

## University Library

Author/Filing Title ..... AL-RAWI, M.M. ....

Class Mark ..... T .....

Please note that fines are charged on ALL  
overdue items.

FOR REFERENCE ONLY

040348183X






**Detecting Network Quality of Service on  
a Hop by Hop Basis for On-Line  
Multimedia Application Connections**

Momen M. Al-Rawi

A thesis submitted in partial fulfilment of the  
requirements for the Doctor of Philosophy degree of  
Loughborough University

August 2006

	Loughborough University Pilkington Library
Date	5/2009
Class	T
Acc No.	040348183X

## **Acknowledgements**

All praise and thanks are due to God without whom this work would not have seen the light at all. I am greatly indebted to my caring father and loving mother for their unlimited and continuous encouragement and my deepest gratitude goes to them for their constant support with all possible means particularly during the various stages of this research which I wished it to had been completed in my father's life time.

My great thanks and appreciation go to my supervisor Prof. David Parish for his constant guidance and valuable consultation throughout my research project. I would also like to thank my friends and colleagues at the HSN group Dr. Mark Sandford, Dr. Tammam Benmusa, Dr. Omar Bashir, and Mr. Peter Sandford for their valuable advice and help during significant phases of the research.

My deepest thanks go to my loving wife who stood by my side during the hardship which I had been through. Indeed her encouragement to me was an inspiration for me during this research and in my entire life. My thanks also are to my loveable and sensible children without whose patience and understanding this work would not have been completed in the first place.

## Abstract

With the highly expanding use of networked on-line Multimedia applications in general and Voice over IP applications in specific, it is vital to assess the quality of service of such applications over public or private networks to identify the problems arising in order to enhance the quality of these applications over networks. This is especially important with increasing large companies utilizing their private intranets for the use of telephone calls to save on their phone budget.

The quality of service can be addressed at various levels. This work is concerned with identifying the weak links or hops in a network path of an on-line multimedia application session which contributes to the degradation of the quality of the designated application. Once identified, the degraded hop can be dealt with or potentially replaced by another. Alternatively, the routing table can be altered to bypass the degraded hops.

Several approaches are considered in the thesis including Active Networks and the Management Information Base (MIB). Eventually, a prototype tool was designed to recognize the hops responsible for degrading the quality. Voice over IP was the application examined during this research. On-line Virtual Reality games and Web-Browsing were other multimedia applications partially considered in the later stage of the work.

A prototype tool was designed to assess the behavior of each network hop by measuring the fixed delay, variation in delay, and packet loss. This information was used to predict the performance of the application in question. Not all of these three parameters need to be considered, as some applications could be affected by those three elements whereas other applications are only affected by just one or two.

# Table of Contents

<b>Acronyms and Abbreviations .....</b>	<b>v</b>
 <i>Chapter 1</i>	
<b>Introduction .....</b>	<b>1</b>
1.1 Problem outline .....	2
1.2 Objective .....	3
1.3 Significance of the study .....	4
1.4 Publications .....	4
1.5 Thesis outline .....	4
 <i>Chapter 2</i>	
<b>Background and Related Research Overview .....</b>	<b>6</b>
2.1 Introduction .....	6
2.1.1 The Internet and its features .....	6
2.1.2 ICMP (Internet Control Message Protocol) .....	10
2.1.3 VoIP (Voice over Internet Protocol) .....	10
2.1.3.1 Main VoIP protocols .....	16
2.1.3.2 Elements of VoIP connection .....	18
2.1.4 Factors affecting VoIP transmission .....	20
2.1.5 Factors affecting voice quality .....	23
2.1.6 Voice quality measurement .....	24
2.2 Related work .....	25
2.2.1 Application detection .....	25
2.2.2 Application grading .....	26
2.3 Conclusion .....	27
 <i>Chapter 3</i>	
<b>Methodology to Determining QoS at the Hop Level .....</b>	<b>28</b>
3.1 Introduction .....	28
3.2 Main types of methods .....	29
3.2.1 Analytical technique .....	29
3.2.2 Simulation technique .....	31
3.2.3 Measurements technique .....	32
3.3 Parameters and metrics .....	33
3.4 Metrics measurements .....	34
3.5 Establishing grading for VoIP .....	38
3.6 Measuring network parameters at hops .....	39
3.6.1 Active Networks .....	40
3.6.2 Management Information Base .....	41
3.6.3 Prototype tool .....	41
3.7 Conclusion .....	42

<b>Chapter 4</b>	
<b>The Grading Algorithm .....</b>	<b>44</b>
4.1 Introduction .....	44
4.2 Calibration of quality of service grading .....	45
4.2.1 LNE (Loaded Network Emulator) .....	47
4.3 Quality classification .....	51
4.4 Grading algorithm .....	51
4.5 Conclusion .....	59
 <b>Chapter 5</b>	
<b>Approaches for Grading Quality of Service .....</b>	<b>61</b>
5.1 Active Networks .....	61
5.1.1 Activation of Active Nodes .....	62
5.1.1.1 Encapsulation approach .....	63
5.1.1.2 Discrete approach .....	64
5.1.2 Active protocols .....	65
5.2 Approaches to hop by quality determination .....	65
5.2.1 Time-stamping .....	66
5.2.2 Active Ping .....	69
5.3 Hop quality detection using MIB .....	72
5.4 Trial Experiment .....	73
5.5 Hop parameters extraction from the MIB .....	75
5.5.1 Network Management elements .....	75
5.5.1.1 Simple Network Management Protocol (SNMP) .....	76
5.5.1.2 MIB extraction .....	79
5.6 Feasibility of using MIB statistics for performance measurements .....	80
5.7 Conclusion .....	81
 <b>Chapter 6</b>	
<b>Hop by Hop Quality Grading .....</b>	<b>83</b>
6.1 Introduction .....	83
6.2 Experiment layout .....	85
6.3 Packet-loss dependency .....	86
6.4 Design of Prototype Tool .....	90
6.5 Algorithm for a Prototype Tool .....	94
6.6 Conclusion .....	99
 <b>Chapter 7</b>	
<b>Results Analysis and Discussion .....</b>	<b>100</b>
7.1 Introduction .....	100
7.2 Test results .....	100
7.2.1 Test network results .....	100
7.2.2 Internet VoIP test .....	111
7.3 Suggested solution .....	113



7.4 Other multimedia applications tests .....	114
7.4.1 VR-games test .....	114
7.4.2 Web-browsing test .....	116
7.5 Quality of Undetermined hops .....	118
7.6 Conclusion .....	120
 <i>Chapter 8</i>	
Conclusion and Recommendation for Further Work .....	122
 <b>References</b> .....	
References .....	126
<b>Appendix 1 (a)</b> .....	133
<b>Appendix 1 (b)</b> .....	134
<b>Appendix 1 (c)</b> .....	139
<b>Appendix 2</b> .....	142

## Acronyms and Abbreviations

ACK .....	Positive acknowledgement reply.
ADC .....	Analogue to Digital Converter.
ARQ .....	Automatic Repeat request.
Codec .....	voice coder/decoder.
Diff-Serv .....	Differentiated Services.
E-Commerce .....	Electronic Commerce.
FEC .....	Forward Error Correction.
ICMP .....	Internet Control Message Protocol
Int-Serv .....	Integrated Services.
IP .....	Internet Protocol.
ISP .....	Internet Service Provider.
LAN .....	Local Area Network.
MC .....	Multipoint Controller.
MCU .....	Multipoint Control Unit.
MG .....	Media Gateway.
MGC .....	Media Gateway Controller.
MGCP .....	The Media Gateway Control Protocol.
MIB .....	Management Information Base.
MOS .....	Mean Opinion Score.
MP .....	Multipoint Processor.
NAK .....	Negative acknowledgement reply.
PAMS .....	Perceptual Analysis/Measurement System.
PCM .....	Pulse Code Modulation.
PSQM .....	Perceptual Speech Quality Measurement.
PSTN .....	Public Switched Telephone Network.
QoS .....	Quality of Service.
RAS .....	Registration Admission Status.
RED .....	Random Early Detection.
RSVP .....	Resource reSerVation Protocol.

RTCP ..... **RTP Control Protocol.**  
RTP ..... **Real-Time Transport Protocol.**  
SG ..... **Signalling Gateway.**  
SNMP ..... **System Network Management Protocol.**  
SONET ..... **Synchronous Optical NETwork.**  
TCP ..... **Transmission Control Protocol**  
TOS ..... **Type Of Service.**  
UDP ..... **User Datagram Protocol.**  
VoIP ..... **Voice over IP.**  
VR Games .... **Virtual Reality Games.**  
VSP ..... **VoIP Service Provider.**  
WDM ..... **Wave Division Multiplexing.**

## *Chapter 1*

### **Introduction**

**W**ith many companies exploiting their private networks to save on their phone calls budget, Voice over IP applications are becoming increasingly popular. The quality of voice over a packetised network however has in some cases drawbacks in comparison with a switched network. Thus the demand to identify any quality degradation is a very active area of research.

Degradation of quality could occur for many various reasons. We are here concerned with the degradation related to specific sections of a network path due to network routing issues. Identifying which part of the network is responsible for the degradation of quality is a key element here. Going down even further to identify the exact hop or router, would point the finger to the exact source of the problem and can prove to be very useful in ensuring good quality over the network.

In this study the degradation of quality is discussed from the hops' performance point of view. There is a need for companies or corporations, which have private networks between their branches around the world over leased paths from various telecommunication companies, to know where the degradation of quality is coming from on its network.

Part of the network may offer lower quality for voice applications than is desirable. This would affect the overall end to end quality. If that part of the network or even the specific hop or router is identified, measures can be taken to overcome the problem either by upgrading the router, or even maybe changing the path all together by changing the leased line provider.

## **1.1 Problem outline**

Recently, the use of real-time multimedia applications over networks has increased rapidly. These applications such as Virtual reality games or Voice over IP are not supported by TCP which is used by network applications for its quality assurance. TCP offers some features which ensure reliable transmission of the data between the two end nodes. One of its important characteristic is retransmission of lost packets. To achieve this TCP makes sure that all the packets sent are accounted for at the destination. If some packets are lost and fail to reach the destination node, TCP retransmits those missing packets, and also puts every out-of-order packets in exactly the order in which they were sent from the source node.

These features of TCP are the reasons that make it not suitable for real-time multimedia applications connections, which cannot afford to wait for TCP to check the receipt of all packets, retransmit the missing ones and reorder all packets. The conversation in voice applications or the game session needs to be instantaneous. Thus using UDP as the transmission protocol is the obvious alternative, since it does not support any traffic problems such as lost packets or congestion, but rather just transmits the packets. This is what is demanded for real-time multimedia applications, since timely delivery of data is essential for these type of applications. However because of these characteristics of UDP, the multimedia session could suffer from delay or packet loss somewhere in the network if a part of the path is performing poorly. In a business environment, if this continues to be a repeatable situation it would mean that using the network for phone calls or video-conferencing is not practical, hence losing out on the economic advantage of using the network for such purposes.

Hence, it can be of importance for companies using network telephony, or for network managers as well as VoIP Service Providers (VSP) to determine if

congestions occurs at a certain section of the network on a regular basis, and whether this congestion can undermine VoIP quality for the overall connection.

Determining at which hop or router a VoIP connection suffers deterioration of its potential quality, would assist -for example- the VSP to change its network provider if the congestion occurs at a router under the support of a specific network manager, or the VSP could approach the network administrator to discuss and tackle this problem.

## **1.2 Objective**

Measuring the deterioration in an established, reliable network means measuring the Quality of service elements at the hop devices, since the medium in between is constant and has no effect on network congestion. Therefore, the aim of this work is to determine the hop or hops in which real-time multimedia applications, and VoIP in specific, suffers from degradation of its quality. The objective of this work can be summarised in the following points:

- Determining the specific QoS elements which affect the quality of VoIP application.
- Investigating and implementing several approaches to extract the information for the designated QoS elements.
- Assigning a grading mechanism for multimedia applications in general to measure the effect of a given QoS element on the application.
- Employing both the grading mechanism and the QoS elements' information extraction approach in order to detect any router or routers which have a degrading effect on the multimedia application and VoIP in particular.

### **1.3 Significance of the study**

This research offers several advantages of which this work can be utilised:

- It offers the end-user the chance to assess the quality of the application over the given network link before the application is practically initiated.
- It offers network manager a simple detection technique to determine the exact router which is responsible for quality degradation.
- The simplicity and universality of the detection technique through the use of low level data statistics of network parameters make quality detection technique a handy tool for companies which are using leased lines through several intermediate communication companies or service providers to connect to their different branches. The tool will allow for the determination of the specific hop or at least the link of the network which is responsible for the degradation of the application quality.

### **1.4 Publications:**

Momen Al-Rawi, David J. Parish, "Determination of VoIP (Voice over IP) service quality on a hop by hop basis", IADAT 27-29 Sept 06 International Conference on Telecoms & Computer Networks held in Portsmouth.

### **1.5 Thesis outline**

The sequence of the chapters in this thesis is as follows:

Chapter two introduces the background to networks and network elements which effect quality of applications, and gives an overview of related research.

Chapter three explains the methodology by which the research of this thesis was carried out.

Chapter four explains the grading method by which applications over networks are graded according to measurable network parameters; mainly Fixed delay, variable delay, and packet-loss.

Chapter five describes an Active Network based approach which was used for the hops' degradation detection. The two techniques experimented with (time-stamping and active-ping), are both discussed. Furthermore, the utilisation of SNMP (System Network Management Protocol) and MIB (Management Information Base) to determine the quality grading at designated hops is discussed.

Chapter six describes the finally adopted statistical approach in this research to determine the quality grading for real-time multimedia applications on a hop by hop basis through the use of a prototype hop quality determination tool.

Chapter seven describes a number of tests performed and the results obtained using the prototype hop quality determination tool to different destinations at various times during the day. The tests include VoIP, VR-games, and Web-browsing. The undetermined hop's quality case is then discussed.

Chapter eight concludes the study and points out some of the recommended further work.



## *Chapter 2*

### **Background and Related Research Overview**

#### **2.1 Introduction**

**W**ith the advent of the Internet, and the continuous development and advancement of hardware, services exploiting the Internet became important issues to the computer and communication community.

Recently the area of Voice over IP (VoIP) has gained momentum mainly for the cost advantage it has to offer over the traditional circuit switched network offered by telephone companies. Two major driving factors are behind the growth of VoIP technology. The quest for cheaper phone calls plays the main reason. Moreover, the expected volumes for those two supply and demand factors are very high, which are perfect elements to boost any market. According to Synergy Research, IP PBX equipment revenues were about \$10 billion in 2005 [Robins 2005]. Major companies have actually started deploying VoIP phones such as Merrill Lynch who are installing 35000 Cisco IP phones in 600 of their branches in the US [Hochmuth 2005].

##### **2.1.1 The Internet and its features:**

Originally, the Internet was intended for Data-transfer use. Nevertheless, it grew to accommodate all kinds of audio, video, and imagery transfer. However the transfer of such media poses new challenges and problems that have to be addressed. Currently the Internet is being upgraded to have a multiservice architecture to be able to efficiently deal with the different media applications as well as data.

The Internet is increasingly being utilised for commercial purposes. E-commerce (Electronic Commerce) is evolving as a wide range of commercial use and financial transactions over the Internet, whereby, for instance, customers

make purchases over the Internet through an automated service that is set up by the designated sales company. E-commerce could extend to the exploitation of VoIP so that both suppliers and customers get into contact and utilise VoIP for online purchases. Moreover, more companies are currently adopting IP based call centres on private intranets, to escape the downsides of the public Internet, which is slow and unreliable. These private VoIP networks are proving to be both high-quality as well as cost-effective.

These are some main Internet features, which define the way in which it generally works:

**Best effort:**

Where traffic is discarded if problems occur. Such as increase in latency for live voice applications, or buffer overflow.

**Address aggregation:**

The concept of grouping several subnets in to one network (ID grouping). Such as all the subnets which are under the same ISP (Internet Service Provider) address. This approach allows the Internet components to be identified with a hierarchical address.

**Multicasting:**

Multicasting is one of the main advantages of the Internet. It allows the user of the Internet to send data, whether e-mails, files, audio or video to many sites or end-users at the same time. This feature itself makes the Internet perfect for conferencing, where several parties in different geographical places can organise a conference between them over the Internet.

**Connectionless:**

The Internet in its general form is connectionless. That is no direct circuits are set up between users. This aspect of the Internet goes hand in hand with the Adaptive routing concept.

**Round-Trip Time (RTT):**

Is a measure of the time it takes to send a packet to a destination node and receive a reply from that node, including the transmission time in both directions plus the processing time at the destination node itself.

Most RTTs in the Internet are within the range of 70-160 ms, although it may exceed 300 ms when Internet traffic is heavy [Black 2000, pp: 38].

**Hop Distance:**

Is the number of hops between the sender and the receiver. That is to say the number of times the IP datagrams have to be processed during this journey. This includes the sender and receiver's workstation as well as each packet switch, such as routers, in which packets pass through. Hop distance is more of a key factor in delay than actual geographical distance.

**Order of arrival of packets:**

The order in which packets arrive is not of significance to data applications, since TCP on the receiver side puts them back in order. Nevertheless, the order of packet arrival is important to voice and video, where TCP is not used for their transfer.

**Packet loss:**

The Packet loss issue can be viewed from two angles:

- (a) How often packet loss occurs.
- (b) How many successive (contiguous) packets are affected.

In order to overcome or reduce the effect of packet-loss, some loss recovery schemes were developed to deal with this problem. ARQ (Automatic Repeat reQuest) is an example, whereby data transfer protocols use both positive (ACK) and negative (NAK) Acknowledgment replies to confirm received packets or request repetition of lost or distorted packets.

FEC (Forward Error Correction) is another technique used to detect and correct errors. The main idea of FEC is to send extra information along with the packet stream, whereby this information could be utilised to reconstruct the exact lost packets or a close approximation of them. FEC is sometimes used along with the ARQ to decrease the number of retransmitted packets and the delay associated with that waiting for the receiver NAK message and for the retransmitted packet to be sent again by the sender.

Interleaving is another technique used to reduce the effect of packet-loss, mainly by reordering the audio information to be sent whereby every two adjacent packets, which are set to be small in duration, are separated by being allocated within different chunks of packets. Accordingly, if a packet is lost the result would be multiple small gaps in the reconstructed stream at the receiving end, which would not be the case if interleaving is not used where the result would be single large gaps resulting in seriously degraded audio quality.

#### **Packet size:**

Generally speaking, the average packet size varies between 576 bytes down to 400 bytes and maybe even to 175 bytes in some cases. Nevertheless 10 percent of the traffic may reach packet sizes of about 1500 bytes [Black 2000]. Acknowledgments, finish, and reset messages are about 40 bytes. As for VoIP, it is important to use small packet such as 10-30 bytes in length or 10-30 ms in duration [ibid pp: 43]. Mainly because if the router has to wait until the entire packet is received before it processes it. Thus if the packet is large the voice will be delayed which will affect the quality as well as the usability the voice application. If a substantial amount of Internet traffic becomes voice traffic, it will require an increase in Internet capacity, because the smaller packets will consume more of the overall bandwidth. However, the increased use of high-speed SONET links (Synchronous Optical NETwork), and the migration to WDM (Wave Division Multiplexing) will provide the needed bandwidth.

### Fixed Routing:

The more the packets change routes, the more this affects real-time delay-sensitive applications such as voice and video applications. Although it appears that the need for fixed routing is essential for such applications, studies reveal that most of the traffic between two or more communicating parties remains on the same physical path during the session [Paxson 1997]. Thus it is rarely that routes will keep changing during a voice conversation.

#### 2.1.2 ICMP (Internet Control Message Protocol):

ICMP is usually thought of as part of the IP layer. It communicates error messages and other conditions that require attention.

ICMP messages are transmitted encapsulated within IP datagram (Figure 2.1).

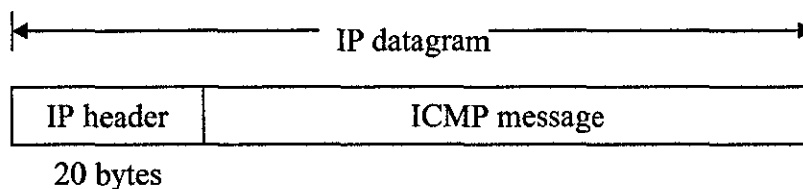


Figure (2.1): Encapsulation of ICMP message within IP datagram.

Some rules are set to prevent the *broadcast storms* that have occurred in the past when ICMP errors were sent in response to broadcast packets. For example, an ICMP error message is never generated in response to an ICMP error messages (If this were not the rule, we could end up with an endless scenarios of an error generating an error, which generates an error, and so on, indefinitely).

#### 2.1.3 Voice Over Internet Protocol (VoIP)

Voice over Internet Protocol is an increasingly popular means of audio communication over internet, whereby the audio conversation is transmitted over the packet switched network instead of the common telephone circuit switched network. The sequence in which a VoIP transmission takes place is as follows:

## **I. Digitisation:**

When using the PC for audio conversation the user would speak through a microphone which is connected to the Mic port of the PC. The analogue voice signal which is transmitted through the Microphone has to pass through an ADC (Analogue to Digital Converter), which is practically part of the sound card hardware attached to the computer's main-board. It is common nowadays that sound cards have the capacity of at least 16 bits per sample with a bandwidth of 22050 Hz. To digitise it the audio component should be sampled at least at twice the frequency of the highest analogue frequency (Nyquist principle) as follows [Pohlmann 2005]:

$$(22050 \text{ Hz}) * 2 = 44100 \text{ Hz}$$

This gives a throughput of:

$$(44100 \text{ samples per second}) * (2 \text{ bytes}) = 88200 \text{ bytes per second.}$$

Obviously, for a VoIP conversation, 88.2 k bytes per second throughput is quite sufficient.

## **II. Compression by Codecs:**

Acquiring digitised data from the analogue signal is just a pre-step. Digital data in its raw format is not sufficient to be transmitted over the network. It has to be transferred into an agreed-upon format that can be recognised by all the different receiver ends.

Moreover, since the digitised signal will be transferred into another format before sending, it is useful to take advantage of this process and compress the data in order to speed the sending process (since the data or number of packets will be reduced), tolerating link error, and saving bandwidth.

When choosing a compression scheme, compression rate must be taken into consideration. The higher the compression, the less bits have to be transferred for a telephone call. This can reduce the traffic (especially if nearly all traffic is

telephony), thus reducing the end-to-end delay. But on the other hand, more encoding and decoding time is needed. Moreover, end-to-end delay can increase because a larger block of speech data is often processed. Loss can often become more of a problem as more speech is lost if a packet is lost.

For such purposes, different compression algorithms have been developed and agreed on by various technical bodies such as the ITU-T. Some of them are popular for general use because they proved to be more efficient for general-purpose applications, such as ITU-T G.723, ITU-T G.728, and ITU-T G.729 [Hardy 2003]. These coders encode the **PCM** (Pulse Code Modulation) digitised data of the user's voice into a compressed format, which is then decoded to PCM data at the receiver end, and then converted into a wave-form to be transmitted over the user's speakers [Hersent 2005].

There are three categories of voice coders:

- (a) **Waveform coding.** These coders basically reproduce the analogue waveform as precisely as possible, which of course includes noise. This is done with a high bit-rate, hence, these coders usually use 64 k-bit per second such as the ITU-T G.711.
- (b) **Vocoding.** These types of coders are set to send certain parameters to the end receiver. These parameters are then used to model the speaker's vocal tract [West 1996]. Such coders do not usually offer a good quality speech when used in telephony systems.
- (c) **Hybrid coding.** Hybrid coders are the best choice to be used for VoIP. This is because they offer close quality to those of the waveform coders in regard to the analogue waveform reproduction, and apply the concepts of Vocoding. Yet they operate at a very low bit-rate of 4 to 16 k-bit per second, whilst toll-quality coding can operate on as low as 8 k-bit/second.

Although using compression compromises the quality of the original image or audio file, some types of compression techniques offer a very good quality that can hardly be distinguished from the original file.

Moreover, compression has positive effect on the network because it reduces the bit-rate thus requires less bandwidth which is positively reflected on the overall network load.

### **III. RTP (Real Time Transport Protocol)**

In addition to IP, VoIP uses the real-time protocol (RTP) to help ensure that packets get delivered in a timely way.

In a typical data packet case, transfer would generally be through TCP and then IP before it is passed down to layer 2 and 1. When it comes to voice packets however, UDP is a preferred protocol over TCP for various reasons that have to do with certain TCP features which contribute to the delay of voice packets.

The problem with UDP however, is that it does not solve problems such as out-of-order packets, which are usually solved by TCP. Hence, RTP is utilised to take care of such and similar problems such as disposing of late packets, yet trying to maintain packet transfer in a real-time manner.

### **IV. UDP (User Datagram Protocol)**

TCP has evolved to be a good transmission protocol that offers reliability to packets transmission because of its support features. These exact same support features make TCP protocol unsuitable for real-time voice transmission, because of the delay it poses on voice traffic. For example some of the packets which have errors are retransmitted. The accumulative delay could pass the acceptable RTT limit for voice transmission of around 400 ms or less. For telephone traffic RTT is recommended to be limited to 300 ms. Otherwise, usability is reduced.

Another example would be the TCP feature of Flow control. TCP halts from sending packets if there is traffic congestion. It then waits for the right time to continue sending packets. Obviously, this feature contradicts the nature of the real-time transmission of voice packets. Since UDP does not have this feature, voice packets will be sent regardless of any traffic congestion. This makes UDP a better alternative to TCP when it comes to VoIP applications.



## V. RSVP (Resource Reservation Protocol)

A lot of effort has been put to develop different approaches to manage the QoS in networks. Two of these are the IETF - Integrated Services (*int-serv*) [Breden 1997] and Differentiated Services (*diff-serv*) [Ferguson 1997] service classification standards which attempt to extend the best effort delivery service provided by the (largely unchanged) current model.

The *int-serv* model is based on reservations-based traffic. This approach reserves resources explicitly using a dynamic signalling protocol (RSVP) and employs admission control, packet classification and intelligent scheduling to achieve the desired level of QoS.

Accordingly, RSVP is an IP layer protocol. Its role is to manage the quality of service. It caters for the needs of the application in hand. That is to say it enables the reservation of capacity on the internet. It sets the traffic requirements that would make that application run smoothly, such as the required throughput and delay limits. These requirements are taken into consideration in all the hops along the route of which that voice traffic, for example, is being transmitted.

Some of the factors that are requested by the RSVP are:

- a) TOS (Type of Service) in the IP header. The router can be configured to recognise voice traffic, in order to treat it differently from data traffic. This can be achieved by using the TOS field in the IP header of the IP datagram. Otherwise it would be considered as just another datagram i.e. Data traffic. In the TOS field, a low value is assigned for a high urgency packet, and a high value for a low urgency packet. Thus, RSVP would request low values in the TOS field for VoIP application in order to achieve real-time manner transmission.
- b) Queuing packet method. Upon receiving the RSVP request, a designated queuing method would be assigned for real-time voice packets, so that they are not delayed, in order to maintain fast packet transfer.
- c) Shaping capability, that allows the limiting of the source to a fixed bandwidth in: download, or upload.
- d) Congestion avoidance, such as RED (Random Early Detection).

The use of RSVP however raised concerns amongst many Internet Service Providers (ISP's) over the amount of computational, memory, and other system resources which might be consumed by managing thousands, or perhaps hundreds of thousands, of flows. This led to the IETF Integrated Services working group to examine several methods to define less resource intensive mechanisms. The *diff-serv* model is based on reservation-less traffic assumptions. It classifies packets into a small number of service types and uses priority mechanisms to provide adequate QoS to the traffic. No explicit resource reservation or admission control is employed although network nodes do have to use intelligent queuing mechanisms to differentiate traffic.

## **VI. IP (Internet Protocol):**

Internet Protocol (IP) is a network layer protocol. Its main job is to transfer the data from the source to the destination through the needed routers and switches. IP, nevertheless, does not guarantee the arrival of the packet or its quality. For example, assume an IP gateway adopts a maximum queue length size, so if this queue length exceeds the maximum size, the buffer will overflow, thus the additional traffic or datagrams will be discarded from the network. For this reason a higher-level transport layer protocol (such as TCP, or RTP in VoIP transfer) is essential to recover from these problems. If the routers are fast, and sufficient bandwidth is available, then IP does not introduce significant overhead to the support of VoIP. There are better mechanisms. Nevertheless, no other mechanism has the universal presence of IP (and the benefit of an international IP addresses).

During the transmission of voice packets through the internet, it is only the IP header of the packet that is processed at every hop. The other upper layers headers such as UDP and RTP headers are not examined except at the receiver end. This concept is known as the VoIP Tunnel, as illustrated in figure (2.2).

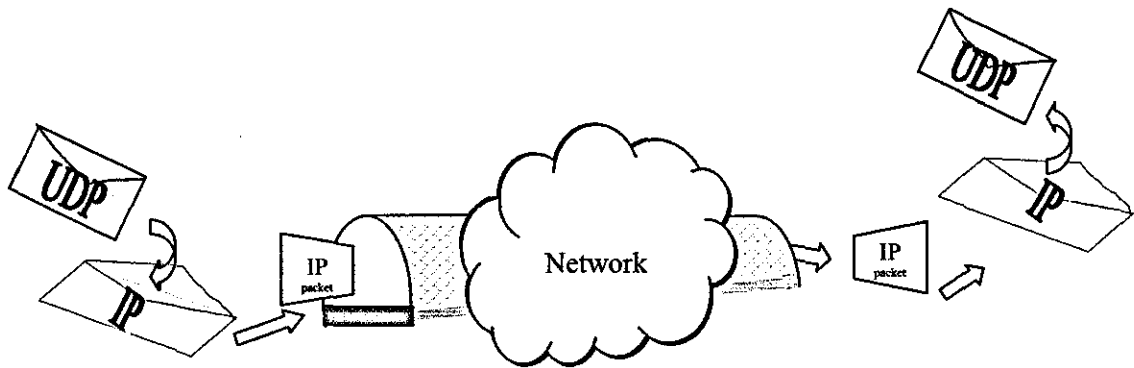


Figure (2.2): The VoIP Tunnel.

## VII. Layer I and II

In the VoIP protocol suite, the lower layer protocols and interfaces differ depending on the placement of the router. Lower layer interfaces between user nodes and routers can be V.90, HDLC, and LAPM. But between the routers, these protocols are not executed. They are replaced with another L\_1 system, typically a DS1, DS3, or SONNET trunk, and the two ATM layers: (a) ATM, and (b) the ATM Adaptation Layer (AAL). The replacements of these lower layers allow the use of more efficient and powerful technologies between the routers.

### 2.1.3.1 Main VoIP Protocols

#### I. Integrated protocols:

These are protocols which are not “stand-alone” specifications. They rely on many other supporting protocols to complete their operations. The current two protocol suites which support VoIP are:

- (1) The **H.323** recommendations published by the ITU-T.
- (2) The **Media Gateway Control Protocol (MGCP)**.

H.323 is the standard protocol used for on-line multimedia over networks. It is a set of standards for packet-based multimedia networks. H.323 was defined by the

International Telecommunications Union - Telecommunications (ITU-T) in order to enable VoIP services to connect to traditional circuit switched voice networks.

H323 is the standard for sending voice (audio) and video using IP on the public Internet and within an intranet. It is a recommended signalling protocol by the ITU-T. It allows different elements to communicate with each other such as terminals, gatekeepers, gateways, multipoint control units, and proxy servers. It doesn't only allow VoIP but also video and data communication. H323 can carry audio codecs such as G.711, G.722, G.723, G.728, and G.729 whereas for video it can carry H.261, and H.263.

Audio Apps	Video Apps	Terminal control and management				Data Apps
G.711	H.261	RTCP	H.225.0	H.225.0	H.225.0	T.120
G.729	H.263		HAS	Call	Control	
G.723.1				Signalling	Signalling	
RTP						
UDP				TCP		
IP						
Link Layer 802.3						

**Table 2.1:** Overview of the H.323 Protocol [Coulter 1999].

## **II. Module-based protocols:**

Unlike H-323 which is a complete integrated protocol suit. SIP (Session Initiation Protocol) is just a module that sets-up the multimedia connection between two or more parties. However it is designed to inter-work with all the Internet

applications. SIP would reside on top of the UDP protocol in the network layers layout.

#### **2.1.3.2 Elements of VoIP connection:**

In order to establish a successful VoIP connection, the proceeding elements have to be established:

1) **H.323 Terminals.** These are clients that initialise a VoIP connection. H.323 terminals support real-time, 2 way communications with other H.323 entities. They include at least one voice codec such as G.711, G.723, and G.729.

2) **Gate keepers.** The Gatekeeper manages H.323 zones and controls the H.323 activities on the IP-based network. It also manages logical collections of devices.

Gatekeepers nevertheless are not a mandatory entity in the H.323 network.

Gatekeepers are usually implemented on a PC, whereas Gateways are often based on proprietary hardware platforms.

The Gatekeeper plays three main roles.

- a) Address translation service. To use names instead of IP addresses.
- b) Admission control. To deny or allow some hosts or some users.
- c) Bandwidth management.

3) **Gateways.** These are points of reference for the conversion between the LAN and the PSTN (Public Switched Telephone Network). They provide two-way, real-time communications interfaces between the IP network and the Telephony network. In other words, the Gateway serves as the interface between the H.323 and non-H.323 network.

For example when converting from an ISDN network to IP network, the gateway has to convert a TDM signal with interleaved bytes of a PCM signal to packets of the IP format. Both standards are defined by the ITU-T, the standard for ISDN is

H.320, whereas H.323 is the standard for voice, video etc. across IP networks. The concept of all gateways can be seen in figure 2.3. The H.323 standard is explained in table 1: all allowed codec-schemes are shown, with their supporting layers (of the OSI-model).

H.323 uses UDP for the audio information, but TCP for the signalling data (to have a reliable connection). The choice of one gateway or another has to be made based on which type of compression scheme is used (G.711, G.722 ...), how many lines are needed and how much it costs.

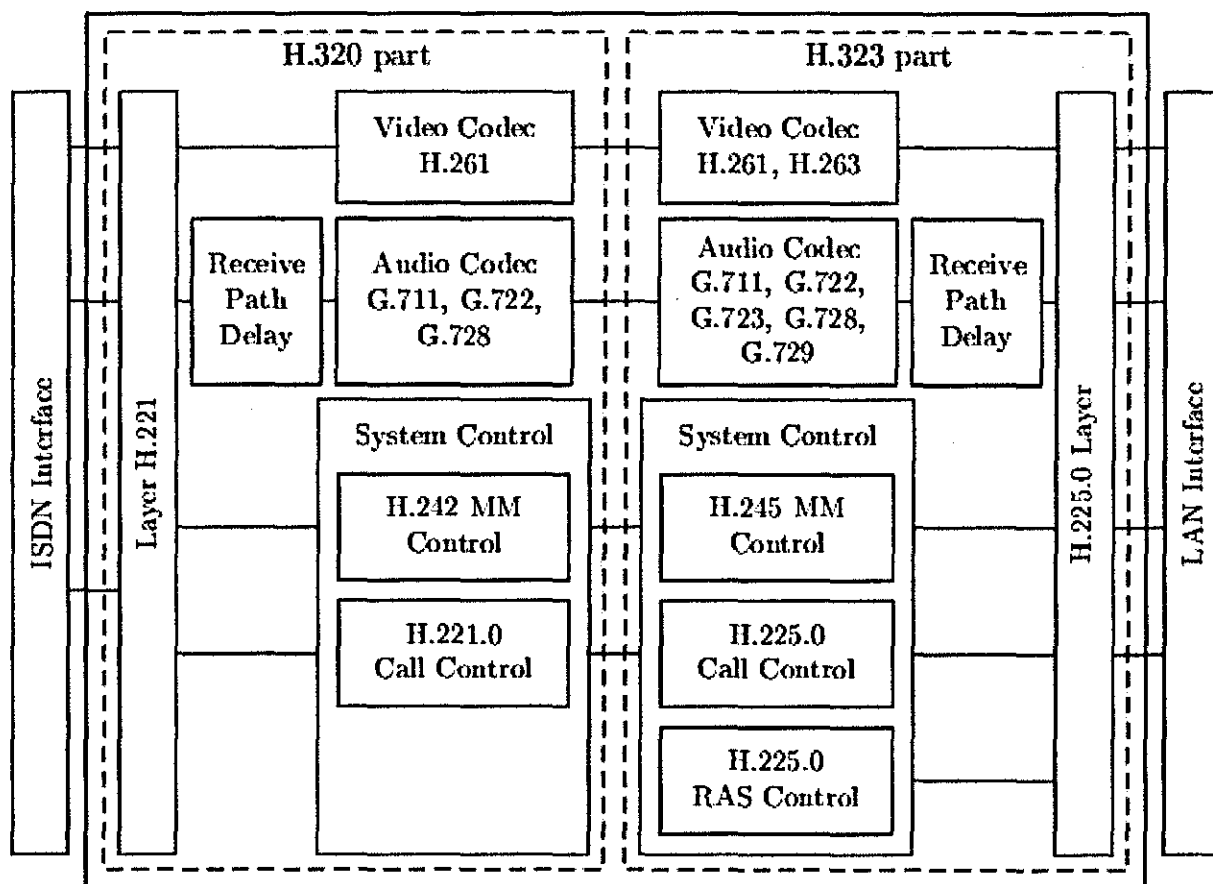


Figure 2.3: H.323 to H.320 Gateway Architectural Overview [Karapetkov 1999].

A 3com study [Cox 1998] reveals that the telephone/gateway approach provides significantly better performance than a PC/router approach.

The Gateway consists of three components:

- I. The **Media Gateway (MG)**.
- II. The **Signalling Gateway (SG)**.
- III. The **Media Gateway Controller (MGC)**: is the over all controller for the system. That is, it controls the Media Gateway (MG) and the Signalling gateway (SG).

4) **MCU (Multipoint Control Unit)**. Facilitates and enhances conference ability. It offers conferences between three or more endpoints. MCU can be stand-alone devices, or be integrated into a terminal, a Gateway or a Gatekeeper.

An MCU contains two parts:

1. **Multipoint controller (MC)** that handles the signalling and control messages necessary to set-up and manage conferences.
2. **Multipoint processor (MP)** that accepts streams from endpoints, replicates them and forwards them to the participating endpoints.

When the MCU is implementing both MC and MP functions it is referred to as centralised MCU. A decentralised MCU would handle only the MC functions, leaving the MP functions to the endpoints.

5) **Backend**. The Backend may be used by the Gateways and Gatekeepers to provide support functions such as billing, database management, routing and address resolution.

#### **2.1.4 Factors affecting VoIP transmission:**

There are three key factors that must be evaluated in order to have a VoIP transmission:

- (a) **Packet delay:**
  - 1) How long it takes to send the traffic from the sender to the receiver.
  - 2) The Jitter or variation in time of the arrival of the packets at the receiver, depending on the traffic.

(b) Bandwidth requirements: This should include bits required to represent speech signal as well as the overhead headers that are used to support the signals. For a minimum, this includes the L\_2 header, the IP header, the UDP header, the L\_7 header, and the headers created by the voice coder.

(c) Computational effort: The expense and complexity involved in supporting services to the audio application. Where millions of instructions per second (MIPS) are required to support the operation, as well as the amount of memory needed; that is the complexity and expense of the voice coder/decoder (codec).

VoIP applications are considered to be bursty applications, since they are transmitted over a VBR-based data network (Variable bit rate). They transmit and receive traffic asynchronously (at any time with periods in which nothing is sent or received). Hence do not require a constant and continuous allocation of bandwidth, such as the CBR-based telephony network. Where application using CBR schemes require constant and continuous (or nearly so) allocation of bandwidth, such applications are said to be non-bursty.

Nevertheless, to accommodate voice traffic efficiently over data networks, the VBR behaviour has to be smoothed to approximate that of CBR in order to permit conventional CBR digital-to-analogue operations to take place while transmitting voice.

Three obstacles must be overcome if the Internet or intranet is to be effective bearers of speech:

- (a) Latency must be reduced to avoid packet loss, and to provide quality close to that of the public switched network (PSN). In [Mier 1999] concluded that once latency drops below 90 to 80 ms, a person cannot tell the difference between the vendors and the public switched network, or between the different vendors.
- (b) Jitter must be reduced.
- (c) Traffic loss must be improved.



- The VoIP designer should pay attention to: (a) Buffer size. (b) Packet size. (c) Packet loss rate.
- H.225 protocol is used to control the RAS channel (**R**egistration **A**dmission **S**tatus) which is the channel used for communication between the end terminals and the gatekeepers used to arrange for terminal to join in or depart and also to arrange for the bandwidth needed and so on (Figure 3.4). RTP carries the actual audio media, RTCP manages and controls RTP and includes periodic status and control messages. All of the aforementioned are carried over the unreliable UDP layer because it would not make sense for it to be retransmitted: should a lost sound fragment be retransmitted, it would most probably arrive too late to be of any use in voice reconstruction.

Both H.245 and Q.931 are used to initiate a call. Thus the TCP transmission protocol is used to establish the VoIP connection setup through the two aforementioned protocols. This ensures that the important messages get retransmitted if necessary so they can reach the destination.

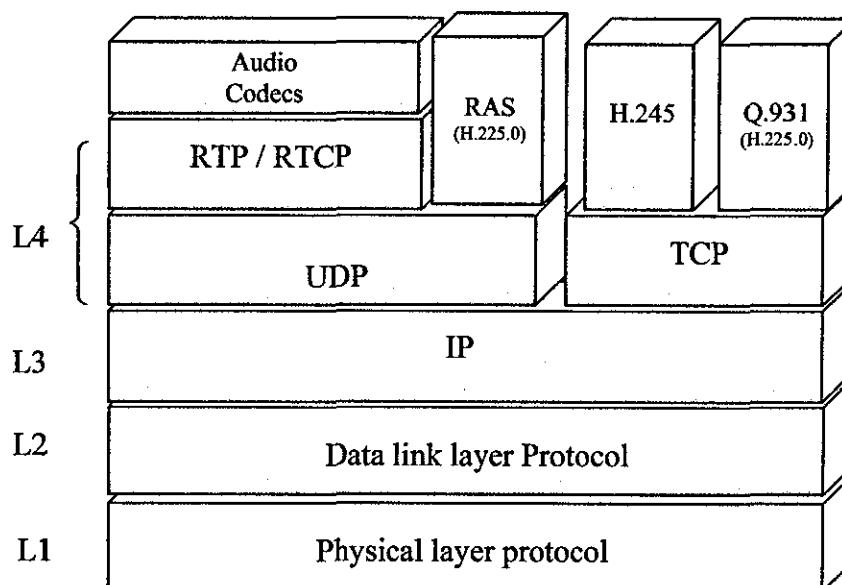


Figure 3.4: The VoIP Layered protocol Suite

- Quality of service is achievable through four parameters:
  1. Reliability.
  2. Delay.
  3. Jitter.
  4. Bandwidth.

These parameters differ according to the required quality of service of certain applications. Email, for instance, requires high reliability since it should be received intact without any errors. However in voice application, reliability is not set as high since running error-check algorithms and other procedures which should ensure the reliability of the voice conversation would greatly degrade the voice traffic and would doom the conversation to failure due to increased delay.

#### **2.1.5 Factors affecting voice quality:**

➤ **CODEC:** Compression techniques were developed to allow reduction in the required bandwidth while preserving voice quality. Different compression schemes can be compared using the following four parameters:

- 1) **Compressed voice rate:** When compressing voice from 64 Kbps down to a certain bit rate.
- 2) **Complexity:** The higher the complexity of implementing the codec, the more CPU resources are required.
- 3) **Voice quality:** Compressing voice in some CODECs results in very good voice quality, whilst others cause a significant degradation.
- 4) **Digitising delay:** Each CODEC algorithm requires different amounts of speech to be buffered prior to compression. This delay adds to the overall end to end delay.

➤ **Latency:** The time a single voice packet spends in the network.

- ❖ ITU-T G.114 has recommended that the maximum one way latency would be 150 mSec to achieve high quality voice. Beyond 250mSec round-trip latency, callers start feeling uncomfortable holding a two-way conversation and usually end up talking over each other.
- **Jitter:** The regularity with which voice packets arrive. It is the variation in the delay or the inconsistency in the delay or latency for the packets sent. Latency is the delay of a packet on the net.

- ❖ Since the receiving decompression algorithm requires fixed spacing between packets, a jitter buffer is implemented within the receiving gateway.
  - ❖ A Jitter buffer could also fix any out-of-order errors by looking at the sequence number in the RTP frames. Moreover, the buffer can reduce the effect of the variation in delay on the voice quality.
- **Packet loss:** Packet loss can occur for different reasons: overloaded links, excessive collision on a LAN, physical media errors, and other reasons. Audio Codecs take into account the possibility of packet loss. They perform fine-tuning voice over packet services among other functions that make occasional packet loss unnoticeable to users. However if the percentage of lost packets exceeds 5% of the packets, then it will start to be a problem in which many Codecs cannot hide.

#### **2.1.6 Voice quality measurement:**

The majority of research and studies about VoIP are concerned with enhancing VoIP to have voice quality equivalent to that of circuit switched networks. Thus there has to be means of measuring voice quality. At the standards bodies level, ITU has introduced P.800 (MOS), P.861 (PSQM), and BT's (PAMS).

The P.800 (MOS) Mean Opinion Score standard involves deriving grading scores based on opinions of mixed groups of men and women listening to pre-selected

voice samples over the desired transmission media. The scores are from “1” to “5” where “4” is considered toll quality.

The P.861 (PSQM) Perceptual Speech Quality Measurement is an automated algorithm which tries to derive scores that correlate with the P.800 (MOS) scores.

The (PAMS) Perceptual Analysis/Measurement System was suggested by BT (British Telecom) as an alternative to P.861 (PSQM), since the latter recommendation was initially designed for circuit switched networks, and thus does not take in to account some of the factors that effect packetised voice networks, such as jitter and frame-loss. According to BT, tests that have been conducted using automated (PAMS)-results and have shown good correlation with the manual (MOS) results [Rix 2000].

## **2.2 Related work**

Some work has been conducted on the subject of application detection and application grading which, in a way, contributes to our study. The following discusses some related work having a direct effect on this current research.

### **2.2.1 Application detection**

In a research by [Bharadia 2001] a scheme was developed to detect the type of application running on the network. The main objective and benefit of this scheme is to enable the network administrator to uniquely identify the applications running over the network. Furthermore, particular parts of a network can be monitored so that “illegal” application usage can be detected and probably refused connection. This scheme can be very useful in a business or academic environment where network users are expected to observe certain rules of conduct towards the use of their network. Nevertheless, some users may misuse their network access accounts for entertainment purposes or commercial gain. Consequently, the use of an application detector could come in handy to control the use of the network access.

The application detection scheme uses two aspects in order to identify the application which is currently running on the network. The first is by identifying

the port number of which the application is using, where certain port numbers are usually used by certain application types. This first aspect addresses the type or the group of applications which the currently running applications belong to. This in it self limits the possibilities of the suspected applications to few number of applications which often use the indicated port number. The second aspect of the application detection scheme observes the network parameters such as Packet size, the distribution of packet size over a given time interval and frame duration.

### **2.2.2 Application grading**

Application grading could be used on various applications and in different types of networks to predict the performance of a designated application. Each application demands individually suited requirements of network resources. The lack of some of these resources results in the application quality being affected accordingly. These resources are identified by three factors; variant delay, fixed delay, and packet loss.

The grading takes place by measuring the performance of a given application after subjecting it to different settings of network load parameters. Subjects of group of people are then asked to assess the quality of the application on a quality scale from 1 to 5. Grading can also be achieved by only measuring the time elapsed as in the case of FTP or Web-browsing.

Application grading was efficiently used in MaGNAQoS, which was a project jointly funded by BT and EPSRC, undertaken at Loughborough University. Its objectives included the development of techniques to predict network application performance before a session of the application was started. The idea was to give the user an idea of how the application was predicted to perform under whatever conditions of load that existed at the time. This work represents a different approach to achieving improved application performance; instead of changing the network infrastructure, one is attempting to change the way in which users access the network [Bharadia 2001].

If a user was informed by the system that the performance of the application was likely to be poor, he/she may decide to abandon the attempt and perhaps try again later when the network was more accommodating. This would avoid the addition of more traffic to an already busy network from futile attempts at starting the application and possibly ease frustration on the users' part.

### **2.3 Conclusion:**

In this chapter an overview on the Internet and its characteristics which pave for the use of Voice over IP was given. The topic of VoIP was outlined along with the quality of service issues related to it.

The following chapters of this thesis deal with the issue of the QoS of VoIP applications from the point of view of the application performance, in particular the quality of sound rather than usability.

## *Chapter 3*

### Methodology to Determining QoS at the Hop Level

#### **3.1 Introduction**

**D**eciding upon a research method as well as a specific technique under that method is vital for the reliability of this approach and its findings, and is essential for achieving as close results as possible to the actual practical situation. Furthermore, upon choosing a research method, the techniques and metrics which will be used under that method have to be carefully selected.

In order to determine the method to be adopted, the different methods and their pros and cons have to be outlined, and hence the availability of the elements of these methods along with their outlined features should give a clear idea as to which proposed method best suit the intended study. Moreover, the metrics or parameters which are to be used under the adopted method have to be correctly identified in order to achieve optimum results.

A brief overview on the three possible method types; Analytical, Simulation, and Measurements is discussed. The method used during this research as well as the metrics used through the different experimental approaches and the logical

sequence in which the main stages of this research were carried out is explained in this chapter.

### **3.2 Main types of methods**

Generally speaking, there are three main methods of evaluating system performance;

- a) Analytical, i.e., through mathematical models.
- b) Simulation, i.e., building an executable model of the system (not the system itself) and taking measurements on the model.
- c) Empirical or experimental, i.e., measurements on an original system.

#### **3.2.1 Analytical technique**

The Analytical technique is an ideal evaluation technique when the proposed system or the improved version of the product to be evaluated is a new concept that does not exist in a real life scenario, which makes it impossible to be evaluated through measurements technique.

The high cost budget usually associated with the use of the measurements technique is often a concern that is eliminated by the use of the analytical technique instead. In the measurements method, a real physical system has to be supplied, including all of the elements such as of routers, switches, end nodes and their required applications etc. Whereas in the analytical technique, mere equations and the parameters involved, take the place of the costly physical



network infrastructure, therefore making the analytical approach appear the best optimal choice of all research methods in this respect [Garson 2002].

When it comes to time-limitations where there are restraints imposed on the evaluation time period, the analytical technique stand out as a fast results acquisition tool in comparison to the simulation or the measurements techniques. Simulation takes longer, whereas measurements not only take longer but the time required to reach the results may vary a lot.

*The analytical technique is advantageous in explaining the effects of the various parameters involved in the study on the over all result. The main objective of performance studies in general is either to find the best value of the studied parameter or to compare different alternatives. Analytical modelling is better in this respect, where the use of various alternative parameters is easy to examine in order to give a clear picture on their interaction and relation with each other during the experiment, as well as the effect they have on the final concluded result.*

One of the drawbacks of the analytical technique is that it requires a lot of assumptions and simplifications which in return would reflect a variant level of accuracy of the outcome results. Moreover, the difficulty in generating the necessary models accurately is an issue that affects the consideration of the analytical technique altogether.

### 3.2.2 Simulation technique

In general, the simulation technique is advantageous over the measurements technique when it comes to cost. It can be very economical as well as uncomplicated to control all network aspects in comparison with the empirical method, especially when simulating a large scale network.

Simulation methods are not only economical but may also be less time-consuming than empirical methods. Furthermore, there are readily available tools as the researcher may not always get complete access to an operational system to perform the research, whereas there may not be such a restriction on the simulation tools, as it is the case in a large scale operational system of which some researchers do not have access to but are interested in carrying out studies on certain elements of this system. As for proposed future systems the simulation technique is useful if the proposed system does not exist in real-life, thus the programmer can accommodate all the hypotheses he wishes in his/her simulation tool to represent the proposed system.

In comparison with analytical modelling, the simulation technique requires fewer assumptions and can accommodate more details of the proposed system and, therefore tends to offer more accurate results than that of the analytical approach.

Despite these advantages, the simulation methods may not always provide precise results as the model may be based around certain assumptions. Accordingly, the simulating method, in some cases, does not reflect the real life situation. The simulated network and its condition depend primarily on the simulation program

to be as close as possible to the real physical network and its behaviour. The model may be built for specific operational scenarios as it may not be practically possible to model all scenarios in a single model. Some aspects of unexpected network behaviour which may have been overlooked or not taken in to account in the simulation program may hinder the validity of this approach altogether. Therefore the simulation method remains a simulated view of the real life situation but does not necessarily reflect it adequately.

### **3.2.3 Measurements technique**

The measurements technique is rather costly to implement on a large scale network. Nevertheless, it is an exact confirmation of the networking scenario since it is implemented in a real network environment. Aspects that may have been overlooked by the programmer in the simulation program are likely to be detected in the measurements approach. This feature in itself gives the measurements approach an advantage over the simulation approach. Moreover, the use of real measurements on a real physical network makes the results more solidly supported when compared to those of the analytical or simulation approaches where the final results remain to be theoretical.

In the case of this study, the availability of a physical lab-environment network infrastructure; Cisco routers, switches, and necessary nodes, instruments and applications, made it possible to run all the test stages of this research in a similar environment to that of the networking system which already exists in the real life internet situation. Thus, the measurements technique was chosen to be the adopted

approach for this research in order to keep it as close as possible to the real network scenario.

### **3.3 Parameters and metrics**

In order to determine which approach to adopt in any experimental research work, the parameters or elements which affect the application in hand and its performance need to be clearly identified and investigated. Subsequently, these parameters would then be the variables which are meant to be measured to achieve consistent results [Jain 1991]. From the previous chapter, the network parameters which could have an effect on a real-time network application are; variation in delay and packet-loss. Thus the study is mainly concerned with those parameters.

In essence, delays experienced by packets are a combination of both switching delays and link delays. The latter are dependent on the link speeds, packet lengths, loading and priority, network queues. Switching delays however, depend on the state of the processor within the node, its buffer-size, interrupt priority structure and its processing power. When it comes to real-time network applications such as those transmitting audio and video, delay variation is also an important measure of performance.

The other parameter which plays a key role in performance quality is Packet-loss. Packet loss must also be considered as a performance measure in packet switched networks. Congestion often causes packet loss due to buffer capacity overflow at

the nodes but also from what is termed *structural resource imbalance*. If a high-speed communication link is attached to a low power network client, the CPU within the client may not be able to service the incoming packets fast enough. Some packets may therefore be lost [Tanennbaum 2003]. Network faults and the transmission medium may also introduce packet loss although the latter of these two causes is uncommon and rare.

As we have seen in the previous chapter, Voice over IP is dependant upon the above network parameters. It is important to point out though, that in this research the issue of voice quality has more consideration than usability. This aspect is important in determining whether fixed delay is considered within the affecting parameters or not as will be discussed in the upcoming chapters, where fixed delay has almost no affect on voice quality in a real-time voice application although it may have an affect on its usability. Consequently, variant delay and packet loss are the parameters which this work took into account during the empirical grading of voice quality.

### **3.4 Metrics measurements**

This research uses a grading algorithm established by an earlier work [Oliver 2000] which was used to empirically grade the quality of certain applications used in that study. The same grading method is used in our study to determine the quality of Voice over IP applications over a designated network.

The grading algorithm in [Oliver 2000] was established to grade the quality of an application according to the general condition of the entire network. Thus this algorithm is meant to be used as an assessment tool prior to executing the application over the network to give the user an idea, in advance, of how that application will behave.

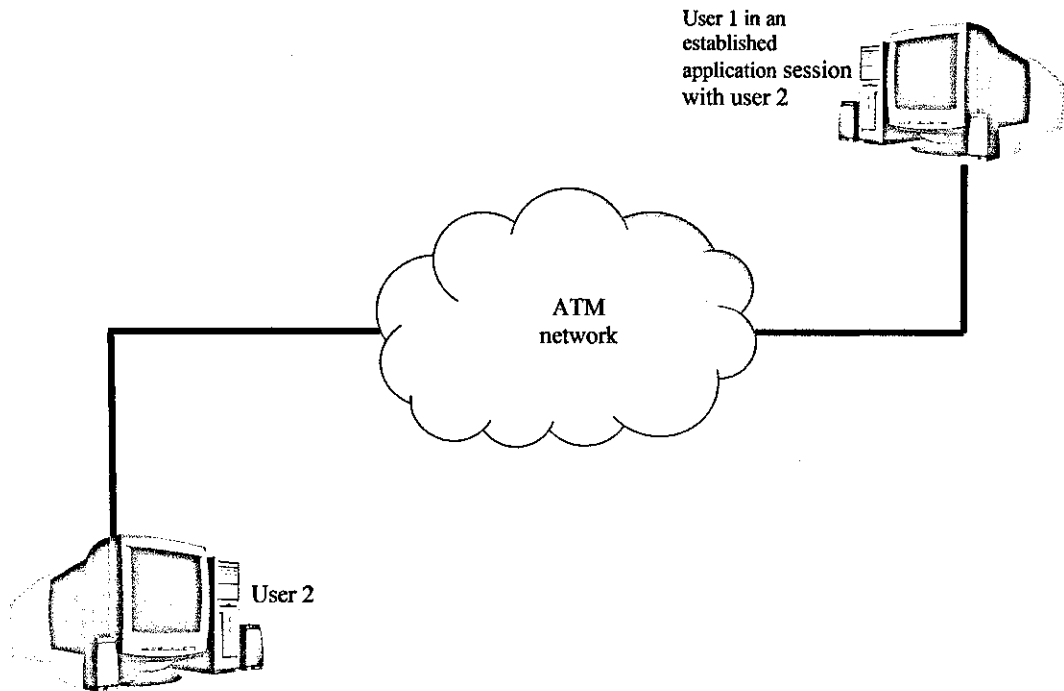
The study in [Oliver 2000] was conducted in a controlled lab environment, where the network condition, namely; constant delay, delay variation, and packet-loss, was assigned to an ATM network. Empirical grading was then assigned to the application which was established between two end-stations, according to the change in the controlled network parameters (Figure 3.1). The grading was then conducted on the following three applications;

- a- Web-browsing,
- b- Video-conferencing,
- c- Virtual-Reality games.

The experiments were devised to achieve the following objectives,

- 1) Provide the user with an indication of the condition of the network from the application perspective so that the user does not waste time or resources trying to execute the application over a poor connection. Thus the user may choose to run the application at some other time when the network condition is better.
- 2) Reduce the load on the network as the users refrain from executing their applications that are expected to provide a poor quality of

service due to network condition. Thus no additional load is added to the currently overloaded network.



**Fig 3.1:** The previous study was aimed at grading the application quality according to the network condition as a whole which is dealt with as a black box without going into detailed path elements.

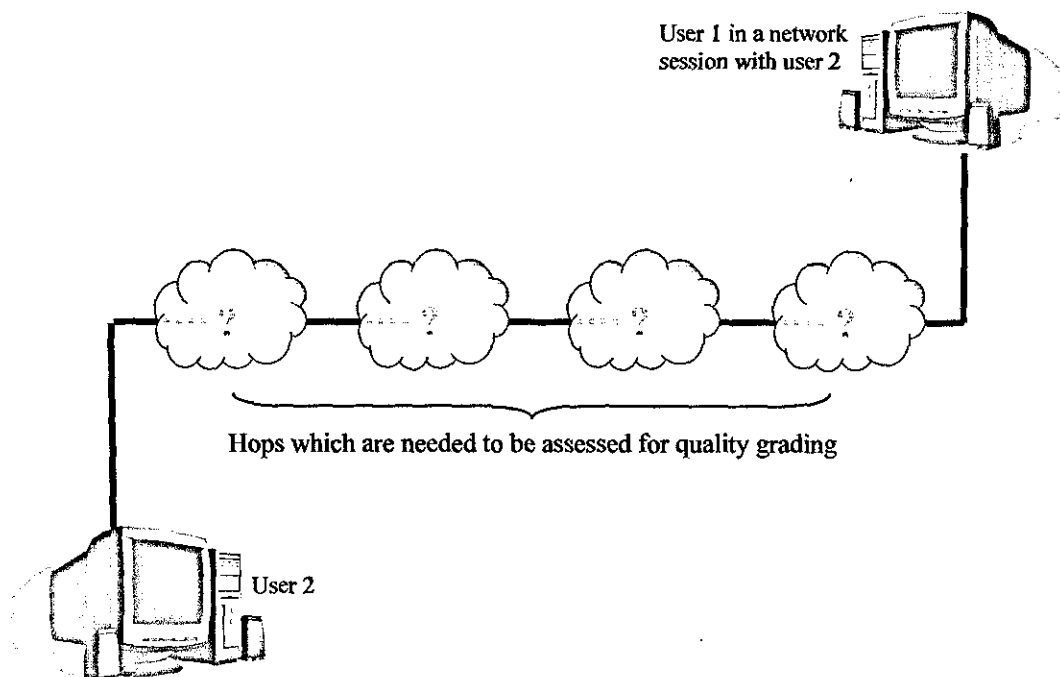
In our research, the grading method established in [Oliver 2000] is applied in a different way to achieve different goals. The grading method in the previous study was based on the overall accumulative network parameters (traffic delay and packet loss) that affect the overall end-to-end connection. Therefore the

application is affected by the general network condition. The current study aims to determine exactly where the adverse network conditions exist on a multi-hop path. It seeks to determine the individual hops along the network path which are responsible for the degradation of the application quality rather than dealing with the path as one unit. The grading of the application quality is not intended for itself but for revealing the incompetent hops on the network. This is accomplished by applying the application grading method on the network parameters that are measured at every hop along the path.

After establishing means of acquiring network parameters (delay variation and loss) for every hop in the network path, the grading method is then used as an indicator of network condition from the application perspective rather than a predictor of an application's behaviour, as was intended for in the previous study.

Once the quality of the network for VoIP applications has been determined, the algorithm treats every hop as a separate network repeating the same process for each hop on the network (figure 3.2). Consequently, rather than determining the overall network condition (figure 3.1), the condition for each hop is assessed here, resulting in an assessment of the quality of service provided by a designated hop to a certain application over a given network.





**Fig 3.2:** The work of this thesis aims to reveal the reliability of every hop in the network by exploiting the grading method and treating every hop as a network by itself.

Accordingly, the grading method is utilised, at every hop along the network route, to assess the conditions that should affect the quality of the Voice over IP applications. In general, this technique can be applied to other real-time multimedia applications.

### 3.5 Establishing grading for VoIP

In the first stage of this study, work was conducted in a controlled network environment in order to establish a method to grade the quality of service for

Voice over IP applications. The use of an LNE (Loaded Network Emulator) [Oliver 1999] was sought so as to emulate real network conditions in a controlled manner. Hence the network parameters, which are fixed delay, variable delay, and packet loss, are known before hand to precisely calibrate the grading algorithm on real-time voice applications. The change in voice quality at the end-terminal is monitored while changing network parameters. Consequently, different empirical grades are assigned for the VoIP application according to the enforced network parameters.

### **3.6 Measuring network parameters at hops**

The second stage of the study explored the possibility of measuring the network parameters at different hops to determining the hop-wise quality of service for VoIP applications. Once the network parameters for the hops are known, the grading algorithm could then be used to determine the quality of service that every hop can offer to VoIP applications. This also gives an indication of the hop that adversely affects the quality of service of the VoIP session .

The experimental work was conducted on active networks as well as current traditional IP networks. As for the latter, two approaches were considered; extracting network parameter at each hop using the MIB (Management Information Base) using SNMP (System Network Management Protocol), and finally using a prototype tool developed to assess such information at each hop along the network connection.

### 3.6.1 Active Networks

Active Networks [Psounis 1999] were used at the start of the work to determine the feasibility of measuring the parameters at the hops with a greater degree of freedom and a finer resolution. Active networks can offer such facilities due to their node programmable structure.

Two approaches of active network techniques were considered to measure the network parameters values which were to be used to reveal the quality of service at the designated hops;

- a- Time-stamping. Where packets are sent to the destination node and time-stamped at each hop on the way in order to reveal the delay experienced at those hops.
- b- Active Ping. Where each hop sends series of ping messages to the proceeding hop. Statistics of the series of pings which are measured at each hop are then collected at the source end-node. The use of locally calculated statistics was aimed to overcome the time synchronisation issues. This problem occurs where system clocks at various hop devices do not have the same precise time reading, which cannot be depended upon to accurately measure the delay parameters which are experienced at the local network path between every two neighbouring hops. Generally speaking, the router state does not change in response time-scale of Active-ping reply. If however the router state changes fast before the reply then there is room for error. On the other hand, the statistics which are calculated at the hops are based on series of pings. Thus if the router-state

changes before some ping replies that will slightly affect the overall statistics.

### **3.6.2 Management Information Base**

A MIB “Management Information Base” is one of the means that could be used to extract information from the network devices which are under a certain administrative network management [Stevens 2000]. Network administrators often use it to acquire databases that could give them a view on how their local network domain is behaving. The use of MIB to acquire the network parameters of the hops was also studied. The purpose was to exploit the database available at the routers on a network path to determine the quality the designated hops will offer to a VoIP session. The ambiguous argument though, is whether or not the use of a MIB would enable data acquisition from network devices which fall under various different network administrative domains. That is because of the possibility that each administration domain management may assign their own password to prevent intruders from accessing their network routers.

### **3.6.3 Prototype tool**

Following the two aforementioned experimental approaches; Active networks and Management Information Base, a third approach was explored and tested to acquire the information needed from each hop along a network route, whereby a

prototype tool was designed to perform the experiments for the two aforementioned approaches.

By performing certain calculations on the information received from both “Traceroute” and “Ping” utilities which are directed at the opposite end-terminal and the in-between hops, the required information of the designated hops are then extracted, with certain limitations at some hops, which will be explained and discussed later on.

The tool can be used to overcome the different management domain issue which could be an obstacle in accessing the data bases of the network devices along the designated network route. In comparison with the two previous approaches described in sections 3.4.1 and 3.4.2, the prototype tool works on standard IP networks and is not confined to Active networks alone. Moreover, permission from a network administrator may not be required while using this tool whilst access privileges are need to access the router’s database.

### **3.7 Conclusion**

In any experimental research work, a method by which the research will be carried out has to be identified. In this chapter the empirical measurement method which is adopted for this study was highlighted. Furthermore, the outline of the main stages of research were discussed and explained in this chapter.

It was pointed out that at the end the prototype tool approach was the technique which this work has adopted to assess the designated hop's performance. The tool was tested and achieved good results which are presented in chapter 7.

The experimental approaches explained in sections 3.4, 3.5.1, 3.5.2, and 3.5.3 are the main stages the research of this thesis and are described further in chapters 4, 5, and 6 consecutively.

## *Chapter 4*

### **The Grading Algorithm**

#### **4.1 Introduction**

**T**he main objective of this research is to determine the quality of service of a given real-time network multimedia application at every hop along the network path with the application session is utilising. Accordingly, a grading algorithm has to be established to identify the various levels of QoS for this application. Consequently, the QoS for the application at each hop will be graded according to this algorithm.

In order to determine the quality of service for the required application and categorise this quality of service by assigning a grade for it, the grading algorithm must take into consideration the application being considered. This is because applications, in general, differ in their network resource requirements and the parameters which play a role in the enhancement or degradation of their performance.

Real-time applications differ in their networking requirements from those of non real-time applications'. FTP, as a non real-time application, can tolerate delay since no specific demands of timing requirements are placed upon packet delivery, although performance, in general, can be improved if the delay is reduced. Moreover, different real-time network applications vary in their networking demands as well. While a typical Internet videoconferencing application, for instance, is mainly affected by packet loss and variation in packet delay, web browsing applications are affected by fixed delay as well as packet

loss and variable delay. Thus, the parameters that play a role in determining the performance of a certain application over the network have to be identified. These are used as metrics to determine the quality of that application over any given network.

The basic principle of developing a grading algorithm for an application is to calibrate the grading by establishing a controlled connection between two end-nodes and then monitoring the quality of service at these end-nodes. Variation in the controlled traffic is introduced on the hops that are spread along the network route. Furthermore, the relationship between the change in network parameters and grading of the quality of service has to be mathematically established. Thus, different grades are assigned according to the different traffic parameter combinations of delay and loss. Consequently, in the reverse approach, if traffic parameters can be collected over any given network, the quality of service of the application executing on that network can be determined a priori using the grading method parameterised by the collected network statistics.

The steps by which the grading method was developed are explained in the following sections.

## **4.2 Calibration of quality of service grading**

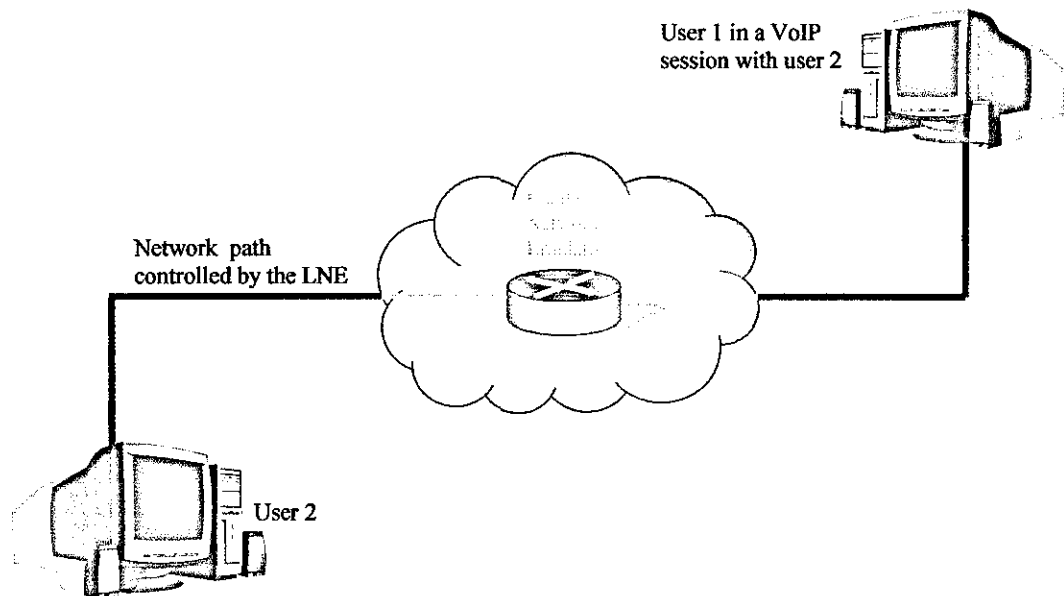
Knowing what the applications require is essential to pave the way to measuring their performance. Specifically, networking parameters which affect the application's performance have to be identified, and moreover, the extent of their



effect needs to be measured in order to have a precise categorisation grading of the quality of service for the application in question.

In this research, Voice over IP is used as the application for which the grading of the performance of hops was calibrated. To grade the quality of service, the procedure had to be performed on a controlled network set-up where the network traffic, as well as network elements, are controlled and altered as desired and there is no interference of the external network traffic. Microsoft NetMeeting<sup>®</sup> 3.01 was used as a voice application. The video send-and-receive options were disabled so as to limit the application to audio communication only. The auto-gain control was also disabled to ensure receipt of the original audio quality without any quality enhancements at the receiving-end. The voice codec used for the experiments was based on G.723.1. The codec streams are then passed via RTP to be carried over UDP.

Two end-users were connected through a network station that had a PC-based LNE (Loaded Network Emulator) executing on it to emulate real network conditions and to introduce the desired heavy or mild traffic parameters of loss and delay (Figure 4.1). This PC basically acted to cause a loaded network, but without adding additional traffic. The quality of service that it provides can be controlled by varying the above mentioned traffic parameters for a loaded network.



**Figure 4.1:** Implementation of the LNE (Loaded Network Emulator). It emulates real network conditions.

The Netmeeting<sup>®</sup> connection between the two end users was established through a congested network emulated using the LNE, as detailed in the next section.

#### 4.2.1 LNE (Loaded Network Emulator)

The LNE is used to emulate the load experienced on networks by setting up and altering the three main parameters that influence the quality of service for most applications over a given network. These parameters include fixed delay, delay variation and packet loss. The LNE is designed to drop or delay the packets that pass through it according to the user's desired parameters of network load. Hence it offers better accuracy of emulating loaded networks than traffic generators since

it does not generate or add any extra traffic to the existing network traffic, besides one can specify the exact parameters of loss and delay to be experienced on the network. In the LNE, packet loss and delay are applied according to Gaussian distribution [Oliver 2000].

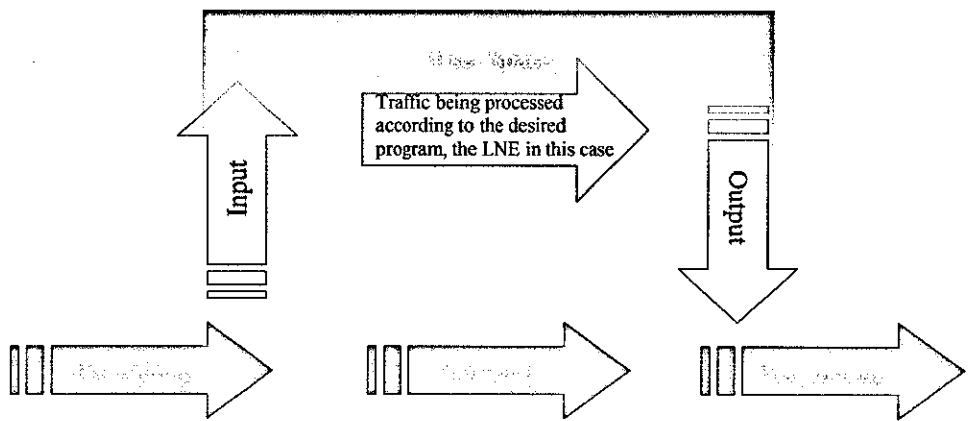
It was found that fixed delay has no effect on the quality of the sound, although it does affect usability. This was also established in earlier work [Oliver 1999] for video conferencing applications. The voice processing and communication components of these applications are similar in behaviour to other voice applications. Hence fixed delay was not considered in the later experiments. Nevertheless it must be noted that high levels of delay may affect VoIP usability.

In order to apply packet loss and delay variation to the network traffic passing through the router which is accommodating the LNE, IP-Tables were exploited to set-up LNE parameters. IP-Tables are commonly used to control the traffic that passes through a Linux based computer by applying IP packet filter rules in the Linux kernel. Different tables can be defined which contain several built-in sets of rules or chains to be enforced on the passing traffic. Furthermore users can also define chains or rules. In this case the rules are defined and enforced by the LNE, which resides in user space.

IP-Tables thus control the queuing strategy at the router. The queuing architecture of IP-tables is shown in (Figure 4.2). When there is no specific queuing strategy i.e. FIFO, the incoming packets at the "Pre-routing" queue are forwarded to the "Post-routing" queue through the "Forward" queue. Thus, the packets bypass any processing in the "User Space". However, if there is a change to the queuing strategy, or the packets need to be processed in a certain way, for example, if there is a need to time-stamp the packets, the packets have to be passed to the "User

Space" through the "Input" queue before forwarding them to the "Post-routing" queue through the "Output" queue.

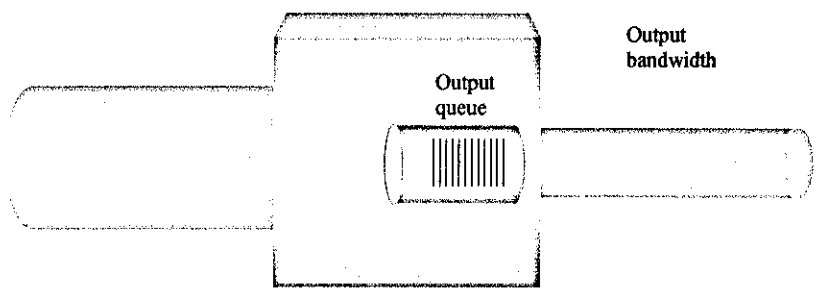
The IP-tables queuing strategy rule is thus altered to force network traffic to pass into user-space. Here, the forwarded packets are affected by the loss and jitter parameters introduced by the LNE. Packets that are not dropped by the LNE are passed on to the output queue.



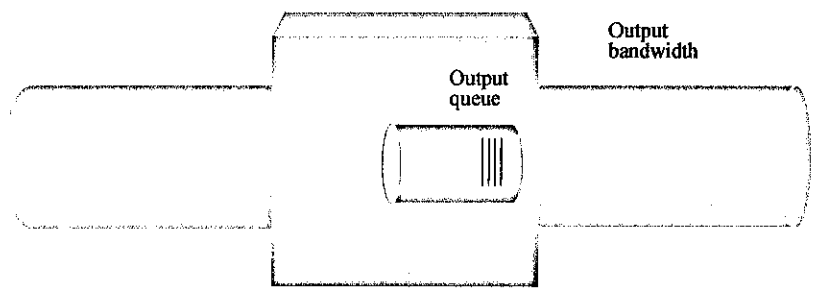
**Figure 4.2:** IP-Tables queuing layout.

The LNE simply drops packets according to the desired loss-percentage value. As for delay variation, this is introduced by controlling the output bandwidth that can increase and decrease between two specified maximum and minimum bandwidth parameters. Thus the number of packets in the output queue will increase when the bandwidth window gets narrower (Figure 4.3a), and decrease when the

bandwidth window gets wider (Figure 4.3b). The maximum size of the output queue was set to be large enough to ensure that no packets are dropped when the packets in the queue increase due to bandwidth window decrement. Hence variation in delay is controlled, emulating jitter over a real network.



(a)



(b)

**Figure 4.3:** Bandwidth window variation as a means to emulate jitter.

### **4.3 Quality classification**

In order to classify the quality of the application under consideration while changing network parameters, a semi-formal evaluation for voice quality was carried out in the laboratory on several subjects. Accordingly, five quality grade values were specified. These quality grade values of 5, 4, 3, 2, and 1 [Daumer 1982] relate to excellent, good, fair, bad, and poor respectively in accordance with various values of loss and jitter.

To ensure an unrecognised conversation which none of the subjects were familiar with, a talk-radio station was chosen as the source of voice transmission. The radio was connected directly to the PC's audio card through the mic-port. On the other end-user station, the subjects listened to the conversation transmitted from the source using headphones.

For each of the different parameters of loss and delay, the subjects were asked to fill in the classification form. This process was repeated for all the subjects. Accordingly, an average of the results of all the evaluation forms was calculated to determine a common agreed-upon classification according to the introduced network parameters.

### **4.4 Grading algorithm**

Through the monitoring and identification of quality grades categorised according to the changes occurring due to the effect of the varying low level network parameters, a mathematical model was developed that outlines the effect of

variation of the network parameters on the quality grade of an application using low level data statistics.

Since quality grading is a function of loss and jitter [Oliver 1999], a simple algebraic general form equation that can be applied to any loss or delay variation value to obtain a grading was developed. The significant of this equation is that it uses low-level statistics of the simple network parameters of loss and jitter. Thus quality grading "G" was set to be:

$$G(l, v) = f_0 + f_1 l + f_2 l^2 + f_3 v + f_4 l v + f_5 l^2 v + f_6 v^2 + f_7 l v^2 + f_8 l^2 v^2 \quad (4.1)$$

where;  $l$  = Loss  
 $v$  = Jitter  
 $G$  = Grade  
 $f$  = Constant coefficients

Equation 4.1 provides an empirical grading of application performance according to the change in network loading conditions. The application quality grading "G" is a function of packet loss "l" and variation in delay "v". Hence "G" value changes in accordance with the change in "l" and "v", thus the quality grading is affected by loss and jitter as reflected by equation 4.1.

The factors in equation (4.1) are constants and independent of network type and network topology, but depend on application type and traffic density.

Partially differentiating equation (4.1) with respect to "l" to eliminate packet loss, will produce an equation containing one variable:

$$\frac{\partial G}{\partial l} = f_1 + 2f_2 l + f_4 v + 2f_5 l v + f_7 v^2 + 2f_8 l v^2 \quad (4.2)$$

Second partial differentiation with respect to "l":

$$\frac{\partial^2 G}{\partial l^2} = 2f_{2l} + 2f_{5l}v + 2f_{8l}v^2 \quad (4.3)$$

Now, going back to equation 4.1 and by fixing "v" i.e. "v = v<sub>1</sub>",

$$G|_{v=v_1} = f_0 + f_1l + f_2l^2 + f_3v_1 + f_4lv_1 + f_5l^2v_1 + f_6v_1^2 + f_7lv_1^2 + f_8l^2v_1^2 \quad (4.4)$$

$$G|_{v=v_1} = (f_0 + f_3v_1 + f_6v_1^2) + (f_1 + f_4v_1 + f_7v_1^2)l + (f_2 + f_5v_1 + f_8v_1^2)l^2$$

$$G|_{v=v_1} = b_{0l} + b_{1l}l + b_{2l}l^2 \quad (4.5)$$

where

$$b_{0l} = f_0 + f_3v_1 + f_6v_1^2 \quad (4.6)$$

$$b_{1l} = f_1 + f_4v_1 + f_7v_1^2 \quad (4.7)$$

$$b_{2l} = f_2 + f_5v_1 + f_8v_1^2 \quad (4.8)$$

equation 4.5 can be written in a more general form:

$$G|_{v=v_1} = b_{0l} + b_{1l}l + b_{2l}l^2 \quad (4.9)$$

Where "b's" are constant coefficients.

The "b" coefficients in equation (4.10) can be evaluated by having 3 equations representing 3 different combinations of "G" and "l". However, the values of "b's" will represent only the given combination cases. Hence, we make a general



universal case representation, where “ $b$ ’s” can be found in an approximate way using the elementary curve fitting technique.

Differentiating equation (4.9) with respect to “ $l$ ”:

$$\left. \frac{dG}{dl} \right|_{v=v_1} = b_1 + 2b_2 l \quad (4.10)$$

Second derivative of equation (4.9):

$$\left. \frac{d^2 G}{dl^2} \right|_{v=v_1} = 2b_{2i} \quad (4.11)$$

Now, equating equation (4.3) and (4.11):

$$2b_2 = 2f_2 + 2f_5 v_1 + 2f_8 v_1^2 \quad (4.12)$$

To find the “ $b$ ” coefficients, the delay variation was fixed at certain value of “ $v$ ” e.g. ( $v=v_1$ ), and several combinations of “ $l$ ” and “ $G$ ” were taken experimentally. These values were applied to the Matlab<sup>®</sup> curve fitting toolbox to obtain the various  $b$  coefficients for the designated delay value.

The above procedure was repeated with a different value of “ $v$ ” i.e. ( $v=v_2$ ) to work out another set of “ $b$ ” factors.

The obtained combinations of “ $v_1$ ” and “ $b$ ” from the above were substituted in equation 4.12 to find the following two equations relating the factors “ $f$ ”

Substituting the new values of “ $b$ ” in equation 4.12, new combinations between “ $f$ ” and “ $v$ ” are obtained. Mathematically:

$$2b_{21} = 2f_2 + 2f_5v_1 + 2f_8v_1^2 \quad (4.13)$$

$$2b_{22} = 2f_2 + 2f_5v_2 + 2f_8v_2^2 \quad (4.14)$$

Where “ $b_{21}$ ” and “ $b_{22}$ ” are the factor “ $b_2$ ” in equations (4.9 – 4.12) for the first and second values of “ $v$ ” where ( $v=v_1$ ), and ( $v=v_2$ ) respectively.

Similarly, equation 1 is partially differentiated with respect to “ $v$ ” and the previous procedure reapplied but with a fixed loss variable “ $l$ ” this time to obtain another two equations relating the factors “ $f$ ”.

Partially differentiating equation (4.1) with respect to “ $v$ ”:

$$\frac{\partial G}{\partial v} = f_3 + f_4l + f_5l^2 + 2f_6v + 2f_7lv + 2f_8l^2v \quad (4.15)$$

Second partial derivative:

$$\frac{\partial^2 G}{\partial v^2} = 2f_6 + 2f_7l + 2f_8l^2 \quad (4.16)$$

By fixing the loss variable “ $P$ ” while using Polynomial Curve fitting “ $G$ ” would be:

$$G|_{\text{constant } l} = g_0 + g_1 v + g_2 v^2 \quad (4.17)$$

Where “ $g$ ’s” are constants.

Differentiating equation (4.17) with respect to “ $v$ ”:

$$\left. \frac{dG}{dv} \right|_{l=l_i} = g_1 + 2g_2 v \quad (4.18)$$

Second derivative:

$$\left. \frac{d^2 G}{dv^2} \right|_{l=l_i} = 2g_2 \quad (4.19)$$

Equating equation (4.16) and (4.19):

$$2g_2 = 2f_6 + 2f_7 l + 2f_8 l^2 \quad (4.20)$$

The Matlab<sup>®</sup> Curve fitting toolbox was again used to find the “ $g$ ” factors. Using two sets of experimental combinations of “ $G$ ” and “ $v$ ”.

The following two sets of “ $g$ ” factors were obtained at two different values of “ $l$ ”:

$g_{21}$ ,  $g_{11}$ , and  $g_{01}$  at  $l = l_1$

$g_{22}$ ,  $g_{12}$ , and  $g_{02}$  at  $l = l_2$

Substituting in equation (4.20) with the above values of " $l$ 's" and " $g$ 's":

$$2g_{21} = 2f_6 + 2f_7l_1 + 2f_8l_1^2 \quad (4.21)$$

$$2g_{22} = 2f_6 + 2f_7l_2 + 2f_8l_2^2 \quad (4.22)$$

In total, nine equations are needed to find the factors " $f$ ". With four equations ("4.13", "4.14", "4.21", "4.22") of factors " $f$ ", the remaining five equations are then calculated by applying five different sets of " $G$ ", " $l$ ", and " $v$ " to equation 4.1.

$$G(l_1, v_1) = f_0 + f_1l_1 + f_2l_1^2 + f_3v_1 + f_4l_1v_1 + f_5l_1^2v_1 + f_6v_1^2 + f_7l_1v_1^2 + f_8l_1^2v_1^2 \quad (4.23)$$

$$G(l_2, v_2) = f_0 + f_1l_2 + f_2l_2^2 + f_3v_2 + f_4l_2v_2 + f_5l_2^2v_2 + f_6v_2^2 + f_7l_2v_2^2 + f_8l_2^2v_2^2 \quad (4.24)$$

$$G(l_3, v_3) = f_0 + f_1 l_3 + f_2 l_3^2 + f_3 v_3 + f_4 l_3 v_3 + f_5 l_3^2 v_3 + f_6 v_3^2 + f_7 l_3 v_3^2 + f_8 l_3^2 v_3^2 \quad (4.25)$$

$$G(l_4, v_4) = f_0 + f_1 l_4 + f_2 l_4^2 + f_3 v_4 + f_4 l_4 v_4 + f_5 l_4^2 v_4 + f_6 v_4^2 + f_7 l_4 v_4^2 + f_8 l_4^2 v_4^2 \quad (4.26)$$

$$G(l_5, v_5) = f_0 + f_1 l_5 + f_2 l_5^2 + f_3 v_5 + f_4 l_5 v_5 + f_5 l_5^2 v_5 + f_6 v_5^2 + f_7 l_5 v_5^2 + f_8 l_5^2 v_5^2 \quad (4.27)$$

Equations 4.13, 4.14, 4.21, 4.22, 4.23-4.27 are reformatted in matrix form:

Equation (4.13)	.....	0	0	2	0	0	2v <sub>1</sub>	0	0	2v <sub>1</sub> <sup>2</sup>	f <sub>0</sub>	2b <sub>21</sub>
Equation (4.14)	.....	0	0	2	0	0	2v <sub>2</sub>	0	0	2v <sub>2</sub> <sup>2</sup>	f <sub>1</sub>	2b <sub>22</sub>
Equation (4.21)	.....	0	0	0	0	0	0	2	2l <sub>1</sub>	2l <sub>1</sub> <sup>2</sup>	f <sub>2</sub>	2g <sub>21</sub>
Equation (4.22)	.....	0	0	0	0	0	0	2	2l <sub>2</sub>	2l <sub>2</sub> <sup>2</sup>	f <sub>3</sub>	2g <sub>22</sub>
Equation (4.23)	.....	0	l <sub>1</sub>	l <sub>1</sub> <sup>2</sup>	v <sub>1</sub>	l <sub>1</sub> v <sub>1</sub>	l <sub>1</sub> <sup>2</sup> v <sub>1</sub>	v <sub>1</sub> <sup>2</sup>	l <sub>1</sub> v <sub>1</sub> <sup>2</sup>	l <sub>1</sub> <sup>2</sup> v <sub>1</sub> <sup>2</sup>	f <sub>4</sub>	G <sub>1</sub>
Equation (4.24)	.....	0	l <sub>2</sub>	l <sub>2</sub> <sup>2</sup>	v <sub>2</sub>	l <sub>2</sub> v <sub>2</sub>	l <sub>2</sub> <sup>2</sup> v <sub>2</sub>	v <sub>2</sub> <sup>2</sup>	l <sub>2</sub> v <sub>2</sub> <sup>2</sup>	l <sub>2</sub> <sup>2</sup> v <sub>2</sub> <sup>2</sup>	f <sub>5</sub>	G <sub>2</sub>
Equation (4.25)	.....	0	l <sub>3</sub>	l <sub>3</sub> <sup>2</sup>	v <sub>3</sub>	l <sub>3</sub> v <sub>3</sub>	l <sub>3</sub> <sup>2</sup> v <sub>3</sub>	v <sub>3</sub> <sup>2</sup>	l <sub>3</sub> v <sub>3</sub> <sup>2</sup>	l <sub>3</sub> <sup>2</sup> v <sub>3</sub> <sup>2</sup>	f <sub>6</sub>	G <sub>3</sub>
Equation (4.26)	.....	0	l <sub>4</sub>	l <sub>4</sub> <sup>2</sup>	v <sub>4</sub>	l <sub>4</sub> v <sub>4</sub>	l <sub>4</sub> <sup>2</sup> v <sub>4</sub>	v <sub>4</sub> <sup>2</sup>	l <sub>4</sub> v <sub>4</sub> <sup>2</sup>	l <sub>4</sub> <sup>2</sup> v <sub>4</sub> <sup>2</sup>	f <sub>7</sub>	G <sub>4</sub>
Equation (4.27)	.....	0	l <sub>5</sub>	l <sub>5</sub> <sup>2</sup>	v <sub>5</sub>	l <sub>5</sub> v <sub>5</sub>	l <sub>5</sub> <sup>2</sup> v <sub>5</sub>	v <sub>5</sub> <sup>2</sup>	l <sub>5</sub> v <sub>5</sub> <sup>2</sup>	l <sub>5</sub> <sup>2</sup> v <sub>5</sub> <sup>2</sup>	f <sub>8</sub>	G <sub>5</sub>

$$= \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \\ f_6 \\ f_7 \\ f_8 \end{bmatrix} = \begin{bmatrix} 2b_{21} \\ 2b_{22} \\ 2g_{21} \\ 2g_{22} \\ G_1 \\ G_2 \\ G_3 \\ G_4 \\ G_5 \end{bmatrix} \quad (4.28)$$

$$\text{Or: } \varphi F = A \quad (4.29)$$

Thus, factors “ $f$ ” can be found from:

$$F = \varphi^{-1} A \quad (4.30)$$

Accordingly, to find out the quality grade of the application, the measured loss and jitter values should be substituted in equation (4.1) along with the evaluated factors from equation (4.30)

#### 4.5 Conclusion

In order to determine the specific hop or hops which play a role in reducing the quality of a certain application over the network, this quality has to be graded and measured according to the application in hand and the network parameters which are affecting the hop during the session time. Applications differ in their behavior and therefore act differently when it comes to their networking demands and their manipulation by the varying network parameters. Accordingly, the grading measure of one application is independent and not applicable to another application. Since this research deals mainly with Voice over IP, a grading measure had to be established for this application specifically. This grading measure will be used later in the thesis to convey the quality experienced by VoIP applications at the examined hops.

In this chapter, an algorithm for the grading of an application according to the existing network performance was proposed and explained. Furthermore, the mathematical model by which the quality would be graded was discussed and analysed.

Once the values of loss and delay in each hop along the network is determined, the grading algorithm would then be utilized in the light of those values to determine the quality of service at each hop independently for the required application according to the grading algorithm discussed in this chapter. This hop by hop-basis quality determination using the grading algorithm discussed in this chapter is addressed in the following chapters along with the results outcome.

## *Chapter 5*

### **Approaches for Grading Quality of Service**

**T**o put the mathematical model discussed in Chapter 4 in to practice, a practical approach needs to be implemented in order to facilitate the means that can be used to acquire the statistical data which are required to be applied to this mathematical model. Two approaches were considered before finally adopting a third approach which is discussed in Chapter 6. The approaches discussed here in this chapter are the "Active Networks" and the "Management Information Base" approaches.

#### **5.1 Active Networks:**

To evaluate the quality at each hop, some means is required to extract information at the designated hop. This chapter discusses obtaining such information using Active Networking techniques.

Active networks were presented as a suggested advanced alternative to the present ongoing packet switched networks. The advantage that active networks have over traditional circuit or packet switched networks is that the nodes can perform customised computations on the messages flowing through them. This includes modifying packet contents [Tennenhouse 1996]. Moreover, this processing can be customised on a per-user or per-application basis. Active networks have emerged



from the DARPA research community in 1994 and 1995 [Tennenhouse 1997]. The objective of the active networks model was to overcome the obstacles which exist in the current Internet technology mainly in the following areas:

- a) The complexity in incorporating new technologies into the shared infrastructure.
- b) The reduced performance due to redundant operations at some of the protocol layers.
- c) The obstacles in accommodating new services into the current architectural model.

#### **5.1.1 Activation of Active Nodes**

Generally, the main idea behind active networks was to advocate a dynamic deployment of application-specific services into networks. Through this concept, the intermediate nodes become active nodes, whereby routers are allowed to perform computations up to the application layer. Current traditional packet networks lack this feature, whereby user application data pass without alteration or examination except perhaps from changing packet headers by the router. Moreover, the processing which takes place at the router is done independently of the application which generated those packets.

In addition, one of the key features that adds to the advantages is that in active networks, third parties such as end-users, service-providers, and operators can program the network by activating the intermediate active nodes in either of two ways; encapsulation, or discrete.

### 5.1.1.1 Encapsulation approach

The encapsulation approach allows programs to be injected into the network by including the executable code in every packet sent by the application. These types of packets are called capsules because they include both the code fragments, which could be one instruction, and the application data. The program instructions within the capsule could be composed of primitives, which can perform basic processing of the data, and perhaps allow access to node resources outside the dynamic execution environment [Tennenhouse 1996]. The execution environment exists only while the capsule is being processed (Figure 5.1), where capsules are processed within a transient execution environment upon arrival at the intermediate nodes resulting in the modification of their state and behaviour [Psounis 1999].

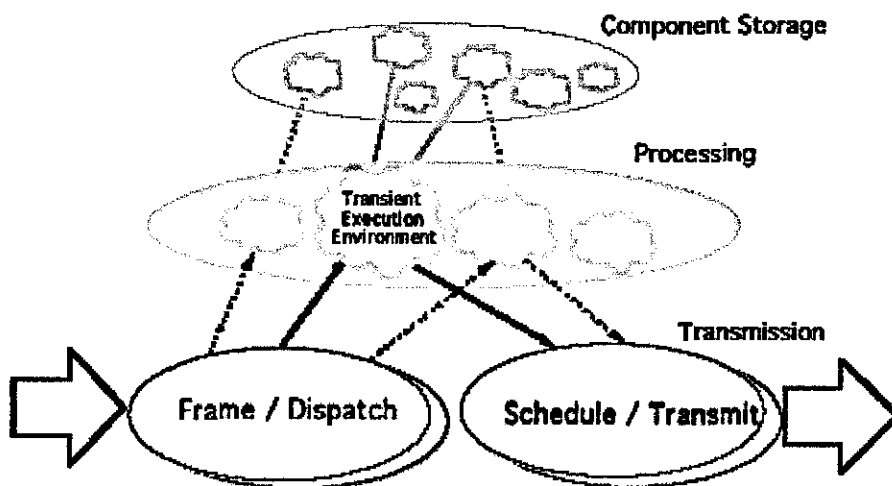


Figure 5.1 Capsule processing in an active node  
(adopted from [Tennenhouse 1996])

### **5.1.1.2 Discrete approach**

The discrete approach, is the approach used in the Intelligent Network architecture recommended by the CCITT [Tennenhouse 1996]. It differs from the previous approach in that the necessary program code is not injected into the same packet with the application data as capsules. Alternatively, another approach is chosen, whereby active applications transmit their packets in the same way as current applications do. Nevertheless, the packet headers contain identifiers that specify which locally stored program should be executed on arrival at the active node. This approach has the advantage of removing the task of program loading from the user. Instead operators of the network hardware could be made responsible for this.

In a work by Bhattacharjee et al [Bhattacharjee 1997], an active network has been developed in order to test its performance against a conventional packet switched network. They have proposed an architecture similar to that described by Tennenhouse et al and involves 'Foundation components'. In this architecture, each node in a network supports a set of functions that can be referenced by a unique identifier. These functions are designed to accept parameters in a similar way to that in which operating system commands are used. Each header in the packets passing through the node would contain a series of these identifiers together with the required parameters. The node would extract and execute the functions indicated in each. Certain basic functions would be those that support private data storage and retrieval of node state information such as the routing tables.

### **5.1.2 Active protocols**

Through the unique active networks feature of dynamic deployment of application-specific services, the constraints imposed by the use of generalised all-purpose transport layer services and packet formats are avoided when taking advantage of this active networks' feature of introducing the programmable-per-application network infrastructure. This is a more advanced concept than that of the present 5-layer IP model, which enforces a strict design approach on the developed applications with regards to their network interaction over conventional networks, which, in many cases, does not always take in to account the actual requirements of these applications.

With the active network concept, current network protocol stacks could be substituted with protocol components, which can be designed and composed to perform specific processing for the application. Consequently, through the dynamic deployment of application-specific services, new protocols can be installed at run-time in any node within the network. An example of this would be the deployment of Internet multicast technology, which has been slow because service providers have been reluctant to upgrade and reconfigure their routing nodes. However, using active networks, users who desire multicast services can have the service automatically installed without any direct intervention of the user or the service provider.

### **5.2 Approaches to hop by hop quality determination:**

The nodes within active networks, thus, can dynamically interact with each other, making it easy to load or execute the required programs on certain nodes or hops by accessing another node at the far end of the network. This feature of active

networks was utilised in this research to test and evaluate the feasibility of accurately measuring loss and delay parameters at each hop. Thus, this would determine the precise QoS at those hops for the application under consideration on a hop-by-hop basis. Initially, this approach would only apply to Active Networks.

The experiments were executed in a controlled environment on an off-line test network. Copies of the LNE were loaded on to every active router in the test network to introduce various network parameters. Since end-to-end delay cannot be directly measured due to the lack of synchronised time bases at the nodes, two approaches were considered; Time-stamping and Active ping, to extract the necessary hop information. Through these two techniques it was hoped that more accurate delay measurement could be achieved using intermediate nodes to measure transit delay. The total delay is then the sum of the time spent at each node. Both of these approaches are discussed in the following sections.

### **5.2.1 Time-stamping**

The delay-variation experienced within a network occurs mainly due to the queuing and routing processes taking place inside the routers along a designated network path. To estimate the delay introduced by a router requires calculating the time that the packets spend inside the router from the time they arrive till the time they are forwarded to the next hop.

Time-stamping the packets passing through a router at both the arriving and departing interfaces should provide the time spent at the router itself. The test packets are stamped with the entry and exit time at each intermediate node along the network path. The overall delay would then be the sum of the times spent at

each intermediate node, excluding propagation delay due to the communication channels. This however is fixed.

Nevertheless, it was realized that the process of time stamping every single packet passing through every intermediate node could, in itself, add to the delay factor thereby causing an error in the analysis. Therefore, rather than stamping every single packet, an attempt to time-stamp a certain percentage of the packets passing through the active routers on a VoIP connection path was made. The time-stamping was set to occur at the ingress and egress of the router. Hence, the difference in the outgoing and incoming time-stamps provides the delay caused by the queuing and routing processes executing in the designated node.

The timestamp however, does not represent the exact arrival time at the intermediate node. Therefore this technique is not totally error free. More seriously, the measurement mechanism intrudes upon node operation and this generally cannot be tolerated [Siddiqui 1989].

Practically, for the time-stamping procedure to be implemented, packets have to be passed into User Space via the IP-tables from the "Input" queue before these are passed to "Post-routing" queue through the "Output" queue. However, in practice this proved to be very difficult to achieve. This is because the packets would bypass the "Forward" queue, which packets are meant to pass through to be redirected in the case of a router in a real network scenario. Hence, the bypassing packets will not suffer from the queuing delay in the forward queue, whose time-span is required to be measured (Figure 5.2). Moreover, redirecting the packets in two other queues; the "Input" and "Output" queues, adds to the delay. Obviously, this procedure does not reflect the actual operation of routers on a real network.

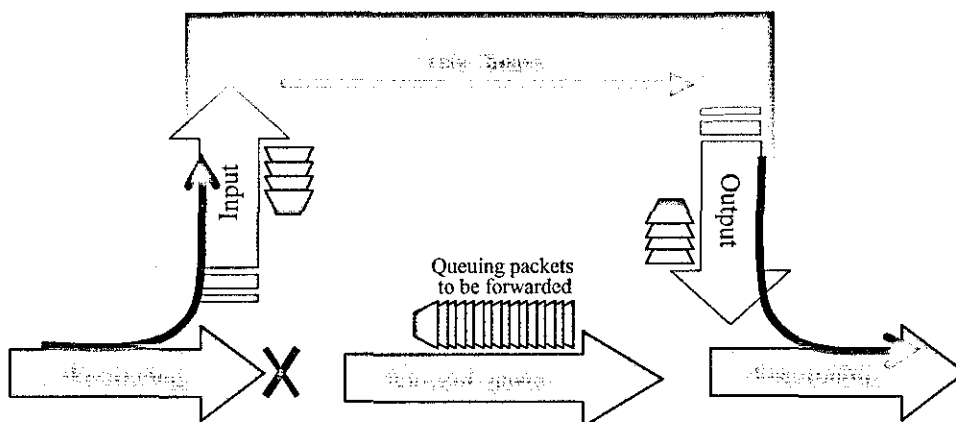


Figure 5.2: To be time-stamped, traffic is diverted to user space thus bypassing the forward queue.

An alternative method was investigated to overcome the previous problem. This involves time-stamping the packet at the pre-routing queue itself. This method is complicated and involves manipulating the system kernel. However, it was found that it was not possible to access the packet to time-stamp it except at the end of the "Pre-routing" queue, i.e. after it passed the entire "Pre-routing" queue. Time-stamping the packet while it is leaving the router should be performed at the "Post-routing" queue. However, the last place the time-stamp can be applied is right before the packet is put into the "Post-routing" queue. This procedure also fails to provide accurate delay information since it does not take into consideration the delay that occurs because of the "Pre-routing" and "Post-routing" queues.

Since the time stamping approach is expected to provide inaccurate and unrealistic delay measurements, it is not considered further in this research. An alternative approach is investigated in the following section.

### 5.2.2 Active Ping

The Active ping approach was used to measure the delay at the relevant hops in a network connection. The active ping overcomes two main shortcomings; one is that of the conventional ping utility, the second is of that of the time-stamping approach.

The conventional ping utility calculates the round-trip delay as the time spent by a ping message during its journey to the destination node and the time spent by its response from the destination node back to the source node. The utility fails to provide information regarding delays at different hops along the path. In this regard Active ping seeks to overcome the problem imposed by the delay accumulated from the intermediate hops between the source of the conventional ping message and the destination node.

In comparison with time-stamping, Active ping approach is advantageous in comparison, because it adequately deals with the lack of synchronisation of clocks at individual nodes, which prevents the calculation of hop-by-hop delay. In the time-stamping approach, the time according to the local node is stamped, hence when calculating the delay at the source, there will be errors since intermediate nodes are not synchronised, including the source node where the calculation will take place. Accordingly, the retrieved timing data for each hop would only show the time of the node according to that node's clock only. Hence, it would not be correct to perform calculations on the received timing data to detect the delay that occurred at these hops due to the variation in the clocks of the nodes. However, if these delay calculations, were carried out independently at each node on the route, they would give correct delay data.



The following is an example of how active-ping operates. If a traffic route exists from node "A" to node "E", passing through three other nodes "B", "C" and "D", the active ping would work as follows:

Upon sending the "Active Ping" command from node "A" to node "E", the "Active Ping" procedure, which was previously loaded on each node between the source and the destination i.e. "B", "C" and "D", is activated. This should result in node "B" sending a sequence of 5 ICMP ping messages every 5 seconds back to node "A", with an interval of 0.1 Seconds between the subsequent pings, Figure(5.3).

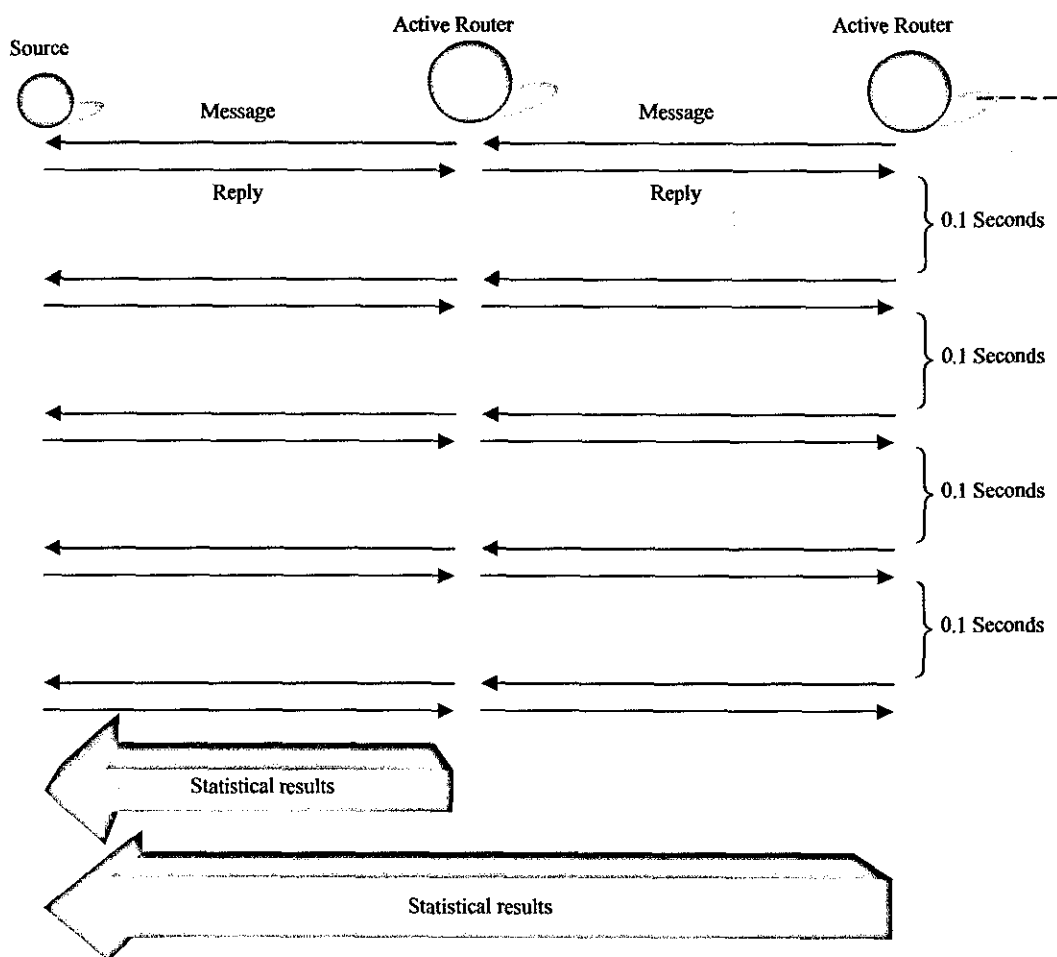


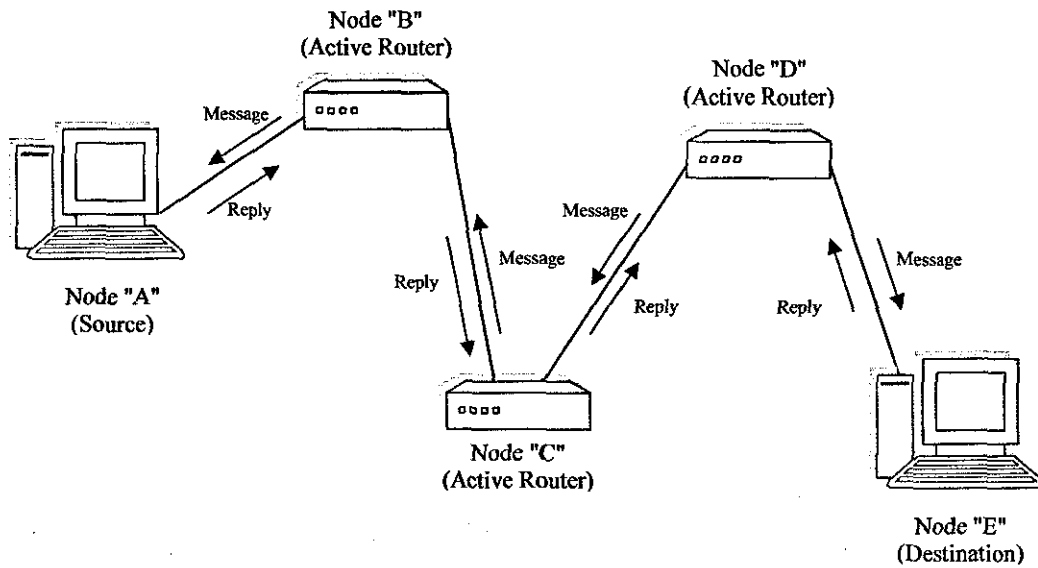
Figure 5.3: The execution of "Active Ping" activates the above procedure every 5 Seconds, from source to destination.

Hence, we can calculate four parameters of delay based on these five ping messages. The four parameters are:

- 1- Minimum delay of the five ping messages-.
- 2- Average delay.
- 3- Maximum delay.
- 4- Standard deviation.

The statistical results are performed at each node independently and are then sent to the end source node (Figure 5.4), at the end of every 5 pings. This algorithm is repeated for a large number of series to minimise any chance of one-off change of hop condition.

Node "B" would then check if the next node is actually the last destination. If this were the case, it would mean that the current router is the last active router that is able to run the active ping program. If not, a ping message would also be sent to the next node and it would work out the delay between the two nodes. This conditional step is carried out in each active router in the network path. Subsequently, node "C" would send an ICMP message to node "B", hence calculating the delay between itself and "B". As for node "D" it would send two ping messages, one for the previous node -which is "C"- and next node -which is "E"-, thus calculating the delay in both directions, Figure (5.4).



**Figure (5.4):** Direction of ping messages in "Active Ping" program.

The active ping technique has been found to be promising because it overcomes the lack of clock synchronisation as each node performs its calculations according to its own clock. The data of the resulting statistics are then sent back to the end source node by each active router separately. However, this technique cannot be applied in an operational network, other than the test network, as active nodes are required for this approach to be exercised. However, this could prove to be very useful in active networks in the future.

### 5.3 Hop quality detection using Management Information Base (MIB)

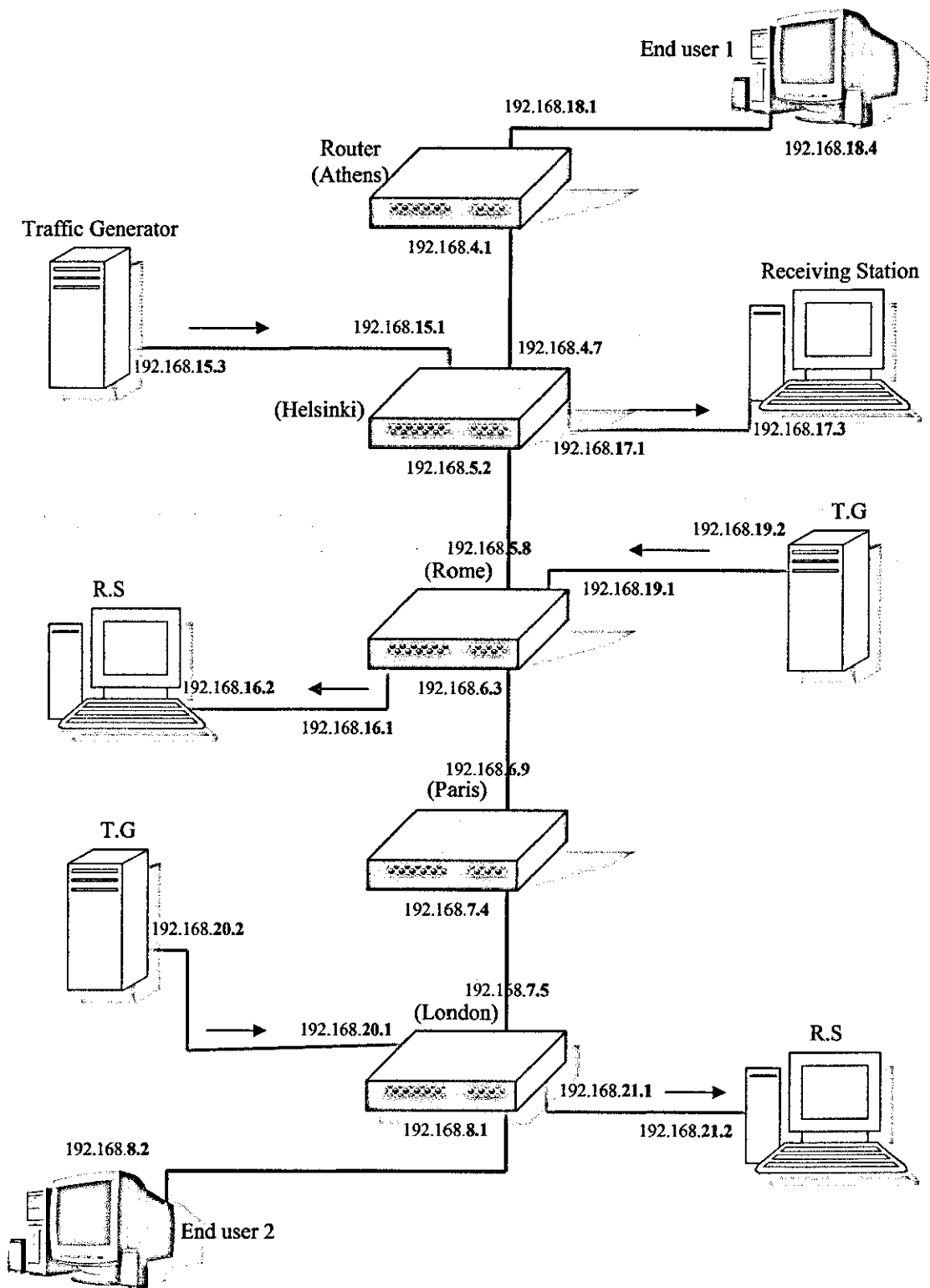
During the evaluation phase, both packet-loss and delay-variation values were known before hand, since they were preconfigured via the LNE. However, these

parameter values are unknown over a real operational network at each individual hop. Hence, a means to determine these values of loss and delay at each individual hop independently on a common network type is required. Such measurements can be made at each hop and maintained in the Management Information Base (MIB) [Waldbusser 2000]. Means to access these measurements from the MIB are addressed in this chapter.

#### **5.4 Trial Experiment**

Initially, an experiment was performed using 5 Cisco routers in between two end-users. Traffic generators were used to simulate various traffic loads on some of the routers (figure 5.5). Loss values were determined using the queuing data available via each router's command prompt. It was possible to retrieve this information because of the controlled nature of the experiment where there was only one connection on the test network. This was the VoIP (Netmeeting) connection between the two assigned end users.

When the interfaces of the routers through which the traffic of the session between the two end users passes are known, the packet loss which is calculated at the routers refer only to the loss of the VoIP application traffic due to heavy cross traffic from the traffic generator (See figure 5.5).



**Figure 5.5:** Network configuration for the experiment.

## **5.5 Hop parameters extraction from the MIB (Management Information Base)**

Because the means to determine packet-loss information previously described is confined to a controlled test network, other means of calculating the packet loss percentage at routers in an operational network were explored. One approach is to use the MIB “Management Information Base” to get the information needed to assess VoIP quality at the designated hops. MIBs are an important part of network management elements and stores management and performance information related to various network elements [Waldbusser 2000].

### **5.5.1 Network Management elements**

Generally, network management consists of the following:

1. **Management Information Base (MIB):** a data structure containing a collection of all possible objects in a network. Objects are the variables (information), maintained by the elements or nodes, which can be queried and set by the manager.
2. **Structure of Management Information (SMI):** is the general framework in which an MIB can be defined and constructed. It is a subset of ASN1 (Abstract Syntax Notation 1) used to define the SNMP data structure, and is an identification scheme used to reference the variables in the MIB.
3. **Simple Network Management Protocol (SNMP):** a protocol, which governs the interaction between the management station and the agents, as regard to packet exchange format details. The protocol allows the management station to query the state of an agent’s local objects, and change them if necessary.

### 5.5.1.1 Simple Network Management Protocol (SNMP)

In 1990 version 1 of SNMP was released with the publishing of RFC1157. followed by SNMP V2, which was defined in RFCs 1441 to 1452.

SNMP model consists of four components:

1. Managed **nodes**.
2. Management **stations**.
3. Management **information**.
4. A management **protocol**.

Each node executes an SNMP management process called an **SNMP Agent**. The agent maintains a local database of variables that describe its state and history and affects its operation.

Network management is conducted from management stations, which are general-purpose computers running special management software (figure 5.6).

The management station interacts with the agents using the **SNMP Protocol**. The protocol enables the management station to query the state of an agent's local objects, and change them if required.

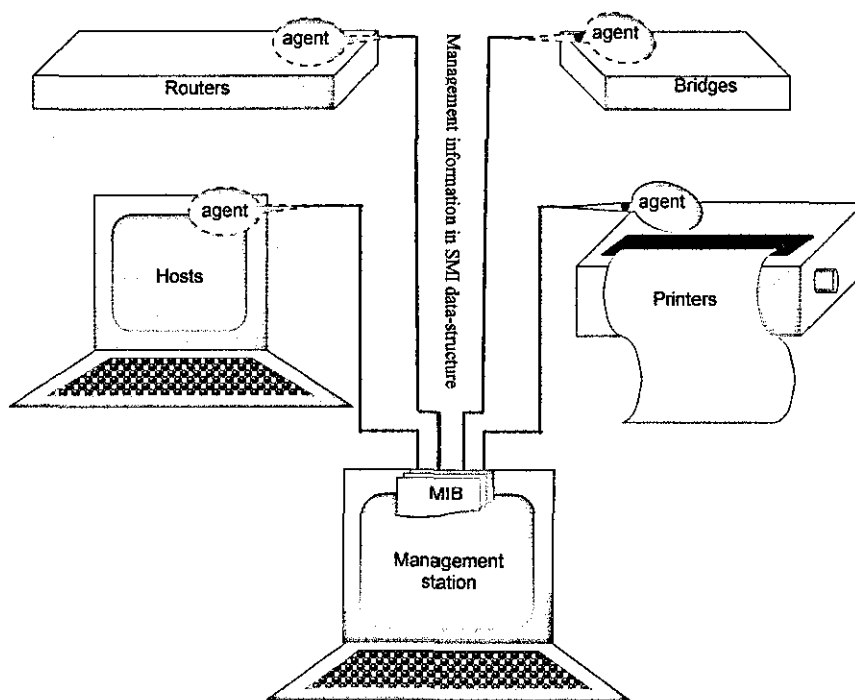


Figure (5.6): SNMP governs how the management information is handled between the SNMP agents in managed nodes and the management stations. The information is set in SMI data-structure used as a reference for MIB variables or objects.

Management stations checks for unusual events by polling each managed node randomly. The model of polling at long intervals with acceleration on receipt of trap, is known as *"trap directed polling"*. When a significant event occurs however, the agent reports an SNMP *"Trap"* to all of its configuration list of management stations. Accordingly, the management stations issues queries, if necessary, to determine the required details.

In the case of older devices, which are not intended for use on a network and thus not capable of running an SNMP agent internally, SNMP defines what is called a *"Proxy agent"*, which is an agent that watches over non-SNMP devices and



communicates with the management station on their behalf, possibly communicating with the devices themselves using some non-standard protocol.

To manage a network with multi-vendor nodes, each device maintains one or more variables that describe its state. These variables are called “*Objects*”, which agents have to provide in a certain format. The collection of all possible objects in a network is given in a data structure called the “*MIB (Management Information Base)*”.

Objects are defined in a standard and vendor-neutral way, so as to make multi-vendor communication possible. Furthermore, a standard way is needed to encode them for transfer over the network. For this reason a standard object definition language, along with the encoding rules is used and is called ASN.1 [MacFaden 2003]. The SNMP data structure is defined using “*SMI (Structure of Management Information)*”, which is a subset of ASN1.

ASN.1 uses a transfer syntax called “*Basic Encoding Rules (BER)*” [Ibid]. BER defines the encoding of the component objects by defining how values of ASN.1 types, both primitive or constructed, are converted to a sequence of bytes for transmission.

SNMP defines only five types of messages:

1. Fetch the value of a variable: (*get – request*)
2. Fetch the value of the next variable: (*get next request*)
3. Set the value of a variable: (*set – request*)
4. Return the value of a variable from the agent to the management station for either of the above queries: (*get – response*)
5. Notifying the management station by the agent when something happens: (*trap*)

### 5.5.1.2 MIB extraction

Network managers often utilise InterMapper [Govindan 2000] to view the state of the network of their domain. InterMapper queries routers, switches, and hubs using SNMP to receive and display critical data about traffic.

In SNMP, a logical group of managed devices and network management stations in the same administrative domain is called a community [Siamwalla 1999]. This community has a community string or a community name which is text string that acts as a password and is used to authenticate messages sent between a management station and a router containing an SNMP agent. The community string is sent in every packet between the manager and the agent.

To acquire MIB information for a certain router, the community string for that router needs to be known to allow access to the designated device. The SNMP Read-only Community string is like a user id or password that allows access to statistics maintained for a router or device. InterMapper sends the community string along with all SNMP requests. If the community string is correct, the device responds with the requested information. If the community string is incorrect, the device simply discards the request and does not respond. SNMP community strings are used only by devices that support SNMP protocol.

There are three community strings types for devices that can be managed by SNMP:

- **SNMP Read-only community string** - enables a remote device to retrieve "read-only" information from a device. InterMapper uses this to request information from devices on its maps.

- **SNMP Read-Write community string** - allows a remote device to read information from a device and to modify settings on that device. InterMapper does not use the read-write community string, since it never attempts to modify any settings on its devices.
- **SNMP Trap community string** - used when sending SNMP Traps to another device. InterMapper accepts any SNMP Trap community string.

## **5.6 Feasibility of using MIB Statistics for Performance Measurements**

By convention, most equipment is shipped from the factory with a read-only community string set to "public". It is standard practice for network managers to change all the community strings so that unauthorised entities cannot retrieve information about the internal network. Additionally, network managers may employ firewalls to block any SNMP traffic to ports 161 and 162 on the internal network [Fenner 2005].

However, this is not a practical approach. Although the loss percentage can be conveniently determined on the test network where preconfigured routers enable MIB extraction, this may not be applicable to an operational WAN that is under the management of various service providers.

The router at each hop has to be configured to allow SNMP access. Moreover, depending on which version of SNMP is running on the router, the community that is configured on the routers has to be known so that it can be used in an SNMP query. Community strings are used in SNMP v1, v2c. However, if the router is configured to run SNMP v3 only, which is more secure than SNMP v2

and v2c, then the user name and engine Id that is configured on the router has to be known in order to access it. Even though many routers and IOS devices support the CBQoSmb “Class-Based Quality of Service Management Information Base”, there is the possibility that some don't, thus it will not be possible to access MIBs information for all hops. Hence this approach proves to be impractical for non-administrative privileges users.

## **5.7 Conclusion:**

In this chapter two approaches to determine the quality of service at the hops along a given network path were discussed; the Active Networks and the Management Information Base.

Active Networks are very flexible and assist in determining the grading of the quality of service at the hops in a network. Their programmable nature makes it easy to gather the information of the variables needed to verify the quality of service at each hop by applying the grading algorithm discussed in the previous chapter to the extracted variables from each hop independently.

The aforementioned time-stamping technique proved to be incompetent in comparison with the active-ping approach because of the inability to synchronise the processors clocks of all the nodes involved in the communicative session, including the two end-user nodes. Moreover, due to the IP-tables design, where the time-stamping will take place, the time-stamping will encompass errors, including added delay which occurs due to the buffering in the two extra queues which are not present in a normal network scenario, while neglecting the forward queue which is the queue that represent the actual delay at the node.

The active ping technique suggested above could prove to be useful and efficient when used in the active network environment. Nevertheless, current networks are not active. This makes the above mentioned approach confined to an active network environment only. Thus other approaches are required to be sought that can be implemented using the current networking technologies. Consequently more techniques are examined in the following two chapters.

The Management Information Base was considered as an approach to acquire the required statistical data along the network path of the VoIP session taking place between two end-users. It was realized however, that it would not be possible to access the MIB at the routers which are not under the network administrative privilege, which require the community string for the designated router, which acts as a password for the network manager. Thus makes this approach confined to the local network domain only.

Accordingly, another method to extract network hop parameter is explored. This mechanism is discussed in the following chapter.

## *Chapter 6*

### **Hop By Hop Quality Grading**

#### **6.1 Introduction**

**M**eans of extracting network performance data at each hop independently to ascertain the quality of service at a designated hop has been examined in the previous chapters. The aforementioned means of data extraction fall short of accomplishing hop by hop application grading detection in a universal open network environment. The use of the active network active ping, which is examined in chapter 5, revealed its feasibility of use only on active networks, and is not applicable to the existing networking technologies. Furthermore, extracting network performance measurements from the Management Information Base available at each hop may not be a feasible approach, as this technique is restricted only to those hops that are under the same administration or network management, since it requires community string permission.

An alternative approach is discussed in this chapter which seeks to overcome the above mentioned deficiencies. The approach aims to reveal the level of the quality of service obtainable for real-time multimedia applications at each intermediate hop by a simple universal means. This means should be simple in that it requires no special networking applications or features, but rather workable through the use of simple attainable means available on the existing network infrastructure without the need to upgrade current routers or their software. The availability of such means through-out the existing network infrastructure makes it possible to use on any given network. Thus, the universality obstacle which arises when

using the active networks discussed in chapter 5 is resolved. Additionally, the simplicity and universality of such means paves the way to enquire about the necessary information from the various routers which fall under different network administrations. Therefore the restraints imposed by the different network managements, which are discussed in the former chapter, are reduced to a large extent.

A prototype tool was developed, utilising the widely commonly used traceroute and ping network utilities. These allow a series of ping messages to be transmitted from one end-user to each intermediate hop across the network to the other end of a VoIP connection. Although the ping utility does not show the hidden information of packet loss and delay for the intermediate hops, this approach can be extended as a mechanism to determine the packet loss and delay variation on each hop on the designated network path.

A periodic check of network path could be applied using traceroute to ensure that the intermediate hops which are being pinged are the same. However, several studies have revealed that most of the traffic between two or more communicating parties remains on the same physical path during the session [Paxson 1997]. Accordingly, routes will rarely keep changing during a voice conversation.

If treated uniformly by the hops, ping messages would suffer, to a large extent, from the same network conditions that the UDP traffic of VoIP connection would suffer from. This approach proved to be simple and can be used by any end-user station in a real network. Thus, it is a universal approach and is not confined to particular networks or domains.

## 6.2 Experiment Layout

A test network was constructed which consisted of two end-users executing voice applications. In this test network, 5 PC-based routers formed the network path. Each of the PCs supported a LNE (Loaded Network Emulator) to emulate real network conditions. As such, each PC would work as a network router with various controlled quality of service conditions (Figure 6.1). The LNE on each of the routers was set to a different loss percentage value.

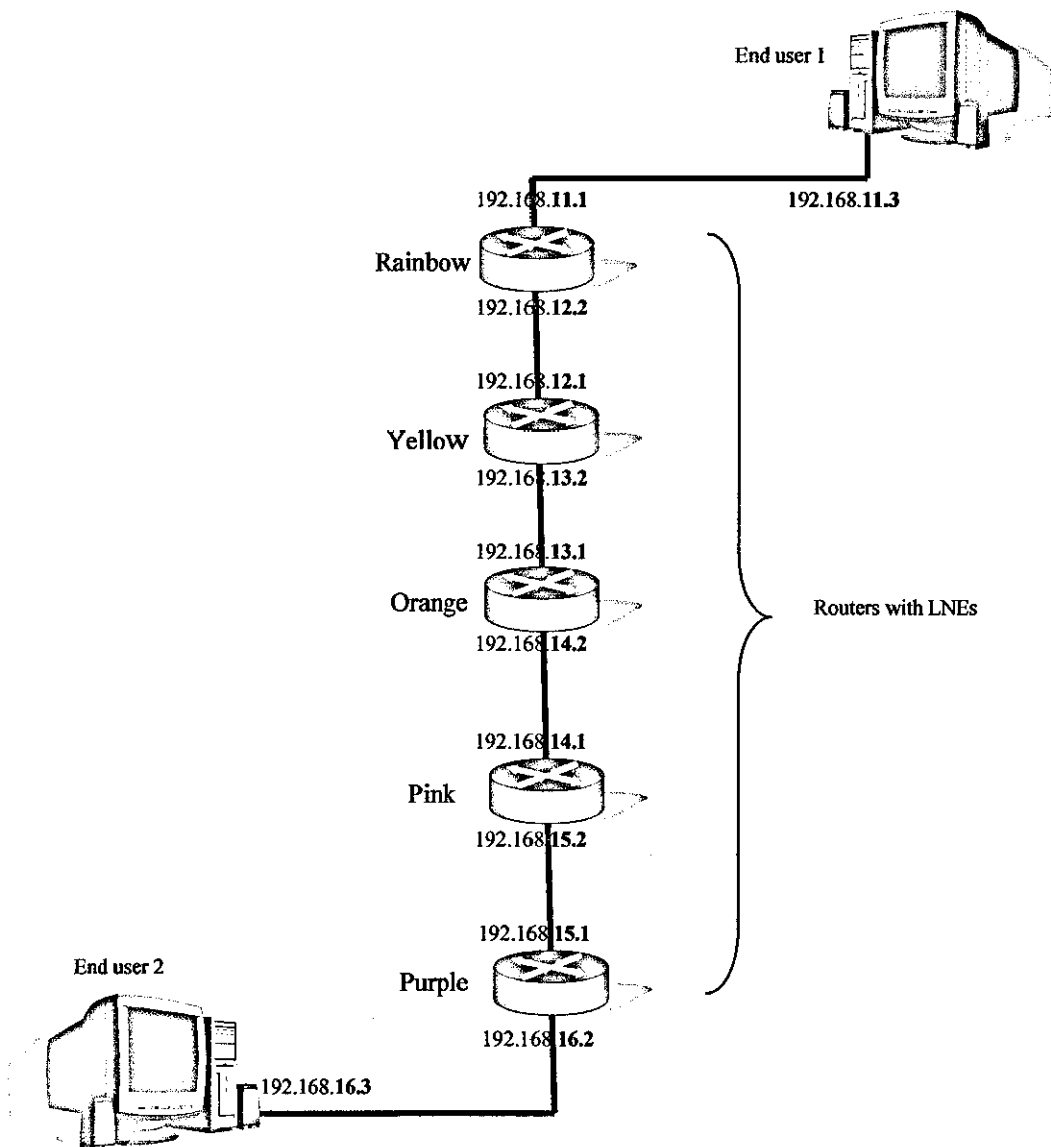


Figure 6.1: VoIP connection established between end user 1 and 2 through 5 generic routers experiencing different emulated network conditions.



### 6.3 Packet-Loss Dependency

A semi-formal evaluation of voice quality was conducted in the lab using several subjects. Accordingly, five quality grade values were assigned to define application quality; a grading of 5, 4, 3, 2, and 1 [Daumer 1982] for excellent, good, fair, bad, and poor respectively. These will be influenced by various values of loss and jitter. The previously discussed calculation methods, which based the quality on the effect of both jitter and loss, were to be applied. However, according to the conducted evaluations in the lab, it was found that packet-loss percentage plays a major role in affecting application quality. (Table 6.1) displays part of the evaluation where the change of the variation of delay value had insignificant change on the quality as demonstrated in the following two figures. Figure 6.2 shows that the change in delay while fixing packet-loss values had no effect on the quality. On the other hand quality was affected in respect to packet loss change in the light of fixed jitter values as seen in (Figure 6.3).

Both fixed delay and delay variation were found to have insignificant effects. In regards to Fixed delay it proved to be irrelevant since it does not directly degrade the quality of the received audio [Oliver 1999]. Delay variation on the other hand, was expected to have an influence in degrading the voice quality. However, most VoIP applications, and Microsoft Netmeeting<sup>®</sup> in this case, have their own built-in buffer, usually known as the jitter buffer, where some packets are held before being released for playout. Thus the application accommodates and tolerates the changes in delay that might occur due to the changes in queuing traffic at the buffers of different routers. This queuing occurs due to the burstiness in network traffic. Hence, the buffering which takes place in the VoIP application is designed to tolerate the variation in delay in order to ensure a continuous and an uninterrupted sound quality as possible.

Case No.	Loss (%)	Delay (ms)	Quality							
			Observer 1 Tammam	Observer 2 Mark	Observer 3 David	Observer 4 Pet	Observer 5 Shuri	Observer 6 Omar	Average G	Aprox G
1	9	300	3	3	3	3	3	3	3	3
2	9	600	3	3	3	3	3	3	3	3
3	9	1500	3	3	3	3	2	3	2.833333	3
4	9	2600	3	3	3	3	2	2	2.666667	3
5	25	300	1	1	1	1	1	1	1	1
6	25	600	1	1	1	1	1	1	1	1
7	25	1500	1	1	1	1	1	1	1	1
8	25	2600	1	1	1	1	1	1	1	1
9	5	300	4	4	5	4	4	4	4.166667	4
10	5	600	4	5	4	4	4	4	4.166667	4
11	5	1500	4	4	4	4	4	4	4	4
12	5	2600	4	4	4	4	4	4	4	4
13	17	300	2	2	2	2	2	2	2	2
14	17	600	2	2	2	2	2	2	2	2
15	17	1500	1	2	2	2	2	2	1.833333	2
16	17	2600	1	2	2	1	1	2	1.5	2
17	2	300	5	5	5	5	5	5	5	5
18	2	600	5	5	5	5	5	5	5	5
19	2	1500	5	5	5	5	5	5	5	5
20	2	2600	5	5	5	5	5	5	5	5
21	21	300	1	1	1	1	1	3	1.333333	1
22	21	600	2	1	1	1	1	1	1.166667	1
23	21	1500	1	1	1	1	1	1	1	1
24	21	2600	1	1	1	1	1	1	1	1
25	12	300	2	2	2	2	2	3	2.166667	2
26	12	600	2	2	2	2	2	2	2	2
27	12	1500	1	2	2	2	1	2	1.666667	2
28	12	2600	1	2	1	2	1	2	1.5	2
29	5	300	4	4	4	4	4	4	4	4
30	5	600	4	4	4	4	4	4	4	4
31	5	1500	4	4	4	4	4	4	4	4
32	5	2600	4	4	4	4	4	3	3.833333	4
33	14	300	2	2	2	3	2	2	2.166667	2
34	14	600	2	2	2	2	2	2	2	2
35	14	1500	1	2	2	2	2	2	1.833333	2
36	14	2600	2	2	1	2	1	2	1.666667	2
37	10	300	3	3	4	3	3	3	3.166667	3
38	10	600	4	3	3	3	3	3	3.166667	3
39	10	1500	3	3	3	2	3	3	2.833333	3
40	10	2600	3	3	3	2	3	3	2.833333	3
41	23	300	1	1	1	1	1	1	1	1
42	23	600	1	1	1	1	1	2	1.166667	1
43	23	1500	1	2	1	1	1	1	1.166667	1
44	23	2600	1	1	1	1	1	1	1	1
45	3	300	4	5	5	5	5	5	4.833333	5
46	3	600	5	5	5	5	3	5	4.666667	5
47	3	1500	5	5	5	5	5	5	5	5
48	3	2600	5	5	5	5	4	5	4.833333	5

Table 6.1: The effect of packet-loss and jitter on the quality.

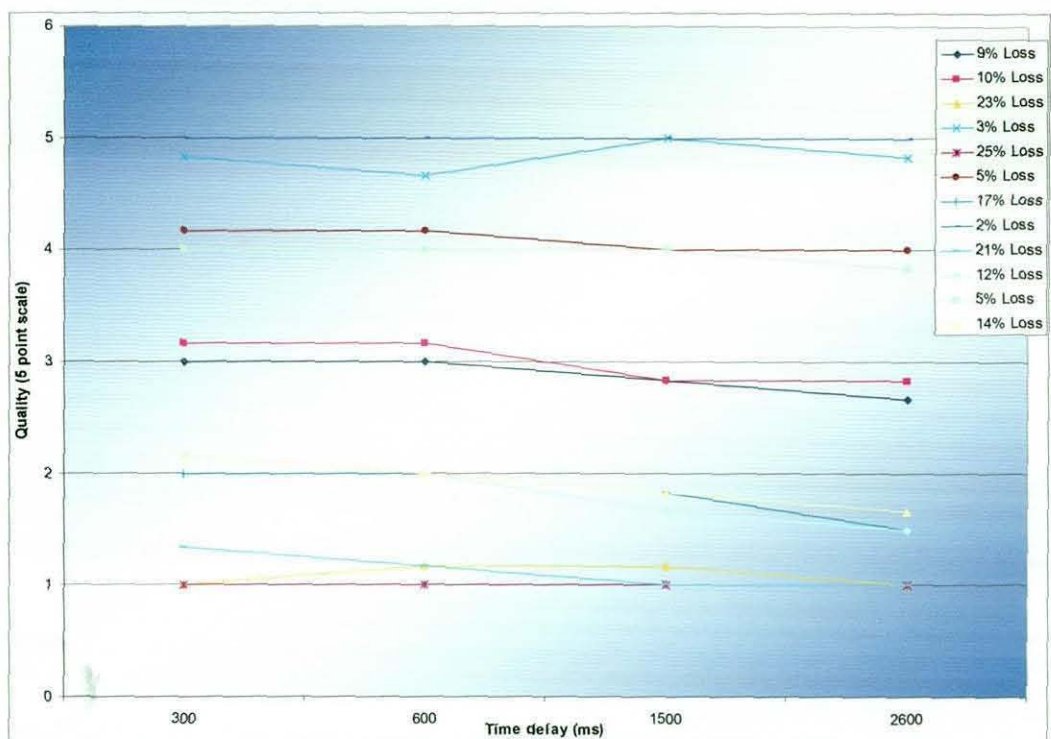


Figure 6.2: Quality vs. Time-delay for fixed Loss.

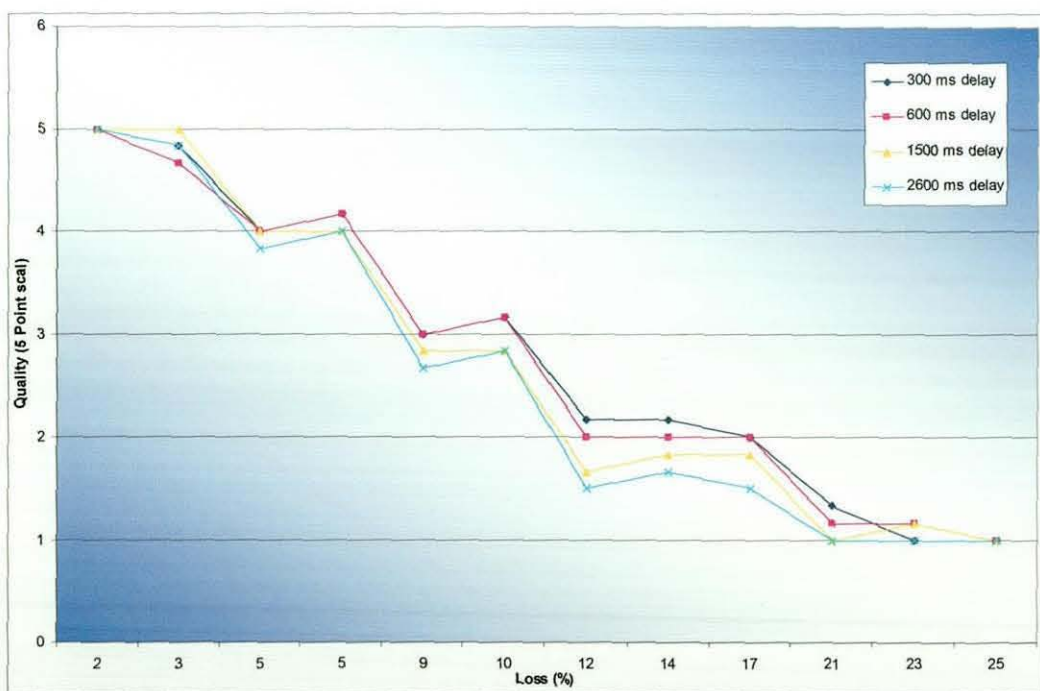


Figure 6.3: Quality vs. Loss for fixed Time-delay.

As previously mentioned, delay variation has little effect on sound quality unless it approaches large values, provided adequate buffers are introduced. Large delay variation values were introduced on the test network and it was found that a delay variation of about 3000 ms or greater will significantly influence voice quality as the audio here is streaming audio. However for a two way real-time audio reproduction, conversation will be affected by a delay value even smaller than 3000 ms.

However, these excessively large jitter values are uncommon in real life situations, where values such as 300 ms are considered to be large in a practical situation. Most RTTs in the Internet are within the range of 70-160 ms, although some may exceed 300 ms when Internet traffic is exceptionally heavy [Black 2000 pp: 38]. Accordingly, based on many studies and observations, RTT was limited to 300ms or less for telephone traffic by the ITU-T G.114 Recommendation [ibid].

Therefore, delay variation was excluded from the grading function. Hence, the VoIP application is mainly dependant upon packet-loss when it comes to quality of service grading. This gives the advantage of increasing the order of the equation to give a more accurate result, so that  $G$  would be a 3<sup>rd</sup> degree function of packet loss percentage as the relation in equation 6.1 suggests:

$$G(l) = f_0 + f_1l + f_2l^2 + f_3l^3 \quad (6.1)$$



## 6.4 Design of Prototype Tool

A prototype tool was designed to determine the quality of service at the intermediate hops over which a VoIP connection passed and used the traceroute algorithm to determine the path. Subsequently, a second process was then invoked, whereby a series of equal number of ping messages were sent independently to each router that is identified in the previous step. Multi-threading was used to concurrently transmit these pings to the hops and collect the resulting measurements. Collecting the packet-loss data for all hops therefore occurs at the same time.

When a series of ping messages are transmitted from a source node to a destination node, through a network (figure 6.4), both transmitted requests and received replies suffer from the same traffic condition which the network is experiencing in both directions, if all the hops treat all the Internet traffic uniformly. Studies have revealed that most of the traffic between two or more communicating parties remains on the same physical path during the session [Paxson 1997].

Accordingly, in a case such as the following example, the packet loss scenario would occur as follows: if 100 ping packets were transmitted to pass through a network which is experiencing 10% loss, 90 packets will arrive at the destination. In return, the destination node will issue 90 reply packets to the source node, equal to the amount of pings it received. However, upon passing through network on the way back, the 90 reply packets will also experience 10% loss. Hence, only 81 reply packets will arrive at the source node.



**Figure 6.4:** The effect of network congestion on both ways; transmitted and received packets.

Thus, the resultant packet loss percentage “ $l$ ”, would not simply be the percentage of received packets over transmitted packets or:  $l = 100 * (1 - \frac{\text{received replies}}{\text{transmitted pings}})$ , but rather:

$$l = 100 * (1 - \sqrt{\frac{\text{received replies}}{\text{transmitted pings}}}) , \text{ [Oliver 1999]} \quad (6.2)$$

The previous arguments and equation help towards determining the loss percentage in a network. The aim remains is how to find out where in the network the deterioration occurs and which router or routers are likely to be the cause of this degradation, due to the losses experienced there. In other words, the segment of the network which is responsible for the poor quality of VoIP applications needs to be localised.

As mentioned earlier; the packet-loss value for each hop is extracted at the source node by the prototype tool. However, this loss value actually includes, and is affected by the loss values experienced at all previous intermediate hops together.

In an example where there are 5 hops “a”, “b”, “c”, “d” and “e”, in between the source and destination end-users (figure 6.5), the packet-loss data extracted at the source-end for hop “d” is an accumulation of all packet losses experienced at hops “a”, “b”, and “c”.

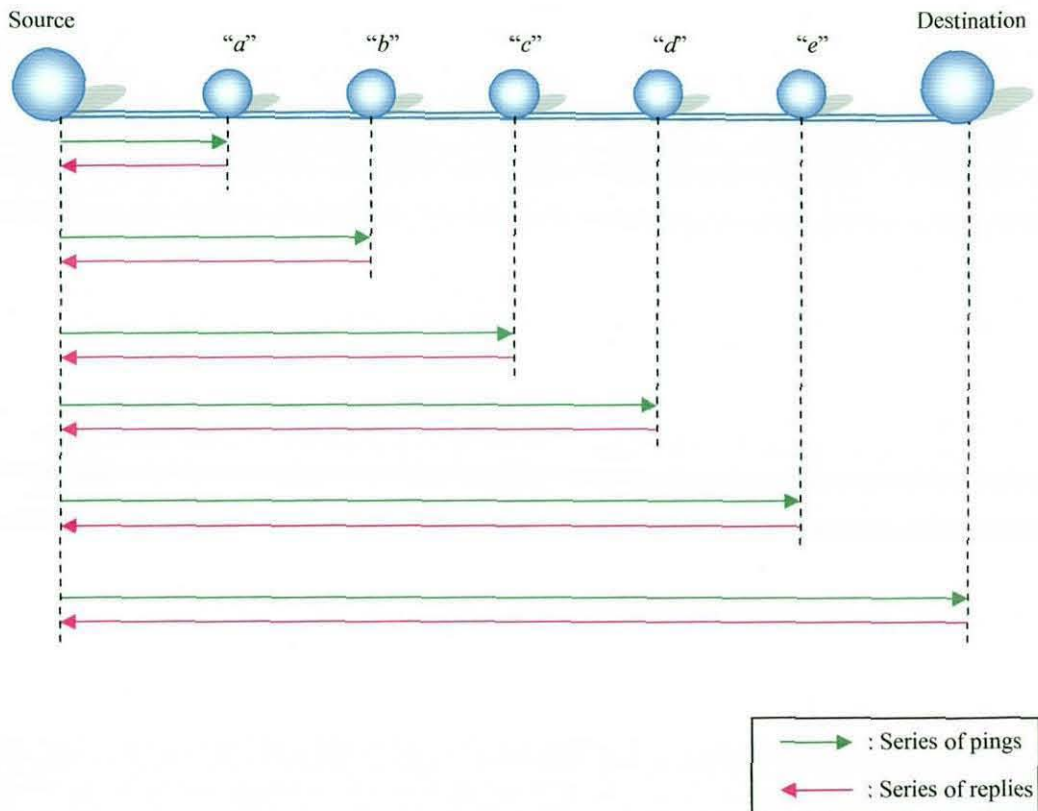


Figure 6.5: The accumulation of packet-loss from intermediate hops.

Hence, the actual packet-loss experienced due to the queuing at a certain hop is the difference between the loss at that hop and the loss at the preceding hop:

$$l\_diff_i = l_i - l_{i-1} \quad (6.3)$$



After running quality grading process on the subjects that took part in the experiment, the two vectors of the packet loss values " $l\_diff$ " and the

corresponding quality grading values " $G$ " which are extracted from the evaluation process of the experiment are fed in to the polynomial curve fitting of the Matlab<sup>®</sup> toolbox in order to acquire the factors " $f$ " in equation (6.1). The Matlab<sup>®</sup> curve fitting toolbox was used in order to find the values of the factors " $f$ " which represent the closest possible curve that pass through all or most of the points resulting from the X and Y axis which are the packet loss " $l\_diff$ " and quality grade " $G$ " in this case. If the factors " $f$ " did not represent a fitted curve but rather represented a line that strictly passes through all the points on the graph, it would mean that equation (6.1) with the derived factors " $f$ " values will not support any new packet loss values " $l\_diff$ " other than the exact original " $l\_diff$ " values.

The packet loss values " $l\_diff$ " along with their corresponding quality grade values " $G$ ", according to the evaluation submitted by the subjects, were run by the Matlab<sup>®</sup> toolbox for curve fitting. The resulting factors " $f$ " were found to be: (-0.0005), (0.0283), (-0.6172), and (6.5559).

Subsequently, the quality grading at the designated hop ( $i$ ), would then be the rounded value of " $G_i$ " which, from equation (6.1) would be:

$$G_i = f_0 + f_1(l\_diff_i) + f_2(l\_diff_i)^2 + f_3(l\_diff_i)^3 \quad (6.4)$$

Substituting the previous values of the factors " $f$ " in equation (6.4):

$$G_i = 6.5559 + (-0.6172 * (l\_diff_i)) + (0.0283 * (l\_diff_i)^2) + (-0.0005 * (l\_diff_i)^3) \quad (6.5)$$



## 6.5 Algorithm for a Prototype Tool

Referring to the example of figure 6.1, the algorithm of the prototype tool is as follows:

- 1- Determine the destination IP-address.
- 2- Execute the *traceroute* program at the source to determine the path to the destination route. This helps determine all IP-addresses on the network route to destination node and registers them.
- 3- Assign a number of threads according to the previously registered IP-addresses.
- 4- Execute a series of *ping* programs within these threads to address all hops.
- 5- Store the replies for each hop in separate individual files.
- 6- Extract the required parameters from the received ping replies of each hop.
- 7- Calculate the *loss percentage* parameter at each hop as follows:
  - a) Determining the accumulated *loss percentage* value "*l*" up to the designated hop "*i*", according to equation (6.2).

- b) Ignoring the first hop and working from the nearest hop onwards, calculate the loss difference " $l\_diff_i$ " according to equation (6.3), between the accumulated *loss percentage* value of a certain hop " $l_i$ " and that of the previous hop " $l_{i-1}$ ".

Originally, the prototype tool was designed to calculate the quality grading according to the main three parameters; fixed-delay, delay-variation, and packet-loss percentage before the quality of VoIP applications was found to be mainly dependant on loss-percentage. Thus the remaining two algorithm stages cater for fixed-delay and delay-variation, as they were originally part of the tool.

- 8- Calculate fixed delay, which is represented by the average delay " $av\_del_i$ " of the total received ping series' replies of each hop. Next, step (b) in the above stage (7) is applied substituting " $d\_diff_i$ " and " $av\_del$ " instead of " $l\_diff_i$ " and " $l_i$ " respectively.
- 9- Calculate delay variation or jitter, which is represented by the standard deviation of delay " $v$ " of the total received ping series' replies of each hop. Next, step (b) of stage (7) is applied substituting " $v\_diff_i$ " and " $v_i$ " instead of " $l\_diff_i$ " and " $l_i$ " respectively.

- 10- Calculate and display the quality grading " $G_i$ " for VoIP applications at each hop, according to equation (6.5).

As indicated before, the factors which are extracted via the Matlab polynomial curve fitting were found to be: (-0.0005), (0.0283), (-0.6172), and (6.5559). Applying these extracted factors along with the packet loss percentage at the designated hop " $l\_diff$ " to equation (6.1) gives us the resultant Quality grade value " $G$ " for that hop. However, the Quality grading " $G$ " in equation (6.1) was intended to be in the range of 1 to 5, as in accordance with the grading tests that took place on the subjects. However, the grading would be 5.533716 at  $l\_diff = 1.8$ . The rounded grading value would be "6" as indicated in (table 6.2) and (figure 6.6). This is an undesired value that is outside the range intended for " $G$ ".

$l\_diff$	$G$	Rounded $G$
1.5	5.692088	6
1.6	5.63878	6
1.7	5.585991	6
1.8	5.533716	6
1.9	5.481954	5
2	5.4307	5
2.1	5.379953	5
2.2	5.329708	5
2.3	5.279964	5
2.4	5.230716	5

Table 6.2: The minimum threshold of the loss percentage " $l\_diff=1.8$ " and its effect on the quality grade " $G$ ".

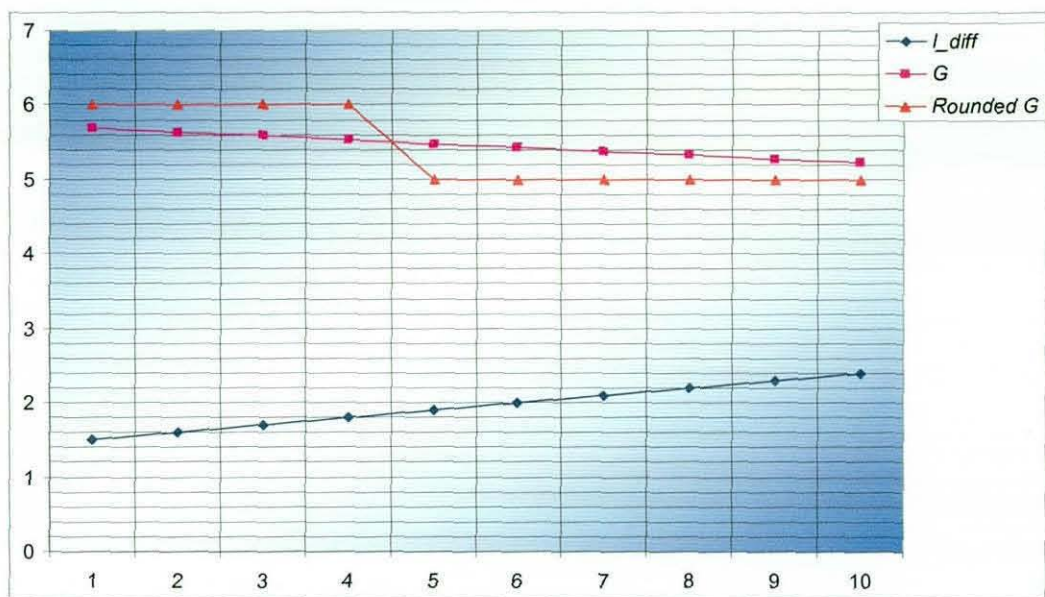


Figure 6.6: The effect of the packet loss % " $l\_diff$ " on the grading " $G$ " in reaching the upper bound of the desired  $G$  value.

Similarly, the grading value is 0.4855 at  $l\_diff = 28$ . This implies that " $G$ " would be "0" as shown in (table 6.3) and (figure 6.7).

$l\_diff$	$G$	Rounded $G$
27.5	0.586338	1
27.6	0.5667	1
27.7	0.546801	1
27.8	0.526636	1
27.9	0.506203	1
28	0.4855	0
28.1	0.464523	0
28.2	0.443268	0
28.3	0.421733	0
28.4	0.399916	0
28.5	0.377812	0

Table 6.3: The maximum threshold of the loss percentage " $l\_diff$ "=28 and its effect on the quality grade " $G$ ".



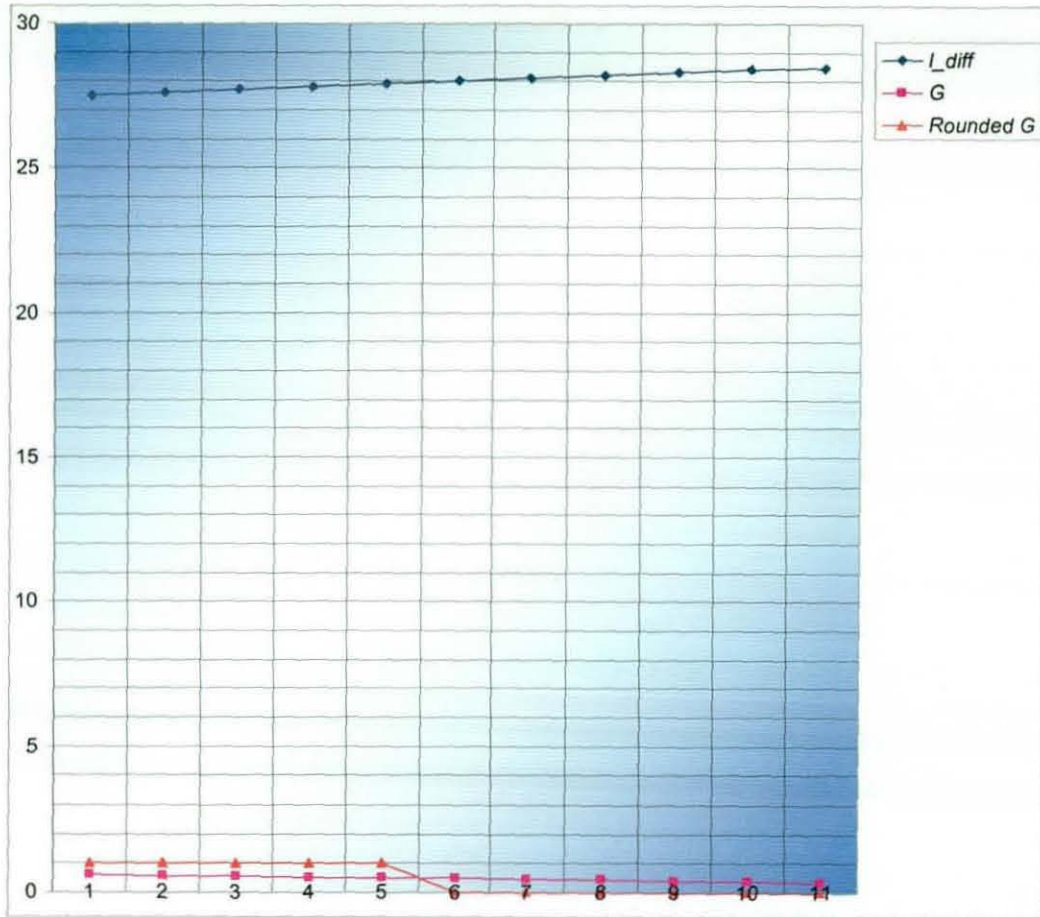


Figure 6.7: The effect of the packet loss % " $l\_diff$ " on the grading " $G$ " in reaching the lower bound of the desired  $G$  value.

Thus, to resolve this, a condition was added to the algorithm of the prototype tool, where  $(1.9 < l\_diff < 27)$  was set to ensure that " $l\_diff$ " stays within the quality grading range where " $G$ " = 5, 4, 3, 2, and 1 corresponding to loss percentage of " $l$ " = 3, 5, 9, 14, and 23. These are the loss values nominated by the subjects in the tests, for the 5 values of " $G$ ". Alternatively, Equation 6.3 can be modified to accommodate this latter condition thus limiting the resulting " $l\_diff$ " value between 1.9 and 27.

## **6.6 Conclusion**

In this chapter, a prototype tool was presented that uses the simplest techniques available in any network, so that it is applicable and usable in any network domain. This approach overcomes the practicality and universality concerns associated with the previously discussed approaches, i.e., the Active Networks and Management Information Base.

Finally, this means to determine the quality of VoIP applications was tested, and proved to be beneficial and promising with some degree of error in some cases. The results and findings will be discussed in the next chapter.

## *Chapter 7*

### **Results Analysis and Discussion**

#### **7.1 Introduction**

**T**he prototype tool was tested on the test network with controlled loss percentages to confirm the quality grading sensitivity to the expected loss value. Several tests were performed of which all gave the expected quality grade according to the given loss percentage value. Six of these tests with various loss sequences are presented here below.

#### **7.2 Test results**

The test for detecting the quality grade introduced by the intermediate hops along a network link for a VoIP connection was carried out on a lab test network. In a later stage the utility was tested albeit with limitations on the Internet to see the feasibility of its use over a WAN link.

##### **7.2.1 Test network results**

The loss values for which the curve fitting was applied were 3%, 5%, 9%, 14%, and 23% to give quality grade of 5, 4, 3, 2, and 1 respectively. Accordingly, with curve fitting the Q-grades would be classified according to the following loss-percentages:

Q-Garde	5	4	3	2	1
Loss %	(-) - 4	5 – 6	7 - 11	12 - 20	21 – (+)

Table 7.1: The adopted results of Packet loss percentage ranges and their corresponding quality grade values.

The tests were constructed on a five-hop link (Figure 6.1). The loss percentage was acquired according to the prototype tool algorithm discussed in section 6.5.

Consequently, the acquired loss percentage value at a certain hop is the accumulated loss-percentage values for all the previous hops added to the loss value at the current hop. This is the value which appears in column 1 of the tables for the conducted 6 tests (tables 7.2 - 7.7). Column 2 indicates the actual loss value at the current hop as extracted according to section 6.5.

The third column is the quality grade which is aimed to be achieved. The LNEs which are used at the five hops are then assigned with the loss percentage value which should reflect the aimed Q-grade in accordance with the results of the quality classification gathered from the subjects input in table 6.1.

Column 4 represents the resultant quality grade value calculated according to the loss percentage values as extracted and worked-out by the prototype tool. A rounded grade value of column 4 is outlined in the fifth column.



The sixth column displays the final result of the Quality grade at the designated hop after applying the condition which was mentioned at the end of section 6.5.

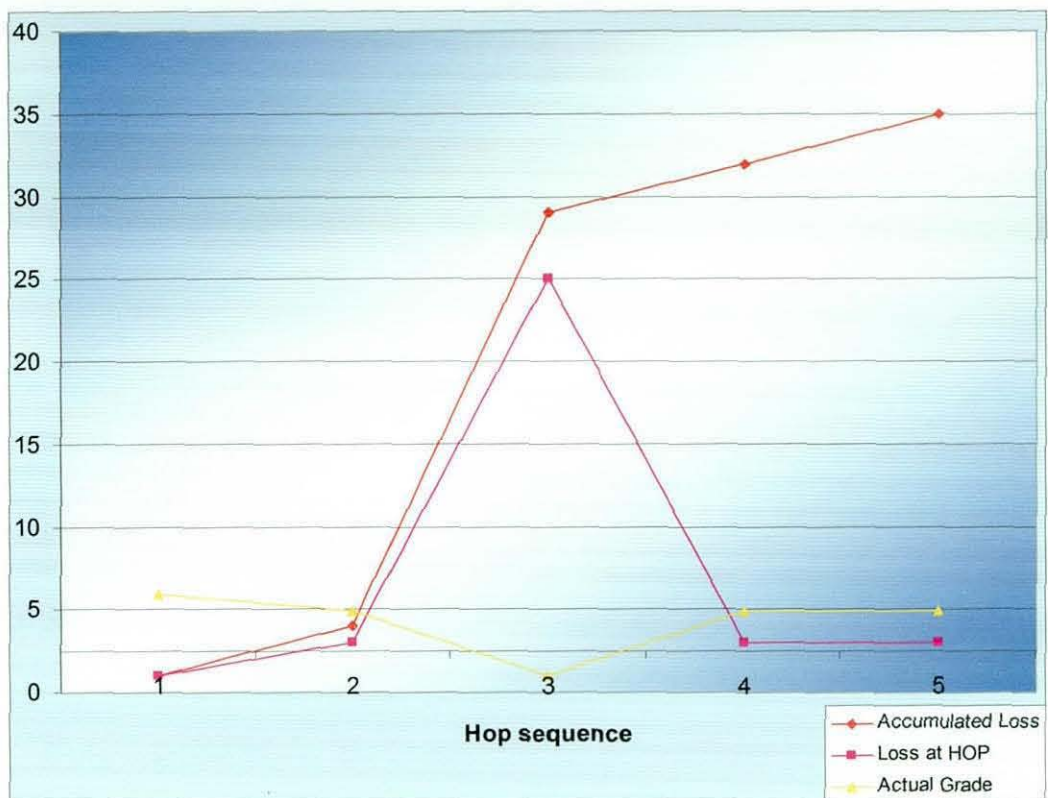
Different loss percentage values were chosen in order to look for any vulnerability in the process. The loss value was set to change very slightly, toggle, or change dramatically to examine the sensitivity of the utility in grading the quality according to loss.

In the first test, three loss percentage values (1, 3, and 4) were set at the hops along the network path, which should be represented with a grade "5" on the grading scale 1 to 5. A loss value of (%23) which should be represented by "1", the other end of the quality scale, was assigned to one of the intermediate hops on the same path.

The result shows that the relevant quality grade values were successfully detected.

Test 1		Accumulated Packet loss	Loss at Hop	Aimed Q Grade	Actual Grade	Rounded Grade	Grade with $1.9 <  _{\_diff} < 27$
Hop 1	Rainbow	1	1	5	5.9665	6	5
Hop 2	Yellow	4	3	5	4.9455	5	5
Hop 3	Orange	27	23	1	1.2475	1	1
Hop 4	Pink	31	4	5	4.5079	5	5
Hop 5	Purple	35	4	5	4.5079	5	5

Table 7.2



Graph (7.1): Outlining the relation between the loss and the grade in test 1.

HOP COUNT	I P A D D R E S S	QUALITY GRADE
1	192.168.11.1	5
2	192.168.12.1	5
3	192.168.13.1	1
4	192.168.14.1	5
5	192.168.15.1	5

Hops which have lower Quality Grade:

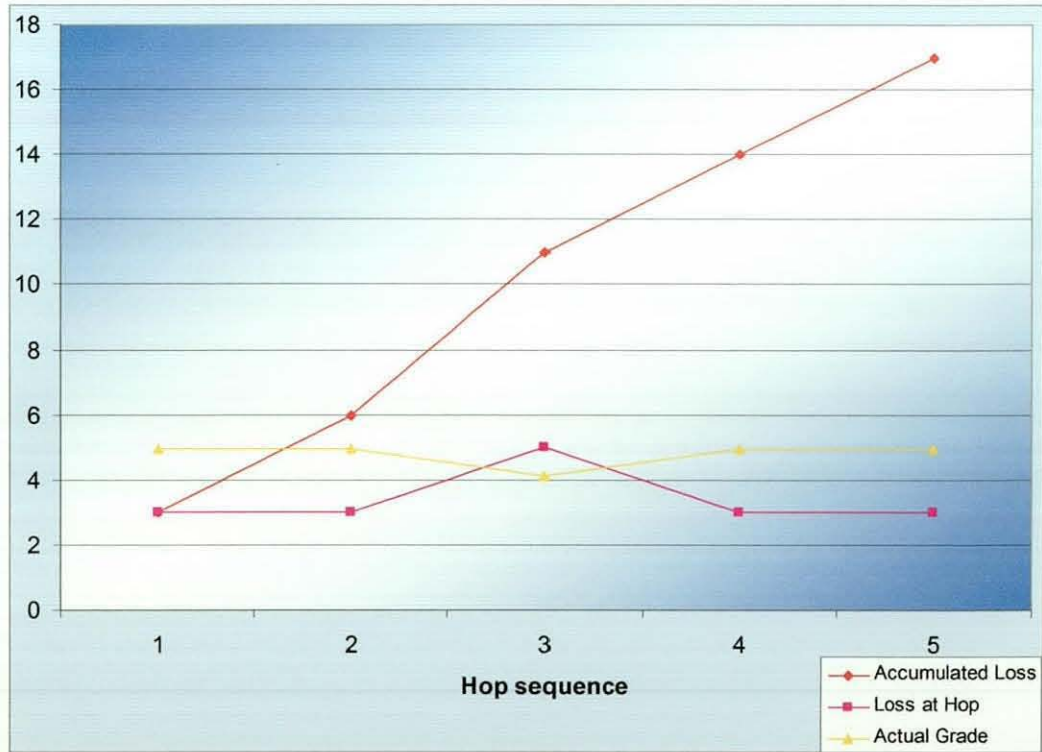
Hop: (3) Host name: (orange) Quality Grade= (1)

Figure 7.1: The result from the prototype utility for the first test.

Similar to the first test, the second test had a loss value of a different Q-grade put among the other equal grade values. This time the grade value was the next one in the grading scale. Still, it was detected and not mistaken for the same grade value as the others.

Test 2		Accumulated Packet loss	Loss at Hop	Aimed Q Grade	Actual Grade	Rounded Grade	Grade with $1.9 < I\_diff < 27$
Hop 1	Rainbow	4	4	5	4.5079	5	5
Hop 2	Yellow	8	4	5	4.5079	5	5
Hop 3	Orange	13	5	4	4.1149	4	4
Hop 4	Pink	17	4	5	4.5079	5	5
Hop 5	Purple	18	1	5	5.9665	6	5

Table 7.3



Graph (7.2): Outlining the loss and grade in test 2.



HOP COUNT	I P A D D R E S S	QUALITY GRADE
1	192.168.11.1	5
2	192.168.12.1	5
3	192.168.13.1	4
4	192.168.14.1	5
5	192.168.15.1	5

Hops which have lower Quality Grade:

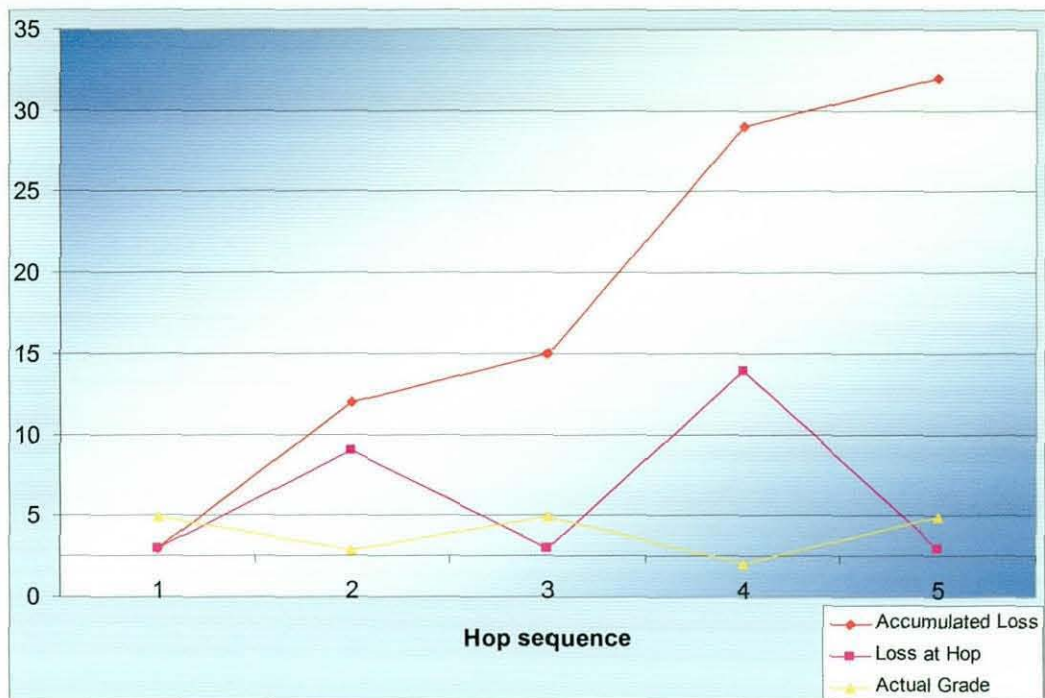
Hop: (3) Host name: (orange)      Quality Grade= (4)

Figure 7.2: The result of the prototype utility for the second test.

In the third test, in table 7.4 and figure 7.3, the aim was to detect a Q-grade toggling between two degraded values of "3" and "2" while maintaining the best quality value "5" otherwise.

Test 3		Accumulated Packet loss	Loss at Hop	Aimed Q Grade	Actual Grade	Rounded Grade	Grade with 1.9<I_diff<27
Hop 1	Rainbow	3	3	5	4.9455	5	5
Hop 2	Yellow	10	7	3	3.4507	3	3
Hop 3	Orange	13	3	5	4.9455	5	5
Hop 4	Pink	26	13	2	2.2165	2	2
Hop 5	Purple	29	3	5	4.9455	5	5

Table 7.4



Graph (7.3): Third test.

HOP COUNT	I P A D D R E S S	QUALITY GRADE
1	192.168.11.1	5
2	192.168.12.1	3
3	192.168.13.1	5
4	192.168.14.1	2
5	192.168.15.1	5

Hops which have lower Quality Grade:

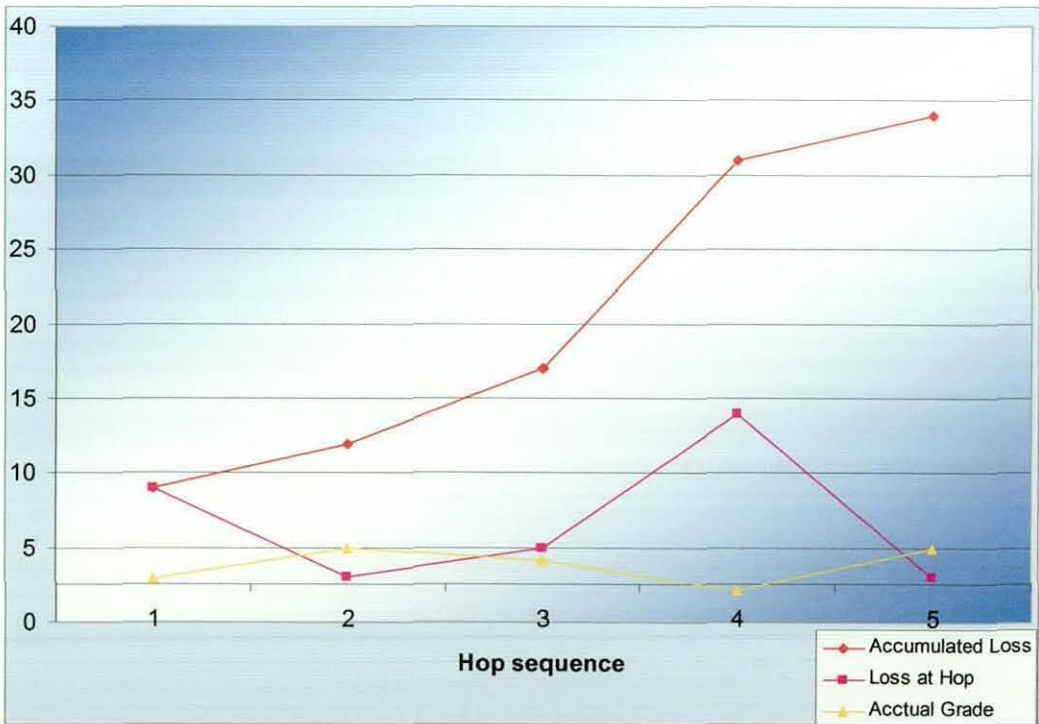
Hop:(2) Host name:(yellow)      Quality Grade=(3)  
Hop:(4) Host name:(pink)      Quality Grade=(2)

Figure 7.3: The result of the prototype utility for test (3).

Test four (table 7.5 & figure 7.4) and test five (table 7.6 & figure 7.5) had combinations of toggle, gradual and large changes between the quality grades of the five hops.

Test 4		Accumulated Packet loss	Loss at Hop	Aimed Q Grade	Actual Grade	Rounded Grade	Grade with $1.9 < I\_diff < 27$
Hop 1	Rainbow	9	9	3	2.9289	3	3
Hop 2	Yellow	13	4	5	4.5079	5	5
Hop 3	Orange	18	5	4	4.1149	4	4
Hop 4	Pink	30	12	2	2.3607	2	2
Hop 5	Purple	34	4	5	4.5079	5	5

Table 7.5



Graph (7.4): Fourth test.

HOP COUNT	I P A D D R E S S	QUALITY GRADE
1	192.168.11.1	3
2	192.168.12.1	5
3	192.168.13.1	4
4	192.168.14.1	2
5	192.168.15.1	5

Hops which have lower Quality Grade:

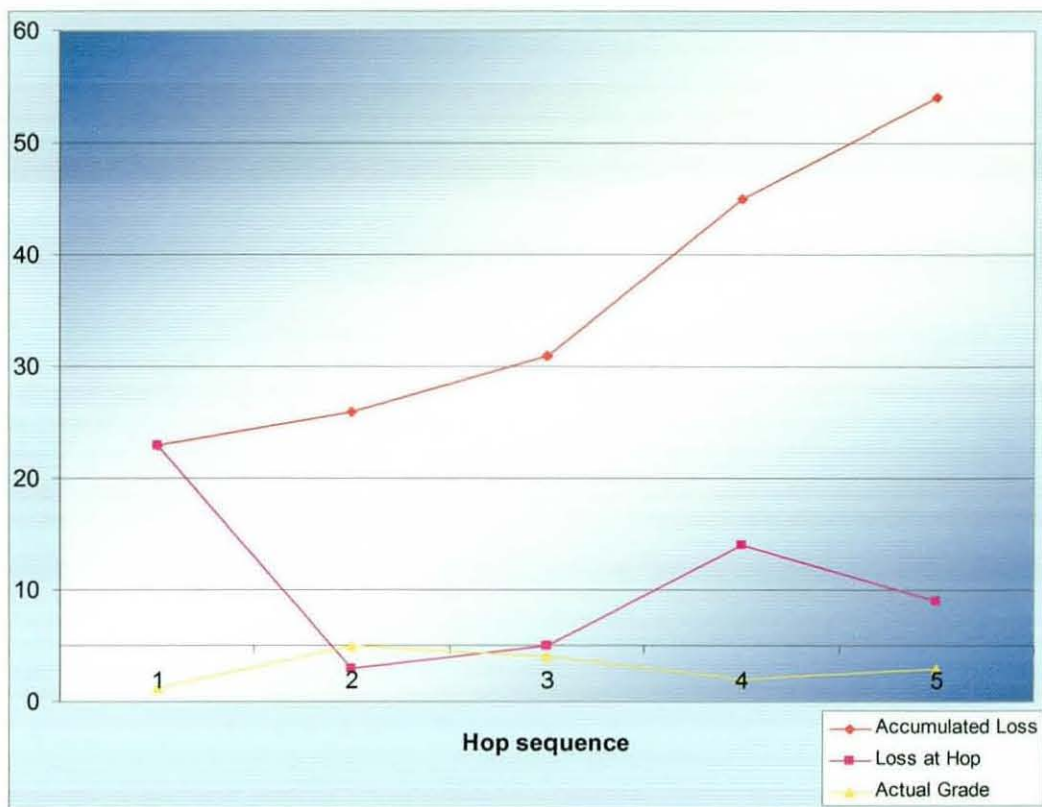
Hop: (1) Host name: (rainbow)      Quality Grade= (3)  
Hop: (3) Host name: (orange)      Quality Grade= (4)  
Hop: (4) Host name: (pink)      Quality Grade= (2)

Figure 7.4: The result of the prototype utility for test (4).

Test 5		Accumulated Packet loss	Loss at Hop	Aimed Q Grade	Actual Grade	Rounded Grade	Grade with $1.9 < I\_diff < 27$
Hop 1	Rainbow	23	23	1	1.2475	1	1
Hop 2	Yellow	26	3	5	4.9455	5	5
Hop 3	Orange	31	5	4	4.1149	4	4
Hop 4	Pink	45	14	2	2.0899	2	2
Hop 5	Purple	54	9	3	2.9289	3	3

Table 7.6





Graph (7.5): Fifth test.

HOP COUNT	I P A D D R E S S	QUALITY GRADE
1	192.168.11.1	1
2	192.168.12.1	5
3	192.168.13.1	4
4	192.168.14.1	2
5	192.168.15.1	3

Hops which have lower Quality Grade:

Hop: (1) Host name: (rainbow)      Quality Grade= (1)  
Hop: (3) Host name: (orange)      Quality Grade= (4)  
Hop: (4) Host name: (pink)      Quality Grade= (2)  
Hop: (5) Host name: (purple)      Quality Grade= (3)

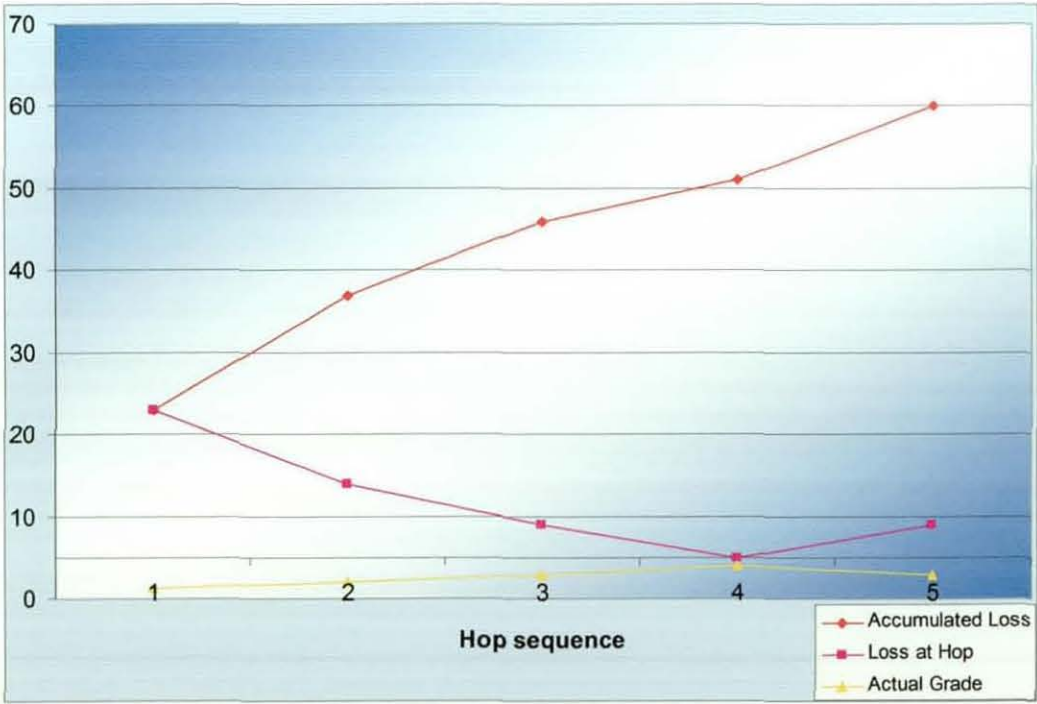
Figure 7.5: The result of the prototype utility for test (5).



The sixth test examined the detection of the gradual change in quality from "1" to "4" and then toggling back to the next previous value "3" (Table 7.7, Figure 7.6).

Test 6		Accumulated Packet loss	Loss at Hop	Aimed Q Grade	Actual Grade	Rounded Grade	Grade with $1.9 <  _{\_diff} < 27$
Hop 1	Rainbow	23	23	1	1.2475	1	1
Hop 2	Yellow	37	14	2	2.0899	2	2
Hop 3	Orange	46	9	3	2.9289	3	3
Hop 4	Pink	51	5	4	4.1149	4	4
Hop 5	Purple	60	9	3	2.9289	3	3

Table 7.7



Graph (7.6): Sixth test.

HOP COUNT	I P A D D R E S S	QUALITY GRADE
1	192.168.11.1	1
2	192.168.12.1	2
3	192.168.13.1	3
4	192.168.14.1	4
5	192.168.15.1	3

Hops which have lower Quality Grade:

```
Hop: (1) Host name: (rainbow)      Quality Grade= (1)
Hop: (2) Host name: (yellow)      Quality Grade= (2)
Hop: (3) Host name: (orange)      Quality Grade= (3)
Hop: (4) Host name: (pink)        Quality Grade= (4)
Hop: (5) Host name: (purple)      Quality Grade= (3)
```

Figure 7.6: The result of the prototype utility for test (6).

Tests 1 to 6 have demonstrated that the prototype tool is capable of detecting the quality of service at each hop along the network path, provided that all the hops were identified and responded, an issue which is discussed in section 7.4 "Quality of undetermined hops".

### 7.2.2 Internet VoIP test

A limited test for the prototype utility was carried out on the Internet. The limitation is from the perspective that there was no established VoIP connection between the two ends of the network. Nevertheless, the test was intended to discover if the utility is able to categorise the in-path hops according the "1" to "5" grading scale.

The test showed that the utility detects the quality and classifies a grade for this quality at every hop along the network path. However, it was not actually possible to verify that the classified grade would reflect the actual quality of a real VoIP connection over this path.

Figure (7.7) demonstrates the resultant display of a test for VoIP applications grading using this Utility. It concludes with itemisation of the defective hops which the quality grading measured to be less than "5". It also lists their hop number, host name, and the quality grade offered. The figure indicates that the quality is degraded at both hops "14" and "19". This means that those two hops will be a reason for the degradation of a voice conversation whenever a connection is established via those two hops.

HOP COUNT	I P A D D R E S S	QUALITY GRADE
1	158.125.51.254	5
2	131.231.13.147	5
3	131.231.13.5	5
4	158.125.8.1	5
5	212.219.212.49	5
6	212.219.212.21	5
7	146.97.40.21	5
8	146.97.35.13	5
9	146.97.33.2	5
10	146.97.35.226	5
11	213.206.159.101	5
12	213.206.128.104	5
13	144.232.9.163	5
14	144.232.7.114	3
16	212.138.64.66	5
19	212.26.18.17	3

Hops which have lower Quality Grade:

Hop:(14) Host name:(sl-gw22-nyc-9-0.sprintlink.net) Quality Grade=(3)  
Hop:(19) Host name:(fe5-0-0.ruh-natl.isu.net.sa) Quality Grade=(3)

**Figure 7.7:** The Internet test result of hop quality grading for Voice over IP application. Both hops (14) and (19) registered a lower quality grading.

### 7.3 Suggested Solution

Some measures could be considered to overcome the degradation encountered on a certain network path. There are some options available including the upgrade of the router which has been periodically responsible for the quality degradation of a designated network path.

The following technique is also a suggestion to bypass the degraded router in the network. Consider that a VoIP call is established between two end points that are connected by two alternate paths (Figure 7.8)

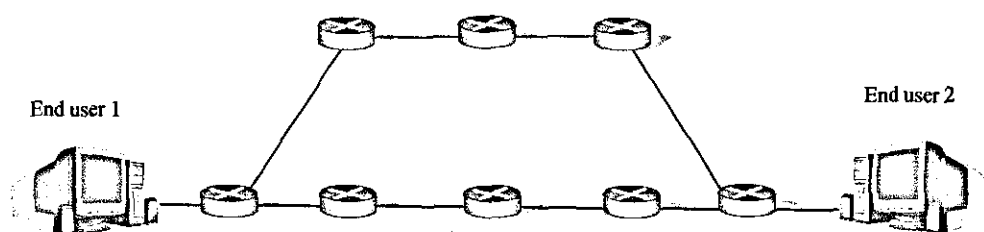


Figure 7.8: Path and link restoration-like technique to bypass the undesired hop.

If a VoIP call is established between the two end points via the black path then the application is monitoring the nodes only on this path. However, if the application can also monitor the nodes along the red path and there is a degradation of QoS on the black path, the application may transparently re-establish the connection via the red path. Thus, the application needs to determine more than 1 (possibly disjoint or partially disjoint) paths between the two end points (possible using a routing protocol or also by inspecting the routing tables on the nodes) and the signaling mechanism should ensure a transparent switchover from one path to another if QoS on the current path drops below a specified threshold.

This technique is similar to path and link restoration in optical ring and mesh networks [Dongyun 2000] and [Ramamurthy 1999]. These networks employ

algorithms to determine paths within the networks. Then the nodes, links and paths are monitored for faults and in the case of a fault, the traffic is switched to alternate paths.

#### **7.4 Other multimedia applications tests**

The hop quality grading utility designed as a prototype tool in this research was initially intended to identify the quality of service provided at each hop according to the grading of Voice over IP applications. Nevertheless, in later work, the feasibility of detecting the quality grading for other multimedia network applications was examined. The QoS level offered at each hop for virtual reality games played between two players across the Internet was investigated. Similarly, the prototype tool was also utilised for the measurement of quality grading for web browsing at each hop on the network path.

The Hop Grading Utility was tested on the Internet. The tests included the quality grading for both aforementioned applications; VR-games and Web-browsing.

##### **7.4.1 VR-games test**

The representations of coefficients that are to be implemented for the grading equations concerning VR-games and Web-browsing were borrowed from previous work on the subject [Oliver 1999].

Unlike VoIP which depended heavily on packet loss, both VR-game and Web-browsing were effected by all three elements; fixed-delay, delay-variation, and packet-loss. The parameters used to represent these three elements in the grading utility were; average delay of the total ping messages, the standard deviation of the total ping messages, and the ping messages' loss-percentage value, respectively.

In figure (7.9), six hops failed to offer good quality for VR-game applications along the designated network path.

HOP COUNT	IP A D D R E S S	QUALITY GRADE
1	158.125.51.254	5
2	131.231.13.147	5
3	131.231.13.5	5
4	158.125.8.1	5
5	212.219.212.49	5
6	212.219.212.21	5
7	146.97.40.21	5
8	146.97.35.13	5
9	146.97.33.2	5
10	146.97.35.226	5
11	195.50.116.201	5
12	212.187.129.129	5
13	4.68.128.102	5
19	12.123.195.214	4
20	12.119.139.66	3
21	161.142.173.90	3
22	161.142.25.86	3
23	61.6.17.58	3
24	161.142.0.214	3

Hops which have lower Quality Grade:

Hop: (19) Host name: (gar1-p370.sn1ca.ip.att.net) Quality Grade=(4)  
Hop: (20) Host name: (12.119.139.66) Quality Grade=(3)  
Hop: (21) Host name: (ge5-0.bkj90.jaring.my) Quality Grade=(3)  
Hop: (22) Host name: (pos0-0.mlk90.jaring.my) Quality Grade=(3)  
Hop: (23) Host name: (pos1-0.mlk15.jaring.my) Quality Grade=(3)  
Hop: (24) Host name: (s0-0.btp15.jaring.my) Quality Grade=(3)

**Figure 7.9:** The test result of hops quality grading for a virtual reality game connection.

#### 7.4.2 Web-browsing test

In this sample test, a hop grading detection was carried out on the network path to the end-target; [www.lanka.net](http://www.lanka.net). The test was carried out several times. Hop (15) registered lower quality in all tests. Figures (7.10) and (7.11) are two of those tests.

HOP COUNT	IP A D D R E S S	QUALITY GRADE
1	158.125.51.254	5
2	158.125.8.1	5
3	212.219.212.49	5
4	212.219.212.21	5
5	146.97.40.21	5
6	146.97.35.13	5
7	146.97.33.2	5
8	146.97.35.226	5
9	195.50.116.201	5
10	212.187.129.209	5
11	212.187.128.57	5
12	4.68.128.106	5
13	64.159.17.72	5
14	64.152.40.2	5
15	66.192.240.33	4
16	168.215.55.49	5
17	66.192.248.141	5
18	207.170.250.6	5
19	198.60.194.1	5
20	198.60.194.34	5
21	198.60.194.34	5

Hops which have lower Quality Grade:

Hop: (15) Host name: (core-01-ge-0-2-0-0.nycl.twtelecom.net) Quality Grade=(4)

**Figure (7.10):** Results of hop quality grading for a web-browsing application.

HOP COUNT	IP A D D R E S S	QUALITY GRADE
1	158.125.51.254	5
1	158.125.51.254	4.909640
2	158.125.8.1	5
2	158.125.8.1	4.909939
3	212.219.212.49	5
3	212.219.212.49	4.908573
4	212.219.212.21	5
4	212.219.212.21	4.909939
5	146.97.40.21	5
5	146.97.40.21	4.891364
6	146.97.35.13	5
6	146.97.35.13	4.907761
7	146.97.33.2	5
7	146.97.33.2	4.909939
8	146.97.35.226	5
8	146.97.35.226	4.907700
9	195.50.116.201	5
9	195.50.116.201	4.904406
10	212.187.129.209	5
10	212.187.129.209	4.909725
11	212.187.128.57	5
11	212.187.128.57	4.507748
12	4.68.128.106	5
12	4.68.128.106	4.909939
13	64.159.17.72	5
13	64.159.17.72	4.908402
14	64.152.40.2	5
14	64.152.40.2	4.896684
15	66.192.240.33	4
15	66.192.240.33	4.423372
16	168.215.55.49	5
16	168.215.55.49	4.908743
17	66.192.248.141	5
17	66.192.248.141	4.793386
18	207.170.250.6	5
18	207.170.250.6	4.909939
19	198.60.194.1	5
19	198.60.194.1	4.909939
20	198.60.194.34	5
20	198.60.194.34	4.909939
21	198.60.194.34	5
21	198.60.194.34	4.909939

Hops which have lower Quality Grade:

Hop:(15) Host name:(core-01-ge-0-2-0-0.nycl.twtelecom.net) Quality Grade=(4)

**Figure (7.11):** Another test to the same end target as that of fig(7.9), with the un-rounded value of "G" revealed.



In the results shown in figures 7.10 and 7.11, hop (15) registered lower quality in both tests, although the two tests were carried out at two different times. The test in figure (7.10) was run on Wednesday at 4:25 pm. The test in figure (7.11), was run on Thursday at 7:47 pm. Consequently, the results suggest that hop (15) which has the host name (core-01-ge-0-2-0-0.nycl.twtelecom.net) and IP address (66.192.240.33) always has a lower performance which affects the overall network connection which passes through this hop.

In a different experiment, several tests were performed on the network route between Loughborough University UK and Stanford University USA at different times of the day. In all the tests, the prototype tool ranked all hops as having the optimum quality grade for Web-browsing applications. However, on only one occasion, one of the hops gave lower grading than the expected optimum. This is an indication that this is just a one-off case and not the overall rule for the specific hop or that network path as was the case in the previous results of figures (7.10) and (7.11).

### **7.5 Quality of Undetermined Hops**

Occasionally, some of the hops which were detected by the Traceroute failed to respond to the Ping messages sent to it. Accordingly the network parameter values of these hops could not be obtained to determine its quality grade.

Traceroute on Linux, which is the platform used for the experiments in this research, "builds its own UDP packets to send and reads ICMP replies." [Stevens 1998]. Although Traceroute in Linux uses UDP by default for probing, it can be forced to use ICMP for probing by a command line option if required.

Traceroute works by progressively increasing the TTL field in the IP layer from 1 upwards for its probes. So for the first probe, TTL is set to 1 and it expires when the packet reaches the first node, where the datagram causes the first encountered hop to return an ICMP "Time exceeded" error. Then for the 2nd probe, TTL is set to 2 which eventually expire at the 2nd node. Therefore, at every node after the host executing the traceroute the TTL value goes to 0 and that node replies to the host with an ICMP Time To Live Exceeded message. This process is repeated until the probe reaches the destination node [Stevens 1998].

In regards to Ping, it is based on ICMP echo request and ICMP echo reply messages which are in short known as ping messages. Hackers can use these to (with some accuracy) chart the topology of a sub-network (i.e., a subscriber's Intranet), which includes the number and types of computers connected to the subscriber's network, number of network switches and gateways. Subsequently, the hackers can plan their attacks on those networks. Therefore for security reasons, it is increasingly common nowadays that network managers configure their network devices such that these devices ignore incoming ICMP messages at the firewall. Hence replies to ICMP echo requests are not generated by these hops. So a hacker may only be able to ping the access router, but cannot ping computers on the other side of the access router and the firewall.

If traceroute probes use ICMP echo request messages, such as the case in Windows Tracert utility, and a node between the source and the destination has been configured to drop ICMP echo request reply messages, no Time To Live Exceeded messages will be returned from the node blocking the pings and the effect will be that of lost pings. However, UDP messages for traceroute are/can not be blocked as they may use random higher order port numbers and that may mean disruption to other services that use UDP and that range of port numbers. The remaining principle stays the same, i.e., TTL value is increased progressively from 1 and the node at which it reaches 0, Time To Live Exceeded message is returned.

Unlike Traceroute which uses UDP datagrams, Ping messages are ICMP packets and not UDP datagrams. Accordingly certain hops which are detected by

Traceroute are invisible to the Ping program because they do not reply to the incoming ping messages. These hops are near the destination node because as we start to approach the periphery of the network, we enter the subscriber's domain, which is imposing blockage on the ICMP echo-request messages.

Accordingly, in the prototype tool an alteration was added, whereby the phrase "Undetermined" appears next to the hop which was detected by Traceroute but failed to reply to Ping messages. Thus, network parameters for that designated hop are not available to determine its quality. Accordingly, this means that the next responding hop includes the parameters' values of itself as well as that of the undetermined hops.

Although this may seem to suggest that this approach would draw short of determining the precise quality at each hop, two arguments should be taken into consideration;

1. This usually occurs close to the destination node, thus near the edge of the network path.
2. Both the Undetermined hop and the responding hop may fall under the same network management domain, thus still falling under their responsibility to support it, which meets the objective of this work whereby the service provider of the part of the network route where deficiencies occur is known.

## **7.6 Conclusion**

The tests carried out on all three VoIP, VR-games, and Web-browsing applications proved to be promising. In a few cases, not all hops were detected. That is because in some cases, the network managers administrating the undetermined routers prevent them from replying to ICMP ping messages for security reasons. Hence the grading utility would fail to register the quality for all hops on that route.

To avoid falling into certain day-time patterns, the tool should be applied to examine a specified network route at different times of the day, so as to conclusively confirm the condition and behaviour of the hops on that route. As the examples discussed in the last two paragraphs of section (7.4.2), a router cannot be classified as a lower quality hop, unless it was repeatedly registered so by the tool during different times. This then proves that there is a problem with the designated hop rather than for any other reason.

Moreover, from the discussion of the opposing arguments in section (7.4), the prototype tool put forward in this research proves to be effective and a useful tool for both network management as well as research.

## *Chapter 4*

### **The Grading Algorithm**

#### **4.1 Introduction**

**T**he main objective of this research is to determine the quality of service of a given real-time network multimedia application at every hop along the network path with the application session is utilising. Accordingly, a grading algorithm has to be established to identify the various levels of QoS for this application. Consequently, the QoS for the application at each hop will be graded according to this algorithm.

In order to determine the quality of service for the required application and categorise this quality of service by assigning a grade for it, the grading algorithm must take into consideration the application being considered. This is because applications, in general, differ in their network resource requirements and the parameters which play a role in the enhancement or degradation of their performance.

Real-time applications differ in their networking requirements from those of non real-time applications'. FTP, as a non real-time application, can tolerate delay since no specific demands of timing requirements are placed upon packet delivery, although performance, in general, can be improved if the delay is reduced. Moreover, different real-time network applications vary in their networking demands as well. While a typical Internet videoconferencing application, for instance, is mainly affected by packet loss and variation in packet delay, web browsing applications are affected by fixed delay as well as packet

loss and variable delay. Thus, the parameters that play a role in determining the performance of a certain application over the network have to be identified. These are used as metrics to determine the quality of that application over any given network.

The basic principle of developing a grading algorithm for an application is to calibrate the grading by establishing a controlled connection between two end-nodes and then monitoring the quality of service at these end-nodes. Variation in the controlled traffic is introduced on the hops that are spread along the network route. Furthermore, the relationship between the change in network parameters and grading of the quality of service has to be mathematically established. Thus, different grades are assigned according to the different traffic parameter combinations of delay and loss. Consequently, in the reverse approach, if traffic parameters can be collected over any given network, the quality of service of the application executing on that network can be determined a priori using the grading method parameterised by the collected network statistics.

The steps by which the grading method was developed are explained in the following sections.

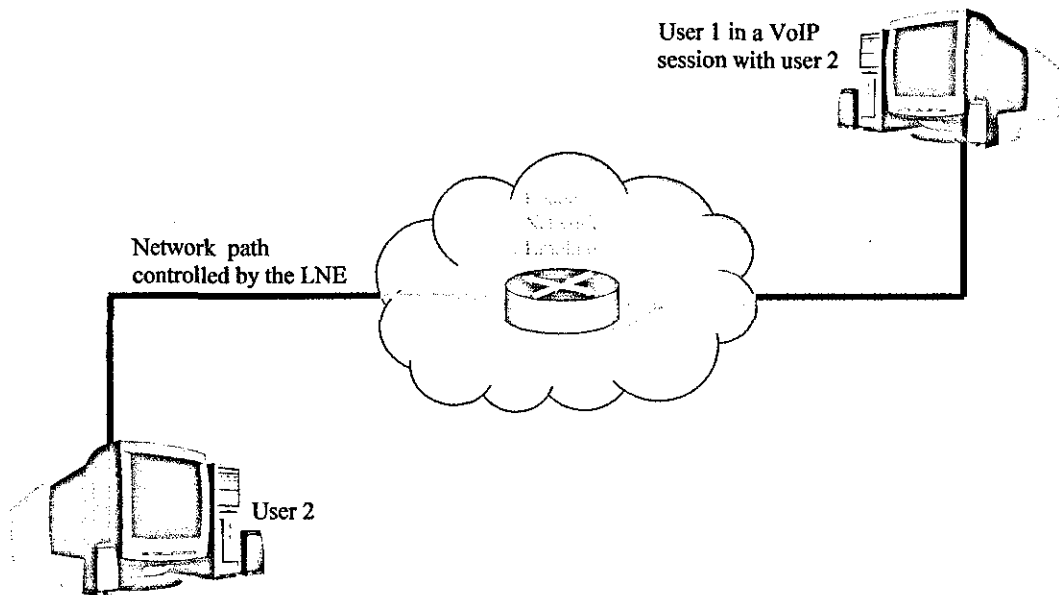
## **4.2 Calibration of quality of service grading**

Knowing what the applications require is essential to pave the way to measuring their performance. Specifically, networking parameters which affect the application's performance have to be identified, and moreover, the extent of their

effect needs to be measured in order to have a precise categorisation grading of the quality of service for the application in question.

In this research, Voice over IP is used as the application for which the grading of the performance of hops was calibrated. To grade the quality of service, the procedure had to be performed on a controlled network set-up where the network traffic, as well as network elements, are controlled and altered as desired and there is no interference of the external network traffic. Microsoft NetMeeting<sup>®</sup> 3.01 was used as a voice application. The video send-and-receive options were disabled so as to limit the application to audio communication only. The auto-gain control was also disabled to ensure receipt of the original audio quality without any quality enhancements at the receiving-end. The voice codec used for the experiments was based on G.723.1. The codec streams are then passed via RTP to be carried over UDP.

Two end-users were connected through a network station that had a PC-based LNE (Loaded Network Emulator) executing on it to emulate real network conditions and to introduce the desired heavy or mild traffic parameters of loss and delay (Figure 4.1). This PC basically acted to cause a loaded network, but without adding additional traffic. The quality of service that it provides can be controlled by varying the above mentioned traffic parameters for a loaded network.



**Figure 4.1:** Implementation of the LNE (Loaded Network Emulator). It emulates real network conditions.

The Netmeeting<sup>®</sup> connection between the two end users was established through a congested network emulated using the LNE, as detailed in the next section.

#### 4.2.1 LNE (Loaded Network Emulator)

The LNE is used to emulate the load experienced on networks by setting up and altering the three main parameters that influence the quality of service for most applications over a given network. These parameters include fixed delay, delay variation and packet loss. The LNE is designed to drop or delay the packets that pass through it according to the user's desired parameters of network load. Hence it offers better accuracy of emulating loaded networks than traffic generators since



it does not generate or add any extra traffic to the existing network traffic, besides one can specify the exact parameters of loss and delay to be experienced on the network. In the LNE, packet loss and delay are applied according to Gaussian distribution [Oliver 2000].

It was found that fixed delay has no effect on the quality of the sound, although it does affect usability. This was also established in earlier work [Oliver 1999] for video conferencing applications. The voice processing and communication components of these applications are similar in behaviour to other voice applications. Hence fixed delay was not considered in the later experiments. Nevertheless it must be noted that high levels of delay may affect VoIP usability.

In order to apply packet loss and delay variation to the network traffic passing through the router which is accommodating the LNE, IP-Tables were exploited to set-up LNE parameters. IP-Tables are commonly used to control the traffic that passes through a Linux based computer by applying IP packet filter rules in the Linux kernel. Different tables can be defined which contain several built-in sets of rules or chains to be enforced on the passing traffic. Furthermore users can also define chains or rules. In this case the rules are defined and enforced by the LNE, which resides in user space.

IP-Tables thus control the queuing strategy at the router. The queuing architecture of IP-tables is shown in (Figure 4.2). When there is no specific queuing strategy i.e. FIFO, the incoming packets at the "Pre-routing" queue are forwarded to the "Post-routing" queue through the "Forward" queue. Thus, the packets bypass any processing in the "User Space". However, if there is a change to the queuing strategy, or the packets need to be processed in a certain way, for example, if there is a need to time-stamp the packets, the packets have to be passed to the "User

Space" through the "Input" queue before forwarding them to the "Post-routing" queue through the "Output" queue.

The IP-tables queuing strategy rule is thus altered to force network traffic to pass into user-space. Here, the forwarded packets are affected by the loss and jitter parameters introduced by the LNE. Packets that are not dropped by the LNE are passed on to the output queue.

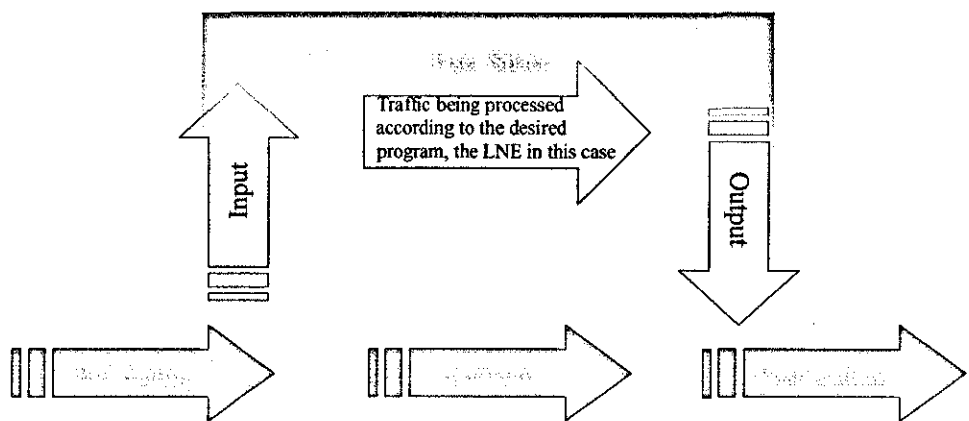
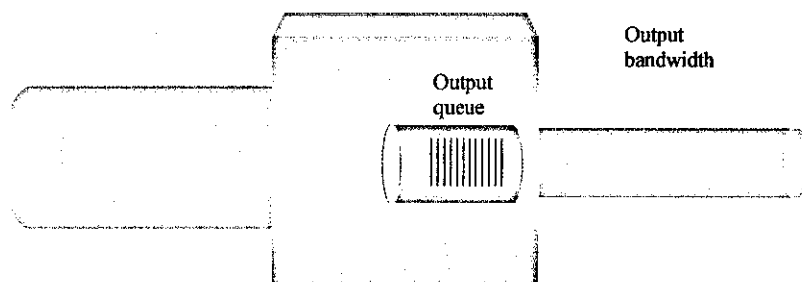


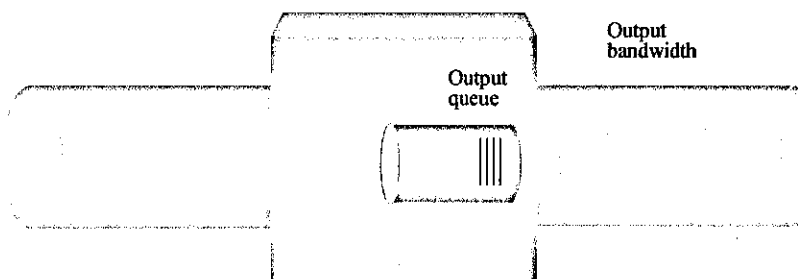
Figure 4.2: IP-Tables queuing layout.

The LNE simply drops packets according to the desired loss-percentage value. As for delay variation, this is introduced by controlling the output bandwidth that can increase and decrease between two specified maximum and minimum bandwidth parameters. Thus the number of packets in the output queue will increase when the bandwidth window gets narrower (Figure 4.3a), and decrease when the

bandwidth window gets wider (Figure 4.3b). The maximum size of the output queue was set to be large enough to ensure that no packets are dropped when the packets in the queue increase due to bandwidth window decrement. Hence variation in delay is controlled, emulating jitter over a real network.



(a)



(b)

**Figure 4.3:** Bandwidth window variation as a means to emulate jitter.

### **4.3 Quality classification**

In order to classify the quality of the application under consideration while changing network parameters, a semi-formal evaluation for voice quality was carried out in the laboratory on several subjects. Accordingly, five quality grade values were specified. These quality grade values of 5, 4, 3, 2, and 1 [Daumer 1982] relate to excellent, good, fair, bad, and poor respectively in accordance with various values of loss and jitter.

To ensure an unrecognised conversation which none of the subjects were familiar with, a talk-radio station was chosen as the source of voice transmission. The radio was connected directly to the PC's audio card through the mic-port. On the other end-user station, the subjects listened to the conversation transmitted from the source using headphones.

For each of the different parameters of loss and delay, the subjects were asked to fill in the classification form. This process was repeated for all the subjects. Accordingly, an average of the results of all the evaluation forms was calculated to determine a common agreed-upon classification according to the introduced network parameters.

### **4.4 Grading algorithm**

Through the monitoring and identification of quality grades categorised according to the changes occurring due to the effect of the varying low level network parameters, a mathematical model was developed that outlines the effect of

variation of the network parameters on the quality grade of an application using low level data statistics.

Since quality grading is a function of loss and jitter [Oliver 1999], a simple algebraic general form equation that can be applied to any loss or delay variation value to obtain a grading was developed. The significant of this equation is that it uses low-level statistics of the simple network parameters of loss and jitter. Thus quality grading "G" was set to be:

$$G(l, v) = f_0 + f_1l + f_2l^2 + f_3v + f_4lv + f_5l^2v + f_6v^2 + f_7lv^2 + f_8l^2v^2 \quad (4.1)$$

where;  $l$  = Loss

$v$  = Jitter

$G$  = Grade

Equation 4.1 provides an empirical grading of application performance according to the change in network loading conditions. The application quality grading "G" is a function of packet loss "l" and variation in delay "v". Hence "G" value changes in accordance with the change in "l" and "v", thus the quality grading is affected by loss and jitter as reflected by equation 4.1.

The factors in equation (4.1) are constants and independent of network type and network topology, but depend on application type and traffic density.

Partially differentiating equation (4.1) with respect to "l" to eliminate packet loss, will produce an equation containing one variable:

$$\frac{\partial G}{\partial l} = f_1 + 2f_2l + f_4v + 2f_5lv + f_7v^2 + 2f_8lv^2 \quad (4.2)$$

Second partial differentiation with respect to "l":

$$\frac{\partial^2 G}{\partial l^2} = 2f_{2l} + 2f_{5l}v + 2f_{8l}v^2 \quad (4.3)$$

Now, going back to equation 4.1 and by fixing "v" i.e. "v = v<sub>1</sub>",

$$G|_{v=v_1} = f_0 + f_1l + f_2l^2 + f_3v_1 + f_4lv_1 + f_5l^2v_1 + f_6v_1^2 + f_7lv_1^2 + f_8l^2v_1^2 \quad (4.4)$$

$$G|_{v=v_1} = (f_0 + f_3v_1 + f_6v_1^2) + (f_1 + f_4v_1 + f_7v_1^2)l + (f_2 + f_5v_1 + f_8v_1^2)l^2$$

$$G|_{v=v_1} = b_{0l} + b_{1l}l + b_{2l}l^2 \quad (4.5)$$

where

$$b_{0l} = f_0 + f_3v_1 + f_6v_1^2 \quad (4.6)$$

$$b_{1l} = f_1 + f_4v_1 + f_7v_1^2 \quad (4.7)$$

$$b_{2l} = f_2 + f_5v_1 + f_8v_1^2 \quad (4.8)$$

equation 4.5 can be written in a more general form:

$$G|_{v=v_1} = b_{0i} + b_{1i}l + b_{2i}l^2 \quad (4.9)$$

Where "b's" are constant coefficients.

The "b" coefficients in equation (4.10) can be evaluated by having 3 equations representing 3 different combinations of "G" and "l". However, the values of "b's" will represent only the given combination cases. Hence, we make a general

universal case representation, where “ $b$ ’s” can be found in an approximate way using the elementary curve fitting technique.

Differentiating equation (4.9) with respect to “ $l$ ”:

$$\left. \frac{dG}{dl} \right|_{v=v_1} = b_1 + 2b_2 l \quad (4.10)$$

Second derivative of equation (4.9):

$$\left. \frac{d^2 G}{dl^2} \right|_{v=v_1} = 2b_{2i} \quad (4.11)$$

Now, equating equation (4.3) and (4.11):

$$2b_2 = 2f_2 + 2f_5 v_1 + 2f_8 v_1^2 \quad (4.12)$$

To find the “ $b$ ” coefficients, the delay variation was fixed at certain value of “ $v$ ” e.g. ( $v=v_1$ ), and several combinations of “ $l$ ” and “ $G$ ” were taken experimentally. These values were applied to the Matlab® curve fitting toolbox to obtain the various  $b$  coefficients for the designated delay value.

The above procedure was repeated with a different value of “ $v$ ” i.e. ( $v=v_2$ ) to work out another set of “ $b$ ” factors.

The obtained combinations of “ $v_1$ ” and “ $b$ ” from the above were substituted in equation 4.12 to find the following two equations relating the factors “ $f$ ”

Substituting the new values of “ $b$ ” in equation 4.12, new combinations between “ $f$ ” and “ $v$ ” are obtained. Mathematically:

$$2b_{21} = 2f_2 + 2f_5v_1 + 2f_8v_1^2 \quad (4.13)$$

$$2b_{22} = 2f_2 + 2f_5v_2 + 2f_8v_2^2 \quad (4.14)$$

Where “ $b_{21}$ ” and “ $b_{22}$ ” are the factor “ $b_2$ ” in equations (4.9 – 4.12) for the first and second values of “ $v$ ” where ( $v=v_1$ ), and ( $v=v_2$ ) respectively.

Similarly, equation 1 is partially differentiated with respect to “ $v$ ” and the previous procedure reapplied but with a fixed loss variable “ $l$ ” this time to obtain another two equations relating the factors “ $f$ ”.

Partially differentiating equation (4.1) with respect to “ $v$ ”:

$$\frac{\partial G}{\partial v} = f_3 + f_4l + f_5l^2 + 2f_6v + 2f_7lv + 2f_8l^2v \quad (4.15)$$

Second partial derivative:

$$\frac{\partial^2 G}{\partial v^2} = 2f_6 + 2f_7l + 2f_8l^2 \quad (4.16)$$



By fixing the loss variable “ $l$ ” while using Polynomial Curve fitting “ $G$ ” would be:

$$G|_{\text{constant } l} = g_0 + g_1 v + g_2 v^2 \quad (4.17)$$

Where “ $g$ ’s” are constants.

Differentiating equation (4.17) with respect to “ $v$ ”:

$$\left. \frac{dG}{dv} \right|_{l=l_1} = g_1 + 2g_2 v \quad (4.18)$$

Second derivative:

$$\left. \frac{d^2 G}{dv^2} \right|_{l=l_1} = 2g_2 \quad (4.19)$$

Equating equation (4.16) and (4.19):

$$2g_2 = 2f_6 + 2f_7 l + 2f_8 l^2 \quad (4.20)$$

The Matlab<sup>®</sup> Curve fitting toolbox was again used to find the “ $g$ ” factors. Using two sets of experimental combinations of “ $G$ ” and “ $v$ ”.

The following two sets of “ $g$ ” factors were obtained at two different values of “ $l$ ”:

$g_{21}$ ,  $g_{11}$ , and  $g_{01}$  at  $l = l_1$

$g_{22}$ ,  $g_{12}$ , and  $g_{02}$  at  $l = l_2$

Substituting in equation (4.20) with the above values of " $l$ 's" and " $g$ 's":

$$2g_{21} = 2f_6 + 2f_7l_1 + 2f_8l_1^2 \quad (4.21)$$

$$2g_{22} = 2f_6 + 2f_7l_2 + 2f_8l_2^2 \quad (4.22)$$

In total, nine equations are needed to find the factors " $f$ ". With four equations ("4.13", "4.14", "4.21", "4.22") of factors " $f$ ", the remaining five equations are then calculated by applying five different sets of " $G$ ", " $l$ ", and " $v$ " to equation 4.1.

$$G(l_1, v_1) = f_0 + f_1l_1 + f_2l_1^2 + f_3v_1 + f_4l_1v_1 + f_5l_1^2v_1 + f_6v_1^2 + f_7l_1v_1^2 + f_8l_1^2v_1^2 \quad (4.23)$$

$$G(l_2, v_2) = f_0 + f_1l_2 + f_2l_2^2 + f_3v_2 + f_4l_2v_2 + f_5l_2^2v_2 + f_6v_2^2 + f_7l_2v_2^2 + f_8l_2^2v_2^2 \quad (4.24)$$

$$G(l_3, v_3) = f_0 + f_1 l_3 + f_2 l_3^2 + f_3 v_3 + f_4 l_3 v_3 + f_5 l_3^2 v_3 + f_6 v_3^2 + f_7 l_3 v_3^2 + f_8 l_3^2 v_3^2 \quad (4.25)$$

$$G(l_4, v_4) = f_0 + f_1 l_4 + f_2 l_4^2 + f_3 v_4 + f_4 l_4 v_4 + f_5 l_4^2 v_4 + f_6 v_4^2 + f_7 l_4 v_4^2 + f_8 l_4^2 v_4^2 \quad (4.26)$$

$$G(l_5, v_5) = f_0 + f_1 l_5 + f_2 l_5^2 + f_3 v_5 + f_4 l_5 v_5 + f_5 l_5^2 v_5 + f_6 v_5^2 + f_7 l_5 v_5^2 + f_8 l_5^2 v_5^2 \quad (4.27)$$

Equations 4.13, 4.14, 4.21, 4.22, 4.23-4.27 are reformatted in matrix form:

Equation (4.13)	.....	0	0	2	0	0	2v <sub>1</sub>	0	0	2v <sub>1</sub> <sup>2</sup>	$\begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \\ f_6 \\ f_7 \\ f_8 \end{bmatrix} = \begin{bmatrix} 2b_{21} \\ 2b_{22} \\ 2g_{21} \\ 2g_{22} \\ G_1 \\ G_2 \\ G_3 \\ G_4 \\ G_5 \end{bmatrix} \quad (4.28)$
Equation (4.14)	.....	0	0	2	0	0	2v <sub>2</sub>	0	0	2v <sub>2</sub> <sup>2</sup>	
Equation (4.21)	.....	0	0	0	0	0	0	2	2l <sub>1</sub>	2l <sub>1</sub> <sup>2</sup>	
Equation (4.22)	.....	0	0	0	0	0	0	2	2l <sub>2</sub>	2l <sub>2</sub> <sup>2</sup>	
Equation (4.23)	.....	0	l <sub>1</sub>	l <sub>1</sub> <sup>2</sup>	v <sub>1</sub>	l <sub>1</sub> v <sub>1</sub>	l <sub>1</sub> <sup>2</sup> v <sub>1</sub>	v <sub>1</sub> <sup>2</sup>	l <sub>1</sub> v <sub>1</sub> <sup>2</sup>	l <sub>1</sub> <sup>2</sup> v <sub>1</sub> <sup>2</sup>	
Equation (4.24)	.....	0	l <sub>2</sub>	l <sub>2</sub> <sup>2</sup>	v <sub>2</sub>	l <sub>2</sub> v <sub>2</sub>	l <sub>2</sub> <sup>2</sup> v <sub>2</sub>	v <sub>2</sub> <sup>2</sup>	l <sub>2</sub> v <sub>2</sub> <sup>2</sup>	l <sub>2</sub> <sup>2</sup> v <sub>2</sub> <sup>2</sup>	
Equation (4.25)	.....	0	l <sub>3</sub>	l <sub>3</sub> <sup>2</sup>	v <sub>3</sub>	l <sub>3</sub> v <sub>3</sub>	l <sub>3</sub> <sup>2</sup> v <sub>3</sub>	v <sub>3</sub> <sup>2</sup>	l <sub>3</sub> v <sub>3</sub> <sup>2</sup>	l <sub>3</sub> <sup>2</sup> v <sub>3</sub> <sup>2</sup>	
Equation (4.26)	.....	0	l <sub>4</sub>	l <sub>4</sub> <sup>2</sup>	v <sub>4</sub>	l <sub>4</sub> v <sub>4</sub>	l <sub>4</sub> <sup>2</sup> v <sub>4</sub>	v <sub>4</sub> <sup>2</sup>	l <sub>4</sub> v <sub>4</sub> <sup>2</sup>	l <sub>4</sub> <sup>2</sup> v <sub>4</sub> <sup>2</sup>	
Equation (4.27)	.....	0	l <sub>5</sub>	l <sub>5</sub> <sup>2</sup>	v <sub>5</sub>	l <sub>5</sub> v <sub>5</sub>	l <sub>5</sub> <sup>2</sup> v <sub>5</sub>	v <sub>5</sub> <sup>2</sup>	l <sub>5</sub> v <sub>5</sub> <sup>2</sup>	l <sub>5</sub> <sup>2</sup> v <sub>5</sub> <sup>2</sup>	

$$\text{Or: } \varphi F = A \quad (4.29)$$

Thus, factors “ $f$ ” can be found from:

$$F = \varphi^{-1} A \quad (4.30)$$

Accordingly, to find out the quality grade of the application, the measured loss and jitter values should be substituted in equation (4.1) along with the evaluated factors from equation (4.30)

#### 4.5 Conclusion

In order to determine the specific hop or hops which play a role in reducing the quality of a certain application over the network, this quality has to be graded and measured according to the application in hand and the network parameters which are affecting the hop during the session time. Applications differ in their behavior and therefore act differently when it comes to their networking demands and their manipulation by the varying network parameters. Accordingly, the grading measure of one application is independent and not applicable to another application. Since this research deals mainly with Voice over IP, a grading measure had to be established for this application specifically. This grading measure will be used later in the thesis to convey the quality experienced by VoIP applications at the examined hops.

In this chapter, an algorithm for the grading of an application according to the existing network performance was proposed and explained. Furthermore, the mathematical model by which the quality would be graded was discussed and analysed.

Once the values of loss and delay in each hop along the network is determined, the grading algorithm would then be utilized in the light of those values to determine the quality of service at each hop independently for the required application according to the grading algorithm discussed in this chapter. This hop by hop-basis quality determination using the grading algorithm discussed in this chapter is addressed in the following chapters along with the results outcome.

## *Chapter 8*

### **Conclusions and Recommendations for Further Work**

**M**asuring load generated deterioration in an established, reliable network means measuring the Quality of Service at the hop devices, since the medium in between is constant and has nothing to do with network congestion. This work investigated and implemented several approaches to determine this information. In this work we aimed to detect any router or routers which present a bottle neck in a network for a multimedia application connection through a grading system for the designated application.

This research is concerned with identifying the degradation of quality in the network, on a hop by hop basis to assist network managers in recognising the weak links of their network which need to be tackled in order to enhance the quality of the application used over their domain.

While applications differ in their requirements for Quality of Service, the approach conducted in this work was also tested for the requirements of Virtual-reality game sessions over the net, and for Web-browsing. The *emphasis though, was on Voice over IP applications, which were tested for all stages on a test network.*

Determining at which hop or router that a real-time multimedia connection suffers deterioration of its potential quality, is valuable information for

network managers to enhance their network domain reliability. In the case of VoIP sessions such data would assist -for example- the VSP (VoIP Service Provider) to change its network provider if the congestion occurs at a router under the support of a specific network manager, or the VSP could approach the network administrator to discuss and tackle this problem.

Unlike TCP based applications, VoIP application quality is vulnerable to some types of traffic congestion. One of the main obvious reasons for this is because; firstly, UDP does not cater for the applications it is supporting regarding any traffic problems such as lost packets or congestion; whereas TCP is designed to retransmit lost packets. Secondly, the VoIP application itself does not tolerate heavy traffic. Hence, it can be of importance for network managers as well as the VSP to determine if congestion occurs at a certain area of the network on a regular basis, and whether this congestion can undermine VoIP quality or the overall connection.

The study was conducted on a specifically built test network. The use of the LNE (Loaded Network Emulator) was introduced to provide a real network congestion environment, which is controllable and can be altered on demand to allow for testing different network congestion scenarios.

In this work several techniques were evaluated, including: 1- Investigating the queue-state in the MIBs (Management Information Base) of the Cisco routers to extract the required parameters to assess the quality, 2- Active Networks; through the use of Time-stamping, and Active-ping. 3- Designing a prototype tool using the simplest techniques available in any network, so as to be applicable and usable on any network.

It was shown that statistical approaches proved to be very useful in determining the hop by hop performance for certain real-time applications. Furthermore, since they are statistical in nature, they do not require capture of every packet present in the traffic stream.

A mathematical model was utilised and explained to quantify the quality of service of real-time multimedia applications in general, and Voice over IP in specific. This mathematical algorithm was used in the later stages to grade the QoS at the designated hops independently.

The goal of any performance study is either to find the optimal parameter value or to compare different alternatives [Jain 1991], which was manifested in this research. Three different alternative approaches were examined to determine the quality of service at the intermediate hops in a real-time multimedia application's network session:

- I. Applying active network capabilities to obtain the parameters which affect the quality of real-time multimedia application at every hop. Two techniques were inspected; Active ping and time-stamping. Active ping was presented as a useful technique to implement in an active network environment.
- II. Utilising the Management Information Base of the System Network Management Protocol as an approach to be considered to acquire the required data for every hop to be analysed further.
- III. Using a prototype tool as a complete package to extract the needed information, analyse it through applying the real-time multimedia application mathematical model algorithm to it, thus determining the various qualities at the different hops and representing the final result to the end-user.



The work of this research could be extended further by establishing an actual VoIP call or a VR-game over a real private WAN. A hop with an LNE could be added some where on the network path or next to either end-user to add a desired delay or packet loss, in order to measure the accuracy of tool over the WAN.

Moreover, exploiting the MIB as a means for determining the weak points of a multimedia connection path could be examined in depth in the light of newer software upgrades and the advent of newer IP versions (IP v6).

## References

Bellamy, John. 1991.

*Digital Telephony*. John Wiley & Sons Inc, 2<sup>nd</sup> edition.

Bharadia, Ketan. 2001

*Network Application Detection Techniques*. Loughborough University, PhD.

Bierman, A. McCloghrie, K. Aug 2005.

*Entity MIB (Version 3)*. The Internet Society, Network Working Group, RFC: 4113, Obsoletes: 2737, <ftp://ftp.rfc-editor.org/in-notes/rfc4133.txt>

Black, Uyless. 2000.

*Voice Over IP*. Prentice Hall.

Boger, Yuval.

*Fine-tuning Voice over Packet services*. VP Business Development. Radcom Ltd. <http://www.protocols.com/pbook/pdf/voip.pdf>

Boisseau, M. Demange, M. Munier, J.-M. 1994.

*High-Speed Networks*. John Wiley & Sons Ltd.

Braden, R. Clark, D. Shenker, S. 1997

*Integrated Services in The Internet Architecture: an Overview Request for Comments*, The Internet Society, Network Working Group, RFC: 1633.

<http://citeseer.ist.psu.edu/cache/papers/cs/4035/ftp:zSzzSzftp.math.utah.eduSzpubzSzrfczSzrfc1633.pdf/braden94integrated.pdf>

Brebnet, Gordon. 1997.

*Computers in Communication*. McGraw-Hill.

Buchanan, Bill. 1999.

*Handbook of Data Communication and Networks*. Kluwer Academic Publishers.

Buchanan, Jr Robert, W. 1996.

*The Art of Testing Network Systems*. John Wiley & Sons Ltd.

Call Center News.

*VoIP equipment market holds steady*. United publication Inc.

<http://www.ccnews.com/february2002/depts/cct/cctstory5.htm>

Comer, Douglas, E. 2004.

*Computer Networks And Internets*. Pearson Education Ltd, Fourth Edition.

Coombs, Clyde, F. Coombs, Catherine, Ann. 1998.

*Communications Network Test & Measurement Handbook*. McGraw-Hill.

Coulier, Kurt. Dombrecht, Roel. Evens, Geert. Seys, Stefaan. Theunis, Johan. 1999.

*LAN-PABX*. Technical report, KUL, ESAT - TELEMIC.

Daumer, W., R. 1982.

*Subjective evaluation of several efficient speech coders*. IEEE Transactions on Communications. COM-30, Vol: 4, pp: 655-622.

Dongyun, Zhou., Subramaniam, S. 2000.

*Survivability in Optical Networks*. Network, IEEE, Nov/Dec 2000, Vol: 14 issue: 6, pp:16-23.

Fenner, B. Flick, J. June-3-2005.

*Management Information Base for the User Datagram Protocol (UDP)*. The Internet Society, Network Working Group, RFC: 4113, Obsoletes: 2454, 2013, <ftp://ftp.rfc-editor.org/in-notes/rfc4113.txt>

Ferguson, Paul. November 1997

*Simple Differential Services: IP TOS and Precedence, Delay Indication, and Drop Preference.* Cisco Systems.

<http://www3.ietf.org/proceedings/97dec/slides/intserv-ferguson/index.htm>

Forouzan, Behrouz. Fegan, Sophia, Chung. 2003.

*TCP/IP Protocol Suit.* McGraw-Hill Companies, 2<sup>nd</sup> Edition.

Garson, David, G. 2002.

*Guide to Writing Empirical Papers, Thesis, and Dissertations.* Marcel Dekker, Inc. Petrou, A. D. Library Quarterly Vol: 73; Part 2, pp:215-216.

Govindan, Ramesh. Tangmunarunkit, Hongksuda. 2000.

*Heuristics for Internet Map Discovery.* IEEE Infocom. Vol:3, pp: 1371-1380.

Halsall, Fred. 1998.

*Data Communications, Computer Networks and Open Systems.* Addison-Wesley, Fourth Edition.

Hardy, William, C. 2003.

*VoIP Service Quality; Measuring and Evaluating Packet-Switched Voice.* McGraw-Hill Companies.

Hersent, Oliver. Gurle, David. Petit, Jean-Pierre. 2000.

*IP Telephony; Packet-Based Multimedia Communications Systems.* Pearson Education Limited, Addison Wesley.

Hersent, Oliver. Petit, Jean-Pierre. Gurle, David. 2005.

*Beyond Voip Protocols; Understanding Voice Technology And Networking Techniques For IP Telephony.* John Wiley & Sons Inc.

Hersent, Oliver. Petit, Jean-Pierre. Gurle, David. 2005.

*IP Telephony; Deploying Voice-Over-IP Protocols.* John Wiley & Sons Inc.

Hochmuth, Phil, 20-9-2005

*Users Discuss Big VoIP Rollout Risks and Rewards.* Network World.com

Huitema, Christian. 1995.

*Routing In The Internet.* Prentice Hall.

Jain, Raj. 1991.

*The Art of Computer Systems Performance Analysis; Techniques for Experimental Design, Measurement, Simulation, and Modelling.* John Wiley and Sons, Inc.

Karapetkov, Stefan. 1999.

*Deploying H.323 for voice, video and data communications in IP networks.*

Siemens, White Paper.

[http://www.icn.siemens.com/icn/products/ip\\_data\\_net/data/h323\\_wht\\_paper.pdf](http://www.icn.siemens.com/icn/products/ip_data_net/data/h323_wht_paper.pdf)

Kurose, James, F. Ross, Keith, W. 2005.

*Computer Networking; A Top-Down Approach Featuring the Internet.* Pearson Education Inc.

MacFaden, M. Partain, D. Saperia, J. Tackabury W. April 2003

*Configuring Networks and Devices with Simple Network Management Protocol (SNMP).* The Internet Society, Network Working Group, RFC: 3512, <ftp://ftp.rfc-editor.org/in-notes/rfc3512.txt>

Mier-1, Edwin, E. January 1999.

*Voice-Over-IP: Better and Better.* Business Communications Review. USA Jan. 1999, Vol: 29, no.1, pp: 28-34.

Mier-2, Edwin, E. January 1999.

*VoIP Gateways-Tradeoffs affecting Voice Quality.* Business Communications Review Voice 2000.

Minoli, Daniel. Minoli, Emma. 1998.

*Delivering Voice over IP Networks*. John Wiley & Sons Inc.

Oliver, M., A. Bharadia, K., R. Phillips, I., W. Parish, D., J. April 2000.

*Grading and Predicting Networked Application Behaviour*. Computing & Control Engineering Journal. Vol: 11, no. 2, pp: 65-72.

Oliver, M., A. Phillips, I., W. Parish, D., J. Bharadia, K., R. July 1999.

*A Scheme for Predicting User-Perceived Networked Application Performance*. BT Technology Journal, Vol:17, No 3. pp: 135-145.

Paxson, Vern. October 1997.

*End to End Routing Behaviour in the Internet*. IEEE/ACM Transactions on Networking USA Oct. 1997, Vol:5, no.5, pp: 601-15.

Pohlmann, Ken C. 2005.

*Principles of Digital Audio*. Fifth edition, McGraw-Hill Companies.

Psounis, K. 1999.

*Active Networks, Applications, Security, Safety and Architectures*. IEEE Communications Surveys, Vol: 2, No. 1, First Quarter.

Räisänen, Vilho. 2003.

*Implementing Service Quality in IP Networks*. John Wiley & Sons Ltd.

Ramamurthy, S., Mukherjee, B. 1999.

*Survivable WDM mesh networks. Part I-Protection*. Infocom '99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. 21-25 March 1999. Vol: 2, PP: 744-751.

Rix, Antony. W. 2000.

*Perceptual analysis measurement system for robust end-to-end speech quality assessment*. ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings 2000 Vol: 3 pp: 1515-1518.

Tennenhouse, D. Garland, S., J. Shrira, L. M., F., Kaashoek. 1996.

*From Internet to ActiveNet*. Request for Comments, Laboratory of Computer Science, MIT. <http://tns-www.lcs.mit.edu/publications/rfc96/rfc96.html>

Tennenhouse, D. Wetherall, D. 1996.

*Towards an Active Network Architecture*. In *Multimedia Computing and Networking* (MMCN 96). San Jose, CA.

[http://ieeexplore.ieee.org/xpl/freeabs\\_all.jsp?arnumber=1003480](http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=1003480)

Tennenhouse, David. Smith, Jonathan. Sincoskie, David. Wetherall, David. Minden, Gary. January 1997.

*A Survey of Active Network Research*. IEEE Communications Magazine, Vol: 35, No. 1, PP: 80-86.

Waldbusser, S. May 2000

*Remote Network Monitoring Management Information Base*. The Internet Society, Network Working Group, RFC: 2819, STD: 59, Obsoletes: 1757.  
<ftp://ftp.rfc-editor.org/in-notes/std/std59.txt>

Westall, F., A. Johnston, R., D. Lewis, A., V. January 1996.

*Speech Technology for Telecommunications*. BT Technology Journal, Vol: 14, No. 1, pp: 9-27.

Robins, Marc. 2005

*Testing is Fundamental*, TMCnet.com <http://www.tmcnet.com/sectors/voip/>

Siamwalla, R. Sharma, R. Keshav, S. 1999.

*Discovering Internet Topology*. IEEE Infocom.

Siddiqui, Mahboob-ul-Haq. 1989.

*Performance Measurement Methodology for Integrated Services Networks*.

Doctoral Thesis, Loughborough University.

Stalling, William. 1998.

*High Speed Networks: TCP / IP and ATM design principles*. Prentice-Hall, Inc.

Stallings, William. 2001.

*Business Data Communications*. Prentice Hall, Fourth Edition.

Stevens, Richard. 1998.

*Unix Network Programming, Volume 1, Networking APIs: Sockets and XTI*.

Prentice Hall Inc, Second Edition.

Stevens, Richard. April 2000.

*TCP/IP Illustrated Volume 1 The Protocols*. Addison-Wesley Longman Inc,

17<sup>th</sup> printing.

Swale, Richard. 2001.

*Voice Over IP; Systems And Solutions*. The Institute of Electrical Engineers, BT

Exact Technologies, London.

Tanenbaum, Andrew. 2003.

*Computer Networks*. Prentice Hall Inc, Fourth edition.



## Appendix 1

### (a)

#### File pfa.H

```
/*
    Header file to extract the ping results.

    pfa.h
*/

#ifndef _PFA
#define _PFA
#define _PFA_DEBUG 0

#include <string.h>
#include <stdlib.h>
#include <stdio.h>

typedef struct
{
    int PacketsTransmitted;
    int PacketsReceived;
    float LossPercentage;
    float MinDelay;
    float MaxDelay;
    float AverageDelay;
    float MeanDelayDeviation;
} PingResults;

/*    Entry point. Function to read the text file containing ping
    measurements and return the results in the pointer to the
    PingResults structure. Returns zero if unsuccessful.
*/
int GetPingResults(char* DataFile, PingResults* Summary);

/*    Function to read the last 2 lines from the text file. Returns
    zero if unsuccessful
*/
int GetLast2Lines(char* DataFile, char* SecondLastLine, char*
LastLine);

/*    Function to extract packet counts from the second last line.
    Returns 0 if unsuccessful */
int GetPacketCounts(char* SecondLastLine, PingResults* Summary);

/*    Function to extract delay statistics from the last line.
    Returns 0 if unsuccessful */
int GetDelayStatistics(char* LastLine, PingResults* Summary);

#endif
```

## Appendix 1 (b)

### File pfa.C

```
/*
    Implementation of functions in pfa.h
*/

#include "pfa.h"

/*
    Entry point. Function to read the text file containing ping
    measurements and return the results in the pointer to the
    PingResults structure. Returns zero if unsuccessful.
*/
int GetPingResults(char* DataFile, PingResults* Summary)
{
    int Success;
    char SecondLastLine[100];
    char LastLine[100];

    SecondLastLine[0] = '\0';
    LastLine[0] = '\0';

    Success = GetLast2Lines(DataFile, SecondLastLine, LastLine);
    if (Success)
    {
#ifdef _PFA_DEBUG
        printf("Last 2 lines in %s\n%s\n%s\n", DataFile, SecondLastLine,
            LastLine);
#endif
        Success = GetPacketCounts(SecondLastLine, Summary);
        if (Success)
        {
            Success = GetDelayStatistics(LastLine, Summary);
        }
    }

    return(Success);
}

/*
    Function to extract packet counts from the second last line.
    Returns 0 if unsuccessful */
int GetPacketCounts(char* SecondLastLine, PingResults* Summary)
{
    char FirstSegment[50];
    char SecondSegment[50];
    char ThirdSegment[50];
    int Success = 1;
    char* TempPtr;

    TempPtr = strtok(SecondLastLine, ",");
    if (TempPtr == NULL)
    {
#ifdef _PFA_DEBUG
        printf("Error tokenising first token in counts\n");
#endif
    }
}
```

```

#endif
        return (0);
    }
    else
    {
        strcpy(FirstSegment,TempPtr);

        TempPtr = strtok(NULL,"");
        if (TempPtr == NULL)
        {
            #if _PFA_DEBUG
                printf("Error tokenising second token in counts\n");
            #endif
            return (0);
        }
        else
        {
            strcpy(SecondSegment,TempPtr);

            TempPtr = strtok(NULL,"");
            if (TempPtr == NULL)
            {
                #if _PFA_DEBUG
                    printf("Error tokenising third token in counts\n");
                #endif
                return (0);
            }
            else
            {
                strcpy(ThirdSegment,TempPtr);

                TempPtr = strtok(FirstSegment," ");
                if (TempPtr == NULL)
                {
                    #if _PFA_DEBUG
                        printf("Error tokenising transmitted packets\n");
                    #endif
                    return (0);
                }
                else
                {
                    Summary->PacketsTransmitted = atoi(TempPtr);

                    TempPtr = strtok(SecondSegment," ");
                    if (TempPtr == NULL)
                    {
                        #if _PFA_DEBUG
                            printf("Error tokenising received packets\n");
                        #endif
                        return (0);
                    }
                    else
                    {
                        Summary->PacketsReceived = atoi(TempPtr);

                        TempPtr = strtok(ThirdSegment,"%");

```

```

        if (TempPtr == NULL)
        {
#if _PFA_DEBUG
            printf("Error tokenising packet loss percentage\n");
#endif
            return (0);
        }
        else
        {
            Summary->LossPercentage = (float) atof(TempPtr);
        }
        return(Success);
    }

/*    Function to extract delay statistics from the last line.
    Returns 0 if unsuccessful */
int GetDelayStatistics(char* LastLine, PingResults* Summary)
{
    char Values[50];
    char* TempPtr;
    int Success = 1;

    TempPtr = strtok(LastLine, "=");
    if (TempPtr == NULL)
    {
#if _PFA_DEBUG
        printf("Error tokenising delay stats headers\n");
#endif
        return (0);
    }
    TempPtr = strtok(NULL, " ");
    if (TempPtr == NULL)
    {
#if _PFA_DEBUG
        printf("Error tokenising delay values\n");
#endif
        return (0);
    }
    else
    {
        strcpy(Values, TempPtr);
    }

    TempPtr = strtok(Values, "/");
    if (TempPtr == NULL)
    {
#if _PFA_DEBUG
        printf("Error tokenising min delay value\n");
#endif
        return (0);
    }
    else
    {
        Summary->MinDelay = (float) atof(TempPtr);
    }

    TempPtr = strtok(NULL, "/");
    if (TempPtr == NULL)
    {
#if _PFA_DEBUG
        printf("Error tokenising avg delay value\n");
#endif

```

```

#endif
        return (0);
    }
    else
    {
        Summary->AverageDelay = (float) atof(TempPtr);;

        TempPtr = strtok(NULL, "/");
        if (TempPtr == NULL)
        {
#ifdef _PFA_DEBUG
            printf("Error tokenising max delay value\n");
#endif
            return (0);
        }
        else
        {
            Summary->MaxDelay = (float) atof(TempPtr);;

            TempPtr = strtok(NULL, "");
            if (TempPtr == NULL)
            {
#ifdef _PFA_DEBUG
                printf("Error tokenising mdev delay value\n");
#endif
            }
            return (0);
        }
        else
        {
            Summary->MeanDelayDeviation = (float) atof(TempPtr);;
        }
        return (Success);
    }
}

/*    Function to read the last 2 lines from the text file. Returns
    zero if unsuccessful
*/
int GetLast2Lines(char* DataFile, char* SecondLastLine, char* LastLine)
{
    int Success;
    int LineCount = 0;
    char TempLine[100];
    FILE* FilePtr;

    FilePtr = fopen(DataFile, "r");
    if (FilePtr == NULL)
    {
        Success = 0;
#ifdef _PFA_DEBUG
        printf("Cannot open %s\n", DataFile);
#endif
    }
    else
    {
        while(!feof(FilePtr))
        {
            fgets(TempLine, sizeof(TempLine), FilePtr);
            if (!feof(FilePtr))
            {

```

```

        LineCount++;
        strcpy(SecondLastLine, LastLine);
        strcpy(LastLine, TempLine);
    }
}
fclose(FilePtr);
if (LineCount < 5)
{
    Success = 0;
#ifdef _PFA_DEBUG
    printf("Line Count = %d\n", LineCount);
#endif
}
else
{
    Success = 1;
}
}

return(Success);
}

```

## Appendix 1 (c)

### G\_util.C

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <pthread.h>
#include <string.h>
#include <time.h>
#include "pfa.h"

#define MAX_HOPS          32
#define PING_INTERVAL    0.1
#define PING_COUNT       500

void* PingIP(void* InIPAddress)
{
    char IPAddress[25];
    time_t TimeValue;
    char PingCommandString[200];
    strcpy(IPAddress, (char*)InIPAddress);
    sprintf(PingCommandString, "ping -i %3.1f -c %d %s >
%s.txt", PING_INTERVAL, PING_COUNT, IPAddress,
        IPAddress);
    time(&TimeValue);
    printf("%s started at
%s\n", PingCommandString, ctime(&TimeValue));
    system(PingCommandString);
    time(&TimeValue);
    printf("%s ended at %s\n", PingCommandString, ctime(&TimeValue));
    pthread_exit(NULL);
    return(NULL);
}

int main(void)
{
    PingResults Summary;
    FILE* tracert;
    float maximum;
    float minimum;
    float std_dev;
    float average;
    float ftotal=0;
    int i, j;
    float array[500000];
    char currentline[600];
    char currentword[25][MAX_HOPS];
    char* word;
    char temp[200];
    char TracerouteCommandString[200];
    char IPAddress[25];
    char Dist_IP[25];
```

```

float median;
int argv;
int ip;
pthread_t ThreadIDs[MAX_HOPS];
int HopCount = 0;

// ***** Input Destination IP-Address *****

printf("Please enter Distention IP_Address: ");
scanf("%s",Dist_IP);
printf("\nDistention IP-Add is: %s\n",Dist_IP);

// ***** executing "Trcaroute" and writting hops add's into file
"trace.txt" *****

sprintf(TracerouteCommandString,"traceroute %s >
trace.txt",Dist_IP);
puts(TracerouteCommandString);
system(TracerouteCommandString);

// ***** segmenting the IP add's into file "hops"
*****

tracert = fopen("trace.txt","r");
if(tracert == NULL)
{
    printf("File cannot be opened\n");
}
else
{
    // fgets(currentline,600,tracert);
    while(!feof(tracert))
    {
        fgets(currentline,600,tracert);
        if (!feof(tracert))
        {
            // if(currentline[0]>='1' && currentline[1]==' ' &&
            currentline[2]==' ')
            // {
                word = (char*) strtok(currentline," ");
                word = (char*) strtok(NULL," ");
                strcpy(currentword[HopCount],word);
                puts(currentword[HopCount]);
                puts(currentword[HopCount]);

                pthread_create(&ThreadIDs[HopCount],NULL,PingIP,currentword[Hop
Count]);
                HopCount++;
            // }
        }
    }
    fclose(tracert);
    for (i=0;i<HopCount;i++)
    {
        pthread_join(ThreadIDs[i],NULL);
    }

    for (i=0;i<HopCount;i++)

```



```

    {
        strcpy(currentline, currentword[i]);
        strcat(currentline, ".txt");
        if (GetPingResults(currentline, &Summary))
        {
            printf("Results from %s\n", currentword[i]);
            printf("Loss          =
%d\n", Summary.PacketsTransmitted-Summary.PacketsReceived);
            printf("Mean Dev   =
%f\n", Summary.MeanDelayDeviation);
        }
    }

    return(0);
}

```

## Appendix 2

# DETERMINATION OF VOIP (VOICE OVER IP) SERVICE QUALITY ON A HOP BY HOP BASIS

**Momen Al-Rawi, David J. Parish**

Department of Electronic and Electrical Engineering, Loughborough University

*M.Al-rawi@lboro.ac.uk, D.J.Parish@lboro.ac.uk*

IADAT 27-29 Sept 06 International Conference on Telecoms &  
Computer Networks held in Portsmouth.

### Abstract

*Voice over IP (VoIP) can provide excellent voice reproduction quality providing the networks over which the packets travel provide an appropriate service. As a VoIP connection can traverse a number of different networks, it can be difficult to identify where a connection is degraded if the quality is not acceptable. This paper proposes a solution to this problem whereby the voice quality at each hop in the network is estimated, and degraded hops are identified. This is achieved by use of a model which predicts voice quality based on low level network measurements including packet loss and delay. Experimental results are presented to show how the approach can identify problem hops in a VoIP path.*

**Keywords** VoIP, voice quality

### 1 Introduction

VoIP [1], has emerged as a valuable alternative to traditional telephony systems. VoIP however relies upon the delivery of its IP packets to the destination with a sufficient level of service that the regenerated speech is of sufficient quality to meet the needs of the users. The network paths over which a VoIP "connection" is carried are not usually under the control of the end users and, if not sufficiently provisioned, could cause the voice quality to be degraded. If the path passes through different operator's networks, it can be difficult to identify where, on a multi-hop path, the degradation is occurring.

This paper proposes a solution to this problem. Initially, the expected quality of a VoIP connection is quantified in terms of the basic path characteristics which influence it [2]. In practice, our investigations have shown that VoIP audio quality over typical Internet paths is largely dominated by packet loss, although variation in delay has a noticeable effect if this becomes significant. Absolute delay will affect the usability of the connection, but not usually the voice quality itself. A model is then constructed which generates a predicted grade for VoIP audio quality when subjected to a range packet transfer characteristics. The packet transfer characteristics (loss and possibly variation in delay) are collected for each path in the connection. In our experiments, these characteristics have been obtained by firstly identifying the main routers through which the VoIP packets pass by using a utility such as *Traceroute* and then measuring the characteristics of each section of the path from the initiating user to the receiving user using a utility such as *Ping*. This allows an approximate indication of the packet-level performance of each hop in the path to be determined. Whilst only an approximation, the approach will identify any hops with significantly degraded packet

performance. Finally, the predicted voice quality resulting from each hop of the path is calculated as a simple grade. The degradation (if any) resulting from each hop of the path can then be seen.

## 2 Application Performance Prediction

Previous work [2] has shown that it is possible to model the approximate user perceived performance of a networked application in terms of the basic packet level characteristics of delay; variation in delay; and loss. An empirical relationship between a numerical grade representing the user perceived "quality" of the application and these characteristics can be developed. This is achieved by running the application over a networked connection with known delay and loss characteristics and assessing its quality by a panel of users. The networked connection characteristics are changed repeatedly and new quality grades assigned. This data can then be used to derive an empirical grading equation and its parameters. The equation order has been found to relate to the general type of application under analysis. For example, a web browsing application can be modelled by a first order equation, whilst a multimedia application such as a video conferencing tool will require a second or third order equation. This grading activity needs to be performed once only as an offline operation. The grading equation will then provide an estimate of the expected user perceived performance quality of the application for different underlying network conditions.

For the VoIP experiments presented in this paper, the audio codec from the Netmeeting conferencing tool was used. It was found that this is most sensitive to packet loss and an approximate grading equation was developed. The general form of this equation is:

$$G_i = f_0 + f_1(l\_diff_i) + f_2(l\_diff_i)^2 + f_3(l\_diff_i)^3 \quad (1)$$

Where  $f$  = constant value relating to the application.

$(l\_diff)$  = packet loss percentage for each hop

The constant values  $f$  were obtained from a set of grading experiments using different packet loss values. The resulting grading equation is:

$$G_i = 6.6 + (-0.62 * (l\_diff_i)) + (0.03 * (l\_diff_i)^2) + (-0.0005 * (l\_diff_i)^3) \quad (2)$$

The above expression will provide a predicted grade for the voice quality achieved by packets subjected to the loss percentage  $l\_diff$ . This grade is rounded to fall in the range 1-5 where 1 equals unusable quality and 5 represents the best subjective quality achievable.

## 3 Measuring hop-by-hop packet loss

To use the above analysis on a live VoIP connection, it is necessary to obtain packet loss estimates for each hop in the connection path. These values were approximated by using the *Ping* utility to send a stream of packets to each node identified in the path by the *Traceroute* utility. This was evaluated using a 5 hop testbed. As *Ping* sends out probe packet and records the reply, the two way (ie outward and return) path characteristics are measured from the initiating end station to each selected node. The single way loss percentage can then be estimated by assuming the loss in each direction is the same and using equation 3.

$$l = 100 * (1 - \sqrt{\frac{\text{received replies}}{\text{transmitted pings}}}) \tag{3}$$

The approximate loss attributable to each hop is then estimated by subtracting the loss for node (n) from that for node (n-1).

### 4 Results

An application was developed using the approach and evaluated on the 5 node testbed shown in Figure 1. Each node was implemented as a PC based router running the Linux OS and contained code which could selectively drop packets giving rise to controlled network path degradation. This code (the LNE – Loaded Network Emulator) allowed a range of network conditions to be emulated, and also allowed specific hops to be established which would adversely affect the quality of the received audio. A series of tests were conducted in which different packet loss profiles were assigned to different hops of the network.

In Test 1, a high packet loss percentage was configured at the third hop and the application used to predict the audio quality grade resulting from the contribution of each hop in the complete path. The configuration and results are presented in Table 1.

Test		Accumulated Packet loss	Loss at Hop	Expected Q Grade	Actual Grade	Rounded Grade	Grade with 1.9< _diff <27
Hop 1	Rainbow	1	1	5	5.9665	6	5
Hop 2	Yellow	4	3	5	4.9455	5	5
Hop 3	Orange	27	23	1	1.2475	1	1
Hop 4	Pink	31	4	5	4.5079	5	5
Hop 5	Purple	35	4	5	4.5079	5	5

Table 1

Hops 1,2,4,5 were configured to have low packet loss characteristics. Individually, these hops would not cause noticeable degradation to the received audio, although their collective effect may be noticeable. Hop three was configured to loose 23% of packets, which would seriously degrade the speech quality. The analysis tool correctly identifies this hop as being responsible for the most serious disruption to the signal.

Test 2 (Table2) has two hops with poorer packet loss performance. Packet loss at hop 2 is 7% (leading to a predicted voice quality grade of 3) and packet loss at hop 4 is 13% (leading to a predicted voice grade of 2). The overall performance of the connection was very low, but the approach has enabled the two most significant degradation points in the connection path to be identified.

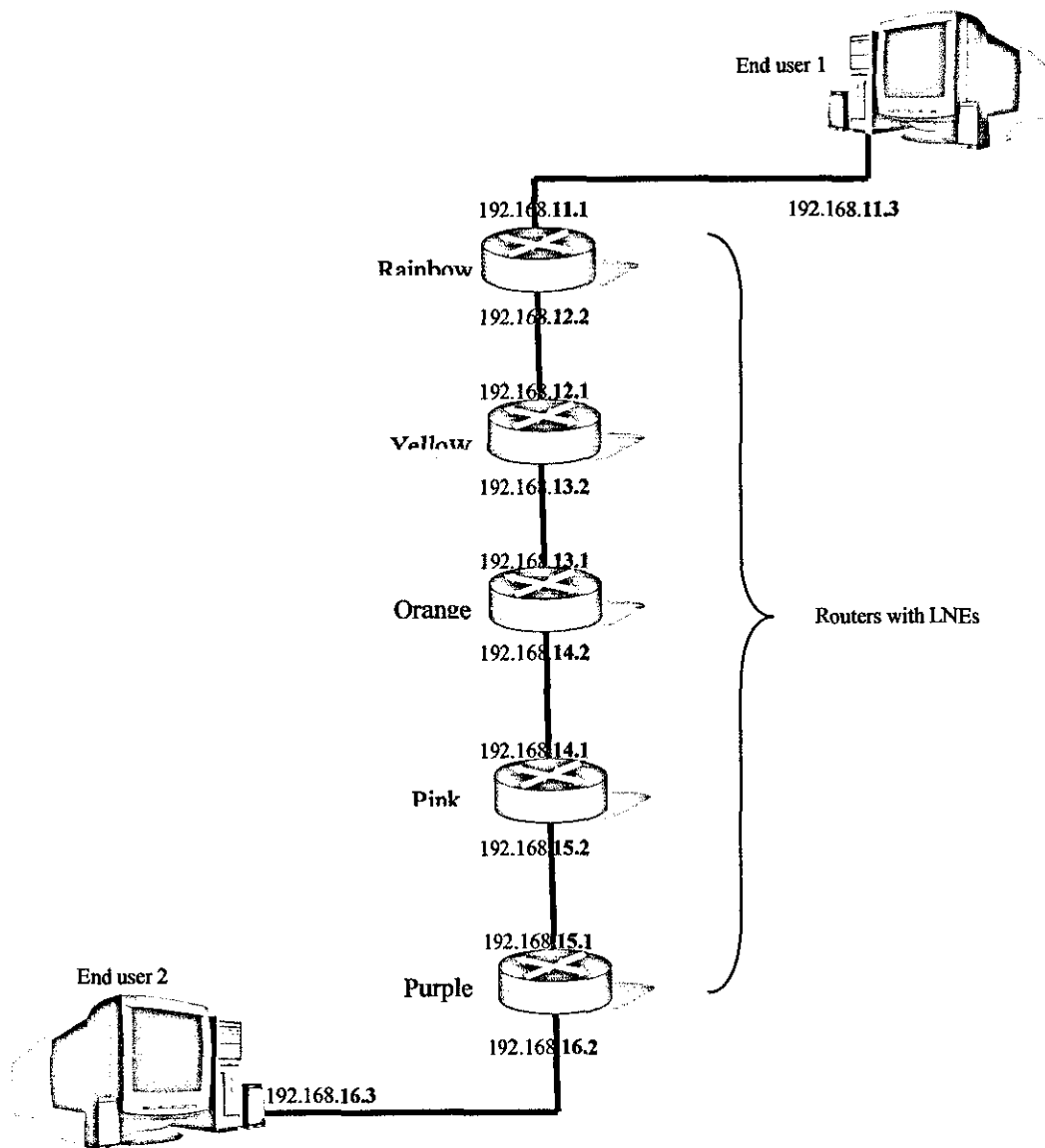


Figure 1 5 Node testbed

Test 2		Accumulated Packet loss	Loss at Hop	Aimed Q Grade	Actual Grade	Rounded Grade	Grade with $1.9 < l\_diff < 27$
Hop 1	Rainbow	3	3	5	4.9455	5	5
Hop 2	Yellow	10	7	3	3.4507	3	3
Hop 3	Orange	13	3	5	4.9455	5	5
Hop 4	Pink	26	13	2	2.2165	2	2
Hop 5	Purple	29	3	5	4.9455	5	5

Table 2

## 5 Conclusions

This paper has presented an approach to the identification of the hops in a network path which cause degradation to a VoIP connection. The approach uses an off-line characterisation procedure followed by real-time prediction of the audio performance on a hop-by-hop basis. The approach could also be used for other networked applications including video conferencing. Whilst the approach is approximate, it is able to identify significantly poorly performing sections of the networked path.

## 6 References

1. Hersent, O., Petit, JP., Gurle, D., IP Telephony; Deploying Voice-Over-IP Protocols, John Wiley & Sons; 2005.
2. Oliver, M., A. Phillips, I., W. Parish, D., J. Bharadia, K., R. A Scheme for Predicting User-Perceived Networked Application Performance. BT Technology Journal, July 1999, Vol 17, No 3.



