

RSBT is being developed for the Omnicast Digital service, provided by Optus Communications. The major features of the service are [3]:

1. transmission is one-way digital; return links must be established separately,
2. transmission is in a continuous stream; all frames may be transmitted contiguously,
3. transmission speeds are between 64 Kbps and 2 Mbps in 64 Kbps increments,
4. Quadrature Phase Shift Keying (QPSK) is used to modulate the input stream,
5. Forward Error Correction (FEC) is used, with a code rate of 1/2,
6. connection is permanent; if it is lost, it may take up to one minute to regain it.

To establish a connection, each receiver progressively seeks a digital carrier, looking for a unique carrier indent pattern. Once locked-on it maintains a lock over +- 3 MHz, providing a permanent connection. If that connection is lost, it may take up to one minute to regain it [3].

Because overall performance will depend on the customer provided receiving equipment, only the minimum signal level received from the satellite is specified, the performance limit (Bit Error Rate, etc.) is not[3].

3.3. *Broadcasting*

As the Omnicast Digital service provides a permanent link between the sending and receiving stations, no addressing is required. Instead, the data transferred will carry identification of its type, and its release (in regards to data that is disseminated at regular time intervals). The identification will also be used by the receiving stations to recognise the transmission of a new data set. This would occur in the event of major transmission errors, where the sender has given up retransmitting corrupted frames, and commenced transmitting a new file.

To optimise transmission efficiency, no flow control is to be used. It has been shown that the use of flow control over a satellite link [6] (and in terrestrial WANs [7][8]) can be a major cause of transmission inefficiency, due to the time spent waiting for acknowledgments. Therefore, as with the Burst Mode Protocol [7][8], all frames may be transmitted contiguously.

As depicted in figure 2, the sending satellite interface will use RSBT to communicate with each receiving satellite interface, while communication between each LAN and its respective satellite interface will be as per FTP/TCP. Hence, some compatibility between RSBT and FTP will be required.

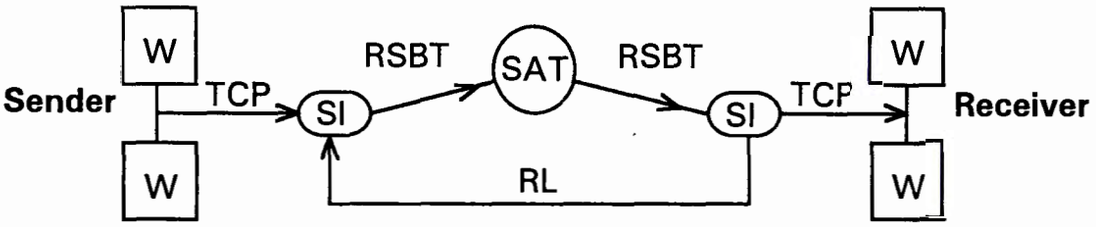

Figure 2. *Transmission path and protocols used.*

3.4. *Return Links*

Omnicast Digital does not provide for reliable transmission. Therefore, RSBT to will have to use separate return links for acknowledgments and other control messages. These return links may be either separate satellite channels, or existing terrestrial links with X.25 or HDLC connections.

4. Frame Format

The format of RSBT frames transmitted by the sender (carrying data) will differ to the frames transmitted by receivers (carrying ACK/NAK). Receivers will send a frame containing only a file identification, a frame sequence number and a flag (ACK, NAK), as shown in figure 3. This frame will be enclosed within the frame of the protocol used by the return links.

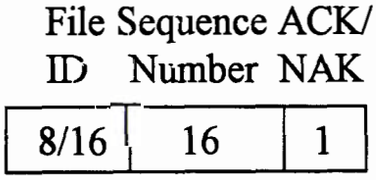

Figure 3. *Format of frames sent by receivers*

Frames transmitted by the sender, as shown in figure 4, will have the following format:

1. flag, 8 bits; start-of-frame delimiter, complemented by zero bit insertion [9],
2. information ID, 8/16 bits; file identification, and control information (see section 5.2) if a control frame is not used. In the event of major reception difficulties, it would allow receivers to detect the transmission of a new file. A similar ID is used by NETBLT in the connection process [5],
3. frame sequence number, 16 bits,
4. length, 16 bits; the number of data octets,
5. more data, 1 bit; a flag for more data to follow; also used by NETBLT [5],
6. header FCS, 16/32 bits (optional); a cyclic redundancy check (CRC) for the header only, which is also used by NETBLT[5] and ATM [10]. The additional header would improve the efficiency of the acknowledgment process,
7. data,
8. data FCS, 16/32 bits; CRC, covering the entire frame if a header FCS is not used, otherwise it will cover the data section only.

Flag	File ID	Sequence Number	Length	More Data	FCS-H	Data	FCS-D
8	8/16	16	16	1	32		32

Figure 4. *Format of frames sent by sender*

The RSBT frame is expected to be quite large, compared to frames of other protocols. This is because the Omnicast Digital service provides forward error correction to minimise errors, which is particularly important for large frame sizes. Furthermore, unlike protocols used on terrestrial links, RSBT is not restricted by buffer sizes of intermediate nodes, and traffic.

5. Protocol Requirements

RSBT is to provide reliable transmission to multiple receivers, interfacing with FTP/TCP at the sending and receiving stations. It does not use addressing, routing, or flow control. Its main functions: session checking, frame preparation, and error control, are described below.

5.1. Sessions

The link between the sender and the receivers is provided by the Omnicast Digital service. If a particular receiver loses its connection, the service will automatically begin a reconnection process, which may take up to one minute [3].

Before broadcasting data, the sender may check for open sessions by broadcasting a control frame several times over. Upon receiving this control frame, which could contain the transmission parameters (see section 5.2), receivers will send a response. If all receivers do not respond, depending on the implementation, the sender may delay the broadcast, allowing receivers time to reconnect. Alternatively, the sender may proceed with the broadcast, knowing that retransmission may be required for those receivers with lost sessions.

If the sender does not intend to delay transmission because of lost sessions, the check for open sessions may not serve much purpose. This is because receivers that regain their session during transmission will sends NAKs for the frames not received (see section 5.3) and carry on reception as normal.

5.2. Frame Preparation

The FTP file transfer from the sending LAN to its satellite interface will include normal control commands. RSBT must pass these control commands to each receiving satellite interface, so that they can transmit the file to their respective LANs in the correct manner. The FTP control commands are [11]:

1. mode: stream, block, and compressed,
2. file structure: structured, unstructured, and random access,
3. data type: binary, text (ASCII and EBCDIC, with format control), and local type.

FTP control commands may be encapsulated into each RSBT frame header, or they may be sent in a control frame preceding data transmission. The control frame, which may also be used as a session check, would require priority in the retransmission queue.

The RSBT frame length may be independent of the FTP packet length, or it may contain an exact number of FTP packets. If the RSBT frame length is independent of FTP, any FTP data headers [11], which provide further information about the data, would have to remain as part of the data stream. The existence of these FTP data headers depends on the transmission type. eg. block mode includes the block length and a descriptor [11], which are required by receivers to segment the data into its original FTP packets for transmission to the LAN.

Having an exact number of FTP packets in a RSBT frame would allow the sender to remove or curtail any FTP data headers in the transmission. It would also require less processing by receivers, and allow them to transmit some FTP packets to the LAN sooner.

5.3. Error Control

The Omnicast Digital service uses FEC to minimise errors [3]. This is sufficient for most random errors, but not good enough to trap and correct burst errors. Therefore, the transmission, in regards to error control, must be enhanced by the use of reverse error control.

To facilitate reverse error control, RSBT frames will include either one or two FCS fields. If only one FCS field is used, which would cover the complete frame, corrupted frames could not be negatively acknowledge. This is because, in most cases, the integrity of the sequence number would not be guaranteed.

If two FCSs are used, one for the data, and another for the header (as with NETBLT[5] and ATM[10]), corrupted frames could be acknowledged if the header is not corrupted. The additional overhead would have insignificant effect on transmission efficiency, as RSBT frames are expected to be quite large. On the positive side though, the early transmission of NAKs would improve efficiency if several frames at the tail end of the transmission are corrupted. This improved efficiency will be minimal if all frames are acknowledged (section 5.3.1), as the sender will retransmit frames after a time-out period.

Receivers will send NAKs for all frames corrupted or missing in the sequence. The sender could retransmit these frames using either selective repeat or go-back-n; unless otherwise stated, selective repeat will be the assumed retransmission strategy.

Receivers must also send ACKs to indicate successful reception. The options are:

1. acknowledge all frames,
2. acknowledge selected frames:
   a) acknowledge last frame only,
   b) acknowledge first and last frames,
   c) acknowledge lapse in transmission.

5.3.1. Acknowledge All Frames

This is a common approach, and is used by protocols such as TCP [12] and HDLC [13] [9] in point-to-point communication. It requires either a ACK or NAK to be sent for each frame received. In addition, if selective repeat retransmission is used, NAKs would also be sent for frames missing in the sequence.

RSBT differs from TCP and HDLC, in that it is a point-to-multipoint protocol. Therefore, RSBT will receive acknowledgments from multiple receivers, and thus will need to maintain a more complex array of variables during transmission, which include:

1. receiver list; a list of all receivers and the frames that they have acknowledged. Frames not acknowledged by all receivers would be retransmitted. An alternative would be to count the number of ACKs received for each frame, disregarding the identity of receivers,
2. retransmission list; the sequence numbers of frames transmitted. These frames would be retransmitted in a round-robin manner, and removed from the list as they receive the correct number of ACKs,
3. timer (optional); used by the sender to trigger a retransmission of a frame for which no response has been received in a specified time-out period. The timer would not be required if frames in the retransmission list are transmitted over and over without delay. This would be quite appropriate for RSBT, because Omnicast Digital provides a dedicated channel which would otherwise sit idle during lapses in transmission,
4. received list; the sequence numbers of all frames received (by receivers).

Although this method of acknowledgment would create some congestion at the sending station, there would be a strong case for its use, if return links were unreliable.

5.3.2. Acknowledge Selected Frames

Acknowledging each frame in point-to-multipoint transmission could overwhelm the sender with acknowledgments if there are many receivers. Thus, techniques that acknowledge selected frames only are required.

5.3.2.1.Acknowledge Last Frame Only

The basic philosophy of this method is that no news is good news. Receivers will send a NAK for each frame corrupted or missing in the sequence. In addition, receivers will retransmit the NAK if the requested frame is not received in a specified time-out period (as with NETBLT[5]). The time-out period should be dynamic, so that it can take into account the transmission or retransmission of other frames in the queue at the sending station. This would be obvious from the continuous reception of other frames, without duplication.

Frames received error free will not be acknowledged, except for the final frame, which will be acknowledged only after all other frames have been received. Thus, the ACK acknowledges the complete file; when the sender receives all ACKs, the transmission will be complete. If one or more receivers do not receive any frames at all, or the last one or more frames in the sequence, then obviously, the number of ACKs received will not tally.

To overcome the problem of missing ACKs, considering that the sender will not know the nature of the transmission error, the sender may retransmit the file from the beginning. Though this will cause unnecessary delay, if in fact it is only the last frame that the receiver(s) is missing. Furthermore, the receiver(s) could fail to receive the last frame again, leaving the sender no wiser. Alternatively, the sender could retransmit the last frame over and over (within limit), until the receiver(s) eventually gets it. If it was only the last frame that was missing, the receiver(s) would send an ACK, otherwise it would send NAKs for all other frames missing in the sequence.

The variables that need to be maintained with this approach, include the retransmission list and received list as in section 5.3.1 (Acknowledge All Frames), and in addition:

1. receiver list; same as the list described in section 5.3.1(Acknowledge All Frames), except that only one response will be received from each receiver. Again, the alternative is to use an ACK counter,
2. timer; used by receivers to retransmit NAKs for frames not received in retransmission.

This method would greatly reduce the processing time and the incoming traffic at the sending station.

5.3.2.2.Acknowledge First and Last Frames

This method is an enhancement to acknowledging the last frame only(section 5.3.2.1). Each receiver would send an ACK for the first frame it receives. If the frame is corrupt, then a NAK would be sent instead. Furthermore, if the first frame received by a particular receiver, is not frame one, then a NAK for all preceding frames would also be sent.

The initial ACK (or NAK) would inform the sender as to how many and which receivers the data is getting to. If the number of receivers does not tally, the sender will know that after the file has been transmitted (including retransmission), it will have to be transmitted again for those receivers that have temporarily lost their sessions.

Transmission of frames with this method of acknowledgment, and transmission with acknowledgment of the last frame only, will not differ prior to all receivers with open sessions receiving the complete file. It is from this point that the two methods differ. With acknowledgment of the last frame only, the last frame will be transmitted over and over until a response is obtained from a receiver that has regained its session. While this method (acknowledge first and last frames) will retransmit the whole file over and over. The advantage of this is that a receiver that has just regained its session will commence normal reception immediately, instead of waiting for its response to trigger a transmission of other frames by the sender. This difference is illustrated in figure 5 (initial acknowledgments are not shown), with the assumptions: the round trip delay is equivalent to the transmission of four frames, and there are 20 frames in the file.

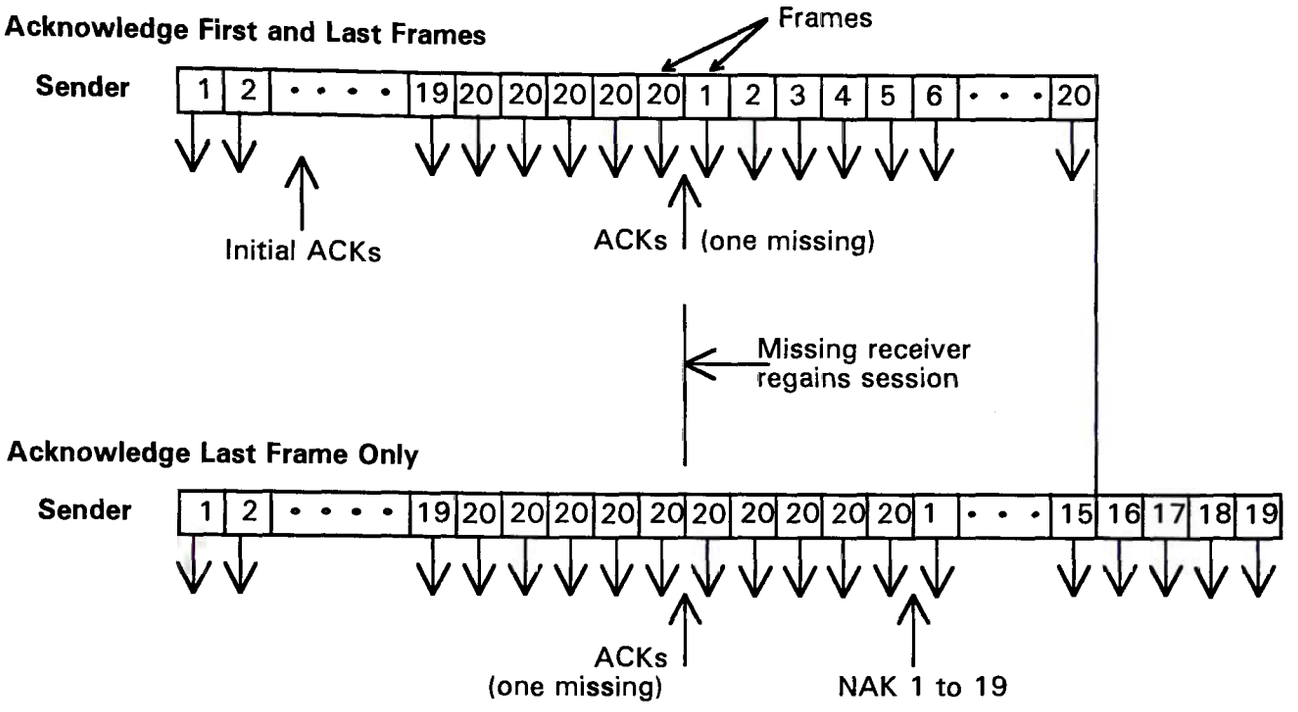

Figure 5. *The transmission efficiency of acknowledging the first and last frames, compared to acknowledging the last frame only.*

Figure 5 shows that all receivers with open session received the whole file error free. While the one receiver with a lost session, was reconnected, and began reception just after the last frame (20) was to be received. With acknowledging the first and last frames, the receiver will receive the remaining frames without requesting them. While with acknowledging the last frame only, the receiver will first have to send NAKs for the missing frames, which, in this scenario is less efficient.

The variables that are needed with this approach are the same as those described in section 5.3.2.1(Acknowledge Last Frame Only), except that the receiver list will hold two acknowledgments: the initial response, and an ACK for the last frame.

If this method of acknowledgment is used, it would be pointless to perform the session check (section 5.1), unless of course the initial intention would be to wait until all receivers have open sessions. Furthermore, if a control frame(section 5.2) is used, its session checking capabilities would be disregarded. Its purpose would be to deliver the FTP control commands, and nothing more.

5.3.2.3.Acknowledge Lapse in Transmission

This method differs to acknowledging the last frame only (section 5.3.2.1) in that it provides an additional NAK from receivers, for the frame expected in a transmission lapse. This additional NAK is also used by NETBLT [5] for expected buffers (groups of packets).

Another difference is that after the last frame in the retransmission list (say 16) has been transmitted, the remaining frames (17, 18, 19,....) in the sequence which were not requested, will be transmitted. Only then will the last frame be transmitted over and over. In this way, receivers that have temporarily lost reception, will receive the remaining frames quicker in some scenarios.

The additional NAK would be most useful when a receiver fails to receive several frames at the end of the sequence. For example, if an assumption is made, that the round trip delay is equivalent to the transmission of four frames, and that receivers will wait the time taken to transmit one frame before sending a NAK for the frame it is expecting. Then, if a receiver does not receive the last five frames of the transmission (say, frames 16, 17, 18, 19 and 20) the sender will receive a NAK for frame 16 just as it finishes transmitting the last frame (20). This would allow the sender to retransmit the lost frames without any delay. The efficiency of this method in overcoming this scenario is highlighted in figure 6.

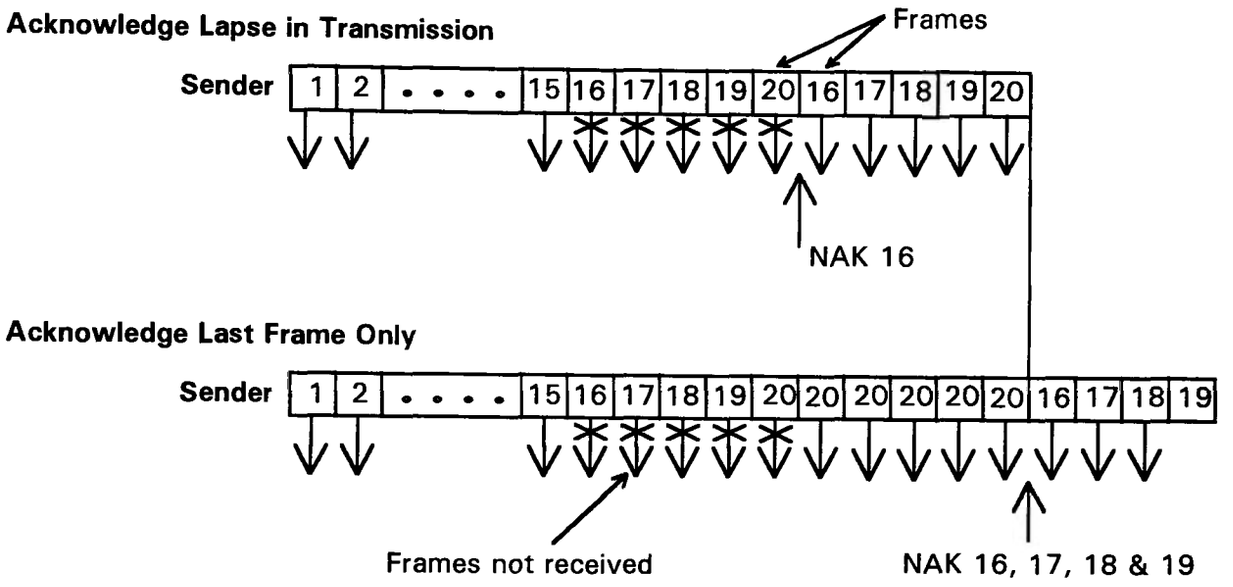

Figure 6. *The transmission efficiency of acknowledging a lapse in transmission compared to acknowledging the last frame only.*

The variables needed with this approach are the same as those described in section 5.3.2.1 (Acknowledge Last Frame Only), except that receivers will use a timer to trigger the transmission of NAKs, if reception stops.

5.4. Retransmission

Retransmission of lost or corrupted frames will be done after all frames have been transmitted once. Therefore, as with NETBLT [5], transmitted frames will not be kept in the buffer for retransmission purposes. If needed, they will be recopied from disk as transmission approaches its end, and room becomes available in the buffer.

The retransmission of frames may be done in two ways: selective repeat, or go-back-n. The go-back-n method is fine in point to point communication, but it has one major disadvantage when broadcasting, particularly if there are a few receivers with very bad reception. As depicted in figure 7, if data is broadcast over bad satellite channels (due to bad weather, inferior receiving equipment, etc.), and someone does not receive the first frame, retransmission begins from frame one. Then, if someone does not receive the second frame for the second time, retransmission begins from frame two, and so on. This could cause major delays for receivers not experiencing reception difficulties. Thus, selective repeat would be the more appropriate retransmission strategy for most implementations.

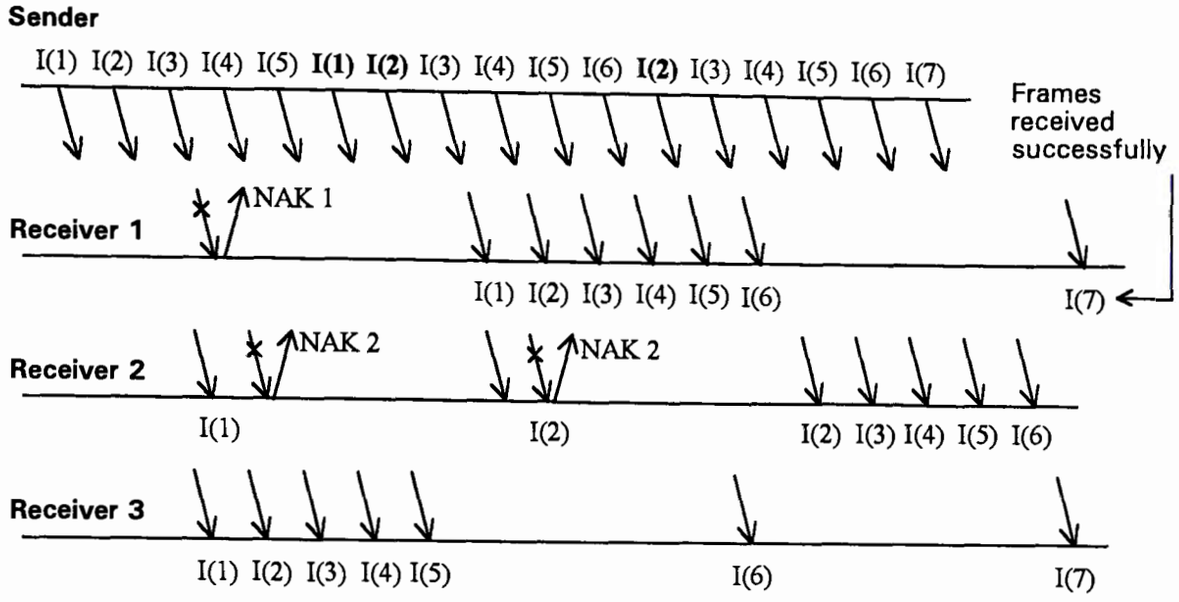

Figure 7. *Time sequence diagram illustrating the inefficiency of go-back-n retransmission.*

6. Protocol Development

In the early stages, the communications hardware available was rather simple, as were the protocols used. But, with continuous developments in hardware, modern communication protocols demand strong, sophisticated, and reliable media standards to cater for the complex nature of communication today. Hence, Formal Description Techniques (FDTs) are used for developing protocols for systems that are complex, concurrent, quality-critical, safety-critical, security-critical or standardised [14]

FDTs share a common basis for specifying behaviour, namely labelled transition systems. These are systems whose transitions between states are labelled with associated actions. An example of an FDT is SDL (Specification and Description Language) which was developed by the Telecommunications Standardisation Sector of the International Telecommunications Union (ITU-T). SDL, which can be found in [14],[15] and [16], is based on an extended state machine model, supplemented by features for specifying Abstract Data Types. This combination is supported by complete formal semantics. SDL provides constructs for representing structures, behaviours, interfaces and communication links. In addition, it provides constructs for abstraction, module encapsulation and refinement. All these constructs were designed to assist the representation of a variety of telecommunications system specifications, including aspects of services and protocols. SDL is quite widely used in the telecommunication community and is well supported by a variety of tools [14].

Protocol specifications play a major part in the development of a protocol. Software tools that support FDTs ensure that the specifications are unambiguous, clear, concise, complete, consistent, tractable, and conformed to [14]. These software tools generally use the specifications to model the system, and then simulate its operation. The simulations are used for analysis, testing, verification, and comparisons against other protocols.

XMelba, which is jointly owned by Telstra and RMIT, is a CASE tool based on SDL, and specifically designed for communications systems. XMelba runs under the X-Windows UNIX system, and like PROSPEC [17], it provides a GUI which enables very easy entry and modification of SDL specifications. Another feature that XMelba has in common with PROSPEC, is the provision for abstraction and modular construction, which simplify complex designs.

The XMelba system includes:

1. XMelba; the graphical user interface for entry of formal specifications. The output is a XGR file,
2. XPR; converts an XGR file to SDL/PR, which is similar to conventional programming languages. Specifications may be entered directly into SDL/PR, although the GUI is much easier to use,
3. SDL to "C" Translator; converts XPR files to "C" code,
4. SDL Analyser; performs syntax and partial static semantics checks on the XPR specifications, and produces two documents: report and an error listing,
5. SDL Runtime Library; linked with the executable specifications compiled from the "C" code, to run the system in real time,
6. debugging utility; used to debug the executable specifications. It creates a log file, which provides feedback to XMelba.

References

[1] Bureau Of Meteorology, *Annual Report*, 1992-93

[2] Bureau Of Meteorology, *Report on Initial Point to Multi-Point Transmission Requirements For Operational Data To Regional Forecast Offices*, 1993.

[3] Optus Communications, *Omnicast Seminar Information Book*, 1993.

[4] Fairhurst G, *Handshaking by Satellite*, 16th Simula Users Conference, Innsburg Austria, 1988, pp. 51-59.

[5] Clark D D, Lambert L M, and Zhang L, *NETBLT: A Bulk Data Transfer Protocol*, Internet RFC 998, 1987.

[6] Kruse H, *Performance of Common Data Communications Protocols Over Long Delay Links; An Experimental Examination*, Third International Conference on Communication Systems, Nashville Tennessee, March 1995.

[7] Thomas R, Burst Mode Boosts Netware's Wide-Area Acumen, Data Communications International, September 1992.

[8] Komisarczuk P, Hadjitheodosiou M H, Coakley F, Jeans T and Smythe C, *End-To-End Protocol Performance in ATM/Satellite Interconnected LANs*, Second International Conference on Broadband Services, Systems and Networks, 3-4 November 1993, Brighton UK, pp. 87-91.

[9] Halsall F, *Data Communications, Computer Networks and Open Systems, 3rd Edition*, Addison Wesley, 1993.

[10] Huang C and McKinley P K, *Communication Issues in Parallel Computing Across ATM Networks*, IEEE Transaction on Software Engineering,, Winter 1994.

[11] Postel J, and Reynolds J, *File Transfer Protocol (FTP)*, Internet RFC 959, 1985.

[12] Postel J, *Transmission Control Protocol (TCP)*, Internet RFC 793, 1981.

[13] ISO 4335, *High Level Data Link Control (HDLC) Procedures.*

[14] Turner K J, *Using Formal Description Techniques; An Introduction to Estelle, Lotos and SDL*, Wiley, West Sussex, England, 1993.

[15] ITU-T Recommendation Z.120, *Criteria For The Use and Applicability of Formal Description Techniques; Message Sequence Chart (MSC), 1993.*

[16] ITU-T Recommendation Z.100 Appendices I and II, 1993, *Programming Languages; SDL Methodology Guidelines, SDL Bibliography*, 1993.

[17] Chow -H C and Lam S S, *PROSPEC: An Interactive Programming Environment for Designing and Verifying Communication Protocols*, IEEE Transaction on Software Engineering, Vol 14, No. 3, Brighton UK, March 1988.