Policy Based Network Management of Legacy Network Elements in Next Generation Networks for Voice Services

By

Vaughn Naidoo

A thesis submitted in partial fulfilment of the requirements for the degree of

Magister Scientiae

In the Department of Computer Science

University of the Western Cape

Friday, July 19, 2002

Supervisor: Mr. William Tucker

KEYWORDS

Common Open Policy Service, Internet Protocol, Legacy, Network Element, Network Management, Next Generation Network, Policy Based Network Management, Quality of Service, Simple Network Management Protocol, Voice over Internet Protocol

ABSTRACT

Policy Based Network Management of Legacy Network Elements in Next Generation Networks for Voice Services

by Vaughn Naidoo
MSc Thesis
Faculty of Computer Science
University of the Western Cape

Telecommunication companies, service providers and large companies are now adapting converged multi-service Next Generation Networks (NGNs). Network management is shifting from managing Network Elements (NE) to managing services. This paradigm shift coincides with the rapid development of Quality of Service (QoS) protocols for IP networks. NEs and services are managed with Policy Based Network Management (PBNM) which is most concerned with managing services that require QoS using the Common Open Policy Service (COPS) Protocol. These services include Voice over IP (VoIP), video conferencing and video streaming. It follows that legacy NEs without support for QoS need to be replaced and/or excluded from the network. However, since most of these services run over IP, and legacy NEs easily supports IP, it may be unnecessary to throw away legacy NEs if it can be made to fit within a PBNM approach.

Our approach enables an existing PBNM system to include legacy NEs in its management paradigm. The Proxy-Policy Enforcement Point (P-PEP) and Queuing-Policy Enforcement Point (Q-PEP) can enforce some degree of traffic shaping on a gateway to the legacy portion of the network. The P-PEP utilises firewall techniques using the common legacy and contemporary NE management protocol Simple Network Management Protocol (SNMP) while the Q-PEP uses queuing techniques in the form Class Based Queuing (CBQ) and Random Early Discard (RED) for traffic control.

DECLARATION

I declare that Policy Based Network Management for Legacy Network Elements in Next Generation Networks for Voice Services is my own work, that it has not been submitted for any degree or examination in any other university, and that all the sources I have used or quoted have been indicated and acknowledged by complete references.

Vaughn Naidoo Friday, July 19, 2002

Signed:-

TABLE OF CONTENTS

LIST OF FIGURES AND GRAPHS

LIST OF TABLES

ACKNOWLEDGEMENTS

My thanks and appreciation goes to my supervisor Mr William Tucker whose input, support and patience is highly valued; and for his interest in my personal development.

A special word of thanks goes to Nadia Williams for her encouragement and support through this learning process.

I also wish to thank my family, Suleiman and Farouz Adams for their support, love and understanding during all my studies.

My fellow students, Tania Paulse and Michael Jeffries for their camaraderie.

CHAPTER 1

INTRODUCTION TO THE RESEARCH

1.1 Introduction

Voice services are currently being deployed on packet switched networks in the form of Voice over Internet Protocol (VoIP). The Internet Protocol (IP) is a connectionless and unreliable technology and therefore cannot provide a guaranteed and reliable user acceptable level of service for real-time services - which includes VoIP - by itself. Contemporary Network Elements (NEs) such as routers and switches support Quality of Service (QoS) protocols such as ReSource reserVation Protocol (RSVP), MultiProtocol Labeled Switching (MPLS) and Differentiated Services (DiffServ). These protocols have resolved many QoS issues that are required for the deployment of user acceptable voice services on packet switched networks. Future NEs will automatically have support for these QoS protocols. However, legacy NEs, defined as NEs that do not have the protocols to support voice adequately, are regarded as unsuitable for voice communications. The focus of this research was aimed at bridging the gap between QoS enabled contemporary NEs and legacy NEs that have no ability to provide QoS. The goal was to show how to provide QoS for legacy filled networks with a Policy Based Network Management (PBNM) paradigm [30].

1.2 The Problem

One could surmise that in order to provide voice services on an existing network, one must purchase voice-enabled equipment that supports the necessary QoS (RSVP, MPLS, DiffServ) protocols. This is a common held opinion that has risen from legacy NE's inability to provide QoS for any real-time service [51]. However, legacy NEs' predominate and have the same abilities to push packets through their ports, but lack the intelligence to provide QoS. These NEs can thus not support voice services during network congestion

and saturation periods. This is certainly not a basic hardware problem but a configuration limitation.

Legacy NEs not only lack QoS functionality, but also lack the protocols and mechanisms necessary for PBNM systems such as the Common Open Policy Service (COPS) Protocol [17] which is used to manage QoS. The only network management commonality between legacy and contemporary NEs is the Simple Network Management Protocol (SNMP) [48]. Although voice services can be deployed on legacy filled networks, they cannot provide user acceptable levels of service as they lack all the protocols necessary for QoS provision.

Telecommunications started with Alexander Graham Bells' invention of the telephone in 1876 [54], which has developed into today's Plain Old Telephone System (POTS) system. The Public Switched Telephone Network (PSTN), the architecture on which the POTS system is deployed, has the innate ability to deliver the world renowned 5 9s level of QoS. This means that QoS is provided 99999 times out of a possible 100000. From this, it is evident why the expectation of the traditional voice service users are high. Thus, a successful deployment of voice services on IP-based packet switched networks requires the QoS to either be similar to the QoS delivered by the PSTN or better. To do this, QoS protocols are necessary.

During non-congestive network periods, legacy NEs can provide user acceptable QoS. However, if the network becomes moderately busy, the possibility exists that the QoS experienced by IP services can degrade, which is not acceptable for real-time services. It is these conditions/situations that QoS protocols were designed for. These protocols compensate for IP's natural service delivery behaviour of providing "best effort" service delivery, by providing a constant and reliable level of QoS.

Depending on the QoS protocol, under congestive or non-congestive network periods, IP services that require high levels of QoS can be provided for, by either, dedicated bandwidth reservation or by service prioritisation. Due to this, QoS prevails during congestive and non-congestive network periods which is not the case without these QoS protocols.

1.3 Research Goals

The aim of this study was to investigate whether it was possible to deploy voice services on legacy filled networks that provide some level of QoS using either the common network management protocol, SNMP within a PBNM paradigm, the Proxy-Policy Enforcement Point (P-PEP) or by including legacy NEs in a PBNM paradigm together with traffic shaping methodologies, the Queuing-Policy Enforcement Point (Q-PEP). The intention was to: integrate the legacy NEs with QoS enabled NEs (contemporary) (refer to Figure 1.1); provide QoS through the proposed software solutions, the P-PEP and Q-PEP, and fit the QoS mechanism/solution to PBNM and COPS. This results in COPS controlling the QoS regardless (refer to Figure 1.1) of having legacy NEs in the network. The ideal situation is to leave as much legacy NEs as possible in a network while introducing voice-enabled NEs on an as-needed basis. If Legacy NEs are to be included in contemporary networks, they should in no way lower the QoS provided by increasing network congestion or decreasing the network saturation point. Network saturation occurs when the network has no more resources to allocate to a service. The research will be proven a success if within mixed legacy and contemporary networks, better QoS is provided. There seems to exist a possibility to bridge the gaps between the two types of NEs in order to provide enough QoS to deliver acceptable voice services.

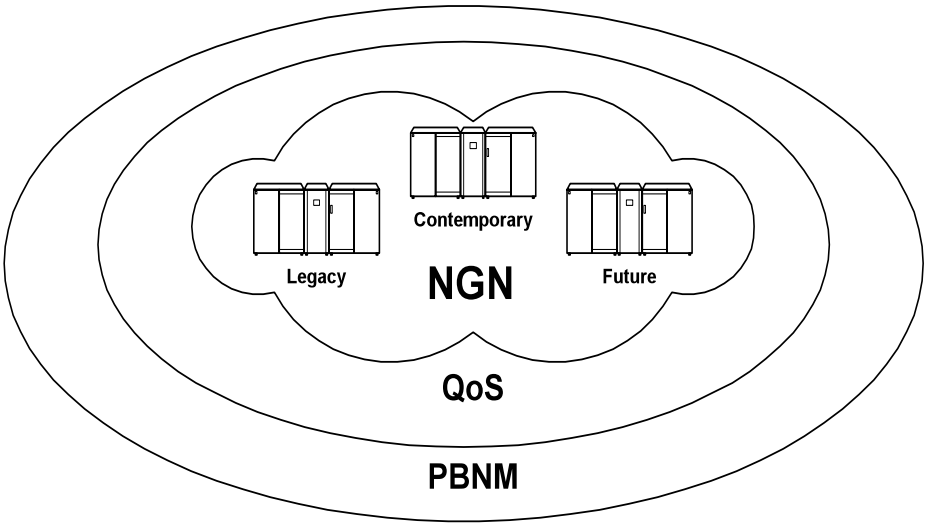


Figure 1.1: The Goal.

1.4 Research Questions

The research questions which guided the study, are as follows:

1. Is it possible for legacy NEs to operate in a PBNM environment?

2. Is it possible for SNMP to emulate a PBNM environment?

3. Can SNMP be used to provide QoS when used to provide admission control?

4. Does traffic shaping in the form of Class Based Queuing (CBQ) queuing provide QoS within a PBNM paradigm during periods of network congestion?

5. What are the affects of employing traffic shaping on legacy IP NEs?

6. How do the legacy NEs affect the QoS provided by contemporary NEs?

1.5 Significance of the Study

Contemporary NEs that support the necessary protocols for voice services are expensive and are not affordable by all. We will show how QoS can be provided by SNMP using admission control or by using PBNM together with traffic shaping methodologies. If this is proven to be so, voice services can be made available to all, especially underprivileged institutions which cannot keep up with the latest technological advances for the deployment of income generating services, such as distance learning (DL). Money can be saved by deploying voice services on the Local Area Network (LAN) due to the flat-rate pricing of the public Internet when compared to telecommunication costs as there are no additional constraints imposed on long distance calls.

1.6 Thesis Structure

The thesis follows an eight chapter structure; Chapter 1 sets the stage for the research and delivers a brief overview of the entire Thesis. Chapter 2, puts the area of research into

perspective by delivering a literature review of the technologies used, the flaws with them and the paradigm shifts with respect to voice services and why these shifts have occurred. After the in depth literature review, Chapter 4 presents the research approach taken to solve this research problem. Chapter 5 presents a design of the proposed P-PEP solution and Chapter 6 of the Q-PEP solution. Chapter 7 presents results based on the various testing scenarios and discrepancies explained through valid reasoning. Chapter 8 concludes with an assessment of and recommendations of the solutions, and observations on future work. The next chapter will provide a review of the current literature to show what mechanism or methods are available and can be used to solve this research problem.

1.7 Overview of Chapter 1

There are many prohibiting factors associated with deploying voice services on legacy filled networks. The main regards the level of QoS provided. The problems with legacy NEs with respect to the quality of voice services delivered under congestive and saturated periods have brought people to the opinion that voice services cannot be deployed on legacy filled NEs.

CHAPTER 2

LITERATURE REVIEW

This chapter discusses the technologies and deployment issues concerning the management of the QoS for voice services in IP environments. A comparison of the old and the new architectures for voice services that are currently being deployed will be presented, revealing their disadvantages, advantages and methods used to control network traffic flow. A discussion follows about the frameworks and mechanisms needed and deployed on these new architectures to equate the QoS for voice services. We then discuss how these mechanisms are controlled, managed and how they can be applied to networks. In trying to find a solution, we compare contemporary NEs that have the necessary mechanisms to provide QoS to legacy NEs that need these mechanisms to overcome QoS problems. We then present the hardware needed to deploy voice services on these new architectures and why companies would want to do so. The next section deals with the comparison of the network architectures used to deploy voice services and discuss these advantages and disadvantages of them.

2.1 Voice Network Architectures: Circuit vs Packet Switching

2.1.1 *Circuit Switched Voice*

Voice services have been traditionally deployed on circuit switched infrastructures set up by telecommunication companies. With these infrastructures, circuits are created reserving network resources all the way from sender to receiver before the start of the transfer. The resources are dedicated to the circuit during the entire transfer. Control signalling such as routing is performed at circuit set up and termination, and is thus kept separate from the payload in data transfers. This results in no overhead in the transferred payload data within the circuit and thus reduces network delay [43].

Circuit-switched networks allow for large amounts of data to be transferred with guaranteed transmission capacity, thus providing support for real-time traffic. However, if the connections are short-lived, the set up delay may represent a large part of the total connection time, thus reducing the network's capacity. Resources reserved by circuit switched networks cannot be used by any other users even if the circuit is inactive, which may further reduce link utilisation. Being able to allocate dedicated bandwidth to every connection set up is ideal for QoS provision that is required for voice services [49]. As more and more resources are allocated, there exists a possibility that network resources can become saturated. This would not result in the degradation of QoS, but rather in services being denied [39]. Looking at the physical aspects of the telecommunications networks, their architecture provides QoS without any additional network configuration or protocols. This differs tremendously for the IP-based Packet Switched networks on which the proposed solutions will be deployed.

2.1.2 *Packet Switched Voice*

Packet switching was developed due to circuit switching's inability to handle random bursts of traffic. In packet switched networks, a data stream is divided into standardised packets, which contain address, size, sequence, error-checking information and the payload data. The packets are then sent through the network, where specific packet switches or routers sort and direct individual packets [13]. Packet switched networks can be thought of as networks within a network of queues.

Each network node contains queues where incoming packets are queued before they are sent out on an outgoing link. If the rate at which packets arrive at a switch point exceeds the rate at which packets can be transmitted, the queues grow. The queuing causes delay, and if the queue overflows, congestion results and packets are lost. Loss of data generally causes retransmissions that may either add to the congestion or result in less-effective utilisation of the network. The ability to support voice traffic in packet switched networks thus calls for advanced control mechanisms such as Random Early Discard (RED) and Explicit Congestion Notification (ECN) for buffer handling and Link State Routing

Protocols and MPLS for intelligent network traffic control [21]. Packet switched networks are based either on connection-oriented or connectionless technology.

In connection-oriented technology such as the Transmission Control Protocol (TCP), a path through the network is established when data transfer begins. Each packet header contains an identifier that is used at the nodes to guide each packet to the correct destination. In connectionless technology, such as IP, packets are treated independently of each other inside the network, as complete information regarding the destination is contained within each packet [31]. The possibility thus exists that the packet order may not always be preserved since packets destined for the same receiver may take different paths through the network, thus causing loss in the QoS experienced by voice services.

2.1.2.1 Internet Protocol

IP is the world's most popular non-proprietary protocol suite because it can be used to communicate across any set of interconnected networks and are equally well suited for both LAN and Wide Area Network (WAN) communication. IP has two primary responsibilities: providing connectionless, best-effort delivery of datagrams through an internetwork and fragmentation and reassembly of datagrams to support data links with different maximum-transmission unit (MTU) sizes [13]. In an IP environment, reliable transmission is provided through TCP.

2.1.2.2 Transmission Control and User Datagram Protocol

TCP corresponds to the transport layer (Layer 4) of the Open System Interconnect (OSI) reference model [26]. TCP provides services for streaming data, reliability through packet acknowledgement and retransmission, efficient flow control through buffering techniques, full-duplex operation, and multiplexing. User Datagram Protocol (UDP) is a connectionless transport-layer protocol (Layer 4) which also belongs to the TCP/IP suite. Unlike the TCP, UDP adds no reliability, flow-control, or error-recovery functions to IP. Because of UDP's simplicity, UDP headers contain fewer bytes and consume less network overhead than TCP. UDP is useful for applications where the reliability mechanisms of

TCP are not necessary, such as in cases where a higher-layer protocol might provide error and flow control, or where retransmission has no effect (e.g. voice services) [30].

TCP/IP services are never denied, unless restricted by network administrators. Services on IP networks degrade gracefully when network resources become saturated. Since VoIP uses IP to transmit voice as packets over an IP network, VoIP can be achieved by any data network that uses IP [56]. Just as TCP was developed to provide reliable data communications on IP-based packet switched networks, QoS protocols were developed to provide user acceptable real-time services. These services include VoIP, video conferencing and video streaming. We now move to IP-based NGNs where the QoS for voice services must be addressed.

2.2 Quality of Service

All types of networks, whether circuit or packet switched, have a saturation point. What is important is the time taken or the amount of traffic needed to reach this point. Network saturation has a direct effect on the level of the QoS provided. Thus, avoiding the network saturation point prolongs QoS provision. It is important to control what happens when the network saturation point is reached. In other words, when the network has no more resources left to allocate, we must be able to control which services degrade and which services maintain their level of QoS, thus increasing network predictability. According to Ferguson & Huston [22], QoS means "providing a consistent and predictable data delivery service" that results in NEs having some level of assurance that their traffic and service requirements can be satisfied. Furthermore, "Any QoS assurances are only as good as the weakest link in the path between sender and receiver" [22]. Thus QoS requires the cooperation of all network layers from top-to-bottom, as well as every NE from end-to-end.

According to Ma & Bingxin [40], a common misconception is that "QoS creates bandwidth." They correctly point out that it is not possible for a network to provide what it does not have. QoS only manages available bandwidth according to application demands and network management settings. Thus, if applications involves sharing of resources,

QoS provision cannot be guaranteed. If guaranteed service levels are required, resources should be allocated as individual data streams. A priority for QoS designers has been to ensure that best-effort traffic is not starved after reservations are made. QoS-enabled (high-priority) applications must not disable the everyday (low-priority) Internet applications. However, the possibility exists that these applications can disable the everyday Internet applications and contravened the traditional IP paradigm.

2.2.1 *The Need for QoS*

The architecture of the Internet is based on the simple concept that datagrams with source and destination addresses traverse IP networks without the help of their sender or receiver. There is a price to pay for this simplicity. IP does not provide reliable data delivery. Routers are allowed to discard IP datagrams en route to the destination without notice to sender or receiver. IP relies on upper-level protocols to keep track of datagrams, and retransmissions as necessary. These reliability mechanisms can only assure data delivery. Neither IP nor its high-level protocols can ensure timely delivery or provide any guarantees about data throughput. IP provides what is called a "best effort" service [30]. It can make no guarantees about when data will arrive, or how much it can deliver.

This limitation has not been a problem for traditional Internet applications like web, email and file transfer. However, the new breed of applications, including real-time audio and video streaming, demand high data throughput capacity and low latency when used in two-way communications. These applications have also increased user expectations about the quality and timely presentation of information [42].

Unlike circuit technologies such as ATM and Frame Relay, IP does not make a provision of resources. These technologies provide much more efficient use of the available bandwidth, and are more flexible [42]. Voice traffic is bursty rather than continuous. IP is datagram-based so it uses the available bandwidth most efficiently, by sharing what is available as needed. This allows IP to adapt more readily to applications with varying bandwidth needs. However, it also leads to some unpredictability in service. The capacity to tolerate this unpredictability relates to the level of guarantee the applications require.

2.2.2 *QoS Frameworks*

According to Karve [36], QoS generally encompasses bandwidth allocation, prioritisation and control over network latency for network applications. Karve claims that there are several ways to ensure QoS, the easiest though is to simply upgrade the bandwidth until service quality becomes acceptable.

2.2.2.1 Over provisioning

The most prominent solution to handle peak periods is to over-provision the network [1], i.e. to provide surplus bandwidth capacity in anticipation of these peak data rates during high-demand periods. This strategy, however, collapses if a network becomes moderately busy. In a complex environment, i.e. one that has a lot of data packets moving in many paths throughout the network, or that has a mixture of data and real-time applications, bottlenecks and congestion occur. Bandwidth allocation does not address the need to distinguish high-priority from lower-priority traffic flows which results in all the traffic being treated equally. This equality is not good, as network traffic by nature is unpredictable. Peak data rates and the network regions on which they might occur are seldom possible to predict, although they occur most notably at LAN/WAN aggregation points. As one can see, additional bandwidth can solve some short-term QoS problems for voice services, but it is not a viable long-term solution, particularly if you already have enough bandwidth to accommodate all but the most highly sensitive network applications. Over provisioning the network results in the reduction of the occurrence of network congestion and saturation. However, this solution does not allow for the prioritisation of voice services and cannot provide reliable or user acceptable services during periods of network congestion. This solution is also not economically viable, at least not with today's bandwidth technologies and infrastructures, especially for WAN links.

Best effort service delivery cannot always provide a usable service, let alone an acceptable one. Even on a relatively unloaded IP network, delivery delays can vary enough to adversely affect voice services. To provide service guarantees, IP services must be supplemented with the ability to distinguish between different traffic and ensure different service levels for different users and applications. Two solutions were put forward to solve

the QoS problem, namely the Integrated Services (IntServ) Architecture [12] and the Differentiated Services (DiffServ) framework [3].

2.2.2.2 IntServ Reservation-based Architecture

The Internet Engineering Task Force (IETF) Integrated Service Working Group has put forward a proposed extension to the Internet architecture and protocols to provide integrated services. To support real-time as well as non-real-time IP services. The IntServ framework gives applications the ability to choose among multiple, controlled levels of QoS for their data packets. Network resources are allocated according to each application's QoS request subject to bandwidth management policies. These new components and mechanisms supplement basic IP services [12]. IntServ defines three classes of service: Firstly, guaranteed service [57] with guaranteed bandwidth, bounded delay, and no-loss guarantees. Secondly, controlled load service [58] which approximates best-effort services in a lightly loaded network and finally best-effort service, similar to what the Internet currently provides under a variety of load conditions [57]. Real-time services require service guarantees, and those guarantees cannot be achieved without reservations. RSVP[1] provides the mechanisms to do this as a part of the IntServ architecture [12]. RSVP assumes that resources are reserved for every flow requiring QoS at every router hop in the path between receiver and transmitter using end-to-end signalling [5]. This, in turn, requires flow-specific state in the routers, which represents an important and fundamental change to the Internet model.

The IntServ model has several strengths. RSVP is designed to operate with current and future unicast and multicast routing protocols [61]. The most problematic issue for IntServ concerns the scalability of RSVP and the amount of resources a router needs as RSVP processing and storage increases proportionally with the number of QoS flows [12]. Traffic measurements show that most end-to-end IP connections are very short-lived, and that there are several thousand active connections at any time in a backbone router. Consequently, numerous IntServ flows on a high-bandwidth link place an excessive burden on routers. RSVP could also lead to a perceived degradation of some network

[1]RSVP is defined in RFC 2205

services. In a non-RSVP network, an application might run slowly or badly, but at least it will run. In an RSVP network, the application might not run at all if there is not enough bandwidth support. Similarly, if high-priority requests consume all the allotted bandwidth, a router could dump non-prioritised flows [38]. Another downside to the IntServ model is that, although applications receive dedicated bandwidth that is required to provide acceptable levels of QoS, there is no method of giving priority to a particular service or application which is needed when network saturation occurs. Bandwidth is allocated on a "first come first serve" basis and the possibility exists that network saturation may occur before voice services are serviced. This brings us to the DiffServ framework.

2.2.2.3 DiffServ Prioritization Framework (reservation-less)

The DiffServ Framework being defined by the IETF is intended to meet the need of providing differentiated service classes for Internet traffic [3]. One must remember that the Internet supports various types of applications, which have different but specific requirements. Diffserv was developed to overcome the limitation of IntServ/RSVP models by providing scalable service discrimination without the need for per-flow state and signalling at every hop. The differentiated services approach to providing QoS in networks employs a small, well-defined set of building blocks from which a variety of services may be built [49]. A small bit-pattern in each packet, in the IPv4 TOS octet or the IPv6 Traffic Class octet, is used to mark a packet to receive a particular forwarding treatment, or per-hop behaviour, at each network node. Network traffic is classified and network resources are allocated according to bandwidth management policy criteria. To enable QoS, network traffic classification mechanisms such as DiffServ give preferential treatment to applications identified as having more demanding requirements [2].

Unlike RSVP, no QoS requirements are exchanged between the source and the destination, eliminating the inherent set up costs associated with RSVP [52]. Short-lived flows benefit from DiffServ because the absence of QoS set up costs improves responsiveness and reduces the overhead required for communication with another NE. DiffServ only maps services with different levels of sensitivities to delay and loss without being associated with explicit values or guarantees [3]. It does not attempt to guarantee a

level of service. Instead, it strives for a relative ordering of aggregations, such that one traffic aggregation will receive better or worse treatment relative to other aggregations, based on the rules defined for each aggregation [20]. Thus, under periods of network congestion, QoS can be provided for voice services.

According to Ma and Bingxin [40], only a fraction of applications need strong and explicit QoS guarantees. Proper network engineering and broad traffic classification into a smaller number of priority classes, coupled with the adaptive nature of many applications, may be sufficient to offer the necessary functionality. As a result, adequate provisioning for peak traffic loads, together with protection from lower priority traffic, will provide the applications that require QoS with the desired level of service. Adequate provisioning also ensures that the rest of the traffic experiences adequate service most of the time, possibly with some differentiation. In case of congestion, flows will adapt their traffic to the available resources and continue operating, although at lower levels of service. The benefit is higher overall efficiency as a result of more flows getting through with greater simplicity, minimal signaling support, and simple data-path mechanisms [26].

2.2.3 *Voice QoS Parameters*

2.2.3.1 Latency

Latency is defined as the time period between a NE sending a message and receipt of the message by another NE. This encompasses the delay in a transmission path or in a NE within a transmission path. In a router, latency is the time period between the arrival of a data packet and when it is retransmitted. This is sometimes referred to as propagation delay [42].

When round-trip delays exceed approximately 300 microseconds, natural human conversation becomes difficult. Depending on the type of voice-compression method used, each one-way VoIP transmission requires between 32 Kbps to 64 Kbps of bandwidth. Some compression methods such as G.729 take the bandwidth required below 8 Kbps. Bandwidth required for VoIP sessions are relatively low. The challenge is to make

that bandwidth available regardless of network utilisation [56]. Delay in VoIP networks results in two possible problems; talker overlap and echo [60].

Talker overlap becomes significant if the one-way delay becomes greater than 250 milliseconds. Echo is caused by the signal reflections of the speaker's voice from the far end telephone equipment back into the speaker's ear. Echo becomes a significant problem when the round trip delay becomes greater than 50 milliseconds [37]. Delay in end to end VoIP calls originate from algorithmic, processing and network delay.

Algorithmic delay is caused by the collection of voice frame samples to be processed by the voice coder. It is related to the type of voice coder used and varies from a single sample time, 0.125 microseconds to many milliseconds. A representative list of standard voice coders and their frame times are as follows:

- G.726 ADPCM (16, 24, 32, 40 Kbps) - 0.125 milliseconds [37]
- G.728 - LD-CELP(16 Kbps) - 2.5 milliseconds [37]
- G.729 - CS- ACELP (8 Kbps) - 10 milliseconds [37]
- G.723.1 - Multi Rate Coder (5.3, 6.3 Kbps) - 30 milliseconds [37]

Processing delay is caused by the actual process of encoding and collecting the encoded samples into a packet for transmission over the packet network. The encoding delay is a function of both the processor execution time and the type of algorithm used. Often, multiple voice coder frames will be collected in a single packet to reduce the packet network overhead [19].

Network delay is caused by the physical medium and protocols used to transmit the voice data, and by the buffers used to remove packet jitter on the receiving NE. Network delay is a function of the capacity of the links in the network and the processing that occurs as the packets transit the network. The jitter buffers add delay which is used to remove the packet delay variation that each packet is subjected to as it traverses the packet network.

This delay can be a significant part of the overall delay since packet delay variations can be as high as 70-100 milliseconds in some Frame Relay networks and IP networks [19].

2.2.3.2 Jitter

Jitter results in abnormalities in video or voice that is transmitted over a network due to packets not arriving at their destination in consecutive order or in a timely manner. Jitter results from signal distortion as it is propagated through the network, resulting in the signal varying from its original reference timing. In packet switched networks, jitter is a distortion of the interpacket arrival times compared to the interpacket times of the original transmission [56].

Removing jitter requires collecting packets and holding them long enough to allow the slowest packets to arrive in time to be played in the correct sequence. However, this causes additional delay. The two conflicting goals of minimising delay and removing jitter have caused various schemes to adapt the jitter buffer size to match the time varying requirements of network jitter removal [56].

2.2.3.3 Packet loss

Packet loss is the number of IP Packets lost in transit to the destination host. As IP networks do not guarantee service, they will usually exhibit a much higher incidence of lost voice packets than ATM networks. In current IP networks, all voice frames are treated like data. Under peak loads and congestion, voice frames will be dropped equally with data frames. The data frames, however, are not time sensitive and dropped packets can be appropriately corrected through the process of retransmission. Lost voice packets, however, cannot be dealt with in this manner [42]. There are several measures to evaluate voice quality.

In the past, telephone companies assembled people in a room to listen to voice. The average opinion of all the participants was referred to as the Mean Opinion Score (MOS) [33]. Perceptual Speech Quality Measurement (PSQM) is a voice quality scoring system endorsed by the ITU [32]. PSQM attempts to listen to voice as a human being would listen, and rates the received voice sounds accordingly. It does this by measuring how

much noise has been added to the originally transmitted voice signal. PSQM ratings range from 0 to 6.5, with 0 noise being the most desirable [55].

Perceptual Analysis / Measurement System (PAMS) provides an objective measure that predicts the results of subjective listening tests on telephony systems. To measure speech quality, PAMS uses a sensory model to compare the original, unprocessed signal with the degraded version at the output of the communications system. PAMS parameterises different classes of error and maps them to predictions of subjective listening quality and listening effort. The mappings are calibrated using a large database of subjective tests. PAMS ratings range from 0 to 5, with 5 noise being the most desirable [47].

Perceptual Evaluation of Speech Quality (PESQ) is a method of determining the voice quality in the telecommunications networks. It combines the time-alignment technique from PAMS with the accurate perceptual modelling of PSQM. PESQ is applicable not only to speech codecs, but also to end-to-end measurement. Similarly to PAMS, PESQ PAMS ratings range from 0 to 5, with 5 noise being the most desirable [47].

2.2.4 *QoS mechanisms*

QoS mechanisms are used within the IntServ and DiffServ framework to aid QoS provision for IP services. These mechanisms include: admission control; traffic shaping and conditioning; packet classification; packet marking; priority mechanisms; queuing; congestion control; flow control and signalling protocols which will be discussed.

2.2.4.1 Admission control

Admission Control determines whether a requested connection is allowed to be carried by the network. The main considerations behind this decision are current traffic load, actual QoS being achieved, requested traffic profile, requested QoS and other policy considerations [34].

2.2.4.2 Traffic shaping/conditioning

In QoS enabled IP networks, it is necessary to specify traffic profiles for connections in order to allocate the required network resources. Traffic shaping/conditioning ensures that

traffic entering at an edge or a core node adheres to specified profiles. This mechanism is used to reduce the burstiness of traffic streams [18].

Traffic shaping forces network traffic to conform to specified characteristics which results in predictable network behaviour. By knowing the precise network traffic behaviour, it is possible to allocate resources inside the network such that guarantees about availability of bandwidth and maximum delays can be given [18].

2.2.4.3 Packet classification

In order to provide the requested QoS, it is critical to classify packets to enable different QoS treatment. This can be done based on various fields in IP headers, source/destination addresses, protocol type, higher layer protocol headers and source/destination port numbers [30].

2.2.4.4 Packet marking

Either as a result of a traffic monitoring mechanism or voluntary discrimination, a packet can be tagged for a particular QoS treatment in the network using the IP header's TOS byte for IPv4 and Traffic Class byte for IPv6 [20].

2.2.4.5 Priority mechanisms

The Priority feature refers to the capability of providing different delay treatment for packets by allowing higher priority packets to be served before the lower priority ones. NEs implement different priority loss treatment, the higher loss priority packets are lost less often than the lower loss priority ones. A priority mechanism is used to satisfy the QoS needs of different connections [7].

2.2.4.6 Queuing

Queuing algorithms are used in NEs to manage congestion. Certain NEs enable fair queuing algorithms so that applications which continue to generate excessive traffic during periods of congestion, do not lower the QoS delivered by other applications. Queuing determines how packets are dropped when congestion occurs in a router [7]. Queuing algorithms include:

- First In First Out (FIFO): FIFO follows a basic store and forward paradigm. It involves storing packets when the network is congested and forwarding them in order of arrival when the network is no longer congested. FIFO is the default and only queuing algorithm for many NE's, thus requiring no configuration. It, however, has several shortcomings. Most importantly, FIFO queuing makes no decision about packet priority; the order of arrival determines the bandwidth, promptness, and buffer allocation. It does not provide protection against ill-behaved applications either. Bursty sources can cause long delays in delivering time-sensitive application traffic [53].

- CBQ: Incoming packets are divided into user definable classes. These divisions can fall along the lines of traffic from a given interface, traffic associated with a particular application, traffic intended for a particular network or device destination, and all traffic of a specific priority classification. Each class of traffic is assigned to a specific FIFO queue, each of which is guaranteed some portion of the total bandwidth [25]. Thus, priority is given to classes by allocating larger bandwidth portion thereby extending the queues.

- Weighted Fair Queuing (WFQ): WFQ is a method used for segmenting traffic into multiple queues, giving greater weight to certain traffic types by assigning larger queues. WFQ is designed to prevent any one traffic type from entirely eclipsing another. By default, WFQ favours lower-volume traffic flows over higher-volume ones [16].

Queuing, which fits into the DiffServ model by prioritising traffic according to user specified specifications, is a viable solution to provide some level of QoS for the legacy filled portion of the network for voice services. Using this mechanism, any service that can be uniquely identified can be given priority over other services by allocating portions of the available bandwidth. Queuing mechanisms will not only provide priority for specified traffic types, but will also reduce the burstiness of traffic flows [7].

2.2.4.7 Congestion Control

In order for IP networks to operate in a stable and efficient fashion, it is essential that they have viable and robust congestion control capabilities to maintain QoS levels. These capabilities refer to the ability to control the flow (flow control) and to shed excessive traffic during the periods of congestion. Random Early Detect (RED) and Early Congestion Notification (ECN) are two prominent congestion control mechanisms. RED prescribes discard probabilities to drop packets in a fair and robust way based on the measured average queue length. RED attempts to avoid congestion rather than reacting to it. It randomly drops packets before queues fill, thus keeping them from overflowing [10]. RED is a viable mechanism to prevent or slow down network congestion. The problem with all congestion algorithms is that once congestion is detected and packets are dropped, it is possible that the network congestion could become worse in the end [21]. This is due to the fact that the packets that were dropped need to be retransmitted either by TCP or by higher applications, this can result in simultaneous flooding of network traffic from all sources affected by the congestion algorithm.

2.2.4.8 Flow Control

Flow control is a function that prevents network congestion by ensuring that transmitting devices do not overwhelm receiving devices with data. Countless possible causes of network congestion exist. A high-speed computer may generate traffic faster than the network can transmit it, or faster than the destination device can receive and process it. According to Bennett & Zhang [35], the three commonly used methods for handling network congestion are buffering, transmitting source-quench messages, and windowing.

Buffering is used by network devices to temporarily store bursts of excess data in memory until they can be processed. Occasional data bursts are easily handled by buffering. Excess data bursts can exhaust memory forcing the device to discard any additional datagrams that arrive [13].

Source-quench messages are used by receiving devices to help prevent their buffers from overflowing. The receiving device sends source-quench messages to request that the

source reduces its current rate of data transmission. When buffers at the receiving device begin discarding received data due to overflowing buffers, the receiving device begins sending source-quench messages to the transmitting device at the rate of one message for each packet dropped. The source device receives the source-quench messages and lowers the data rate until it stops receiving the messages. The source device then gradually increases the data rate as long as no further source-quench requests are received [26].

Windowing is a flow control scheme in which the source device requires an acknowledgement from the destination after a certain number of packets have been transmitted. With a window size of three, the source requires an acknowledgement after sending three packets, as follows: First, the source device sends three packets to the destination device. Then, after receiving the three packets, the destination device sends an acknowledgement to the source. The source receives the acknowledgement and sends three more packets. If the destination does not receive one or more of the packets for some reason, such as overflowing buffers, it does not receive enough packets to send an acknowledgement. The source then retransmits the packets at a reduced transmission rate [13]. Buffering, windowing source-quench messages are built within the TCP/IP stack and no extra configuration is needed. These mechanisms have no priority functionality and work fine for non real-time services, but not for real-time services. When congestion is detected, these mechanisms affect all network traffic and this cannot be changed.

2.2.4.9 Signalling protocols

In order to obtain the required QoS from a network, end-systems need to signal the network for the desired QoS as well as the anticipated offered traffic profile. This has been a fundamental part of various connection-oriented networks. For connectionless networks such as IP, this is relatively new and is implemented with RSVP [61]. Many of the QoS standards and protocols discussed are still in their infancy and not yet widely deployed. The growing need for QoS in IP-based networks will soon make them a requirement for all networks.

However, many problems still exist and will do so into the distant future. RSVP, multiple queues, and packet tagging can be valuable mechanisms to provide QoS, but they are not the only ways to address QoS problems [59]. The next section discusses a fairly new idea in the QoS realm: making networks themselves more intelligent by incorporating Policy Based Network Management (PBNM) which is used to manage QoS.

2.3 Policy Based Network Management

The rise of PBNM has resulted from the growing need for automated QoS management. In the PBNM paradigm, a network manager creates policies to define how resources or services in the network can be used. The PBNM system transforms these policies into configuration changes and applies these changes to the network. Policies provide a mechanism of abstraction to simplify the management of QoS and security mechanisms. They contain rules that govern how resources can be used, or how applications and users should be treated [28].

To overcome the significant costs of buying additional bandwidth or living with the current congestion at the edges of a network, PBNM tries to alleviate bottlenecks created by the rapid increases in internetworking which occur most notably at LAN to WAN aggregation points. PBNM uses QoS mechanisms which introduce unfairness, i.e. some applications receive preferential treatment at the expense of others. PBNM policy rules express and control this desired unfairness [30].

These predefined policies contain one or more rules. These rules specify a set of conditions that, when met, result in an action being taken. A PBNM solution implements the policy rules in the network. The goal of PBNM is to dynamically meet the users' varied needs and expectations regarding how their applications perform by reducing the time, cost, and problems associated with individual device configuration [45].

2.3.1 *PBNM Architecture*

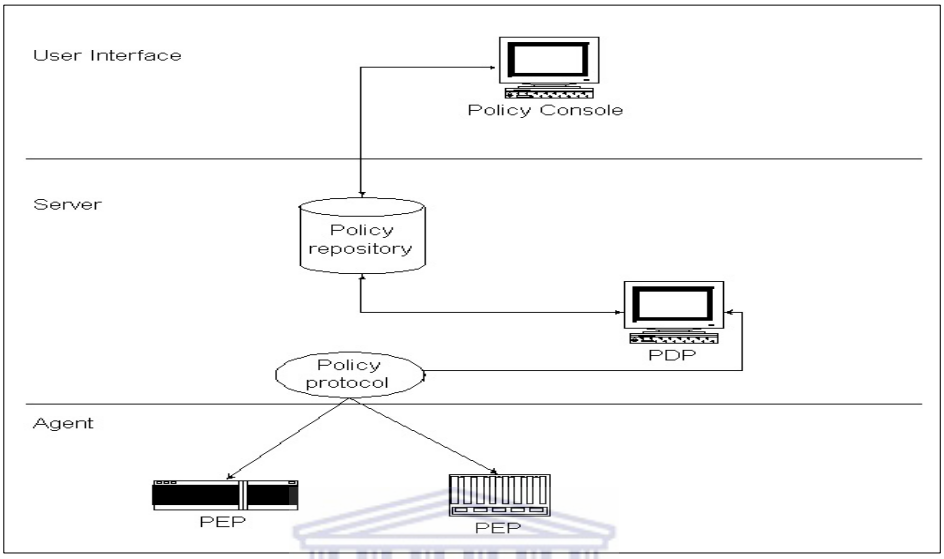


Figure 2.1: PBNM Architecture.

From Figure 2.1, the policy console is the user interface which is used to construct and deploy policies, and monitor the status of the PBNM environment. The Policy Decision Point (PDP) is an application that makes decisions based on policy rules and enforces these decisions on an Policy Enforcement Point (PEP). A PEP is an agent running on or within a NE that enforces a policy decision and/or makes a configuration change [45]. The PEP communicates with the policy server/PDP to obtain these policy decisions or directives. In the absence of a PDP, an optional Local Policy Decision Point (LPDP) can be used by the device to make local policy decisions. However, the PDP remains the authoritative decision point at all times.

The user-created policies are modified and deployed within the policy console and stored in a repository, e.g. a directory and/or other storage services. A policy communication protocol, Common Open Policy Service (COPS), is a protocol which allows communication between PDP and PEP within the context of a particular type of client [45]. This protocol employs a client/server model where the PEP sends requests, updates,

and deletes to the remote PDP and the PDP returns decisions back to the PEP. COPS uses TCP as its transport protocol for reliable exchange of messages and thus no additional mechanisms are necessary for reliable communication between a server and its clients [17].

2.3.2 *Common Open Policy Service (COPS)*

In the COPS PBNM model, communication between the PEP and remote PDP is mainly in the form of a stateful request or decision exchanges. However, the remote PDP may occasionally send unsolicited decisions to the PEP to force changes in previously approved request states. The PEP is responsible for initiating a persistent TCP connection to a PDP. The PEP uses this TCP connection to send requests to and receive decisions from the remote PDP. The PEP has the capacity to report to the remote PDP that it has successfully completed performing the PDP's decision locally. The PEP is responsible for notifying the PDP when a request state has changed and is no longer applicable due to events at the client or decisions issued by the server [17].

When the PEP sends a configuration request, it expects the PDP to continuously send named units of configuration data to the PEP via decision messages as per configuration request. When a unit of named configuration data is successfully installed on the PEP, a report message can be generated by the PEP and sent to the PDP confirming the installation. The server may then update or remove the named configuration information via a new decision message. When the PDP sends a decision to remove named configuration data from the PEP, the PEP will delete the specified configuration and send a report message to the PDP as confirmation. In its simplest form, the PEP can be thought of as a software application that makes requests for traffic types or services and based on the PDP decision, either enacts the request or does not [17]. A direct comparison between the legacy and contemporary NEs reveals that the legacy NEs lack not only QoS functionality but also the protocols and mechanisms necessary for PBNM, i.e. COPS. The only commonality found was the network management protocol, SNMP. The next section gives a brief introduction to SNMP and discusses the advantages and disadvantages of this network management protocol.

2.4 SNMP

SNMP is an application-layer protocol that facilitates the exchange of management information between network devices. It enables network administrators to manage network performance, find and solve network problems, and plan for network growth. A SNMP managed network consists of three key components: managed devices, agents, and network-management systems (NMSs) [8].

Managed devices collect and store management information and make this information available to NMSs using SNMP. An agent has local knowledge of management information and translates that information into a form compatible with SNMP. A NMS executes applications that monitor and control managed devices. NMSs provide the bulk of the processing and memory resources required for network [8]. Managed devices are monitored and controlled using four basic SNMP commands: read, write, trap, and traversal operations.

2.4.1 *SNMP Commands*

- The read command is used by an NMS to monitor managed devices. The NMS examines different variables that are maintained by managed devices.
- The write command is used by an NMS to control managed devices. The NMS changes the values of variables stored within managed devices.
- The trap command is used by managed devices to asynchronously report events to the NMS. When certain types of events occur, a managed device can be configured to send a trap message to the NMS.
- Traversal operations are used by the NMS to determine which variables a managed device supports and to sequentially gather information in variable tables, such as a routing table.

All these commands operate on information stored by the NE in a Management Information Base (MIB) [46].

2.4.2 *SNMP Management Information Base*

The MIB stores the information collected hierarchically, it is accessed using a network management protocol such as SNMP. MIBs are comprised of managed objects and object identifiers [27].

Managed objects are specific characteristics of a managed device. Two types of managed objects exist: scalar objects which define a single object instance and tabular objects which define multiple related object instances that are grouped together in MIB tables. An object identifier uniquely identifies a managed object in the MIB hierarchy. The MIB hierarchy can be depicted as a tree with a nameless root, the levels of which are assigned by different organisations. The managed object at input can be uniquely identified either by the object name or by the equivalent object descriptor [46]. SNMP's simplicity provides both advantages and disadvantages. A major disadvantage is that SNMPv2 is incompatible with SNMPv1.

SNMPv2 messages use different header and protocol data unit (PDU) formats than SNMPv1 messages. SNMPv2 also uses two protocol operations that are not specified in SNMPv1. RFC 1908, however, defines two possible SNMPv1/v2 coexistence strategies: proxy agents and bilingual network management systems. SNMP lacks any authentication capabilities which results in vulnerability to a variety of security threats. Because SNMP does not implement authentication, many vendors do not implement SNMP Set operations, thereby reducing SNMP to a monitoring facility [27].

We have discussed the need for acceptable voice services, but we have not looked at the hardware needed to deploy voice services. But firstly, it is necessary to consider why companies want to deploy voice services on their existing data network while the PSTN provides a 99.999% service level.

2.5 VoIP

According to Golden Gateway [29], the consolidation of the separate voice and data networks offers opportunities for significant savings in the rising communications costs by

taking advantage of excess capacity on broadband networks for voice and data communications. VoIP has become especially attractive due to its low cost (compared to telecommunications), flat-rate pricing of the public Internet. Money will be saved by end users as there are no additional constraints imposed on long distance calls.

With VoIP, "telephone conversations are converted to a stream of IP packets and sent over an Ethernet network between two end-points" [14]. These streams are transported between these endpoints by the Real Time Protocol (RTP). RTP is a connection-oriented, end-to-end protocol that is designed to transport delay-sensitive information. RTP identifies the encapsulated payload type and includes sequence numbers and time stamps that can be used to synchronise real-time information flows. RTP uses the connectionless, unreliable UDP transport protocols rather than TCP because retransmission delays disrupt real-time audio and video streams. The Real Time Control Protocol (RTCP) works with RTP to provide sending software with feedback on the QoS being experienced by the receiver. RTP reports QoS parameters including packet loss and the amount of jitter being experienced. The sender can adjust transmission rates based on this feedback [50].

Ohio State University [44] argues that many keys issues need to be resolved in order for VoIP services to become popular. Some of these issues stem from the fact that IP was designed for transporting data; other issues have arisen because vendors are not conforming to the standards. As previously discussed, IP does not provide any real-time guarantees but only best-effort services. In a public network environment, products from different vendors need to operate with each other. Security issues exist because, within the Internet, anyone can capture the packets meant for someone else.

According to Kulathumani [38], one of the main issues when implementing VoIP is the design of an IP-based network that meets QoS requirements that are comparable in performance to conventional circuit-switched telephone networks. The high latency forwarding and best effort delivery provided by traditional routers, i.e. legacy NEs, is generally not acceptable for streaming VoIP as it provides neither maximum latency or minimum bandwidth guarantees. The International Telecommunications Union (ITU-T)

Recommendation H.323 is a global standard for packet-based multimedia communications, including VoIP.

H.323 is a set of network components and protocols that support real-time audio, video, and data communications. H.323 defines the data stream formats and protocols that endpoints use to communicate with one another. It also defines the management and control protocols used between terminals, gatekeepers, gateways and Multi-protocol Control Units (MCU) [37]. The diagram illustrated in Figure 2.2 depicts the components that make up an H.323 VoIP network. These components support real-time voice communications between end users and provide PBX-like network control functions. Terminals/endpoints are the end-user devices that support two-way, real-time voice communications across H.323 networks.

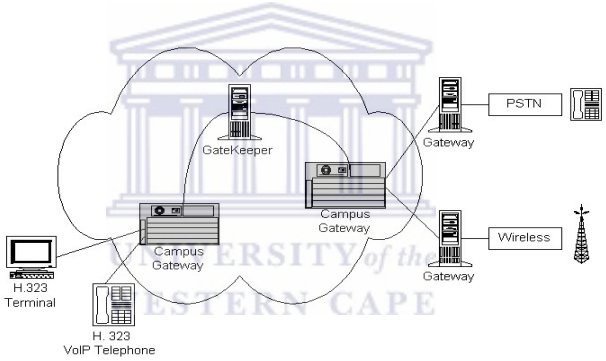


Figure 2.2: H. 323 VoIP Network

The key requirement for successful and acceptable VoIP communications is QoS provision which is driven by very high user expectations and cannot be provided by legacy NEs. The next section discusses the problems with legacy NEs with respect to QoS.

2.6 Legacy NEs

The main problem with legacy NEs is that VoIP packets cannot be differentiated from data packets containing part of an e-mail; thus, QoS cannot be provided for voice services. Both packets are received on an ingress port and need to be forwarded out the egress port. Legacy NEs might lack the intelligence to provide QoS but they have the same abilities to

push packets through their ports. Their QoS provision issues are not basic hardware problems but a configuration limitation. Legacy NEs predominate and not all institutions can afford to jump on the bandwagon and purchase voice enabled NEs that have resolved many QoS issues needed for voice services.

2.7 Summary of Chapter 2

The deployment of voice services on packet switched networks has led to the development of protocols and mechanisms necessary for the deployment of user acceptable voice services. These protocols are, however, only available on contemporary NEs and thus legacy NEs have been declared unfit for voice deployment. It is this statement that this study hopes to investigate and prove invalid by developing a software solution to provide some level of QoS for legacy NEs.

CHAPTER 3

THE PROBLEM

This chapter identifies the problems associated with deploying voice services on legacy filled networks, the research hypothesis and the research questions which guide the study. There are many prohibiting factors associated with deploying voice services on legacy filled networks. However, the main prohibiting factor regards the level of QoS provided.

3.1 VoIP Legacy NE Deployment Enigma

Since IP is connectionless and an unreliable protocol, it cannot provide a guaranteed and reliable level of service for VoIP by itself [56]. VoIP is IP-based and inherits all IP's QoS flaws [37]. Thus, QoS protocols such as RSVP, MPLS and DiffServ are needed in order to provide user acceptable voice services on IP-based packet switched networks. Contemporary NEs support these QoS protocols and can thus provide the QoS needed for VoIP. Legacy NEs may not have the necessary QoS protocols to adequately support VoIP. Legacy NEs, however, have the same abilities to push packets through their ports and can thus provide user acceptable levels of QoS for VoIP under non-congestive conditions. However, when the network becomes congested or moderately busy, QoS provision becomes unpredictable.

The inability to differentiate VoIP packets from data packets is a contributing factor to legacy NEs inability to provide sound QoS during periods of network congestion. Both types of packets are received on an ingress port and need to be forwarded out an egress port. Switching the data packet before the VoIP packet results in a degradation in QoS experienced due to the additional latency. Queuing mechanisms as defined in section 2.2.4.6 provide packet identification and prioritisation functionality and can thus provide the QoS needed for VoIP for legacy NEs [7].

The physical capabilities of legacy NEs are at a disadvantage compared to the circuit-switched networks' innate ability to provide QoS. There is nothing that can be done to change the physical architecture of the legacy NEs so that QoS can be provided unless the physical components are upgraded. If this option were to be chosen, all the NEs within the network would have to undergo this process for interoperability, incurring huge costs. Thus, it is necessary to purchase voice-enabled equipment in order to provide voice services on shared Ethernet environments. However, legacy NEs predominate and not all organisations can afford to jump on the bandwagon and purchase voice enabled NEs.

As stated in Section 2.2, Ferguson & Huston [22] define QoS as "providing a consistent and predictable data delivery service". Furthermore, they state that "any QoS assurances are only as good as the weakest link in the path between sender and receiver." Legacy NEs constitute as a weak link and limit the QoS provision (refer to Figure 3.1). So as not to rule out the possible QoS contemporary NEs can provide, the legacy NEs will be isolated from the contemporary NEs thus creating one logically weak link instead of many individual ones (refer to Figure 3.2).

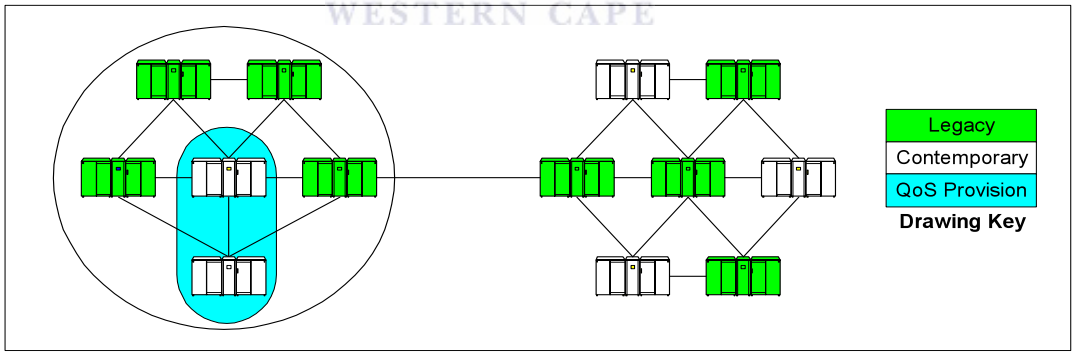


Figure 3.1: Mixed NE Environment

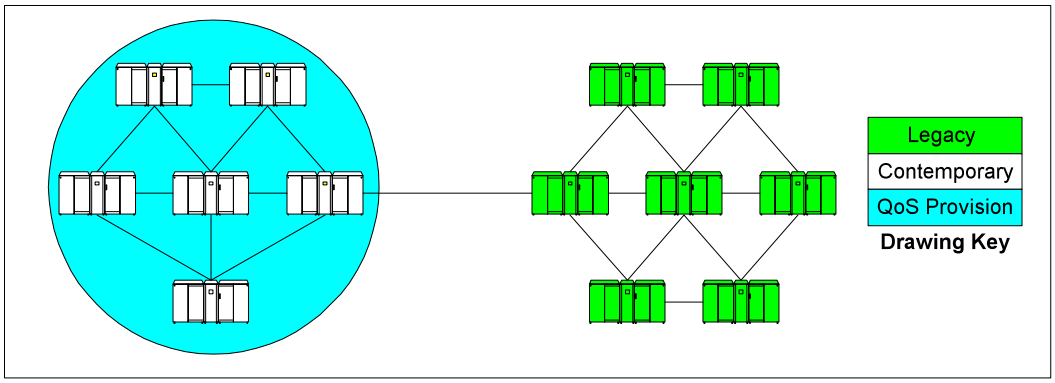


Figure 3.2: Physically Separated Contemporary and Legacy Networks.

Legacy NEs not only lack QoS functionality but also lack the protocols and mechanisms necessary for PBNM systems such as COPS which is used to manage QoS [17]. Essentially, a COPS PEP can be thought of as a software application that makes requests to the PDP on behalf of IP services. Based on the PDP decision, the PEP either enacts the request or does not. Since the PEP is a software application, it is possible to develop a PEP with programming tools to simulate a contemporary PEP that conforms to the standards specified in RFC 2748. In doing this, the possibility exists that legacy NEs can participate within a PBNM paradigm.

The only network management commonality between legacy and contemporary NEs is SNMP [48]. Comparing SNMP to COPS (refer to Table 3.1) it seems as if it is possible to emulate the COPS PBNM paradigm with SNMP using SNMP Sets and Traps .

Table 3.1: COPS versus SNMP

Criteria	**COPS**	**SNMP**
Connection	Reliable, TCP	Non-reliable, UDP
Session initiator	PEP router	SNMP Server
Protocol State	Stateful, no need for polling	Stateless, need constant polling
Multiple controlling servers	Not possible or permissible	Possible
Resource lock	Possible	None
State updates	Asynchronous bi-directional, transactional	SNMP sets & traps
Data model and representation	Policy Info Base (PIB).	MIB

Legacy NEs inability to provide QoS for voice services under congestive and saturated conditions have brought people to the opinion that voice services cannot be deployed on legacy filled NEs. This study aims to disprove this common held opinion: so it closes the gap between QoS enabled contemporary NEs and legacy NEs.

3.2 Research Hypothesis

Using the ideas gained, mechanisms learned from the literature review and taking into account the problems associated with legacy NEs, this research hypothesis is that legacy NEs can support voice services adequately through the use of either SNMP for admission control in a PBNM paradigm or COPS together with traffic shaping in the form of CBQ.

3.3 Research Questions

The research questions used to prove whether or not our hypothesis holds true are as follows:

1. Is it possible for legacy NEs to operate in a PBNM environment?

This question investigates whether or not it is possible for the legacy NEs to obtain COPS directives from the PDP.

2. Is it possible for SNMP to emulate a PBNM environment?

Here we investigate the possibility of the SNMP protocol simulating the COPS protocol by using the SNMP commands, Set and Trap to emulate the COPS REQ and DEC process and thereby emulating a PBNM paradigm.

3. Can SNMP be used to provide QoS when used to provide admission control?

This question asks whether or not it is possible to use SNMP to control the flow of network traffic, can SNMP provide a firewalling service by allowing only predetermined IP services to transit the network. This capability would allow the amount of network traffic

load to be controlled which has a direct relation on the QoS experienced as network saturation results in a higher possibility of QoS degradation and vice versa. Thus the possibility exists, that by not allowing the network to become saturated, better QoS should be experienced for longer periods of time.

4. **Does traffic shaping in the form of Class Based Queuing (CBQ) queuing provide some level of QoS within a PBNM paradigm during periods of network congestion?**

The effects of applying CBQ to IP Services, specifically voice services, is studied under congestive network periods to verify whether or not better QoS can be provided under congestive periods than when CBQ is not applied. "within a PBNM paradigm" refers to configuring the CBQ based on the COPS directives.

5. **What are the affects of employing traffic shaping on legacy IP NEs?**

The findings of a brief study of the effects on other IP services - FTP, SNMP, Telnet and HTTP - of employing traffic shaping for voice services are discussed.

6. **How do the legacy NEs affect the QoS provided by contemporary NEs?**

As stated in Section 1.5, contemporary NEs have the ability to provide QoS while legacy NEs do not. Thus, legacy and contemporary NEs are not interoperable with respect to QoS. This question asks whether or not the legacy NEs hinders the QoS delivered by the contemporary NEs due to this interoperability problem.

3.4 Summary of Chapter 3

The problems with legacy NEs with respect to the QoS provision has prohibited the deployment of voice services. This study hypothesises that it is possible to provide user acceptable QoS for voice services. The answers to the research questions will identify whether the hypothesis holds true or not.

CHAPTER 4

RESEARCH METHODOLOGY AND APPROACH

This chapter focuses on the High Level Design (HLD) of the systems developed and the research and testing methodology used to approach the research problem. The essence of the HLD is to provide an abstraction of the system to be designed.

4.1 Research Methodology

This study takes the approach of what Comte called a positivism approach in Cratty [15]. Comte defines positivism as a position that holds that the goal of knowledge is simply to describe the phenomena that we observe. He continues by saying that the purpose of science is simply to adhere to what we can observe and measure. Knowledge of anything beyond that, a positivist would hold, is impossible.

4.2 Research Approach

During non-congestive network periods, legacy NEs can provide user acceptable QoS for voice services. However, if the network becomes moderately busy, the possibility exists that the QoS experienced by IP services can degrade, which is not acceptable for voice services. The goal of this research was to provide QoS for voice services on networks comprised of legacy and contemporary NEs within a PBNM paradigm.

PBNM is used to dynamically and universally manage the QoS within a network. Legacy NEs lack the QoS functionality, protocols and mechanisms necessary for PBNM. However, Contemporary NEs, do not. By separating the networks into QoS providing and legacy networks, we are able to control and manage the networks independently. This enables us to propagate the identical QoS-based network policies across both networks

resulting in homogonous network management using heterogeneous network configuration mechanisms for QoS provisioning.

Separating the legacy and contemporary NEs rules out the possibility of the legacy NEs affecting the QoS provided by the contemporary NEs. The legacy NEs, however, do not understand the decisions made and instructions sent by the PDP. Thus, in order to provide universal QoS management, the COPS-based PDP messages need to be translated into a format that the legacy NEs can understand or an external mechanism is needed to provide the QoS functionality for the legacy NEs required by voice services.

The only network management commonality between the legacy and contemporary NEs is SNMP. In designing this solution, we assume that all the legacy NEs at a minimum support SNMPv1. Another option exists using an external prioritisation mechanism to give voice services priority over other services.

4.3 Proposed System Designs

Two solutions are proposed to solve the research problem defined in Section 1.2. The first solution uses SNMP for admission control for the legacy NEs within a PBNM paradigm (P-PEP); the second solution uses traffic shaping in the form of CBQ queuing (Q-PEP) to provide differential traffic treatment for QoS within the PBNM paradigm.

As illustrated in Figure 4.1, deploying the proposed solutions between the contemporary and legacy networks, enables them to interact with the QoS/COPS enabled NEs within a PBNM paradigm and receive COPS directives from a PDP. These COPS messages can then be enforced on the legacy NEs whether it be in the form of SNMP commands or traffic shaping and thus conform to the policies set by the Network Administrator on the PDP.

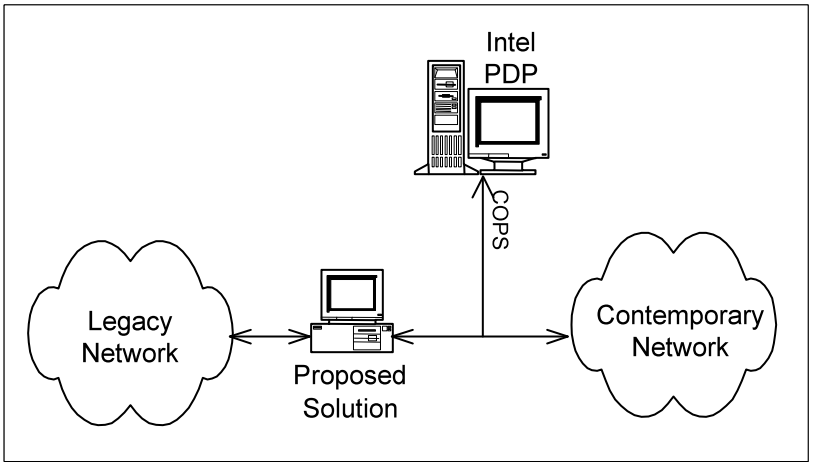


Figure 4.1: Proposed PBNM-based Solution

Tests performed on the system reveal whether or not a single QoS-based network policy can be filtered through to contemporary-based and legacy-based networks so that QoS can be provided.

4.4 Summary of Chapter 4

This study conforms to the positivistic approach. Two solutions, the Q-PEP and P-PEP, were proposed and briefly described to solve the legacy NE QoS provision problem.

CHAPTER 5

PROXY-PEP: SNMP INTERFACE FOR COPS

5.1 Proxy-PEP Overview

The P-PEP was designed as a translation gateway, i.e. the COPS messages are translated into SNMP Set commands either denying or allowing traffic to transit the legacy NEs (refer to Figure 5.1). As indicated in the literature review, PBNM is used to manage the QoS for IP services. Since COPS directives are translated into SNMP Set commands, the legacy NEs will conform to the PBNM policies. This is, however, limited to admission control functionality, i.e. bandwidth cannot be reserved or priority cannot be given for IP services, but IP services can be denied or allowed to transit the legacy NE based on the PBNM policies defined. Admission control allows for the control of the network traffic load as stated in Section 2.2.4.1. Thus, network saturation and congestion can be avoided by controlling the network traffic load. Since the occurrence of network congestion and saturation will be avoided, QoS for voice services can be provided as legacy NEs have the ability to provide user acceptable voice services under non-congestive periods.

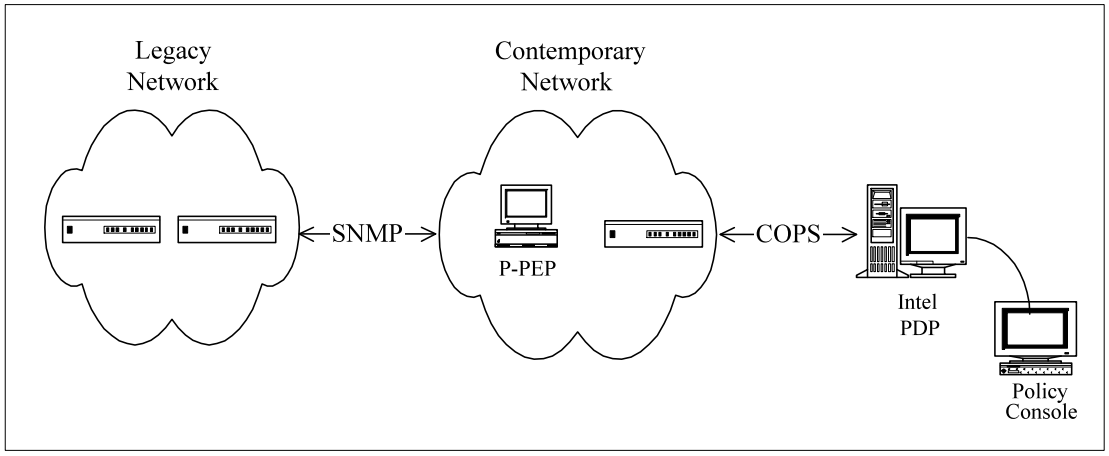


Figure 5.1: Envisaged Deployment of the P-PEP Solution.

5.2 User Interface Specification

The user interface takes the form of a command line interface (CLI) with two user specifiable arguments.

P-PEP Broadcast-IP PDP-IP

The **Broadcast-IP** argument specifies the broadcast IP address. The P-PEP performs a SNMP discovery using the **Broadcast-IP** address to locate legacy NEs that should be managed by the PBNM system. The **PDP-IP** specifies the IP address of the PDP[2] of the PBNM system. The P-PEP requires three configuration files which reside in the application's home directory, the user defined *trafficTypeTrapList* (Service List) file, COPS *config* file, and creates the *snmpEnabledNEs* file.

5.2.1 *trafficTypeTrapList Configuration File*

The *trafficTypeTrapList* file contains the list of IP services in the form of application port numbers. SNMP Traps are set on the legacy NEs defined in the *snmpEnabledNEs* configuration file. Thus, when a legacy NE detects these IP services, it signals the P-PEP using SNMP Traps for further instructions on whether or not to allow the IP service traffic to transit the network. A screenshot of the *trafficTypeTrapList* file can be seen in Figure 5.2.

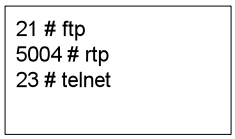


Figure 5.2: *trafficTypeTrapList* file screenshot

5.2.2 *config Configuration File*

The *config* configuration file must follow the format specified below and include all the listed parameters,

---

[2] The PDP forms part of the Intel COPS Client Software Development Kit that can be freely downloaded at http://www.intel.com/labs/manage/cops/download.htm

N PEPID

C Client-Type

The PEPID is the IP address of the P-PEP and the Client-Type specifies the type of COPS client, in this case it is 1, a value defined as a RSVP Client-Type [17]. The PEPID is used by the PDP to uniquely identify the P-PEP while the Client-Type informs the PDP of the P-PEPs' capabilities. A screenshot of the *config* file can be seen in Figure 5.3.

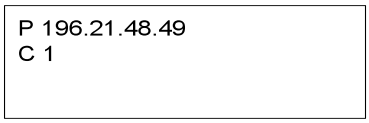


Figure 5.3: *config* file screenshot.

5.2.3 *snmpEnabledNEs Configuration File*

The *snmpEnabledNEs* configuration file contains a list of SNMP enabled NEs by IP address. The file is created from the SNMP replies received from the SNMP system description broadcast discovery sent. These legacy NEs are the NEs which will be managed by the PBNM system. A screenshot of the *snmpEnabledNEs* file can be seen in Figure 5.4.

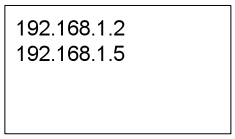


Figure 5.4: snmpEnabledNEs file example.

5.3 High Level Design

The P-PEP can be broken into two logical processes, the SNMP Manager and the COPS PEP signalling process as can be seen in Figure 5.5.

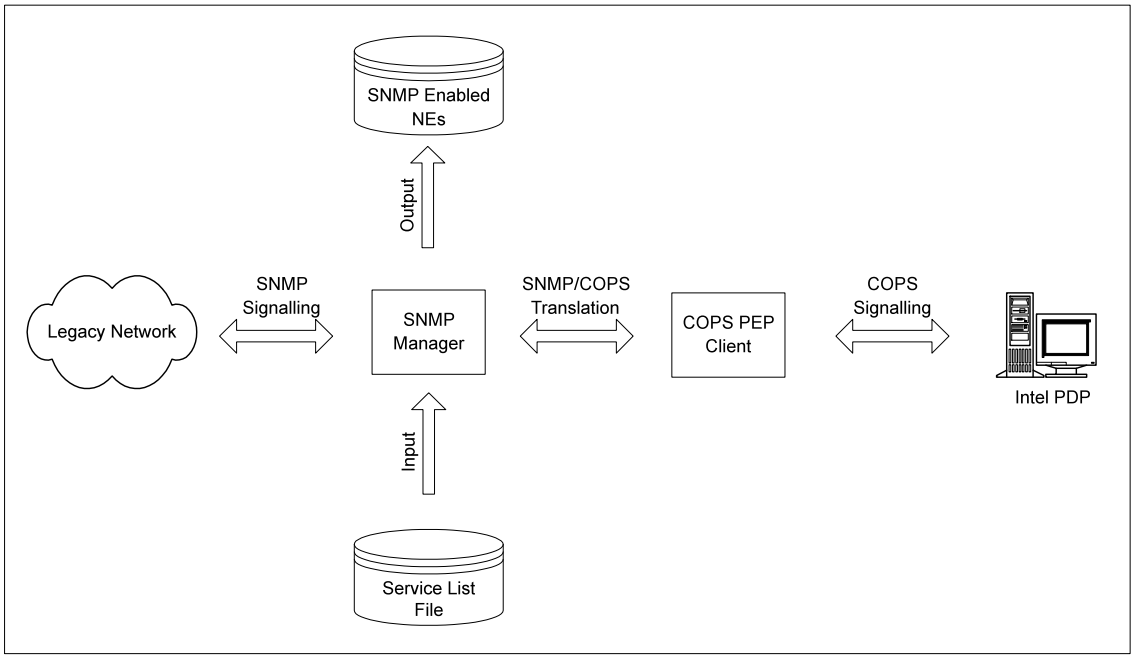


Figure 5.5: P-PEP Process Illustration.

5.3.1 *SNMP Manager Process*

The SNMP Manager process starts with the P-PEP sending a SNMP system description broadcast to the legacy network. Each SNMP enabled device responds with its system description which contains the NEs system name and IP address. Each NE's IP address is stored within the *snmpEnabledNEs* file. The SNMP Traps for the various IP services and applications specified in the Service List file, *trafficTypeTrapList*, are then conFigured on all the SNMP enabled devices specified in the *snmpEnabledNEs* file using the SNMP Set command. These SNMP Traps will signal the SNMP Manager when a specific traffic type is detected on the ports of the legacy NEs. The functions used for the SNMP Manager are defined in Table 5.1. The COPS signalling process then starts as depicted in Figure 5.6.

Table 5.1: Functions used by the SNMP Manager

Function Name	Description
SNMPGet	This function retrieves a specified MIB Object Identifier (OID) from a NE.
SNMPSet	This function changes a specified MID OID value.
SNMPLegacyNEDiscovery	This function retrieves all the SNMP enabled devices on a network and saves the output to a file, *snmpEnabledNEs*

5.3.2 *COPS PEP Signalling Process*

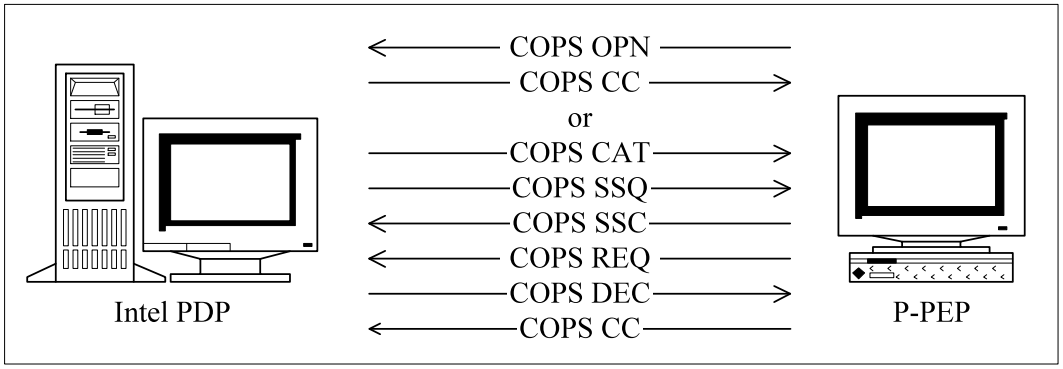


Figure 5.6: COPS PDP P-PEP message exchange.

Before any COPS messages are exchanged between the PDP and P-PEP, the P-PEP has to establish a TCP connection on the well known COPS port, 3288. On this connection, the following COPS messages are exchanged in the order depicted in Figure 5.6. The communication process is initiated by the P-PEP, by sending a COPS Client Open (OPN) message which contains the PEP Identifier (PEPID) and Client-Type. The PEPID and Client-Type is specified in the COPS configuration file, *config*. If the Client-Open message is malformed, the PDP generates a Client-Close (CC) message specifying the appropriate error as defined in RFC 2748. If the COPS OPN message is not malformed, the PDP sends a Client Accept (CAT) message which is used to positively respond to the COPS OPN message [17].

The PDP then sends a COPS Synchronise State Request (SSQ) message as it wishes the client to re-send its state. To this the P-PEP sends a Synchronise State Complete (SCC) message which is always null due to the state being null at activation. The P-PEP then sends COPS Request (REQ) messages to the PDP. On receipt, the PDP sends a COPS Decision (DEC) message based on the policies defined. Each SNMP Trap is triggered by a particular type of network traffic on the legacy NEs and results in one COPS REQ message being sent from the P-PEP to the PDP. Only one COPS REQ can be sent at a time. Another request cannot be sent until a COPS DEC is received from the PDP for that particular COPS REQ. Denial of a COPS REQ results in the SNMP Manager terminating the corresponding IP service connection on the legacy NE port specified in

the COPS REQ. This is done through the SNMP Set command. If the COPS REQ is approved, the IP service connection on the legacy NE port is not terminated and thus results in the IP services' traffic to transit the NE. This process continues until the P-PEP is deactivated by the PDP itself by sending a COPS CC message. The functions used to establish the COPS TCP connection and create the COPS messages are defined in table 5.2.

Table 5.2: Functions used by the P-PEP COPS Signalling Simulator

Function Name	Description
ClientConnect	This function is used to create the socket connection for communication between the PDP and Q-PEP.
COPSSendTo	This function sends a specified packet to a connected PDP according to the OpCode specified.
MakeCOPSOPNPacket	This function is used to create the COPS OPN packet.
MakeCOPSSCPacket	This function is used to create the COPS SSC packet.
MakeCOPSREQPacket	This function is used to create the COPS REQ packet.
ReadFrom	This function interprets messages sent by the connected PDP on a predefined socket.

5.4 Low Level Design

Chapin diagrams are used to illustrate the algorithms of the P-PEP system. We start off by illustrating the controlling function of the program, *main*. The system functions will be described under each logical process, SNMP Manager and COPS PEP Signalling, and in order of their occurrence in the *main* function which is *SNMPLegacyNEDiscovery, SNMPGet, clientConnect, MakeCOPSOPNPacket, COPSSendTo, MakeCOPSSCPacket, MakeCOPSREQPacket, ReadFrom* and *SNMPset*.

5.4.1 *Main Function*

Description: Controlling function of the P-PEP system.

Algorithm:

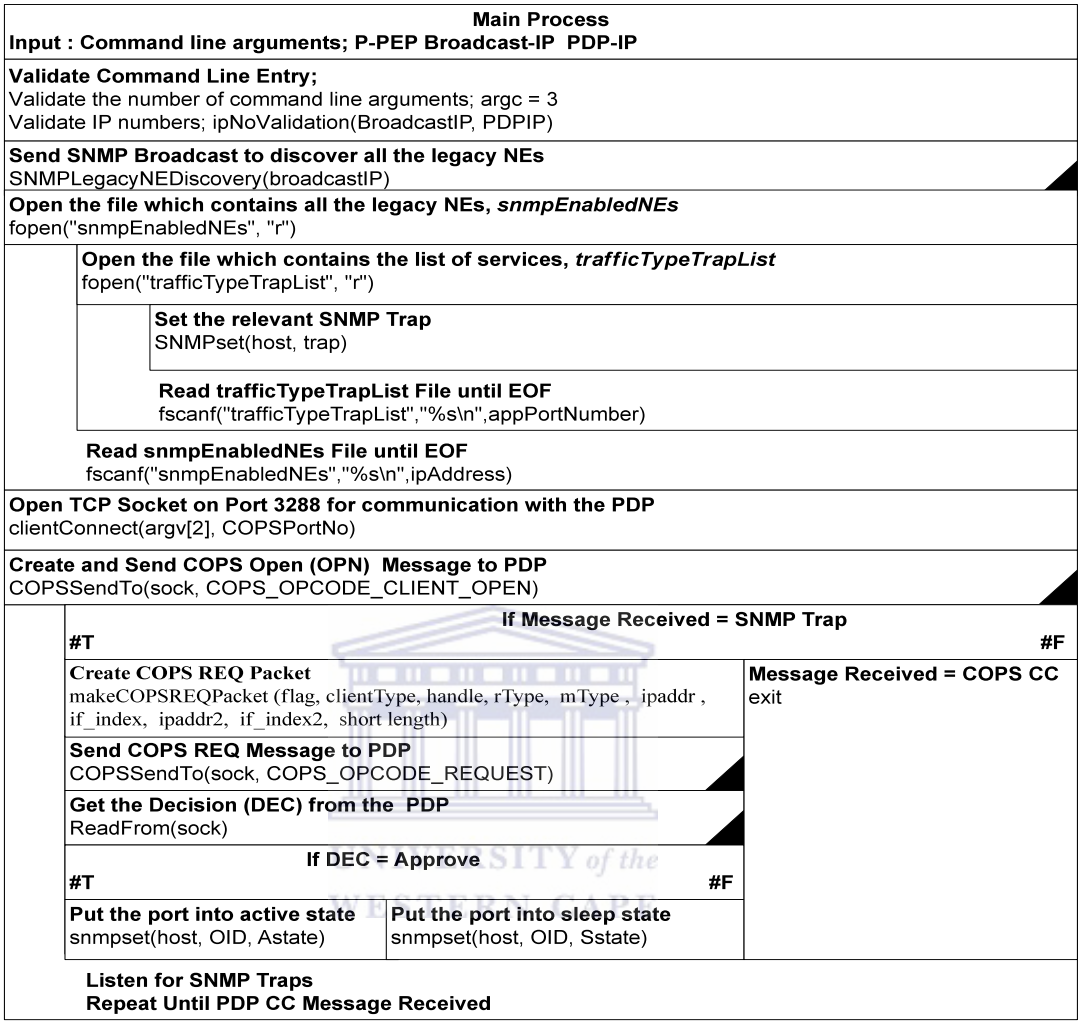


Figure 5.7: *main* Chapin Diagram.

Inputs:

Broadcast-IP: specifies the broadcast domain where the SNMP discovery will be performed.
PDP-IP: specifies the IP address of the PDP of the PBNM system.

Outputs: None.

5.4.2 *SNMP Manager Process*

5.4.2.1 SNMPLegacyNEDiscovery Function

Description: This function retrieves all the SNMP enabled devices on a network and saves the output to a file, *snmpEnabledNEs*

Algorithm:

SNMPLegacyNEDiscovery Function
Input : IP address range that snmp discovery will be performed on (**broadcastIP**),
Send the SNMP Broadcast for the NEs systemDescriptions snmpGet(broadcastIP, systemDescription)
Open UDP Port to listen for SNMP Traps snmpOpenTrapSession(port, timeOut)
Open file to store SNMP Trap Information, snmpEnabledNEs fopen(snmpEnabledNEs, "w")
Write SNMP Trap information to file, snmpEnabledNEs fprint(snmpEnabledNEs, NEIPAddress)
Repeat until Timeout

Figure 5.8: *SNMPLegacyNEDiscovery* Chapin Diagram.

Inputs:

broadcastIP : specifies the broadcast IP address to send the SNMP query to.

Outputs: None.

5.4.2.2 SNMPGet Function

Description: This function retrieves a specified MIB Object Identifier (OID) from a NE.

Algorithm:

SNMPGet Function
Input : Broadcast IP address **(Host)** SNMP OID **(OID)**
Open SNMP Session on port 161 SNMPSession=SNMP_Session->open (Host, 161)
Send SNMP Request on SNMP Session SNMPRequest(SNMPSession, systemDescription)

Figure 5.9: *SNMPGet* Chapin Diagram.

Inputs:

Host: specifies the IP address of the destination NE.
OID: specifies the SNMP MIB instance.

Outputs: None.

5.4.2.3 SNMPset Function

Description: This function changes a specified MID OID value.

Algorithm:

SNMPSet Function
Input : Broadcast IP address **(Host)** SNMP OID (**OID)** New SNMP OID value **(value)**
Open SNMP Session on port 161 SNMPSession=SNMP_Session->open (Host, 161)
Send SNMP Set on SNMP Session SNMPSet(SNMPSession, systemDescription, value)

Figure 5.10: *SNMPSet* Chapin Diagram.

Inputs: None

Host: specifies the IP address of the destination NE.
systemDescription: specifies the SNMP MIB instance.
value: specifies the value to be changed to.

Outputs: None.

5.4.3 *COPS PEP Signalling Process*

5.4.3.1 clientConnect Function

Description: This function is used to create the socket connection for communication between the PDP and Q-PEP.

Algorithm:

clientConnect Function
Input : PDP ip addres **(PDP_IP_addr)** Port to connect on **(port)**
Open TCP socket sockfd = socket(AF_INET, SOCK_STREAM, 0)
Connect to PDP Server connect (sockfd, (struct sockaddr *) &their_addr, sizeof (struct sockaddr))
Return socket file descriptor return sockfd

Figure 5.11: *clientConnect* Chapin Diagram.

Inputs:

PDP_IP_addr: IP address of the PDP.
Port: Port to connect to the PDP (3288).

Outputs: Sockfd: socket file descriptor used to communicate on.

5.4.3.2 MakeCOPSOPNPacket Function

Description: This function is used to create the COPS OPN packet.

Algorithm:

makeCOPSOPNPacket Function
Input : OPN Flag Identification **(flag)** COPS OpCode **(OpCode)** COPS Client-Type **(clientType)**
Insert COPS version and flag into the packet COPSOPNPacket.verFlag = setHeaderVerFlag(1,flag)
Insert COPS Opcode into the packet COPSOPNPacket.OpCode = OpCode
Insert COPS Client-Type into the packet COPSOPNPacket.clientType = clientType
Create COPS PEPID Object makePEPID(PEPID)

Figure 5.12: *makeCOPSOPNPacket* Chapin Diagram.

Inputs:

flag: specifies the COPS flag.
OpCode: uniquely identifies OPN message.
clientType: specifies COPS Client-Type.
PEPID: uniquely identifies the PEP.

Outputs: None.

5.4.3.3 COPSSendTo Function

Description: This function sends a specified packet to a connected PDP according to the OpCode specified

Algorithm:

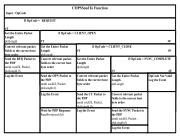


Figure 5.13: *COPSSendTo* Chapin Diagram.

Inputs:

OpCode: specifies the type of COPS Packet.

Outputs: None.

5.4.3.4 MakeCOPSSCPacket Function

Description: This function is used to create the COPS SSC packet.

Algorithm:

makeCOPSSCPacket Function
Input : OPN Flag Identification **(flag)** COPS OpCode **(OpCode)** COPS Client-Type **(clientType)**
Insert COPS version and flag into the packet COPSSCPacket.verFlag = setHeaderVerFlag(1,flag)
Insert COPS Opcode into the packet COPSSCPacket.OpCode = OpCode
Insert COPS Client-Type into the packet COPSSCPacket.clientType = clientType;

Figure 5.14: *MakeCOPSSCPacket* Chapin Diagram.

Inputs:

flag: specifies the COPS flag.
OpCode : uniquely identifies SCC message
clientType: specifies COPS Client-Type

Outputs: None.

5.4.3.5 MakeCOPSREQPacket Function

Description: This function is used to create the COPS REQ packet.

Algorithm:

makeCOPSREQPacket Function
Input : OPN Flag Identification **(flag)** COPS OpCode **(OpCode)** COPS Client-Type **(ClientType)** Pep message identifier **(Handle)** Type of REQ **(Rtype)** Client Specific message type **(MType)** Incomming interface IP address **(Ipaddr)** Incomming interface identifier **(IfIndex)** Length of the entire packet **(Length)** Application or IP service port number that triggered the SNMP Trap **(ServicePortNo)**
Insert COPS version and flag into the packet COPSREQPacket.verFlag = setHeaderVerFlag(1,flag)
Insert COPS Opcode into the packet COPSREQPacket.OpCode = OpCode
Insert COPS Client-Type into the packet COPSREQPacket.clientType = clientType;
Create COPS REQ Packet Unique Handle makeCOPSObjHandle(handle)
Create COPS REQ Context Object makeCOPSObjContext(rType, mType)
Create COPS REQ In-Interface Object makeCOPSObjInInterface(ipaddr, ifIndex)
Create COPS REQ Out-Interface Object makeCOPSObjOutInterface(ipaddr2, ifIndex2)
Create COPS REQ Client Specific Object makeCOPSObjClientSI()
Calculate COPS REQ packet length COPSREQPacket.length = sizeof(COPSHeader) + COPSREQPacket.HandleObLength + COPSREQPacket.ContextObLength + COPSREQPacket.InInterfaceObLength + COPSREQPacket.OutInterfaceObLength + COPSREQPacket.ClientSIObLength;

Figure 5.15: *MakeCOPSREQPacket* Chapin Diagram.

Inputs:

OpCode: uniquely identifies REQ message.
ClientType: specifies COPS Client-Type.
Handle: unique Pep message identifier.
Rtype: Identifies the type of REQ.
MType: Identifies Client Specific message type.
Ipaddr: specifies the incomming interface IP address.
IfIndex : specifies the incomming interface identifier.
Length: specifies the length of the entire packet.
ServicePortNo: specifies the application or IP service port number that triggered the SNMP Trap.

Outputs: None.

5.4.3.6 ReadFrom Function

Description: Interprets messages sent by the connected PDP on a predefined socket.

Algorithm:

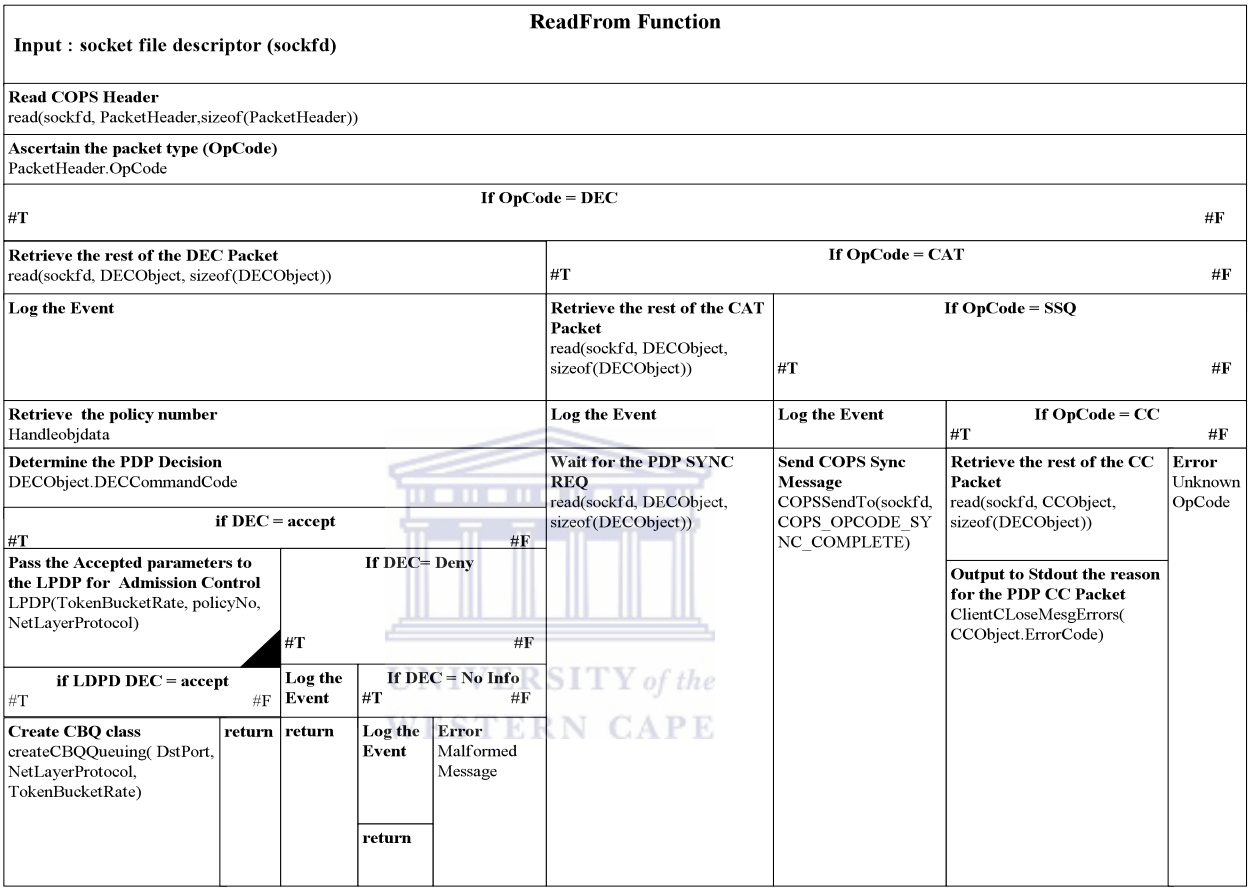


Figure 5.16: *ReadFrom* Chapin Diagram.

Inputs:

Sockfd: socket file discriptor.

Outputs: None.

5.5 Summary of Chapter 5

An overview of the functionality of the P-PEP was given. The UIS defined what the user interface appears as while the HLD provided an abstraction to the system to be designed. The LLD defined a detailed view of the system to be developed using Chapin Diagrams

CHAPTER 6

QUEUING-PEP

6.1 Overview

The Q-PEP was designed within the Diffserv framework, so that the legacy NEs need not communicate with the contemporary NEs for QoS provision but only with the PDP for COPS directives (refer to Figure 6.1). This does not rule out the contemporary NEs operating within the Intserv framework and using protocols such as RSVP to provide QoS.

The Q-PEP sends COPS requests to the PDP based on the information stored in the *rsvpInput* (refer to Figure 6.2) configuration file. The PDP in turn returns COPS decisions based on the PBNM policies. A CBQ class is created for each COPS request that is approved in the *cbq* file (refer to Figure 6.5). Once decisions for all the COPS requests have been retrieved, the ALTQ CBQ daemon is initialised based on the *cbq* file. As stated in Section 2.2.4.6, CBQ mechanisms have the ability to prioritise IP services and can therefore provide QoS for voice services under congestive and non-congestive periods.

6.2 Research Instruments

A queuing mechanism, ALTQ[3], was used as part of the proposed Q-PEP solution. ALTQ is designed to support a variety of queuing disciplines with different components which include scheduling, packet drop and buffer allocation strategies. The ALTQ framework is designed to support both research and real life operations. The framework was designed for the FreeBSD kernel and is thus limited to that operating system. ALTQ implements queuing disciplines including CBQ, WFQ and the congestion algorithm RED [9]. The

[3] ALTQ is a free software application that can be downloaded at http://www.csl.sony.co.jp/person/kjc/programs.html

CBQ queuing mechanism was chosen over WFQ due to WFQ's inability to prioritise network traffic based on traffic types (refer to Section 2.2.4.6). WFQ is designed to

prevent any one traffic type from overshadowing another [21], which is aim of this research regarding voice services. CBQ has the ability to do this and is thus the chosen queuing mechanism.

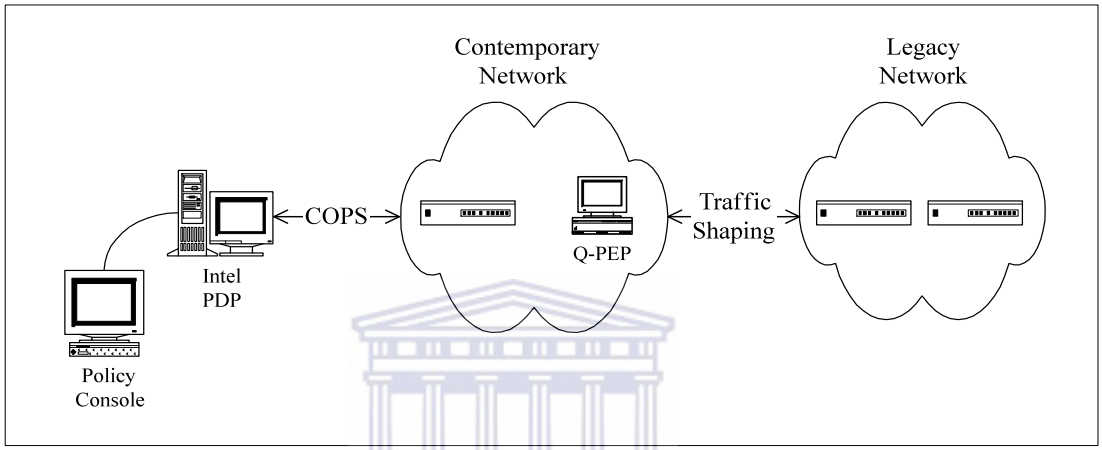


Figure 6.1: Envisaged Deployment of the Q-PEP Solution.

6.3 User Interface Specification

The user interface takes the form of a CLI with one user specifiable argument, the IP address of the PDP, PDP-IP.

Q-PEP PDP-IP

The **PDP-IP** specifies the IP address of the PDP of the PBNM system. The Q-PEP requires 3 configuration files and creates two output files which resides in the application's home directory. The configuration files are: *config* file, as defined in Section 5.2.2 which specifies the COPS properties; RSVP input file, *rsvpInput*, which specifies the parameters needed to create the RSVP packets; and the queuing configuration file, *queuing*, which specifies the parameters needed to set up the CBQ queues.

The Output files created are the log file, *log*, which records the events of the Q-PEP and the CBQ configuration file, *cbq*, which consists of ALTQ commands that specifies the parameters necessary for the ALTQ CBQ daemon.

6.3.1 *Configuration Files*

The required format and purpose of the configuration files are as follows.

6.3.1.1 rsvpInput Configuration File

The *rsvpInput* file should include the following parameters,

R Policy-number Transport-protocol Destination-protocol Output-Interface-IP Bandwidth-requested-as-percentage

The Policy-number is a random positive number which is used to identify a specific policy and used as the COPS Client Handle; it is used to identify a unique request state for a single PEP per Client-Type. The Transport-protocol is either UDP or TCP, while the Destination-protocol specifies the application specific protocol requesting the QoS. The Output-Interface-IP specifies the interface IP address from which the COPS REQ was sent and the Bandwidth-requested-as-percentage, specifies the requested amount of bandwidth. The total amount of bandwidth may not exceed 100 percent, however, it need not total 100 percent. These lines can appear numerous times in the *rsvpInput* file as illustrated in Figure 6.2. The parameters stored in this file is used for the creation of RSVP Packets.

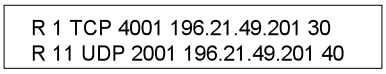


Figure 6.2: *rsvpInput* screenshot.

6.3.1.2 queuing Configuration File

The queuing file must include the following parameters,

I Interface-Name

B Interface-bandwidth

T TCP-bandwidth-allocation-as-percentage

U UDP-bandwidth-allocation-as-percentage

The Interface-Name specifies the string name allocated to a Network Interface Card (NIC) on which the traffic shaping will occur. The Interface-bandwidth specifies the maximum transmission speed of the NIC while the TCP-bandwidth-allocation-as-percentage and UDP-bandwidth-allocation-as-percentage specifies the amount of bandwidth for the two main CBQ queues, TCP and UDP. The TCP-bandwidth-allocation-as-percentage and UDP-bandwidth-allocation-as-percentage bandwidth allocation must not total more than 95 percent, as the remaining 5 percent is used for signalling protocols such as Internet Control Message Protocol (ICMP). An example of this configuration file is illustrated in Figure 6.3. The parameters Interface-bandwidth, TCP-bandwidth-allocation-as-percentage and UDP-bandwidth-allocation-as-percentage stored in the *queuing* file is used by the LPDP to make local decisions for the Q-PEP based on available resources. If any of the Input files do not conform to the specified formats the Q-PEP will terminate with an error, specifying which file and what error is that has occurred.

Figure 6.3: *queuing* file screenshot.

6.3.2 *Output Files*

6.3.2.1 log Output File

An example of the events recorded by the Q-PEP stored in the *log* file is illustrated in Figure 6.4.

```
COPS OPN Message Sent To PDP at time: Wed Oct 31 18:42:07 2001
CAT Message Received from PDP at time: Wed Oct 31 18:42:07 2001
PDP Requires Synchronisation information at time: Wed Oct 31 18:42:07 2001
COPS SYNC Message Sent To PDP at time: Wed Oct 31 18:42:07 2001
COPS REQ, Policy No 3 Sent To PDP at time: Wed Oct 31 18:42:07 2001
COPS DEC, for Policy No 3 Received from PDP at time: Wed Oct 31 18:42:07 2001
COPS REQ Policy No 3 Accepted
COPS REQ, Policy No 2 Sent To PDP at time: Wed Oct 31 18:42:07 2001
COPS DEC, for Policy No 2 Received from PDP at time: Wed Oct 31 18:42:07 2001
COPS REQ Policy No 2 Accepted
LPDP Decision, Policy REQ 2 declined. Reason: All Available bandwidth Allocated
COPS CC Message Sent To PDP at time: Wed Oct 31 18:42:08 2001
```

Figure 6.4: *log* file screenshot.

6.3.2.2 cbq Output File

An example of the *cbq* file created by the Q-PEP is illustrated in Figure 6.5. This configuration example is depicted visually in Figure 6.6 illustrating the child parent relationship and the amount of bandwidth allocated to each class/queue. For example, the "voice" class is a child of the "UDP" class. The "UDP" class in turn is a child class of the "Default" class and a parent class for the "voice" class (refer to Figure 6.6). The *cbq* file is used by the ALTQ daemon to set up the CBQ.

```
interface fxp1 bandwidth 100M cbq
class cbq fxp1 root NULL pbandwidth 100
class cbq fxp1 ctl_class root pbandwidth 4 control
class cbq fxp1 def_class root borrow pbandwidth 95 default

class cbq fxp1 TCP def_class borrow pbandwidth 75
class cbq fxp1 21 TCP borrow pbandwidth 20
filter fxp1 21 0 0  0 21 6
filter fxp1 21 0 21 0  0 6
class cbq fxp1 20 TCP borrow pbandwidth 25
filter fxp1 20 0  0 0 20 6
filter fxp1 20 0 20 0  0 6
class cbq fxp1 23 TCP borrow pbandwidth 15
filter fxp1 23 0  0 0 23 6
filter fxp1 23 0 23 0  0 6
class cbq fxp1 80 TCP borrow pbandwidth 15
filter fxp1 80 0  0 0 80 6
filter fxp1 80 0 80 0  0 6


class cbq fxp1 UDP def_class borrow pbandwidth 20
class cbq fxp1 161 UDP borrow pbandwidth 10
filter fxp1 161 0   0 0 161 17
filter fxp1 161 0 161 0   0 17
class cbq fxp1 3004 UDP borrow pbandwidth 10
filter fxp1 3004 0    0 0 3004 17
filter fxp1 3004 0 3004 0    0 17
```

Figure 6.5: *cbq* file screenshot.

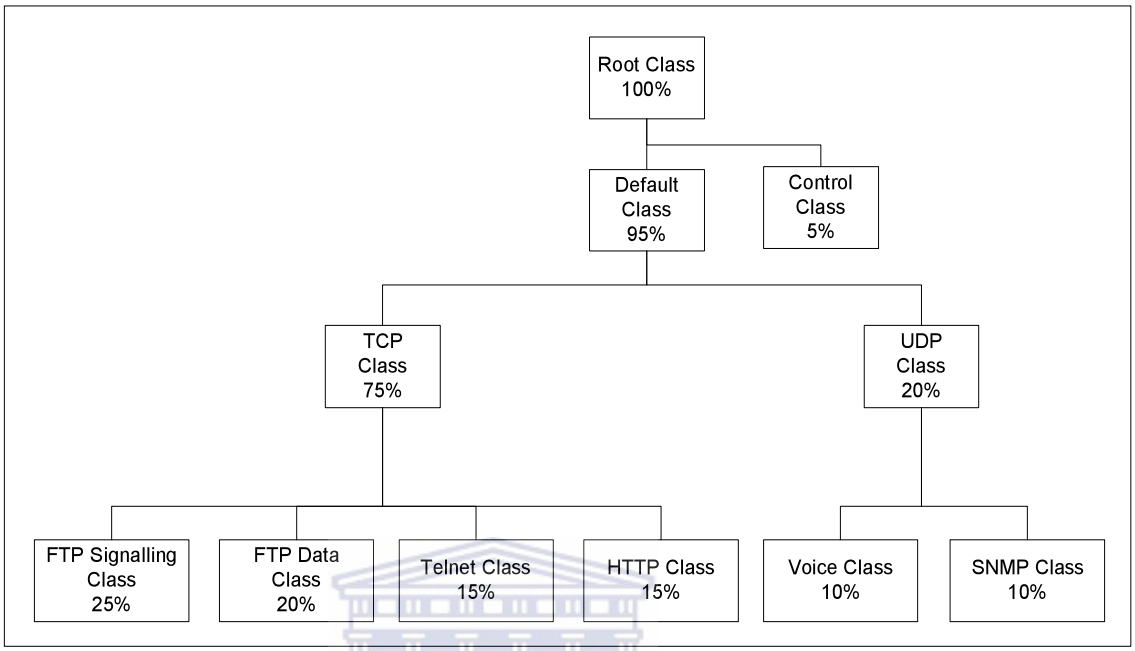


Figure 6.6: Visual child parent relationship of the *cbq* file

The *cbq* file is comprised of ALTQ commands. The available commands are as follows;

- **interface if_name bandwidth cbq**: the **interface** command sets up the interface by name using the **if_name** variable while the **bandwidth** variable specifies the amount of bandwidth available on the specified interface for queuing. The **cbq** variable specifies the type of queuing implemented. An example of the interface command would be **interface fxp1 bandwidth 10M cbq** (refer to Figure 6.5) where "fxp1" specifies the interface name,"10M" specifies the amount of bandwidth available and "cbq" specifies the queuing mechanism type.

- **class cbq if_name class_name parent borrow pbandwidth [default|control]**: the **class** command creates a class on a specified interface using the **if_name** variable. The name of the class is set with the **class_name** variable while **t**he **parent** variable indicates whether or not the class being created is a parent class or a child class. If the **parent** option is set to NULL, it indicates that it is a parent class while any other name

indicates a child class. The **borrow** variable indicates that the child class can borrow bandwidth from its parent class. The **pbandwidth** variable specifies the amount of bandwidth allocated to the class. The **default|control** optional variables specifies whether the class is a **default** or **control** class. Any packet that does not fit into a specific class will be allocated and processed in the **default** class. The **control** variable specifies that the class being defined is a controlling class which is used to send control packets such as ICMP packets. An example of a parent class would be **class cbq fxp1 root NULL pbandwidth 100** (refer to Figure 6.5) where "fxp1" specifies the interface, "root" specifies the class name, "NULL" and "pbandwidth 100" specifies the amount of bandwidth as a percentage available to the class. An example of a child class would be **class cbq fxp1 def_class root borrow pbandwidth 95 default** (refer to Figure 6.5), where "fxp1" specifies the interface, "def_class" specifies the class name and "borrow" indicates that this class can borrow bandwidth from the class "root". The amount is specified with "pbandwidth 95" which indicates that 95 percent of the bandwidth from the parent class "root" can be borrowed.

- **filter if_name class_name dst_addr dst_port src_addr src_port protocol**: the **filter** command sets a packet filter to a specified class using the **class_name** variable. Packets are matched based on the variables set in the filter command: **dst_addr** specifies the destination IP address, **dst_port** specifies the destination port address, **src_addr** specifies the source IP address and the **src_port** specifies the source port address that will be assigned to the specified class, **class_name**. A zero (0) specified in any of the filter command parameters indicates that the field is ignored, i.e. no packet comparison matching for that specified parameter will occur. An example of the filter command would be **filter fxp1 21 0 0 0 21 6** (refer to Figure 6.6), where "fxp1" specifies the interface, "21" specifies the class name, the first "0" specifies the destination IP address, the second "0" specifies the destination port address, the third address specifies the source IP address while the "21" specifies the destination port which would be Telnet and finally the "6" specifies the TCP protocol. Thus all packets destined to a Telnet server will be allocated to the "21" class.

6.4 High Level Design

The Q-PEP can be broken into 4 logical processes (Figure 6.7): the COPS PDP signalling, the creation of RSVP Packets, the LPDP decision process and the CBQ queue file creation. A more detailed discussion of the Q-PEP processes as named above follows.

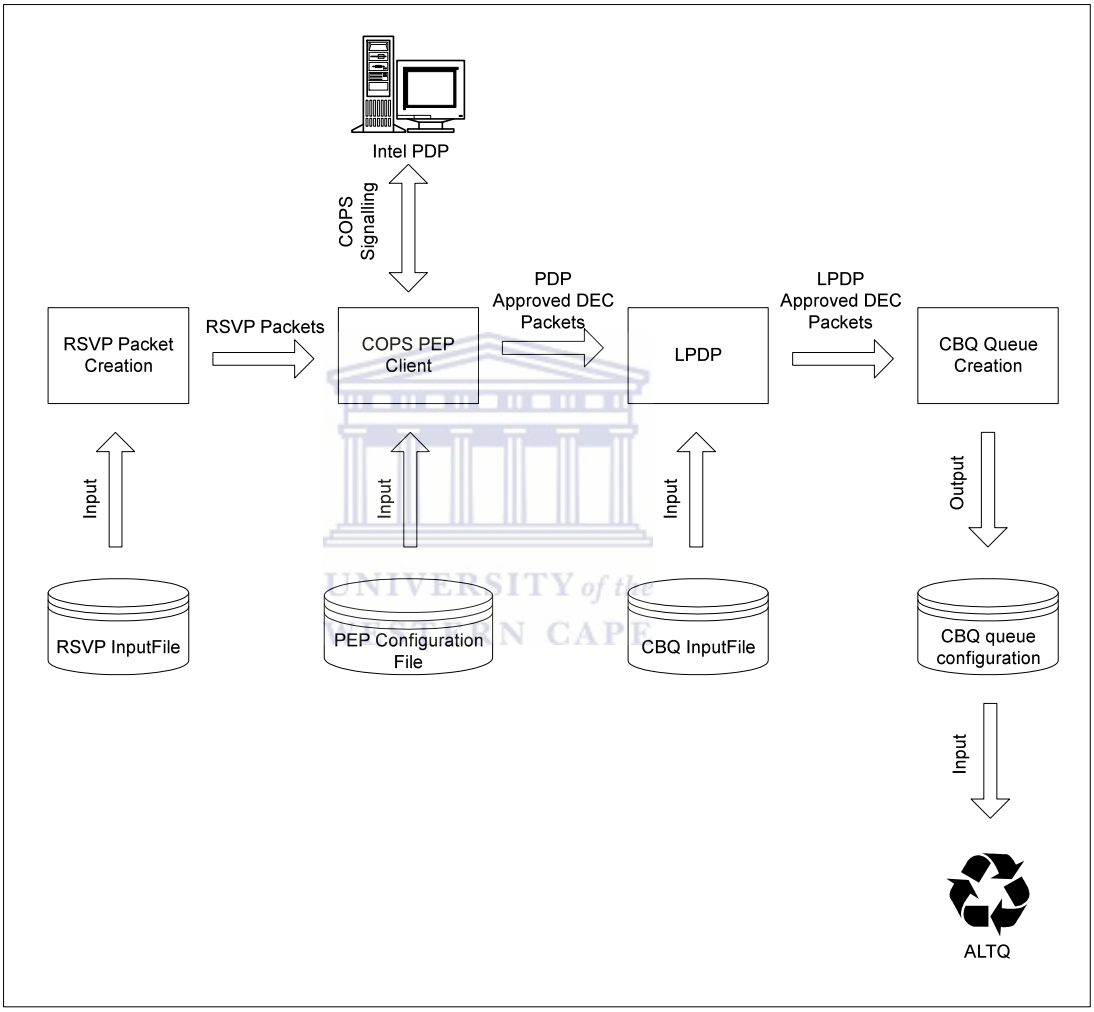


Figure 6.7: Q-PEP Process Illustration.

6.4.1 *COPS Signalling Process*

The COPS signalling process is similar to the COPS signalling process of the P-PEP as described in Section 5.3.2. The only difference occurs with the creation of the COPS REQ

packet and the Q-PEP sending a COPS CC packet to terminate the TCP connection between the Q-PEP and PDP.

6.4.2 *RSVP Packet Creation Process*

The COPS REQ messages (refer to Figure 6.8) created from the information extracted from the RSVP packets that are created from the *rsvpInput* file. For each RSVP Packet, one COPS REQ is created. The Q-PEP sends a COPS CC Packet after all the COPS DEC packets are received to terminate the connection. The Q-PEP remains connected to the PDP until the PDP terminates the connection.

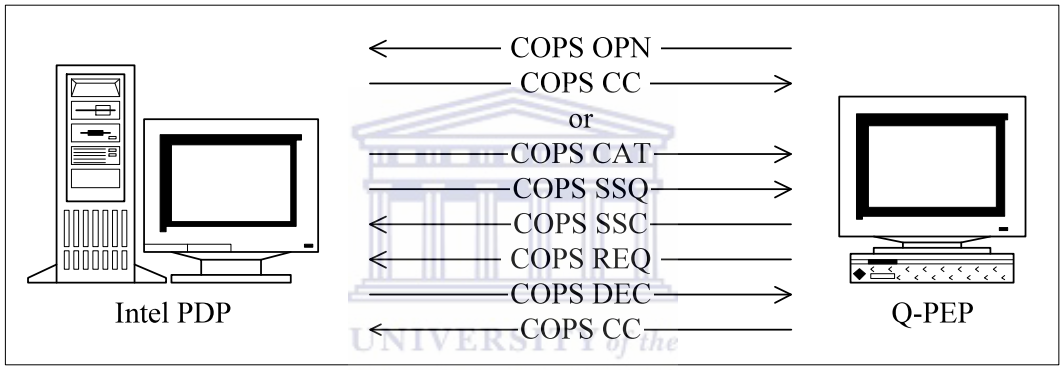


Figure 6.8: COPS PDP Q-PEP message exchange.

The functions used to establish the COPS TCP connection and create the Q-PEP COPS PEP messages are the same as described in Table 5.2. The additional functions used in the Q-PEP are described in the Table 6.1 and the functions used to create the RSVP packets are detailed in Table 6.2.

Table 6.1: Additional Functions used by the Q-PEP Signalling Simulator

Function Name	**Description**
MakeCOPSClientClosePacket	This function is used to create the COPS CC packet

Table 6.2: Function used to create the RSVP Packets

Function Name	**Description**
MakeRSVPResvPacket	This function is used to create the RSVP Packet

6.4.3 *LPDP Decision and CBQ Queue Creation Process*

Each approved COPS REQ is passed to the LPDP. The LPDP makes decisions to approve or deny the COPS REQ for CBQ class creation. The decisions are based on the bandwidth available, i.e. bandwidth not assigned to a CBQ queue, and the user defined constraints of the *queue* file which includes the maximum bandwidth allocated for the TCP and UDP class. If any of the COPS REQ parameters cannot be met, i.e. if the bandwidth requested is larger than the parent class's UDP and TCP, or greater than the bandwidth available, the COPS REQ is denied and a CBQ class is not created. If the LPDP approves the COPS REQ, the CBQ queue class is created and is stored in the *cbq* file. Once all the COPS REQ messages are sent and the relevant COPS DEC messages are retrieved, the queuing process starts by initialising the ALTQ daemon, as specified in the *cbq* file. The functions used in the creation of the *cbq* file are described in the Table 6.3 below.

Table 6.3: Functions used for the creating of the *cbq* file

Function Name	Description
LPDP	This function makes local decisions for the approved COPS REQ messages for the creation of the CBQ queue class entries. These messages are approved or denied based on the constraints set in the *queuing* file.
CreateCBQQueuing	This function is used to create the CBQ configuration file, *cbq*
GetQueueingConfigData	This function is used to retrieve the TCP and UDP bandwidth constraints and NIC name and bandwidth capabilities.

6.5 Low Level Design

Chapin diagrams are used to illustrate the algorithms of the Q-PEP system. We start off by illustrating the controlling function of the program, *main*. The system functions will be described under each logical process, COPS Signalling, RSVP Packet Creation, LPDP Decision and CBQ Queue Creation Process and in order of their occurrence in the *main* function which is *MakeCOPSClientClosePacket, makeRSVPResvPacket, RSVP, LPDP* and *CreateCBQQueuing*.

6.5.1 *Main Function*

Description: Controlling function of the Q-PEP system.

Algorithm:

Main Process **Input : Command line arguments; Q-PEP [PDP-IP]**
Validate Command Line Entry; Validate the number of command line arguments; argc = 2 Validate IP number; ipNoValidation(PDPIP)
Open TCP Socket on Port 3288 for communication with the PDP clientConnect(argv[1], COPSPortNo)
Get COPS Client-Type and PEPID from Input File; (config) getCOPSConfigData()
Get Interface Type, Total Bandwidth, TCP and UDP Bandwidth Allocation parameters from the Input File; (queuing) getQueueingConfigData()
Sort RSVP Packet Information Input File; rsvpInputFile popen(sortCmd, "r")
Setup default CBQ entries and TCP meta class createCBQHeader(TCPPort)
Create and Send COPS Open (OPN) Message to PDP COPSSendTo(sock, COPS_OPCODE_CLIENT_OPEN)
Create RSVP Packets, COPS Request Packets, CBQ queue entries RSVP(sock, clientType)
Create COPS Client Close (CC) Message makeCOPSClientClosePacket(flag, clientType)
Send COPS CC Message to PDP COPSSendTo(sock, COPS_OPCODE_CLIENT_CLOSE)
Start CBQ process popen(altqdCmd, "r")

Figure 6.9: *Main* Chapin Diagram.

Inputs:

PDP-IP: specifies the IP address of the PDP of the PBNM system.

Outputs: None.

6.5.2 *COPS Signalling Process*

Refer to Section 5.4.3 for the function definitions of the *ClientConnect, COPSSendTo, MakeCOPSOPNPacket, MakeCOPSSCPacket, MakeCOPSREQPacket, ReadFrom* functions which are used in the COPS Signalling Process. The *MakeCOPSClientClosePacket* function is defined as follows:

Description: This function is used to create the COPS CC packet

Algorithm:

makeCOPSClientClosePacket Function **Input :** COPS OpCode **(OpCode)** COPS Client-Type **(ClientType)**
Insert COPS version and flag into the packet COPSClientClosePacket.verFlag = setHeaderVerFlag(1,flag)
Insert COPS Opcode into the packet COPSClientClosePacket.OpCode = OpCode
Insert COPS Client-Type into the packet COPSClientClosePacket.clientType = clientType
Insert COPS Class-Num Type (Identifies the COPS Object Type) COPSClientClosePacket.CNum = COPS_OBJ_ERROR
Insert COPS Error Code (Identifies the reason for sending the COPS CC message) COPSClientClosePacket.ErrorCode = SHUTTING_DOWN

Figure 6.10: *MakeCOPSClientClosePacket* Chapin Diagram.

Inputs:

OpCode: uniquely identifies CC message.
ClientType: specifies COPS Client-Type.

Outputs: None.

6.5.3 *RSVP Packet Creation*

6.5.3.1 makeRSVPResvPacket

Description: This function is used to create a RSVP Packet.

Algorithm:

makeRSVPResvPacket Function
Input : Ipdestaddr, protocol_id, flag, dstport , Iphopaddr, Lih, refreshperiod, flagandOptionvector, messageFormat, overallLength, serviceHeader, CtrlLoadDataLength, char parameterId, parameterflag , parameterLength, tokenBucketRate, tokenBucketSize, peakDataRate, minpolicedUnit, maxPacketSize, Ipsrcaddr, srcport
Fill the RSVP header setHeaderVerFlag(Version, flag)
Create RSVP Session Object makeRSVPObjSession(Ipdestaddr, protocol_id, flag, dstport)
Create RSVP Hop Object makeRSVPObjHOP(Iphopaddr,Lih)
Create RSVP Time Object makeRSVPObjTimeValues(refreshperiod)
Create RSVP Reservation Style Object makeRSVPObjStyle(flagandOptionvector)
Create RSVP FlowSpec Style Object makeRSVPObjFlowSpec (messageFormat, overallLength, serviceHeader , CtrlLoadDataLength, parameterId, parameterflag , parameterLength, tokenBucketRate, tokenBucketSize, peakDataRate, minpolicedUnit, maxPacketSize)

Figure 6.11: *makeRSVPResvPacket* Chapin Diagram.

Inputs:

Ipdestaddr: specifies the destination IP address session.
protocol_id: specifies protcol Id for the data flow.
flag: specifies Resv flag.
dstport: specifies the UDP/TCP destination port of the Session.
Iphopaddr: specifies the IP address of the interface who forwarded the message.
Lih: specifies the logical interface handle.
refreshperiod: specifies the timeout period used to created the message.
flagandOptionvector: specifies the reservation style.
messageFormat: specifies the message format version default 0.
overallLength: specifies the flowspec Length, defualt 7 words.
serviceHeader: specifies the service number, default 1.
ParameterId: specifies the token bucket specification, default 127.
Parameterflag: Not used.
ParameterLength: specifies the parameter length, default 5 words.
TokenBucketRate: specifies the token bucket rate.
TokenBucketSize: specifies the token bucket size.
PeakDataRate: specifies the peak data rate.
MinpolicedUnit: specifies the minimum policed unit/traffic control.
MaxPacketSize: specifies the maximum packet size.
Ipsrcaddr: specifies the source IP address of the sender of the message.

Srcport: specifies the source port of the sender of the message.

Outputs: None.

6.5.3.2 RSVP Function

Description: Reads the RSVP Input file (rsvpInputFile) and creates the relevant RSVP Packet and COPS REQ Packets.

Algorithm:

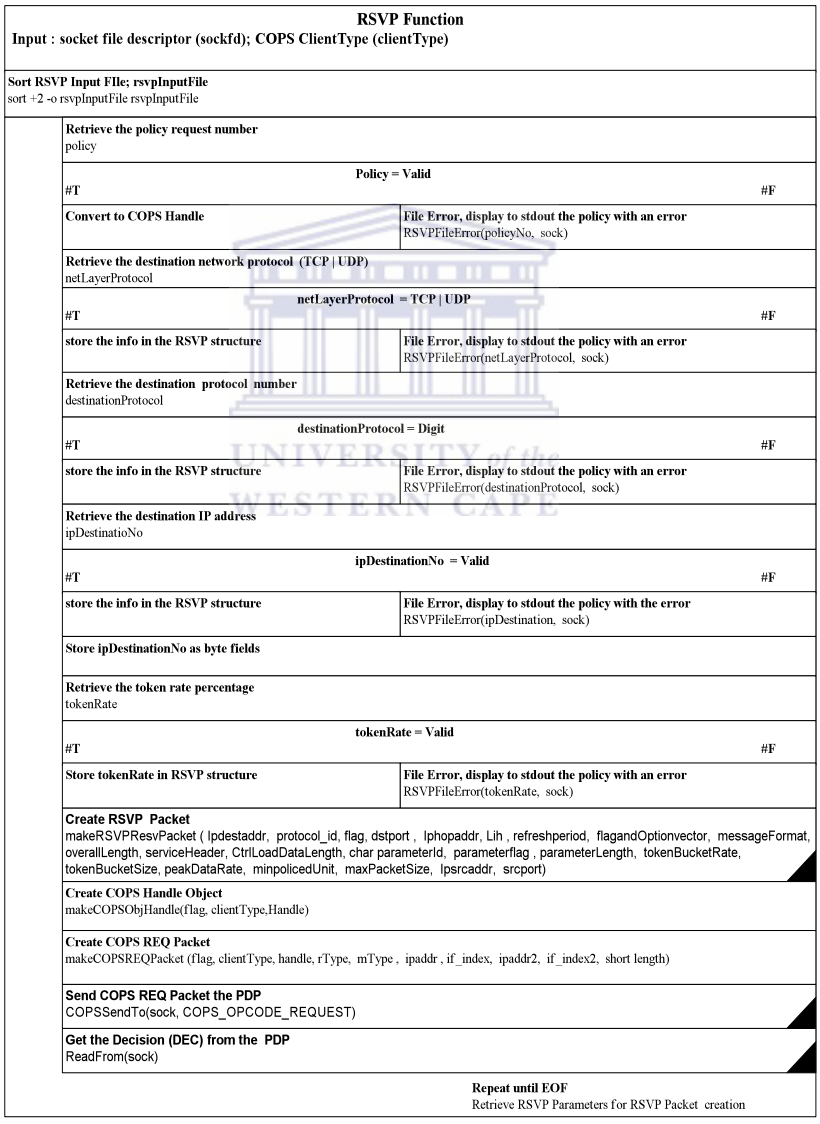


Figure 6.12: *RSVP* Chapin Diagram.

Inputs:

sockfd: specifies the file descriptor.

Outputs: None.

6.5.4 *LPDP Decision Process*

The *LPDP* function of the LPDP Decision Process is defined as follows:

Description: This function makes local decisions for the approved COPS REQ messages for the creation of the CBQ queue class entries. These messages are approved or denied based on the constraints set in the *queuing* file.

Algorithm:

<table>
<tr><td colspan="6">LPDP Function
Input : Bandwidth requested (tokenRate)
Policy Request Number (policyNo)
Transport Layer Protocol (netLayerProtocol - TCP/UDP)</td></tr>
<tr><td colspan="6">netLayerProtocol = TCP
#t #f</td></tr>
<tr><td colspan="3">Check if the Bandwidth exceeds the TCP class maximum allocation
tokenRate > TCPCBQClassMaxAllocation
#t #f</td><td colspan="3">Check if the Bandwidth exceeds the UDP class maximum allocation
tokenRate > UDPCBQClassMaxAllocation
#t #f</td></tr>
<tr><td>Log Event</td><td colspan="2">Check if there is badwidth available within TCP Class
tokenRate + TCPbandwidthAllocated >
TCPCBQClassMaxAllocation</td><td>Log Event</td><td colspan="2">Check if there is badwidth available within TCP Class
tokenRate + UDPbandwidthAllocated >
TCPCBQClassMaxAllocation</td></tr>
<tr><td rowspan="2">Exit
return 1</td><td>Log Event</td><td>Add bandwidth requested to bandwidth allocated
TCPbandwidthAllocated += tokenRate</td><td rowspan="2">Exit
return 1</td><td>Log Event</td><td>Add bandwidth requested to bandwidth allocated
UDPbandwidthAllocated += tokenRate</td></tr>
<tr><td>Exit
return 1</td><td>Exit
return 0</td><td>Exit
return 1</td><td>Exit
return 0</td></tr>
</table>

Figure 6.13: *LPDP* Chapin Diagram.

Inputs:
TokenRate: specifies the bandwidth allocation.
PolicyNo: specifies the PEP Request policy Number.
NetLayerProtocol: specifies the network layer protocol.

Outputs:
0: specifies that the COPS Request has been approved.
1: specifies that the COPS Request has been denied.

6.5.5 *CBQ Queue Creation Process*

The *CreateCBQQueuing* function of the CBQ Queue Creation Process is defined as follows:

Description: This function is used to create the CBQ configuration file, *cbq*

Algorithm:

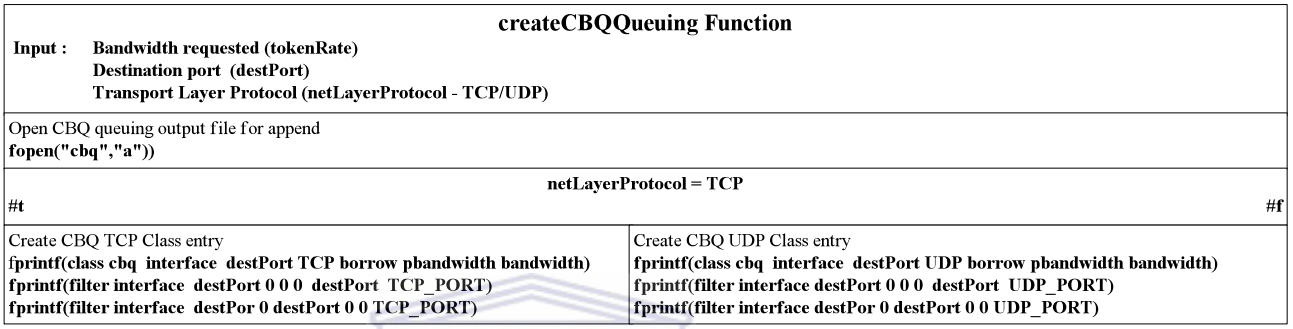


Figure 6.14: CreateCBQQueuing Chapin Diagram.

Inputs:

DestPort: specifies the application port that will have a queue allocated to it.
NetLayerProtocol: specifies the network layer protocol
Bandwidth: specifies the amount of bandwidth allocated to the queue.

Outputs: None.

6.6 Summary of Chapter 6

An overview of the functionality of the Q-PEP was given. The UIS defined what the user interface appears as, while the HLD provided an abstraction of the system to be designed. The LLD defined a detailed view of the system to be developed using Chapin Diagrams.

CHAPTER 7

EXPERIMENTATION

The purpose of the experiments described in this chapter is to verify whether or not the P-PEP and Q-PEP can be implemented according to the ideas and designs presented in the previous two chapters to solve the legacy NEs QoS problems. The network testing scenarios discussed in this chapter are based on the research questions so that it can be assessed whether or not the hypothesis held true. The proposed SNMP based P-PEP solution proved to be undeployable due to inherent SNMP problems and was thus not rigorously tested.

When we refer to normal conditions in this chapter, we are referring to a network condition in which none of the network resources are saturated. The QoS delivered for voice services is based on the latency, jitter and packet loss experienced by the voice traffic and the amount of available bandwidth.

7.1 Overview of Network Testing Equipment and Applications

7.1.1 *SmartBits*

The Spirent Communications SmartBits (SMB) is the industry standard for network performance analysis for 10/100/Gigabit Ethernet, ATM, Packet over SONET, Frame Relay, xDSL, Cable Modem, IP QoS, VoIP, Routing, MulticastIP and TCP/IP. The SMB supports sophisticated automated industry standard performance tests as defined in RFC 1242. The SMB is designed to be used by network equipment manufacturers to perform performance analysis, to test and certify different devices with different manufacturers, to benchmark their performance and to perform comparative analysis of network equipment to determine suitability to a specific application prior to deployment.

7.1.2 *SmartVoIPQoS Application*

The SmartVoIPQoS application has the ability to generate and analyse thousands of VoIP flows and millions of IP data flows simultaneously, in real time. This application is designed to analyse the QoS delivered by NEs that are critical to high quality converged services. The QoS is reported for both the voice and data flows, and the corresponding voice quality scores are reported for voice flows. SmartVoIPQoS is a Windows-based application designed for the SMB. The SmartVoIPQoS utilises the capabilities of SmartMetrics cards to generate traffic that represents different classes of service (voice, data and multimedia), at up to full-line rate. The application then analyses the performance of each incoming stream to test a device's or network's ability to forward very large numbers of data and voice flows by measuring statistics such as loss, latency, and jitter for every packet transmitted. It also analyses the device's ability to correctly handle policies implemented in the network or device under test. SmartVoIPQoS supports testing for all applicable VoIP standards. It provides ITU-T P.861 PSQM voice quality scoring for each voice flow in the test, giving users a standard and objective benchmark to compare voice quality. Thus, using the SmartVoIPQoS application eliminates human error as the application itself collects and analyses the QoS data; thus increasing the validity of and user acceptance a proposed solution.

7.1.3 *Networking Equipment*

The networking equipment (refer to Figure 7.1) used in the experimental testing can be divided into two categories, contemporary and legacy NEs. The contemporary NEs consisted of the Cisco 3640 and 3620 routers which are QoS and COPS-enabled. The legacy NEs consisted of the Q-PEP and the SMB. The SMB was used to simulate a legacy NE as to collect the QoS statistics the legacy NE would experience for voice services.

7.2 Q-PEP Experimental Testing

The network testing is based on two scenarios: congestive and non-congestive. To create a congestive environment, a 100M network segment was channelled into a 10M network segment (refer to Figure 7.1) while the non-congestive scenario was created by channelling

a 100M network segment into a 100M network segment (refer to Figure 7.2). The test data used for these experiments was created using the SmartVoIPQoS application and generated by the SMB.

7.2.1 *Congestive Testing*

The network testing starts off by ascertaining the QoS for all the IP services, i.e. Voice, FTP, HTTP, Telnet and SNMP, when the Q-PEP is not activated and then comparing it to the results when the Q-PEP is activated. The test data used for this experiment is illustrated in Table 7.1. For these experiments, a random packet size was used to simulate a real networking environment. The results of the testing performed when the Q-PEP was deactivated can be seen in Table 7.2.

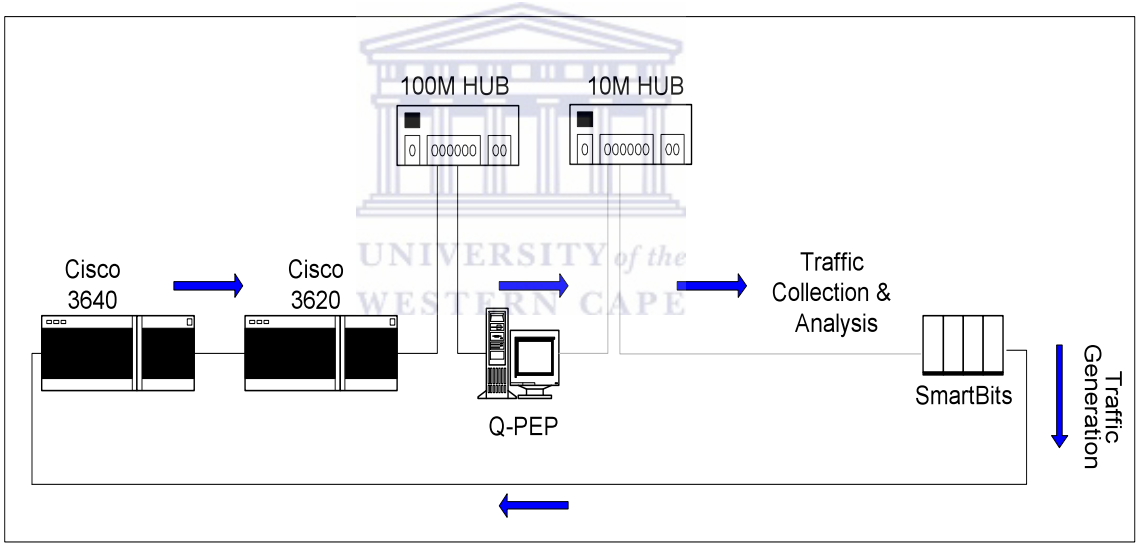


Figure 7.1: Congestive Network Design.

Table 7.1 Test Data Traffic Profile

Packet Type	**No of Packets/min**
Voice	11997
FTP	65534
HTTP	32767
Telnet	32767
SNMP	32767

Table 7.2: Congestive Period Network Testing Results

Packet Type	QoS Metric	QoS Value
Voice	Packet Loss (%)	7.99
	Avg Jitter (μsecs)	292.43
	Avg Latency (μsecs)	806.93
	PSQM	1
FTP	Packet Loss (%)	89.50
	Avg Latency (μsecs)	47177.60
HTTP	Packet Loss (%)	89.47
	Avg Latency (μsecs)	47244.3
Telnet	Packet Loss (%)	89.50
	Avg Latency (μsecs)	47174.20
SNMP	Packet Loss (%)	89.52
	Avg Latency (μsecs)	47162.6

The results show that the packet loss for voice services is above the acceptable limit of 5 percent [60]. The jitter for voice services was found to be 292.43 μmsecs and the latency, 806.93 μmsecs. As indicated in the literature review, we expect the latency to increase and the packet loss and jitter to decrease when the Q-PEP is activated. If this occurs, it proves that the Q-PEP can in fact provide QoS for voice services. Table 7.3 illustrates the network testing results when the Q-PEP was activated using the same test data (refer to Table 7.1), and the same network configuration (refer to Figure 7.1) and using the CBQ configuration illustrated in Table 7.3.

Table 7.3: Congestive Testing CBQ Configuration

CBQ Class	Bandwidth Allocation (%)
Voice	30
FTP	40
HTTP	10
Telnet	10
SNMP	5

Table 7.4: Congestive Period Network Testing Results

Packet Type	QoS Metric	QoS Value
Voice	Packet Loss (%)	0
	Avg Jitter (μsecs)	214.57
	Avg Latency (μsecs)	930.55
	PSQM	1
FTP	Packet Loss (%)	89.52
	Avg Latency (μsecs)	8338.30
HTTP	Packet Loss (%)	97.74
	Avg Latency (μsecs)	11944.3
Telnet	Packet Loss (%)	67.05
	Avg Latency (μsecs)	5085.00
SNMP	Packet Loss (%)	68.29
	Avg Latency (μsecs)	5154.2

The results in Table 7.4 indicate that the packet loss for voice services was reduced to 0 percent from 7.99 percent, the jitter was decreased by 77.86 μmsecs and the latency increased by 123.62 μmsecs when compared to the results in Table 7.2 when the Q-PEP was not active. Thus, the results indicate that the Q-PEP does indeed provide better QoS than when it was deactivated. It must, however, be noted that the latency for all the other IP services which include FTP, SNMP, Telnet and HTTP, increased tremendously as a result of prioritising voice services. It was also found that certain IP services experienced a degradation in QoS with respect to packet loss, FTP, HTTP, while Telnet experienced better QoS. We now investigate whether it is possible to provide QoS for all the services that occur on the network when the Q-PEP is activated, i.e. a CBQ configuration that caters for all IP services.

7.2.1.1 The Search for the Perfect CBQ Configuration

This network testing scenario employs the same network configuration as illustrated in Figure 7.1 and test data illustrated in Table 7.1. For the first experiment, we investigated the effects of reducing voice services bandwidth allocation to 5 percent from 30 percent. The CBQ configuration used for this experiment is illustrated in Table 7.5 and the results are shown Table 7.6.

Table 7.5: Congestive Testing CBQ Configuration

CBQ Class	Bandwidth Allocation (%)
Voice	5
FTP	35
HTTP	30
Telnet	15
SNMP	10

Table 7.6: Congestive Period Network Testing Results

Packet Type	QoS Metric	QoS Value
Voice	Packet Loss (%)	0.34
	Avg Jitter (μsecs)	536.95
	Avg Latency (μsecs)	6267.9
	PSQM	1
FTP	Packet Loss (%)	91.84
	Avg Latency (μsecs)	9014.5
HTTP	Packet Loss (%)	94.61
	Avg Latency (μsecs)	32290.40
Telnet	Packet Loss (%)	69.62
	Avg Latency (μsecs)	5053.90
SNMP	Packet Loss (%)	70.65
	Avg Latency (μsecs)	5122.90

The results revealed that voice services experienced a degradation in QoS. These results are however still better than when the Q-PEP was deactivated with respect to packet loss (refer to Table 7.2), the latency and jitter are, however, worse. The FTP service also experienced a degradation in QoS. For this particular service (FTP), the QoS experienced was worse than when the Q-PEP was deactivated (refer to Table 7.2). The resulting degradation in QoS can be accounted for due to the bandwidth allocations being decreased in the CBQ configuration (refer to Table 7.5). Thus, it can be concluded that the bandwidth allocated to a service affects the QoS delivered. It is possible for the QoS experienced by a IP service to be worse than when the Q-PEP was not activated.

The main aim of this research was to provide better QoS for voice services. Thus the first priority is to configure the CBQ queuing such that no packet loss, low latency and jitter are experienced for voice services and then try to accommodate the rest of the IP services. In

the light of this, we increase the voice services bandwidth allocation to 10 percent from 5 percent (refer to Table 7.7).

Table 7.7: Congestive Testing CBQ Configuration

CBQ Class	Bandwidth Allocation (%)
Voice	10
FTP	15
HTTP	35
Telnet	25
SNMP	10

Table 7.8: Congestive Period Network Testing Results

Packet Type	QoS Metric	QoS Value
Voice	Packet Loss (%)	0
	Avg Jitter (μsecs)	206.58
	Avg Latency (μsecs)	932.75
	PSQM	1
FTP	Packet Loss (%)	97.04
	Avg Latency (μsecs)	11187.25
HTTP	Packet Loss (%)	92.90
	Avg Latency (μsecs)	8336.60
Telnet	Packet Loss (%)	66.23
	Avg Latency (μsecs)	5252.20
SNMP	Packet Loss (%)	67.39
	Avg Latency (μsecs)	5122.50

The QoS for voice services improved, no packet loss was experienced, the jitter decreased by 330.37 μmsecs to 206.58 μmsecs and latency decreased by 5335.15 μmsecs to 932.75 μmsecs (refer to Table 7.8). In fact, the network testing results from this CBQ configuration for voice services (refer to Table 7.8) are similar to the network testing results when 30 percent of the bandwidth was allocated for voice services (refer to Table 7.4). Thus, over-allocating the CBQ queue for voice services (i.e. over provisioning), does not improve the QoS experienced. All that is needed is the correct amount of bandwidth to cater for voice services. Thus, a careful study of the network is needed when deploying such a solution due to the possibility that a deterioration in the QoS could be experienced. An example of which would be when 5 percent of the bandwidth was allocated for voice

services (refer to Table 7.6). Bandwidth could also be wasted when the CBQ queues are over-provisioned. An example of which would be when 30 percent of the bandwidth was allocated for voice services (refer to Table 7.4) and similar QoS was experienced as when only 10 percent was allocated. Over-allocating the bandwidth for voice services does not have an impact on the QoS experienced by voice services but does impact on other IP services as less bandwidth is available for them.

We then experimented with the possibility of providing the same QoS for another IP service as for voice services by keeping the bandwidth allocation of voice service at 10 percent and increasing Telnet's bandwidth allocation to 50 percent.

Table 7.9: Congestive Testing CBQ Configuration

CBQ Class	Bandwidth Allocation (%)
Voice	10
FTP	25
HTTP	5
Telnet	50
SNMP	5

Table 7.10: Congestive Period Network Testing Results

Packet Type	QoS Metric	QoS Value
Voice	Packet Loss (%)	0
	Avg Jitter (μsecs)	198.95
	Avg Latency (μsecs)	891.60
	PSQM	1
FTP	Packet Loss (%)	92.45
	Avg Latency (μsecs)	8682.20
HTTP	Packet Loss (%)	98.20
	Avg Latency (μsecs)	12789.30
Telnet	Packet Loss (%)	62.62
	Avg Latency (μsecs)	5071.90
SNMP	Packet Loss (%)	63.78
	Avg Latency (μsecs)	5145.70

Comparing the results in Table 7.10 where Telnet was allocated 50 percent of the bandwidth to the results of Table 7.7 where Telnet was allocated 25 percent of the bandwidth, it can be concluded that, the more bandwidth allocated to a service the better the QoS will be experienced, up to the point where an excess of bandwidth is added and no better QoS is experienced. An example of which would be when comparing 10 percent (refer to Table 7.10) and 30 percent (refer to Table 7.4) allocated to the voice services, the QoS experienced was almost exactly the same. Thus, we predict that, by allocating more bandwidth to the Telnet, service better QoS should be experienced. Telnet's bandwidth allocation is increased to 70 percent from 50 percent.

Table 7.11: Congestive Testing CBQ Configuration

CBQ Class	Bandwidth Allocation (%)
Voice	10
FTP	5
HTTP	5
Telnet	70
SNMP	5

Table 7.12: Congestive Period Network Testing Results

Packet Type	**QoS Metric**	**QoS Value**
Voice	Packet Loss (%)	0
	Avg Jitter (μsecs)	176.45
	Avg Latency (μsecs)	840.05
	PSQM	1
FTP	Packet Loss (%)	97.57
	Avg Latency (μsecs)	11516.70
HTTP	Packet Loss (%)	97.92
	Avg Latency (μsecs)	10793.80
Telnet	Packet Loss (%)	53.31
	Avg Latency (μsecs)	5035.30
SNMP	Packet Loss (%)	57.38
	Avg Latency (μsecs)	5117.90

It can be seen from the results (refer to Table 7.12) that the QoS experienced by the Telnet service was improved. The packet loss decreased to 53.31 percent from 62.62 percent, the latency however remains similar, a decrease in 36.6 μmsecs. Thus, the

bandwidth allocated to a service is directly related to the QoS experienced. The more the better, up to the point where increasing the bandwidth has no effect on the QoS experienced.

What is made clear from the last 3 experiments when voice services were allocated 10 percent of the bandwidth, is that the QoS for voice services remained constant. No packets were lost, the jitter ranged from 206.80 μmsecs to 176.45 μmsecs and the latency ranged from 932.75 μmsecs to 840.05 μmsecs. The variation in the jitter and latency could be as a result of the random packet sizes used for these experiments.

7.2.1.2 IP Service Packet Size Testing

We then tested whether or not the size of IP packets of IP services other than voice services affect the QoS experienced by voice services. The tests performed before this point had a variance in packet size to simulate a real networking environment. We now control the packet size of the non-voice services. The following packet sizes, 80 bytes; 128 bytes; 500 bytes; 1000 bytes and 1500 bytes were allocated to FTP, HTTP, Telnet and SNMP services respectively. The networking testing performed used the CBQ configuration illustrated in Table 7.13 and traffic profile illustrated in Table 7.14. Graphs will be used to visually illustrate the results obtained at the end of this section. In this section, when we refer to non-voice packets, we are referring to FTP, HTTP, Telnet and SNMP IP Packets.

Table 7.13: Congestive Testing CBQ Configuration

CBQ Class	Bandwidth Allocation (%)
Voice	10
FTP	45
HTTP	20
Telnet	10
SNMP	10

Table 7.14 Test Data Traffic Profile

Packet Type	No of Packets/min
Voice	11997
FTP	131068
HTTP	65534
Telnet	65534
SNMP	65534

Table 7.15: Congestive Network Testing Frame Size 80 Results

Packet Type	QoS Metric	QoS Value
Voice	Packet Loss (%)	0
	Avg Jitter (µsecs)	420.58
	Avg Latency (µsecs)	1094.18
	PSQM	1
FTP	Packet Loss (%)	79.68
	Avg Latency (µsecs)	6164
HTTP	Packet Loss (%)	81.83
	Avg Latency (µsecs)	6239.70
Telnet	Packet Loss (%)	58.00
	Avg Latency (µsecs)	9664.80
SNMP	Packet Loss (%)	58.80
	Avg Latency (µsecs)	4695

Table 7.16: Congestive Network Testing Frame Size 128 Results

Packet Type	QoS Metric	QoS Value
Voice	Packet Loss (%)	0
	Avg Jitter (µsecs)	226.38
	Avg Latency (µsecs)	894.18
	PSQM	1
FTP	Packet Loss (%)	86.09
	Avg Latency (µsecs)	6527.30
HTTP	Packet Loss (%)	87.56
	Avg Latency (µsecs)	6605.20
Telnet	Packet Loss (%)	71.43
	Avg Latency (µsecs)	4774.00
SNMP	Packet Loss (%)	72.27
	Avg Latency (µsecs)	4837.30

Table 7.17: Congestive Network Testing Frame Size 500 Results

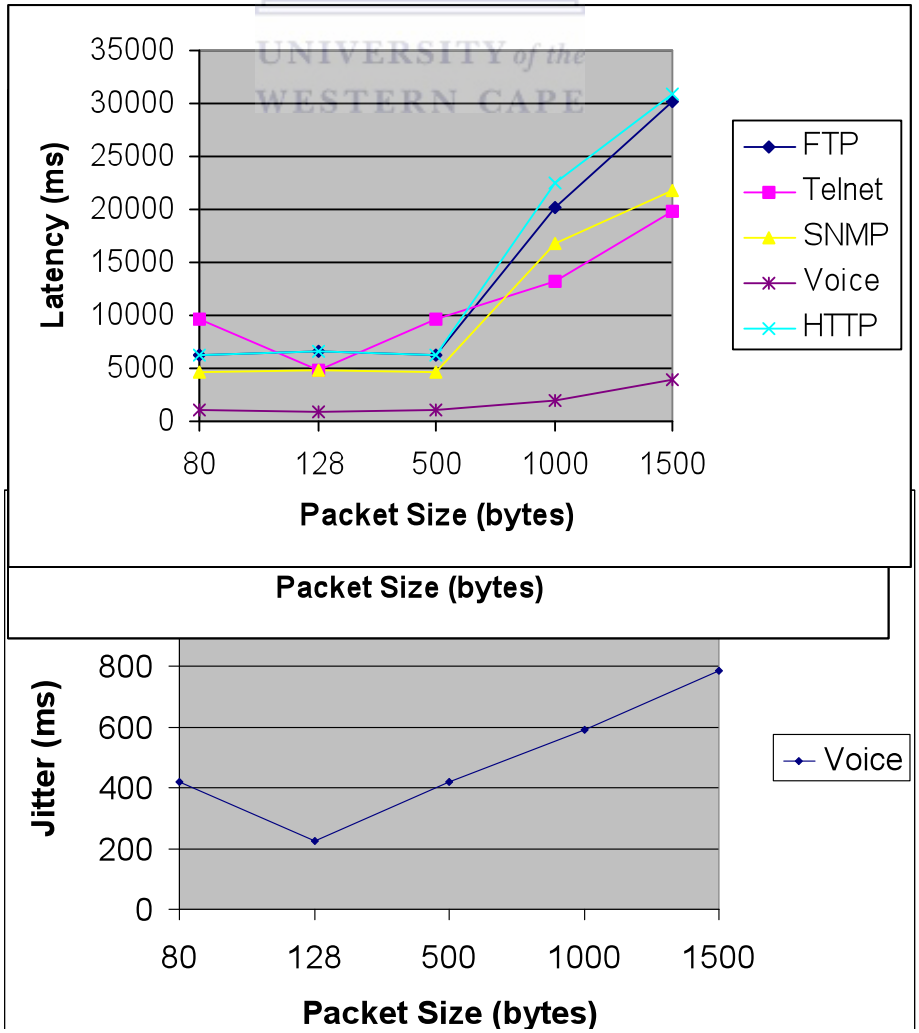
Packet Type	QoS Metric	QoS Value
Voice	Packet Loss (%)	0
	Avg Jitter (μsecs)	420.58
	Avg Latency (μsecs)	1094.18
	PSQM	1
FTP	Packet Loss (%)	79.68
	Avg Latency (μsecs)	6164
HTTP	Packet Loss (%)	81.83
	Avg Latency (μsecs)	6239.70
Telnet	Packet Loss (%)	58.00
	Avg Latency (μsecs)	9664.80
SNMP	Packet Loss (%)	58.80
	Avg Latency (μsecs)	4695

Table 7.18: Congestive Network Testing Frame Size 1000 Results

Packet Type	QoS Metric	QoS Value
Voice	Packet Loss (%)	0
	Avg Jitter (μsecs)	592.40
	Avg Latency (μsecs)	1913.33
	PSQM	1
FTP	Packet Loss (%)	85.50
	Avg Latency (μsecs)	20174.80
HTTP	Packet Loss (%)	87.24
	Avg Latency (μsecs)	22439.50
Telnet	Packet Loss (%)	93.76
	Avg Latency (μsecs)	13133.20
SNMP	Packet Loss (%)	97.87
	Avg Latency (μsecs)	16764.50

Table 7.19: Congestive Network Testing Frame Size 1500 Results

Packet Type	QoS Metric	Q-PEP Activated Table 7.13 CBQ Configuration
Voice	Packet Loss (%)	0
	Avg Jitter (µsecs)	786.33
	Avg Latency (µsecs)	3903.38
	PSQM	1
FTP	Packet Loss (%)	87.66
	Avg Latency (µsecs)	30227.70
HTTP	Packet Loss (%)	89.09
	Avg Latency (µsecs)	30877.50
Telnet	Packet Loss (%)	90.41
	Avg Latency (µsecs)	19835.80
SNMP	Packet Loss (%)	95.06
	Avg Latency (µsecs)	21702.10



Graph 7.3: Congestive Period Jitter vs Packet Size

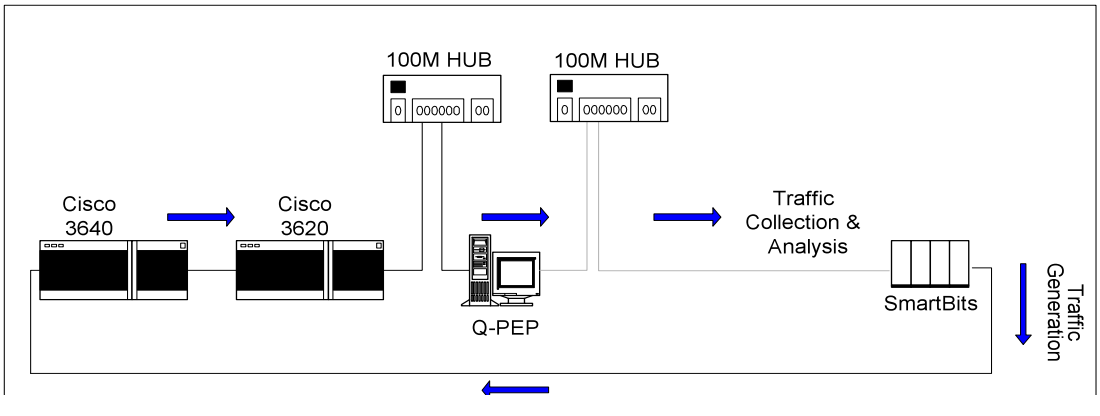
It is apparent from the experiments performed that voice services experienced the lowest packet loss when the non-voice packet sizes were of size 80 bytes and 500 bytes. Voice services experienced the lowest latency and jitter when the non-voice packet sizes were fixed to 128 bytes. Latency increased for voice services from packet size 500 bytes and did not taper off. Jitter followed the similar pattern, but from packet size 128 bytes onwards. It is clear from the graphs that for all QoS factors, the bigger the packet size, the worse the QoS can be expected for voice services, except for packets with the packet size of 500 bytes. There is no logical explanation for this, but it must be noted that VoIP packets tend to be smaller than 250 bytes, which is the range for which voice services experienced the best jitter and latency and average packet loss.

7.2.1.3 Congestive Network Testing Conclusion

Looking at the results of the congestive network testing, we observed that QoS cannot be provided for all IP services, and that the smaller the packet size of non-voice services, the better QoS will be experienced for voice services. It is also apparent that no matter how effective the traffic shaping is, there will come a time when the networking infrastructure needs to be upgraded. This becomes apparent when the network cannot support a particular service. This phenomenon was observed with the Telnet service when 70 percent of the bandwidth was allocated to this IP service, it still experienced 53.31 percent packet loss (refer to Table 7.12).

7.2.2 *Non-congestive Testing*

A non-congestive test follows the same experimental scheme as for congestive network testing, by providing testing results when the Q-PEP is deactivated and then when the Q-PEP is activated. The network configuration is illustrated in Figure 7.2 and the data



generated illustrated in Table 7.20 for these experiments. For this experiments, a random packet size was used to simulate a real networking environment.

Figure 7.2: Non congestive Network Design

Table 7.20 Test Data Traffic Profile

Packet Type	No of Packets/min
Voice	11997
FTP	65534
HTTP	32767
Telnet	32767
SNMP	32767

Table 7.21: Non-Congestive Period Network Testing Results

Packet Type	QoS Metric	QoS Value
Voice	Packet Loss (%)	0.02
	Avg Jitter (μsecs)	20.7
	Avg Latency (μsecs)	463.80
	PSQM	1
FTP	Packet Loss (%)	0.05
	Avg Latency (μsecs)	3440.55
HTTP	Packet Loss (%)	0.06
	Avg Latency (μsecs)	3440.30
Telnet	Packet Loss (%)	0.04
	Avg Latency (μsecs)	3440.70
SNMP	Packet Loss (%)	0.05
	Avg Latency (μsecs)	3440.30

From the network testing results (refer to Table 7.21), the QoS for voice services provided during these periods of non-congestion is user acceptable. The packet loss for all the IP services are minimal, the worst case being, HTTP at 0.06 percent. The first network testing scenario, will use the CBQ configuration illustrated in Table 7.22, and the results are illustrated in Table 7.23.

Table 7.22: Non-Congestive Testing CBQ Configuration

CBQ Class	Bandwidth Allocation (%)
Voice	5
FTP	35
HTTP	30
Telnet	15
SNMP	10

Table 7.23: Non-Congestive Period Network Testing Results

Packet Type	QoS Metric	QoS Value
Voice	Packet Loss (%)	0
	Avg Jitter (μsecs)	14.63
	Avg Latency (μsecs)	149.25
	PSQM	1
FTP	Packet Loss (%)	0.89
	Avg Latency (μsecs)	3179.75
HTTP	Packet Loss (%)	0
	Avg Latency (μsecs)	565.40
Telnet	Packet Loss (%)	0
	Avg Latency (μsecs)	611.8
SNMP	Packet Loss (%)	0
	Avg Latency (μsecs)	632.8

Comparing Table 7.21 and Table 7.23, the QoS for voice services improved for all the parameters, packet loss, latency and jitter. This was not expected. According to the literature review, the latency should increase. The results (refer to Table 7.21) also indicate that the services, FTP, HTTP, Telnet and SNMP, experienced better QoS when the Q-PEP was activated. This is a result of the Q-PEP employing a CBQ mechanism. When the Q-PEP is deactivated, the standard traffic shaping mechanism FIFO is used. In this queuing mechanism, packets are forwarded as they are received and no priority is given for any services resulting in variable QoS. From the results (refer to Table 7.21), all the services experienced packet loss. However when the Q-PEP was activated, using the CBQ configuration in Table 7.22, only FTP suffered packet loss, however the latency experienced was better (refer to Figure 7.23). This proves that the Q-PEP can provide QoS for voice services under normal conditions. We next tried to find the perfect configuration, so that all the IP services can receive better QoS. FTP is the only service that experienced packet loss; thus in order to alleviate the packet loss, more bandwidth was allocated for this service.

The bandwidth allocated for FTP services as illustrated in Table 7.24 was increased by 10 percent, to 45 percent. We now investigate whether it is possible to provide QoS for all the services that occur on the network when the Q-PEP is activated, i.e. a CBQ configuration that caters for all IP services.

7.2.2.1 The search for the perfect CBQ configuration

The network testing scenario in this section employs the same network configuration as illustrated in Figure 7.2 and test data illustrated in Table 7.20. The CBQ configuration used for this experiment is illustrated in Table 7.24 and the results in Table 7.25.

Table 7.24: Congestive Testing CBQ Configuration

CBQ Class	Bandwidth Allocation (%)
Voice	10
FTP	45
HTTP	10
Telnet	20
SNMP	10

Table 7.25: Non-Congestive Period Network Testing Results

Packet Type	QoS Metric	QoS Value
Voice	Packet Loss (%)	0
	Avg Jitter (μsecs)	18.63
	Avg Latency (μsecs)	134.63
	PSQM	1
FTP	Packet Loss (%)	0.23
	Avg Latency (μsecs)	2612.65
HTTP	Packet Loss (%)	0.90
	Avg Latency (μsecs)	4989.10
Telnet	Packet Loss (%)	0
	Avg Latency (μsecs)	508
SNMP	Packet Loss (%)	0
	Avg Latency (μsecs)	535.70

The results in Table 7.25 reveal that the QoS experienced for the FTP services has been improved but packets are still being lost. HTTP experienced a degradation in QoS which can be accounted for by the bandwidth allocation decrease of 20 percent, to 10 percent.

We now increase FTP's bandwidth allocation to 48 percent and Telnet's to 13 percent, to see if all the services' QoS needs can be catered for.

Table 7.26: Congestive Testing CBQ Configuration

CBQ Class	Bandwidth Allocation (%)
Voice	10
FTP	48
HTTP	14
Telnet	13
SNMP	10

Table 7.27: Non-Congestive Network Testing Results

Packet Type	QoS Metric	QoS Value
Voice	Packet Loss (%)	0
	Avg Jitter (μsecs)	18.35
	Avg Latency (μsecs)	134.85
	PSQM	1
FTP	Packet Loss (%)	0
	Avg Latency (μsecs)	645.95
HTTP	Packet Loss (%)	1.82
	Avg Latency (μsecs)	5367.40
Telnet	Packet Loss (%)	0
	Avg Latency (μsecs)	511.6
SNMP	Packet Loss (%)	0
	Avg Latency (μsecs)	539

The results (refer to Table 7.27) indicate that the only service that is experiencing packet loss is HTTP. Thus we increased the bandwidth allocation for HTTP to 20 percent, but in order to do that, the bandwidth allocation for voice, FTP, Telnet and SNMP needs to be slightly decreased. The CBQ configuration is illustrated in Table 7.28 and the results in Table 7.29.

Table 7.28: Non-Congestive Testing CBQ Configuration

CBQ Class	Bandwidth Allocation (%)
Voice	8

FTP	46
HTTP	20
Telnet	12
SNMP	9

Table 7.29: Non-Congestive Period Network Testing Results

Packet Type	QoS Metric	QoS Value
Voice	Packet Loss (%)	0
	Avg Jitter (μsecs)	22.5
	Avg Latency (μsecs)	153.75
	PSQM	1
FTP	Packet Loss (%)	0
	Avg Latency (μsecs)	677.65
HTTP	Packet Loss (%)	1.80
	Avg Latency (μsecs)	5197.00
Telnet	Packet Loss (%)	0
	Avg Latency (μsecs)	613.4
SNMP	Packet Loss (%)	0
	Avg Latency (μsecs)	635.60

From the results illustrated in Table 7.29 the QoS experienced improved; by increasing the bandwidth allocation for HTTP, however, packet loss was still experienced. More importantly though, is that all the services, Voice, Telnet, FTP that had their bandwidth allocation reduced, experienced a degradation in QoS with respect to latency and jitter. However, no packet loss was experienced. It can be concluded that it is not possible to cater for all services, but the services that are important or that need high levels of QoS can be provided for. The solution, thus, can provide QoS not only for voice services but for any service that needs higher levels of QoS.

7.2.2.2 IP Service Packet Size Testing

We then tested whether or not the size of IP packets of IP services under normal conditions other than voice services affects the QoS experienced by voice services. The tests performed before this point had a variance in packet size to simulate a real networking environment. We now control the packet size of the non-voice services. The following packet sizes, 80 bytes; 128 bytes; 500 bytes; 1000 bytes and 1500 bytes were allocated to FTP, HTTP, Telnet and SNMP services. The networking testing performed used the CBQ configuration illustrated in Table 7.13 and traffic profile illustrated in Table

7.14. Graphs will be used to visually illustrate the results obtained at the end of this section.

Table 7.30: Non-Congestive Network Testing Frame Size 80 Results

Packet Type	QoS Metric	QoS Value
Voice	Packet Loss (%)	0.53
	Avg Jitter (μsecs)	30.03
	Avg Latency (μsecs)	250.00
	PSQM	1
FTP	Packet Loss (%)	41.51
	Avg Latency (μsecs)	1977.15
HTTP	Packet Loss (%)	37.83
	Avg Latency (μsecs)	1975.50
Telnet	Packet Loss (%)	38.33
	Avg Latency (μsecs)	1978.00
SNMP	Packet Loss (%)	40.19
	Avg Latency (μsecs)	1977.30

Table 7.31: Non-Congestive Network Testing Frame Size 128 Results

Packet Type	QoS Metric	QoS Value
Voice	Packet Loss (%)	1.10
	Avg Jitter (μsecs)	17.03
	Avg Latency (μsecs)	200.45
	PSQM	1
FTP	Packet Loss (%)	44.49
	Avg Latency (μsecs)	2071.80
HTTP	Packet Loss (%)	39.20
	Avg Latency (μsecs)	2069.80
Telnet	Packet Loss (%)	40.98
	Avg Latency (μsecs)	2072.70
SNMP	Packet Loss (%)	43.03
	Avg Latency (μsecs)	2072.71

Table 7.32: Non-Congestive Network Testing Frame Size 500 Results

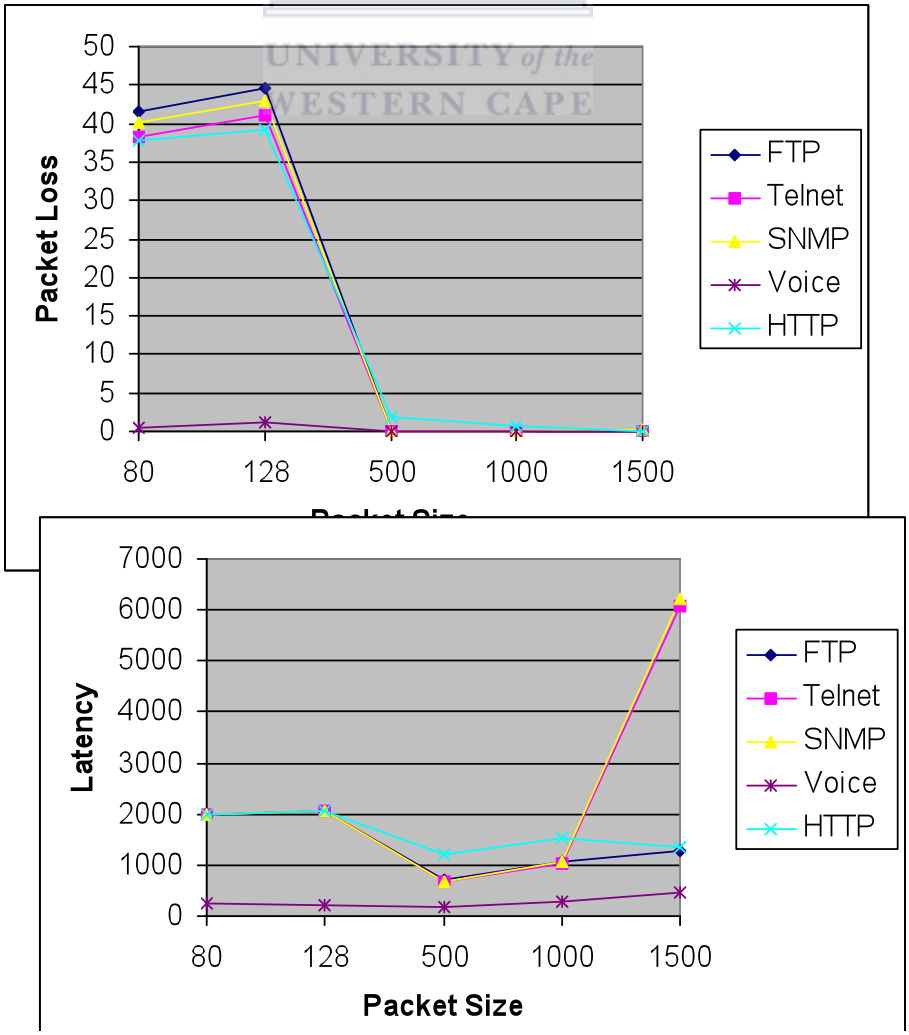
Packet Type	QoS Metric	QoS Value
Voice	Packet Loss (%)	0
	Avg Jitter (μsecs)	22.08
	Avg Latency (μsecs)	162.73
	PSQM	1
FTP	Packet Loss (%)	0
	Avg Latency (μsecs)	695.95
HTTP	Packet Loss (%)	1.77
	Avg Latency (μsecs)	1209.5
Telnet	Packet Loss (%)	0
	Avg Latency (μsecs)	660.30
SNMP	Packet Loss (%)	0
	Avg Latency (μsecs)	683.70

Table 7.33: Non-Congestive Network Testing Frame Size 1000 Results

Packet Type	QoS Metric	QoS Value
Voice	Packet Loss (%)	0
	Avg Jitter (μsecs)	38.62
	Avg Latency (μsecs)	276.35
	PSQM	1
FTP	Packet Loss (%)	0
	Avg Latency (μsecs)	1065.70
HTTP	Packet Loss (%)	0.62
	Avg Latency (μsecs)	1532.80
Telnet	Packet Loss (%)	0
	Avg Latency (μsecs)	1016.60
SNMP	Packet Loss (%)	0
	Avg Latency (μsecs)	1076.30

Table 7.34: Non-Congestive Network Testing Frame Size 1500 Results

Packet Type	QoS Metric	QoS Value
Voice	Packet Loss (%)	0
	Avg Jitter (µsecs)	58.48
	Avg Latency (µsecs)	476.88
	PSQM	1
FTP	Packet Loss (%)	0
	Avg Latency (µsecs)	1270.15
HTTP	Packet Loss (%)	0
	Avg Latency (µsecs)	1339.50
Telnet	Packet Loss (%)	0
	Avg Latency (µsecs)	6083.90
SNMP	Packet Loss (%)	0.27
	Avg Latency (µsecs)	6229.90



Graph 7.6: Congestive Period Jitter vs Packet Size

Similar results for voice services were experienced for jitter in this experiment as for the congestive network packet size testing (refer to Section 7.2.12). The jitter increased from packet size 128 bytes and did not taper off. Under normal conditions, the results indicate that voice services experienced better QoS with respect to packet loss when the non-voice packets were greater than 128 bytes. However, the contrast was experienced with respect to latency.

7.2.2.3 CBQ Priority Testing

The next set of results illustrates the priority factor of CBQ by bandwidth allocation. In this testing scenarios the CBQ parameters were varied (refer to Table 7.35 & Table 7.36) using the same test data illustrated in Table 7.20, to observe the effects of changing the bandwidth allocation sizes of voice services on the other traffic types and itself. The network depicted in Figure 7.2 - a network under normal conditions - was used for this testing scenario.

It can be seen that packets were dropped and the latency experienced by the voice services was extremely high when the bandwidth allocation size of the voice services was dropped to 5 percent (refer to Table 7.37). The CBQ configuration in Table 7.36 allocated 65 percent of the bandwidth compared to 35 percent when using the CBQ configuration in Table 7.35 to FTP services. We obtained similar results for FTP services as for voice services when more bandwidth was allocated: better QoS was experienced. However, this results in less bandwidth available for other IP services and a possible degradation in QoS. HTTPs' bandwidth allocation was kept constant for both testing scenarios (refer to Table 7.37) and resulted in similar QoS being experienced. This indicates that it is possible to predict the QoS for any IP service for a particular amount of bandwidth allocated to a CBQ queue for a known traffic load.

Table 7.35: Extreme High Voice priority CBQ Configuration

CBQ Class	Bandwidth Allocation (%)
Voice	35
FTP	35
HTTP	10
Telnet	10
SNMP	5

Table 7.36: Extreme Low Voice priority CBQ Configuration

CBQ Class	Bandwidth Allocation (%)
Voice	5
FTP	65
HTTP	10
Telnet	5
SNMP	10

Table 7.37: CBQ Variation Network Testing Results

Packet Type	QoS Metric	Q-PEP Activated Table 7.35 CBQ Config	Q-PEP Activated Table 7.36 CBQ Config
Voice	Packet Loss (%)	0	2.60
	Avg Jitter (μsecs)	123.60	562.15
	Avg Latency (μsecs)	902.93	6381.52
	PSQM	1	1
FTP	Packet Loss (%)	90.67	76.78
	Avg Latency (μsecs)	8224.9	6381.52
HTTP	Packet Loss (%)	98.3	97.97
	Avg Latency (μsecs)	13549.70	13642.20
Telnet	Packet Loss (%)	65.08	71.43
	Avg Latency (μsecs)	5080.10	5047.60
SNMP	Packet Loss (%)	66.52	72.66
	Avg Latency (μsecs)	5149.50	5114.20

7.2.2.4 Non-Congestive Network Testing Conclusion

Looking at the results of the non-congestive network testing, it is apparent that perfect QoS (no packet loss, very low jitter and latency) cannot be provided for all IP services. It was also made clear that voice services experienced better latency and jitter when the non-IP service packet size ranged from 80 bytes to 500 bytes. Better packet loss was

experienced when the non-IP service packet size ranged from 500 bytes to 1500 bytes. These results strengthens the results of the congestive networking testing with respect to not being able to provide QoS for all IP services; as under both conditions, congestive and non-congestive it was not possible to provide QoS for all IP services.

7.3 Summary of Chapter 7

QoS experienced by a network user can be measured based on the QoS provided by the network for a particular service, in our case voice services. The QoS is measured based on the latency, jitter and packet loss experienced by a service. The popularity of VoIP has led to the development of QoS measurement tools, such as the SMB and SmartVoIPQoS application to verify the ability of networks to provide QoS. The proposed SNMP based P-PEP solution proved to be undeployable due to inherent SNMP problems. Performance measurements and experiments were carried out with the Q-PEP using the SmartVoIPQoS application. The results revealed that QoS can be provided for voice services using the proposed Q-PEP solution but with adverse effects for other IP based services.

CHAPTER 8

CONCLUSION

The aims of this research were two-fold. Firstly, the study sought to investigate whether it was possible for the legacy NEs to be managed by the common network management protocol, SNMP, so that it could be seamlessly integrated into a PBNM environment. Secondly, it aimed to find out whether or not voice services could be deployed on legacy filled networks providing some level of QoS, using PBNM together with traffic shaping.

Chapter 7 describes how the Spirent Communications SMB was used to stress test the solution on different network configurations and collect and analyse the various results. The aim of this chapter is to provide answers to the research questions and to provide a discussion of the findings of the study and some of the more important implications relevant for QoS delivery for legacy NEs. We with some suggestions discuss directions for further research, and the limitations of the study will be discussed.

8.1 Research Questions

The research questions which guide the study, as outlined in Section 1.4 and the corresponding findings, are described in the same order as Section 1.4:

1. Is it possible for legacy NEs to operate in a PBNM environment?

In its simplest form, a COPS PEP is a software entity operating on a contemporary NE. This was proven true by the software designed and developed. Figure 6.8 illustrates the messages exchanged between the PDP and Q-PEP COPS client. The COPS aspect of the Q-PEP was developed according to RFC 2748 and thus met all the standards required for interoperability with any PDP. Thus, it is possible for a non-contemporary NE to get COPS PDP directives and communicate with the PDP. The Q-PEP, however, differed to

a contemporary NE by applying the COPS PDP decisions to Cho's [9] queuing mechanism ALTQ instead of RSVP flows.

2. Is it possible for SNMP to emulate a PBNM environment?

SNMP has the functionality to simulate the COPS Protocol by employing SNMP Sets and Traps to emulate the COPS request and decision messages. From Section 2.4, SNMP Traps are used to identify a predefined NE instance and then inform the SNMP network manager of the occurrence. However, the firmware has limitations as to which SNMP Traps are set. The SNMP Traps that can be set are defined by RFC 1157 and do not support the proposed solution [27]. Thus the idea of using SNMP Traps to trigger the arrival of certain traffic types to provide differential treatment was ruled out. However, NE vendors have developed proprietary SNMP MIB extensions, but even these lack the functionality. Thus, to cater for the functionality needed for the P-PEP, it is necessary to create new SNMP Traps for the specific instances needed; they can be deployed on the firmware. This needs to be done for all the Vendor specific NEs to ensure interoperability. The likelihood of this happening is grim, as these NEs are old and most of them have been discontinued.

3. Can SNMP be used to provide QoS when used to provide admission control?

It was found that due to the limitations of the MIB structures of SNMPv1 and SNMPv2, it was not possible for the legacy NEs to provide admission control for transiting network traffic. As a result of SNMPv1 and SNMPv2 security issues [27], SNMP Management applications are usually reduced to monitoring tools and not configuration tools, thus, configuring the legacy NEs to either deny or allow network traffic to transit the NE with SNMP was not possible. These problems are however not hardware associated but firmware related and can be rectified by the associated NE Vendors by adding proprietary SNMP MIB extensions.

4. Does traffic shaping in the form of CBQ queuing provide QoS within a PBNM paradigm during periods of network congestion?

The test data used for this experiment (refer to Table 7.20) was created using the SmartVoIPQoS application and generated by the SMB. Table 7.3, illustrates the CBQ queuing configuration created. It should be noted that in this CBQ configuration example, voice services was given 30 percent of the total bandwidth.

The analysis of the results collected by the SMB, which is illustrated in Table 7.2 and Table 7.4, revealed that the number of packets dropped for voice services when the Q-PEP was deactivated, was much higher than when it was activated. In fact, no voice packets were dropped when the solution was activated. The jitter for voice services was also considerably lower. However, the average latency experienced by voice services increased. From these results, we can conclude that the Q-PEP does in fact provide some level of QoS for voice services under congestive network periods and thus proves our hypothesis true according to Comte.

5. What are the effects of employing traffic shaping?

Under congestive network conditions, we found that the average latency experienced by all the IP services was higher (refer to Table 7.2 and Table 7.4). This can be accounted for by the latency added by the Q-PEP which sort the packets into the correct CBQ queue and the packets waiting to be serviced in these queues, if at all. We also observed, that the HTTP CBQ queue which was allocated 10% (refer to Table 7.3) of the bandwidth, experienced heavier packet loss when the Q-PEP was activated (refer to Table 7.2 and Table 7.4). This occurrence is due to either the packet time-to-live (TTL) expiring or HTTP's CBQ queue buffer overflowing and resulting in packets dropped.

Using the test data illustrated in Table 7.20 and the CBQ configuration in Table 7.22, the Q-PEP was tested under normal conditions (refer to Figure 7.2). The results revealed that voice services still received better QoS when the Q-PEP was activated (refer to Table 7.23) compared to when the Q-PEP was not activated (refer to Table 7.21). However FTP

experienced higher latency and packet loss when the Q-PEP was activated when compared to the latency and packet loss experienced when the Q-PEP was deactivated. This is a result of the CBQ configuration not allocating enough bandwidth resulting in CBQ queues overflowing or packet TTLs expiring.

6. Do the legacy NEs affect the QoS provided by contemporary NEs?

The results indicate that the legacy NEs do not affect the QoS experienced by the contemporary NEs . Basing the Q-PEP solution on the DiffServ model, the legacy NEs have no need to communicate with the contemporary NEs for QoS provision. Data generated by the legacy NEs which have their destination on the contemporary networks are handled in the same way as contemporary NE traffic is handled which is based on the PDP policies. However, the legacy NEs do add extra load on networks which could result in non-prioritised network traffic experiencing a degradation in QoS. Adding additional legacy NEs would equate to adding extra contemporary NEs to the network with respect to the additional network traffic (load) experienced by the network.

8.2 Observations

8.2.1 *P-PEP*

The idea of using SNMP for admission control within a PBNM paradigm (P-PEP) was proven unsuccessful due to SNMP's limitations as stated in Section 7.3. The idea of using SNMP for PBNM is, however, not far fetched. According to the SNMPCONF Working Group [41], SNMP can be used successfully to develop configuration management systems for a broad range of devices and networks. Boros [4] argues that SNMP, being the ubiquitous acceptable network management protocol, is the future of policy provisioning for QoS delivery. He then claims that the flaws with SNMPv2 will be solved and that the newer versions of SNMP will be backward compatible with the earlier versions and thus support the older NEs.

Even though SNMP has improved and remains the standard at the heart of network management, it has not been the panacea it once promised to be. SNMPv3 finally includes security and bulk transfer, but its implementation remains inconsistent. Various systems offer SNMP MIBs as checklist items, but they have not made SNMP support a strategic management methodology. PBNM with SNMP will only be realised in the future, and it remains to be seen whether it will be available for legacy NEs.

8.2.2 *Q-PEP*

From the network congestive testing (refer to Table 7.37), it was observed that the Q-PEP can provide some level of QoS for voice services for the receiving legacy NEs when activated. This proves our hypothesis true as stated in Section 3.2. Comparing these results with the results to when the Q-PEP was deactivated, the jitter by 77.86μsecs and ensured that no voice packets were lost compared to the 7.99 percent packet loss experienced when the Q-PEP was deactivated. However, the latency increased by 123.62μsecs when the Q-PEP was activated. An increase in packet loss was also experienced for IP services that were not prioritised such as HTTP and FTP (refer to Table 7.37). These occurrences, however, were expected through the literature review (refer to Section 2.2.4.6) and were expected results.

Non-congestive testing (refer to Table 7.23), revealed that the latency experienced during normal periods was elevated by the Q-PEP , as envisaged (refer to Section 2.2.46). However, it was not envisaged that the Q-PEP, when activated, would increase the packet loss for IP services FTP, HTTP and Telnet. This can, however, be accounted for due to the CBQ queues for these services overflowing or the packet TTLs expiring while the voice services packets were being serviced, resulting in packet loss.

From Table 7.37, it is clear that varying the bandwidth allocated for voice services affected the QoS experienced. By over-provisioning the CBQ queue, the best possible QoS was experienced. However, this results in less bandwidth available for other IP services and a degradation in QoS is then experienced by them. Thus, a direct relation can be seen, by prioritising voice services and over-provisioning the CBQ queue, it should be expected

that other IP services suffer QoS degradation even under normal conditions. Thus careful network planning and design is needed when deploying traffic shaping as it can have disastrous effects on other IP services if enough bandwidth is not allocated for a specified service.

From the data collected in Table 7.37, it was made clear that a major factor in QoS delivery is network design, i.e. allocating the correct amount of bandwidth per service so that the service receives the QoS it needs, to operate in an acceptable manner and in doing so minimally affecting the other services. A badly designed network filled with QoS enabled NEs could have repercussions due to the design that would actually cause QoS delivery to be worse than when no QoS mechanisms were applied to the network. The results indicate that careful network design and deployment could alleviate QoS issues to an extent without additional QoS protocols - RSVP, MPLS and COPS.

One must remember that QoS does not create bandwidth, but merely provides prioritisation for IP services, which results in QoS provision. However, situations exist where the available bandwidth of a network cannot support the service when it is operating by itself, i.e. if a network only deploys voice services. A solution to this is to upgrade the bandwidth capabilities of the networking media or restrict the use of the service to certain network users. This is the case for FTP services (refer to Table 7.37) when the bandwidth allocated was 65 percent, the packet loss experienced was still 76.78 percent.

It was observed from the results that the PSQM rating was 1 for all network testing environments, which is high, as 0 is the best result obtainable (refer to Section 2.2.3.3). This proves that although the legacy NEs are old they do not add a considerable amount of line noise, if any, to affect the voice quality.

8.3 Suggestions for QoS Provision

Voice packets can be dropped if the network quality is poor, if the network is congested, or if there is too much variable delay in the network. Poor network quality can lead to

sessions frequently going out of service due to lost physical or logical connections. Thus, a network for voice services should be designed and implemented so that the physical and logical network follows sound design methodologies for stability and network quality. The significance of this is that the level of QoS provided can be raised by careful network design and deployment.

Cisco recommends a six step approach designing and deploying a multiservice infrastructure. Their approach begins with an evaluation of the current network, then sets objectives and goals, evaluates available technologies, provides technical design considerations for engineering a network for the unique characteristics of real-time communications and ends of with a financial analysis and guidelines to plan the deployment of the network [11].

8.4 Further Research

Congestion algorithms have been included in TCP/IP stacks for NEs, to avoid or detect network congestion. These algorithms, however, lower the network traffic throughput of all network traffic packets between source and sender when congestion occurs. Applying a prioritisation mechanism to congestion algorithms allowing it to be predetermined which network traffic flows will suffer throughput loss could possibly alleviate the QoS for these prioritised services. In this way, congestion algorithms would be able to help QoS provision, by either reducing the window size or dropping the packets of non-prioritised network traffic. But this requires further research and much is to be learned.

Another idea is to build intelligence into network applications so that network traffic flows based on application needs can be set up based on current network usage. The solution should follow an all or nothing paradigm i.e. if there is enough bandwidth available, the QoS delivered results in 5 9s service level if there's no bandwidth available or not enough, the service should be denied. This results in a framework in which QoS provision is high for network services or denied. What will be interesting in this research is the user aspect, i.e. whether users would prefer lower QoS or only be allowed to use the service when it is available and receive the telecommunications acclaimed 5 9s level of QoS.

Another idea surrounds better bandwidth utilisation for the LAN by using a compression mechanism. Better bandwidth utilisation should extend the congestion point of a network and thus resulting in better QoS experienced. However, the trade off is that a compression tool induces network delay which reduces the QoS experienced by real-time services. It is thus questionable whether or not the QoS experienced by IP services increase or decrease due to the added delay when compression tools are introduced.

8.5 Limitations

The Q-PEP solution could possibly deliver better results by implementing congestion algorithms. This research indicates that applying congestion algorithms to non-real time TCP based IP services could alleviate the network congestion and allow real-time services to benefit. From Section 2.1.2.2, it was learned that TCP retransmits lost packets, thus, losing TCP packets is not a momentous affair, but losing UDP packets is. This is due to UDP providing a connectionless service and not retransmiting packets. However, due to the limitations of the SMB, the effect of dropping TCP packets to the overall network congestion and how the TCP services are affected, cannot be tested. The SMB merely interprets all the packets as being lost. The IETF, however, are of the opinion that inappropriate resets of TCP connections, which congestion algorithms do, will increase network congestion in the end [24].

The queuing mechanism employed ALTQ is static. By this it is meant that once the queuing mechanism is set up it cannot be changed, unless the queuing mechanism is deactivated, the changes made and then reinitialised. A dynamic solution would be more "intelligent" than the all or nothing as suggested earlier and would better bandwidth utilisation.

The Q-PEP can only provide QoS for voice traffic that transits the Q-PEP itself (refer to Figure 6.1), i.e. for voice services that have their source in the legacy network and destination in the contemporary networks or vice versa. QoS will be provided for voice services that have their source and destination within the contemporary network by the contemporary NEs themselves through their QoS protocols. However, QoS will not be

provided for voice services that have their source and destination within the legacy network as the legacy NEs do not have the necessary QoS protocols.

8.6 Relevance

The daily development of new technologies in the Information Technology (IT) field is resulting in bandwidth requirements increasing and applications becoming more bandwidth hungry. Underprivileged schools and universities cannot afford to upgrade their infrastructures to use these applications. However with the Q-PEP solution, it is possible to prioritise their infrastructure to cater for these applications and use the available bandwidth, so that the required QoS for these applications can be provided and can be utilised.

A focus of the Centre of Excellence (CoE) at the University of the Western Cape (UWC) is, providing distance education to disadvantaged people in the rural areas of South Africa (S.A.). There is a lack of educational facilities in S.A. but with the integration of the PSTN with IP networks, distance learning (DL) centres can now reach former unattainable areas through on-line Internet services. The proposed Q-PEP solution will aid disadvantaged educational centres by providing user acceptable DL services with a legacy based networking infrastructure.

8.7 Summary of the Chapter 8

It is clear from the results that Traffic Shaping in the form of queuing within a PBNM paradigm can alleviate QoS issues for legacy NEs to an extent. However, when prioritising a service as, in our case voice services, it should be expected that non-prioritised IP services suffer QoS degradation. Whether this is acceptable, depends on the network users. This research has revealed that there are some forgotten networking aspects in existence that can be modified and used for QoS provision in legacy networks that provide access to services in broadband arenas.

BIBLIOGRAPHY

[1]. Aurrecoechea, C. Cambell, A. & Hauw, L. (March 1996) *A survey of QoS architectures.* IFIP International Conference, Paris, France. Available online http://citeseer.nj.nec.com/aurrecoechea98survey.html

[2]. Baker, F. Black, D. Blake, S. & Nichols, K. (1998). *Request for Comments: 2474 Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers.* Network Working Group. Available online at http://www.ietf.org/rfc/rfc2474.txt

[3]. Blake, S. & Black, D. & Carlson, M. & Wang, Z. & Weiss, W. (December 1998) *Request for Comments: 2475 An Architecture for Differentiated Services.* Available online at http://www.ietf.org/rfc/rfc2475.txt

[4]. Boros, S. (2001). *Policy Based Network Management with SNMP.* Research Paper. Computer Science Department, University of Twente, The Netherlands.

[5]. Braden, R. & Zhang, L. & Berson, S. & Herzog, S. & Jamin, S. (September 1997). Request for Comments: 2205. Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification. Available Online at http://www.ietf.org/rfc/rfc2205.txt

[6]. Bradner, S. (July 1991). *Request for Comments: 1242 Benchmarking Terminology for Network Interconnection Devices.* Havard University Available online at http://www.ietf.org/rfc/rfc1242.txt

[7]. Campbell, A. Coulson, G. & Hutchison, D. (April 1994). *A Quality of Service Architecture.* Research Paper. ACM Computer Communications Review. Available online at http://citeseer.nj.nec.com/campbell96quality.html

[8]. Case, J. & Fedor, M. & Davin, J & Schoffstall,*M.* (May 1990). *Request For Comments 1157: A Simple Network Management Protocol.* Available online at ftp://ftp.sri.ucl.ac.be/pub/rfc/rfc1157.txt

[9]. Cho, K. (June,1999) *Managing Traffic with ALTQ.* USENIX 1998 Annual Technical Conference, Communication Standards Information Service IEEE 1999.

[10]. Christiansen, M. Jeffay, K. Ott, D. & Smith, F. D. (August 2000). *Tuning RED for web traffic.* In ACM SIGCOMM2000. Available online at http://citeseer.nj.nec.com/christiansen00tuning.html

[11]. Cisco Systems. (2001). *Cisco Architecture for Voice, Video and Integrated Data - The Architecture for E-Business.* White Paper. Available online at http://www.cisco.com/warp/public/cc/so/cuso/epso/avpnpg/vvaid_wp.htm

[12]. Clark, D. & Braden, R. & Shenker, S. (June 1994). *Request for Comments: 1633 Integrated Services in the Internet Architecture: an Overview.* Available online at http://www.ietf.org/rfc/rfc1633.txt

[13]. Comer, D.E. (2000). *Internetworking with TCP/IP Principles, Protocols, and Architectures.* Prentice Hall. Pages 56-57, 97-107 ISBN:0-13-018380-6.

[14]. Communication Standards Information Service IEEE. June 1999. *Voice over IP Technology.* Research Paper. Available online at http://www.comsoc.org.mx/standard/voip.html

[15]. Cratty, M. (1998) *The foundations of research meaning and perspective in the research process.* Allen & Unuch, Pages 18 - 23.

[16]. Degermark, M. (1997). *Advance Reservations for Predictive Service in the Internet.* ACM Springer Journal of Multimedia Systems, Available online at http://citeseer.nj.nec.com/degermark97advance.html

[17]. Durham, D & Boyle, J. & Cohen, R & Herzog, S & Rajan, R & Sastry, A. (Januaury, 2000). *Network Working Group : Request for Comments: 2748, The COPS (Common Open Policy Service) Protocol.* Available online at www.ietf.org/rfc/rfc2748.txt

[18]. Elwalid, A. & Mitra, D. (1997). *Traffic shaping at a network node: Theory, optimum design, admission control.* Proceedings IEEE INFOCOM'97.

[19]. ETSI ETR 003. (October 1994) *Network Aspects, General Aspects of Quality of Service and Network Performance.* ETSI Technical Report, Second Edition.

[20]. Feng, W. Dilip D. Kandlur, Saha, D. & Shin, K.G. (October 1998). *Adaptive Packet Marking for Providing Differentiated Services in the Internet.* Research Paper. Proceedings of International Conference on Network Protocols. Available online at http://citeseer.nj.nec.com/feng98adaptive.html

[21]. Feng, W.C. (1999). *Improving Internet congestion control and queue management algorithms.* PhD thesis, University of Michigan, USA, Available online at http://citeseer.nj.nec.com/feng99improving.html

[22]. Ferguson, P & Huston, G. (1998). *Quality of Service: Delivering QoS on the Internet and in Computer Networks.* John Wiley and sons, Inc. ISBN: 0-471-24358-2

[23]. Fertuck, L. (1992). *System Analysis and Design. With Case Tools.* Wm. C. Brown Publishers. ISBN: 0-697-12069-4

[24]. Floyd, S (June, 2001) *Internet Draft Inappropriate TCP Resets Considered Harmful.* Internet Draft. Available online at http://www.aciri.org/floyd/papers/draft-floyd-tcp-reset-01.txt

[25]. Floyd, S. & Jacobson, V. (August 1995.). *Link-sharing and Resource Management Models for Packet Networks* IEEE/ACM Transactions on Networking, Vol. 3 No. 4, pages 365-386. Available online at http://www-nrg.ee.lbl.gov/floyd/cbq.html

[26]. Floyd, S. (October 1994) *TCP and Explicit Congestion Notification*, ACM Computer Communication Review, V.24 N.5, pages 10-23. Available online at http://citeseer.nj.nec.com/floyd94tcp.html

[27]. Frye, R & Levi, D & Routhier, S & Wijnen,B. (March 2000) *Request for Comments: 2576, Coexistence between Version 1, Version 2, and Version 3 of the Internet-standard Network Management Framework*. Available online at http://www.ietf.org/rfc/rfc2576.txt

[28]. Gai, S. Strassner, J. Durham, D. Herzog, S. Mahon, H. & Reichmeyer, F. (February 1999.). *QoS Policy Framework Architecture*. Internet-Draft. Available online at ftp://ftp.ietf.org/internet-drafts/draft-sgai-policy-framework-00.txt

[29]. Golden Gateway. (2000). *Voice over Packet White Paper*. White Paper. Telogy Networks, Incorporate.

[30]. Grampín, E. Rubio, J. Vardalachos, N. Galis, A. & Serrat, J. (October 2001) *Implementation Issues of Policy Based Network Management Systems*. Design of Reliable Communication Networks (DRCN). Available online at http://emerge.ttt.bme.hu/PAPERS/83.PDF

[31]. Information Sciences Institute University of Southern California. (1981). *Internet Protocol DARPA Internet Program Protocol Specification*. Internet Draft. Available online at http://www.ietf.org/rfc/rfc0791.txt

[32]. International Telecommunications Union (February 1998) *P.861. Objective quality measurement of telephone-band (300-3400 Hz) speech codecs.*

[33]. International Telecommunications Union. (August 1996) *P.800. Methods for subjective determination of transmission quality.*

[34]. Jamin, S. Danzig, P. Shenker, S. & Zhang, L. (Dec. 1996). *A Measurement-based Admission Control Algorithm for Integrated Services Packet Networks*. IEEE/ACM Transactions on Networking. Available online at http://www.aciri.org/floyd/admit.html

[35]. Jon C.R. Bennett & H. Zhang. (August 1996) *Hierarchical Packet Fair Queueing Algorithms*. Research Paper. SIGCOMM'96 Proceedings. Available online at http://www.aciri.org/floyd/cbq.html

[36]. Karve, A. (1998). *IP Quality of Service*. Network Magazine. Magazine Article. Available online at http://www.networkmagazine.com/article/NMG20000727S0024/2

[37]. Kostas, T.J. Borella, M.S. Sidhu, I. Schuster, G.M. Grabiec, J. & Mahler, J. (February 1998). *Real-Time Voice Over Packet-Switched Networks.* IEEE Network.Conference Available online at http://citeseer.nj.nec.com/kostas98realtime.html

[38]. Kulathumani, V. (1999). *Voice over IP: Products, Services and Issues.* Ohio State University. Available online at ftp://ftp.netlab.ohio-state.edu/pub/jain/courses/cis788-99/voip products/index.html

[39]. Lorenz, D.H. & Orda, A.. (1998). *QoS Routing in Networks with Uncertain Parameters.* ACM S 1063-6692(98)09580-6

[40]. Ma, T and Bingxin, S. (2000). *Bringing Quality Control to IP QoS*. Network Magazine. Available online at http://www.networkmagazine.com/article/NMG20001103S0009

[41]. MacFaden, M. & Saperia, J. & Tackabury, W. (September 4, 2001). *IETF Draft : Configuring Networks and Devices with SNMP*. SNMPCONF Working Group. Available online at http://www.ietf.org/internet-drafts/draft-ietf-snmpconf-bcp-06.txt

[42]. McDysan, D. (2000). *QoS & Traffic Management in IP & ATM Networks*. McGraw-Hill. ISBN: 0-07-134959-6

[43]. McDysan, D.E. & Spohn, D.L. (1998). *ATM Theory and Applications.* McGraw-Hill Series. Pages 123-129. ISBN 0-07-045346-2.

[44]. Ohio State University. (2001). *Voice over IP: Protocols and Standards.* White Paper. TradeSpeak. Available online at http://www.tradespeak.com/htmldocs/1859.html

[45]. OpenView Network Management Division Hewlett-Packard Company. (September 1999). *A primer on Policy-based Network Management.* Hewlett-Packard.

[46]. Perkins,D. & McGinnis, E. (1997) *Understanding SNMP MIBs.* Prentice Hall. ISBN: 0-13-437708-7.

[47]. Rix, A.W. Beerends, J.G. Hollier, M.P. & Hekstra, A.P. (May 2001) *Perceptual evaluation of Speech Quality (PESQ) – a new method for speech quality assessment of telephone networks and codecs.* IEEE Signal Processing Society International Conference on Acoustics, Speech, and Signal Processing (ICASSP).

[48]. Rose M.T. (1996). *The Simple Book.* Prentice Hall, Inc Upper Saddle River, NJ 07458. ISBN 0-13-451659-1

[49]. Ruz, R. *Quality of service guaranteed in virtual circuit switched network.* IEEE Journal on Selected Areas in Communications, 13(6):1048--1056, August 1995.

[50]. Schulzrinne, H. Casner, S. Frederick, R. & Jacobson, V. (January 1996). *RFC 1889: RTP: A Transport Protocol for real-time application.* Available online at http://www.ietf.org/rfc/rfc1889.txt

[51]. Schwantag, U. (February 1997). *An Analysis of the Applicability of RSVP*. Diploma Thesis. University of Oregon and Institute of Telematics. Available online at http://ns.uoregon.edu/ursula/thesis/thesis.htm

[52]. Seddigh, N. Nandy, B. & Pieda, P. (December 1999). *Bandwidth Assurance Issues for TCP Flows in a Differentiated Services Network*. Proceedings of the IEEE GLOBECOM, page 6. Available online at http://citeseer.nj.nec.com/seddigh99bandwidth.html

[53]. Stevens ,W.R. (1990). *UNIX Network Programming*. Prentice Hall Software Series, Englewood Cliffs:110.

[54]. The International Engineering Consortium. (2000). *Fundamentals of Telecommunications*. Web Proforum Tutorials. Available online at www.iec.org/online/tutorials

[55]. Walker, J & Hicks, J. (2001). *Using Chariot To Evaluate Data Networks For Voice Readiness*. NetIQ Corporation. Available online at www.netiq.com

[56]. Wright, D.J. (2001). *Voice over Packet Networks*. John Wiley & Sons, Ltd. ISBN: 0-471-49516-6

[57]. Wroclawski, J. (1997). *Request For Comments 2210* The Use of RSVP with IETF Integrated Services. Network Working Group. Available online at http://www.ietf.org/rfc/rfc2210.txt

[58]. Wroclawski, J. (September 1997). *Request For Comments: 2211 Specification of the Controlled-Load Network Element Service*. Available online at http://www.faqs.org/rfcs/rfc2211.html

[59]. Xiao, X. & Ni, L. M. (April 1999) *Internet QoS: A Big Picture*. Research Paper. IEEE Network Magazine. Page 23. Available online at http://www.cse.msu.edu/~xiaoxipe/download.html

[60]. Yuval, B. (2001). *Fine-Tuning Voice over Packet Services*. Business Development, RADCOM Ltd. Available online at http://www.protocols.com/papers/voip2.htm

[61]. Zhang, L. Deering, S. Estrin, D. Shenker, S. & Zappala, D. (September 1993). *RSVP: A New Resource ReSerVation Protocol*. Research Paper. IEEE Network. Available online at http://citeseer.nj.nec.com/zhang93rsvp.html