In-Band Flow Establishment for End-to-end QoS in Rapidly Deployable Radio Networks

By

Saravanan Radhakrishnan
B.Eng., Computer Science and Engineering, Anna University, India, 1997

Submitted to the Department of Electrical Engineering and Computer Science and the Faculty of the Graduate School of the University of Kansas in partial fulfillment of the requirements for the degree of Master of Science

Professor in Charge

Committee Members

Date Thesis Accepted

To my Parents

Radhakrishnan & Yesodha

Acknowledgements

I would like to thank Dr. Victor S. Frost, my advisor and committee chair for his encouragement throughout my study here in KU. His guidance and advice have been very valuable throughout the course of this work. I would also like to thank him for giving me an opportunity to work on the RDRN project. It certainly has been a tremendous learning experience and I am grateful to him for the same. I would also like to thank Dr. Joseph B. Evans and Dr. John Gauch for serving on my masters committee. I would also like to thank Dr. Joseph B. Evans specially for encouraging me in pursuing research in IP Quality of Service and for introducing me to the world of Cisco Routers.

I thank my colleagues on the project, especially Fazal. I enjoyed working with him. I also thank Ricardo and Fadi for their useful suggestions during the implementation of the Network Control Protocol.

I would also like to thank Deepak, Anu, Gowri, Rajashree, and Pramodh for making my stay at KU a memorable one. I would also like to thank Ananth, Aarti, Ram, Sachin, Deepak (ys), Dhananjaya Rao, Sampath, Sandeep, Ranjith, Thyagarajan and Anand Iyer for their support throughout my stay at KU.

Last but not the least, I would like to thank my parents for their infinite love and care, which kept me going in the most difficult of times, and my sister for her understanding, and the Almighty, for without him, nothing would be possible. I owe all my achievements to them.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Abstract

In this thesis, we present an end-to-end QoS provisioning mechanism for highly dynamic networking environments like Rapidly Deployable Radio Network (RDRN). A QoS architecture has been proposed, implemented and evaluated which provides the framework for the configuration, prediction and maintenance of the end-to-end QoS. As part of the architecture, a flow specification tailored for highly mobile networking environments has been proposed. The instability of wireless links and the mobility of the nodes influence the flow specification. A flow establishment scheme that uses in-band signaling to establish the flows has been proposed, implemented and evaluated. This approach is designed to make efficient use of the features of IP at the network level and ATM at the link level. In the event of non-availability of the requested QoS, the flow establishment scheme constantly attempts to scale up to the maximum requirements of the application, and establishes the flow when the resources become available. QoS is implemented by the introduction of the QoS layer in the RDRN protocol stack. The scheduling algorithm used at the QoS layer has been proposed, implemented and evaluated.

Chapter I

Introduction

Provisioning of Quality of Service (QoS) in mobile networks has been a topic of active research during the last few years. Before delving into the details, a short note on the definition of QoS is worth mentioning. QoS has no standard definition in the literature. Goujun Lu, in [1], presents the notion of QoS at three levels in the system:

1. User Level – perceptual quality
2. Application Level – processing and presentation of logical media data units
3. System Level – processing and transmission of packets

For example, “NTSC Video” is a user specification having application parameters of 30 frames per second with a resolution of 640 x 480, which maps onto system parameters consisting of a bit rate of 220 Mbits/sec, without compression.

In general, the user’s view of QoS should be expressed by parameters that are relevant to user-perceivable effects, independent of the network design and measurable. These provide input to the system design, but are not necessarily usable in developing network performance parameters. Network-specific parameters are more concerned with expressing performance that relate to planning, operation and maintenance of the network system. These parameters usually, need not be meaningful to the user. Thus, relating these views could be very challenging.

Some of the most important QoS parameters of interest are the throughput, the delay and the reliability. Throughput, the most prominent QoS parameter, specifies the amount of data per time unit (average or maximum) that can be transferred from the source to the destination. In general, it is not sufficient to specify the throughput in terms of bits per second, it should also describe the packetization, i.e., the average and the maximum packet size and the packet rate. Delay specifies the maximum or average delay observed by a data unit during an end-to-end transmission. Reliability pertains to the loss and corruption of data, which depends on the specified loss probability.

1.1 End-to-End QoS in wireless/mobile networks

QoS provisioning in wired networks has been the centerpiece of many research activities in integrated packet-switched systems. In such networks, all types of traffic are transported through common framing – for example, *Asynchronous Transfer Mode* (ATM) cells or *Internet Protocol* (IP) [2] packets. Since they share network resources like buffers, congestion can occur. Traffic characterization, call admission control, resource reservation and packet scheduling in such environments are all elements of a general QoS architecture, which has two main purposes [3]:

1. Avoid congestion
2. In case of congestion, react in such a way that results in elimination of congestion over some reasonable time.

In wireless and mobile networks, the QoS provisioning problem is even more challenging than in wired networks. This is due to the wireless channel fading and mobility [3], two aspects that have not been considered in QoS architectures defined for wired broadband networks. As a result, the fluctuations in available resources are much higher in wireless and mobile networks, when compared to wired networks. Let us consider each of these aspects and understand how they contribute to the highly dynamic resource availability.

In contrast to today's wired networks, wireless links suffer from high bit error rate (BER) and fading. This results in packet loss in the wireless medium, which in turn translates into packet delay and jitter. In general, there is a tradeoff between BER and the bandwidth, and a reduction in packet loss can translate into higher packet transmission delay and jitter.

Mobility and handoff is the other reason for fluctuation in the resource availability. As a mobile node roams and hands off from one access point to another, there is a change in the wireless resources. This change in resources can result in a lot of fluctuation in the resource availability. Thus, resource availability is one of the main issues to be addressed while designing an end-to-end framework for QoS support in mobile and wireless networks.

Before going into the issues regarding end-to-end QoS in RDRN, let us briefly discuss the features of the RDRN system.

1.2 RDRN – An Overview

Rapidly Deployable Radio Networks (RDRN) [4,5] is a highly dynamic mobile network and it consists of portable (mobile) communication nodes which can be deployed on the ground or on mobile platforms such as trucks, helicopters or fixed wing aircrafts. When deployed, the nodes use Global Positioning System (GPS) derived location information to automatically configure themselves into a high capacity, fault tolerant network infrastructure.

RDRN consists of two overlaid radio networks. They are:

1. The low power, low bandwidth, omni-directional network called the *orderwire* network, which is used for location dissemination, topology configuration and link setup management among the RDRN nodes. This network is a collection of 19.2 Kbps Packet Radio systems that run the X.25 protocol to exchange this control information.
2. The high capacity, highly directional, multiple beams link that is used for data transfer. This features point-to-point links between the different RDRN nodes.

A network control protocol (NCP) runs over the orderwire network and manages the setting up and tearing down of the high-speed links. The NCP receives the current location of the node from the GPS receiver and disseminates this information to the nearby nodes over the orderwire network. It uses a topology configuration algorithm to determine the nodes to which a high-speed link needs to be set up.

There are two types of nodes in the RDRN system:

1. Mobile Access Point: The Mobile Access Point (MAP) is a node that provides connectivity to other MAPs and Mobile Nodes (MNs). The MAP is capable of doing both the layer 3 routing and the layer 2 switching. Another feature of the MAP is its capability to inter-operate with a wired ATM network through an ATM switch.

2. Mobile Node: A Mobile Node (MN) is a typical end user node that connects to the network through a MAP. The MN is capable of neither routing nor switching and relies on the MAP for connecting to the network. The high level RDRN architecture is shown in Figure 1.1.

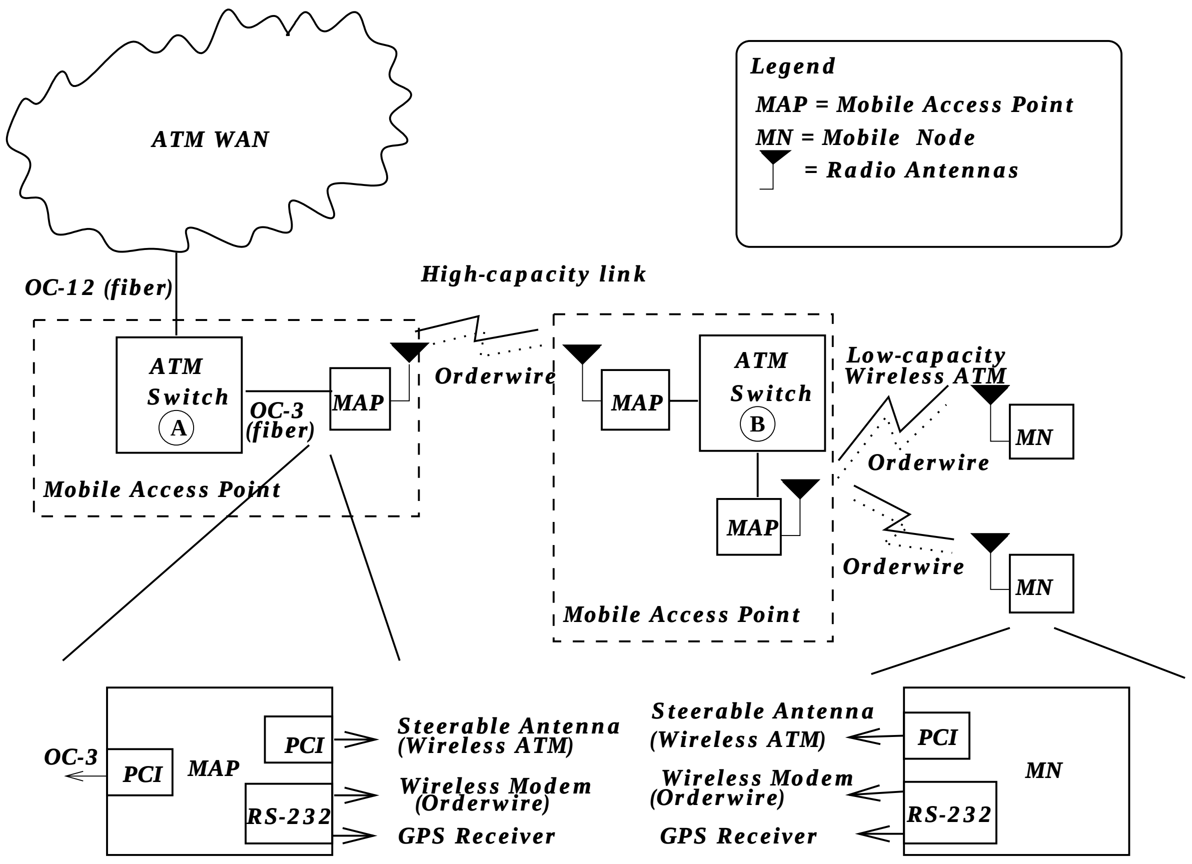

Figure 1.1: RDRN High Level Architecture

Each MAP has an interface for the high-speed antenna, and serial interfaces to the GPS receiver and the packet radios. In addition, a MAP may have an OC-3 interface if it needs to connect to the wired world. Each MN has an interface for the high-speed antenna and a serial interface to the GPS receiver and packet radios.

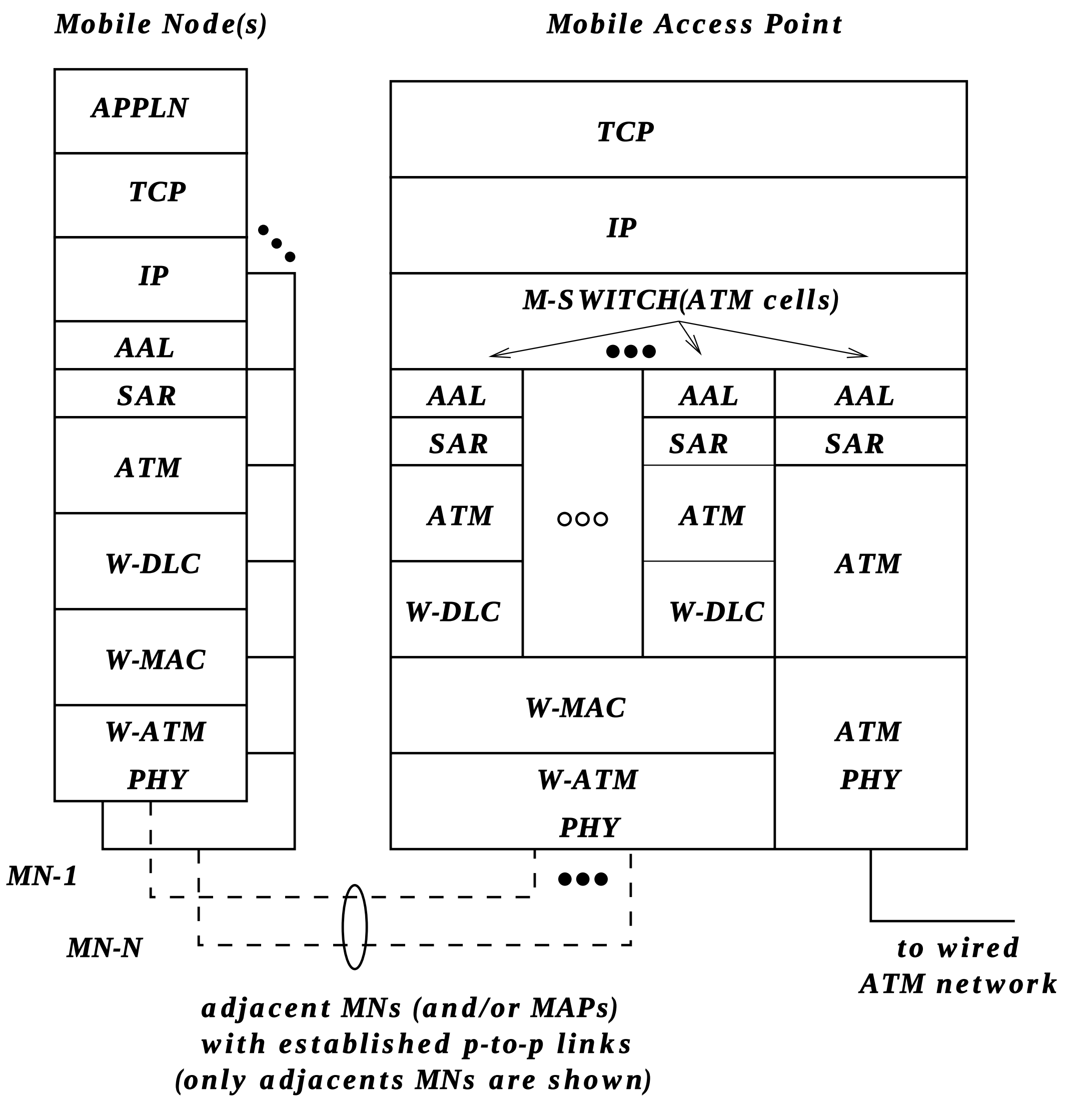

Figure 1.2: RDRN Protocol Stack

The RDRN protocol stack is shown in Figure 1.2. The protocol stack at the MAPs consists of a software ATM switch, which also acts as a router. The switch sends the cells to the

ATM Adaptation Layer (AAL) layer. The AAL layer adds the header and trailer, and sends the AAL-Protocol Data Unit (PDU) to the (Segmentation and Re-assembly) SAR layer. The SAR layer segments the AAL-PDUs and sends 53-bytes cells to the Wireless Data Link Control Layer (W-DLC). The W-DLC layer puts a fixed number of cells together in a frame, adds the Cyclic Redundancy Check (CRC) and sends it to the W-Medium Access Control, which schedules the transmission of the bits over the high speed radio. On the MAPs, there is an RDRN protocol stack for each MN and MAP. There may be multiple MNs and MAPs on each radio beam from the MAP. The MAP also has a different protocol stack for the wired networks. This stack does not contain the W-DLC and the W-MAC layers. The protocol stack at the MN is similar to that present at the MAP, except that there is only one high-speed interface, and it connects to the RDRN system via this interface.

The RDRN system is different from most mobile ad-hoc networks and conventional wireless ATM networks. In mobile ad-hoc networks, there is absolutely no hierarchy among the nodes. The RDRN system, however, has a pseudo hierarchy that is imposed by ATM in the protocol stack. In conventional wireless ATM networks, only the last hop is considered to a wireless hop. The base stations are connected to a wired network. However, in RDRN, multiple wireless hops are supported. The prototype of the RDRN system is described in detail in [4]. The implementation aspects and the experiences in the project have been discussed in [5].

1.3 End-to-end QoS Issues in RDRN

QoS is inherently an end-to-end requirement. The user sees a perceivable difference between the various services offered only if QoS is provided from the source to the destination. Provisioning of end-to-end QoS in a highly dynamic RDRN environment involves a number of issues including:

1. The RDRN system consists of heterogeneous links, since the MAPs may attach to the wired world as well as to other MAPs and MNs. As a result, the issue of providing end-to-end quality of service is complex. The flow specification, which will be used to characterize the quality of service required and the traffic pattern that will be expected,

will have to reflect the required characteristics of the various types of links in the system.

2. The nodes in an RDRN system are highly mobile as a result of which the path between the source and the destination changes frequently. The new path may or may not have the requested resources. Therefore, a method to re-negotiate the QoS parameters and adapt to the available resource becomes a necessity.
3. The other important issue in an RDRN system is that of QoS mapping. The QoS requirements from the application need to be mapped at each level in the protocol stack. QoS in the Internet Protocol (IP) is a topic of active research, and many approaches have been suggested for the same. The integrated services [6] and the differentiated services architecture [7] are two such approaches for the provisioning of QoS in IP. The mapping of the QoS characteristics from the IP layer to ATM layer is yet another complex issue.
4. Flow establishment is another issue concerning the heterogeneous nature of the network. A flow establishment scheme that establishes the flows from the source to the destination needs to be designed.

1.4 The contributions of this thesis

The RDRN protocol stack consists of IP over ATM. Use of ATM signaling to set up an end-to-end connection is not suitable for a dynamic environment like RDRN, where the links between the nodes are highly unstable. The resources available in the system are also less compared to traditional wired networks. A link failure may result in a significant amount of signaling overhead to re-establish the end-to-end connection, which is not desirable. This thesis proposes, implements and evaluates an in-band flow establishment protocol, tailored for a highly dynamic IP over ATM mobile network. Flow establishment is the process of setting up an end-to-end flow. The flow establishment is done along with the transfer of data, i.e., there is no explicit out-of-band signaling. It also uses the features of both IP and ATM in setting up an end-to-end flow. IP is a robust network layer protocol, which makes it suitable for handling changes in routes because of the dynamic nature of the system. ATM, on the other hand, provides high speed switching capabilities.

This thesis also proposes, implements and evaluates a QoS architecture for the RDRN system. The proposed QoS architecture defines the framework that will be used to specify and implement the required performance characteristics of applications running on a RDRN network. A QoS layer that schedules the transmission of the cells in the source and the intermediate MAPs has been introduced into the RDRN protocol stack. The design of this layer has been proposed, implemented and evaluated. A flow specification that is tailored for a highly dynamic networking environment like RDRN has also been proposed. Flow Specification is a data structure that is used by the inter-network hosts to request resources from the inter-network. There are essentially two components in the flow specification:

1. The QoS that is requested by the host, which identifies the resources that the host requires from the network.
2. The traffic specification, which identifies the profile of the traffic generated by the host.

1.5 Lessons Learned

This section discusses the lessons that were learned during the implementation of the proposed QoS architecture for RDRN. During the implementation process, the flow establishment block in the proposed QoS architecture (discussed in Chapter 3) was implemented above the IP layer, and a QoS layer that schedules the transmission of cells was implemented below the SAR layer in the protocol stack. One of the major hurdles that were experienced during this phase was the halving of throughput when the software switch was used. It was then determined that for every cell switched by the software switch, two cells were being sent out at the physical level because of the overhead added by the AAL and DLC layers in the protocol stack. It is this per-cell overhead that caused the halving of the throughput. The problem was overcome with the introduction of the QoS layer in the protocol stack, which schedules the transmission of cells. The QoS layer groups together a set of cells based on the priority assigned to them and sends a train of cells down to the DLC layer, thereby avoiding the per-cell overhead.

Yet another problem that was faced during the implementation was the re-assembly of packets from cells, by a node whose outgoing link fails. When a switching table entry is deleted, the SAR layer was not capable of re-assembling the cells since the VCs were deleted. This problem was overcome with the implementation of an I/O system call that would delete the switching table entries corresponding to the link that had failed, but would keep the incoming VCs active. This ensured the correct re-assembly of packets from the cells. These were some of the lessons that were learned from the experience gained during the implementation.

1.6 Organization of thesis

The rest of the thesis is organized in the following manner. Chapter 2 discusses the related work, this includes a brief description of the existing QoS architectures, QoS provisioning mechanisms in mobile networks, end-to-end connection setup mechanisms and QoS requirements in mobile tactical applications. Chapter 3 discusses the proposed flow establishment protocol, the proposed QoS architecture for RDRN and the proposed flow specification for highly dynamic networking environments. It also discusses the features of the scheduler that has been designed for QoS provisioning in RDRN. Chapter 4 presents the implementation issues in the thesis. This includes the various algorithms that have been used in the implementation. Chapter 5 demonstrates the set of tests that were conducted to determine the validity and performance of the flow establishment protocol. Chapter 6 contains the conclusion and future work.

Chapter II

<u>Related Work</u>

This chapter discusses some of the previous work that has been done in the area of Quality of Service in wired and wireless networks. It includes a brief description of some of the QoS architectures that have been proposed for the Internet and for mobile networks, end-to-end connection setup mechanisms and QoS requirements in mobile tactical applications.

2.1 QoS Architectures

Until recently, research in providing QoS guarantees has mainly focussed on connection oriented traffic models and service scheduling disciplines. Guarantees, like the delay, throughput, jitter etc. are not, however, end-to-end in nature. Rather, they preserve QoS guarantees only between network access points to which the end-systems connect. Work on QoS-driven end-system architecture needs to be integrated with the network configurable QoS services and protocols to meet application-to-application requirements. In recognition of this, researchers have recently proposed new communication architectures, which are broader in scope and cover both the network and end-system domains. In this section, we will review a number of proposals, which have emerged in the literature.

2.1.1 Heidelberg QoS Model

The Heidelberg project [8] at IBM’s European Networking Center in Heidelberg has developed a comprehensive QoS model, which provides guarantees in the end-system and network. The communications architecture includes a continuous media transport system [9], which provides QoS mapping and media scaling [10]. The model of the architecture is shown in Figure 2.1. Underlying the transport is an internetworking layer, which supports both guaranteed and statistical levels of service. In addition, the network supports QoS-based routing (via a QoS finder algorithm) and QoS filtering. Key to providing end-to-end guarantees is *HieRAT* [8](resource administration technique). HieRAT contains a comprehensive QoS management scheme, which includes QoS negotiation, QoS calculation, admission control, QoS enforcement and resource scheduling. The HieRAT operating system scheduling policy is a rate-monotonic scheme whereby the priority of a system thread performing protocol processing is proportional to the message rate requested.

The Heidelberg model has been designed to handle heterogeneous QoS demands from the individual receivers in a multicast group and to support QoS adaptivity via flow filtering and media scaling techniques. Media scaling [10] and codec translation at the end systems, and flow filtering and resource sharing in the network are fundamental to meeting heterogeneous QoS demands. Flow filtering is the process of identifying flows based on certain fields (e.g. destination address) in the packet. Media scaling matches the source with the receivers' QoS capability by manipulating flows at the core of the network as flows traverse bridges, switches and routers.

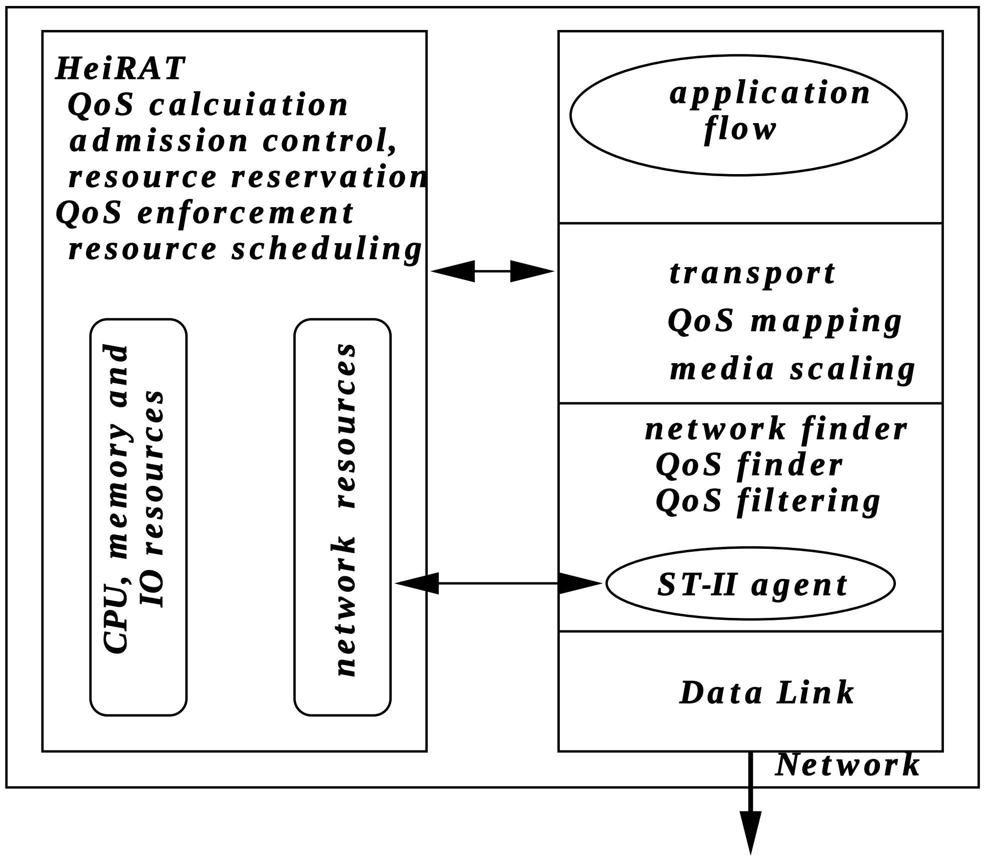

Figure 2.1: Heidelberg QoS Model

2.1.2 XRM (Extended Reference Model)

The COMET group at Columbia University has developed an Extended Integrated Reference Model [11] (Extended Reference Model) as a modeling framework for control and management of multimedia telecommunications networks (which comprise multimedia computing platforms and broadband networks). The COMET groups argues that the foundations for interoperability (control and management) of multimedia computing and networking devices are equivalent; that is, both classes of devices can be modeled as producers, consumers and processors of media. The only difference is the overall goal that a group of devices has set to achieve in the network or end-system. The XRM is divided into five distinct planes [12]:

1. Management plane, which resides in the network management plane (N-plane) and covers the Open System Interconnect (OSI) functional areas of network and system management.
2. Traffic control function, which comprises the resource control (M-plane) and the connection management and control (C-plane) planes. Resource control constitutes cell scheduling, call admission, call routing in the network, process scheduling, memory management, routing, admission control and flow control in the end-systems.
3. Information transport function, which is located in the user transport plane (U-plane), models the media protocols and entities for the transport of user information in the network and the end-systems.
4. Telebase, which resides in the data abstraction and management plane (D-plane) and collectively represents the information, data abstractions existing in the network and end-systems. The telebase implements data sharing among all other XRM planes.

The XRM is built on theoretical work of guaranteeing QoS requirements in ATM networks and end-systems populated with multimedia devices. General concepts for characterizing the capacity of network [13] and end-system [14] devices (e.g. disks, switches etc.) have been developed. At the network layer, XRM characterizes the capacity region of an ATM multiplexer with QoS guarantees as a schedulable region. Network resources such as switching bandwidth and link capacity are allocated based on four cell-level traffic classes (class I, II, III and C) for circuit emulation, voice and video, data, and network management respectively. A traffic class is characterized by its statistical properties and QoS requirements. Typically QoS requirements of cell level, scheduling and buffer management algorithms dynamically allocate communication bandwidth and buffer space appropriately.

In the end-system, flow requirements are modeled through service class specifications with QoS constraints. For example, in the audio video unit, the service class specification is in terms of JPEG, MPEG-I, MEG-II video and CD audio quality flows with QoS guarantees. QoS for these classes is specified by the set if frame delay and loss constraints. The methodology of characterizing network resources is extended to the end-systems to represent the capacity of multimedia devices. Using the concept of a multimedia capacity region, the problem of scheduling flows in the end-systems becomes identical to the real-time bin packing exercise of the network layer.

2.1.3 Integrated Services Architecture

The Integrated Services architecture [6] is a significant contribution by the Internet Engineering Task Force (IETF) towards providing controlled QoS for multimedia applications over an integrated services internetwork. The Integrated Service (int-serv, as it is widely known) working group in the IETF has defined a comprehensive int-serv architecture [6] and a QoS framework [15] used to specify the functionality of internetwork system elements, which make multiple, dynamically selectable QoS levels available to applications. The behavior of the elements, which constitute routers, subnetworks and end-point operating systems, is captured as a set of services, of which some or all are offered by each element. Each element is QoS-aware and supports interfaces required by the service definition. The concatenation of service elements along an end-to-end data path provides an overall statement of end-to-end QoS. The following int-serv architectures are offered in addition to best effort.

1. *Controlled delay* [16], which attempts to provide several levels of delay, which the application can choose from.
2. *Predicted delay*, which provides statistical delay bounds.
3. *Guaranteed Delay* [17], which provides an absolute guaranteed delay bound.

Flows in an int-serv architecture are characterized by two specifications: a traffic specification, which is a specification of the traffic pattern which a flow expects to exhibit, and a service request specification, which is a specification of the QoS a flow desires from a service element. The int-serv architecture, which is restricted to the network, but applicable to the end system too, is comprised of four components [6] (shown in Figure 2.2.):

1. A *packet scheduler*, which forwards packet streams using a set of queues and timers.
2. A *classifier*, which maps each incoming packet into a set of QoS classes.
3. An *admission controller*, which implements the admission control algorithm to determine whether a new flow can be admitted or not.
4. A *reservation setup protocol* (e.g. RSVP [18]), which is necessary to create and maintain flow-specific state in the routers along the path of the flow.

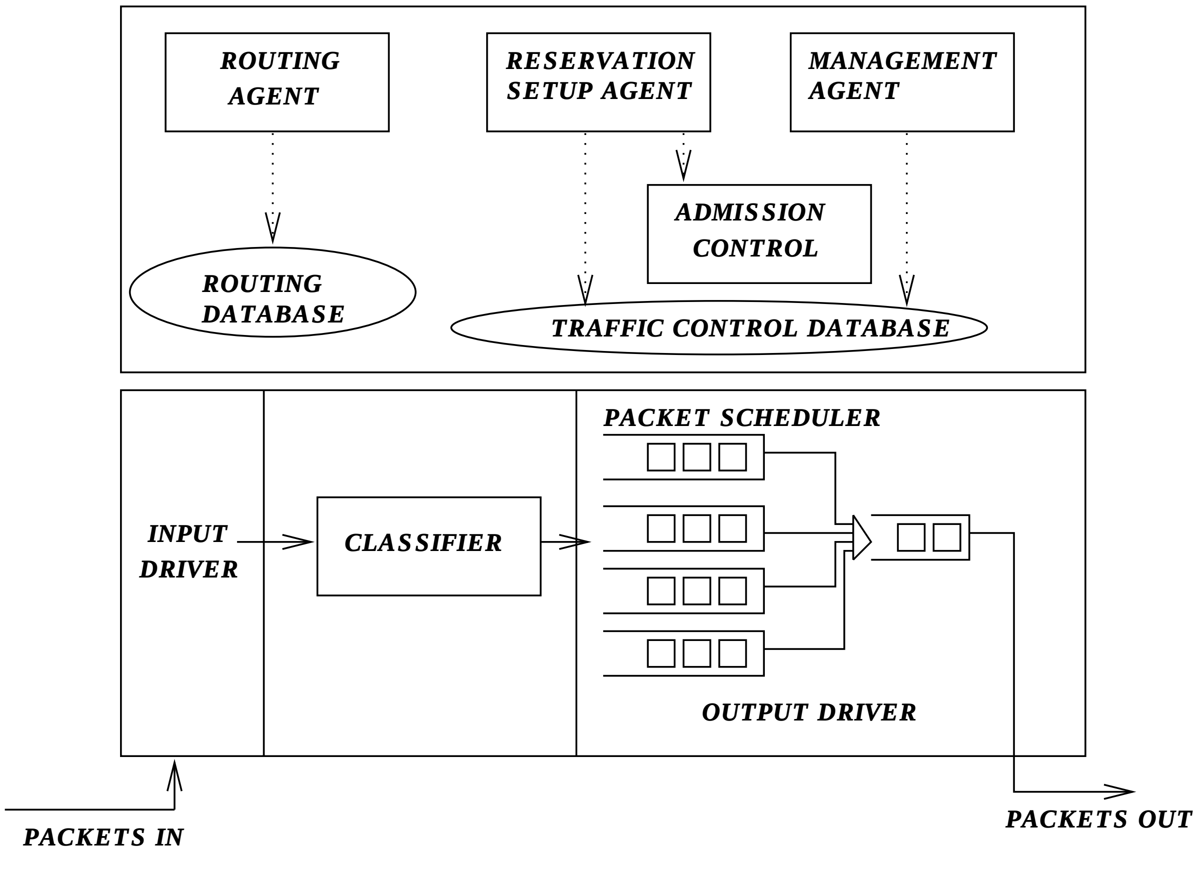

Figure 2.2: Integrated Services Architecture

Clark introduced a Quality of Service manager (QM) as part of the end system int-serv architecture. The QM presents as abstract management layer designed to isolate applications from underlying details of specific service provided in a QoS-driven internet. The introduction of the QoS manager ensures that the applications can negotiate the desired QoS without needing to know the details of a specific network service. The QM provides a degree of transparency, whereby applications express levels of QoS in application-oriented language rather using communication QoS specifics. The QM is responsible for determining what QoS management capabilities are available on the application's communication path, and chooses the path best suited to the application.

2.1.4 Differentiated Services Architecture

More recently, the IETF has been focussing on a different architecture [7] for the provisioning of QoS in the Internet. Differentiated Services (or diff-serv, as is it widely known) is a scalable means of QoS provisioning in the Internet. The differentiated services architecture is based on a simple model where traffic entering a network is classified and possibly conditioned at the boundaries of the network, and assigned to different behavior aggregates. Each behavior aggregate is identified by a single diff-serv (DS) codepoint (CP) [19]. A DS codepoint is specific value of the DSCP portion of the Type-of-service (TOS) byte in the IP header. Within the core of the network, packets are forwarded according to the per-hop behavior (PHB) associated with the DS codepoint.

A contiguous set of nodes that operate with a common set of service provisioning policies and PHB definitions form a diff-serv (DS) domain. Differentiated Services are extended beyond a DS domain boundary, by establishing service level agreements (SLA) between an upstream DS domain and downstream DS domain. The SLA may specify packet classification and re-marking rules and may also specify traffic profiles and actions to traffic streams, which are in- or out-of-profile. Based on the SLA, a traffic conditioning agreement is derived. Traffic Conditioning Agreement (TCA) is an agreement specifying classifier rules and any corresponding traffic profiles and metering, marking, discarding and/or shaping rules which are to apply to the traffic streams selected by the classifier. A TCA encompasses all of the traffic conditioning rules explicitly specified within a SLA along with all of the rules implicit from the relevant service requirements and/or from a DS domain's service provisioning policy.

The differentiated services architecture is shown in Figure 2.3. A classifier is used to select packets based on some fields in the packet header, e.g. the TOS byte. There are two types of classifiers: behavior aggregate (BA) classifiers and multi-flow classifiers (MFC). A BA classifier classifies packets based on the DS codepoint. A MFC classifies packets based on a number of fields like the source IP address, destination IP address, source port number, destination port number, protocol etc.

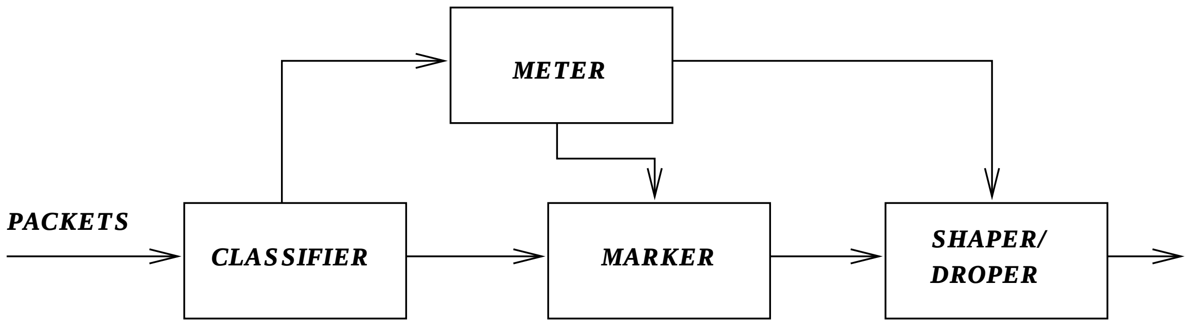

Figure 2.3: Differentiated Services Architecture

A traffic conditioner may contain the following elements: meter, marker, shaper, and dropper. A traffic stream is selected by a classifier, which steers the packets to a logical instance of a traffic conditioner. A meter is used (where appropriate) to measure the traffic stream against a traffic profile. The state of the meter with respect to a particular packet (e.g., whether it is in- or out-of-profile) may be used to affect a marking, dropping, or shaping action.

Based on this architecture, many services can be provided. The most common services proposed are:

1. *Premium Service*: This service [21] is meant for applications requiring *low delay* and *low jitter* services. The applications generate fixed peak bit-rate traffic. The SLA will specify the desired peak-rate for a specific flow or aggregation of flows. Any traffic that

exceeds the peak rate will be dropped. Premium service guarantees that the contracted bandwidth is available when traffic is sent. This service is suited for Internet Telephony, video conferencing and for creating *Virtual Leased Lines (VLL)* for *Virtual Private Networks (VPN)*.

2. *Assured Service*: This service [20] is meant for applications requiring a better reliability than best effort service. The SLA will specify the amount of bandwidth that is allocated for a specific flow or aggregate of flows. Assured Service offers four classes and three drop precedences within each class. Each class is allocated a specific bandwidth and each drop precedence within each class has a drop probability associated with it. The classes and the drop precedences are all assigned a specific codepoint, based on which the incoming packets are classified. Unlike premium service, where bandwidth is guaranteed, in assured service, no strong guarantees are made. Traffic in excess of the allowed rate will be reclassified as best effort.

The differences between per-flow QoS architecture (e.g. Integrated Services) and aggregate QoS architecture (e.g. Differentiated Services) are summarized in Table 2.1.

Table 2.1: Differences between per-flow QoS architecture and aggregation based QoS architecture

Feature	**Per-flow QoS architecture**	**Aggregation based QoS architecture**
Scalability	In per-flow QoS architecture, service is offered in the granularity of a flow. The amount of state information that needs to be maintained is proportional to the number of flows. Thus, this architecture does not scale well for large networks like the internet.	In aggregation-based architecture, service is offered in the granularity of a class. The amount of state information that needs to be maintained is proportional the number of classes. Multiple flows may be mapped to the same class, which is identified by the Diff-serv Code Point.
Flow State and Per-Hop Behavior	Per-flow QoS architecture suggests the use of "soft state", that is, intermediate nodes timeout on the absence of traffic and recover the resources that have been allocated for a flow.	In the aggregation-based architecture, Per-Hop Behaviors are implemented in the nodes that support the architecture. There is no concept of "soft state" in this architecture.
Resource Management	A flow admission control module does the resource management to determine if the flow's requirements can be satisfied.	A policy manager (also referred to as a *Bandwidth Broker*) does resource management. There is only one policy manager in a domain supporting this architecture.
End-to-end QoS	The per-flow architecture supports end-to-end QoS by explicit out-of-band signaling to set up the resources.	The aggregation-based architecture achieves end-to-end QoS by setting up static bilateral agreements between domains supporting the architecture.
Implementation	The per-flow architecture would work only if all the nodes in the path from the source to the destination support the architecture.	The aggregation-based architecture needs deployment only in the egress nodes in a compliant domain.

2.1.5 Multi-Protocol Label Switching (MPLS)

MPLS is a packet forwarding scheme. It evolved from Cisco's *Tag Switching*. In the OSI seven-layer model, it is between Layer 2 (link layer) and Layer 3 (network layer). Each MPLS packet has a header. The header contains a 20-bit label, a 3-bit label, a 3-bit *Class of Service (COS)* field, a 1-bit label stack indicator and an 8-bit TTL field. The MPLS header is encapsulated between the link layer header and the network layer header. An MPLS capable router, termed *Label Switched Router (LSR)* examines only the label in forwarding the packet. The network can be IP or any other network layer protocol. It is for this reason that this architecture is called Multi-Protocol Label Switching.

MPLS needs a protocol to distribute labels to set up *Label Switched Paths (LSPs)*. Whether a generic *Label Distribution Protocol (LDP)* [22] should be created or whether the *Resource Reservation Protocol (RSVP)* [23] should be extended for this purpose is a topic of discussion in the IETF. A Label Switched Path is similar to an ATM Virtual Circuit (VC) and is uni-directional from the sender to the receiver. MPLS LSRs use the protocol to negotiate the semantics of each label, i.e., how to handle a packet with a particular label from the peer. LSP setup can be control-driven, i.e., triggered by control traffic such as routing updates. Or, it can be data driven, i.e., triggered by the request of a flow or a *Traffic Trunk*. In MPLS, a traffic trunk is an aggregation of flows with the same service class that can be put into a LSP. The LSP between two routers can be set up as the Layer 3 hop-by-hop route, or the sender LSR can specify an Explicit Route (ER) for the LSP. The ability to set up ERs is one of the most useful features of MPLS. A forwarding table indexed by labels is constructed as the result of label distribution. Each forwarding table entry specifies how to process packets carrying the indexing label.

Packets are classified and routed at the ingress LSRs of an MPLS-capable domain. MPLS headers are then inserted. When a LSR receives a labeled packet, it will use the label as the index to look up the forwarding table. This is faster than the process of parsing the routing table in search of the longest match done in IP routing. The packet is processed as specified by the forwarding table entry. The incoming label is replaced by the outgoing label and the packet is switched to the next LSR. This label-switching process is similar to ATM's

VPI/VCI processing. Inside a MPLS domain, packet forwarding, classification and QoS services are determined by the labels and the COS fields. This makes LSRs simple. Before a packet leaves a MPLS domain, its MPLS label is removed.

MPLS LSPs can be used as tunnels. When a packet enters the start point of the tunnel, its path is completely determined. The packet will emerge at the end of the tunnel. With MPLS, a packet's path is completely determined by the label assigned by the ingress LSR. There is no need to enumerate every intermediate router of the tunnel. MPLS is therefore more efficient in terms of header overhead than any other tunneling mechanism.

In short, MPLS is strategically significant because:

1. it provides faster packet classification and forwarding
2. it provides an efficient tunneling mechanism

2.1.6 Traffic Engineering

QoS schemes like Integrated Services [6] and Differentiated Services [7] provide graceful degradation of performance when the traffic load is heavy. When the traffic load is light, *int-serv, diff-serv* and *best effort service* make little difference. Avoiding congestion is the motivation for *traffic engineering*.

Network congestion can be caused by lack of network resources or even by uneven distribution of traffic. In the first case, all routers and links are overloaded and the only solution is to provide more resources by upgrading the infrastructure. In the second case, some parts of the network are overloaded while other parts are lightly loaded. Uneven traffic distribution can be caused by the current Dynamic Routing protocols such as Routing Information Protocol (RIP) [28], Open Shortest Path First (OSPF) [29] and Intermediate System to Intermediate System (IS-IS) [30], because they always select the shortest paths to forward packets. As a result, routers and links along the shortest path between two nodes may become congested while routers and links along a longer path are idle. The Equal-Cost Multi-Path (ECMP) option of OSPF and recently of IS-IS is useful in distributing the load to several shortest paths. But, if there is only one shortest path, ECMP does not help. For simple networks, it may be possible for network administrators to manually configure the cost of the links, so that traffic can be evenly distributed. For complex ISP networks, this is almost impossible.

Traffic Engineering is the process of arranging how traffic flows through the network, so that congestion caused by uneven network utilization can be avoided. *Constraint Based Routing* is an important tool for making Traffic Engineering automatic. Avoiding congestion and providing graceful degradation of performance in the case of congestion are complementary. Traffic Engineering therefore complements Differentiated Services [24].

2.2 End-to-end signaling mechanisms in mobile networks

This section discusses some of the end-to-end signaling mechanisms that have been proposed for mobile networks. Signaling is the process of setting up a connection from the source to the destination. If the connection set-up process involves reservation of resources in the intermediate nodes from the source to the destination, then this process is referred to as *QoS Signaling*.

Signaling can be broadly classified into out-of-band signaling and in-band signaling. In out-of-band signaling, prior to any data transfer, a connection is set up from the source to the destination. In other words, there is an explicit connection setup phase. ATM, for example, goes through a connection set up phase before any transfer of data. In in-band signaling, connection is set up along with the transfer of data, i.e., there is no explicit connection set up phase.

2.2.1 Mobile PNNI – Out-of-band Signaling

Connection Management process consists of call establishment, location management and call handoff. A mobile node is nominally associated with one of the nodes within mobile enhanced private network designated the Home Agent (HA). The HA maintains a database known as the Home Location Register (HLR) containing a list of mobile nodes that are nominally associates with the particular HA including their current address and network attachment information. As shown in Figure 4, mobile enhanced node A.3.1.1 is the HA of node A.3.1.2. However, mobile node A.3.1.2 is not currently attached to HA node A.3.1.1

but is attached to the Foreign Agent (FA) node A.3.2.1. The current address information and the FA address of the mobile node A.3.1.2 are maintained in A.3.1.1.'s HLR.

Figure 2.4 shows the sequence of messages across the Wireless PNNI [25] when the call is initiates and cleared from Terminal (TE-x) attached to the fixed node F.1 to TE-y attached to mobile node A.3.1.2. There are 2 classes of nodes within the wireless PNNI topology, normal nodes with the existing PNNI functions and mobile enhanced nodes with the wireless PNNI functions. In Figure 2.4, F.1 is a normal node while nodes A.1.1, A.2.1, A.3.1.1, A.3.2.1 and A.3.1.2 are mobile-enhanced nodes. Mobile-enhanced nodes have the capability to support radio access layer (RAL) protocols and the mobile PNNI extensions. When A.3.1.1 receives the setup message and determines that the mobile node A.3.1.2 is not currently attached to it and its current FA address is A.3.2.1, it transmits a RELEASE message with a Crankback information to the preceding node with a Crankback cause "mobile node relocated" and Crankback cause diagnostic specifying the mobile node's current address, A.3.1.2 and F.A address A.3.2.1.

The Crankback level is determined by an algorithm which takes into account the addresses of the mobile node's HA, FA and the level of entry border node into the mobile-enhanced private network. It would be ideal to crank back to the highest possible level so as to maximize the routing alternatives to the destination. However, network topology and mobile node dynamics may limit the level of Crankback, possibly yielding sub-optimal routing decisions.

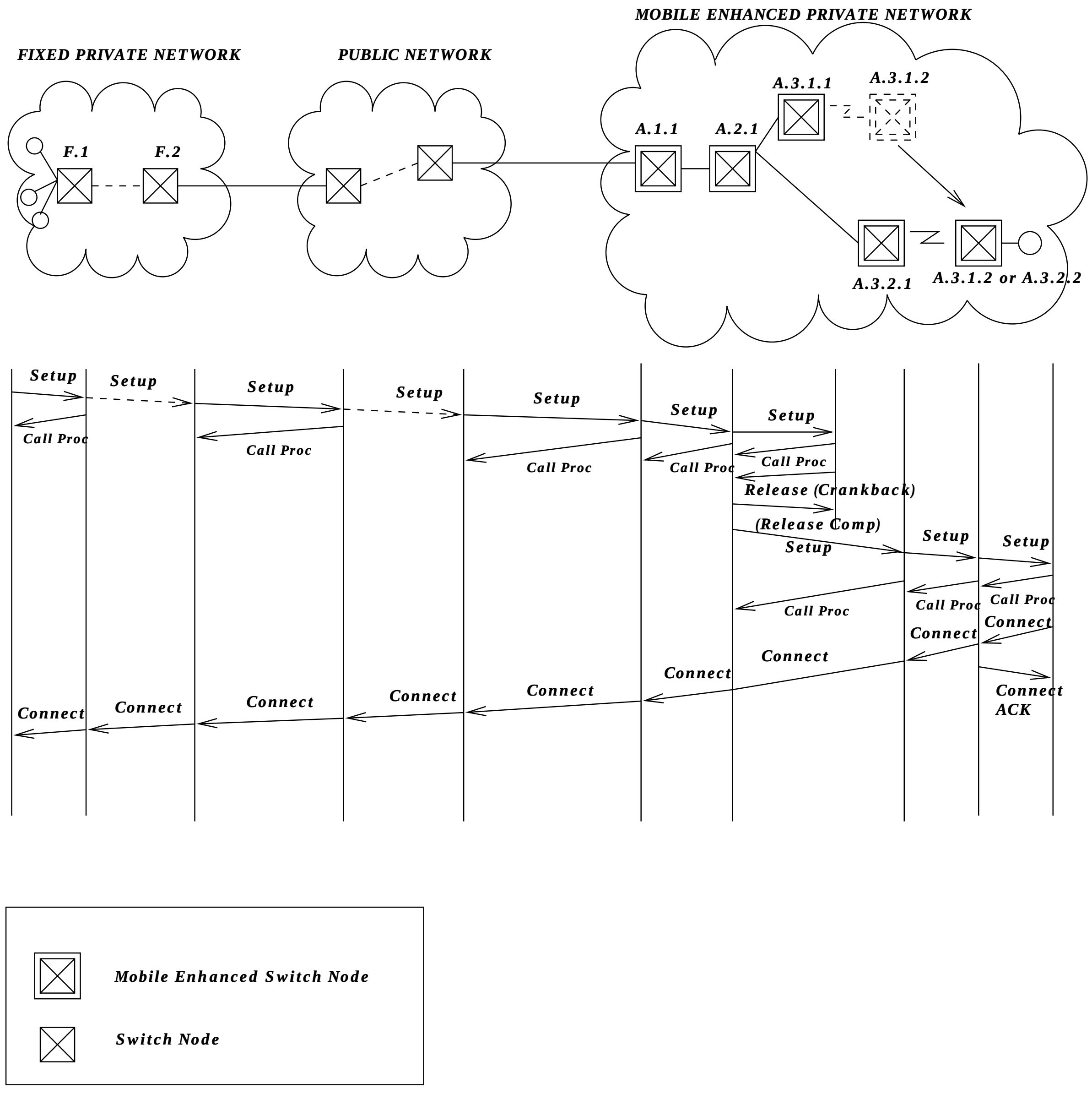

Figure 2.4: Mobile PNNI Signaling

Thus, in this approach, out-of-band signaling is done to set up the connection before starting the transfer of data. Also, in the event of a mobile node changing its point of attachment during a session, out-of-band signaling is done to re-establish the connection.
Thus, there is a very high signaling overhead in this approach.

2.2.2 INSIGNIA – In-Band Signaling

INSIGNIA – In-Band Signaling for QoS in Mobile Adhoc Networks [26], was proposed by the COMET group in the Columbia University. INSIGNIA is a component of the wireless flow management architecture for the delivery of adaptive real-time services in dynamic mobile adhoc networks. INSIGNIA supports fast flow reservation, restoration and adaptation algorithms that are specifically designed to deliver adaptive real-time service in a mobile adhoc networking environment.

INSIGNIA flow setup and QoS reporting are shown in Figure 2.5. To establish an adaptive real-time flow, INSIGNIA uses a new IP option to establish, restore and adapt resources between source-destination pairs. When the intermediate nodes receive packet with the appropriate option field, they reserve the resources if available and forward the packet towards the destination. The destination sends a QoS report message to the source periodically. The QoS report will indicate the state of the network to the source. This report could take a different path to the source. The source takes adaptation decisions based on the QoS report. The flow setup process is shown in Figure. All the intermediate nodes maintain soft state. The absence of traffic will result in the resource allocated for the flow being recovered.

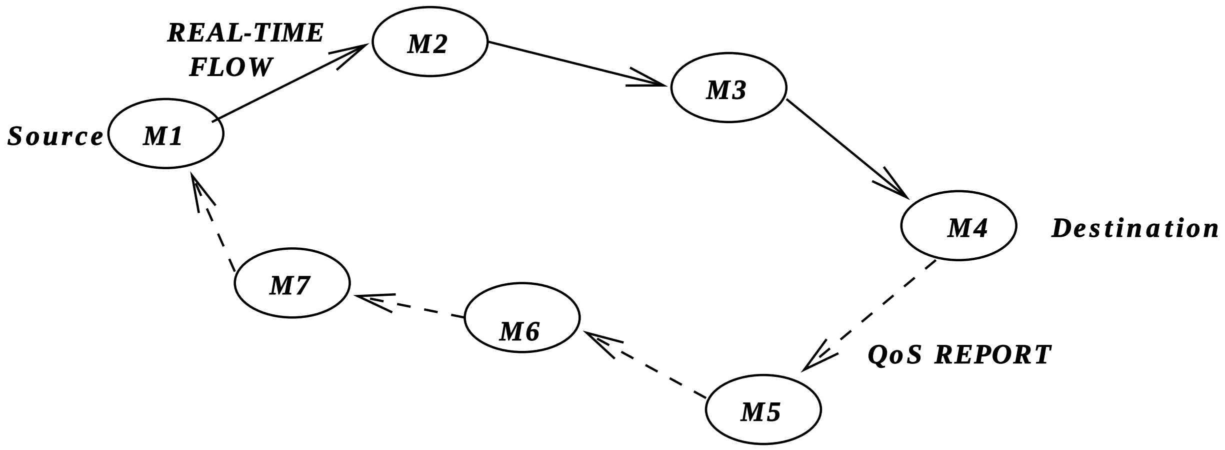

Figure 2.5: INSIGNIA: Flow Establishment and QoS Reporting

The evaluation of the signaling protocol indicates that in-band signaling is more efficient that out-of-band signaling for supporting end-to-end QoS in highly dynamic networking environments like mobile adhoc networks, where network topology, node connectivity and end-to-end QoS are strongly time varying.

2.3 QoS Requirements in Mobile Tactical Applications

Defining QoS requirements for mobile tactical applications provides a significant challenge. For example, the user level specification may vary according to the environment in which the application is being used: under field tactical conditions, a highly degraded image or a barely intelligible voice message will likely be much better than no communication at all. This variable requirement is in contrast to normal government, commercial and consumer specifications. Tables 2.2 through 2.7 [27] describe the various QoS parameters for each of the services. Some of the parameters are service specific. Some of the system level parameters are associated with ATM QoS descriptors.

Table 2.2: Voice Services

	Telephony	**Teleconferencing**	**Push-to-talk**	**Voice messaging**
Service Access Time	5 sec (desired) 20 sec (max)	5 * n sec (n= number of participants	250 msec	5 sec (desired) 20 sec (max)
Service completion rate	90 % (min) 98 % (desired)	90%	99.5%	98%
Latency	< 250 msec	< 250 msec	< 250 msec	< 500 msec
Delay Jitter	< 100 msec	< 100 msec	< 100 msec	<250 msec
Bit Error Rate	<10E-4	<10E-4	<10E-4	<10E-4
Traffic Symmetry	Symmetric	Many-to-many	Symmetric	One-way
Service Priority	To be assigned		To be assigned	
Service Security	Optional	Optional	Optional	Optional

Table 2.3: Electronic Messaging

	One-way (e.g. email)	**Interactive**
Service Access Time	1 sec (desired) 10 sec (max)	1 sec
Service Completion Rate	98%	99.5%
Latency	<1 minute	<1 sec
Delay Jitter	<500 msec	<500 msec
Bit Error Rate	10E-6	10E-6
Traffic Symmetry	Not applicable	Symmetric
User acknowledgements	Yes/No (selectable)	Not applicable
Message Size	20Kb	2Kb
Attachments	1-10Mb	1-5 Mb
Data Rate/ Bandwidth	As available	As available
Service Priority	Routine, priority, immediate, flash, flash over ride	

Table 2.4: Interactive transaction processing (including database query)

	Server-to-Client	**Client-to-Server**
Service Access Time	1 sec (desired) 10 sec (max)	1 sec (desired) 10 sec (max)
Service Completion Rate	95 % (min) 99% (desired)	90% (min) 99% (desired)
Latency	< 1sec (text and image) <250 msec (video	<1 sec
Delay Jitter	<500 msec (text and image)	50 msec
Bit Error Rate	10E-4 (text) 10E-6 (image and video)	10E-6
Traffic size per transaction	< 20 Kb(text) < 2 Mb (image) < 10 Mb (video)	2 Kb (query)
Data Rate/Bandwidth	As available	As available
Service Priority	To be defined	To be defined
Service Security	Required	Required

Table 2.5: High Volume Data

	Broadcast	**Interactive**
Service Access Time	<5 sec <20 sec (max)	5 sec (max)
Service Completion Rate	90% (min) 98% (desired)	98 %
Latency	500 msec	500 msec
Delay Jitter	250 msec	250 msec
Bit Error Rate	10E-6	10E-6
Traffic Symmetry	One-way	Asymmetric
Service Security	Desirable	Desirable

Table 2.6: Web Services

	Web Browsing	**Web-based transactions**
Service access time	< 1 sec (50%) < 5 sec (90%)	< 1 sec (50%) < 5 sec (90%)
Service Completion Rate	90% (min) 98% (desired)	95% (min) 99% (desired)
Latency	From the user – 100msec To the user (specifiable)	From the user – 100msec To the user – 500 msec
Latency Jitter	From the user – 50 msec To the user – 50% of average latency	From the user – 50 msec To the user – 500 msec
Bit Error Rate	10E-6	10E-6
Traffic Symmetry	Asymmetric	Asymmetric
Service Priority	User definable	User definable
Service Security	Optional	Required

Table 2.7: Video

	Video Conferencing	**Video Broadcast**	**Video-on-demand (interactive)**
Service Access Time	5 sec (desired) per connection 20 sec (max) per connection	15 sec (desired)	From the user: 1sec To the user 5 sec (max)
Service completion rate	95 % (min) 98% (max)	90%	98%
Latency	500 msec	500 msec	100 msec
Delay Jitter	100 msec	250 msec	50 msec

Bit error rate	10E-4(wireless)
	10E-7 (cable)
	10E-10 (fiber)
		10E-4 (wireless)
		10E-7 (cable)
		10E-10 (fiber)
			10E-4 (wireless)
			10E-7(cable)
			10E-10(fiber)

Traffic symmetry	Symmetric Many-to-many	One-way	Asymmetric
Data Rates/ Bandwidth	H.261 MPEG-1 MPEG-2	H.261 MPEG-1, MPEG2 MPEG-4	MPEG-1, MPEG-2
Service Security	Optional	Optional	Required

This chapter has thus given a brief overview of some of the QoS architectures, end-to-end connection setup mechanisms and the QoS requirements in mobile tactical applications. The next chapter discusses the problems in using any one of the standard architectures described in this chapter for RDRN and proposes a QoS architecture tailored for a highly dynamic networking environment like RDRN.

Chapter III

Design

This chapter first begins with a description of the requirements from a QoS architecture for highly dynamic networking environments like RDRN. It then discusses the motivation behind the thesis, in which the applicability of the various QoS architectures discussed in the second chapter, are considered for use in a highly dynamic networking environment. This is followed by a discussion of the proposed QoS architecture for RDRN and the proposed flow specification. The chapter concludes with a detailed description of the proposed flow establishment protocol.

3.1 Requirements for the QoS architecture for RDRN

This section discusses the RDRN QoS architecture requirements. The QoS architecture defines the framework that will be used to specify and implement the required performance characteristics of applications running on a RDRN network. The requirements for the architecture include the ability to:

1. Configure, predict and maintain the requested QoS (as far as possible) during the lifetime of the flow. Maintenance of the requested end-to-end QoS requires monitoring of the link to determine link failures and adapt the requested QoS appropriately.
2. Shield the application from the complexity of the underlying QoS specification and QoS management. The user needs to be provided with a QoS based application programmer interface (API). Thus the amount of functionality in the application can be reduced and it delegates the complexity of handling the QoS management to the underlying framework.
3. Set up resources in all the MAPs in the path from the source to the destination. Strict end-to-end QoS guarantees are required because of the time-critical nature of the applications that will use the RDRN network.
4. Reduce or avoid any out-of-band signaling to set up the resources in the MAPs.
5. Ensure that no flow is rejected connection, i.e., if requested resources are not available, then the flow should be considered as best effort.

3.2 Motivation

The RDRN protocol stack consists of IP over ATM, and as a result, many of the architectures described in Chapter 2 could be considered. However, the unique characteristics of the RDRN network make the use of these existing architectures inappropriate. The following subsections discuss the applicability of some of the QoS architectures for the RDRN system.

3.2.1 Integrated Services

The basic philosophy of integrated services [6] is to offer end-to-end guarantees by reserving the resources from the source the destination. This is done by the Resource Reservation Protocol (RSVP) [18]. RSVP is a receiver-oriented protocol, i.e., the resource reservation is initiated by the destination. The resource request message contains the flow specification, which will be used in the flow admission control process. The routers maintain soft state, that is, the resources are recovered in the absence of a flow.

The RDRN system needs strict guarantees on the end-to-end QoS, because of the time-critical nature of some of the applications that will be using the network. Thus, resources have to be reserved from the source to the destination. However, RSVP is an out-of-band signaling protocol that sets up the resources before any transfer of data. Besides, it is a receiver-based protocol. In highly dynamic networking environments like RDRN, the resources available are less compared to wired networks. Thus, the available resources need to be made use of in the most efficient manner. Given these requirements, the integrated services architecture is not suitable for use in RDRN.

3.2.2 Differentiated Services

Differentiated Services [7] suggests a scalable solution for the provisioning of differential treatment to packets. The preferential treatment is given based on the DS byte (commonly referred to as the TOS byte) in the IP header. Routers classify packets based on behavior aggregates. There is no explicit connection set up phase in this architecture. The marking, shaping and policing are all done at the edge of the network.

This architecture is inappropriate for the RDRN network because of the lack of guarantees of preferential treatment. Also, differentiated services architecture assumes that the backbone networks has abundant bandwidth, which is not the case in RDRN. As the authors in [6] argue, guarantees cannot be given unless resources are reserved. Assured Service really offers no guarantee of resources. Premium Service, on the other hand, may offer some guarantees with respect to delay, but other QoS parameters of interest are not within the scope of the service.

3.2.3 Multi-Protocol Label Switching

In this architecture [23], the nodes runs the Label Distribution Protocol (LDP) to exchange labels and set up Label Switched Paths (LSPs). Packets are classified and routed at the ingress Label Switched Router (LSR) of an MPLS-capable domain. MPLS headers are then inserted. When a LSR receives a labeled packet, it will use the label as the index to look up the forwarding table. This is faster than the process of parsing the routing table in search of the longest match done in IP routing.

However, this architecture is not suitable for highly mobile networking environments like RDRN. Frequent changes in the network topology results in frequent changes in the routing table, as a result of which LDP has to exchange more messages to track the current network topology. This in turn results in an inefficient use of the available bandwidth. Thus, this architecture is not suited for RDRN.

The Hiedelberg and the XRM QoS model are also not suited for highly dynamic networking environments because of the out-of-band signaling that is advocated by these architectures. Thus, none of the existing QoS architectures/resource reservation models could be used for the RDRN network. The motivation behind this thesis is to define one such architecture and propose a flow establishment protocol that works on the framework provided by the QoS architecture to set up end-to-end flows with QoS guarantees.

3.3 RDRN QoS Architecture

Based on these requirements, a QoS architecture has been proposed for Rapidly Deployable Radio Networks. This is shown in Figure 3.1. The focus of this thesis is on the flow management functionality, and its interactions with the rest of the elements in the architecture. This section describes the RDRN QoS architecture in detail.

3.3.1 The application layer

The application layer specifies the following:

1. The *flow specification*, which is used to indicate the required QoS and specify the characteristics of the traffic that will be generated by the application.
2. The *filter specification*, which is used to identify the packets that belong to the flow.

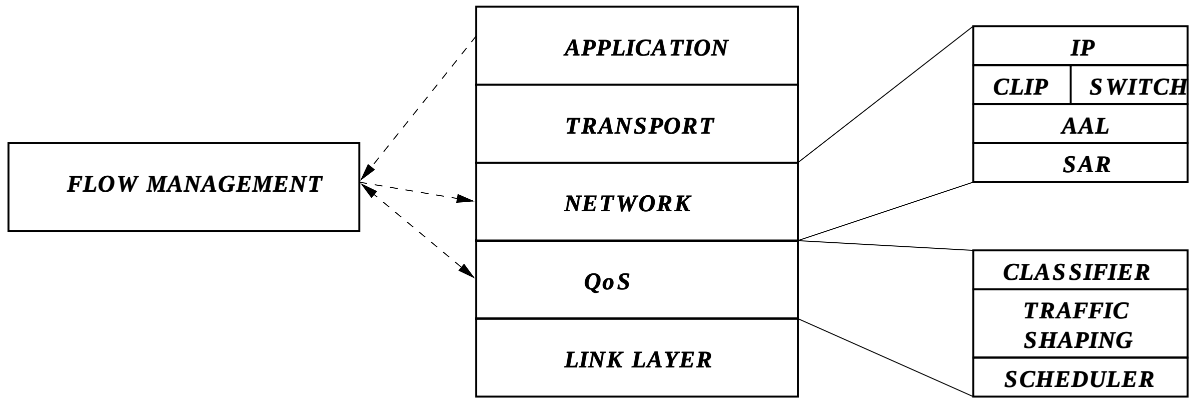

Figure 3.1: QoS Architecture for RDRN

The programmer will be provided with an application programmer interface (API) that can be used to specify the flow specification. The filter specification is obtained from information available in the packet. The flow specification and the filter specification put together is referred to as the *flow descriptor*. Before going into the details of the rest of the architecture, it is important to first define the flow specification and filter specification that will be used in RDRN.

<u>3.3.1.1 Flow Specification</u>

Flow specification (commonly referred to as *flowspec*) is a data structure used by the internetwork hosts to request special services of the internetwork. These services often guarantee how the internetwork would handle some of the traffic generated by the hosts. A flow specification (usually referred to as a flow spec) is part of the negotiation that will be done by the host with the internetwork to use a portion of the internetwork's resources. The flow specification is typically used to specify the service requirements in one direction only.

There are two different components in flow specification:

1. The QoS that is requested by the host, which identifies the resources requested by the host to the network.
2. The traffic that will be generated by the host, which identifies the traffic pattern that the network should expect from the host.

The proposed flow specification is tailored for highly dynamic networking environments like the RDRN network. The instability of the wireless links and the highly mobile nature of the nodes influence both the flow specification and the flow specific information that is maintained at each node.

This rest of this section discusses the flow specification that is used by the host for negotiation with the network. The specification typically includes the traffic type of the flow, the priority that needs to be assigned to the flow, the traffic parameters and the QoS specification.

Traffic Type

This represents the type of traffic that the flow will carry and may be one of *real time/ non-real time*. The traffic type will determine the type of commitments that will be required from the flow. Real time traffic will need deterministic commitments while non-real time traffic may need only best effort commitments. However, there may be certain non-real time applications that need deterministic commitments too. Such requirements are specified by the application.

Priority

There may be multiple real time and non-real time applications in the same node (MAP or MN). It may be necessary to give one particular application's traffic preference over another application's traffic, e.g., FTP traffic might need a higher priority over TELNET traffic. For this purpose, the priority field is provided in the flow specification. A class-based queue is used to differentiate traffic from the various applications based on the priority field. Priority can be one of *high priority/ medium priority/ low priority*.

Traffic Parameters

The traffic pattern should indicate the type of traffic that the source expects to give to the network. The traffic pattern is characterized with the help of the token bucket algorithm. The traffic control parameters of interest are the bucket size, maximum transmission rate, maximum burst size and token arrival rate.

Quality of Service

The QoS field should indicate the quality of service that the host expects from the network, given the traffic that will be generated by the host. For real-time flows, there are essentially two types of payloads, namely the *base layer* and the *enhancement layer*. The requirements from both these layers need to be specified in the *flowspec*. The base layer represents the minimum QoS requirements of the flow while the enhancement layer represents the maximum QoS requirements, both of which are specified in the flowspec.

The end-to-end QoS will characterize the traditional QoS parameters supported in wired networks, namely the *delay, jitter, loss and throughput*. This typically represents the QoS that will be directly requested by the user, and that which is directly visible to him. The delay indicates the expected *end-to-end delay*. The jitter indicates the *end-to-end delay variation*. The loss indicates the *loss ratio* that is acceptable. The throughput indicates the *number of packets* that are received successfully without any error. As mentioned earlier, the media type will be used to determine the end-to-end QoS parameters. For example, in the case of audio, there is a stringent requirement in terms of the end-to-end delay and jitter. Throughput and reliability can be compromised to a certain extent as far as audio is concerned. In general, for audio, the throughput requirement can be as low as 4 Kbytes/second, the end-to-end delay requirement is 100ms and jitter acceptable is 10ms. These values are derived from the specifications given by the end user. Usually, for non-real time traffic the throughput is very important while the delay is not a very significant parameter.

Derived flow specification

This section discusses the flow specific information that is maintained by the mobile access points in the RDRN system. The flow specific information is derived from the flow specification that is sent by the host. Based on the nature of the RDRN system, the derived flow specific information that is maintained in the nodes consists of two aspects, namely the *wireless QoS* and the *mobile QoS* requirements.

Wireless QoS Requirements

The wireless QoS parameters that need to be maintained in the derived flow specific information deals with the nature of the wireless links. This includes the *link delay, error rate* and the *channel reservation*.

The *error rate* is derived based on the media type and the required end-to-end loss. For non-real time traffic, the error rate has to be very low, whereas for real-time traffic, the error rate can be higher. Additional protection can be offered to the packets desiring a low error rate, at the link level. The *link delay* is derived based on the media type and the required end to end delay. For real-time traffic, the link delay has to be very low, and it also depends on the required end to end delay. For non-real time traffic, though, the link delay is not a very significant parameter, and the end to end delay need not be guaranteed very strictly. The *channel reservation* is influenced by the priority that is specified in the flow specification. For high priority flows, a higher channel bandwidth is reserved. These are the QoS requirements that arise because of the characteristics of the wireless links.

Mobile QoS Requirements

Mobile QoS is mainly concerned with the QoS associated with the handoff. Each flow is associated with certain handoff parameters, namely the *handoff urgency* and the *handoff loss.* This flow specific information that is maintained in every node is also derived from the flow specification specified by the host application.

The need to do a seamless handoff leads to *the handoff urgency* or *deadline parameter*, which represents the priority that needs to be assigned to this handoff process. This could be one of *fast, medium or slow* representing the quickness with which the handoff needs to be completed. This is derived from the media type and the required end to end delay. Real time flows typically need low end-to-end delay and as a result, need a fast handoff. Non-real flows vary widely in delay requirements, and might require either a medium or a slow handoff.

The *handoff loss* (i.e. no loss allowed, loss allowed) essentially determines the type of handoff that needs to be done and is again dependent on the media type of the flow and the end to end loss requirements specified in the flow specification. Typically, for real-time flows, fast handoff takes precedence over the loss. As a result, the urgency is set to a fast, whereas the loss is set to *loss_allowed.* For non-real time flows, the urgency is set to slow, while the loss is set to *no_loss*. These are the QoS requirements that arise because of the highly dynamic nature of the RDRN system.

3.3.1.2 Filter Specification

The filter specification (commonly referred to as *filterspec*) provides the identification for the flow that is to get the QoS specified by the *flowspec*. The filter specification consists of the source IP address, the destination IP address, the source port number, the destination port number and the protocol (UDP/TCP). This information is available in all the packets that are sent from the source to the destination. Thus the filter spec is identified by the 5 tuple:
<Source IP, Source Port, Destination IP, Destination port, Protocol>

The next section discusses the flow management block, which is the main functional unit in the QoS architecture.

3.3.2 Flow Management

This section describes the flow management process in the RDRN QoS architecture. Flow management is the most important functional block in the architecture. The detailed flow management block is shown in Figure 3.2. It is responsible for performing the following functions:

1. QoS mapping: The application layer specifies the type of the application i.e. *real-time* or *non-real time*, as part of the *flow specification*. The flow management block maps the flow specification into a type-of-service (TOS) byte. The TOS byte is interpreted by the intermediate nodes for the provisioning of specific services to the flow. The application specifies its requirements in the form of a flow specification, which is mapped by the flow management module to a TOS byte. This

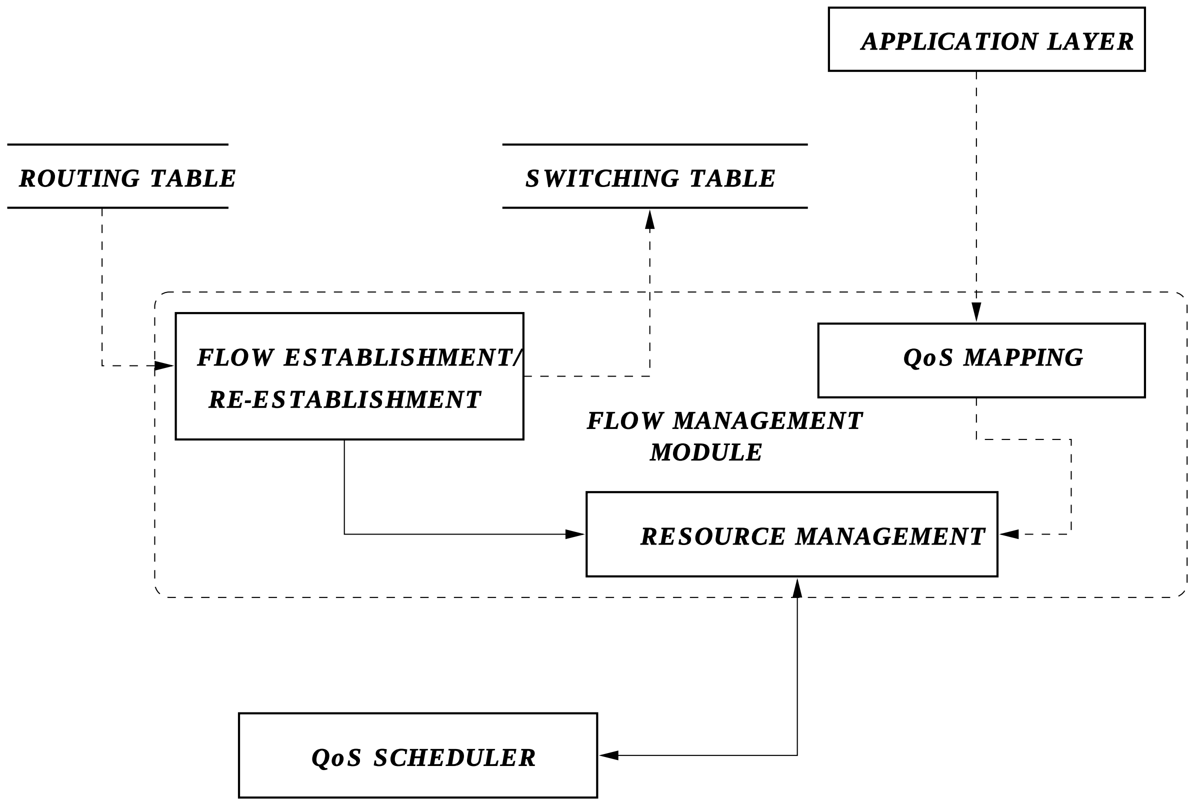

Figure 3.2: Flow Management Module

is given as an input to the classifier block in the architecture, which classifies the data cells to the appropriate queues.

2. *Flow Establishment/Re-Establishment:* The flow management block is responsible for end-to-end flow establishment. An in-band flow establishment protocol has been proposed for this purpose, which will be discussed in detail in the later sections. The flow establishment protocol will set up end-to-end flows based on the QoS requirements of the flow. The flow management block is also highly responsive to changes in the network (e.g., failure of a wireless link) and re-establishes the flows from the source to the destination. As part of this functionality, the flow management block controls the switch and the Classical IP over ATM (CLIP) blocks. The flow establishment protocol is explained in detail in section 3.1.1.2.
3. *Resource Management:* The flow management block is also responsible for resource management in the RDRN network. It does flow admission control to determine if a flow's requirements' can be satisfied. The intermediate nodes use the TOS byte to interpret the requirements of the flow. It then configures the scheduler for the resources requested.

Before going into further details on the various functions of the flow management block, it is important to understand the overall flow management model, that is, the various components that will be involved in flow management.

3.3.2.1 RDRN Flow Management Model

This section discusses the RDRN flow management model. The goal of the RDRN Flow Management model is to support delivery of real-time and non-real time services to the RDRN nodes under time-varying conditions. The flow management model allows packet audio, video and real-time data application to specify their maximum and minimum QoS requirements using the flow specification discussed in the previous section. The RDRN flow management model is shown in Figure 3.3. The functions of the modules in the flow management model are discussed in this section.

Wireless Multi-Hop Routing Protocol

This protocol dynamically tracks changes in the RDRN network topology and makes the current routing table visible to flow management module. The wireless multi-hop routing protocol maintains multiple paths to the destination node and installs the best route among these in the routing table. It also ensures a loop-free path to the destination with the help of the predecessor information that is maintained for each route. The routing protocol exchanges HELLO packets with the nearby nodes to indicate that the node and link are active. Absence of three consecutive HELLO packets is interpreted as a link failure and an alternate route is installed for the destination. The flow management module will use the routing table for the flow establishment and the flow re-establishment process.

Flow Management

As already mentioned in the previous section, the flow management block is responsible for the end-to-end flow establishment, flow re-establishment (in the event of a change in the path from the source to the destination) and resource management. The flow establishment protocol, as will be seen later, uses the routing protocol in the event of a link failure. Also, it will set up the switching table entry for a flow during the process of flow establishment. The flow management block also sets up the scheduler for the transmission of cells.

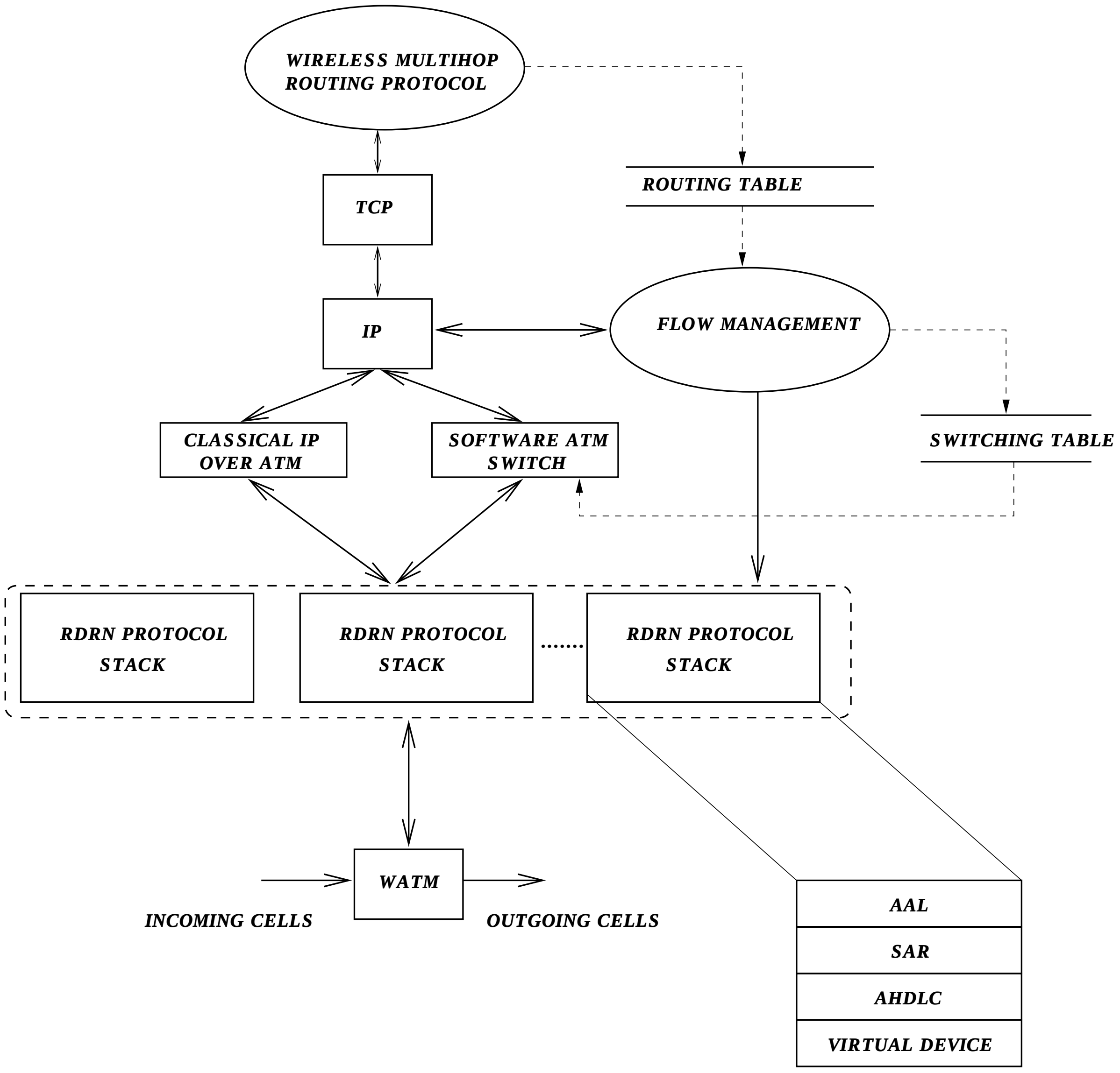

Figure 3.3: RDRN Flow Management Model

Having discussed the RDRN flow management model, let us now discuss the proposed in-band flow establishment protocol.

3.3.2.2 Flow Establishment

Flow establishment is the process of setting up the flow from the source to the destination. This involves reservation of resources in the nodes from the source to the destination. The RDRN system has a highly dynamic networking environment. The RDRN protocol stack consists of IP over ATM. It is different from typical mobile adhoc networks in the sense that hierarchy is imposed on the network by the presence of ATM in the protocol stack. The proposed flow establishment scheme intends to make use of the features of both IP and ATM in the protocol stack. The flow establishment protocol uses the routing table that is built by the wireless multi-hop routing protocol discussed in the previous section.

In the RDRN system, the MAPs discover each other (using the low speed orderwire system) and set up high speed point-to-point wireless ATM links to each other. A default ATM VPI/VCI is used for the exchange of IP datagrams between the MAPs. The default VPI/VCI is used for all the flows through the MAP for which a specific VPI/VCI has not yet been allocated. Specific VPI/VCIs will be set up during the process of flow establishment, and will be used once the end-to-end flow has been established. This default VPI/VCI is also used in the process of flow re-establishment.

The end-to-end flow establishment is done at the IP layer and is accomplished through the introduction of a new IP option field. This IP option is defined as the *QoS option.* The format of the QoS option is shown in Figure 3.4.

REQ/RES	***MAX/MIN/BEST***	***ALLOC/DE-ALLOC***	***VPI***	***VCI***
1	***2***	***1***	***16***	***16***

Figure 3.4: QoS Option Field

The basic idea of the proposed flow establishment scheme is to establish the flows at the IP layer. Once the flow is set up at the IP layer, the data is then switched at the link level (ATM). Switching at ATM layer is faster because the datagrams need not be reassembled at every MAP in the path from the source to the destination. The IP datagrams can be reassembled at the destination. Also, the time involved in processing the QoS option at every node can be avoided if layer 2 switching is done.

The establishment of the flows is done in-band, i.e., along with the transfer of data. Since the resources in a wireless environment are scarce, in-band signaling serves to improve the efficiency by avoiding the additional overhead in out-of-band signaling. The MAPs all maintain soft state, and as a result, the resources are released in the absence of data on the links.

<u>*QoS Option Field*</u>

The QoS option field is shown in Figure 3.4. The first field is the *REQ/RES* bit. This is used to indicate whether the flow has resources already reserved in the network (REQ/RES=1), or if the resources have to be reserved (REQ/RES=0).

The second field is the *MAX/MIN/BEST* bits. This is used by the intermediate MAPs to indicate to the subsequent MAPs in the downstream path whether or not the resource management block needs to be invoked for this particular flow. As already mentioned, the MAP is capable of doing both layer 3 IP routing and layer 2 ATM switching. The interpretation of the bits is shown in Table 3.1.

<u>Table 3.1 MAX/MIN/BEST bits</u>

Bit 0	**Bit 1**	**Message**
0	0	Unused
0	1	MAX
1	0	MIN
1	1	BEST

If a particular intermediate node is unable to fulfill the maximum requirements of the flow, it attempts to fulfill the minimum QoS requirements. If the minimum QoS requirements are satisfied, it sets the MAX/MIN/BEST bits to 10. The subsequent nodes need not try to fulfill the maximum QoS requirements, since the maximum end-to-end requirements will not be satisfied anyway. Thus, the flow admission control can be invoked to try and satisfy the minimum requirements of the flow. In a similar manner, if the minimum requirements are also not satisfied at any intermediate node, the MAX/MIN/BEST is set to 11. On receiving an IP datagram with the REQ/RES field=0 and MAX/MIN/BEST=11, the intermediate node treats the flow as best effort and does not invoke the resource management block at all. The destination will send periodic *QoS reports* to the source to indicate the status of the flow establishment. The path taken by the QoS reports from the destination to the source can be different from the path that has been set up from the source to the destination, and they are treated as best effort.

The third field is the *ALLOC/DE-ALLOC* bit. When this bit is set to 0, the node makes an attempt to allocate resources for the flow. When this bit is set to 1, the node de-allocates resources for the flow. The use of this field will be elaborated in the next section. It helps in scaling up the resources when they become available.

The last two fields are the VPI and VCI that will be used to send ATM cells from this flow to the next hop. The interaction between the layer 2 switching and the layer 3 routing will be explained later. This VPI and VCI will be allocated by the flow admission control module, and will be done so only if the maximum QoS requirements are satisfied or if the user requirements are only best effort. This VPI/VCI is different from the default VPI/VCI between the two MAPs that is setup during the process of discovery.

Source

The source node calls the resource management block to determine if the flow's requirements can be satisfied. If the flow's requirements can be satisfied, it sends the IP datagrams with the following options.

1. REQ/RES=0, indicating that this flow is requesting resources to be set up.
2. MAX/MIN/BEST = 01, indicating that the flow requires maximum resources to be set up.
3. ALLOC/DEALLOC = 0, indicating that the resources have to be allocated for the flow.

The VPI and the VCI that were returned by the resource management block (the flow admission module returns this only if the maximum QoS requirements are satisfied or if the user requirements are just best effort) are put in the VPI/VCI fields. This is the specific VPI/VCI that will be used for the flow once the end-to-end flow is established.

If the application requests only best effort service, then set REQ/RES = 0, MAX/MIN/BEST = 11 and the ALLOC/DE-ALLOC = 1. The intermediate MAPs just allocate a VPI/VCI for the flow and send it to the next hop. This flow will be treated as best effort throughout the life time of the connection and no attempt is made to scale up the resource allocation.

The source then continues to send all the IP datagrams in a similar manner to the next hop, until it gets a *QoS report* message from the destination. This QoS report message contains the IP options that the destination received in the IP datagram from the source.
The source checks the IP options field that was received from the destination in the QoS report message. It processes the IP options field in the following manner:

1. If MAX/MIN/BEST = 01, it indicates that the maximum end-to-end QoS requirements have been satisfied during the flow establishment. The source sets the REQ/RES bit to 1 and starts sending the packets over the specific VPI/VCI that has been allocated for the flow during the flow establishment process. It also retains the MAX/MIN/BEST=01 in these packets. This is shown in Figure 3.5. The first phase shows the process of flow establishment, while the second phase shows the link level connection that has been set up. The advantage of doing the data transfer at the link level has already been explained.

2. If MAX/MIN/BEST = 10 and the ALLOC/DE-ALLOC=0, it indicates that only the minimum QoS requirements have been satisfied, in which case the additional resources that have been allocated in the intermediate nodes that come early in the path (before detecting a bottleneck node, that could satisfy only the minimum end to end QoS requirements) have to be de-allocated. The source sets the MAX/MIN/BEST = 10, REQ/RES =1 and ALLOC/DE-ALLOC = 1. This conveys to the intermediate nodes that the flow has been assigned minimum QoS requirements, and that any additional resources can be de-allocated.

3. If the MAX/MIN/BEST = 10 and the ALLOC/DE-ALLOC=1, it indicates that the minimum QoS requirements have been satisfies and that the additional resources that were allocated in the path have been de-allocated. The source continues to send the IP datagrams with the same set of IP options. After a random time 't', which is defined as the scale-up time, the source sets the MAX/MIN/BEST=01, the ALLOC/DE-ALLOC=0

and the REQ/RES=RES in the IP option and sends them. This is done in an attempt to scale up to the maximum requirements if the resources become available. This pattern of the bits conveys to the intermediate nodes that the flow is attempting to scale up to the maximum QoS requirements. This is shown in Figure 3.6.

4. If the MAX/MIN/BEST = 11 and the ALLOC/DE-ALLOC=1, it indicates that even the minimum QoS requirements have not been satisfied and that the flow needs to be treated as best effort. The additional resources that were allocated in the path have been de-allocated. The source continues to send the IP datagrams with the same set of IP options. After the scale up time 't', the source sets the MAX/MIN/BEST=01, the ALLOC/DE-ALLOC=0 and the REQ/RES=RES in the IP option and sends them. This is done in an attempt to scale up to the maximum requirements if the resource become available. This pattern of the bits conveys to the intermediate nodes that the flow is attempting to scale up to the maximum QoS requirements.

If the source of flow get a QoS report from the destination specifying that the maximum QoS requirements have been satisfied, it starts sending the flow on the specific VPI/VCI that was assigned for the flow. However, the source continues to send the IP options in the IP datagram, which will be broken down into cells. The destination, on reassembling the datagram, will process the IP options field. If, due to handoff, or due to the degradation of a wireless link, an intermediate MAP is unable to continue with the provisioning of the QoS requirements of the flow, it reassembles the IP datagram of the flow, and sets the MAX/MIN/BEST to 10 or 11 based on the resources that are currently available (determined by the resource management module). The destination sends this option in the QoS report message to the source. The source, on receiving the IP option, decides on the action to be taken based on the adaptation policy that was specified for this flow by the application. The action could range from disconnecting the flow to re-negotiating the QoS, with the new flow specification that was specified. Then, it does steps 3 or 4 explained above. This is shown in Figure 3.7. Phase 1 shows the link level connection being set up from the source to the destination. It also shows the link that is broken. Phase 2 shows the upstream node (of the link that is broken) reassembling the IP datagram and changing the appropriate fields in the QoS option. The QoS report message is thus used by the flow management block to determine the current status of the network. This explains the behavior of the source nodes of the flows. The state diagram for the source of the flows is shown in Figure 3.8.

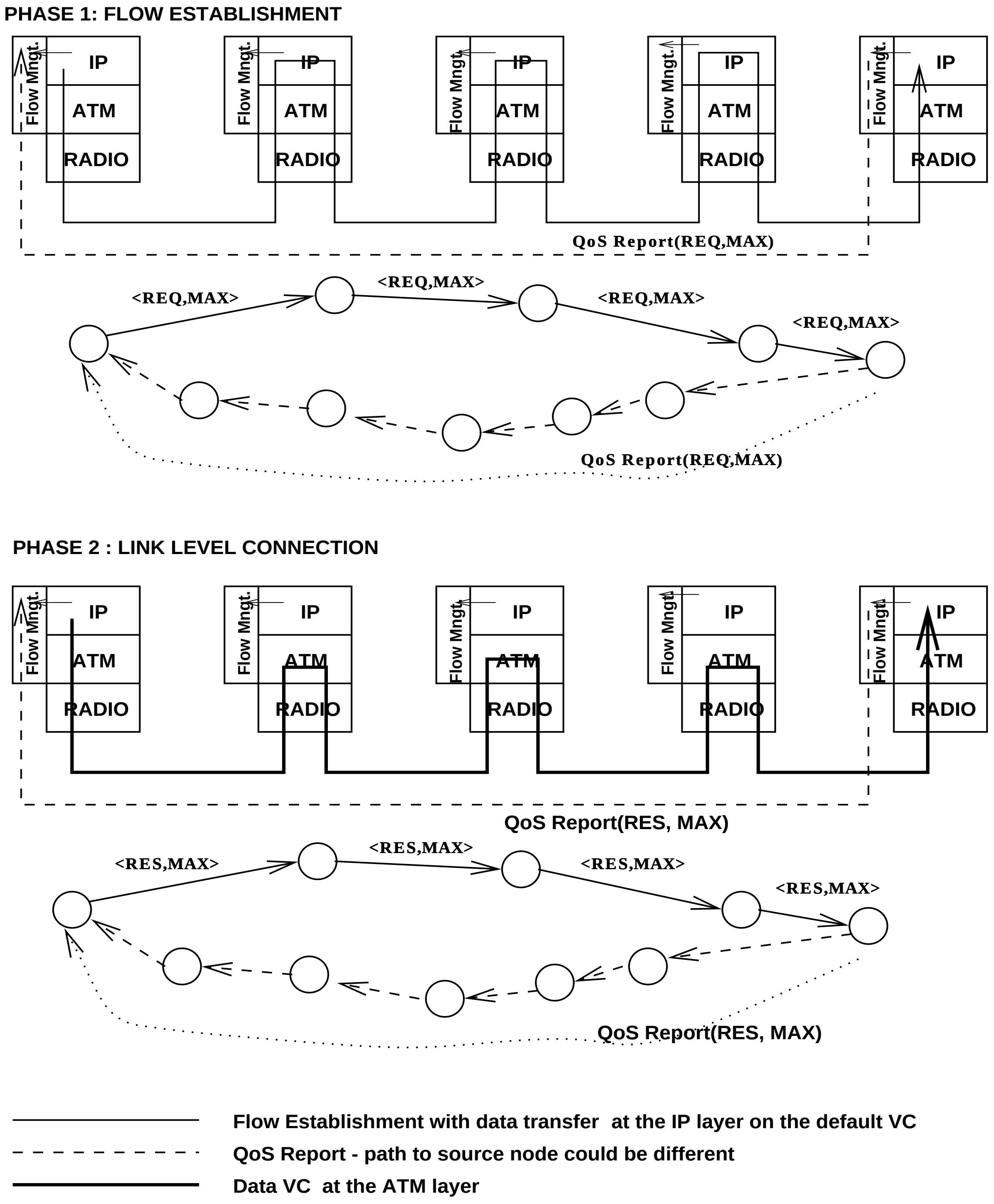
PHASE 1: FLOW ESTABLISHMENT
Flow Mngt.
IP
ATM
RADIO
QoS Report(REQ,MAX)
<REQ,MAX>
PHASE 2 : LINK LEVEL CONNECTION
QoS Report(RES, MAX)
<RES,MAX>
Flow Establishment with data transfer at the IP layer on the default VC
QoS Report - path to source node could be different
Data VC at the ATM layer

Figure 3.5: Successful flow establishment – Maximum Resources Reserved

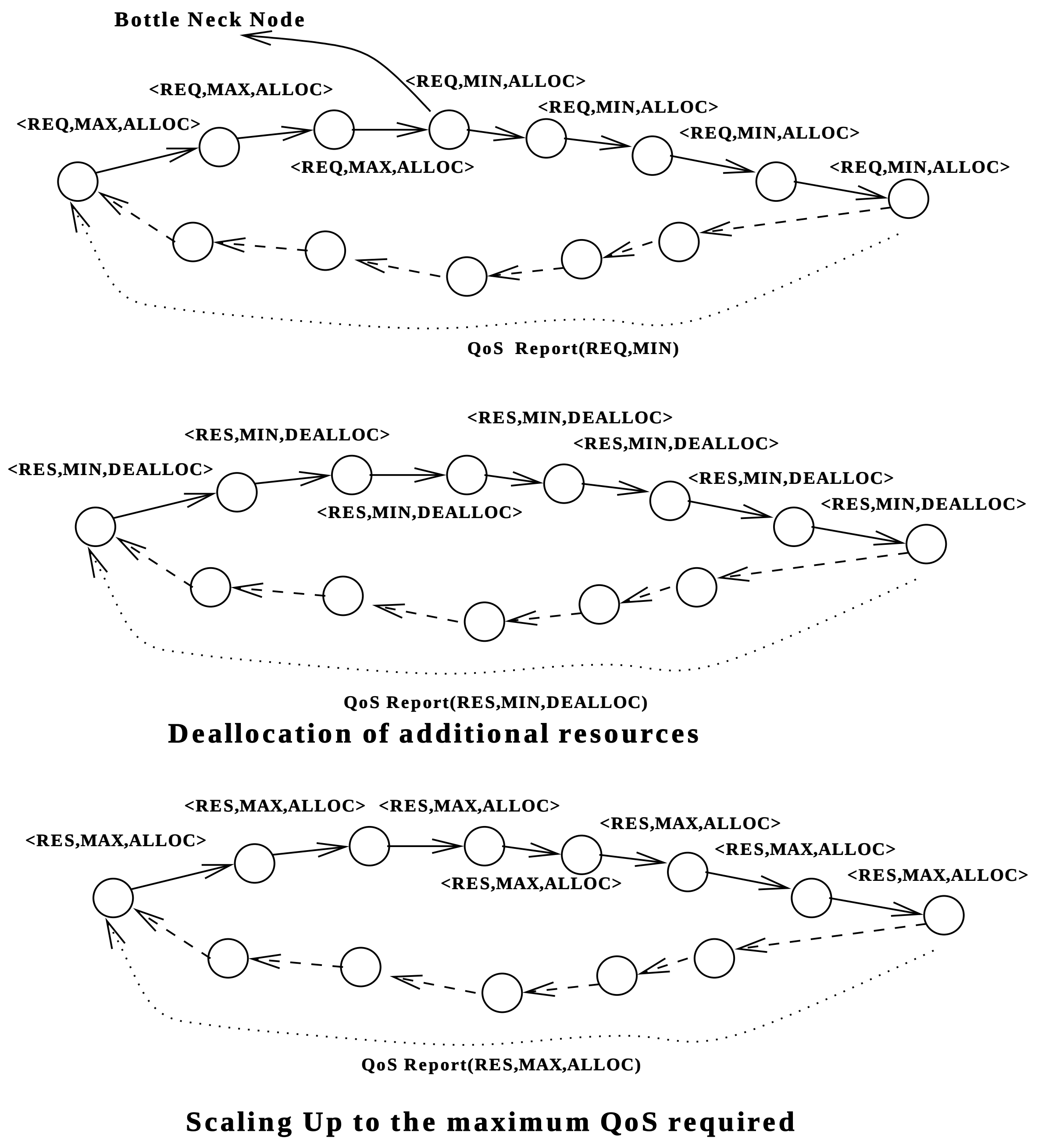

Figure 3.6: Scaling up resources

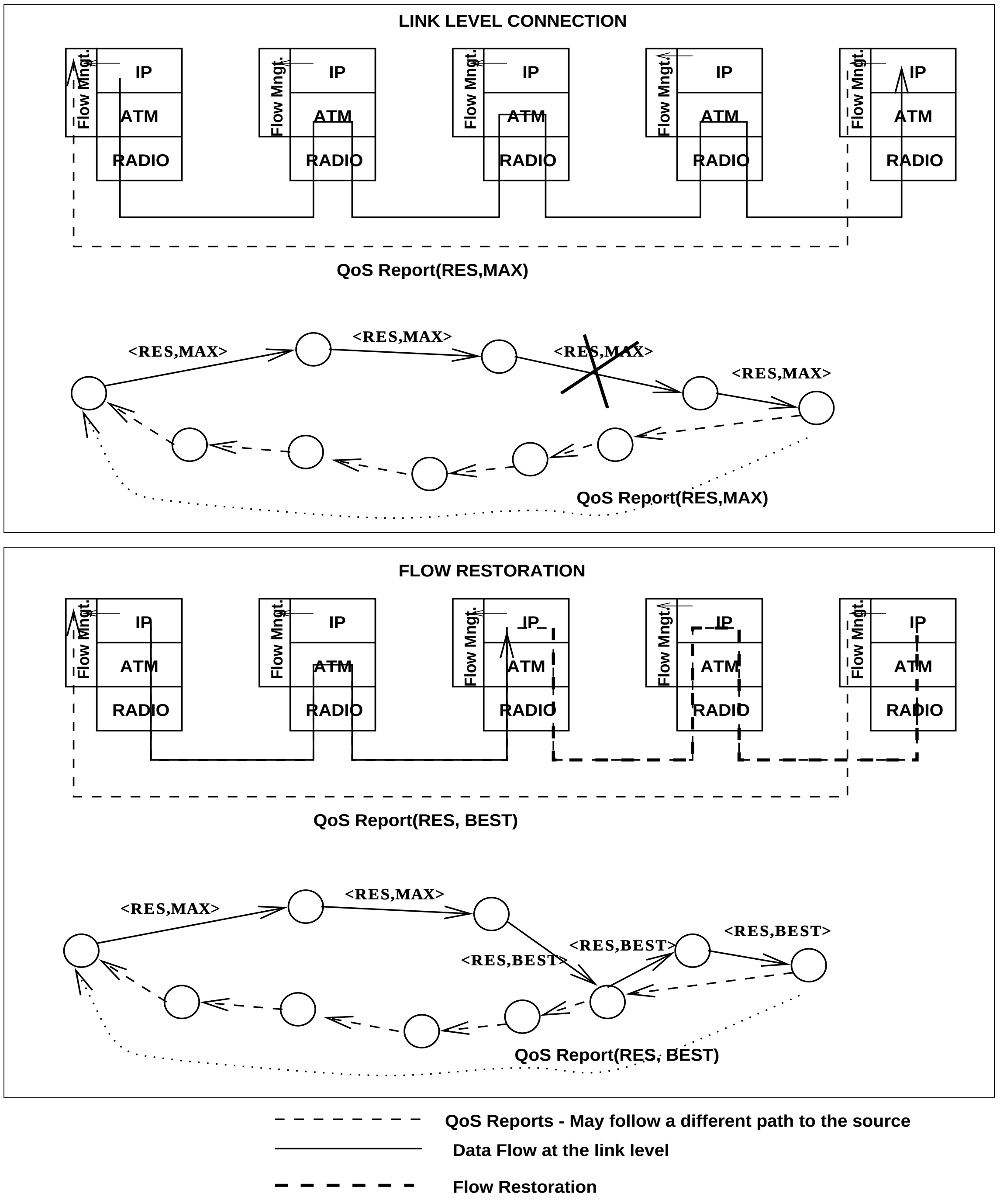

Figure 3.7: Flow Re-establishment

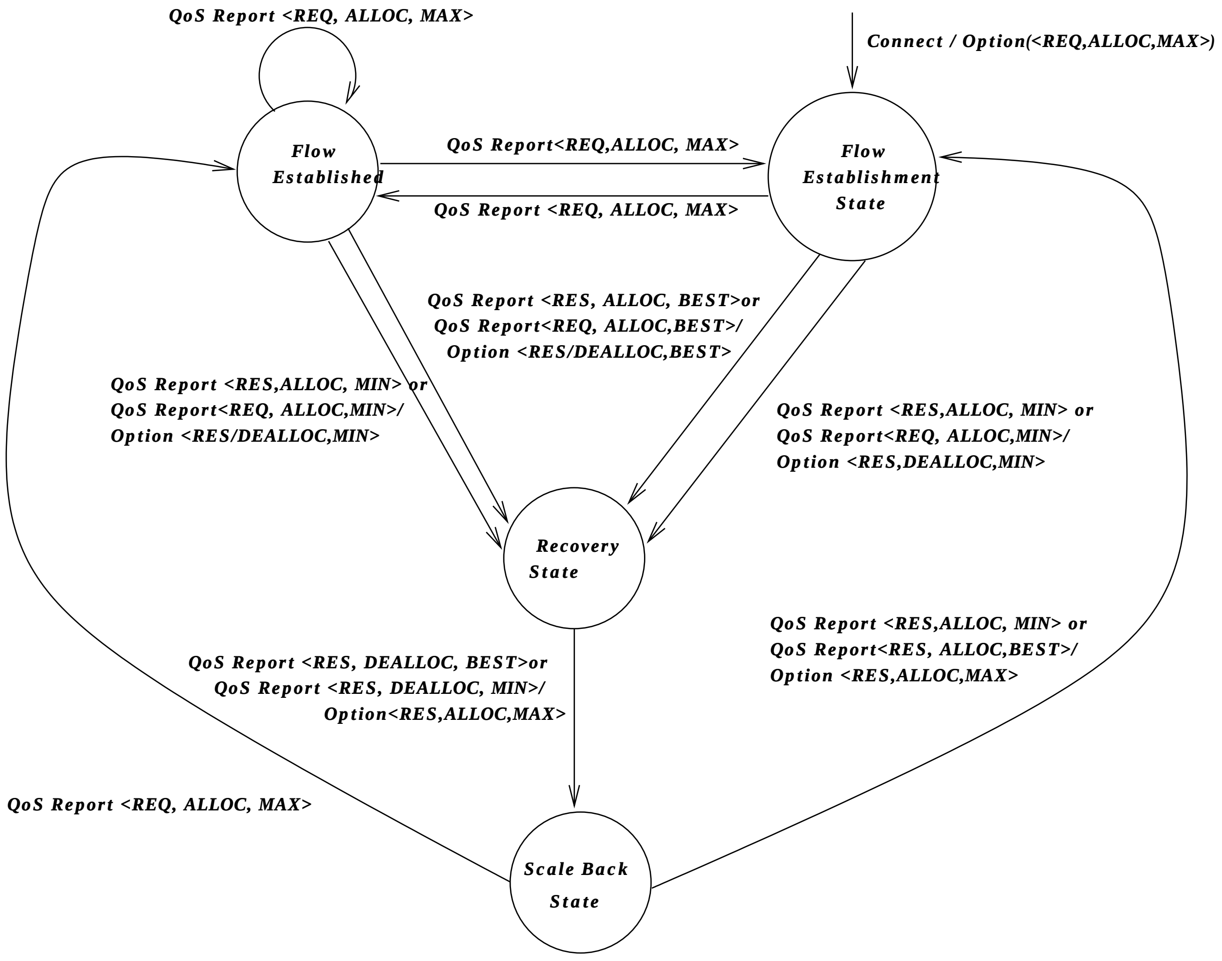
QoS Report <REQ, ALLOC, MAX>
Connect / Option(<REQ,ALLOC,MAX>)
Flow Established
QoS Report<REQ,ALLOC, MAX>
QoS Report <REQ, ALLOC, MAX>
Flow Establishment State
QoS Report <RES, ALLOC, BEST>or
QoS Report<REQ, ALLOC,BEST>/
Option <RES/DEALLOC,BEST>
QoS Report <RES,ALLOC, MIN> or
QoS Report<REQ, ALLOC,MIN>/
Option <RES/DEALLOC,MIN>
QoS Report <RES,ALLOC, MIN> or
QoS Report<REQ, ALLOC,MIN>/
Option <RES,DEALLOC,MIN>
Recovery State
QoS Report <RES,ALLOC, MIN> or
QoS Report<RES, ALLOC,BEST>/
Option <RES,ALLOC,MAX>
QoS Report <RES, DEALLOC, BEST>or
QoS Report <RES, DEALLOC, MIN>/
Option<RES,ALLOC,MAX>
QoS Report <REQ, ALLOC, MAX>
Scale Back State

Figure 3.8: Source State Diagram

<u>Intermediate Nodes</u>

When an intermediate node (Mobile Access Point) receives an IP datagram with an IP option field for flow establishment, it processes it in the following manner. The nodes in the system maintain soft state, that is, in the absence of activity, the resources will be de-allocated.

1. If the REQ/RES=0, ALLOC/DE-ALLOC=0 and MAX/MIN/BEST=01, then the intermediate node invokes the resource management block to determine if the flow can be admitted. If the flow can be admitted, then it sends the same IP option to the next node with the VPI and the VCI that was allocated for this flow by the flow admission control module. The flow management module also makes the corresponding entries in the switching table. This switching table contains the incoming VPI and VCI (available from the IP option field that was received from the previous node) and VPI and the VCI

for the flow that was allocated at this node. If the minimum QoS requirements are only satisfied, then the node sets MAX/MIN/BEST=10 and sends it to the next hop. If the minimum QoS requirements are also not satisfied, then the node sets MAX/MIN/BEST=11 and sends it to the next hop. The flow management module also maintains a flow table, which contains the filter specification (used to identify the packets from a flow), the corresponding flow specification, and a flag specifying what portion of the flows requirements have been satisfied (maximum, minimum or best effort).

2. If the REQ/RES=0, ALLOC/DE-ALLOC=0 and MAX/MIN/BEST=10, then the intermediate node does the flow admission control for the minimum QoS requirements only. The flow management module does not allocate a VPI/VCI if only the minimum QoS requirements are needed. It however, maintains the flow information in the flow table.

3. If the REQ/RES=0, ALLOC/DE-ALLOC=0 and the MAX/MIN/BEST=11, then intermediate node does not invoke the resource management module at all. The classifier will simply forward the datagram as best effort, and does not maintain any information about the flow at all.

4. If the REQ/RES=1, ALLOC/DE-ALLOC=1 and the MAX/MIN/BEST=10, then the flow had been allocated only minimum QoS requirements and the additional resources that have been allocated have to be recovered. This happens during the recovery state of the source as explained in Figure 3.7. From the flow table, the node determines the amount of resources that have been allocated for this flow. The additional resources are de-allocated (since ALLOC-DEALLOC is set to 1). If the maximum requirements have been allocated (available from the flow table), then the additional resources will be de-allocated. On the other hand, if only minimum requirements have been allocated for this flow, then the IP option field is retained in the packet that is sent to the next hop. These packets refresh the minimum resources that have been allocated for this flow.

5. If the REQ/RES=1, ALLOC/DE-ALLOC=1 and the MAX/MIN/BEST=11, then the flow needs to be considered best effort and all the nodes on the way that have allocated minimum or maximum requirements will de-allocate them and recover the resources. Information about the flow that is maintained in the flow table is removed. The switching table is also updated appropriately.

6. If the REQ/RES=1, ALLOC/DE-ALLOC=0 and the MAX/MIN/BEST=01, then it indicates that the source is attempting to scale up to the maximum QoS requirements. The node invokes the resource management block to determine if the maximum resources are available. If so, it allocates the maximum requirements. If the maximum QoS requirements cannot be satisfied, and if the flow already has the minimum requirements satisfied (indicated from the flow table), then it changes the MAX/MIN/BEST=10 and sends it to the next hop. If the maximum QoS requirements are not satisfied and if the flow had been considered as best effort until now, then the resource management block is invoked in an attempt to allocate the minimum resources. If the minimum resources are available, it sets the MAX/MIN/BEST=10 and sends it to the next hop. However, if the minimum requirements are also not satisfied, then the MAX/MIN/BEST=11 and the datagram is sent to the next hop.

7. If the REQ/RES=1, ALLOC/DE-ALLOC=0 and the MAX/MIN/BEST=10, and if the flow has been allocated the minimum QoS requirements already, the IP options field is ignored, and is just used to refresh the minimum resources that have already been allocated for the resources. If however, the flow had been considered best effort so far, (which is indicated by the absence of this flow information from the flow table) the node invokes the resource allocation block in an attempt to reserve resources for the minimum QoS requirements. If the minimum QoS requirements are satisfied, then the flow table is updated and the IP options field is retained in the datagram and sent to the next node. If the minimum requirements are not satisfied, then the node sets MAX/MIN/BEST=11 and the flow is continued to be considered as best effort.

8. If a particular wireless link on the path from the source to the destination breaks, then the upstream node on the link that was broken does a flow recovery. All the flows that were allocated the maximum requirements and were being switched at the link level, now need to be reassembled at the recovery node. This node now has the new next hop for the destination, and determines if the link can satisfy the maximum QoS requirements. If so, it sets the MAX/MIN/BEST=01 and sends it to the next node on the default VPI/VCI to this node. However, if the requirements are not satisfied, it sets REQ/RES=0 and MAX/MIN/BEST=10 or 11 based on whether the minimum requirements are satisfied or not.

9. When an intermediate node receives an IP datagram that is NOT destined to it, with REQ/RES=1 and MAX/MIN/BEST=01, it infers that this IP datagram is re-directed from a recovery node (since the IP datagram with this option is normally reassembles only at the destination). The flow admission control determines if the resources have already been allocated for this flow. This is done to ensure that a node in the old path that receives a datagram from this flow continues to send the packets using the established VPI/VCI to the destination. However, if the REQ/RES=1 and MAX/MIN/BEST=10 or 11 (since some node in the new path did not have the required resources), the node on the old path de-allocates the additional resources and sends towards the destination. The destination as always sends a QoS report message to the source. The actions of the source have already been described in the previous section.

The state diagram for the intermediate node is show in Figure 3.9.

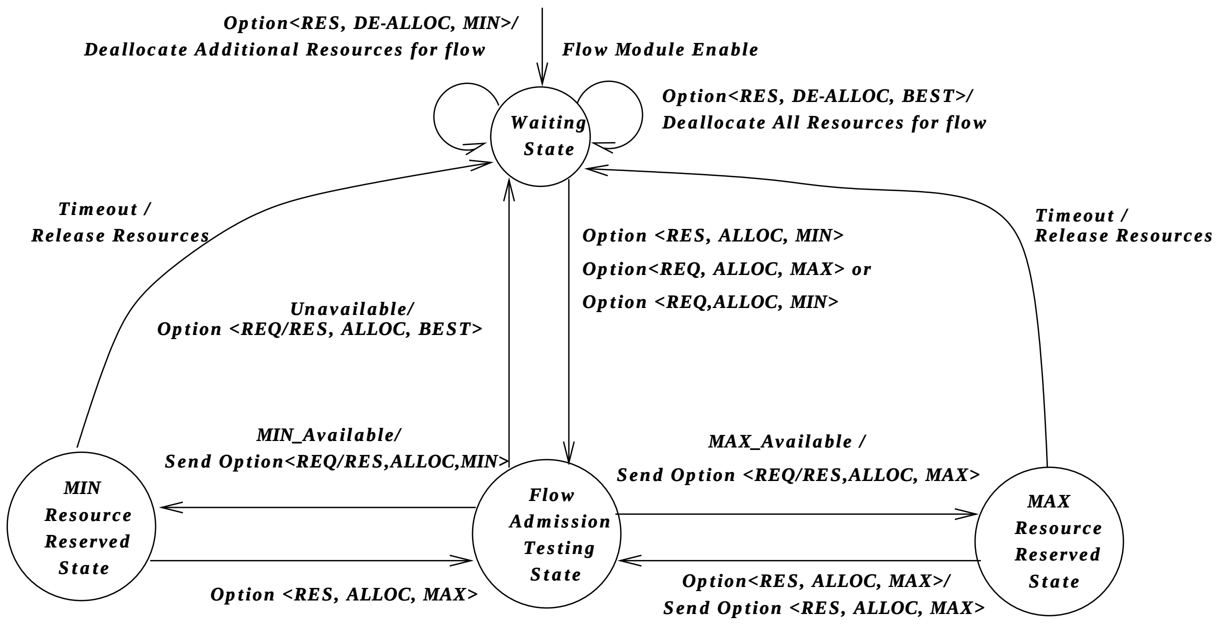
Option<RES, DE-ALLOC, MIN>/
Deallocate Additional Resources for flow
Flow Module Enable
Option<RES, DE-ALLOC, BEST>/
Deallocate All Resources for flow
Waiting State
Timeout /
Release Resources
Option <RES, ALLOC, MIN>
Option<REQ, ALLOC, MAX> or
Option <REQ,ALLOC, MIN>
Timeout /
Release Resources
Unavailable/
Option <REQ/RES, ALLOC, BEST>
MIN_Available/
Send Option<REQ/RES,ALLOC,MIN>
MAX_Available /
Send Option <REQ/RES,ALLOC, MAX>
MIN Resource Reserved State
Flow Admission Testing State
MAX Resource Reserved State
Option <RES, ALLOC, MAX>
Option<RES, ALLOC, MAX>/
Send Option <RES, ALLOC, MAX>

Figure 3.9: Intermediate node - State diagram

<u>Destination Node</u>

The destination node, on receiving the IP datagram with the options field, first invokes the resource allocation manager to determine the admissibility of the flow. On acceptance, it sets up the flow table and the switching table appropriately. The destination sends periodic QoS reports to the source. As already mentioned, the path taken by the QoS report messages from the destination to the source might be different from the path that has been set up from the source to the destination. QoS report messages are treated as best effort by all the nodes. Each QoS report message could take a different path to the source flow. This QoS report consists of the IP options field that it received from the source. For the flows that have their maximum end to end requirements satisfied, the destination node does the re-assembly of the cells into a datagram and extracts the IP options field.

The state diagram for the destination of the flow is shown in Figure 3.10.

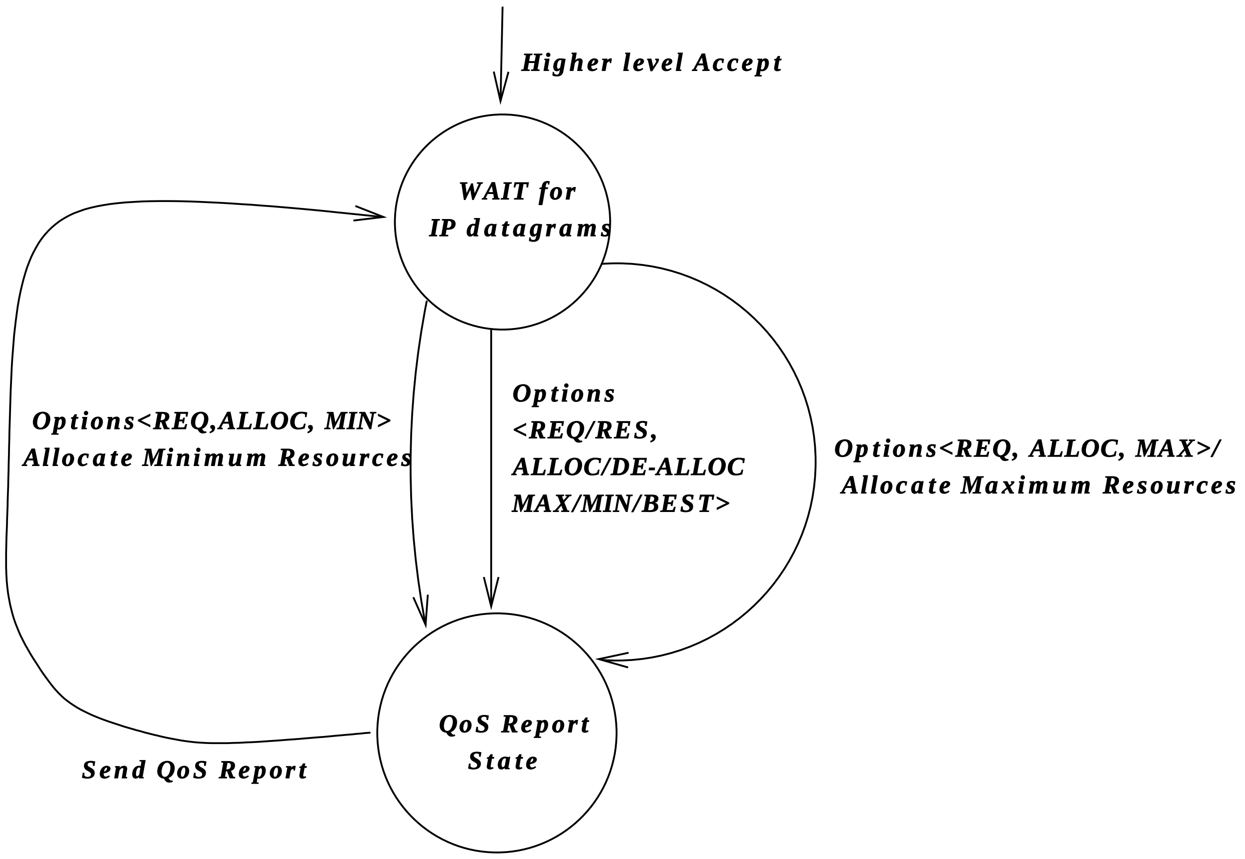
Higher level Accept
WAIT for IP datagrams
Options<REQ,ALLOC, MIN> Allocate Minimum Resources
Options <REQ/RES, ALLOC/DE-ALLOC MAX/MIN/BEST>
Options<REQ, ALLOC, MAX>/ Allocate Maximum Resources
QoS Report State
Send QoS Report

Figure 3.10: Destination node - State Diagram

Having discussed the flow management block and the application block, let us now discuss the rest of the blocks in the RDRN QoS architecture.

3.3.3 Network Layer

This section discusses the network block in the RDRN QoS architecture. The network block in turn consists of the *IP* layer, the *Classical IP over ATM (CLIP)* layer, the *switching* layer, the *ATM Adaptation Layer (AAL)* and the *Segmentation and Re-Assembly (SAR)* layer.

<u>3.3.3.1 IP Layer</u>

At the source, the IP layer is responsible for adding the RDRN QoS options to the packets that are sent to the destination. The current value of the RDRN options field is obtained from the flow management block. At the source, the intermediate MAPs and the destination, the IP layer is responsible for invoking the flow management block when an IP packet is received with the RDRN options.

3.3.3.2 CLIP and Switch

At the source node, CLIP is responsible for mapping the flows to a specific VPI/VCI that is assigned for the flows by the flow management block. It then sends the packets down to the AAL layer. At the intermediate nodes, the switch looks up the switching table that is set up by the flow management block, and switches the cells accordingly.

3.3.3.3 ATM Adaptation Layer (AAL)

At the source, the AAL is responsible for adding a Cyclic Redundancy Check (CRC) to the packets received from CLIP. After adding the trailer, it sends the AAL PDU to the SAR layer. At the destination, the AAL simply re-assembles the AAL PDU and performs the CRC check. It then hands over the packet to CLIP, which in turn hands it over to IP.

3.3.3.4 Segmentation and Re-Assembly (SAR)

At the source, the SAR layer is responsible for segmenting the AAL5 PDUs into cells and sending the train of cells down to the scheduling layer in the architecture. At the destination, the SAR layer is responsible for re-assembling the cells to form an AAL5 PDU and hand it over to the AAL layer.

3.3.4 QoS Layer

This section discusses the scheduling layer in the RDRN QoS architecture. This layer comprises the classifier, traffic shaper and the scheduler. This layer provides all the functionality needed to do service differentiation in the RDRN network.

3.3.4.1 Classifier

The classifier is responsible for classifying the trains of cells received from the SAR layer into the queues that are maintained for the various classes. The classifier will be set up by the flow management block for classifying based on the TOS byte.

3.3.4.2 Traffic Shaping

The traffic-shaping layer will shape the traffic to the rate specified in the flow specification. This too, will be configured by the flow management block. The traffic shaper will regulate the rate at which the cells are sent out of the interface. At the source, the shaper will shape traffic from a flow, while at the switch, the shaper will shape all the traffic that belongs to a particular class, i.e., it shapes traffic aggregates.

3.3.4.3 Scheduler

The scheduler is responsible for scheduling the transmission of the cells on the physical interface. The requirements from the scheduler include the following:

1. It should provide support for configuration of the peak cell rate for the various classes.
2. It should assign bandwidth to the various classes in proportion of the available bandwidth, i.e., it should do weighted round robin
3. The scheduler should do the scheduling only in the event of network congestion. If the network is not congested, it should make the entire bandwidth available to a flow.

Based on these requirements, a scheduling algorithm has been proposed. The scheduling algorithm is discussed in the next chapter on implementation.

This concludes the discussion on the proposed RDRN QoS architecture. The main functional unit in the RDRN QoS architecture is the flow management block. The functionality of the flow management block was discussed in detail. The in-band flow establishment protocol, which is the focus of this thesis, was discussed in detail. The flow establishment protocol is suited for highly dynamic networking environments like RDRN because of the following reasons:

1. The *in-band signaling* ensures efficient utilization of the available bandwidth. The out-of-band connection establishment and signaling done in typical wired ATM networks is not suitable in the highly dynamic mobile scenario because the resources available in a mobile environment are scarce. Out-of-band signaling involves a lot of additional overhead in establishing the connection. Also, handoff of a mobile node results in a number of messages being exchanged for the restoration of the connections to and from the node. Since the networking environment is very dynamic, there would be a number of such messages.
2. The protocol makes use of the features of both IP and ATM. *The robust and connectionless* nature of IP makes it appropriate for handling the dynamic nature of the network. Thus, the *end-to-end flow establishment* is done at the IP layer. ATM is used as the link level protocol because of its *low serialization delays* on moderate-speed links. ATM also offers the well-known constant bit rate (CBR), variable bit rate (VBR), available bit rate (ABR) and unspecified bit rate (UBR) services. Once the flow is

established from the source to the destination at the IP layer, the packets are switched at the link level for simplicity and speed.

Thus the proposed QoS architecture is highly suited for highly dynamic networking environments like RDRN. The next chapter discusses the issues involved and the details of the implementation of the RDRN QoS architecture described in this chapter.

Chapter IV

Implementation

This chapter discusses the implementation details of the in-band flow establishment protocol for end-to-end Quality of Service (QoS) in Rapidly Deployable Radio Networks. The implementation details are discussed on the following basis. First, the details involved in the implementation of QoS in the RDRN nodes are discussed. This is followed by the implementation details of the flow management module discussed in the third chapter.

4.1 Quality of Service in RDRN

This section discusses the details involved in the implementation of QoS in the RDRN network. This section consists of two sub-sections. The first sub-section discusses the QoS support that is available in the Linux operating system. The second sub-section discusses the implementation of QoS in the RDRN Protocol Stack.

4.1.1 QoS Support in Linux

This sub-section discusses the QoS support that is available in the Linux Operating System. This discussion is followed by a description of the problem involved in making use of these features in the RDRN protocol stack.

4.1.1.1 Linux QoS – Basic Philosophy

The basic traffic control functionality in Linux is shown in Figure 4.1. The de-multiplexing unit examines the incoming packet to determine if it destined for the local host. If they are destined for the local host, they are sent to the higher layer (namely TCP/UDP) for further processing. If the packet is destined for another host, they are sent to the forwarding component. The forwarding component may receive packets from both the higher layer as well as the input de-multiplexing unit. It looks up the routing table and determines the next hop for the packet. After determining the next hop for the packet, it queues the packet at the output interface for transmission. It is at this point that Linux traffic control comes into play. It is at this point that the various scheduling algorithms to schedule the transmission of packets are applied.

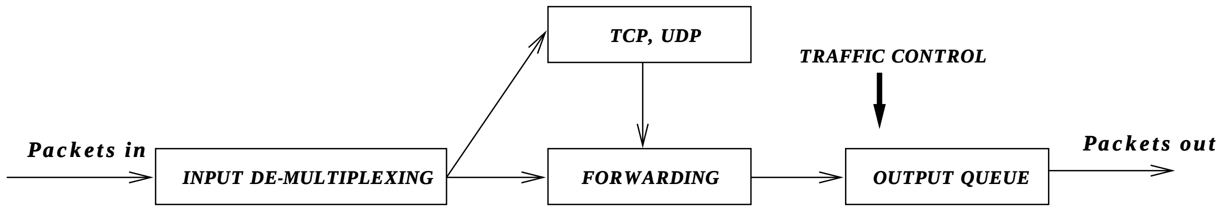

Figure 4.1: Linux Traffic Control

The Linux traffic control implementation is based on the following three basic blocks, namely:

1. Queuing disciplines
2. Classes
3. Filters

<u>Queuing disciplines</u>

Each output device has a queue associated with it. There are 11 different types of queuing disciplines supported in Linux, including:

1. Class Based Queue (CBQ) [32]
2. Token Bucket Flow (TBF) [33]
3. Clark-Shenker-Zhang Scheduler (CSZ) [6]
4. First In-First Out (FIFO)
5. Priority FIFO
6. True-Link Equalizer (TEQL)
7. Stochastic Fairness Queuing (SFQ) [34]
8. Asynchronous transfer mode Queuing discipline (ATM)
9. Random Early Detection (RED) [35]
10. Generalized Random Early Detection (GRED)
11. Differentiated Service (DSMARK)

Queues are identified by a handle <*major number: minor number*>, where the minor number is zero for queues. This handle is used when classes are to be associated to the queues.

Some queuing disciplines use filters to classify the packets into different classes and process them accordingly. That is, it provides priority to certain classes over other classes (e.g. FIFO, CBQ).

Queuing disciplines and classes are tied to one another. The presence of classes and their semantics are fundamental properties of the queuing discipline. In contrast, filters can be arbitrarily combined with queuing disciplines and classes.

Figure 4.2 shows an example queue. In this example, there are two types of queuing disciplines, one for high priority and one for low priority. The filter selects the one for high priority, while the remaining is treated low priority. The low priority packets are served by a FIFO queue, while a Token Bucket Flow (TBF) serves the high priority packets. The high priority flow is shaped to 1Mbps so that the low priority packets are not starved. One of the most important features of the QoS support in Linux is the flexibility with which a complex link-sharing structure can be setup. There could be any number of classes and queues arranged in a hierarchical manner.

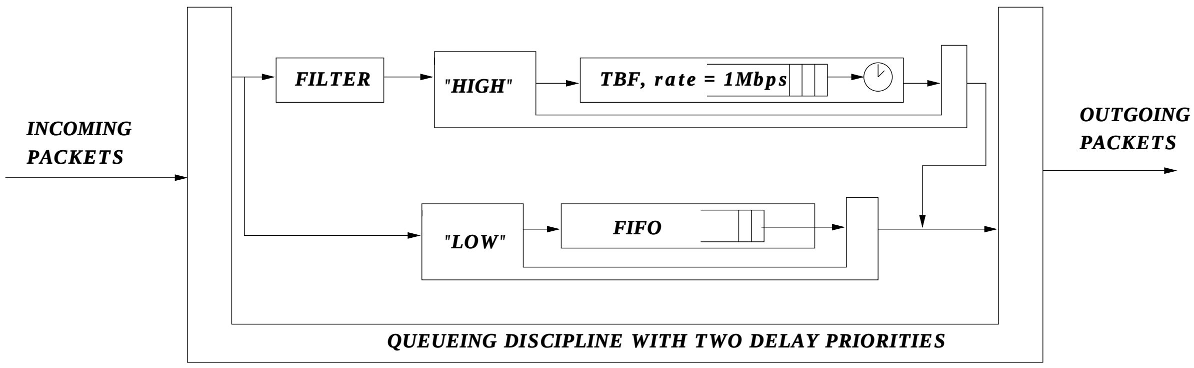

Figure 4.2: An example queuing discipline setup

Each queuing discipline provides a set of functions to control its operations. These functions are summarized below:

1. *Enqueue* - enqueues a packet with the queuing discipline. Packets are enqueued in the following manner. When the enqueue function of a queuing discipline is called, it runs all the filters one by one until a match occurs. Once the match occurs, the enqueue function of the queuing discipline "owned" by that class is executed.

2. *Dequeue* - dequeues a packet for sending. It returns the next packet that is eligible for sending on the device.
3. *Requeue* - requeues a packet for transmission. After dequeueing the packet, if for some reason, the packet is not transmitted, the packet needs to be put back in the queue at the same position where it was dequeued from.
4. *Drop* - drops a packet from the queue.
5. *Init* - Initialize the parameters for the queue, and configures it.
6. *Reset* - Resets the queue to the initial state. Clears all the queues and resets all the timers.
7. *Destroy* - Removes the queuing discipline. It removes all the filters and classes and cancels all the pending events.
8. *Dump* - Returns diagnostic data for maintenance

Classes

Queues and classes are tied to one another. Each class owns a queue. That is, when the enqueue function of a class is called, the enqueue function of the queuing discipline owned by that class is called. However, not all queuing disciplines have classes associated with them.

Classes are identified by the *class ID* and the *internal ID*. The class ID is assigned by the user while the internal ID is assigned by the queuing discipline. Like queue handles, the class ID is also structured in the form of a <*major number: minor number*>. The major number corresponds to their instance of the queuing discipline The minor number identifies the class within that instance.

Classes offer a set of functions to control its operations. These functions are summarized below:

1. *Graft* - This is used to attach a new queuing discipline to the class. It returns the old queuing discipline.
2. *Get* - looks up the class by its class ID and returns the corresponding internal ID.
3. *Put* - class previously referenced by get is de-referenced by calling *put*.

4. *Change* - change the properties of a class.
5. *Delete* - delete a class associated with the queuing discipline. It checks if the class is not in use before deleting it.
6. *Walk* - This iterates over all the classes in the queuing discipline. It invokes a call back function for all the classes. This is used to obtain diagnostic data from each of those classes.
7. *Tcf_chain* - This returns a pointer to the list of filters associated with the class.
8. *Bind_tcf* - This is used to bind a filter to the class. Similar to get, except that the queuing discipline will refuse deletion of the class when associated with a filter.
9. *Unbind_tcf* - This is used to unbind a filter from the class.
10. *Dump* - Used to dump statistics.

<u>Filters</u>

Filters are used to assign incoming packets to one of the configured classes, based on certain information available in the packet, for e.g., the destination IP address, destination port number, protocol etc. There are two types of filters based on the scope of the packets they classify, namely *Generic filters* and *Specific filters.*

1. *Generic filters*: When only one instance of a filter per queuing discipline is needed to classify packets for all classes, then the filter is referred to as a g*eneric filter*. These filters take the class ID from the packet descriptor, where it was stored before by some other entity in the protocol stack. The route classifier and the firewall classifier are examples of generic filters. They are illustrated in Figure 4.3.

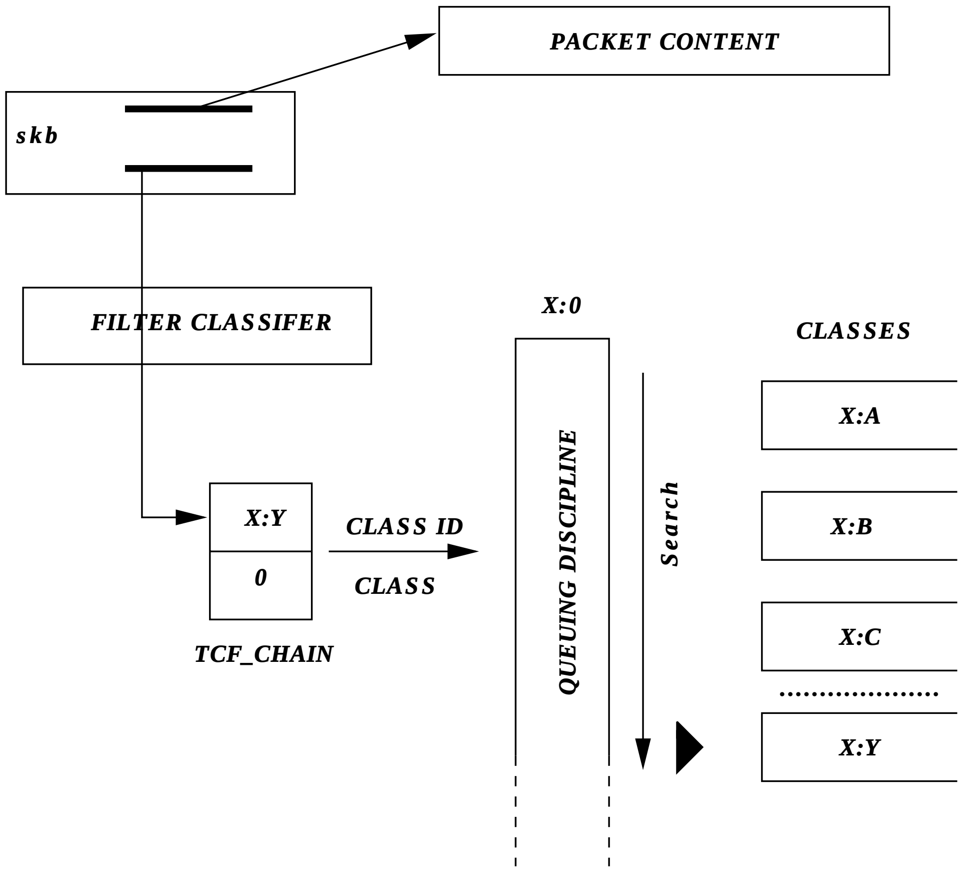

Figure 4.3: Generic Filters

2. *Specific filters*: When more than one instance of a filter is needed per queuing discipline to classify packets to all classes, then the filter is referred to as a *specific filter*. Since specific filters have at least one instance or element per class, they can store the internal ID of that class and provide this as a result of the classification, thereby making the process of classification faster. This is illustrated in Figure 4.4, where a pointer to the class structure is used as the internal ID. The *RSVP* classifier, the *u32* classifier and the *tcindex* classifier are examples of specific filters.

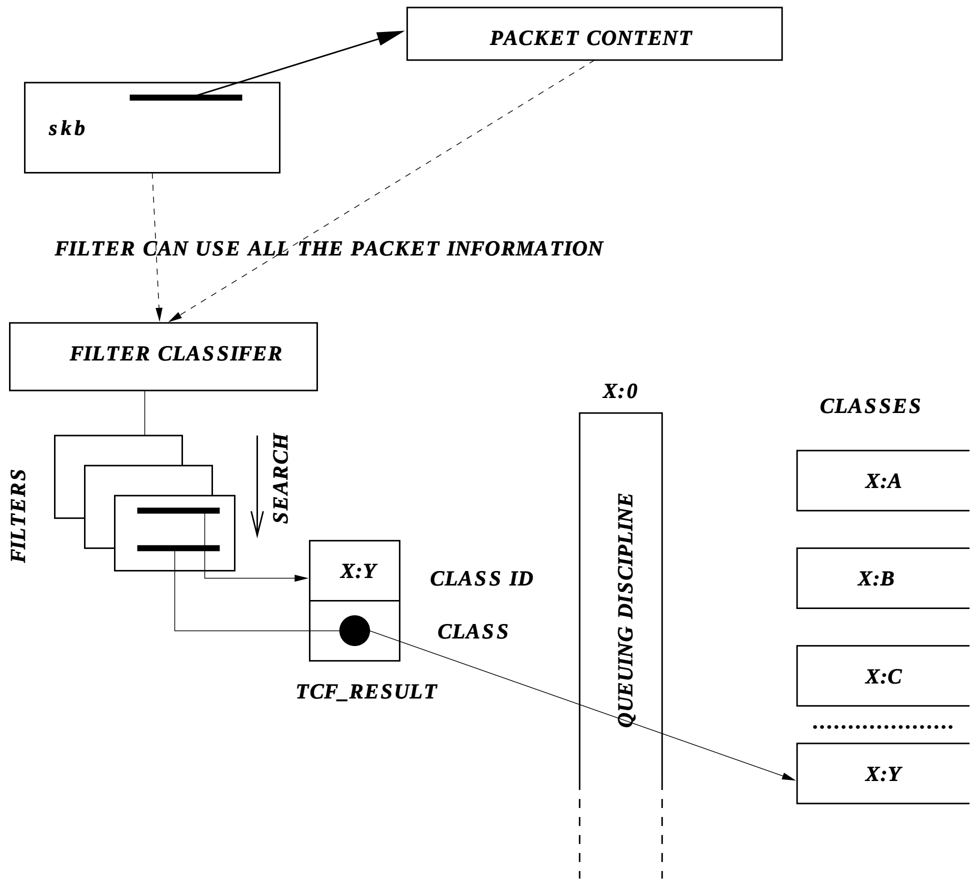

Figure 4.4: Specific filters

4.1.1.2 RDRN Protocol Stack

This sub-section discusses the problems involved in using the QoS support available in Linux for the RDRN network. The RDRN protocol stack was discussed in Chapter 1. Figure 1.2 shows the RDRN protocol stack. The stack consists of ATM over IP.

The QoS support in Linux consists of different schedulers for the various queuing disciplines. These schedulers depend heavily on timers and end-of-event signals given by the device driver for the network interface card. In the protocol stack, they operate just below the IP layer, before any device specific operations are performed. Implicitly, the design of the scheduler assumes that the device driver layer exists immediately below the IP layer. However, in the RDRN protocol stack, a Classical IP over ATM (CLIP) layer is required to map the IP traffic on to a VPI/VCI. The CLIP layer is merely a tunnel and not a device, as a result of which it does not generate the end-of-event signals. Thus, from the perspective of the RDRN protocol stack, the design of the scheduler is broken. This in turn suggests that the QoS support in Linux cannot be made use of in the RDRN protocol stack. In other words, the Linux QoS scheduler is at the wrong layer in the RDRN protocol stack.

4.1.2 QoS Layer in RDRN

This section discusses the details involved in the implementation of QoS in RDRN. As mentioned in the previous section, traditional schedulers implemented in Linux exist at the wrong layer. A scheduler which will schedule the transmission of cells will be appropriate for the RDRN protocol stack. Thus, the revised RDRN protocol stack is shown in Figure 4.5, in which the QoS layer exists below the SAR layer. The QoS layer, as discussed in Chapter 3, will perform classification, traffic shaping and scheduling. Classification is the process of identifying the cells and putting them in the queue for the appropriate class. Traffic Shaping is the process of regulating the flow of traffic to conform to certain traffic specifications. Scheduling is the process of scheduling the transmission of cells based on the assigned priorities. A *weighted round robin (WRR)* scheduling algorithm has been implemented for scheduling the transmission of cells.

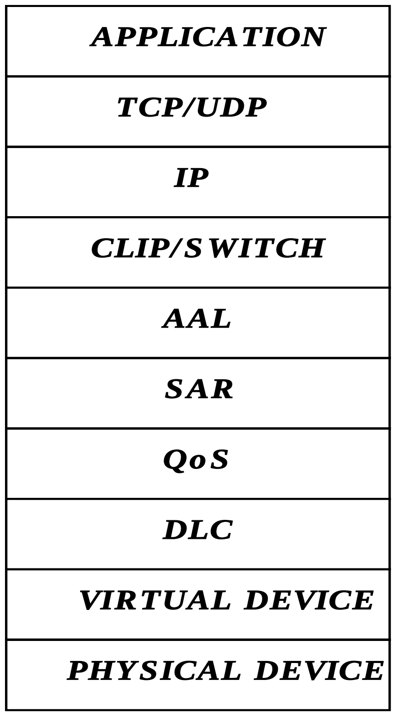

Figure 4.5: Revised RDRN Protocol Stack

The queuing structure that is used for the implementation of the classifier, traffic shaper and the scheduler is shown in Figure 4.6. As shown in the figure, the classifier will receive a train of cells from the SAR layer. It then classifies the train of cells into the various queues depending on the class information that it available in the data structure received from the SAR layer. There are three types of queues that are supported currently:

1. *High Priority:* This type of queue is intended to support applications that require a constant bit rate (CBR) service. This service will continue to get the required bandwidth irrespective of the existing network conditions. In other words, even during network congestion, the high priority flows receive the requested resources. However, it is not a loss-less service. In times of congestion, some cells may be dropped.
2. *Medium Priority:* This type of queue is intended to support applications that require a variable bit rate (VBR) service with higher reliability. The throughput of the medium priority flows is affected by the existing network conditions. That is, in times of congestion, the resources allocated for the medium priority flows may fluctuate. The medium priority flows, however, receive a loss-less service. Even in times of congestion, no cells are dropped from the medium priority flows. In other words, this service emulates the non-real time VBR services.

3. *Low Priority:* This type of queue is intended to support applications that do not have any specific QoS requirement i.e., best effort or Unspecified Bit Rate (UBR) service. The cells from a low priority flow will be dropped during network congestion, and at the same time, there are no guarantees about the available bandwidth.

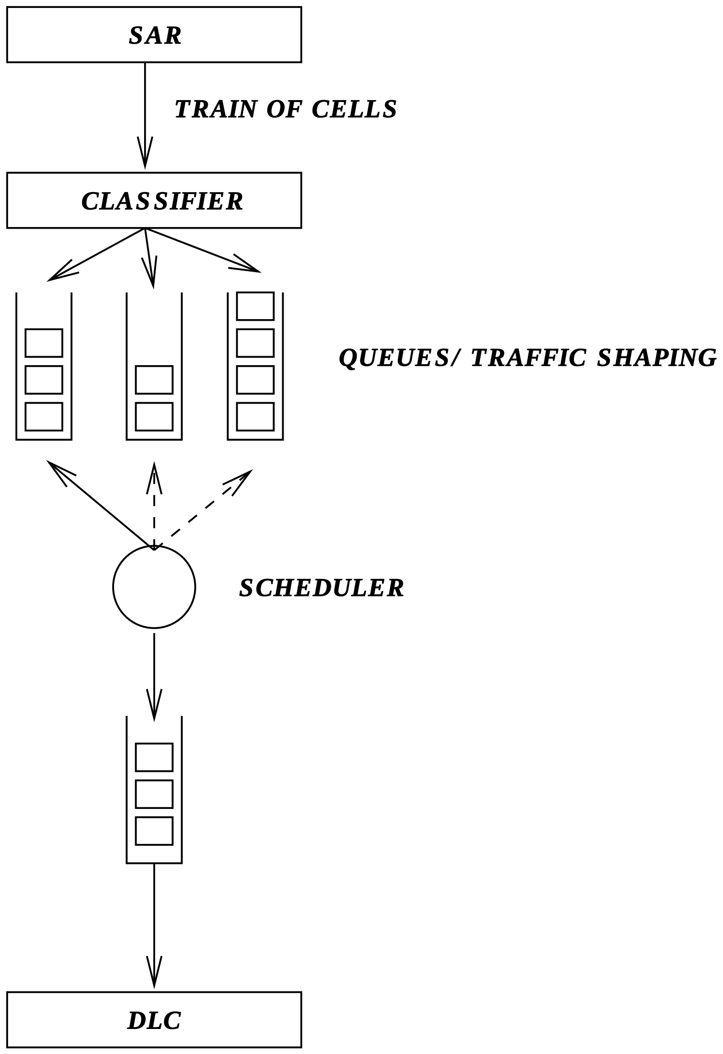

Figure 4.6: RDRN QoS Layer

The scheduler uses a *weighted round robin (WRR)* algorithm to schedule the transmission of cells on the output interface. The algorithms used in the QoS layer are discussed below.

Let the high priority queue be represented by $Q_{H.}$ Let the medium priority queue be represented by Q_M. Let the low priority queue be represented by Q_L. Let the peak cell rate corresponding to the three queues be represented by PCR_H, PCR_M and PCR_L (in cells per second) respectively and let PCR represent the maximum allowable rate on the output interface in cells per second. Let the weight with which the bandwidth is allocated be represented by W_H:W_M:W_L, which is configurable by the user while creating the QoS layer. Let the incoming cell be represented by C_I. The data structure containing the cell consists of a priority field that is used to identify the priority level to which the cell is assigned. Let this be represented by C_{ID}. There are essentially two modules in the QoS layer, one to classify and queue the incoming cells and one to schedule the transmission of the cells.

The classifier module processes an incoming cell C_I in the following manner:

1. Examine the C_{ID} to determine the priority assigned to the cell.
2. If C_{ID} is set to 1, enqueue the cell in Q_H and return.
3. If C_{ID} is set to 2, enqueue the cell in Q_M and return.
4. If C_{ID} is set to anything other than 1 and 2, enqueue the cell in Q_L and return.

This summarizes the functions of the classifier. As seen from the algorithm, the classifier identifies cells based on C_{ID}. Thus it classifies cells based on the classes, and not based on individual flows. One of the issues that need to considered here is the higher layer entity that actually sets the value of C_{ID}. This is dealt with in Section 4.2.2.

The scheduler will be invoked periodically. Let T be the time interval between two successive calls to the scheduler. T is usually of the order of milliseconds, given that the smallest resolution possible on the Linux operating system is 10 milliseconds. Based on the required peak cell rate and the value selected for T, the scheduler determines the number of cells to be transmitted each time it is invoked. The scheduler will work in the following manner:

1. Examine the queue Q_H. Calculate the number of high priority cells, represented by C_H, to be sent from the queue.

$$C_H = \frac{T \text{ (in milliseconds)}}{1000} * PCR_H$$

2. Examine the queue Q_M. Calculate the number of medium priority cells, represented by C_M, to be sent from the queue.

$$C_M = \frac{T \text{ (in milliseconds)}}{1000} * PCR_M$$

3. Examine the queue Q_L. Calculate the number of low priority cells, represented by C_L, to be sent from the queue.

$$C_L = \frac{T \text{ (in milliseconds)}}{1000} * PCR_L$$

4. If C_H=0, then allocate the available bandwidth, represented by B_A, based on the ratio of the weights.

$$C_M = C_M + \frac{W_M * B_A}{(W_H+W_M+W_L)}$$

$$C_L = C_L + \frac{W_L * B_A}{(W_H+W_M+W_L)}$$

5. If C_M=0, then allocate the available bandwidth B_A, based on the ratio of the weights.

$$C_H = C_H + \frac{W_H * B_A}{(W_H+W_M+W_L)}$$

$$C_L = C_L + \frac{W_L * B_A}{(W_H+W_M+W_L)}$$

6. If C_L=0, then allocate the available bandwidth B_A, based on the ratio of the weights.

$$C_H = C_H + \frac{W_H * B_A}{(W_H+W_M+W_L)}$$

$$C_M = C_M + \frac{W_M * B_A}{(W_H+W_M+W_L)}$$

7. Remove C_H cells from Q_H, C_M cells from Q_M and C_L cells from Q_L. Send these cells to the DLC layer for transmission.

This summarizes the functions of the scheduler. It should be noted that the scheduling algorithm also results in shaping the traffic that is sent out from the output interface based on the peak cell rate (PCR) allowable for the various queues. At the same time, it also maintains the priorities between the cells in the different queues. Having looked into the implementation details of the QoS layer in the RDRN protocol stack, let us now discuss the details of the flow management module in the proposed QoS architecture for RDRN.

4.2 Flow Management Module

The implementation of the flow management module will be divided into three sections. The first section discusses the API that is available for specifying the QoS requirements of the flow. The second section discusses the QoS mapping element in the flow management module. The last section discusses the details involved in the implementation of the in-band flow establishment protocol.

4.2.1 Application Programmer Interface

This section discusses the API that is available for specifying the QoS requirements of the flow. As already discussed in the third chapter, the flow establishment process is done by sending the *RDRN QoS* option in the IP header. The application programmer will be provided with a '*setsockopt*' call that will used to set the RDRN QoS options on the IP header. The flow specification will be specified as part of the system call. The data structure used for the flow specification:

```
struct end2end_qos {
        int application_type; /* This will one of real time/non-real time */
       int throughput;         /* This will be translated to the peak cell rate */
      int priority;               /* This indicates the priority that will be given to the flow */
};
```

An example code using the *setsockopt* call to set up the flow with the required resources is given in Appendix A. It should be noted that the API specifies only the flow specification and does not contain any of the derived flow specific information. The derived flow specific information, which includes the wireless and mobile QoS parameters discussed in Chapter 3, is inferred from the application type in the flow specification. However, the current implementation does not make use of these parameters, because of the absence of a supporting handoff protocol.

4.2.2 QoS Mapping

As mentioned in Chapter 3, the flow management block at the source maps the QoS requirements of a flow into a Type-of-Service (TOS) byte in the IP header, to indicate the requirements of the flow to the intermediate nodes. This section discusses the mapping of the requirements specified by the user into a TOS byte in the IP header. The QoS mapping block in the flow management module maps the requirements specified by the application layer in the form of a flow specification to a TOS byte, based on the following criteria:

1. If the type of application is real-time, then:
   - If the throughput requirements of the flow can be satisfied, then the flow is assigned a priority of 1. This enables the intermediate nodes to identify the cells belonging to this flow as high-priority real-time flow. It also sets the C_{ID} for all the cells belonging to this flow to 1, so that the classifier in the QoS layer can queue the cells in the appropriate queue.
   - If the throughput requirements of the flow cannot be satisfied, then the flow is assigned a priority of 0 to indicate that the flow will be considered as best effort. Note that this flow is not assigned the next priority level of 2, since this level is meant for flows that require a loss-less service, with no specific requirement on the end-to-end delay. However, as mentioned in the flow establishment algorithm in Chapter 3, an attempt will be made to scale up to the maximum requirements, when the resources become available.
2. If the type of the application is non-real time, then:
   - If the throughput requirements of the flow can be satisfied, then the flow is assigned a priority of 2 to indicate to the intermediate nodes that the cells belonging to this flow should not be dropped. It also sets the C_{ID} for all the cells belonging to this flow to 2, so that the classifier in the QoS layer can queue the cells in the appropriate queue.
   - If the throughput requirements of the flow can be satisfied only *partially*, even then the flow is assigned a priority of 2, and the available bandwidth is allocated to the flow. This is because the flows belonging to this class need strict guarantees on reliability, while the throughput is not a very important end-to-end QoS requirement. It sets the C_{ID} to 2 so that the classifier can classify the cells belonging to this flow.

- If the throughput requirements of the flow *cannot* be satisfied *at all,* the flow is assigned a priority of 0, that is, it is considered best effort. However, an attempt is made to scale up to the maximum requirements when the resources become available.

This summarizes the mapping functionality in the flow management block. The QoS mapping block is responsible for the configuration of the classifier in the QoS layer.

4.2.3 Flow Establishment Protocol

This section discusses the details involved in the implementation of the flow establishment protocol. The implementation of the flow establishment protocol required the following components:

1. Modification to Classical IP over ATM (CLIP) to support multiple flows.
2. Configuration and modification of the software ATM switch on the MAPs to support QoS.
3. Flow Establishment.

The rest of this section discusses these components in detail.

<u>4.2.3.1 CLIP Modification</u>

This section discusses the modifications that needed to be made to CLIP [31] to support multiple flows. Before looking into the details of the modifications that were made to CLIP, let us briefly discuss CLIP.

In an Ethernet LAN, the Address Resolution Protocol (ARP) takes the target IP addresses and searches for a target physical address, which is 48-bit MAC address. If the corresponding entry is not found the ARP module sends a broadcast into the LAN. The broadcast is the ATM Request. If one of the machines receiving the broadcast recognizes its IP address in the ARP request, it will return an ARP reply back to the inquiring host. The reply contains the Ethernet address of the queried host.

ATM is a Connection Oriented technology unlike Ethernet. ATM provides a Virtual Connection (VC) switched environment. The function of mapping user Protocol Data Units

(PDUs) into the information field of the ATM cell and vice versa is performed in the ATM Adaptation Layer (AAL). Since ATM is not a broadcast medium, it is not possible to broadcast ARP requests as in Ethernet. Hence, ATM makes use of an ATM ARP Server to resolve the IP addresses to ATM addresses. A Logical IP Subnet (LIS) is an administrative entity, which consists of hosts and routers. Hosts communicate directly via ATM to other hosts within the same LIS using the ATMARP service as the mechanism for resolving target IP addresses to target ATM endpoint addresses.

The ATMARP service has scope only within a Logical IP Subnet i.e., it serves all hosts that are within the same LIS. Communication to hosts located outside of the local LIS is provided via an IP router. In short, a host in a LIS uses CLIP to create a VC to another host in the same subnet. That is, there is a one-to-one mapping between an IP address and a VC assigned for the IP address. This is illustrated in Figure 4.7.

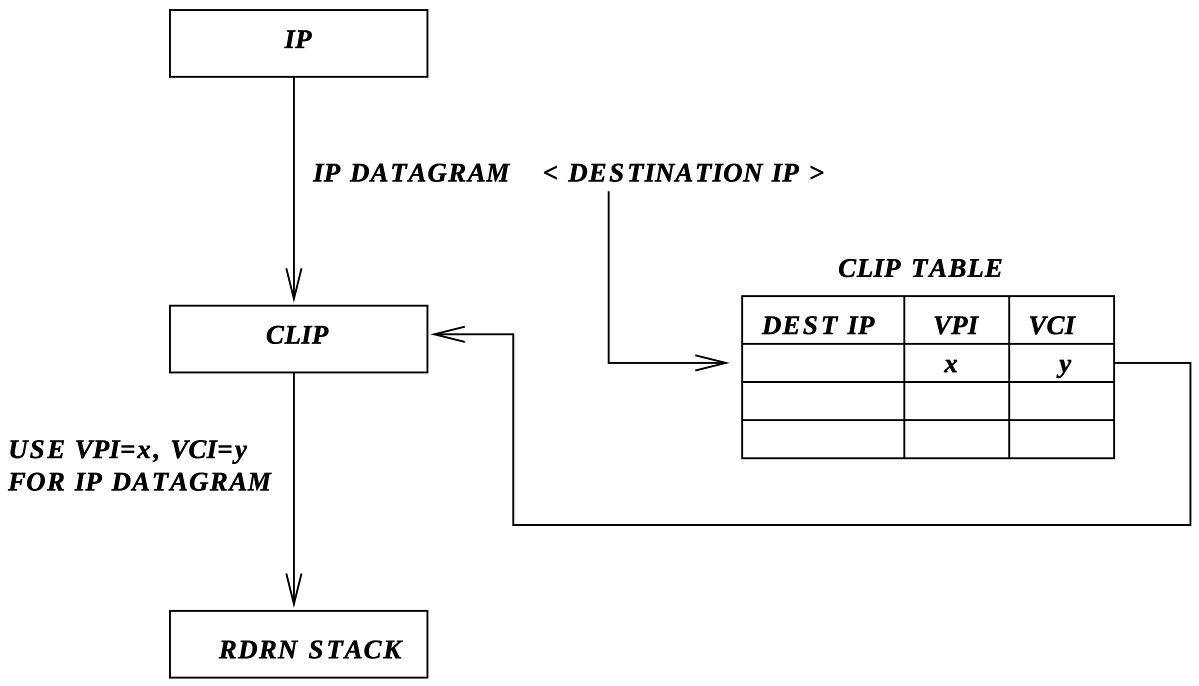

Figure 4.7: Classical IP over ATM

However, the proposed RDRN flow establishment protocol creates a VC for every flow from the host, as the QoS requirements could be different for different flows. Thus, CLIP needed to be modified to support multiple flows to a given IP address. The existing implementation uses the destination IP address available in the packet to obtain the corresponding VC. This needed to be modified to use the combination of the tuple <*Source IP address, Source port number, Destination IP address, Destination port number, Protocol*> (the filter specification discussed in Chapter 3) to obtain a VC. Thus, the CLIP table needed to be modified to include the additional fields. This is illustrated in Figure 4.8.

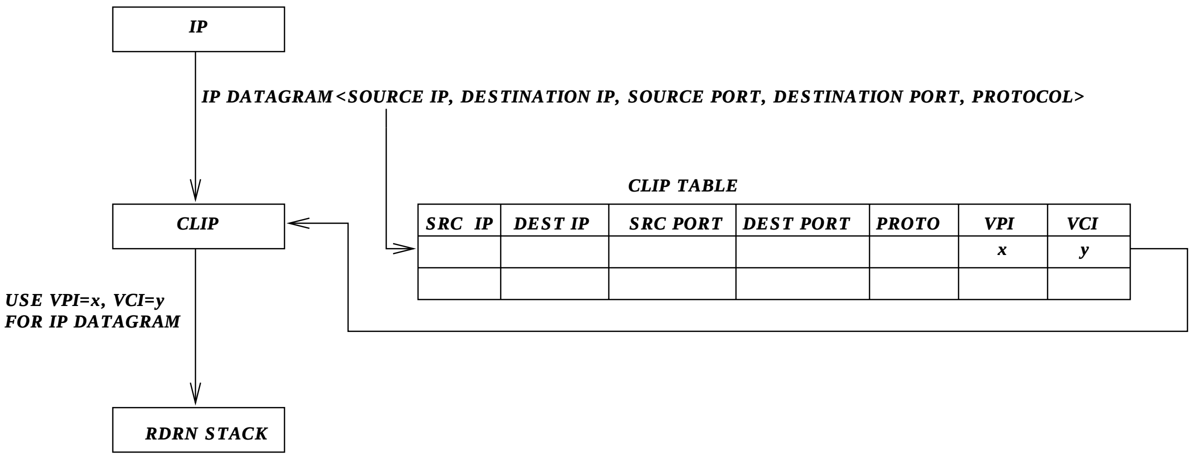

Figure 4.8: CLIP Modification

4.2.3.2 Configuration and Modification of the software ATM switch on the MAPs to support QoS

This section discusses the configuration of the software ATM switch during the flow establishment process and modifications that were made to the switch to support QoS. The software ATM switch maintains a switching table. The switching table consists of the *incoming interface, incoming VPI, incoming VCI, outgoing interface, outgoing VPI* and *outgoing VCI.* When the switch receives a cell from an incoming interface with a specific VPI/VCI, it looks up the switching table and determines the corresponding outgoing interface and VPI/VCI. It replaces the incoming VPI/VCI in the cell header with the outgoing VPI/VCI and sends it out on the outgoing interface. When the switch is configured to set up a switching table entry, the following functions are performed:

1. Two Permanent Virtual Circuits (PVCs) are created by the switch, one for the incoming VPI/VCI and one for the outgoing VPI/VCI.
2. One of the features of the switch is that it sets up the PVCs in both directions. That is, two switching table entries are set up, one to switch cells on the incoming interface with

the incoming VPI/VCI to the outgoing interface with the outgoing VPI/VCI, and one for the cells coming in the other direction too.

In Section 4.2.2, it was mentioned that the flow establishment protocol in the source sets the TOS byte in the IP header to indicate the flow's requirements to the intermediate nodes. The intermediate nodes will use the TOS byte in the IP header to identify the class to which the cells belonging to this flow need to be classified. The flow establishment block configures the switch to set up a switching table entry. At this point, in addition to specifying the incoming interface, incoming VPI, incoming VCI, outgoing interface, outgoing VPI and outgoing VCI, it also specifies the value of the TOS byte as set by the source. The switch then configures the classifier in the QoS layer to identify the cells belonging to this flow and put them in the appropriate queue. In this manner, the switch can be configured to support QoS. This summarizes the configuration and modification made to the software ATM switch to support QoS.

4.2.3.3 Flow Establishment

As discussed in Chapter 3, the flow establishment protocol requires different functionality at the source, destination and the intermediate nodes. This section discusses the implementation of these functional units in detail. It also makes a mention of the features of the flow establishment protocol that were described in the original design but were not implemented.

Source

As already mentioned in Section 4.2.1, an API is needed to specify the QoS requirements of a flow. This has been implemented with the help of a *setsockopt* system call, an example for which is specified in Appendix A. Based on the description of the algorithm given in Section 3.3.2.2, the source sets the fields in the RDRN QoS options field appropriately. It then sends this as an IP option field in all the IP datagrams belonging to this flow. When the source receives a QoS report message from the destination, it examines the RDRN QoS options field in the IP header to determine if the end-to-end flow has been established. It then sends the datagrams over the specific VC that has been setup for this flow.

If the RDRN options field in the QoS report indicates a broken flow, then a flow re-establishment needs to be performed. As part of the re-establishment process, the source starts sending the datagrams on the default VC. On completion of the flow re-establishment process, the source starts sending the datagrams once again on the specific VC. This again, is identified by the combination of the MAX/MIN/BEST and REQ/RES bits discussed in Chapter 3.

The features of the flow establishment protocol that were not implemented include the scaling up feature. In other words, the ALLOC/DE-ALLOC bits in the RDRN QoS options field is not utilized. However, the current implementation provides the infrastructure for the implementation of these features.

Intermediate Nodes

When an intermediate node receives an IP datagram with the RDRN QoS options field, it uses the filter specification to determine if the datagram belongs an existing flow. If it is a new flow, then it sets up a switching table entry, as specified in Section 4.2.3.2, and the filter specification is recorded in the state machine. There are two possibilities that arise based on the combination of the MAX/MIN/BEST and REQ/RES bits, as described in Chapter 3. The datagram could belong to a new flow that is in the process of flow establishment, in which case it is forwarded to the next hop, or it could belong to flow that is broken and is in the process of recovery. When the outgoing link at an intermediate node fails, it takes the following actions as part of the flow re-establishment procedure:

1. It deletes the default VC that was set up to the neighboring MAP on the failing link.
2. It deletes all the switching table entries corresponding to the failing interface.
3. It deletes all the Permanent Virtual Circuits (PVCs) that were created by the software ATM switch on the failing interface.
4. It keeps alive all the PVCs on the incoming interfaces (corresponding to the switching table entries that were deleted) for the purpose of re-assembling the IP packets for flow re-establishment.

When the switch receives cells for which there is no switching table entry, it reassembles the IP datagram and hands over the packet to IP. The flow management block then interprets the options field in the IP datagram and processes it as described in Chapter 3, to re-establish the flow.

All the features required from an intermediate node as part of the flow establishment protocol have been implemented.

Destination

As discussed in Chapter 3, only a limited functionality is needed at the destination for the flow establishment protocol. When the destination receives an IP datagram with the RDRN QoS option field, it uses the filter specification to determine if it belongs to an existing flow. If it is a new flow, it creates a PVC for the flow, and sends a QoS report back to the source on the default VC. It monitors the options field of all the datagrams belonging to this flow and sends a QoS report back to the source whenever there is a change in the options field, which indicates a change in the network conditions, e.g., a link failure.

This concludes the discussion on the details involved in the implementation of the QoS architecture for RDRN. The implementation broadly consisted of the flow management module in the QoS architecture and the QoS layer in the RDRN protocol stack. The performance of this architecture is evaluated in the next chapter on experimentation and results.

Chapter V

Experiments and Results

This chapter discusses the experiments that were conducted to test the validity and measure the performance of the proposed QoS architecture for RDRN. The blocks in the QoS architecture that need to be tested include the flow management module and the QoS scheduling layer in the RDRN protocol stack. Before going into the details of the test setup scenarios and the results, let us discuss the parameters that need to be monitored to measure the performance of the RDRN QoS architecture.

5.1 Test Parameters

This section discusses the metrics that need to be measured to test the validity and determine the performance of the proposed QoS architecture for RDRN. Broadly, there are two entities that need to be tested in the architecture. The first entity is the flow management module, while the second entity is the QoS layer in the revised RDRN protocol stack (Figure 4.5). From the flow management module's perspective, the metrics include:

1. *Flow Establishment time*: This will be the time taken to establish the end-to-end flow. The time interval between the transmission of the first packet and the receipt of the QoS Report message from the destination is referred to as the flow establishment time. This parameter will be measured for various numbers of Mobile Access Points from the source to the destination.
2. *Flow Re-establishment time*: This will be the time taken to establish the end-to-end flow from the source to the destination, in the event of an intermediate node losing connectivity to the next node in the path. This again will be measured for various numbers of Mobile Access Points from the source to the destination.
3. *End-to-end throughput with/without flow-establishment with/without mobility*: This set of experiments will be used in determining the efficiency of the proposed flow establishment protocol. End-to-end flows are set up using the flow establishment scheme, and the throughput is measured and compared with IP level forwarding of the packets. Again, the same set of experiments will be repeated with mobility, and the throughput is measured.

The set of tests conducted here is discussed in Section 5.3. Section 5.4 discusses the experiments conducted to test the scalability of the proposed flow establishment protocol. The validity of the QoS layer in the RDRN protocol stack will be tested by setting up multiple flows from the source to the destination, and measuring the throughput obtained for the various classes. The performance of the traffic belonging to the various classes during network congestion is also tested. The tests conducted for this purpose are discussed in Section 5.5. The next section discusses an example test setup to show the relationship between the physical and logical experimental configuration.

5.2 Test Scenarios

Before delving into the details of the test setup for the various metrics of interest, an example test environment is explained to indicate the relationship between the physical and logical experimental configuration. Let us consider a case where there are three nodes in the system: one MAP and two MNs. The two MNs are connected to each other via the MAP. The logical configuration is shown in Figure 5.1.

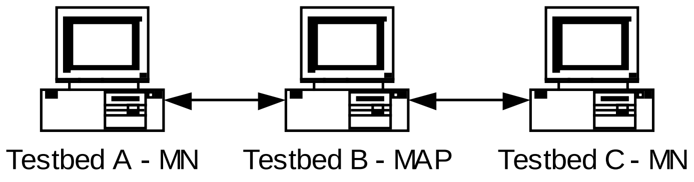

Figure 5.1: Logical Test Configuration

The physical setup consists of the Fore ATM Switch, which will be used to emulate the wireless link between the nodes in RDRN. The physical setup for the logical setup in Figure 5.1 is shown below in Figure 5.2.

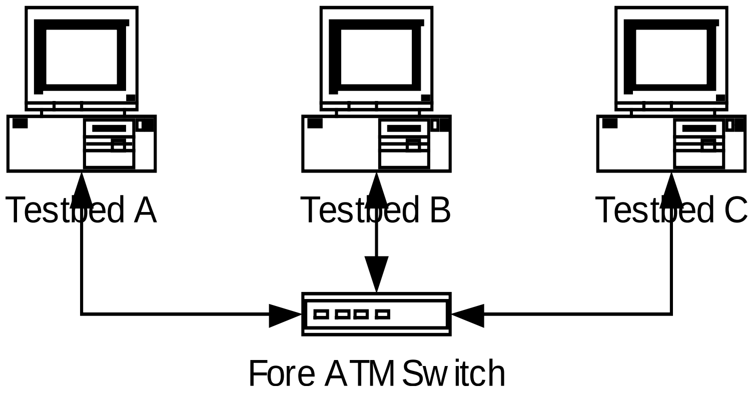

Figure 5.2: Physical Test Configuration

PVCs will be setup in the ATM switch to link the various testbeds together. The failure of a link is emulated by the deletion of the PVC from the hardware ATM switch. For the rest of this report, the logical connectivity will only be shown.

Having understood the relationship between the logical test configuration and the corresponding physical test configuration, let us now discuss the experiments conducted and the results obtained.

5.3 Flow Management Module

This section discusses the experiments that were conducted to test the performance of the flow management module in the RDRN QoS architecture. As mentioned in Section 5.1, the metrics that determine the performance of the flow management module are the flow establishment time, the flow re-establishment time and the obtained end-to-end throughput as compared to that obtained by doing simple IP level forwarding with and without mobility.

5.3.1 Flow Establishment Time

Test Configuration

This section discusses the experiments that were performed to determine the time taken to establish an end-to-end flow. The flow establishment time is expected to increase with the number of MAPs in the path from the source to the destination. The logical test configuration for this experiment is shown in Figure 5.3. In this configuration, flows are established from testbed A, which is the source, to testbed D, which is the destination. This is done using the flow establishment algorithm described in Chapter 3. The source uses the REQ/RES and the MAX/MIN/BEST bits in the RDRN QoS options field to set up the flow. It sends the datagrams on the default VC during this period. Once the flow is set up, the datagrams are sent on the specific VC and are thus switched. The time to establish the flow from the source to the destination is measured. The flow establishment time is the time interval between the transmission of the first packet from the source and the receipt of the QoS Report message from the destination. The test is performed for a number of MAPs between the source and the destination.

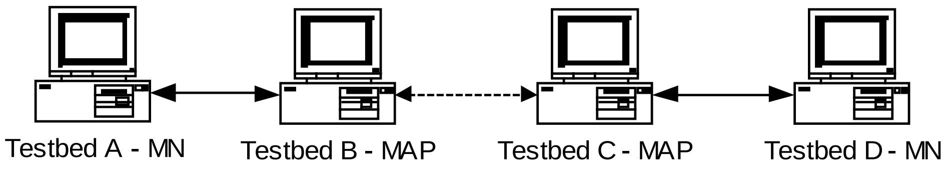

Figure 5.3: Logical Test Configuration to test flow establishment time

Result

The flow establishment times were measured for cases when the number of MAPs between the source and the destination is 1 through 4. The tests were repeated multiple times. The mean flow establishment time and the deviation from the mean have been plotted in Figure 5.4. The figure shows that the flow establishment time increases with an increase in the number of MAPs in the path between the source and the destination. The dot on the plot shows the mean flow establishment times, while the vertical line through it shows the deviation from the mean.

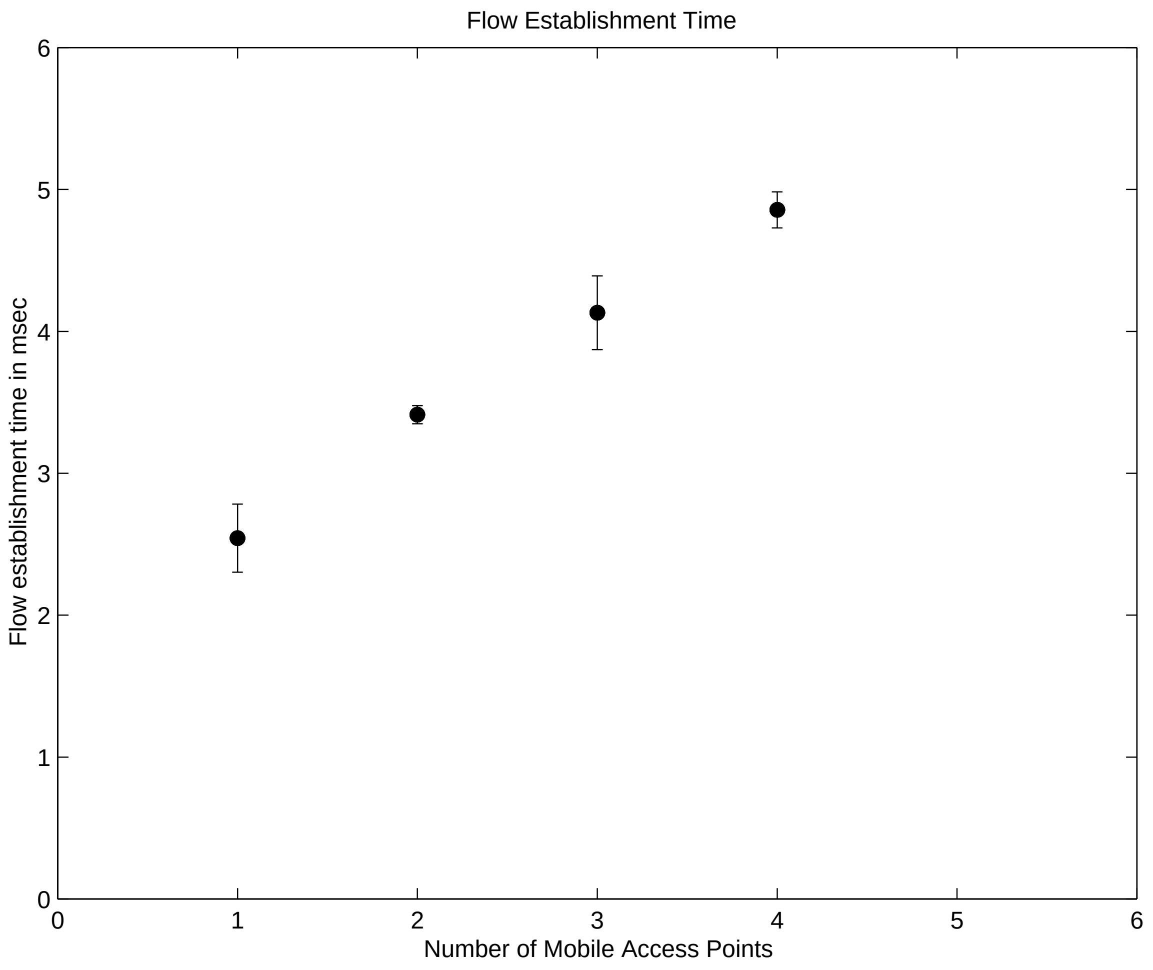

Figure 5.4: Flow Establishment Time

5.3.2 Flow establishment throughput

This section discusses the experiments that were conducted to compare the throughput that was obtained by using the flow establishment protocol as opposed to that obtained by doing simple IP level forwarding. Since the flow establishment protocol results in a switching table entry being set up in the intermediate MAPs, the intermediate MAPs need to only switch cells, and do not have to re-assemble the IP packets, as done in IP forwarding. As a result, the flow establishment protocol is expected to perform better than IP forwarding. The logical configuration for this experiment is shown in Figure 5.5. In this set up, flows are established from testbed A, which is the source, to testbed D, which is the destination, and the end-to-end throughput is measured. The throughput obtained is compared with the throughput obtained by doing simple IP level forwarding.

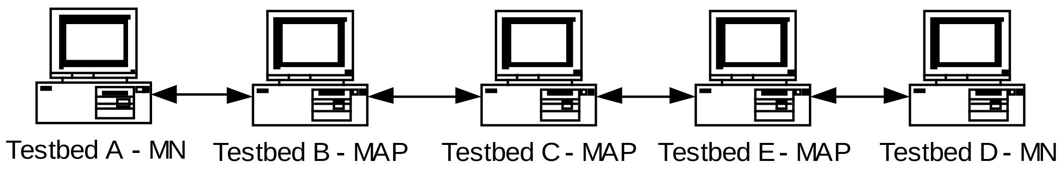

Figure 5.5: Test Configuration to measure throughput

Results

A flow was established from testbed A to testbed D and the end-to-end throughput was measured. The peak rate for all the output interfaces is configured to 10Mbits. As shown in Figure 5.5, there were 3 MAPs between the source and the destination. The tests were repeated multiple times with different packet sizes. The throughput obtained thus is compared with the throughput obtained by doing simple IP level forwarding. The results show that the proposed flow establishment protocol performs slightly better or equivalent to IP level forwarding. The results have been plotted in Figure 5.6. The plot has been obtained by taking the average of the throughputs obtained in different trials.

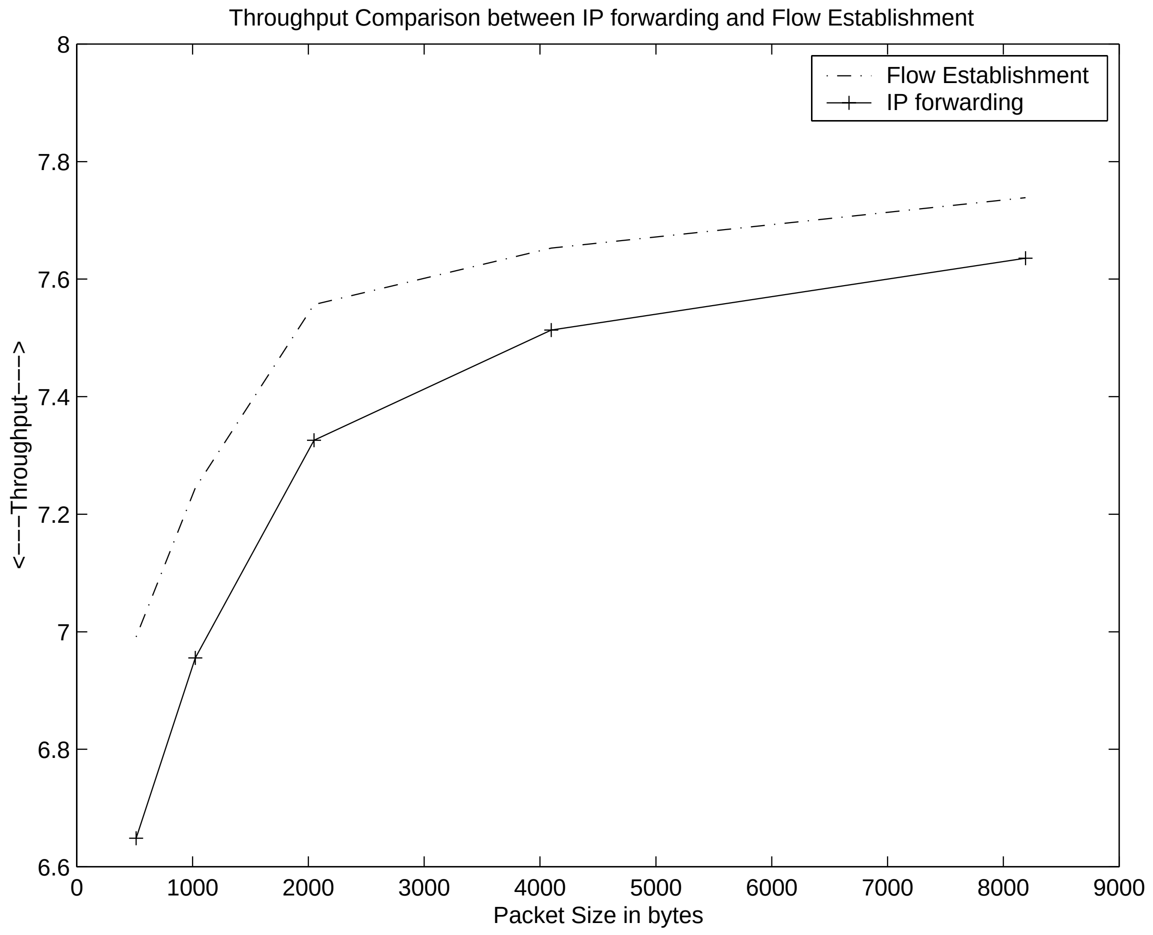

Figure 5.6: Comparison between IP forwarding and Flow Establishment

5.3.3 Flow Re-establishment time

This section discusses the performance of the flow management module in terms of the time taken to re-establish an end-to-end flow after a link failure. During flow re-establishment, the software ATM switch re-assembles the cells destined for an interface that has failed to create an IP datagram. As described in Chapter 3, the flow management module sets the REQ/RES and the MAX/MIN/BEST bits in the RDRN QoS options field appropriately, to indicate to the source of the flow that the connection has failed. The destination, on receiving the IP datagram with the modified options field, sends a QoS report to the source. On receiving this QoS report message, the source then starts the flow re-establishment process, and sends the datagrams on the default VC during this period. After the re-establishment of the flow, the source resumes transmission of the datagrams on the specific VC. The time interval between the detection of link failure and re-establishment of the flow is referred to as the flow re-establishment time. The logical configuration used for this experiment is shown in Figure 5.7. The links from testbed C to testbed D, testbed F to testbed G and testbed J to testbed K were broken and the flow re-establishment was measured. The results are tabulated in Table 5.1. As can be seen from the table, the flow re-establishment time depends on distance of the failing link from the destination of the flow. The flow re-establishment time is more than the flow establishment time, because of the additional overhead involved in the process of flow re-establishment.

Table 5.1: Flow Re-establishment time

Distance of failing link from destination	**Flow Re-establishment Time (in msec)**
7	36.38
5	33.83
2	29.89

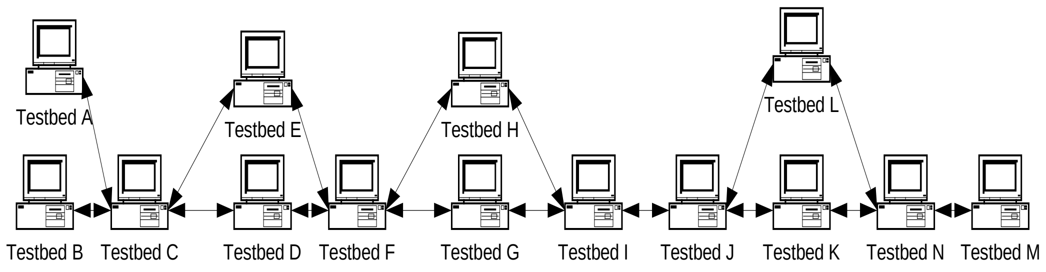

Figure 5.7: Logical Configuration to show Flow Re-establishment

5.4 Scalability

This section discusses the experiments conducted to test the scalability of the proposed flow establishment protocol. The logical configuration of the testbed is shown in Figure 5.8. Flows are set up from testbed A to testbed Z via the 10 testbeds and the flow establishment time is measured. The end-to-end throughput is also determined. The experiments are performed multiple times. The mean and the deviation from the mean of the flow establishment time are plotted in Figure 5.9. This plot also shows the flow establishment times for different number of MAPs between the source and the destination. The results show that the flow establishment times increase with an increase in the number of MAPs between the source and the destination. However, the increase is reasonable, indicating one facet of the scalability feature. The table also contains the average throughput obtained for the case when there are only 4 MAPs between the source and the destination, for different packet sizes. The difference in the throughput obtained has been plotted in Figure 5.10. The plot shows a marginal decrease in the throughput with an increase in the number of MAPs between the source and the destination. This indicates the other facet of the scalability feature.

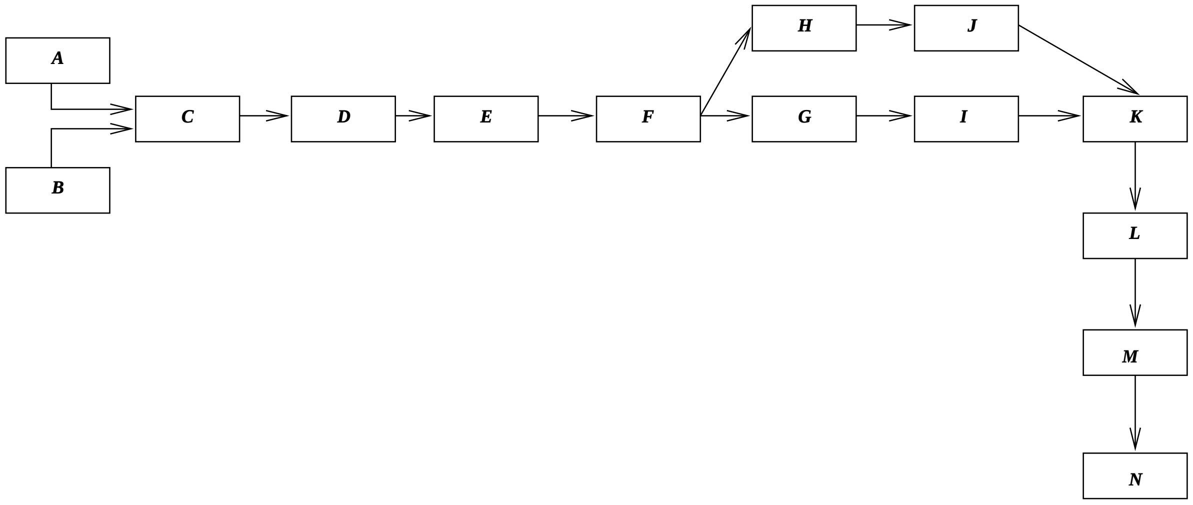

Figure 5.8: Logical Test Configuration to test scalability

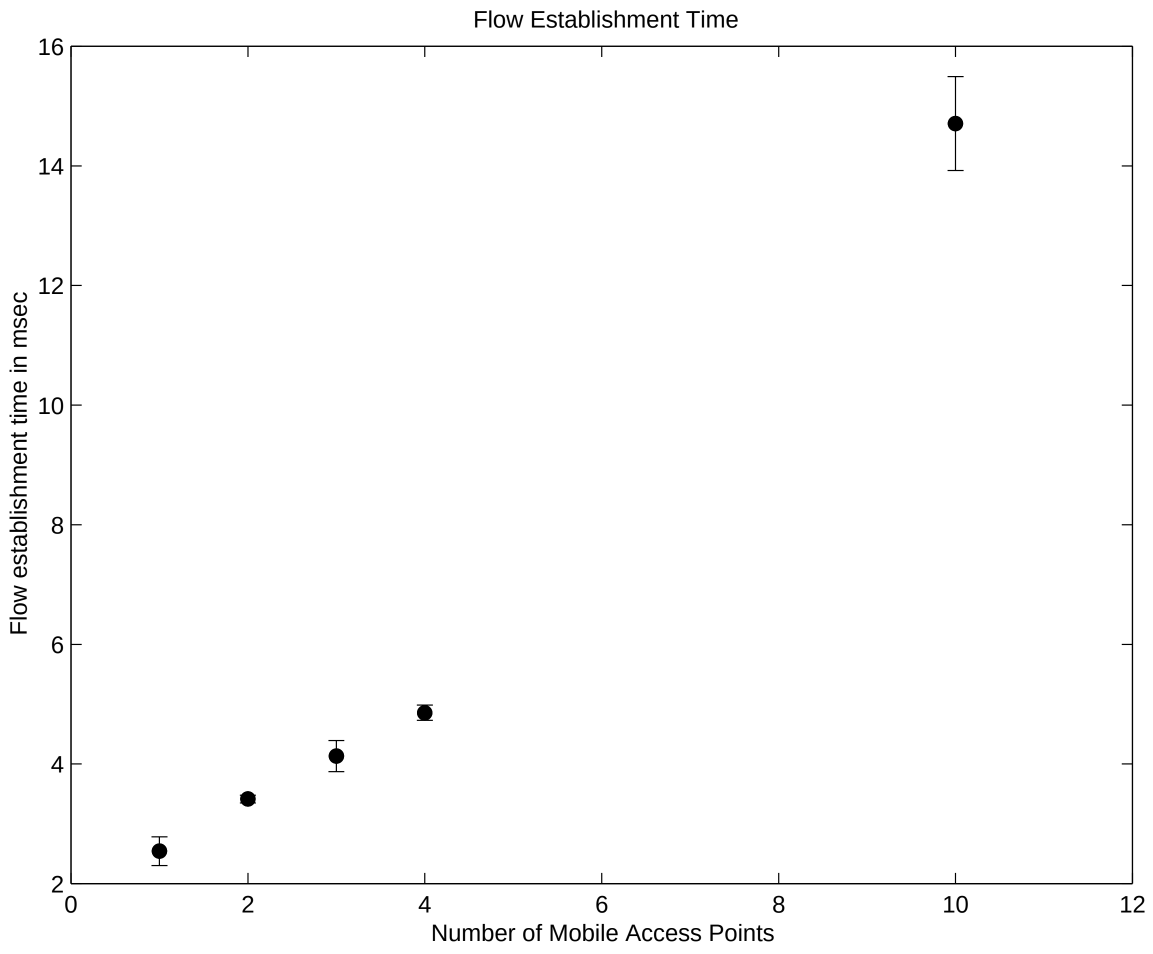

Figure 5.9: Scalability – Flow Establishment Time

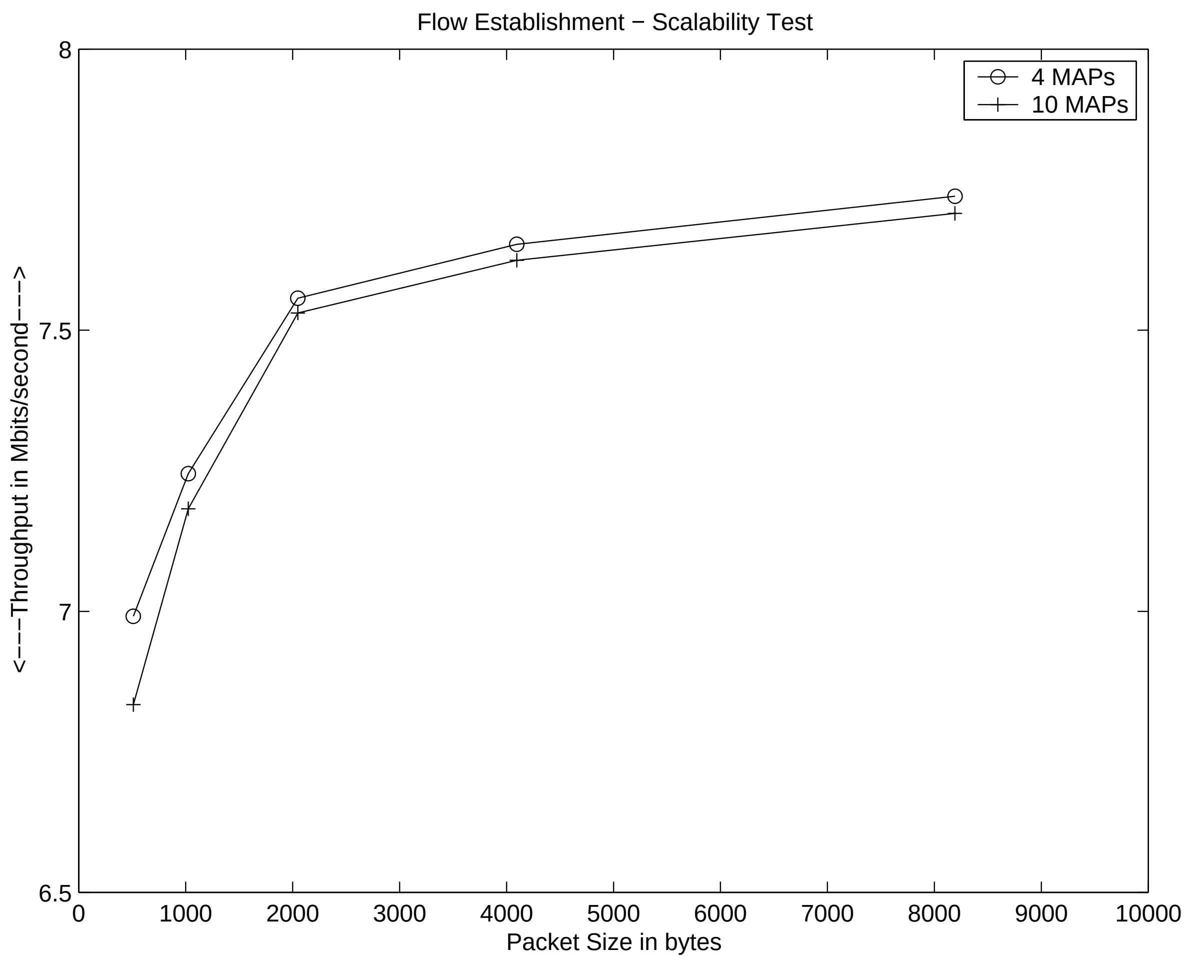

Figure 5.10: Scalability – Throughput

5.5 QoS Layer

This section discusses the experiments that were conducted to test the validity of the QoS layer that has been introduced in the RDRN protocol stack. Various features of the QoS layer have been tested and the results discussed. The tests that conducted were intended to test the validity of the QoS layer in the source nodes and in the intermediate nodes (namely the switches). The following features at the QoS layer were tested during the experimentation:

1. The performance of the WRR scheduler, that is, the ability to provide different services to different flows based on the QoS requirements of the flow.
2. The ability to provide a Constant Bit Rate (CBR) service under conditions of network congestion, that is, against increasing load.
3. The ability of the scheduler to allocate the available bandwidth in a weighted manner.

5.5.1 Performance of WRR Scheduler

This section discusses the range of tests that were conducted to test the validity and performance of the WRR scheduler. As discussed in Chapter 4, the scheduler would provide the entire bandwidth available to a flow, irrespective of the priority assigned to the flow, in the absence of network congestion. This feature of the scheduler is tested by setting up single flows, each with a different priority, and measuring the throughput obtained. When the network is congested, the scheduler would assign the bandwidth in the ratio of the weights assigned to the classes. The scheduler is tested for this feature by setting up multiple flows with different QoS requirements from the source (testbed A) to the destination (testbed D). The throughput obtained for the flows belonging to the various classes are measured.

Result

This section discusses the results that were obtained during the experiments that were conducted to test the functioning of the WRR scheduler. The logical set up for the experiments are shown in Figure 5.11. The peak cell rate is set to 10 Mbps on all outgoing links. In the first experiment, single flows were set up between the source and the destination, each with different QoS requirements. The tests were performed multiple

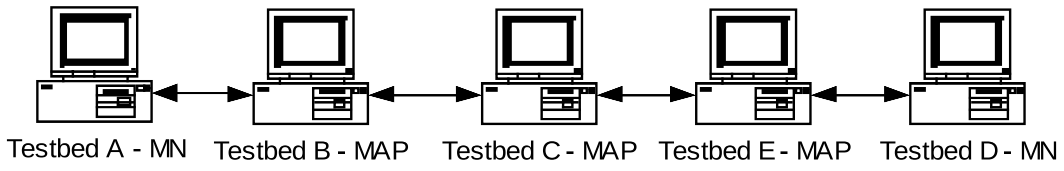

Figure 5.11: Logical Configuration to test WRR scheduler

times. The mean and the deviation from the mean have been plotted as a circle and a star in Figure 5.12. The deviation from the mean is very minimal for all the three types of flows. The results show that in the absence of network congestion, the flows are allocated the entire available bandwidth, irrespective of the priority assigned to them.

In the second experiment, three flows are set up simultaneously between the source and the destination, each with different QoS requirements. The ratio of the weights assigned for the classes is 5:3:2. The mean and the deviation from the mean have been plotted in Figure 5.13. The dots represent the throughput, while the vertical line represents the deviation from the mean. The results show that the deviation experienced by the high priority traffic is much lesser than that experienced by the low priority and medium priority flows.

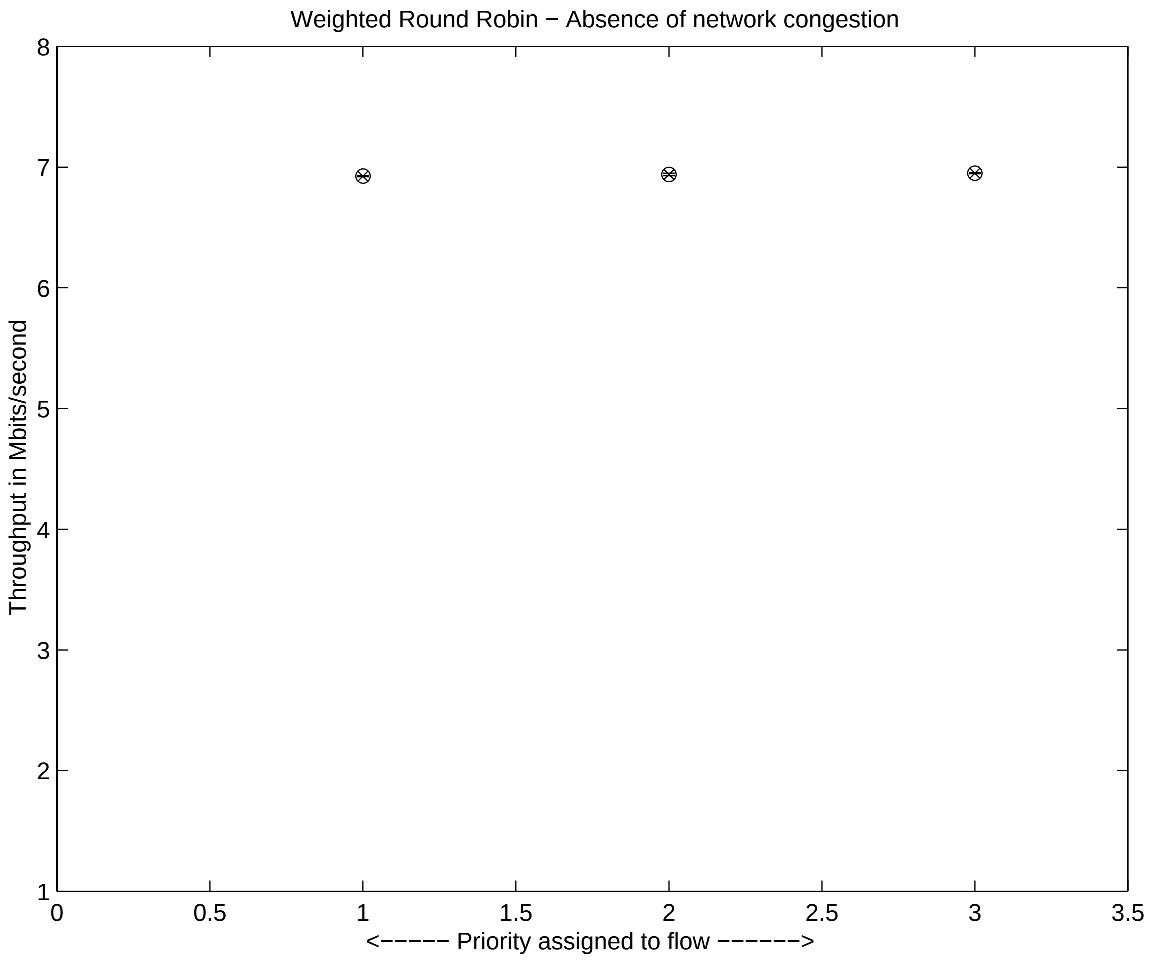

Figure 5.12: WRR Scheduler Performance in the absence of network congestion

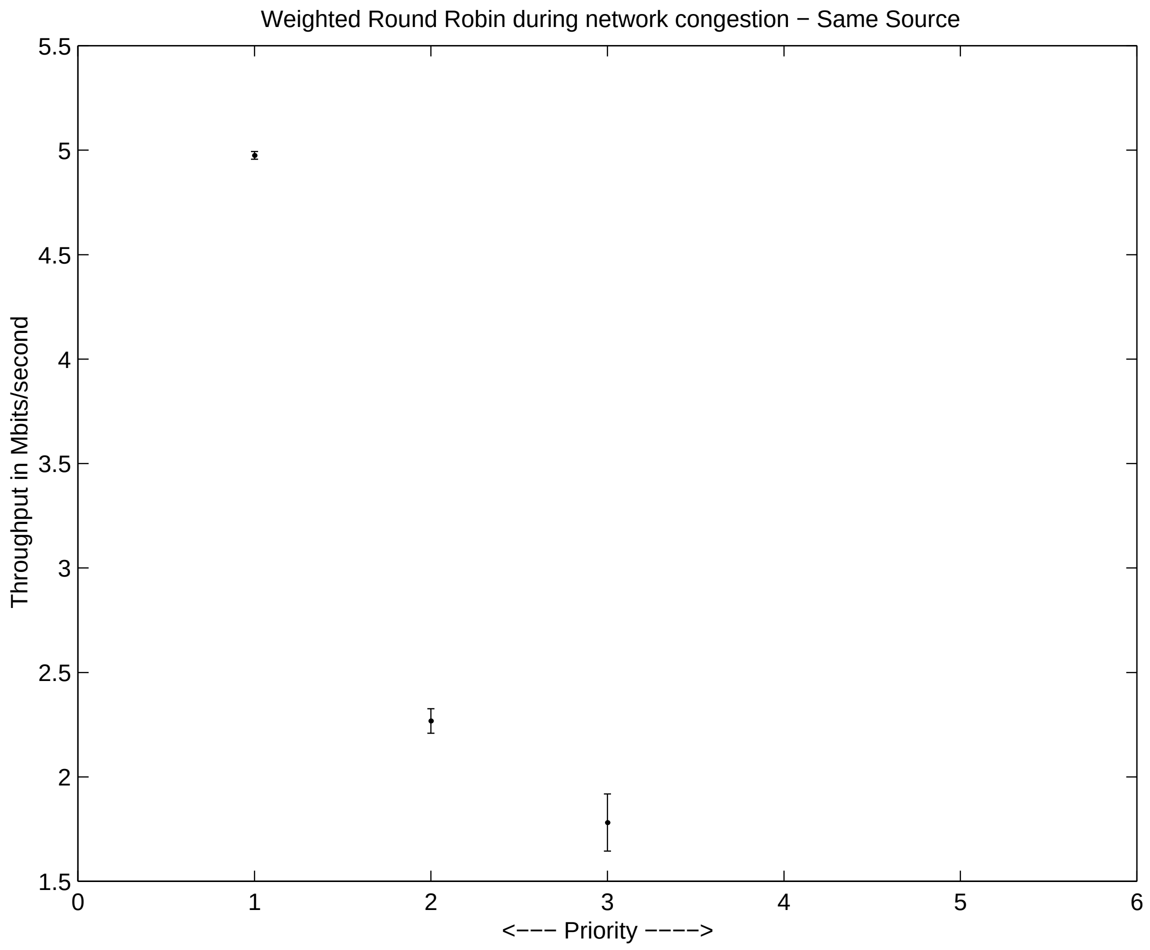

Figure 5.13: WRR Scheduler Performance during network congestion – Same Source

Having tested the performance of the WRR scheduler with multiple flows from the same source to the same destination, the next experiment would be to have multiple flows from different sources to the same destination. As already shown in Figure 5.12, in the absence of other flows, the source would allocate the entire bandwidth available to a flow irrespective of the priority assigned to it. Thus, the intermediate switches would have to ensure the provisioning of required characteristics to the various flows. The logical set up used for this test is shown in Figure 5.14.

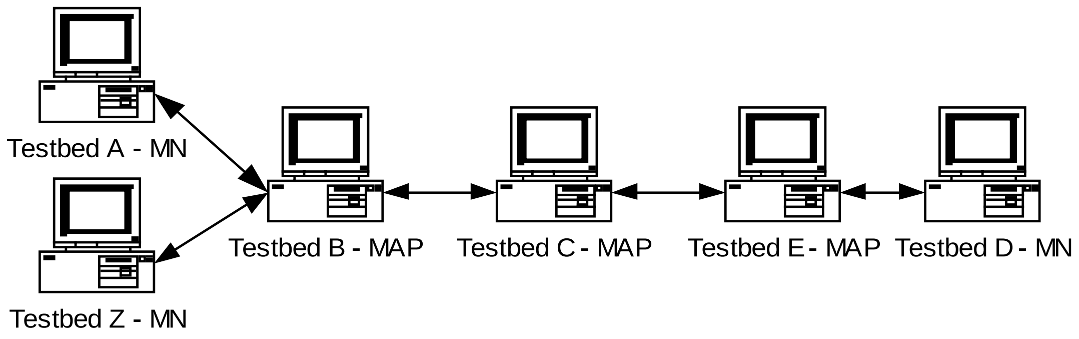

Figure 5.14: Logical Configuration to test flows from multiple sources

Flows were set up from testbed A and testbed Z to testbed D simultaneously. The plot showing the mean and the deviation from the mean is shown in Figure 5.15. The results show little deviation in the mean, and at the same time, the results obtained are different from those observed for flows from the same source, because no queuing is done on the input interface in testbed B.

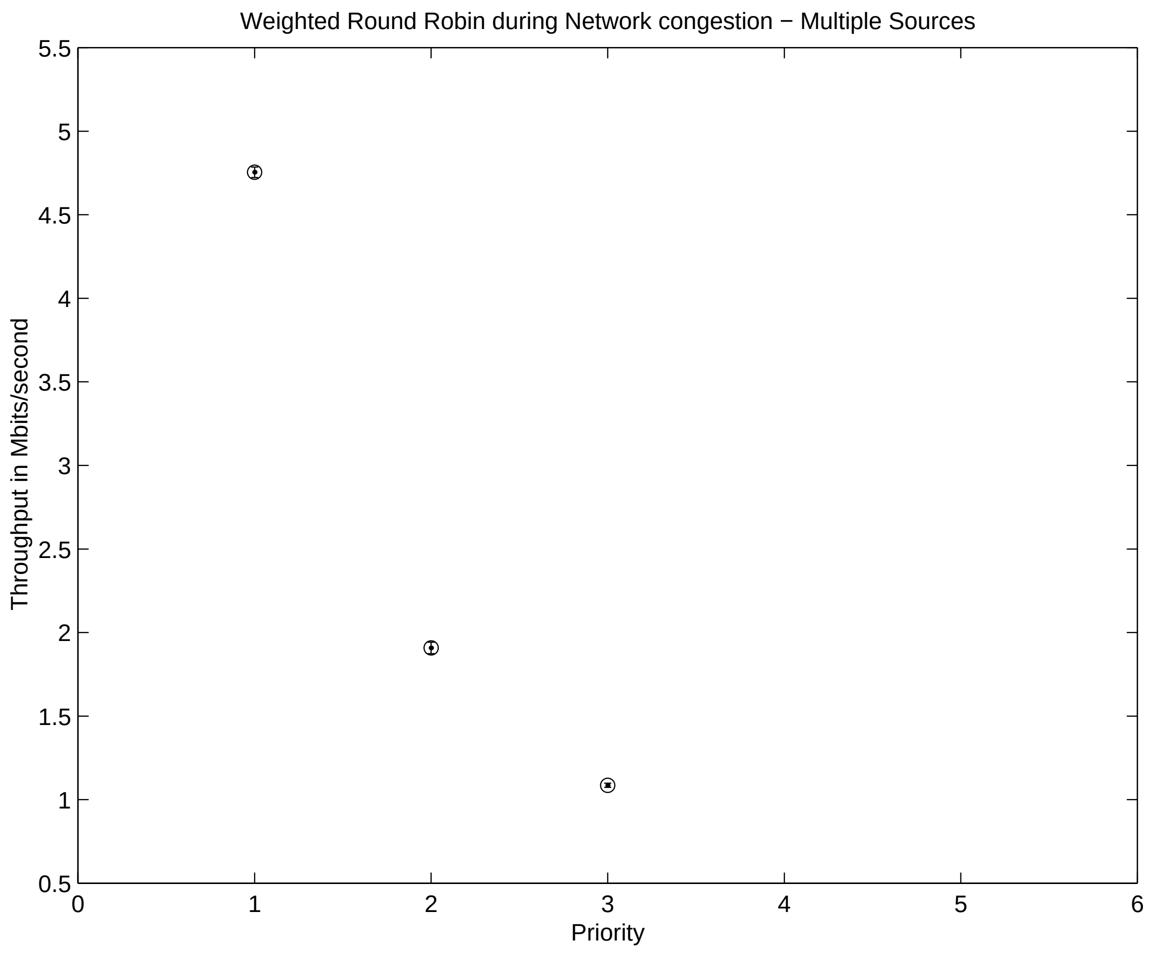

Figure 5.15: WRR Scheduler Performance during network congestion – Multiple Sources

5.5.2 Performance under increasing load conditions

This section discusses the performance of the WRR scheduler in terms of the throughput obtained by a high priority flow against increasing medium priority. The test setup used for this purpose is similar to the one shown in Figure 5.11. The maximum possible rate at which the cells can be sent on the output interface is set to 10 Mbps. One high priority flow and multiple medium priority flows are set up simultaneously from the source testbed A to the destination testbed D. The throughput obtained by the different flows are measured. The throughputs obtained by the medium priority flows are averaged. The results obtained have been plotted in Figure 5.16. The plot shows that as the load on the network increases, i.e., as the number of medium priority flows increase, the high priority flow remains unaffected.

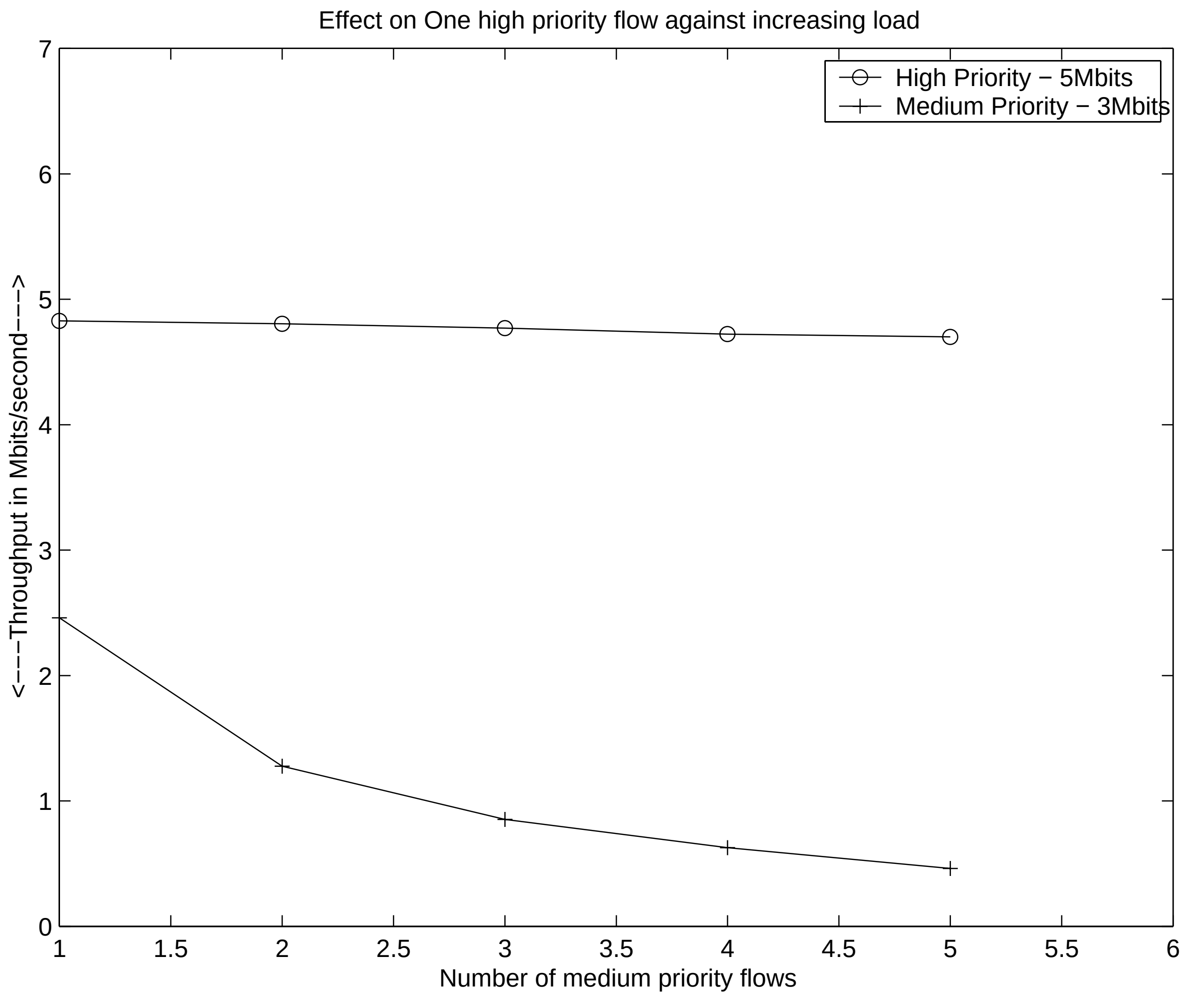

Figure 5.16: Performance of high priority flow under increasing medium priority load

5.5.3 Allocation of Available Bandwidth in a Weighted manner

The next experiment is performed to test if the scheduler allocates the available bandwidth in a weighted manner. The logical test setup is similar to the one shown in Figure 5.11, where the output links are capable of sending at a maximum rate of 10 Mbps. One medium priority flow and one low priority flow are set up from the source testbed A to the destination testbed D. Since there are no high priority flows, the available bandwidth is divided among the medium and low priority flows in the ratio of the weight specified. The results are plotted in Figure 5.17. The circle represents the throughput received by the individual flows, in the presence of all three types of flows. The star represents the throughput received by the medium and low priority flows in the absence if a high priority flow. The results show that the bandwidth available due to the absence of a high priority flow is shared in a weighted manner by the medium and low priority flows.

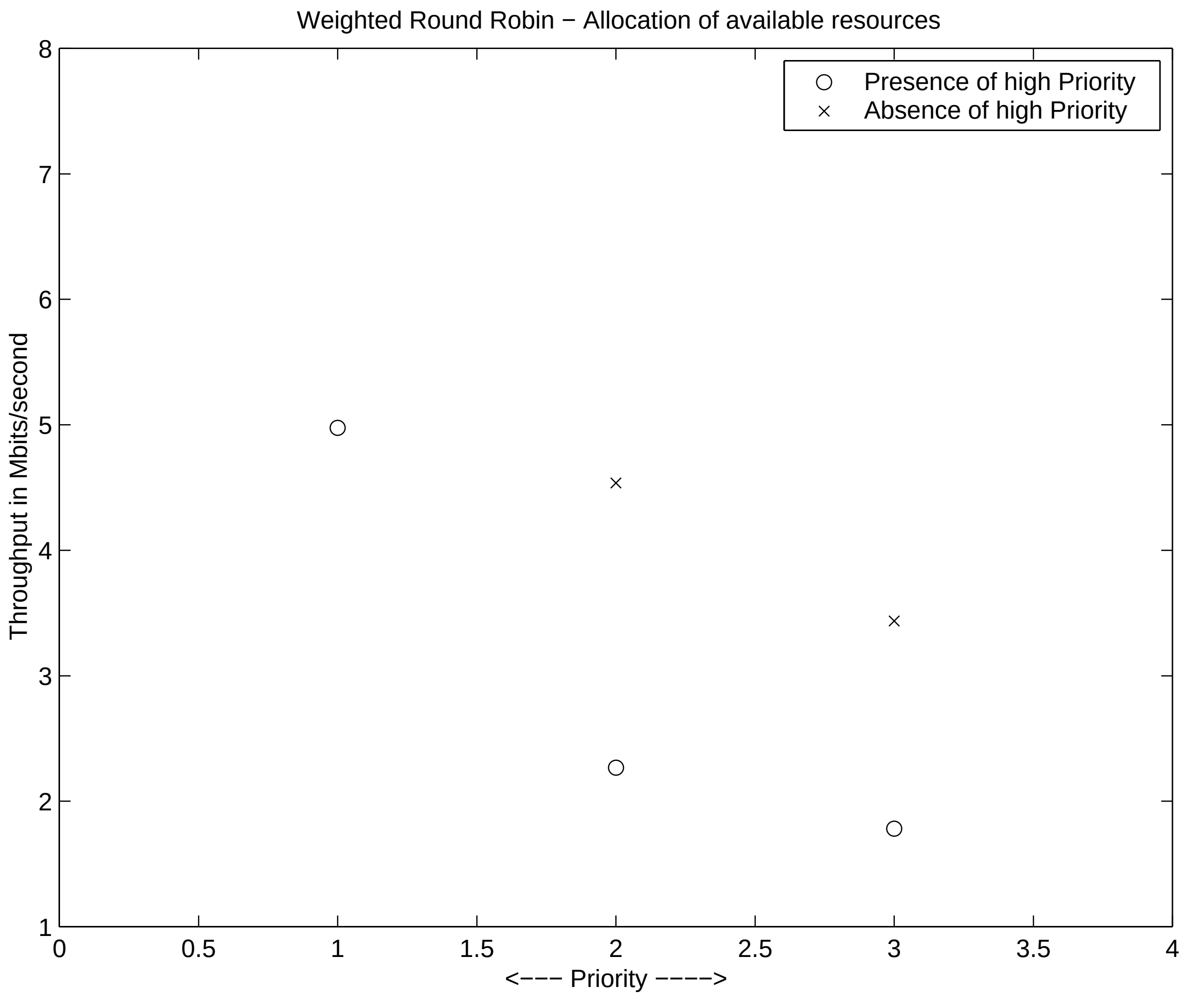

Figure 5.17: Weighted Round Robin – Allocation of available resources in a weighted manner

5.6 MOBILE QoS

This section discusses the experiments that were conducted to test the validity of the QoS layer in the presence of highly dynamic networking conditions. The emulation manager, developed at the University of Kansas, was used to emulate moving nodes. The scenario that was used to test this feature is shown in Figure 5.18. This scenario consists of four MAPs and 3 MNs.

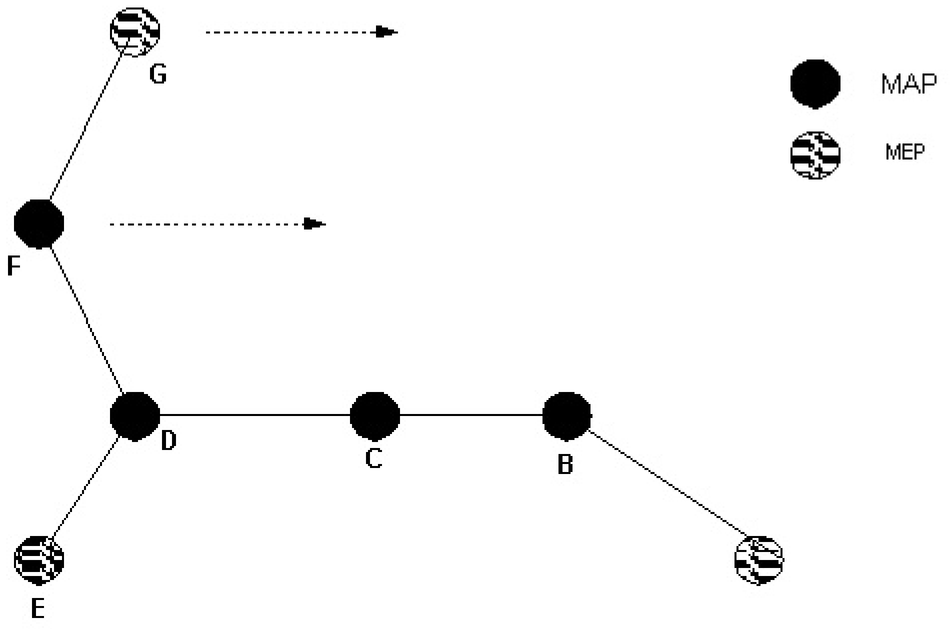

Figure 5.18: Scenario to test Mobile QoS

As shown in the figure, MAP F and MN G, keep moving eastwards at the same speed. MAP F serves as the only point of attachment for MN G. As the nodes keep moving, a flow is established from MN E to MN G. Initially the path used would be via MAPs D and F. After sometime, the link between MAPs F and D breaks, and the flow is re-established from E to F via MAPs D, C and F. Again, after sometime, the link between MAPs C and F breaks, and the end-to-end flow is re-established via MAPs D, C, B and F. The end-to-end throughput is measured and the results are tabulated in Table 5.2. The results show that the rate at which the packets are sent at the source is almost equal to the rate at which the packets are received at the destination, inspite of the dynamic condition in the network. The test was then repeated for two flows from the source to the destination. One was a high priority flow and the other was a low priority flow (in the ratio of 5:2). The results are tabulated in Table 5.3. The results show that even as the high priority and the low priority flows send data at the same rate, because of the ratio of the weights, the throughput achieved by the high priority flow is higher than that achieved by a low priority flow, thereby proving that the QoS characteristics are maintained even under highly dynamic conditions.

Table 5.2: Throughput achieved by an end-to-end flow in a highly dynamic environment

Sending Rate (in KB/second)	**Receiving Rate** (in KB/second)
466.17	466.15
459.61	459.59
464.19	464.17
462.25	462.02
467.27	478.39

Table 5.3: Throughput achieved by two flows simultaneously in a highly dynamic environment

Flow 1		**Flow 2**	
Sending Rate (in KB/second)	**Receiving Rate** (in KB/second)	**Sending Rate** (in KB/second)	**Receiving Rate** (in KB/second)
483.34	472.64	466.16	393.60
479.83	467.12	459.59	396.80
483.41	470.74	464.79	392.80
481.13	474.33	457.84	394.53

5.7 Dependence on Routing Protocol

This section discusses the experiments that were conducted to show the dependence of the flow establishment protocol on the routing protocol used. The scenario shown in Figure 5.18 is used in this case also. There were two phases in this experiment. During the first phase, the Wireless Multi-hop Routing Protocol developed at the University of Kansas was used. The instantaneous throughput as seen by the receiver is determined for flow establishment and IP forwarding. The routing protocol takes about 10 seconds to install a new route in the kernel. The plots of the instantaneous throughputs are shown in Figure 5.19 and Figure 5.20. The breaks in the flow result in cells being lost. During the second phase of the experiment, no routing protocol was run, the routes were hard-coded and were made available as soon as a link failed. The instantaneous throughputs were measured for flow establishment and IP forwarding, and the results are plotted in Figures 5.21 and 5.22. Figure 5.22 shows two dips, which indicate the times when the link breaks. The switch re-assembles AAL0 cells to create an IP datagram, which is time consuming. These plots show the dependence of the flow establishment protocol on the performance of the routing protocol.

This concludes the discussion on the experiments performed to prove the validity and measure the performance of the proposed RDRN QoS architecture, which encompasses the flow management module and the QoS layer in the protocol stack. An attempt has been made to exercise most of the features of the proposed algorithms discussed in Chapter 3 and 4. The results obtained show that the flow establishment protocol performs better than traditional IP forwarding. Also, the WRR scheduler at the QoS layer has been tested extensively for various features and its validity has been proved. The next chapter discusses the conclusions and the future work that needs to be done in this area.

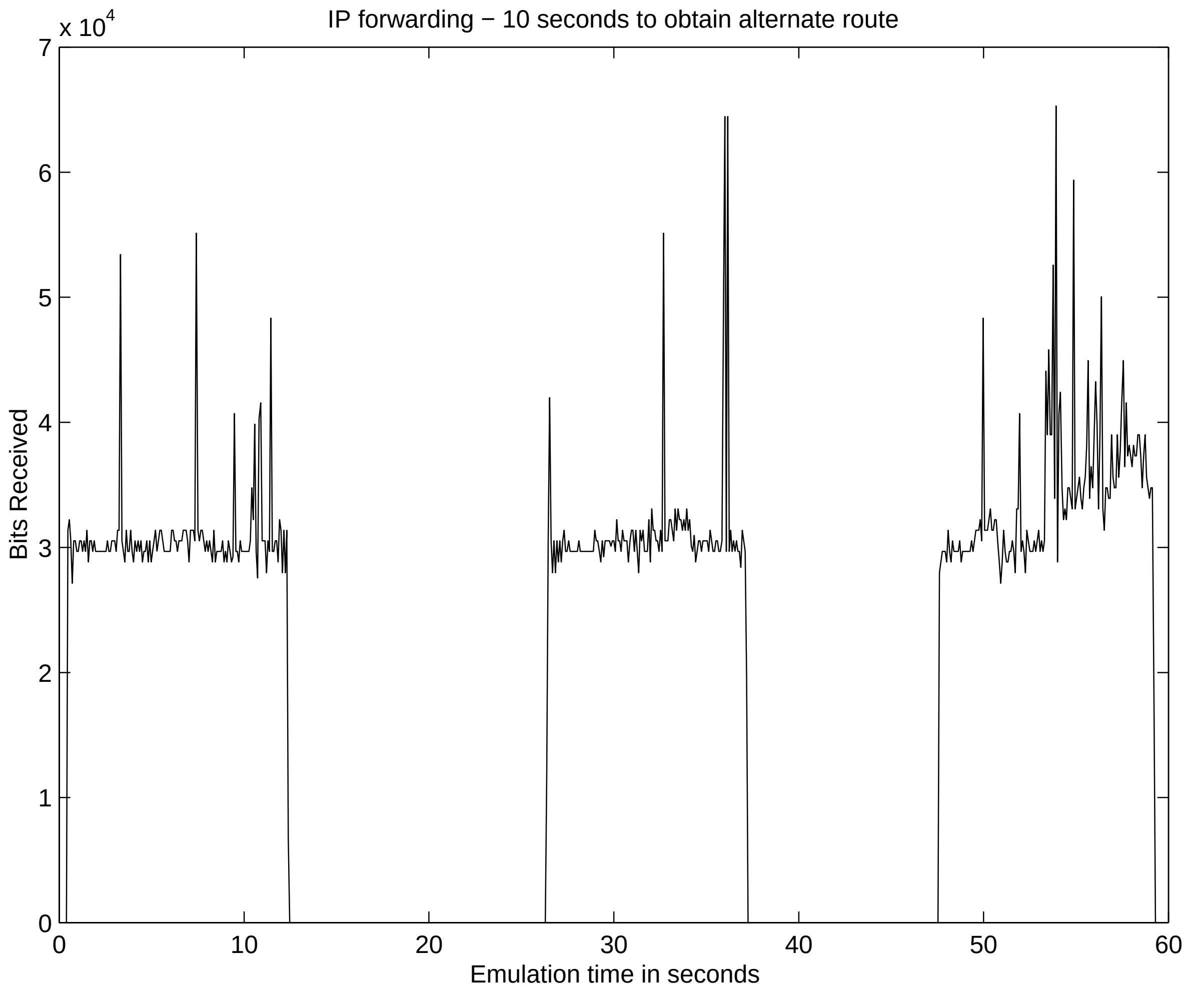

Figure 5.19: IP forwarding – 10 seconds to obtain alternate route

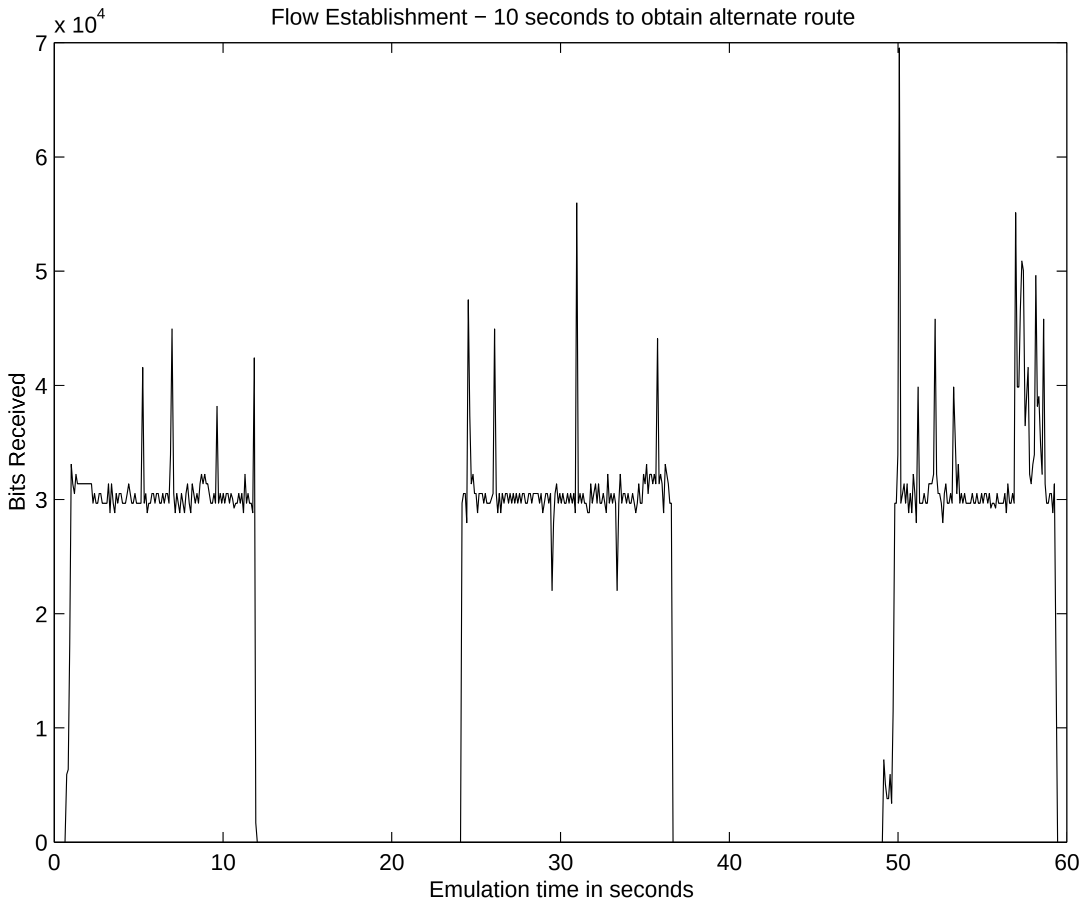

Figure 5.20: Flow Establishment – 10 seconds to obtain alternate route

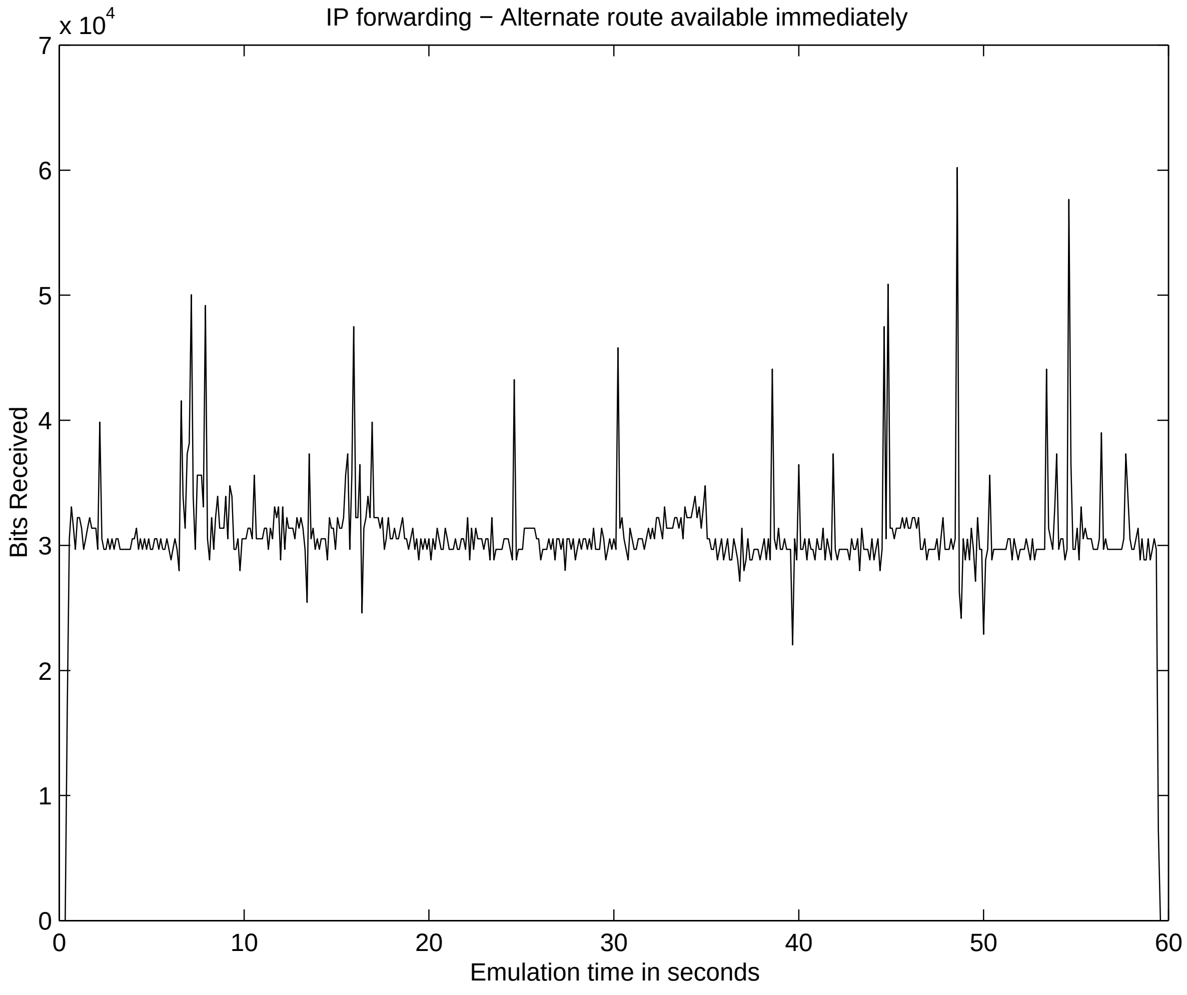

Figure 5.21 – IP Forwarding – Alternate Route immediately available

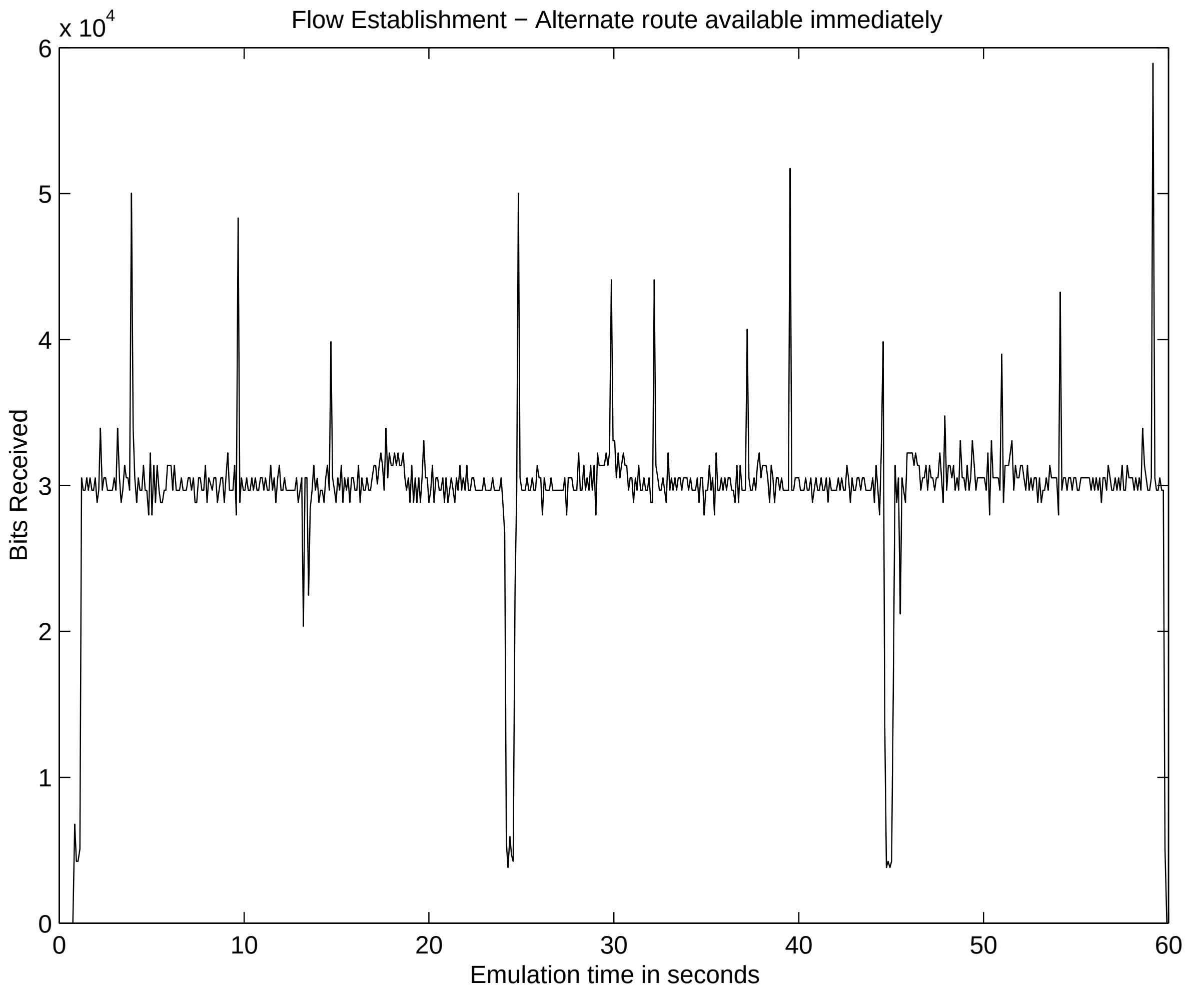

Figure 5.22 – Flow Establishment – Alternate Route Readily Available

Chapter VI

Conclusions and Future Work

This chapter discusses the conclusions and the future work that needs to be done in this area. A Quality of Service architecture for a highly dynamic RDRN environment has been proposed, implemented and evaluated. A flow specification has also been proposed for this environment. A QoS layer, which schedules the transmission of cells, has been introduced in the RDRN protocol stack, and its features have been validated and evaluated. The flow management module, coupled with the QoS layer in the protocol stack, provide the basic framework for the implementation of end-to-end QoS in such an environment.

As part of the flow management module, a new flow establishment scheme has been proposed, implemented and its performance has been evaluated. The flow establishment protocol uses IN BAND SIGNALING for the establishment of flows results in efficient utilization of the wireless resources. Because of the switching that is done at the link level, the flow establishment protocol performs slightly better than traditional IP level forwarding. The time to establish and re-establish the end-to-end flow is reasonable. The scalability of the protocol was also tested by setting up a configuration with a large number of nodes and measuring the flow establishment time and the end-to-end throughput. The flow establishment time was found to be acceptable and the end-to-end throughput showed only a marginal drop with an increase in the number of nodes, thereby establishing the scalability of the protocol. The approach proposed in this thesis moves mobility related issues to the IP layer, thereby using ATM just as a link level protocol. This approach is justified because of the highly robust nature of IP. Given the highly dynamic environment that persists in RDRN because of node mobility and instability of the wireless links, robustness is a much-needed feature. ATM, being a connection-oriented technology, would result in a lot of signaling overhead if it is used to handle the dynamic conditions that arise in the network. The robustness of IP makes mobility management issues easier to handle.
There are additional benefits in mobility management in using the in-band signaling. When a MN moves from one MAP to another, a handoff of the various flows from the MN needs

to be made. With in-band signaling and the flow establishment scheme proposed, the nodes can continue to transmit the packets without interruption. The new flow will be established and the end-to-end QoS will be set up. This claim, though, has been not been validated with results.

The QoS layer uses a Weighted Round Robin Scheduler to schedule the transmission of cells. This ensures that the high priority flows obtain the requested resources, irrespective of the network conditions, i.e., in the presence or absence of network congestion. In the absence of network congestion, the scheduler assigns the entire resources to a flow, irrespective of the priority assigned to it. The implementation of the scheduler has been validated with results.

One of the issues that were discussed in the design, but that were not implemented, includes the scaling up feature. If the resources requested by a flow are not available, the flow is either allocated the minimum requested resources or is considered best effort. After some time T, determined on the basis of the priority assigned to a flow, the flow management module attempts to scale up the resources reserved for the flow. The current implementation provides the necessary framework to add this feature in an efficient manner.

Yet another feature that has not been implemented, but has been proposed, is the use of the wireless and mobile QoS parameters, which is maintained as part of the flow specific information that is maintained in the intermediate nodes. The current implementation maintains this information, but has not made use of them because of the absence of a supporting handoff protocol. Thus, the design of a handoff protocol for RDRN should ensure the usage of the mobile QoS parameters that are maintained in the intermediate nodes.

Another issue that would be interesting to investigate would be difference between providing QoS at the IP layer and at the ATM layer. The current implementation provides the QoS features at the ATM cell level. Because of the limitations discussed in Chapter 4, IP

QoS could not used in the RDRN protocol stack. This would be a very interesting comparison to observe.

APPENDIX A

This appendix shows a simple user level code that can be used to specify the RDRN flow specification and send data to the destination using the flow establishment protocol. This is achieved using a setsockopt call.

```
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <linux/ip.h>
#include <linux/rdrn_qos.h> /* RDRN Header File */

#define FLOW_PORT 3835
#define FLOW_DEST "10.0.0.14"
#define MSGBUFSIZE 256

main (int argc, char *argv[])
{
   struct sockaddr_in serv_addr,cli_addr;
   int i, fd, nbytes, addrlen;
   int source_port, flow_port;
   char buff[MSGBUFSIZE],msgbuf[MSGBUFSIZE];
   struct rdrn_options options;
```

```
flow_port = atoi(argv[1]);
source_port = atoi(argv[2]);

/* create what looks like an ordinary UDP socket */
if ((fd=socket(AF_INET,SOCK_DGRAM,0)) < 0) {
        perror("socket");
        exit(1);
}

/* Set up destination address */
memset(&serv_addr,0,sizeof(serv_addr));
serv_addr.sin_family=AF_INET;
serv_addr.sin_addr.s_addr=inet_addr(FLOW_DEST);
serv_addr.sin_port=htons(flow_port);

 /* Set up my address */
memset(&cli_addr,0,sizeof(cli_addr));
cli_addr.sin_family = AF_INET;
cli_addr.sin_addr.s_addr = inet_addr("10.0.0.1");
cli_addr.sin_port  = htons(source_port);

/* Bind to receive address */
if (bind(fd,(struct sockaddr *) &cli_addr,sizeof(cli_addr)) < 0) {
        perror("bind");
        exit(1);
 }
options.opcode = IPOPT_RDRN; /* Setting the Option Code */
options.state  = NEW_FLOW; /* Specify that this is a new flow */
options.req_qos.app_type = RT;   /* Specify the application type */
```

```
strcpy(options.req_qos.throughput, “1Mbps”); /* Set the requires throughput */
options.dest_ip = inet_addr(FLOW_DEST); /* To set the filter specification */
  options.dest_port =  htons(flow_port); /* Filter specification again */
  options.optlength = sizeof (struct rdrn_options); /* Size of the Options field */

  /* Use the setsockopt call to specify the socket options, i.e., is to use flow   establishment
*/
  if  (setsockopt(fd,SOL_IP,IP_OPTIONS, &options,sizeof(struct rdrn_options)) <
 0) {
    perror("setsockopt");
    exit(1);
  }

  for (i=0; i < 30000; i++)  {
    sprintf(buff,"RDRN Demo %d\n",i);
    len = strlen(buff);
    /* Sending data to the destination */
    if (sendto(fd,buff,len,0,(struct sockaddr *) &serv_addr, sizeof(serv_addr)) < 0) {
           perror("sendto");
           exit(1);
    }
    printf("Sent: %s\n",buff);
    fflush(stdout);
    }
 close(fd);
}
```

References

[1] Goujun Lu, "*Communication and Computing for Distributed System*", 3rd Edition, 1991.

[2] Information Sciences Institute, University of Southern California, "*Internet Protocol – Protocol Specification*", RFC 791.

[3] Mahmoud Naghshineh and Marc Willebeek-Lemair, "*End-to-End QoS Provisioning in Multimedia Wireless/Mobile Networks Using an Adaptive Framework*", IEEE Communication Magazine, November 1997.

[4] Ricardo Sanchez, Joseph B.Evans, Gary J.Minden, Victor S.Frost and K. Sam Shanmugam, "*RDRN – A Prototype for a Rapidly Deployable Radio Network*", Mobile Computing and Communication Review of the ACM, Vol. 2, No.2, April 1998.

[5] Ricardo Sanchez, Joseph B.Evans, Gary J.Minden, Victor S.Frost and K. Sam Shanmugam, "*A Rapidly Deployable Radio Network – Implementation and Experience*", Proceedings of IEEE 1998 International Conference on Universal Personal Communications (ICUPC '98), Florence, Italy, October 1998.

[6] R. Braden, D.Clark, S.Shenker, "Integrated Services in the Internet Architecture: An Overview", RFC 1633

[7] Steven Blake, David Black, Mark Carlson, Elwyn Davies, Zheng Wang, Walter Weiss, *"An Architecture for Differentiated Services*", RFC 2475.

[8] Volg C., Wolf L., Herrtwich R., H. Wittig, *"HieRAT – Quality of Service Management for Distributed Multimedia Systems"*, Multimedia Systems Journal, 1996.

[9] Hehmann D.B., Herrtwich, R.G., Schulz W., Schuett T. And R. Steinmetz, *"Implementing HeiTS: Architecture and Implementation Strategy of the Heidelberg High Speed Transport System"*, Second International Workshop on Network and Operating System Support for Digital Audio and Video, IBM ENC, Heidelberg, Germany, 1991.

[10] Delgrossi L., Halstrinck C., Henhann D.B., Herrtwich R.G., Krone J., Sandvoss C and C. Vogt, "*Media Scaling for Audiovisual Communication with the Hiedelberg Transport System"*, Proceedings of ACM Multimedia 1993, Anaheim, August 1993.

[11] Lazar A.A, Bhonsle S., Lim K.S., *"A Binding Architecture for Multimedia Networks"*, Proceedings of COST-237 Conference on Multimedia Transport and Teleservices, Vienna, Austria, 1994.

[12] Lazar A.A, "*A Real-time Control, Management, and Information Transport Architecture for Broadband Networks*", Proceedings of International Zurich Seminar on Digital Communications.

[13] Hyman J., Lazar A and G. Pacifici, *"Joint Scheduling and Admission Control for ATS-based Switching Node"*, Proceedings of ACM SIGCOMM 1992, Baltimore, Maryland, U.S.A, August 1992.

[14] Lazar A., *"Challenges in Multimedia Networking"*, Proceedings of International Hi-Tech Forum, Osaka, Japan, February 1994.

[15] Shenker S. and J. Wroclawski, *"Network Element Specification Template"*, RFC 2216.

[16] J. Wroclawski, *"Specification of Controlled-Load Network Element Service"*, RFC 2211.

[17] Shenker S, C Partridge and R Guerin, *"Specification of Guaranteed Quality of Service"*, RFC 2212.

[18] R. Braden, Ed., L. Zhang, S. Berson, S. Herzog, S. Jamin, "*Resource Reservation Protocol (RSVP) – Version 1 Functional Specification"*, RFC 2205.

[19] K. Nichols, S. Blake, F. Baker, D. Black., *"Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers"*, RFC 2474.

[20] J. Heinanen, F. Baker, W. Weiss, J. Wroclawski, *"Assured Forwarding PHB Group"*, RFC 2597.

[21] V. Jacobson, K. Nichols, K. Poduri, *"An Expedited Forwarding PHB"*, RFC 2598.

[22] L. Anderson, P. Doolan, N. Feldman, A. Fredette and R. Thomas, *"Label Distribution Protocol"*, Internet Draft, 1998.

[23] P. Vaannen and R. Ravikanth, *"Framework for Traffic Management in MPLS Networks"*, Internet Draft, 1998.

[24] D. Awduche, J.Malcolm, J. Agogbua, M. O'Dell and J. McMaus, *"Requirements for Traffic Engineering over MPLS"*, Internet Draft, 1998.

[25] Arun Ayyangari, Jeff Harrang, Sankar Ray, *"Call Establishment/Termination in Wireless PNNI"*, ATM Forum/96-1410.

[26] Lee, S-B and A.T. Campbell, *"INSIGNIA: In-Band Signaling Support for QoS in Mobile Adhoc Networks"*, Proceedings of 5th International Workshop on Mobile Communications, 1998, Berlin, Germany.

[27] DARPA, *“QoS Requirements for Mobile Tactical Application”*, DARPA Draft, 1998.

[28] C.L. Hedrick, *“ Routing Information Protocol”* – RFC 1058.

[29] J.Moy, *“OSPF Version 2”* – RFC 2328.

[30] D. Oran, *“OSI IS-IS Intra-domain Routing Protocol”* – RFC 1142.

[31] M. Laubach, J. Halpern, *“Classical IP and ARP over ATM.”* – RFC 2225.

[32] Sally Floyd, V. Jacobson, *“Link Sharing and Resource Management Models for Packet Network”*, IEEE/ACM Transactions on Networking, 1995.

[33] A. Tanenbaum, *“Computer Networks”*, Third Edition, 1997.

[34] Paul E. McKenney, *“Stochastic Fairness Queuing”*, IEEE INFOCOMM ’90 Proceedings, San Francisco, 1990.

[35] Sally Floyd and Van Jacobson, *“Random Early Detection Gateways for Congestion Avoidance”*, 1993, IEEE/ACM Transactions on Networking.