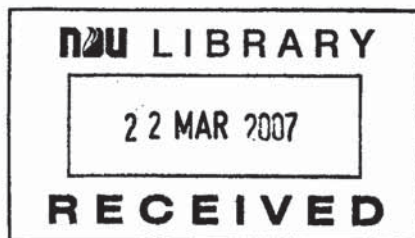


# **Performance Analysis of Video Transport in Two-Stage Hybrid Networks**

**By**  
**Maria Ghosn Chelala**

**A Thesis Submitted in Partial Fulfilment of the Requirements for the  
Degree of Master of Science  
in  
Computer Information Systems  
Department of Computer Science**



**Faculty of Natural and Applied Sciences  
Notre Dame University – Louaize  
Zouk Mosbeh, Lebanon  
Fall 2006**

Performance Analysis of Video Transport in Two-Stage Hybrid Networks

By

Maria Ghosn Chelala

Approved:



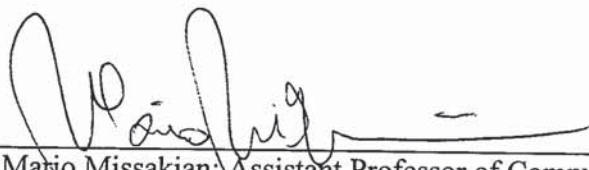
---

Hoda Maalouf: Assistant Professor of Computer Science  
Advisor.



---

Hikmat Farhat: Assistant Professor of Computer Science  
Member of Committee.



---

Mario Missakian: Assistant Professor of Computer Science  
Member of Committee.



---

Ramez Maalouf: Assistant Professor of Mathematics and Statistics  
Member of Committee.

---

Date of Thesis Defense: February 5<sup>th</sup>, 2007

---

## ACKNOWLEDGEMENTS

I would like to express my deepest appreciation to my advisor Dr. Hoda Maalouf for her tireless support and invaluable advice while working on this thesis. I would also like to extend this appreciation to all NDU instructors for their bestowal of unstinting knowledge.

My sincere thanks go to my parents for their endless guidance and encouragement in all my endeavours.

Last but not least I would like to say how indebted I am to my fiancé who has always been there for me.

---

## ABSTRACT

With the current rising trend in wireless based systems, applications such as video on demand, remote monitoring and surveillance are now being used in greater abundance. Coupled with this growth comes the ongoing need for improved network performance. This is due to the resource hungry nature of video data which places stringent requirements on communication networks.

This study investigates performance improvement for remote surveillance applications over hybrid 2-stage disordering networks. It specifically focuses on resequencing alternatives for video data.

A resequencing and batch aggregation procedure has been proposed in this thesis to improve performance in 2-stage disordering hybrid networks. The main aim is to decrease delay and improve received video quality by forming ordered batches at a specialized gateway called a gateway controller. Requirements and characteristics of video data are considered in the implementation of the gateway controller.

The trials performed show that with the integration of the gateway controller we can improve performance and ameliorate the video quality for the end user.

---

**TABLE OF CONTENTS**

<b>Acknowledgements</b> .....	I
<b>Abstract</b> .....	II
<b>List of Figures</b> .....	VI
<b>List of Tables</b> .....	VII
<b>CHAPTER 1 INTRODUCTION AND PROBLEM DEFINITION</b> .....	1
<b>1.1 Introduction</b> .....	1
1.1.1 Problem Overview.....	1
1.1.2 Summary.....	2
<b>1.2 Problem Definition</b> .....	2
1.2.1 Introduction .....	2
1.2.2 Basic Scenario Details.....	3
1.2.3 Summary.....	4
<b>1.3 Thesis Organisation</b> .....	5
<b>CHAPTER 2 BACKGROUND AND MOTIVATION</b> .....	7
<b>2.1 Introduction</b> .....	7
<b>2.2 Real-Time Video Transport over Wireless Networks</b> .....	8
2.2.1 Introduction.....	8
2.2.2 TCP and Wireless Real-Time Applications.....	8
2.2.2.1 Introduction.....	8
2.2.2.2 Why is TCP Unsuitable?.....	9
2.2.2.3 Summary.....	10
2.2.3 UDP, UDP Lite and Wireless Real-Time Applications.....	10
2.2.3.1 Introduction .....	10
2.2.3.2 UDP Specifics.....	11
2.2.3.3 Summary.....	12
2.2.4 SCTP and Video Streaming.....	12
2.2.4.1 Introduction.....	12
2.2.4.2 SCTP Specifics.....	12
2.2.4.3 Summary.....	14
2.2.5 RTP, RTCP, RTSP.....	14
2.2.5.1 Introduction .....	14
2.2.5.2 RTP Specifics.....	14
2.2.5.3 Summary .....	16

---

2.2.6 Recovering from Errors	16
2.2.6.1 FEC and ARQ.....	16
2.2.6.2 PPP.....	16
2.2.6.3 Summary.....	17
<b>2.3 Mpeg-4 Video.....</b>	<b>17</b>
2.3.1 Introduction.....	17
2.3.2 Video On-Demand vs. Download and Play.....	17
2.3.3 MPEG-4 Streaming.....	17
2.3.4 Video Compression in MPEG.....	18
2.3.5 Summary.....	20
2.3.6 Scalable Video Coding.....	20
2.3.7 Studying Network Performance for Video Stream Data .....	22
<b>2.4 Disordering Communication Networks.....</b>	<b>23</b>
2.4.1 Introduction .....	23
2.4.2 Packet-Switched Network Delay.....	24
<b>2.5 The Resequencing Problem.....</b>	<b>25</b>
2.5.1 Introduction.....	25
2.5.2 Resequencing Methods.....	25
2.5.3 Resequencing Delay.....	26
2.5.4 Resequencing in Multistage Disordering Systems.....	27
2.5.5 Star Customers and Batch Departure.....	27
<b>2.6 Research Motivation.....</b>	<b>28</b>
2.6.1 Introduction.....	28
2.6.2 Necessity of Partial Reordering.....	29
2.6.3 Two-Phase Resequencing Using Gateway Controllers.....	29
2.6.4 Batch Processing and Aggregation.....	29
<b>2.7 Summary.....</b>	<b>31</b>
<b>CHAPTER 3 SOFTWARE TOOLS, SIMULATION TEST-BED AND PERFORMANCE EVALUATION METRICS.....</b>	<b>32</b>
<b>3.1 Software Tools – OMNeT++.....</b>	<b>32</b>
3.1.1 Introduction.....	32
3.1.2 OMNeT++ Design.....	32
<b>3.2 Simulation Test-Bed.....</b>	<b>34</b>
3.2.1 Introduction.....	34
3.2.2 Assumptions and Network Characteristics.....	34
<b>3.3 Performance Evaluation Metrics for Networking and Video.....</b>	<b>37</b>
3.3.1 Introduction.....	37
3.3.2 Starvation Probability and Information Loss Probability.....	37
3.3.3 Video Playback Quality Assessment.....	38

---

---

<b>3.4 Summary</b> .....	40
<b>CHAPTER 4 PERFORMANCE EVALUATION AND ANALYSIS</b> .....	41
<b>4.1 Introduction</b> .....	41
<b>4.2 Methods and Results</b> .....	42
4.2.1 Delay.....	42
4.2.1.1 Evaluation Method.....	42
4.2.1.2 Results.....	42
4.2.2 Batch Size vs. Maximum Waiting time.....	43
4.2.2.1 Evaluation Method.....	43
4.2.2.2 Results.....	43
4.2.3 Overhead/Payload Ratio vs. Maximum Waiting Time.....	44
4.2.3.1 Evaluation Method.....	44
4.2.3.2 Results.....	45
4.2.4 Starvation Probability vs. Pre Buffer Time.....	46
4.2.4.1 Evaluation Method.....	46
4.2.4.2 Results.....	47
4.2.5 I-Frame Priority.....	48
4.2.5.1 Evaluation Method.....	48
4.2.5.2 Results.....	48
<b>4.3 Summary</b> .....	51
<b>CHAPTER 5 MAIN CONTRIBUTIONS &amp; FUTURE WORK</b> .....	53
<b>5.1 Introduction</b> .....	53
<b>5.2 Main Contributions</b> .....	53
<b>5.3 Future Work</b> .....	54
<b>Appendix A1</b> .....	55
<b>Appendix A2</b> .....	58
<b>Bibliography</b> .....	60

---

## LIST OF FIGURES

Figure 1.1	Basic Scenario.....	3
Figure 1.2	Packet Disorderng.....	3
Figure 1.3a	1-Phase Resequencing.....	4
Figure 1-3b	2-Phase Resequencing.....	5
Figure 1.3c	2-Phase Resequencing with Aggregation.....	5
Figure 2.1	TCP Header Fields.....	10
Figure 2.2	UDP Header.....	11
Figure 2.3	TCP Connection vs. SCTP Association.....	13
Figure 2.4	Relationship of an SCTP Association to Streams.....	13
Figure 2.5	RTP Packet Header Fields.....	15
Figure 2.6	RTP as a Sub-Layer of the Transport Layer.....	15
Figure 2.7	I, P and B Frames Make Up a Group of Video Object Planes.....	19
Figure 2.8	Temporal Scalable Encoding.....	21
Figure 2.9	Spatial Video Coding.....	21
Figure 2.10	Video Trace File.....	23
Figure 2.11a	1-Phase Resequencing.....	30
Figure 2.11b	2-Phase Resequencing.....	30
Figure 2.11c	2-Phase Resequencing with Batch Processing and Aggregation...	30
Figure 3.1	Simple and Compound Modules.....	33
Figure 3.2	Connections.....	33
Figure 3.3	Connections Example.....	34
Figure 3.5	Simulation Network.....	35
Figure 3.6	Trace File with PSNR Values.....	39
Figure 4.1	Video Frame Size vs. Delay for 3 Gateway Modes.....	42
Figure 4.2	Batch Size in bytes vs. Maximum Waiting Time ( $W_{tMax}$ ).....	43
Figure 4.3	3 Batch Size in Fragments vs. Maximum Waiting Time ( $W_{tmax}$ )...	44
Figure 4.4	Overhead/Payload Ratio vs. Maximum Waiting Time ( $W_{tmax}$ ).....	45
Figure 4.5	Overhead/Payload Ratio vs. Fragments per Batch.....	46
Figure 4.6	Frame Loss Probability vs. Fragments per Batch.....	47
Figure 4.7	Information Loss Probability vs. Fragments per Batch.....	47
Figure 4.8	Information Loss Probability vs. I-Frame Maximum Waiting Time.....	48
Figure 4.9	Frame Loss Probability vs. I-Frame Maximum Waiting Time.....	49
Figure 4.10	Overhead Payload Ratio vs. I-Frame Maximum Waiting Time.....	49
Figure 4.11	Fragments per Batch vs. I-Frame Maximum Waiting Time.....	50
Figure 4.12	Delay vs. Video Frame Size with I-Frame Priority.....	50



---

**LIST OF TABLES**

Table 2.1      Star Customers and Batch Departure..... 27

# Chapter 1

## Introduction and Problem Definition

### 1.1 Introduction

#### 1.1.1 Problem Overview

With the growth of on-demand video and other video networking applications such as surveillance, more stringent requirements have been placed on our networks. This is due to the fact that the transfer of such real-time multimedia data has stringent bandwidth, delay, loss and ordering constraints. Within this framework several points of interest arise.

1) When it comes to video data must be played back in the correct order, therefore it is important that packets are ordered upon being played back by the receiver. Since it is not guaranteed that the order in which the packets are received is the same as that in which they were sent, for instance in multi-path routing or adaptive routing (load balancing) to handle congestion problems [7], a reordering or resequencing of these packets is needed.

2) Generally it is known that wireless networks such as cellular or wireless sensor networks have a higher BER (bit error rate) [13]. Due to this, smaller packet sizes are often adhered to for effectiveness when transmitting over such networks. In addition, the actual MTU (maximum transfer unit) of the wireless network may be smaller than that of a fixed network. Therefore, when packets are transported from a wireless to a fixed network, it is possible that the packet size being employed will be smaller than what the fixed network effectively allows. This implies that the number of packets propagating the fixed network is greater than necessary. Due to the larger number of

---

packets, the traffic on the fixed network is increased which in turn causes more overheads, higher bandwidth consumption and more possibility for packet disordering

3) In addition to the above points, if a younger packet (packet with a greater timestamp) arrives before an older packet (packet with a smaller timestamp), the younger packet must wait until the older packet arrives so that correct resequencing (reordering) can be achieved. Therefore, if an older packet is, for example, lost and retransmitted or sent via a route encountering greater delays, the resequencing delay is substantially increased. Such older packets are referred to as "star customers" because they make other customers –or packets – wait [9]. In order to avoid being bogged down by star customers, it is possible to use partial order resequencing as will be explained later.

The application assumed in this study is a remote video-surveillance system. The network is a two-stage hybrid disordering network. The access network is a wireless sensor network (WSN) with four channels and the fixed network has four channels of type ISDN H.

### **1.1.2 Summary**

The objective of this thesis is to improve performance and resource usage in the case of multi-path video transport over two-stage hybrid disordering networks by addressing different resequencing possibilities and the related issues that arise such as video properties, the problem of star customers, loss of data and different network MTUs. The application assumed is a remote video-surveillance system.

## **1.2 Problem Definition**

### **1.2.1 Introduction**

As previously stated, the work in this thesis is based upon a scenario in which real-time video data is being sent via a wireless access network after which it must propagate a fixed network in order to reach the receiver (see fig 1.1 on the next page). The wireless access network is a WSN and the fixed network is ISDN.

---

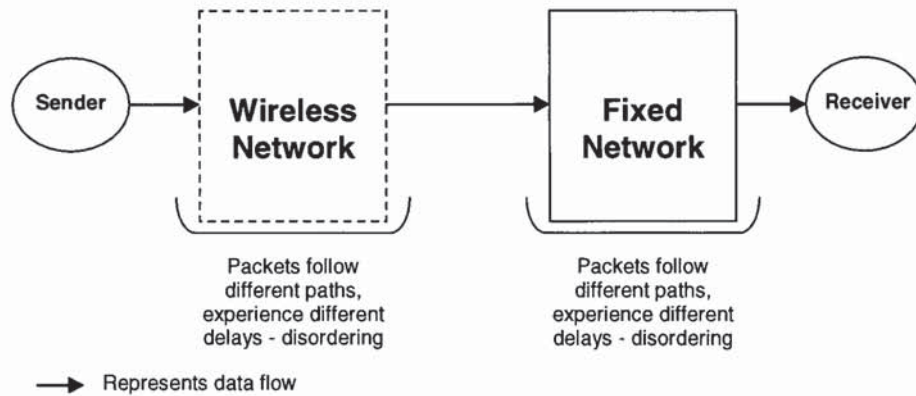


Figure 1.1 – Basic Scenario

### 1.2.2 Basic Scenario Details

The sender has a wireless connection and the data leaves the sender in order. However, by the time the data has propagated the wireless network, it is likely that the order has been lost. Therefore it will enter into the fixed network unordered where even more disordering is likely to occur (fig 1.2).

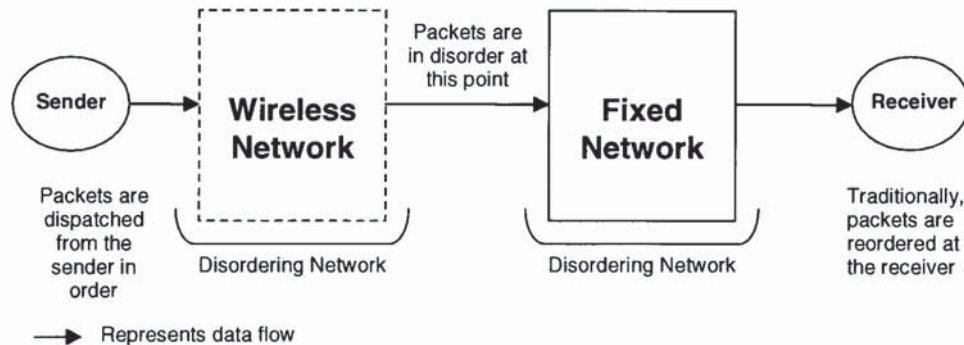


Figure 1.2 – Packet Disordering

When the disordered data reaches the receiver, it must be reordered in the resequencing buffer. Two kinds of resequencing are possible:

- Total Order Resequencing – in this kind of resequencing, a younger packet must wait for all outstanding older customers before leaving the resequencing buffer. In this way, the original order is restored and no packets are dropped but the delay may be large causing cuts in video playback.
- Partial Order Resequencing - in this kind of resequencing, a younger packet must wait only for a fixed time interval before leaving the resequencing

buffer. Once the younger packet has left the resequencing buffer, any older customers arriving thereafter are discarded. This implies that some packets may be dropped which could degrade playback quality.

From the above explanation, it is obvious that resequencing causes a delay. For online video and other real-time applications, the total order resequencing method can cause a large delay and therefore degrade the quality of the playback. Furthermore, if partial order resequencing is to be used, it would not be optimal for all frames to be treated equally because as far as video is concerned, certain frames are more important than others [2]. The importance of these frames is relevant to how long they should be made to wait for other frames and whether they can be dropped without significantly degrading the playback quality, since dropping frames implies loss. Overall, the advantage of using partial order resequencing is that large delays due to the resequencing procedure are avoided.

### 1.2.3 Summary

As a whole, the problem is finding the right balance between the factors which cause delay and congestion as well as those which affect the overall quality of the playback. The main question to be addressed in this thesis is whether the resequencing process should be left until the end (1-phase or end-to-end resequencing – fig 1.3a) or whether it is advantageous for packets to be resequenced and possibly reassembled (aggregated) into larger packets before entering the fixed network (2-phase or hop-by-hop resequencing - fig 1.3b and 1.3c). The issue of packet loss and packet priorities is relevant to this problem as well.

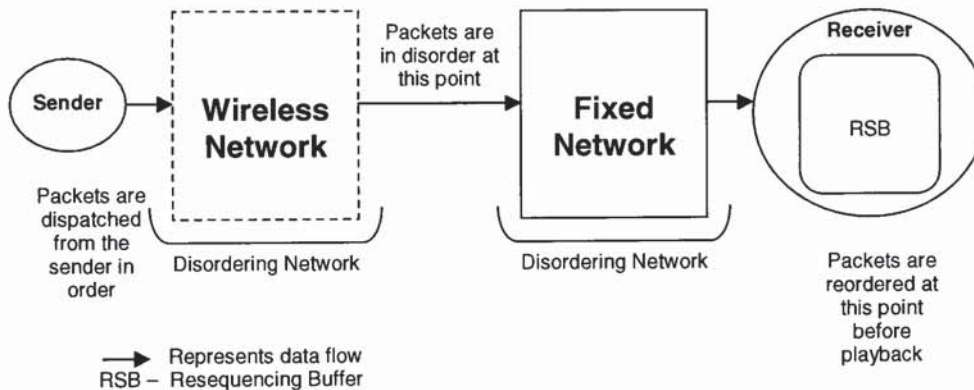


Figure 1.3a – 1-Phase Resequencing

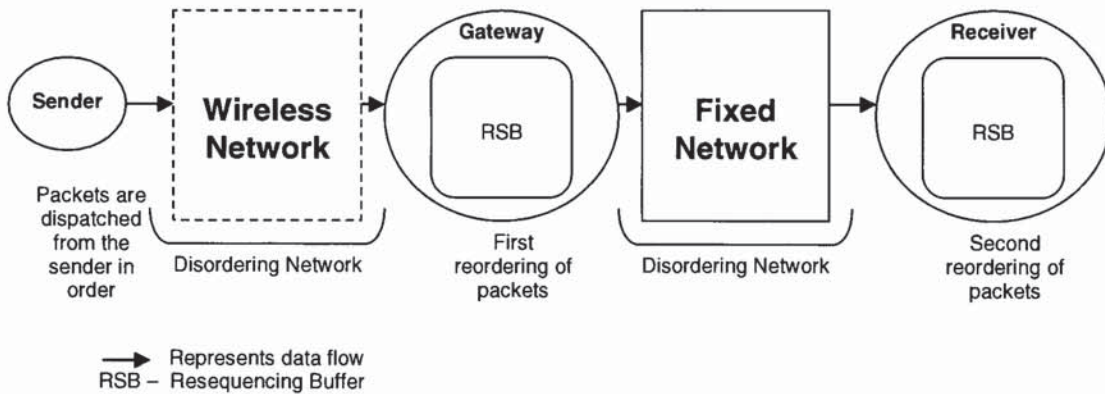


Figure 1.3b – 2-Phase Resequencing

It should be mentioned that the 2-phase resequencing to be studied in this thesis is where the differences in packet sizes between wired and wireless networks are to be addressed. Since wireless networks often have a smaller MTU than wired networks, frames could be reordered and thus aggregated into larger messages before entering the wired network to exploit the larger (by a factor  $\geq 2$ ) MTU of the fixed network. This is shown in the figure 1.3.

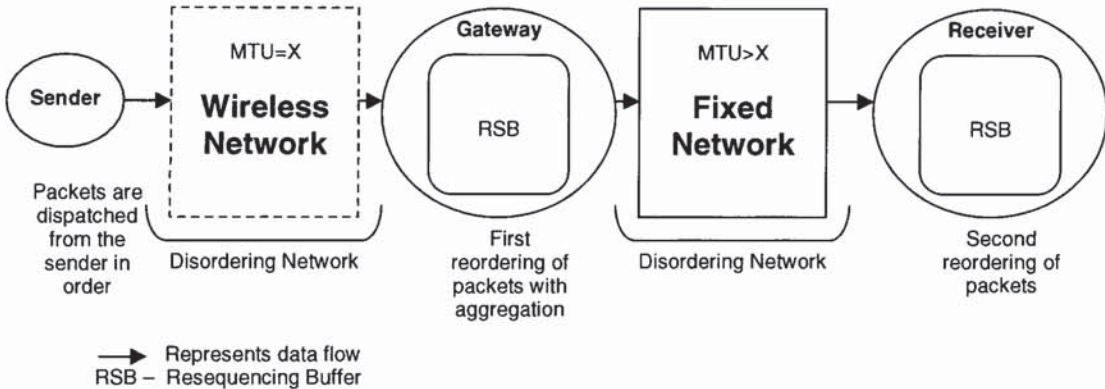


Figure 1.3c – 2-Phase Resequencing with Aggregation

### 1.3 Thesis Organisation

This thesis is organised into five chapters. Chapter 2 gives background information on multimedia networking, resequencing and previous related work. Chapter 3 introduces the software tools and evaluation metrics used in the implementation part. It also gives an overview of the network assumptions and characteristics. Chapter 4

presents the evaluation methods in detail along with the simulation results. Finally chapter 5 summarizes conclusions and possible future work.

## Chapter 2

### Background and Motivation

#### 2.1 Introduction

This thesis addresses the problem of real-time video transport for surveillance purposes in hybrid two-stage disordering networks. The scenario assumed involves a wireless disordering network connected to a fixed disordering network via a gateway. Since video files must be played back in sequential order, a reordering or resequencing procedure is necessary. The main aim is to assess the possible impact of applying different resequencing strategies to improve performance, improve playback quality and decrease delay. To analyse results, some network performance metrics as well as the quality of the received video stream will be evaluated.

This chapter starts by discussing general points relevant to the transport of real-time video over wireless networks and different networking protocols. It then moves specifically to video characteristics, the resequencing problem and delay analysis.

---



## 2.2 Real-Time Video Transport over Wireless Networks

### 2.2.1 Introduction

The transport of good quality real-time video is subject to the performance of the carrier network. In comparison to wire-line links, wireless links are not so reliable. They usually suffer high bit error rates (BER) as well as lower data rates [13]. Furthermore, protocol performance across the layers (in both wired and wireless networks) is an impacting factor. TCP, for one, is not suitable for real-time multimedia data as will be discussed later. In addition, where video and multi-path transport is concerned, a resequencing process is necessary to enable playback in the correct sequence [7]. This is because multi-path routing creates disordering amongst packets. Different resequencing possibilities and related issues will be investigated in this thesis.

Enabling real-time video over wireless networks means adapting hardware, software, processes and protocols to suit the nature of real-time multimedia data and improve the perceived quality by the user. In turn, this requires video codecs to be able to recover efficiently from errors. Most video codecs have become quite error resilient. Consequently, these codecs can deal with a damaged or missing frame but often prefer a damaged frame to a missing one [6]. Bearing in mind this information and the time-sensitive nature of real-time video applications such as video on demand, alternatives and modifications to common protocols have been investigated and/or implemented for this nature of data. Nevertheless, the quest for improved performance is ongoing.

A brief discussion of TCP and other protocols follows after which an overview of MPEG-4 is given. The resequencing problem is then explained in some detail.

### 2.2.2 TCP and Wireless Real-Time Applications

#### 2.2.2.1 Introduction

As previously stated, wireless networks have higher *bit error rates* (BERs) than wired networks. Mobile wireless systems are also subject to signal fade (attenuation), base-station handover, link outages, and variable levels of load. TCP was initially designed under the assumption of wired medium [14] which makes it less suited to wireless and real-time applications

---

### 2.2.2.2 Why is TCP Unsuitable?

Since TCP was designed under the assumption of a wired medium, TCP reacts to packet loss as being due to congestion in the network and not packet corruption. This assumption generally holds in wired networks; however it is not necessarily the case in wireless networks which have variable BERs. TCP also makes the assumption that the round trip time ( $RTT$ ) is generally stable, because TCP uses a formula which tends to minimise the differences between  $RTT$  estimates [14]. Jacobson's algorithm is used to obtain the  $RTT$  estimate. It is the following:

$$RTT = \alpha RTT + (1 - \alpha)M \quad (\text{Eq. 2.1})$$

$2 \times RTT$  was originally used as the timeout period ( $TO$ ). However this was later changed to account for possible variation in the actual roundtrip time. Therefore  $TO$  is set to:

$$TO = RTT + 4D \text{ where } D = \alpha D + (1 - \alpha)|RTT - M| \quad (\text{Eq. 2.2})$$

*Note:  $\alpha$  is a smoothing parameter and is thus responsible for minimising differences between  $RTT$  estimates.*

Since TCP assumes that the cause of packet loss is network congestion the sender will enter "Slow Start" mode. "Slow Start" mode causes significant delays within the data transfer, because no sending occurs during the timeout interval and when sending is resumed, it starts with a single packet exchange, gradually recovering the data rate that was active prior to the packet loss. Furthermore, due to its ordered delivery constraint, when packet loss occurs, TCP creates head of line blocking which means that all packets are delayed until the missing packet is retransmitted and received.

As far as overheads go, TCP assumes that an overhead of a minimum of 40 bytes per TCP packet (20 bytes of IP header and 20 bytes of TCP header) is an acceptable overhead when compared to the available bandwidth and the average payload size (see figure 2.1). When applied to low-bandwidth links or links with small MTUs, this is no longer the case, and the protocol overheads may make the resultant communications system too inefficient to be useful [14]. For instance, in wireless unreliable links that may experience high loss and BERs, it is often preferable to keep the frame size small to avoid

having to retransmit large frames. For example, in WSN the frame size is 120 bytes. Having a header of 40 bytes means one third of each fragment is overhead.

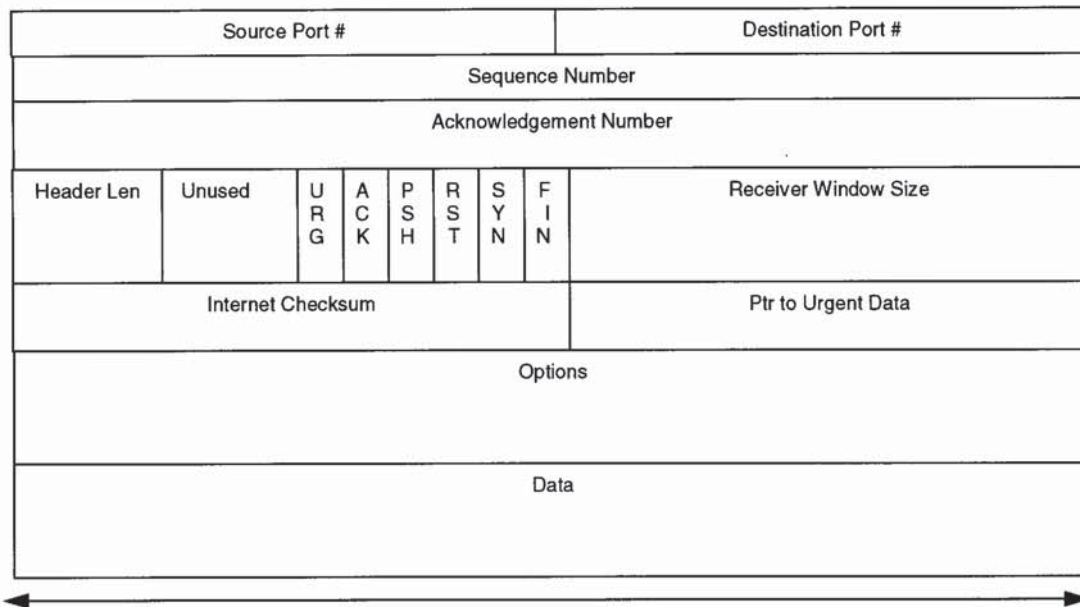


Figure 2.1 – TCP Header Fields

### 2.2.2.3 Summary

In summary, the mechanisms that TCP implements to achieve reliable and in-order delivery can result in delays unfavourable to real-time applications since such applications can handle some data loss but are very time-constrained. Further to this, large overheads consume bandwidth which is often scarce in wireless links. For this reason UDP is the preferred choice for real-time applications.

## 2.2.3 UDP, UDP Lite and Wireless Real-Time Applications

### 2.2.3.1 Introduction

The big breakthrough that enabled the streaming revolution was the adoption of the User Datagram Protocol (UDP). UDP made streaming media feasible by transmitting data more efficiently than previous protocols from the host server over the Internet to the receiver.



### 2.2.3.3 Summary

UDP is lightweight, connectionless, prioritizes timeliness and produces less overhead. It is therefore the preferred protocol for real-time applications. The suggested adaptation called UDP Lite can also be implemented making the protocol more flexible.

## 2.2.4 SCTP and Video Streaming

### 2.2.4.1 Introduction

The Stream Control Protocol (SCTP) is a transport protocol considered as an option for streaming media content [2]. It was developed with the goal of overcoming some of the performance limitations of TCP resulting from its mechanisms for reliable data delivery. SCTP is a reliable transport protocol that provides reliable, ordered/unordered delivery of data between two endpoints (much like TCP) but operates in a message-oriented way and preserves data message boundaries (like UDP). However, unlike TCP and UDP, SCTP adds advantages like multi-homing and multi-streaming capabilities, both of which increase availability.

### 2.2.4.2 SCTP Specifics

*Multi-homing* provides applications with higher availability than those that use TCP. A multi-homed host is one that has more than one network interface and therefore more than one IP address for which it can be addressed. In TCP, a *connection* refers to a channel between two endpoints (in this case, a socket between the interfaces of two hosts). SCTP introduces the concept of an *association* that exists between two hosts but can potentially collaborate with multiple interfaces at each host. Figure 2.3 illustrates the difference between a TCP connection and an SCTP association.

---

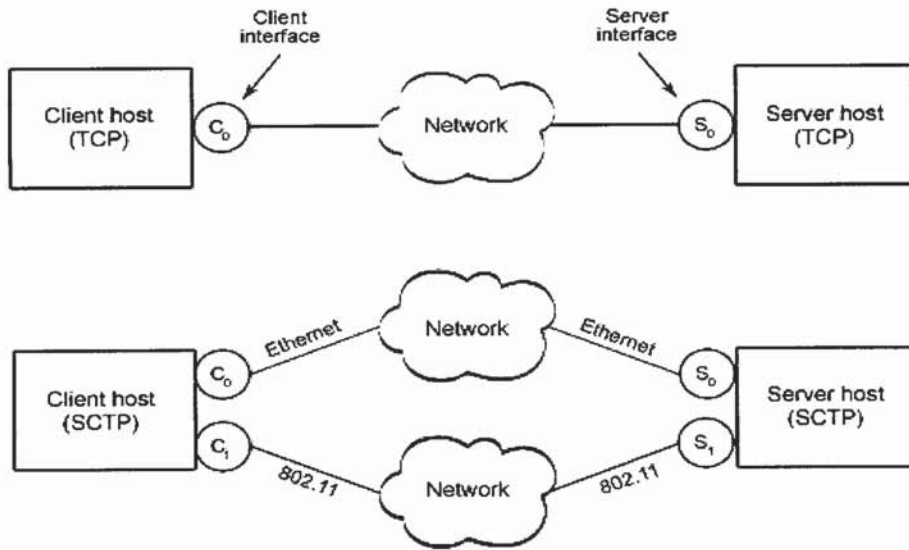


Figure 2.3 – TCP Connection vs. SCTP Association

Upon detecting a path failure, the protocol sends traffic over the alternate path. It is not even necessary for the applications to know that a failover recovery occurred.

This is a powerful mechanism for providing high availability and increased reliability.

SCTP supports multi streaming which allows multiple streams within an association. All the streams within an association are independent but related to the association (see Figure 2.4).

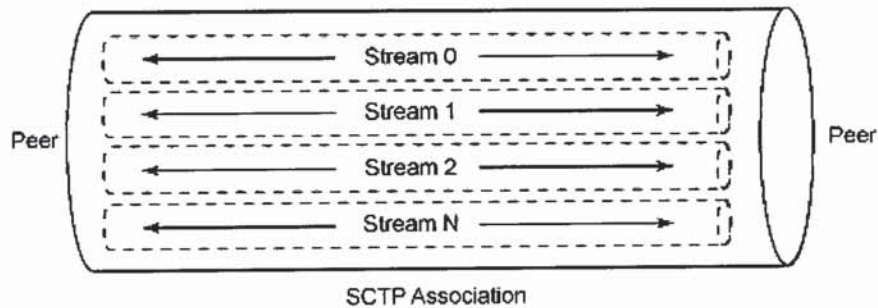


Figure 2.4 - Relationship of an SCTP Association to Streams

Each stream is given a stream number that is encoded inside SCTP packets flowing through the association. Multi-streaming is important because a blocked stream (for

example, one awaiting re-transmission resulting from the loss of a packet) does not affect the other streams in an association as has been seen with TCP (*head-of-line blocking*).

The reliability and delivery order constraints in SCTP can be set for each channel. Reliability is set by choosing a level. This level designates the number of retransmissions to be attempted in case of a lost packet [2]. For example, if the reliability level is set to 1, 1 retransmission will be attempted for each packet that is lost. In order to avoid retransmission and consequently decrease delay, the reliability level can be set to 0. For delivery order in SCTP, the unordered delivery flag can be set to allow packets to be passed on in the order that they arrive. Head of line blocking can be avoided in this way.

*Note: Packets in SCTP are divided into chunks, each of which may belong to a different logical stream within the association. Each chunk must be smaller than the link MTU. Several chunks can be multiplexed into one packet if the need arises and debundling is performed by the SCTP receiver.*

#### **2.2.4.3 Summary**

Overall, the essential alteration with SCTP is the use of many transport modes. This makes it flexible and adaptable to packet priorities and allows better availability. However SCTP has similar congestion control mechanisms to TCP which can cause similar delays.

### **2.2.5 RTP, RTCP, RTSP**

#### **2.2.5.1 Introduction**

The Real-time Transport Protocol (RTP) provides end-to-end delivery services for real-time data. It can also be integrated into the application layer. RTP and the Real-Time Control Protocol (RTCP) complement each other. RTCP messages are transmitted by participants in multicast sessions to monitor performance [5].

#### **2.2.5.2 RTP Specifics**

Real Time Protocol (RTP) provides services such as sequence numbers and time-stamping to multimedia applications. RTP packet header fields and their sizes are shown

---

in figure 2.5.

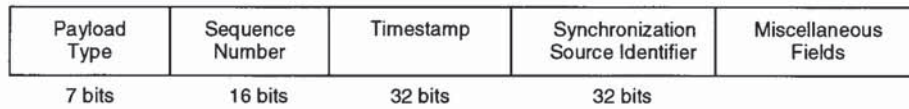


Figure 2.5 – RTP Packet Header Fields

RTP packets encapsulate chunks of video data. RTP enables source identification, QOS feedback and plays an important role in synchronising audio and video. It typically runs on top of UDP as a “sublayer” of the transport layer (see figure 2.6).

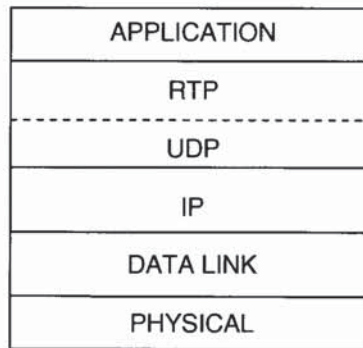


Figure 2.6 – RTP as a Sub-Layer of the Transport Layer

Real Time Control Protocol (RTCP) provides control information and statistics for an RTP flow. It does not actually transport any video data. RTCP is used periodically to transmit control packets to participants in a streaming multimedia session. The primary function of RTCP is to provide feedback on the quality of service being provided by RTP such as statistics on a media connection and information such as bytes sent, packets sent, lost packets, jitter, feedback and round trip delay. An application may use this information to increase the quality of service perhaps by limiting flow, or maybe using a low compression codec instead of a high compression codec. There are several types of RTCP packets: Sender Report Packet, Receiver Report Packet, Source Description RTCP Packet, Goodbye RTCP Packet and application specific RTCP packets.

Real Time Streaming Protocol (RTSP) allows a client to remotely control a streaming media server, issuing VCR-like commands such as "play" and "pause", and allowing time-based access to files on a server. Some RTSP servers use RTP as the



transport protocol for the actual audio/video data but it is not a requirement. RTSP messages can be sent over TCP or UDP [5].

### **2.2.5.3 Summary**

The RTP protocol suite is adapted especially for time sensitive data thus making it the protocol suite of choice for multimedia applications.

## **2.2.6 Recovering from Errors**

### **2.2.6.1 FEC and ARQ**

Different solutions to the high BER of wireless links are available. Forward error correction (FEC) codes and automatic retransmission systems (ARQ) can be used on the wireless link. FEC introduces a relatively constant coding delay and a bandwidth overhead into the path by adding redundant data to help correct damaged frames. However it cannot correct all forms of bit error corruption [5]. ARQ relies on retransmission and deals with retransmission in a similar way to TCP causing a kind of "head of line blocking". Due to this each packet may then experience additional delay while retransmissions are attempted to correct errors, because the packet flow may halt for an entire link round-trip time (*RTT*) interval. Therefore such methods are not necessarily efficient in time-sensitive scenarios.

### **2.2.6.2 PPP**

Point to point protocol (PPP) is a link layer protocol for point-to-point links. It basically encapsulates datagrams and detects errors using a polynomial frame check sequence (FCS). If the checksum shows an error in the frame, the frame is dropped at the receiver. However video codecs are good at recovering from errors which means it would be better for erroneous frames not to be dropped. PPP Lite [13] enables PPP checksums to be ignored at the receiver thus allowing erroneous frames to reach the upper layers instead of being dropped directly. This way the codec can decide whether to drop the packet or not.

---

### 2.2.6.3 Summary

For reliable data transfer link level error checking can be implemented. However for video applications, packets must reach the codec even if they have errors. This allows the codec to attempt partial decoding.

Within the scope of this thesis it is neither feasible nor relevant to discuss all the different protocols available. The above were discussed as examples of the prevalent protocols to show how the requirements for real-time multimedia data differ from that of static or pre-downloaded data and how these protocols have been developed to help better support applications such as video on demand.

## 2.3 Mpeg-4 Video

### 2.3.1 Introduction

MPEG-4 is a file specification for digital media. It is a codec and file type geared towards use over media with low bandwidth such as mobile networks. It is also used over DSL and broadband. MPEG-4 can be read on different players such as RealPlayer and Quicktime [4].

### 2.3.2 Video On-demand vs. Download and Play

As opposed to download and then play, on-demand video means that clients can view content while the file is downloading. In this scenario clients do not have to wait for the whole file to download in order to view it. The only waiting involved is at the beginning for some of the content to prebuffer in order to allow the user to keep watching even if the stream encounters slight delays while traversing the network. This is all made possible by streaming technology [5].

### 2.3.3 MPEG-4 Streaming

MPEG-4 files can be linked to just like other files using streaming via a Web Server over HTTP/TCP. However it is preferable to employ a server specifically for streaming media such as RealMedia or Apple Quicktime Streaming Server in order to

---

provide on-demand audio or video. This is because unlike in web server streaming, the stream being sent to the client is managed in such a way that it delivers the content at the exact data rate associated with the compressed audio and video streams. The server and the client communicate constantly during the delivery process, and the streaming media server can respond to any feedback from the client and adapt the quality of the video being sent depending on network restrictions. While streaming media servers can use the HTTP/TCP protocols used by Web servers, they can also use specialized protocols such as UDP to greatly improve the streaming experience. As mentioned earlier on, unlike TCP, UDP is a fast, lightweight protocol without any re-transmission or data-rate management functionality. This makes UDP an ideal protocol for transmitting real-time audio and video data, because of the time-sensitivity of such real-time applications. Protocols such as RTP are usually used on top of UDP to control and manage video transfer. As a bonus, because of the back-off policies implicit in the TCP protocol, UDP traffic gets higher priority than the TCP traffic on the Internet. Furthermore, instead of the blind retransmission scheme employed by TCP, streaming media servers such as Microsoft's Windows Media Services use an intelligent retransmission scheme on top of UDP. Windows Media Services' UDP Resend feature ensures that the server only retransmits lost packets that can be sent to the client in time to get played. Streaming servers also allow for multicast transmission. Multicast enables hundreds or thousands of users to play a single stream, but will only work on networks with Multicast-enabled routers. Multicast is becoming prevalent on corporate networks, but is still rare on the Internet.

#### **2.3.4 Video Compression in MPEG**

MPEG video compression draws its strength from its ability to detect and efficiently code motion in consecutive frames. It uses the Group of Object Video Planes (GOV) concept in which different frames are predicted from each other. All related frames form a GOV [2]. Most codecs used today adopt a similar approach.

In MPEG, intra-coded frames (I-frames) are key frames which are coded on a standalone basis without the need for data from other frames. Difference frames, known as predicted frames and bi-directional frames (P-frames and B-frames respectively), are

---

then coded using only the difference between themselves and other frames within the GOV. P-frames are coded based only on the difference between them and the previous I- or P-frame. B-frames, on the other hand, are coded according to differences between both previous and latter frames. The benefit of this forward prediction lies in the fact that objects can become revealed in subsequent frames and would have no reference if only backward prediction were applied.

In the figure below an MPEG-4 GOV is shown. The arrows show how the frames are related. For instance, it can be seen that the first two P-frames depend directly on the I-frame belonging to this GOV.

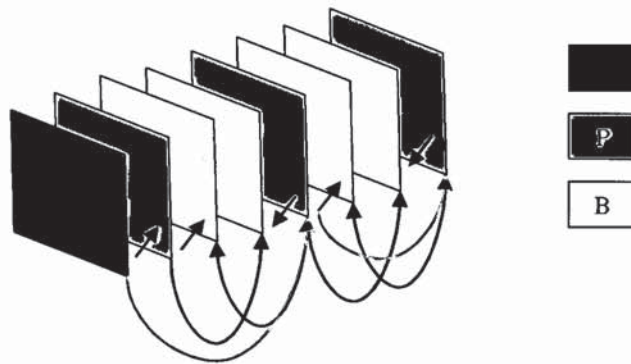


Figure 2.7 - I, B and P frames make up a Group of Video Object Planes (GOV) in the MPEG encoding scheme.

*Note: Every GOV has one I-frame only but may have several P and B-frames*

Key frame (I-frame) placement is critical to delivering high quality because key frames allow the codec to reset from errors. I-frames are inserted at scene changes when subsequent frame differences are large. They are also used for random access, since the decoder can easily display them without relying on information from other frames.

Bearing in mind the GOV concept, it is evident that errors in the transmission of I-frame data will cause degradation of quality not only for that I-frame but also in all the P-frames and B-frames that are predicted from the erroneous I-frame (propagation of errors). If the error rate on the I-frame data is high enough to cause the decoder to drop the I-frame, the playback quality of the video stream will be affected severely [2]. Trials were implemented in [2] which show that retransmitting lost I-frame data a second time

is highly beneficial for quality assuming that some of the retransmitted I-frames have time to reach the decoder before its display time. As mentioned earlier, streaming media servers such as Microsoft's Windows Media Services have a retransmission scheme which ensures that the server only retransmits lost packets that can be sent to the client in time to get played. This conserves bandwidth and improves quality as opposed to no resend.

### **2.3.5 Summary**

The MPEG-4 codec as described above outlines the complexity of video data. Overall, the nature of video data and link limitations has led to advances in codec error resilience in addition to protocol adaptation.

### **2.3.6 Scalable Video Coding**

In order to accommodate differences in bandwidth of different clients, MPEG-4 files can be stored at a variety of qualities. The file with the most basic quality can be streamed to clients with low bandwidth links and those with higher bandwidth can make use of full quality versions. The problem with this system is that it takes up a lot of disk space on the server since multiple copies of the same file at different qualities are needed.

In order to overcome the need for multiple files, scalable coded files can be used. Scalable coded files are able to deliver a mixture of qualities depending on the available connection in what is called an adaptive control scheme. This means that the stream quality being sent is adapted to adjust to changes in network resources. As shown in figure 2.8 this technique divides the video file into layers, possibly two or three layers. The first layer is called the base layer and must be delivered to achieve basic quality. Depending on the available resources, a second or third layer can be delivered along with the base layer to improve the quality of the playback. Such layers are called enhancement layers. The enhancement layer is dependent on lower layers. Therefore, an enhancement layer cannot be decoded without the layers beneath it being present. However the base layer can be decoded independently of any enhancement layers. Ultimately this type of encoding is known as temporal scalable encoding. Since enhancement layers may be dropped, fewer frames will be delivered. Therefore the frame rate is reduced for clients with inadequate resources [11].

---

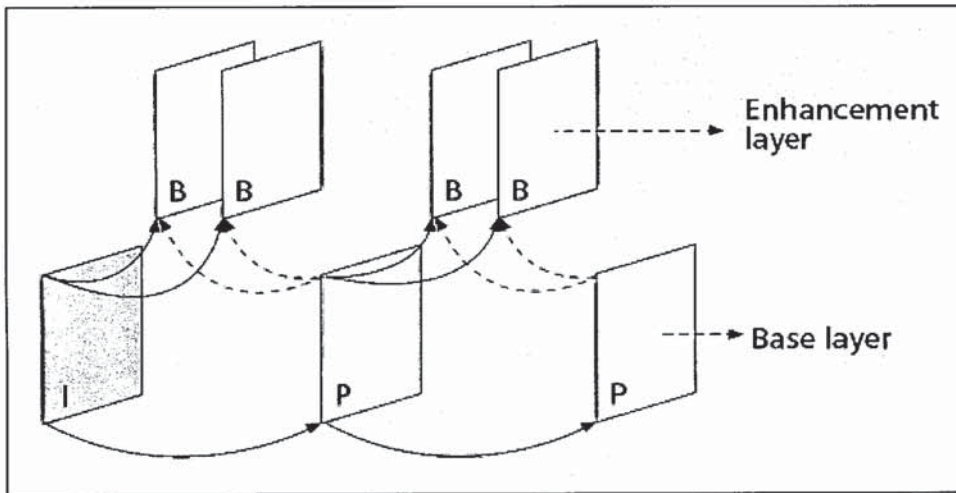


Figure 2.8 - Temporal scalable encoding; the base layer consists of I and P-frames. The enhancement layer contains only B-frames. The B-frames are dependent on the frames in the base layer.

Spatial as opposed to temporal video scaling can also be used. Similarly to temporal scaling, it ultimately reduces the bit rate of the stream to be delivered. However, the method employed to do this is different. As can be seen in figure 2.9, spatial scaling subsamples video frames into smaller sizes implying that fewer bits will be sent per frame. If the connection allows, enhancement layers can be sent to improve on the quality [11].

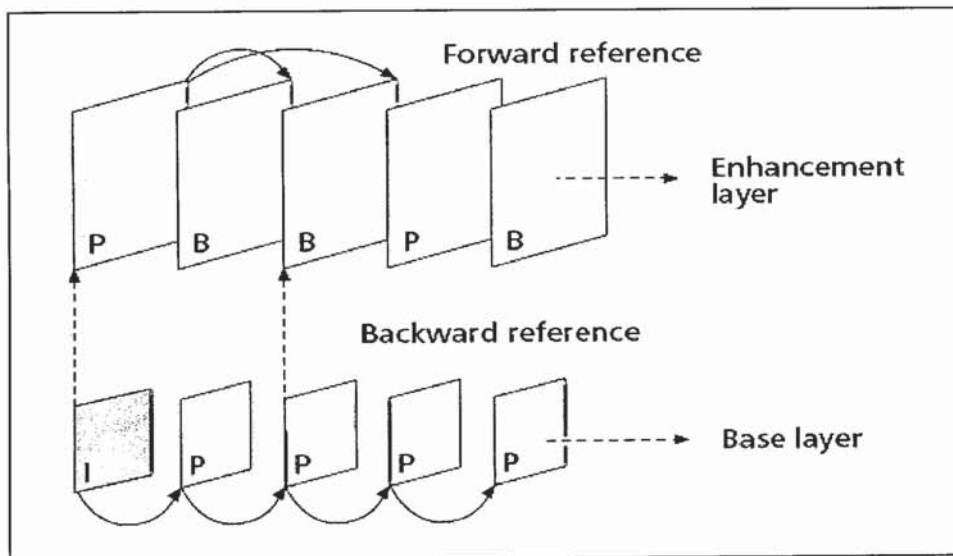


Figure 2.9 - Spatial Video Coding

Another method of scalable video coding that has recently come onto the scene is Fine Granularity Scalable Video Coding (FGS). As opposed to layered video coding where the goal is to deliver a series of complete layers, FGS not only encodes video into base and enhancement layers, but enables the granularity of these layers to be reduced by a number of bits depending on the available bandwidth. In other words, fewer bits can be transmitted for each enhancement layer. Therefore the streaming server can adapt to bandwidth differences by increasing or decreasing the number of bits to be transmitted from the enhancement layer. Such adaptations can also be performed by intermediate network nodes with such capabilities in case bandwidth is lacking. The advantage of this procedure is that clients unable to receive a full enhancement layer could receive part of the enhancement layer which improves on the quality of playing back the base layer only [12].

### **2.3.7 Studying Network Performance for Video Stream Data**

“In the last decade the networking community has witnessed an explosion in research on all aspects of video transport.” In order to study video transport in the networking field, video data must be available to researchers. However several issues arise with respect to video bit streams:

- The actual bit streams for videos are often copyrighted
- Manipulating the stream requires in-depth knowledge of video codecs
- Video files tend to be very large

In order to supply networking researchers with the needed data, video trace files are typically used in networking studies. A video trace file is a normal text file of small size relative to the actual video bit stream it represents. Video traces do not contain the actual video data. This resolves copyright issues. Trace files contain information such as video frame sizes, frame types and playout times. The downside of this is that since the video trace does not contain the actual bit stream and therefore a perceptual evaluation of the quality via playback is not possible. To solve this, quantitative values are given in the trace file to allow quality evaluation. These allow the determination of the quality metric called the peak signal-to-noise ratio (PSNR) of the received stream [11].

---

When examining video trace files one must keep in mind that the display sequence of frames is not the same as the order in which they are decoded (decoder sequence). Let us take the GOV seen previously in figure 2.7 as an example. The frames are to be displayed in the sequence IPBBPBBP. However they cannot be decoded in that order since B-frame 1 depends on P-frame 2 and B-frame 3 depends on P-frame 3. Therefore P-frame 2 must arrive at the decoder before B-frame 1 and P-frame 3 must arrive at the decoder before B-frame 3. Thus the correct decoding sequence would be: IPPBBPBB. The trace file shown below stores frames decoder sequence. The playback sequence can be deduced by looking at the playback time attribute - Time [ms]. Part of the trace file used in this thesis is in figure 2.10. The attributes are as follows:

- Frame number (the number of the frame according to the decoder sequence)
- Display start time in seconds
- Frame type (I, P or B)
- Frame size in bytes
- YPSNR (luminance)
- UPSNR (chrominance – hue)
- VPSNR (chrominance – intensity)

Fr#	Time [s]	Type	Len (B)	YPSNR	UPSNR	VPSNR
0	0.000000	I	12432	25.155399	35.763599	32.319801
3	0.100000	P	952	25.043400	35.551601	32.270901
1	0.033333	B	296	25.108299	35.524399	32.199600
2	0.066667	B	288	25.065201	35.489498	32.244598
6	0.200000	P	496	25.100500	35.387199	32.243401
4	0.133333	B	160	25.091400	35.387001	32.280201
5	0.166667	B	152	25.133900	35.704800	32.348598
9	0.300000	P	328	25.142700	35.524200	32.176

Figure - 2.10 Video Trace File

## 2.4 Disordering Communication Networks

### 2.4.1 Introduction

The resequencing problem arises due to the disordering nature of packet-switched networks. Packet-switching networks do not necessarily guarantee that packets will reach the receiver in the same chronological order in which they left the sender. This is because



not all packets necessarily follow the same path as in adaptive routing to handle congestion or link failure [1]. Therefore, packets may need to be reordered at the receiver in case of video or audio playback otherwise reconstruction of the multimedia sequence would not be possible.

### 2.4.2 Packet-Switched Network Delay

Since packets with the same source and destination can follow different paths, it is likely packets may experience different delays as discussed in [1]. Therefore a packet  $n$  sent at time  $T_n$  may experience a delay of  $D$  before reaching the receiving node, while a packet sent at time  $T_{n-1}$  may experience a delay of  $x \times D$  where  $x$  is a factor. In this scenario, the packet dispatched at  $T_n$  will arrive at time  $T_n + D$  whereas the packet dispatched at  $T_{n-1}$  will arrive at time  $T_{n-1} + x \times D$ . This produces the following possibilities:

Let  $A_n$  be the arrival time of packet  $n$  and  $A_{n-1}$  be the arrival time of packet  $n-1$ ;

$$\begin{aligned} \text{If } T_{n-1} + D_{n-1} > T_n + D_n \text{ then } A_{n-1} > A_n \therefore \\ \text{packet } n-1 \text{ arrives after packet } n \\ \text{If } T_{n-1} + D_{n-1} < T_n + D_n \text{ then } A_{n-1} < A_n \therefore \\ \text{packet } n-1 \text{ arrives before packet } n \end{aligned} \quad (\text{Eq. 2.3})$$

As seen above, the disordering of packets occurs due to differences in the overall delay. In general, the overall delay  $D_n$  encountered by a packet  $n$  is the sum of all possible associated delays  $d$ :

$$D_n = d(\text{processing}) + d(\text{transmission}) + d(\text{queueing}) + d(\text{propagation}) \quad (\text{Eq 2.4})$$

Obviously the reordering or resequencing procedure will cause part of the delay. The resequencing delay is part of the processing and queuing delay.

## 2.5 The Resequencing Problem

### 2.5.1 Introduction

The resequencing problem arises in distributed systems or systems receiving data over a communication network. Certain applications such as those supporting real-time video playback have packet ordering constraints i.e. they require data packets to be in the correct order before presentation. Running such applications in a distributed environment implies that packets arriving in disorder at the receiver must be reordered, or resequenced, before presentation.

### 2.5.2 Resequencing Methods

As mentioned in chapter 1, there are 2 types of reordering or resequencing:

- Total Order Resequencing - in this kind of resequencing, a younger packet must wait for all outstanding older customers before leaving the resequencing buffer. This way, the original order is restored.
- Partial Order Resequencing - in this kind of resequencing, a younger packet must wait only for a fixed time interval before leaving the resequencing buffer. Once the younger packet has left the resequencing buffer, any older customers arriving thereafter are discarded.

Total order resequencing is not suitable for real-time applications as it may cause large delays which could degrade the presentation more than a missing frame would do (similar to the head-of-line blocking problem in TCP). Partial order resequencing is better suited to real-time applications. The time for packets to wait in the resequencing buffer is subject to different parameters. In the case of video for the network studied in [2] it was shown that playback quality would improve if the delayed or lost I-frame is retransmitted once. However, this may imply a longer waiting time for younger customers already in the buffer. This interval of time is dependent on the network performance and the playback target time for the younger packet. In some cases if the waiting time is too long playback time may be missed. It could also be feasible to minimise the amount of time that I-frames wait in the buffer. The impact on performance resulting from various

---

waiting times with and without frame priority implementation will be investigated in this thesis.

### 2.5.3 Resequencing Delay

Naturally the resequencing process adds to the overall delay in the system. In [1], the total delay ( $G_n$ ) of packet  $n$  is represented as the following:

$$G_n = D_n + S_n + W_n \quad (\text{Eq. 2.5})$$

$D_n$  is the time taken for the packet to reach the resequencer (as shown in the overall network delay),  $S_n$  is the service time and  $W_n$  is the time during which a packet has to wait before being serviced and released. Note that in the case of total order resequencing packet  $n$  can only be serviced and released after packet  $n-1$ . This restores the original order. Therefore assuming  $A_n$  is the arrival time of packet  $n$ :

$$\begin{aligned} & \text{If } G_{n-1} + T_{n-1} \geq D_n + T_n \quad \text{then } T_{n-1} \text{ will wait for} \\ & T_{n-1} + A_{n-1} - (A_n - D_n) \quad \text{and will be released at} \\ & D_n + S_n + [T_{n-1} + A_{n-1} - (A_n - D_n)] \end{aligned} \quad (\text{Eq. 2.6})$$

What needs to be taken into account here is that the buffer at the receiver could be under-run due to the waiting time shown above causing jagged or absence of playback. This is where pre-buffering comes in. Pre-buffering is a waiting time assigned to the receiver during which video frames will buffer before playback begins. This means that the end user will have to wait for one period of the assigned pre-buffering time before playback begins. By implementing a pre-buffering period, it is less likely that the buffer will be under-run during playback.

A full mathematical module for the resequencing problem is provided in [1] but as a whole is beyond the scope of this thesis.

### 2.5.4 Resequencing in Multistage Disordering Systems

Resequencing can be applied as end-to-end or hop-by-hop resequencing.

1 - End-to-end resequencing - customers are resequenced after having traversed all disordering queues.

2- Hop-by-hop resequencing - customers are resequenced after each disordering queue.

The study on distributed database systems in [8] showed that hop-by-hop resequencing with batch processing in a two stage system, gives better performance than end-to-end resequencing.

### 2.5.5 Star Customers and Batch Departure

Customers arriving at the resequencer have sequence numbers or timestamps to show their relative order. The job of the resequencer is to reorder its customers or hold them until reordering is possible. In other words, once a customer  $C_n$  arrives, it cannot be processed unless  $C_{n-1}$  has been processed and therefore it must wait in the buffer.  $C_{n-1}$  is referred to as a star customer if it leaves the network after all customers having a lower timestamp. Customers leave the resequencing buffer in batches. It is possible to regroup these batches into new messages before sending them on to the next part of the network as in [8]. Consider the following initial input sequence of customers:

1 2 3 4 5 6 7 8 9 10 11 12

A possible output sequence from the network may be:

4 2 3 1 6 5 7 10 9 12 8 11

The numbers reflect, from left to right, the increasing value of network departure times.  $C_4$  is the first customer to reach the buffer. 1, 5, 7, 8, 11 are the **star customers**. The table below shows how the customers are processed.

Upon arrival of:	Leave the resequencing buffer
1	1, 2, 3, 4
5	5, 6
7	7
8	8, 9, 10
11	11, 12

Table 2.1 – Star Customers and Batch Departure

## 2.6 Research Motivation

### 2.6.1 Introduction

The motivation for this thesis comes from the need for improved network performance for video networking applications. The two-stage hybrid network scenario studied in this thesis assumes a WSN as the access network and an ISDN as the fixed network. Video data originates on the WSN. The technique proposed to improve performance in this study is rooted in [8] where resequencing and batch dispatching in gateway controllers is investigated for distributed database systems with resequencing constraints and from the findings in [2], in which streaming MPEG4 video over SCTP is investigated.

In [8] the application of gateway controllers and partial reordering to distributed systems is investigated for real-time systems with reordering constraints such as distributed databases. It was shown that delay could be minimised using these gateway controllers. In this thesis, the gateway controller will be adapted to account for video-specific data such as frame types (priorities) while the receiver will account for frame target playout times. Furthermore, a packet resequencing and aggregation scheme will be employed at the gateway to possibly reduce congestion on the fixed network. The network performance will then be assessed to verify whether this technique is beneficial.

In [2], a study of streaming video over SCTP shows that with a certain amount of pre-buffering – depending on network load and performance – and by assigning a reliability level of 1 to I-frames, viewing quality is noticeably enhanced. From this, it is possible to ascertain, as is well known, that since P and B frames are derived frames, they can be sent with a lower reliability than I-frames. This is because a missing P or B frame will not degrade the playback quality as much as a missing I-frame which would make decoding of its related GOV impossible. The percentage of I-frame data actually received was increased when their reliability was set to 1 instead of 0. This benefits quality.

This thesis will adapt approaches similar to those in [2] and [8] integrating them into the network with parameters such as waiting time, pre-buffering time and frame priority. As a whole this thesis focuses on the need for improved performance in real-time video transport in two-stage disordering hybrid networks by using the findings in the

---

two aforementioned research papers and aiming to implement some adaptation of these findings as suited to video data.

### **2.6.2 Necessity of Partial Reordering**

It follows that, in case of lost/late frames, sometimes a partial reordering must be done otherwise the buffer will be under-run while waiting for retransmitted frames in a total-reordering scheme. When considering a partial reordering there are of course parameters which need to be studied such as the amount of pre-buffering needed, the waiting time of customers, frame priority and the network load. The partial reordering scheme will be applied in this research.

### **2.6.3 Two Phase Resequencing Using Gateway Controllers**

When resequencing in a multistage disordering system we define end-to-end resequencing and hop-by-hop resequencing. The network in our research consists of a WSN which ultimately links to an ISDN through a gateway. Both networks are disordering networks. Multiple channels are used to send the data. Therefore a resequencing procedure is necessary. In [16] Varma shows that the total delay in end-to-end resequencing is stochastically smaller than total delay in a hop-by-hop resequencing system. However in [8] we see that if batch processing and aggregation is implemented along with hop-by-hop resequencing the delay is smaller.

### **2.6.4 Batch Processing and Aggregation**

Upon being reordered in the resequencing buffer of a gateway, packets could be reassembled into new, larger packets using a batching procedure such as the one described in [8] if the consecutive network's MTU permits. Once a batch is ready to be dispatched, it could be aggregated into one packet. This can decrease traffic in the consecutive network and possibly result in less disordering due to fewer packets traversing the network. With this approach, resequencing will be done at two points on the network route implying a 2-phase/hop-by-hop resequencing. The different scenarios are shown in figures 2.1 1a, b and c on the next page.

---

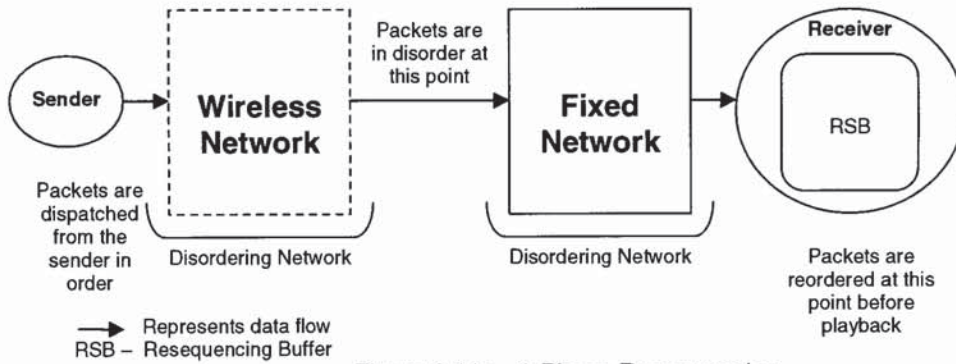


Figure 2.11a - 1-Phase Resequencing

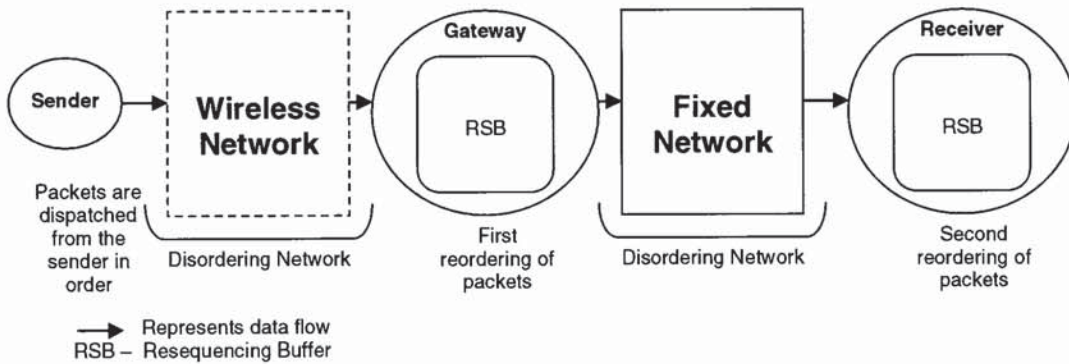


Figure 2.11b - 2-Phase Resequencing

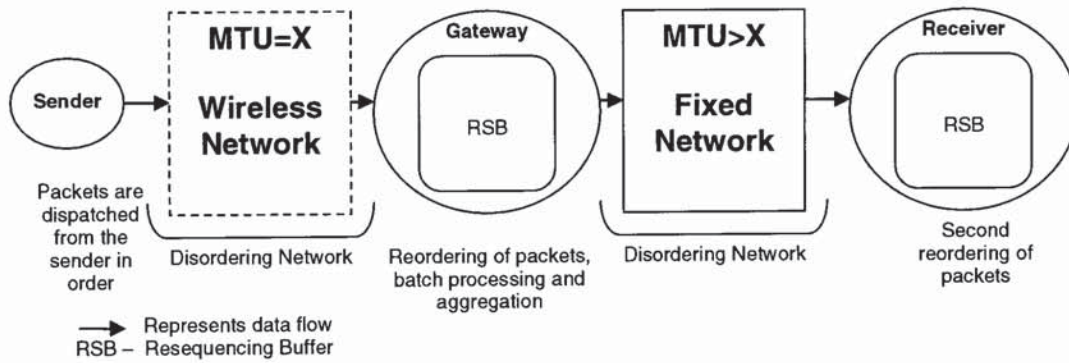


Figure 2.11c - 2-Phase Resequencing with Batch Processing and Aggregation

In paper [8] on distributed databases it is shown that the delay for such a two-phase system is smaller than that of a one-phase system of the same size and parameters. What remains to be assessed is the service time at the gateway controller in the different

scenarios along with video related parameters and whether the delay will be plausible in a real-time video system.

## **2.7 Summary**

This chapter introduced the main problems faced when transporting real-time video over two-stage disordering hybrid networks. MPEG 4, networking protocols, resequencing and related issues were discussed. The findings in several relevant papers were also noted. Finally the motivation for and scenario addressed in this thesis was presented.

---



## **Chapter 3**

# **Software Tools, Simulation Test-Bed and Performance Metrics**

## **3.1 Software Tools - OMNeT++**

### **3.1.1 Introduction**

The OMNeT++ discrete event simulation system [15] was used to create the network simulation in this thesis. OMNeT++ is a portable object-oriented modular discrete event network simulator and works on both Windows and Unix platforms. It provides a powerful set of libraries as well as a graphical environment for simulation based analysis and design of a broad range of systems such as traffic modelling of telecommunication networks, protocol modelling, modelling queuing networks, modelling multiprocessors and other distributed hardware systems, validating hardware architectures, evaluating performance aspects of complex software systems or modelling of any other system where the discrete event approach is suitable. OMNeT++ also provides tools for visualisation of output and statistics after a simulation run.

### **3.1.2 OMNeT++ Design**

The modelling paradigm used in OMNeT++ is an hierarchical block diagram. The lower layers encapsulate details of protocols and system behaviour and are known as “simple modules” while the top layer in the hierarchy represents the network topology and is known as the “system module”. Simple modules are coded in the C++ language using

---

classes provided in the simulation library. The system module contains submodules, which can also contain submodules themselves thus the hierarchical nesting concept. Modules that contain submodules are termed compound modules, as opposed to simple modules. This relationship is shown in figure 3.1. Modules can have their own parameters added to them depending on the requirements. Parameters allow customisation of a module's behaviour and flexibility of the model's topology depending on the given parameters.

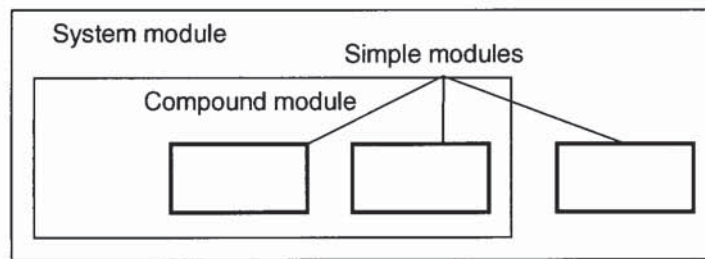


Figure 3.1 - Simple and compound modules

Modules pass messages (which can have complex data structures within them) to each other in order to communicate. Messages can be sent directly to their destination or via a predefined path i.e. a route, through gates and connections belonging to the modules on the path. Connections can take on certain parameters such as data rate and bit error rate. In figure 3.2 two different connection possibilities are shown. In the first submodules are shown connected directly to each other. In the second, submodules are shown connected to their parent module.

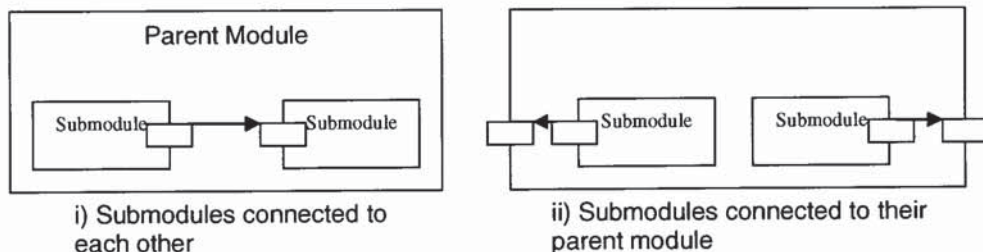


Figure 3.2 - Connections

Real-life examples for the module hierarchies and their possible connections are shown in figure 3.3. More details on OMNeT++ modeling tools can be found in appendices A1 and A2.

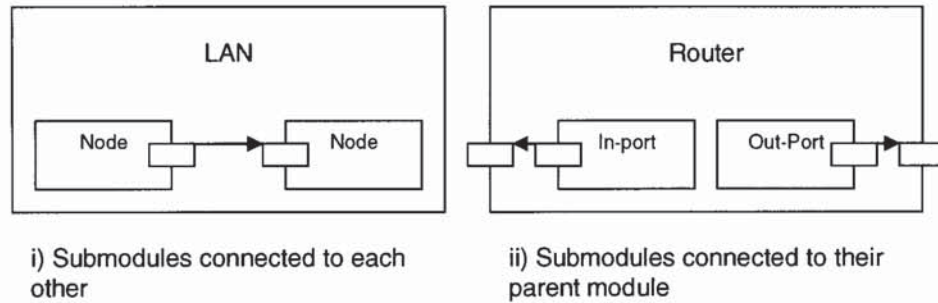


Figure 3.3 - Connections Example

## 3.2 Simulation Test-Bed

### 3.2.1 Introduction

In this research, OMNeT++ is used to create a network simulation that models a two-stage hybrid network made of a wireless access network connected to a wired network through a gateway node. In this model the data traffic begins at a single source and is received at one final destination. Different characteristics of the wireless and wired networks as well as the gateway are considered along with the amount and type of traffic being generated. Performance will be assessed according to the requirements of real-time multimedia such as Mpeg 4 files. In order to quantify the results, evaluation metrics for networking and video such as starvation probability, overhead/payload ratios and video quality can be used. These metrics will be discussed in this chapter and are also described in [11].

### 3.2.2 Assumptions and Network Characteristics

The network for which the simulation will be implemented is shown in figure 3.5 on the next page. A list of the assumptions for this network is given thereafter.

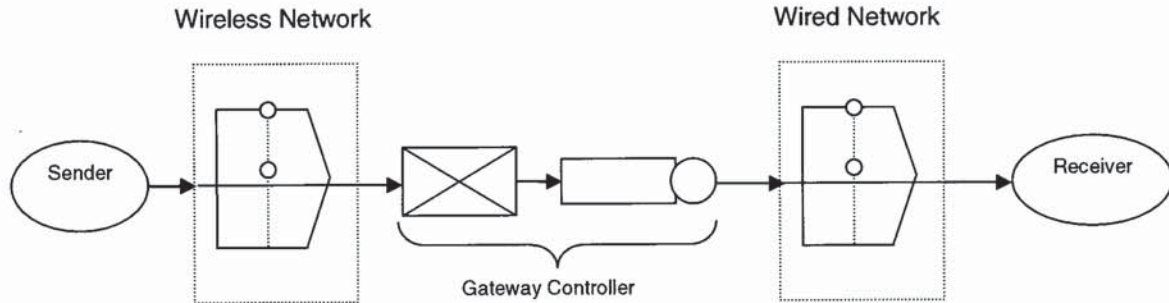


Figure 3.5 - Simulation Network

### Network

- The sender is located on the wireless access network while the receiving node is part of the fixed network.
- The frame size for the wireless network is assumed to be smaller than that of the wired network so as to allow aggregation of packets at the gateway controller. The frame size will be a changeable parameter.
- For simplicity, trials will assume that the BER is 0 and that there will be no retransmissions.
- Network service times are exponential however each channel also has an added fixed delay to provide for path differentiation.
- Channel speeds are taken to be 250Kbps and 384Kbps for the wireless and wired networks respectively. These speeds conform to wireless sensor network (WSN) and ISDN H channel standards.
- Results are obtained for an implementation consisting of four channels per network.

### Gateway

- Packets traverse the wireless network to the gateway where they are processed depending on the mode that the gateway is in. The gateway may be in one of three modes:
  - 1) normal mode with end-to-end resequencing only
  - 2) gateway controller mode with hop-by-hop resequencing

3) gateway controller mode with hop-by-hop resequencing and batch aggregation.

Performance metrics will be compared for the different modes.

- The resequencing buffer will implement partial order resequencing which is more suited to real-time applications than total order resequencing as discussed in chapter two.
- Batch processing and aggregation service time is assumed to be negligible and will therefore have a delay of 0 seconds. In [8] the batch service time is also assumed thus.
- For simplicity, no maximum gateway buffer size is assigned.
- Packets will be distinguished by their frame type priorities. Since this is to be done at the network or link level, the IP TOS field could be used to store this data. Depending on whether the file is 2-layer or 3-layer encoded we would need 2 or 3 priority levels respectively, one for I-frames, one for P-frames and another for B-frames since we are in ASP mode. With this data, I-frames can be given higher priority than P-frames and P-frames can be given higher priority than B-frames in lieu of their respective effect on playback quality. This is implemented by assigning different timeouts for waiting customers according to their priority. In other words, an I-frame would have a smaller timeout than a P-frame or B-frame. It is assumed that the target playback time of a frame is unknown at the gateway and therefore is not considered in the waiting timeout.

### **Traffic Generator**

- Traffic generation is based on a trace file in accordance with [11].
  - The transfer of the video data is taken to be unicast as opposed to multicast. In other words, for every client player a new connection is opened.
  - The MPEG file will be in MPEG-4 Advanced Simple Profile (ASP) as opposed to Simple profile (SP). The difference between these two profiles is that ASP contains I, P and B-frames whereas SP contains only I and P-frames.
-

### **Receiver**

- Target frame playback time and decoder sequences must be considered at the receiver in addition to frame priority when analysing loss and quality.
- Waiting time for late packets depends on the deadline of the video frame to which they belong. If packets do not make their deadline they are considered lost.

## **3.3 Performance Evaluation Metrics for Networking and Video**

### **3.3.1 Introduction**

This section discusses performance evaluation methods that can be used on traffic generated from video traces. When using video traces standard performance metrics such as network resource utilisation, delay and loss can be measured. These will be measured in the implementation part. It is also possible to measure video-specific metrics, such as the starvation probability and the video quality delivered to the receiver. Video related performance evaluation metrics are also described in the next two sections.

### **3.3.2 Starvation Probability and Information Loss Probability**

The starvation probability is related to packet loss or delay. If a fragment is delayed so much that it misses the playback time of the video message to which it belongs, the video frame is considered lost. For this reason, to assess the starvation probability, fragments arriving to the gateway controller or the receiver past their deadlines will be considered lost and so will the video frames to which they belong. Therefore, we define the starvation probability  $P(S)$  as the ratio of video frames that miss their deadline and are therefore lost, to the number of frames sent [11].

$$P(F) = \frac{\sum framesLost}{\sum framesSent} \quad (\text{Eq. 3.1})$$

It must also be recalled that I and P frames can have two deadlines:

- 1) Decoder sequence deadline – The time at which the frame must be present at the decoder in order to enable decoding of dependent frames whose playback/display times are smaller. If a frame misses its decoder sequence deadline it means that

frames which are dependent on it will not be decoded properly or in time and this will contribute to loss and quality degradation. A frame upon which other frames are dependent is called a “reference frame”.

- 2) Display sequence deadline – the display sequence deadline is the time at which the video frame itself must actually be displayed. If a frame misses its display sequence time then it will not be displayed and this could cause either a void in the playback or jitter, leading to frame loss and quality degradation.

As suggested in [11], video frames are given timestamps and sent in their decoder sequence order rather than their display sequence order to enable timely decoding of reference frames. When calculating the starvation probability, the loss of a reference frame constitutes the loss of all dependent frames. However the same does not apply for the information loss probability which is described next.

The information loss probability is similar to the starvation probability in that it quantifies data loss. The difference is that the frame loss probability is calculated at video frame granularity whereas the information loss probability is calculated at bit granularity. In other words the bits of a partially delivered frame would be considered lost in the starvation probability metric whereas in the information loss metric they would be added to the delivered bits when calculating the ratio. This ratio could be used to take into account that even when there is partial data loss, video codecs can employ techniques which may allow for the decoding of partially received frames [11].

$$P(I) = \frac{\sum bitsLost}{\sum bitsSent} \quad (\text{Eq. 3.2})$$

### 3.3.3 Video Playback Quality Assessment

When studying video transport it is important to have an indicator of the playback quality perceived by the user. It would be natural to assume that the more loss encountered, the lower the quality will be but this does not say anything about the quality perceived by the human eye. To get a better idea of the actual quality perceived by the user, the peak signal-to-noise ratio (PSNR) for received frames is calculated. To enable this, each frame in the video trace file has a PSNR value. For each frame fully received, its corresponding

PSNR can be used to quantify its effect on the quality. Part of a trace file is shown below. The YPSNR is the figure of interest since the luminance component since the human eye is most sensitive to luminance [10].

Fr#	Time [s]	Type	Len (B)	YPSNR	UPSNR	VPSNR
0	0.000000	I	12432	25.155399	35.763599	32.319801
3	0.100000	P	952	25.043400	35.551601	32.270901
1	0.033333	B	296	25.108299	35.524399	32.199600
2	0.066667	B	288	25.065201	35.489498	32.244598
6	0.200000	P	496	25.100500	35.387199	32.243401
4	0.133333	B	160	25.091400	35.387001	32.280201
5	0.166667	B	152	25.133900	35.704800	32.348598
9	0.300000	P	328	25.142700	35.524200	32.176

Figure 3.6 - Trace File with PSNR Values

The PSNR for a given video trace sequence such as the one above is measured in decibels (dB).

When using video traces, it is difficult to estimate the effect of a lost or partially delivered frame. However to get an idea of the received quality of the video sequence, PSNR values of lost frames and their dependent frames can be set to a low value e.g. less than 20dB. The PSNR of the sequence can then be taken to be the mean PSNR of all sequence frames. This is not completely accurate but serves as an estimate [11].

In order to more accurately quantify perceived video quality in “lossy” networks with dependent video frames, offset distortion traces have recently been introduced in [10]. In brief when a frame is lost, the last correctly received frame is usually re-displayed. Assuming that the lost frame stopped  $d$  consecutive frames in its GOV from being decoded then the last received frame will be displayed for  $d+1$  frame periods. Offset distortion traces allow the quality of such repeated frames to be evaluated by



including a PSNR value for each time a frame is repeated. Thus a more accurate quality evaluation over “lossy” networks can be made.

### **3.4 Summary**

This chapter presented the software tools used in this thesis as well as the performance and qualitative evaluation schemes. An overview of the assumptions made with respect to the underlying network characteristics was given and will be discussed in further detail in the next chapter.

---

## Chapter 4

### Performance Evaluation and Analysis

#### 4.1 Introduction

As a reminder, the three gateway modes included in this research are:

- 1) normal mode with end-to-end resequencing only
- 2) gateway controller mode with hop-by-hop resequencing
- 3) gateway controller mode with hop-by-hop resequencing and batch aggregation.

Modes 2 and 3 send messages according to a timeout maximum waiting time ( $W_{\text{max}}$ ). In other words, every  $W_{\text{max}}$  seconds a batch is sent. Each frame is also subject to a timeout such that high priority frames such as I-frames may have a lower timeout than others.

To evaluate performance and compare the 3 gateway modes included in this research, the simulation was run in a succession of several different trials per gateway mode to obtain the required quantitative results. The effect of different parameters on the system is shown in these trials. Data evaluation methods implemented are presented in detail in the next sections with their corresponding results. An analysis of the implications of these results is given thereafter.

---

## 4.2 Methods and Results

### 4.2.1 Delay

#### 4.2.1.1 Evaluation Method

The overall or Global delay ( $G$ ) of a fragment  $F$  is measured as the difference between its corresponding sending time ( $s_i$ ) and arrival time at the receiver plus end resequencing waiting time ( $a_i + r_i$ ).

$$G_F = (a_i + r_i) - s_i \quad (\text{Eq. 4.1})$$

Fragments are then reassembled into video frames at the receiver. The overall delay of the video frame itself  $G_{VF}$  is taken to be the difference between the smallest sending and largest arrival time of any fragment belonging to the frame in question.

$$G_{VF} = \max(a_i + r_i) - \min(s_i) \quad (\text{Eq. 4.2})$$

#### 4.2.1.2 Results

Figure 4.1 shows the average delay encountered by a video frame against its size for a fixed waiting time at the gateway controller.

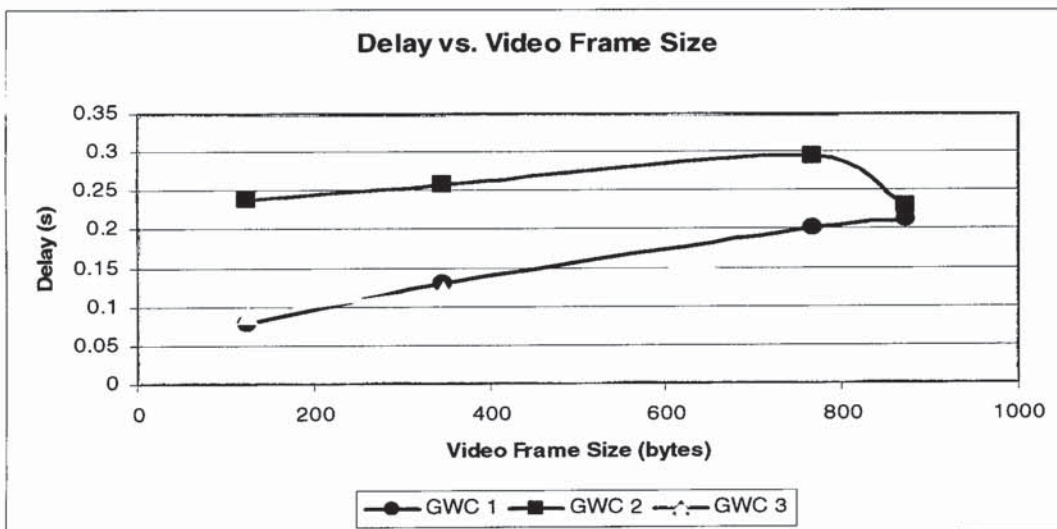


Figure 4.1 – Video Frame Size vs. Delay for 3 Gateway Modes

In the figure, one line for each gateway mode is shown. The results show that the delay generally increases with the video frame size. We can see that gateway type 3 exhibits the least overall delay while gateway type 2 shows the highest delay. It is already well known that hop-by-hop resequencing is less efficient than end-to-end resequencing and this is corroborated in the graph.

## 4.2.2 Batch Size vs. Maximum Waiting Time

### 4.2.2.1 Evaluation Method

The maximum waiting time ( $W_{\max}$ ) is the maximum time for which any fragment may wait in the resequencing buffer before it is aggregated into an ordered batch and sent onto the fixed network. The simulation was run for the following waiting times in order to evaluate the relationship to the batch size obtained:  $W_{\max} \approx 1/60, 1/30, 1/15, 1/10, 1/8$

### 4.2.2.2 Results

Figures 4.2 and 4.3 show the average batch size (in bytes and in fragments) aggregated by the gateway controller in mode 3 against the maximum gateway waiting time ( $W_{\max}$ ). From the figures we can see that as  $W_{\max}$  increases the batch size increases and then caps when it nears the MTU of the wired network.

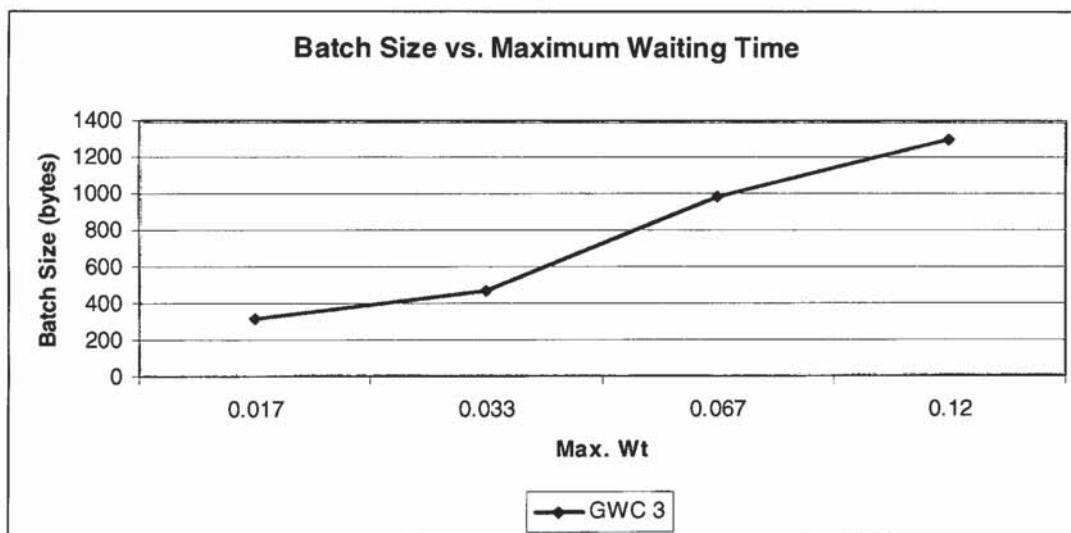


Figure 4.2 – Batch Size in Bytes vs. Maximum Waiting Time ( $W_{\max}$ )

It must be noted that difference between the MTUs in each part of the hybrid network determines the maximum number of fragments that can be aggregated into a packet at the gateway controller. The actual fragment size also affects this if it is smaller than the MTU. Overheads must also be taken into consideration. In our scenario, the maximum number of fragments per batch is around eleven. This is shown in figure 4.3.

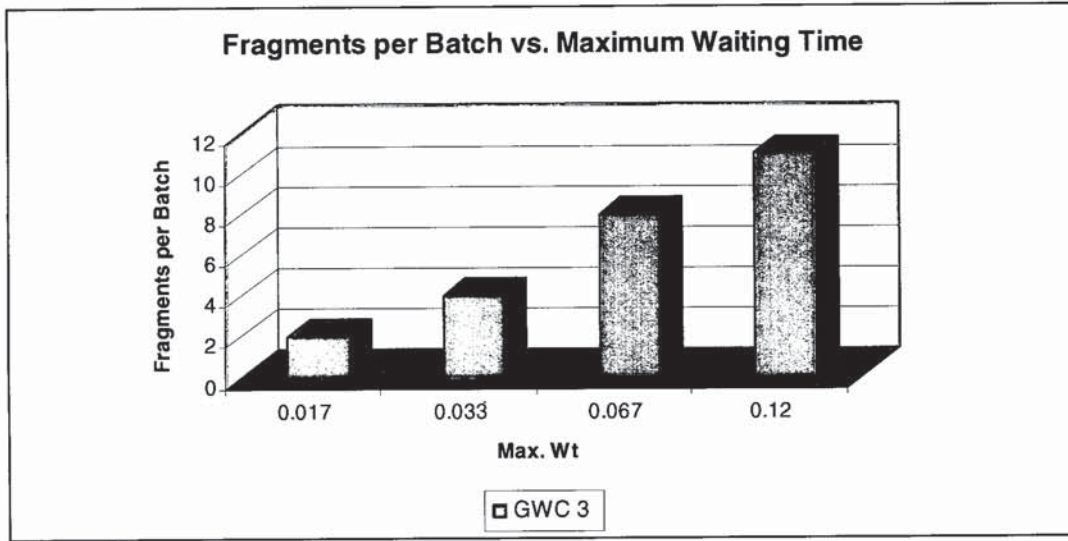


Figure 4.3 – Batch Size in Fragments vs. Maximum Waiting Time ( $W_{tmax}$ )

### 4.2.3 Overhead/Payload Ratio vs. Maximum Waiting Time

#### 4.2.3.1 Evaluation Method

The overhead/payload ratio ( $R_f$ ) is used in this study to calculate the relative amount of overheads to actual video data sent over the network. To evaluate the ratio for one fragment ( $R_f$ ) we would divide the header size ( $F_{LH}$ ) by the payload size ( $F_{LP}$ ):

$$R_f = F_{LH} / F_{LP} \quad (\text{Eq. 4.3})$$

In this study we take the  $R$  to be the average of all overhead/payload ratios for all fragments. If we have  $n$  fragments this implies:

$$R_{ALL} = \frac{\sum_{i=1}^n R_{Fi}}{n} \quad (\text{Eq. 4.4})$$

The overhead/payload ratio affects the efficiency of resource utilisation and network congestion. With higher overheads efficiency decreases and congestion increases.

#### 4.2.3.2 Results

In figure 4.4, for gateway type 3, the overhead/payload ratio is plotted against the maximum waiting time.

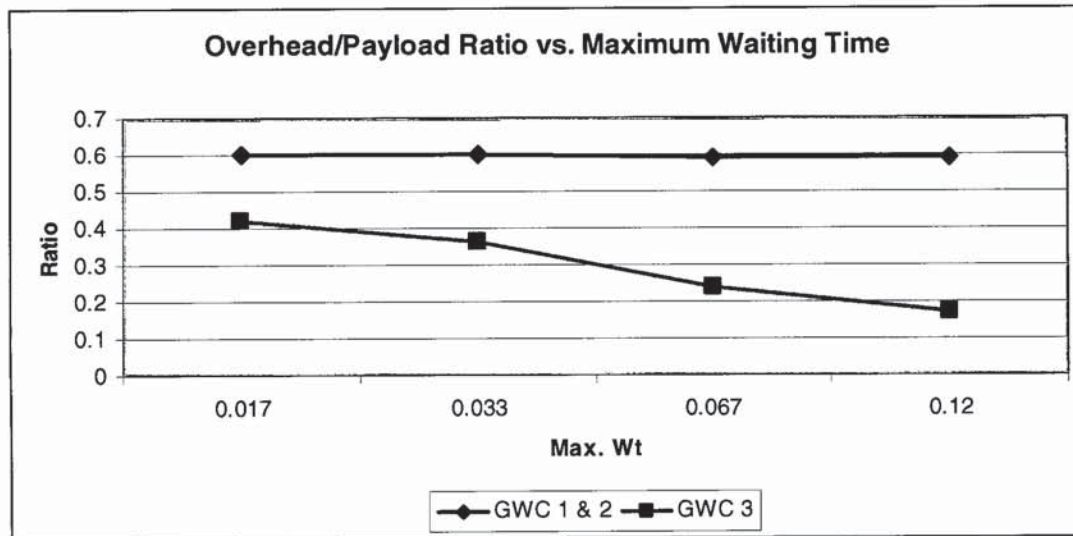


Figure 4.4 – Overhead/Payload Ratio vs. Maximum Waiting Time ( $W_{tmax}$ )

It can be concluded that the relationship is an inverse one with the overheads starting off high when the waiting time is small and decreasing by over half as the waiting time increases. This is a huge improvement on gateway types 1 and 2 which exhibit a constant and much higher overhead in all cases.

In figure 4.3 in the previous section we saw that the number of fragments per batch increases with the waiting time. This is relevant when looking at figure 4.5 which shows an inverse relationship between the number of fragments per batch and the overhead/payload ratio. As the number of fragments per batch increases, overhead decreases.

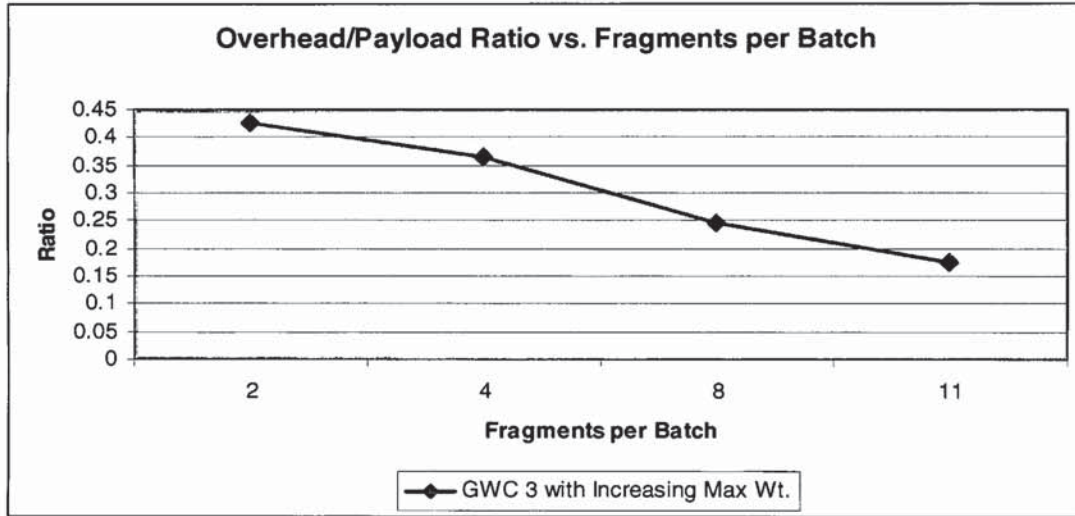


Figure 4.5 – Overhead/Payload Ratio vs. Fragments per Batch

#### 4.2.4 Starvation Probability vs. Pre-Buffer Time

##### 4.2.4.1 Evaluation Method

The starvation probability is one of the metrics used here to evaluate the performance of the network specifically for video purposes. Here two types of starvation probability are evaluated – the frame starvation probability  $P(F)$  and the information loss probability  $P(I)$ . Their equations were given in chapter 3. As a reminder they are the following:

$$P(F) = \frac{\sum \text{framesLost}}{\sum \text{framesSent}}$$

$$P(I) = \frac{\sum \text{bitsLost}}{\sum \text{bitsSent}}$$

The starvation probability here is measured against the amount of pre-buffering time set i.e. the amount of time that frames will buffer at the end user before playback begins. We take the results for two different maximum waiting times ( $W_{\text{max}}$ ) at the gateway controller.

#### 4.2.4.2 Results

In figures 4.6 and 4.7 we can see that the starvation probability decreases inversely with pre-buffering time. At 0 Pre-buffering time, frame starvation is 100%. It is clear therefore that a certain pre-buffering time is necessary in order to enhance playback.

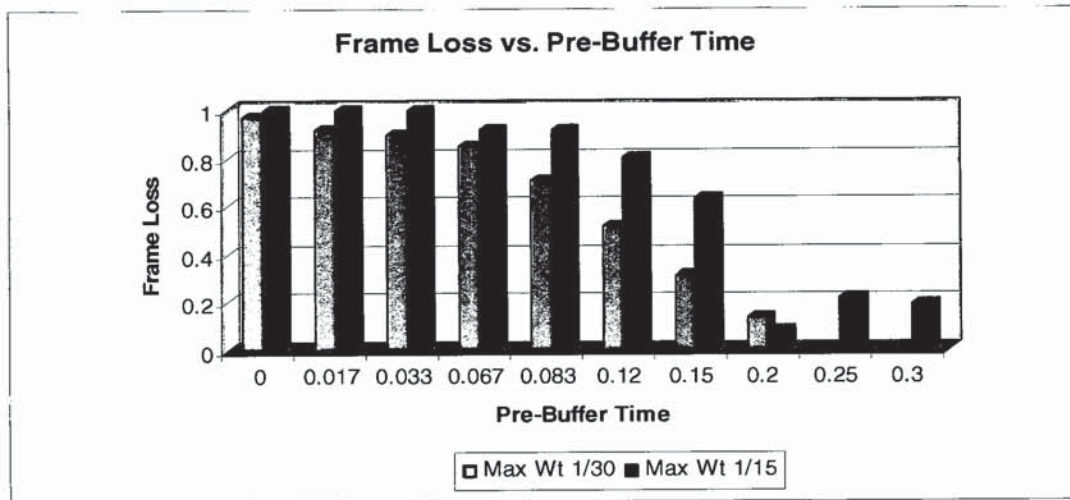


Figure 4.6 – Frame Loss probability vs. Fragments per Batch

The pre-buffering time is shown to be coupled with the maximum waiting time ( $W_{tmax}$ ) at the gateway controller. At  $W_{tmax} = 1/30$  a prebuffering time of 0.67 begins to decrease starvation. This is also the case for  $W_{tmax} = 1/15$ , however the starvation is decreased by a smaller factor. It is therefore necessary to balance these two parameters.

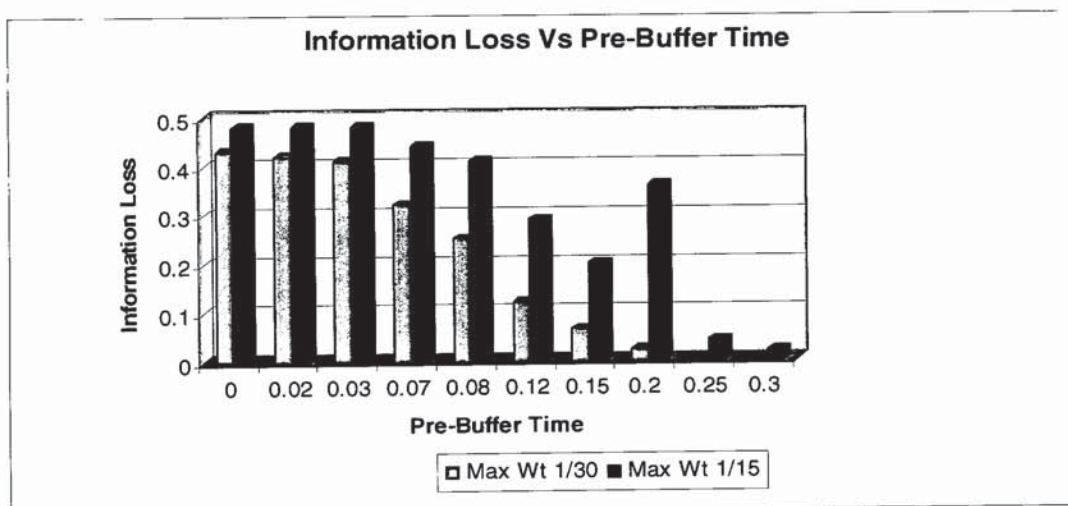


Figure 4.7 – Information Loss Probability vs. Fragments per Batch



Another important observation to be made is that the frame loss probability seems to exceed the information loss probability by an approximate factor of 2. This is because of the difference in their granularities as explained in chapter 3. Due to the error resilient video codecs used today, the information loss probability is often the one to be taken into consideration.

## 4.2.5 I-Frame Priority

### 4.2.5.1 Evaluation Method

In order to address the differences in priority/importance between video frame types, a special timeout value for I-frames ( $W_{ii}$ ) was added to the simulation. This timeout value is smaller than the gateway controller's maximum waiting time ( $W_{tmax}$ ). With a smaller waiting time, I-frames would move more quickly through the system. The impact of this special timeout is compared to the no priority cases shown previously.

### 4.2.5.2 Results

Figures 4.8 and 4.9 show the effect on the starvation probability when the I-frame timeout ( $W_{ii}$ ) is set to different values and ( $W_{tmax}$ ) is fixed. We can see that with a lower ( $W_{ii}$ ) the starvation probability is decreased. This is due to the decreased delay on I-frames which enables more of them to reach the receiver before their playout deadlines.

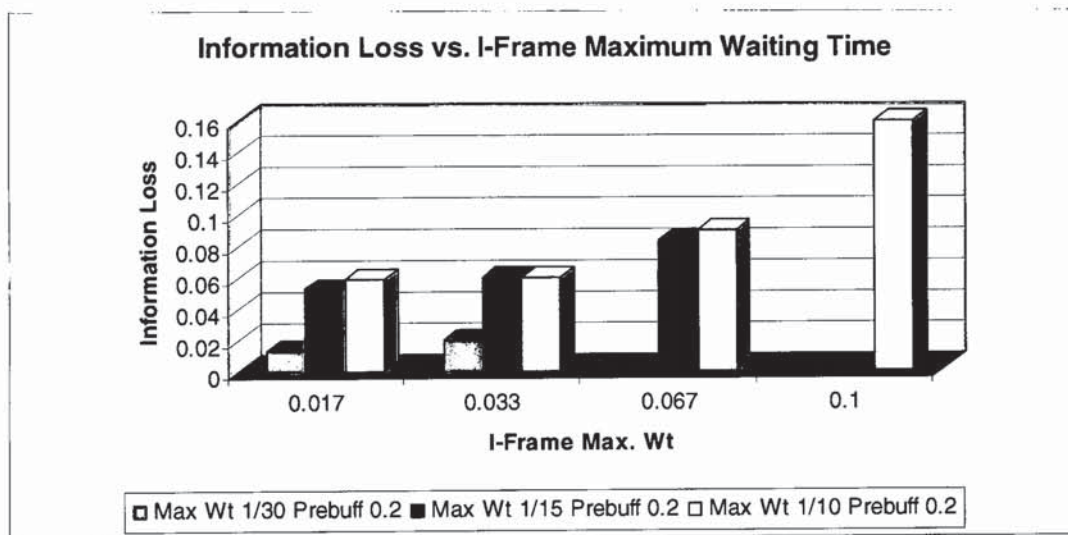


Figure 4.8 – Information Loss Probability vs. I-frame Maximum Waiting Time

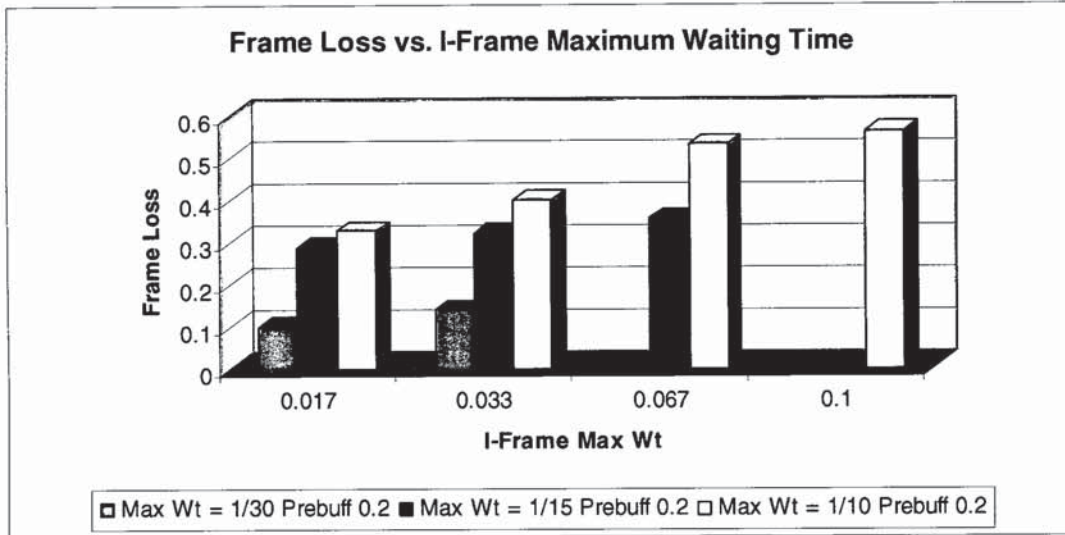


Figure 4.9 – Frame Loss Probability vs. I-frame Maximum Waiting Time

The trade-off when implementing I-frame priorities is that because I-frames have a decreased waiting time, and they are the biggest frames, the batches formed will be smaller and thus the overhead increases. That being said, the values can be adjusted depending on need. In figure 4.10 we see the differences in overhead for each I-frame timeout. Furthermore in figure 4.11 we see the number of fragments per batch depending on the I-frame waiting time. These results show that depending on the values chosen in the system, we can achieve a lot of flexibility in order to meet system requirements.

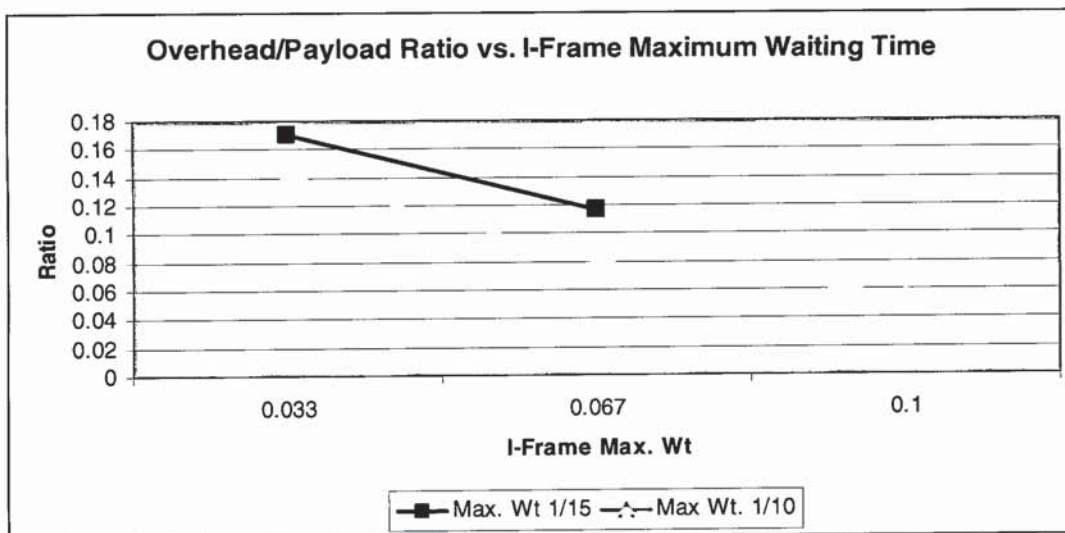


Figure 4.10 – Overhead Payload Ratio vs. I-frame Maximum Waiting Time

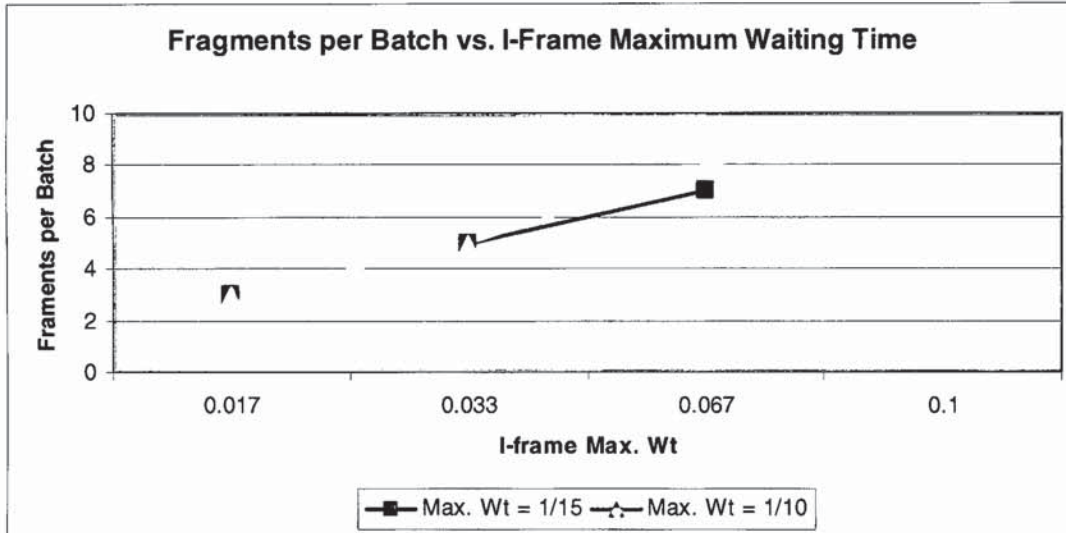


Figure 4.11 – Fragments per Batch vs. I-frame Maximum Waiting Time

To add to the above, in figure 4.12 the overall delay is shown for different video frame sizes. Each line represents a different I-frame timeout ( $W_{ti}$ ) with a fixed gateway maximum waiting time ( $W_{tmax}$ ). It can be seen that the delay is greatest when  $(W_{ti}) = (W_{tmax})$  again showing the benefit of decreasing the I-frame timeout.

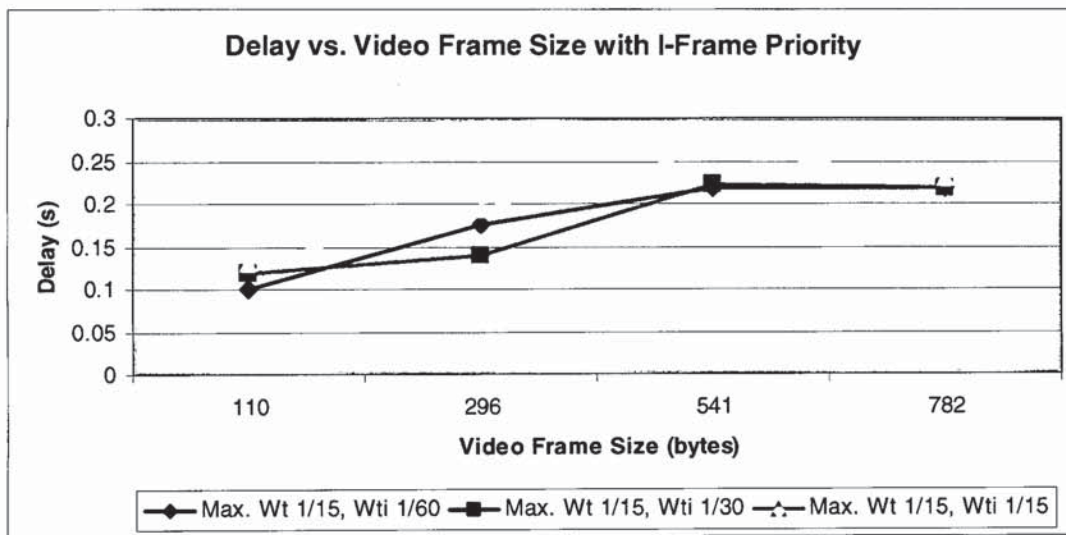


Figure 4.12 – Delay vs. Video Frame Size with I-frame Priority

### 4.3 Summary

In this chapter, the results of the simulation were shown for three gateway modes. Relationships between different parameters in the system were also analysed. The simulation consisted of a two-stage hybrid network with channels similar to those of WSN and ISDN H. Four channels were assumed with varying and exponential delays. The traffic generator sends realistic video traffic from a video trace file in ASP format.

The effect on delay ( $G_{VF}$ ) of the different gateway modes was evaluated and the different parameters of the system adjusted to define their relationships. The delay was taken to be the end-to-end delay of a whole video message (as opposed to a single fragment) in the system including the resequencing waiting time at the receiver. The delay was found to decrease under the impact of gateway mode 3 with resequencing and batch aggregation. Mode 2, implementing resequencing only, proved to be the least efficient.

It was also shown that batching highly reduces overheads and this is coupled with the gateway controller maximum waiting time ( $W_{tmax}$ ). In other words, the assigned  $W_{tmax}$  affects the size of the batch. If  $W_{tmax}$  is too small, the batching process will not be efficient because fewer fragments will make it in time to join the batch before timeout.

The starvation probability was found to be coupled with  $W_{tmax}$  and the pre-buffering time. Results showed that in order not to lose too many frames and therefore degrade playback quality, a certain pre-buffering time must be assigned according to the given  $W_{tmax}$ .

Concerning frame priorities, trials were run with two timeouts, one for  $W_{tmax}$  and a smaller one especially for I-frames. This meant that I-frames spent less time in the system. The impact of this has two results. On the one hand, starvation probability was decreased and on the other hand overheads were increased (batch sizes were decreased).

According to the above findings we can see that a combined resequencing and batch aggregation procedure at the gateway of a two-stage hybrid network is beneficial to performance and video playback. The system parameters are related in such a way that

---

they can be modified in a flexible manner to control batch sizes, overheads and information loss depending on the system requirements.



## **Chapter 5**

### **Main Contributions and Future Work**

#### **5.1 Introduction**

In this chapter the main contributions of the suggested resequencing and batch aggregation procedure for two-stage hybrid networks are highlighted. Noteworthy observations and results are summarized and possible future work on the topic is considered.

#### **5.2 Main Contributions**

The proposed adaptation in this thesis consists of adding “gateway controller” functionalities to two-stage hybrid networks for transfer of time and order sensitive video data. The job of the gateway controller is to resequence and aggregate batches of packets. This procedure is shown to improve performance. The need for improved performance in video transfer along with the different characteristics of hybrid networks paved the way for the proposition of implementing a gateway controller in this scenario.

The network studied in this research consists of a video traffic generator connected to a wireless access network which is in turn connected to a fixed network via a gateway. Thus we have a two-stage hybrid network. The gateway has added

---

functionalities terming it a “gateway controller”. These added functionalities are resequencing and batch aggregation. Several system parameters were considered including maximum waiting time at the gateway, different network MTUs, video frame priorities and receiver pre-buffering time.

The gateway controller was run in three modes 1- end-to-end resequencing 2 – hop-by-hop resequencing 3 – hop-by-hop resequencing with batch aggregation.

Type 3, which is the proposed solution, achieved a smaller delay than types 1 and 2. It also showed improved network utilisation by decreasing overheads.

Results also show that the relationship between the different system parameters that were studied allows for flexible modifications depending on system requirements. The maximum waiting time at the gateway can be increased to a certain threshold to create larger batches with less overhead. The pre-buffering time at the receiver is coupled with this to avoid increasing the starvation probability. I-frames can achieve priority standing by decreasing their waiting time which in turn decreases the starvation probability as well as delay.

### 5.3 Future Work

The work in this thesis can be extended from several angles. The following briefly outlines some possibilities for future work:

- The PSNR could be evaluated using offset distortion traces to more effectively evaluate the quality of the received video as perceived by the human eye. For this a new trace file with its corresponding offset distortion file would have to be used.
  - Results for different numbers of channels and different channel rates could be investigated to analyse the effect on frame loss, delay and aggregation.
  - A comparison of frame loss, delay and overhead for different waiting times could be done to achieve a middle-ground for all metrics. This would be done for a fixed pre-buffer time.
-

## Appendix A1

Overall, an OMNeT++ model consists of the following parts:

- NED language topology description(s) (.ned files) which describe the module structure with parameters, gates etc. NED files can be written using any text editor or the GNED graphical editor provided with OMNeT++.
- Message definitions (.msg files). Various message types can be defined with their own data fields. OMNeT++ translates message definitions into C++ classes.
- Simple modules which are C++ files, with the .h/.cc extension.

The simulation system also provides the following components:

- Simulation kernel. This contains the code that manages the simulation and the simulation class library. It is written in C++, compiled and put together to form a library (a file with .a or .lib extension)
- User interfaces. OMNeT++ user interfaces are used in simulation execution, to facilitate debugging, demonstration, or batch execution of simulations. There are several user interfaces, written in C++, compiled and put together into libraries (.a or .lib files).

The procedure of building a simulation programme in OMNeT++ with the above components is as follows:

First, .msg files are translated into C++ code using the `opp_msgc` programme. Then all C++ sources are compiled, and linked with the simulation kernel and a user interface library to form a simulation executable. NED files can either be also translated into C++ (using `nedtool`) and linked in, or loaded dynamically in their original text forms when the simulation program starts. Figure 3.4 shows the internal process of building and running simulations.

---



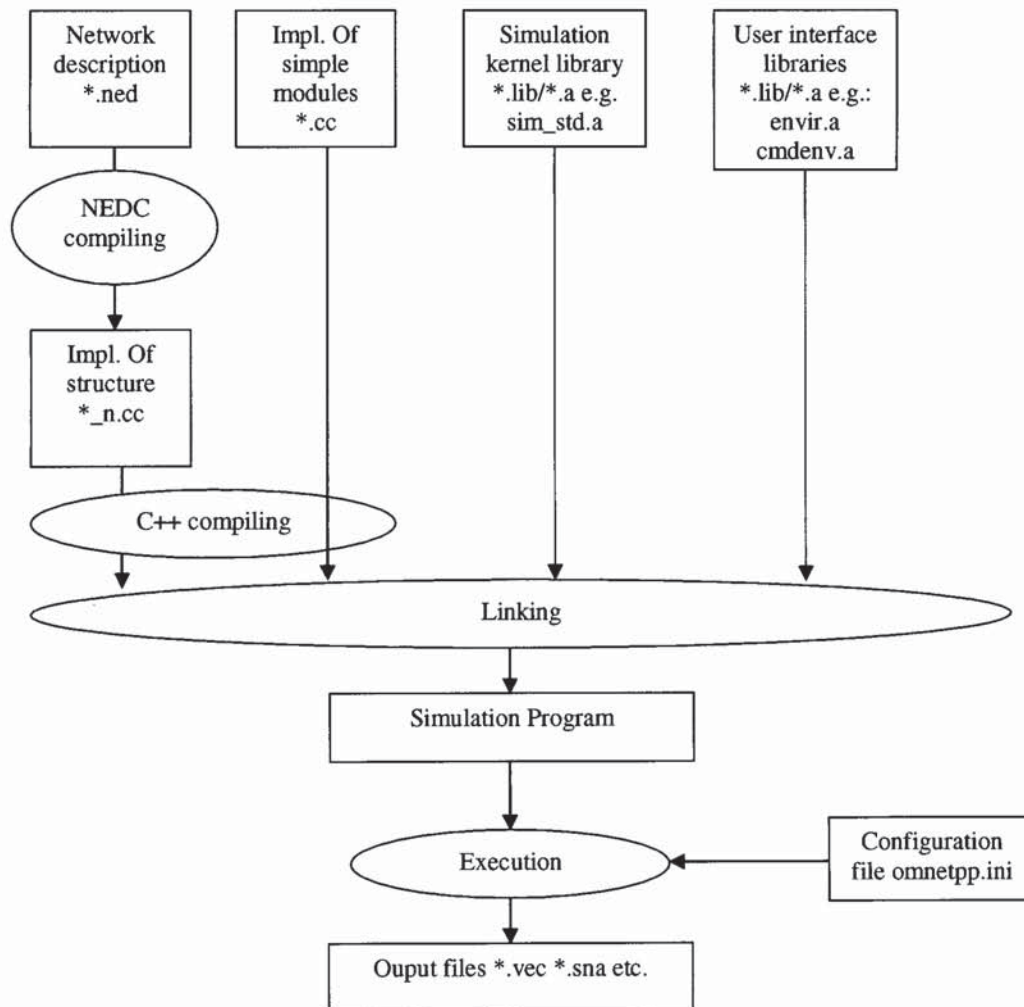


Figure 3.4 - Building and Running a Simulation

OMNeT++ simulations can feature varying user interfaces for different purposes: debugging, demonstration and batch execution. The simulator as well as user interfaces and tools are portable: they are known to work on Windows and on several Unix flavours, using various C++ compilers. OMNeT++ also supports parallel distributed simulation.

### 3.2.3 The INET Framework

The INET Framework is an open-source simulation package for wired and wireless networks. The INET Framework contains models for several Internet protocols: IP, IPv6,

TCP, UDP, 802.11, Ethernet, PPP, MPLS with LDP and RSVP-TE signalling, OSPF and several other protocols. Modules in the INET framework can be reused and integrated into simulations.

### **3.2.4 OMNeT++ Plove and Scalars**

OMNeT++ enables the recording of output data by incorporating output vectors and output scalars. Output vector and scalar files are typically obtained after a simulation run. They are recorded from simple modules using the provided library. Output vectors are time series data and can be used to record data such as queuing times, queue lengths or the number of dropped packets. Scalars are useful when comparing model behaviour under different parameter settings. In other words scalars could be used to record the number of packets dropped versus the set buffer size. OMNeT++ provides two graphical interfaces for plotting vector and scalar outputs to obtain graphical representations; these are OMNeT++ Plove and OMNeT++ Scalars respectively. Other tools can also be used to accomplish this task by simply reading or exporting the output vector and scalar files.

---

## Appendix A2

The code for the fixed exponential network in this simulation is shown here as an example of an OMNeT++ simple module.

```
#include <string.h>
#include <vector>
#include <omnetpp.h>

class FixedNet : public cSimpleModule
{
private:
    int messageCounter;
protected:

    virtual void initialize();
    virtual void handleMessage(cMessage *msg);
    simtime_t getServiceTime(cMessage *msg);
    void finish();
};

// The module class needs to be registered with OMNeT++
Define_Module(FixedNet);

void FixedNet::initialize()
{
    messageCounter = 0;
    WATCH(messageCounter);
}

void FixedNet::handleMessage(cMessage *batchMsg)
{
    int outGateNo;
    simtime_t service_time;
    if (strcmp("Net2", name()) == 0)
    {
        if(!batchMsg->isSelfMessage())
        {
            messageCounter++;
            ev << " Message Number = " << batchMsg->timestamp() << " has
            reached the fixed net " << endl;
            outGateNo = outGateNo = batchMsg->arrivalGate()->index();
            service_time = getServiceTime(batchMsg);
            sendDelayed(batchMsg,service_time, "out", outGateNo);
        }
    }
}
```

```
simtime_t FixedNet::getServiceTime(cMessage *msg)
{
    double Mu = 384000;
    double W4;
    double msglen;

    msglen = msg->length();

    W4 = (-1/Mu)*(msglen)*(log(1-uniform(0,1,0)));

    return W4;
}

void FixedNet::finish()
{
    recordScalar("MTU 2", par("frame_size2"));
}
```

**BIBLIOGRAPHY**

- [1] Francois Baccelli, Erol Gelenbe, Brigitte Plateau. An End-to-End Approach to the Resequencing Problem. INRIA, Le Chesnay, France, Universite de Paris-Sud, Orsay, France. Journal of the Association for Computing Machinery, Vol. 31, No. 3, July 1984, pp. 474-485.
  - [2] Alex Balk, Marc Sigler, Mario Gerla, M.Y. Sanadidi. Investigation of MPEG-4 Video Streaming over SCTP. Network Research Laboratory, Computer Science Department, UCLA, Los Angeles, USA. In proc. 6<sup>th</sup> World Multiconference on Systemics, Cybernetics and Informatics, 2002.
  - [3] Josh Beggs, Dylan These. Designing Web Audio, published by O'Reilly 2001
  - [4] Martin Jacklin, Professor Klaus Diepold, Olivier Ovaro. MPEG4 - The Media Standard. Technical University of Muncih, Apple Computer, France Telecom, MPEG-4 Industry Forum, 2002.
  - [5] James F. Kurose, Keith W. Ross. Computer Networking: A Top-Down Approach Featuring the Internet. University of Massachusetts, Amherst, Institute Eurecom. Pearson Addison Wesley, 2004.
  - [6] Lars-Ake Larzon, Mikael Degermark, Stephen Pink. UDP Lite for Real Time Multimedia Applications. Extended Enterprise Laboratory, HP Laboratories Bristol, April 1999. In Proc. IEEE Internation Conference 1999.
  - [7] Yihan Li, Shiwan Mao, Shivendra S. Panwar. The Case for Multipath Multimedia Transport over Wireless Ad Hoc Networks. Department of Electrical and Computer Engineering Polytechnic University, Brooklyn, NY USA, The Bradley Department of Electrical and Computer Engineering Virginia Tech, Blacksburg, USA. In Proc. 1<sup>st</sup> International Conference on Broadband Networks, IEEE, 2004.
  - [8] Hoda W. Maalouf, Mustafa K Gurcan. Minimisation of the Update Response Time in a Distributed Database System. Communications and Signal Processing Group, Department of Electrical and Electronic Engineering, Imperial College of Science Technology and Medicine, London. Performance Evaluation, Vol. 50, Elsevier, 2002, pp. 245-266
-

- [9] H.W. Maalouf. Optimisation of the Communication Network Performance of Distributed Systems with Resequencing Constraints, Ph.D. Thesis, Digital Communication Section, Department of Electrical and Electronic Engineering, Imperial College, University of London, 1988
  - [10] Patrick Seeling, Martin Reisslein, Frank H. P. Fitzek. Offset Distortion Traces for Trace-Based Evaluation of Video Quality after Network Transport (Extended Version). Dept. of Electrical Engineering, Arizona State University, USA, Department of Communication Technology, Aalborg University, Aalborg, Denmark. IEEE Consumer Communications and Networking Conference (CCNC) San Diego, CA, USA, pp.375-380, October 2005
  - [11] Patrick Seeling, Martin Reisslein, Beshan Kulapala. Network Performance Evaluation Using Frame Size and Quality Traces of Single-Layer and Two-Layer Video: A Tutorial. IEEE Communications Surveys and Tutorials, pp.58-78, Third Quarter 2004, Volume 6 No.3
  - [12] Patrick Seeling, Philippe de Cuetos, Martin Reisslein. Fine Granularity Scalable Video: Implications for Streaming and a Trace-Based Evaluation Methodology. Arizona State University, ENST, Paris. IEEE Communications Magazine, pp. 138-142, April 2005.
  - [13] Amoolya Singh, Almudena Konrad, Anthony D. Joseph. Performance Evaluation of UDP Lite for Cellular Video. Computer Science Division UC Berkley, USA. ACM Press, New York, 2001.
  - [14] Shweta Sinha. A TCP Tutorial. [www.ssf.org](http://www.ssf.org) , November 1998.
  - [15] Andras Varga. OMNet++ Discrete Event Simulation System Version 3.2 User Manual, March 2005.
  - [16] S. Varma. Performance Evaluation of the Timestamp Ordering Algorithm in a Distributed Database, IEEE Trans. Parall. Distri. Syst. 4 (6), pp. 668-676, 1993.
-