Dynamic Quality-of-Service for Mobile Ad Hoc Networks

M. Mirhakkak, N. Schult, D. Thomson
The MITRE Corporation
Network and Communications Engineering Department
Reston, Virginia

Abstract

In this paper we discuss the dynamic network characteristics of mobile ad hoc networks, and explore problems that arise when attempting to provide QoS support in this environment. A dynamic approach to Quality of Service (QoS) is presented as a possible solution to these problems. With this approach, resource reservations represent ranges, and applications adapt to an allocated level of QoS provided by the network at some point within the requested range. To explore this approach, we have implemented a new protocol called dynamic RSVP (dRSVP), which is an extension to RSVP. We describe this protocol and our implementation and testbed, and discuss its applicability to mobile ad hoc networks. We survey related work by other authors, and discuss the unique features of our work.

1. Introduction

This paper presents an approach for providing Quality-of-Service (QoS) in a dynamic network environment such as that presented in a mobile ad hoc network. The paper also describes an implementation of this approach, which we have created by extending the Resource Reservation Setup Protocol, RSVP [1], [2]. The paper also describes adaptive streaming video and audio applications that we have used demonstrate and evaluate the benefits of our approach.

Our approach, which we refer to as "Dynamic QoS", is a resource reservation based approach to QoS which fits within the Integrated Services Internet (intserv) architecture, with an expanded meaning of the term "reservation". With Dynamic QoS, a resource reservation request specifies a range of values, and the network makes a commitment to provide service at a specified point within this range. Applications request QoS by specifying the minimum level of service they are willing to accept and the maximum level of service they are able to utilize, and then adapt to the allocation within this range provided by the network, which may change with time.

Treating reservations as ranges, and providing a mechanism for the network to signal to adaptive applications the current allocation within the range, provides the flexibility needed for operation in a dynamic environment. As several authors have pointed out (e.g. [3]), the rapid deployment of wireless and mobile networking technology creates numerous examples of such dynamic networking environments. Section 2 of this paper discusses in more detail the dynamic characteristics that can be expected in mobile ad hoc networks, at the physical, link, and network layers. The difficulties of applying a resource reservation-based approach to QoS in such an environment are also discussed.

Having discussed the problem in Section 2, we go on in Section 3 to describe our approach to tackling the problem. We define the semantics of reservation ranges and discuss how this concept provides the flexibility needed for QoS support in a dynamic environment. We also discuss the

implications on applications, which must be capable of expressing reservation requests as a range of desired values and of adapting to changes in the allocation provided by the network.

Section 4 provides an overview of the dynamic RSVP (dRSVP) protocol, which we have implemented to demonstrate the dynamic QoS approach.

In Section 5 we review related work. The notion of treating resource reservations as ranges, and adaptively adjusting QoS within this range, was introduced by Lu and Bharghavan [4], [5]. Other projects, for example the Mobiware effort [6], have also explored the notion of QoS ranges, adaptivity, and other mechanisms for providing QoS in a mobile or wireless environment. More recently, the INSIGNIA protocol [7], combines the notion of QoS ranges with lightweight signaling carried in the data packet headers as an approach to providing QoS in a mobile ad hoc network. We survey and briefly examine these and other efforts, and we discuss the differences between those efforts and our Dynamic QoS approach. The unique features of our work include support for multicast flows, a new bandwidth allocation algorithm, and a new protocol design.

In Section 6 we go on to describe our testbed and the adaptive streaming video and audio applications that work with dRSVP. We also describe our test network, which includes software emulation of variable speed links.

Finally, in Section 7, we conclude by summarizing what we have accomplished and demonstrated, and by raising issues for further study.

2. Dynamics in Mobile Ad Hoc Networks

The most obvious source of dynamics in a mobile ad hoc network is changes in network topology. However, this is not the only source of dynamics. We must also consider the dynamic nature of the wireless links that are used in a mobile ad hoc network. Also, variable application demand creates another source of dynamics in mobile ad hoc networks, just as it does in conventional networks. In this section we examine these sources of dynamics and the problems they create when attempting to provide QoS support.

2.1. Variable Link Characteristics

While links between network nodes are often treated as if they have fixed characteristics (e.g., bandwidth, error rate), in fact most links have characteristics that change with time. This is especially true for wireless links, which are subject to variations in transmission quality due to factors such as interference and fading. (We should note that variable link characteristics also occur in wired links. For example, V.90 modems will not only negotiate the data rate at connect time, but may also renegotiate the data rate during the course of the connection [9].)

To understand how variable link characteristics at the physical layer will affect a QoS mechanism, we must examine how this variability will affect the link layer and the network layer. Assuming that changes in transmission quality are not handled by the physical layer (for example by increasing transmit power), the result observed by the link layer will be changes in bit error rate. The effect observed by the network layer depends on whether or not the link layer mechanisms in use recognize and respond to variations in error rate.

First, consider the case where the link layer does not detect or respond to the change in bit error rate. In this case, when the link degrades the network layer sees an increase in lost or corrupted packets, however the interface bandwidth perceived by the network layer will not change. This situation would significantly complicate the design of a network layer QoS solution. Some form of network layer error detection and correction would probably be required to provide QoS assurances, and it would be difficult for the network layer to

distinguish between packet loss due to congestion and loss due to link layer corruption. It would also be difficult for the network layer to determine the current available bandwidth, which is a key parameter in any resource reservation-based QoS mechanism. For these reasons, we believe that it is preferable to deal with variable bit error rate within the link layer.

Therefore, let us assume that the link layer reacts to the change in link error rate. Several different types of reaction are possible. For example, if the link layer protocol includes automatic repeat-request (ARQ), then, as transmission quality decreases, the number of retransmissions will increase, and the main affect on the network layer will be a decrease in the effective throughput of the link. A sophisticated link layer could also employ an adaptive error correction mechanism to increase or decrease the amount of error correction coding in response to changes in transmission quality. Products that do this are not widespread, but they do exist. For example, Scientific Research Corp. has produced a high-speed wireless modem that adaptively applies forward error correction (FEC), depending on link conditions [10]. Adaptive FEC has also been demonstrated in a mobile wireless environment in the Mobiware project, described in [6]. The effect on the network layer will be variation in effective delay or throughput, depending on the coding algorithm. The link layer could also react by changing its modulation technique. Advanced 802.11 products, for example Lucent Technologies' "WaveLAN Turbo" wireless network interface cards, are capable of operating with different modulation techniques, at speeds ranging from 1 to 11 Mbps, depending on observed transmission characteristics. Again, the main effect observed by the network layer will be a change in throughput.

From the above, we observe that, as the link layer reacts to variable bit error rate, the main affect observed by the network layer is a change in effective throughput, while packet loss due to corruption remains low. This is significant, because interface bandwidth is a key parameter in any resource reservation-based QoS approach. Current link throughput information must be made available to the network layer software whenever conditions change, preferably by explicit signaling from the link layer. In current implementations interface speeds are likely to be treated as a configuration parameter which is read when the system is booted or a daemon is started, and never checked again. Clearly, this is inadequate when link bandwidth is variable. In order to appropriately perform admission control, allocate resources, and other functions necessary for providing QoS, the network layer needs updated information from the link layer on the effective data rate of each interface, as well as possibly other parameters such as latency. As part of the DARPA funded Global Mobile (GloMo) effort, a general framework for internet devices operating over wireless links was developed. This framework, specified in [11], defines interfaces between different layers or components, and includes the ability of the network layer to obtain information on current conditions observed by the link layer, including current link speed.

Another possible architecture would be for a subnet bandwidth manager such as that described in [12] or [13] to assume responsibility for controlling link layer QoS and determining available bandwidth on an interface. The subnet bandwidth manager would interact with a network layer QoS mechanism, and provide information on available bandwidth. The subnet bandwidth manager also deals with link layer QoS and resource management in a shared media environment, which is not addressed further in this paper.

2.2. Node Movement

An obvious source of network dynamics in mobile ad hoc networks is node movement, which has several consequences. First, it exacerbates the problem of variable link characteristics discussed above, as nodes move in and out of areas of good signal strength, and so on. Another consequence of node

movement is that nodes may have to switch to different media as they move in and out of coverage. For example, an aircraft parked at the gate could be connected to a high speed wireless LAN link, then switch to VHF data radio connectivity after pulling back from the gate, and finally switch to satellite connectivity when out of radio range over the ocean. Or a laptop or PDA could be connected to a wireless LAN within a building, then switch over to PCS or cellular connectivity when moving outside. A "vertical handoff" approach has been described by Stemm and Katz [3], in which seamless connectivity to mobile nodes is maintained by handing off between small cells with high bandwidth and wide area cells with lower bandwidth. Changes in media mean that available bandwidth can vary, perhaps by several orders of magnitude.

Node movement also means that the network topology can change. In the simple case, this consists of the movement of end systems through a fixed network infrastructure. Mobile end systems are "handed off" between fixed access points. In the more general case of a mobile ad hoc network, intermediate systems (routers) also move, possibly causing relatively rapid routing changes. Clearly, this has a major impact on a resource reservation-based QoS approach, as resources that were available and reserved to support a reservation along one route may not be available on the new route. Various approaches (discussed in Section 5) have been proposed for using "pre-reservation" (in the simpler "handoff" case), or "standby routes" (in the more complex mobile ad-hoc network case) to allow the network to make a QoS commitment that can be honored even when the network topology changes. However, it seems intuitively clear that to fully ensure that a fixed level QoS can be maintained in a mobile ad hoc network would result in prohibitive underutilization of the network. This is because, to allow totally general mobility with a guarantee of no loss of QoS, it would be necessary to reserve resources on *every* link for *every* flow. Therefore, we believe that a "weaker" approach, such as the dynamic QoS approach presented in the next section, is more appropriate in a mobile ad hoc network. Our approach does not attempt to guarantee that a fixed level QoS will be preserved through topology changes. Rather, when topology changes occur, our approach allows the new level of available resources to be discovered, and allows QoS levels to be adjusted accordingly.

2.3 Variable Application Demand

The final source of dynamics that we wish to consider is one that is common to both mobile ad hoc and conventional networks. That is: variable demand on network resources by end-system applications.

The conventional response to variable application demand is based on admission control. That is, some end users may be denied access in order to preserve QoS for those that have been admitted. This results in some users being granted the requested QoS, at the expense of others being denied service or forced to use a lower grade service-model, for example best-effort. Admission control may be performed on a "first come, first served" basis, or it may include priority and preemption mechanisms to implement some desired policy.

Admission control is a valuable tool for dealing with variable application demand for QoS, because it is generally better to deny service to some users in order to grant service to others, rather than allowing a "free-for-all" which results in an unacceptable QoS for all users. The limitation with this approach, however, is that it provides only an "all or nothing" service. For many applications, it may be preferable to admit a larger number of users while continuing to provide QoS support, but at a lower level of service. In order to do this, the network and applications need some way to communicate about the levels of QoS that would be acceptable and the level that can currently be supported. The "Dynamic QoS" approach described in the next section allows this communication. Therefore, not only does this approach provide a means for dealing with network dynamics due to

wireless links and node movement as discussed above, it also has the significant benefit of allowing more flexible sharing of available QoS resources among applications. This could be valuable in conventional as well as mobile ad hoc networks.

3. The Dynamic QoS Approach

In this section we provide a general discussion of the approach we have taken to tackle the problems described above. A description of the protocol that implements this approach is provided in the following section.

3.1. Reservation Ranges

In conventional QoS mechanisms, such as internet integrated services (intserv) or ATM, a reservation can be represented by point in an n-dimensional space, with coordinates defining the characteristics of the service. For example, with the controlled load service in an intserv network a reservation contains an n-tuple defining the token bucket traffic specification parameters (r, b, p, m, M), where r is the average rate, b is the token bucket depth, p is the peak rate, m is the minimum policed unit, and M is the maximum packet size [14]. Other network types and QoS service models may use other parameters, such as maximum delay, delay jitter, packet loss rate, and so on, to specify a reservation. Acceptance of a reservation means that the network makes a "commitment" to provide the service with characteristics specified by the reservation n-tuple[1].

The problem with representing a reservation as a single point is that a binary decision must be made: either service can be provided as requested, or no service commitment (beyond best-effort) is given at all. This does not allow the flexibility needed for handling the types of dynamics discussed above; it does not allow any way to respond to varying network resources, and it does not allow any way to respond to varying application demand on available resources.

To solve this problem, we allow a reservation to specify a range of values, rather than a single point. We focus on the average data rate, and we replace the single parameter r with a range: (r_{min} to r_{max}). The other parameters (b, p, m, M) in the reservation request may also represent ranges, however in our implementation, although they are carried through the network and taken into account when reservations are aggregated, little is done with the other parameters. For our implementation, which uses class-based queuing [15] and the controlled load service model, this simplification works very well, since average data rate is in fact the critical managed resource. The CBQ implementation we have utilized in our testbed ([16], [17])] creates a separate queue for each flow, and services each queue in such a way as to attempt to provide that queue with its allocated share of the outgoing interface bandwidth. For the RTP-based streaming video and audio applications we have experimented with in our testbed, controlled load turns out to be a very useful service model, and average transmission bandwidth does turn out to be the critical reservation parameter.[2]

By treating a reservation as a request for service somewhere within a specified range, we create the flexibility needed to deal with all of the types of network dynamics discussed in the previous section. As available resources change, the network can readjust allocations within the reservation range. If resources decrease below the level currently allocated, the

[1] In the controlled load service model, acceptance of a reservation with these parameters means that the network can provide service equivalent to that which could be expected on an uncongested network, as long as the traffic flow does not exceed the token bucket specification. In the guaranteed QoS service model the acceptance of a reservation with these parameters, with a specified reservation rate R and slack term S, means that the network can provide end-to-end behaviour conforming to the fluid model, with a known (computable) maximum delay, as long as the offered traffic does not exceed the token bucket specification.

[2] We are not alone in our focus on bandwidth as the key parameter to be managed in providing QoS. A similar focus can be found in many of the related works cited in this paper.

network can offer a more reasonable response than simply dropping the reservation, as long as there is some point within the reservation range that can be supported. Similarly, as the number of application flows competing for resources increases, rather than simply refusing to admit new flows or preempting existing flows, the network can attempt to adjust the allocation for all flows, so that each flow can be accommodated at some point within its requested range.

For service models other than controlled load, further study is required to determine the extent to which it is appropriate to focus on average data rate as the parameter which is represented as a range of values. In particular, in a service model which guarantees zero packet loss, output buffer size becomes a critical managed resource, and therefore token bucket depth becomes a critical parameter in reservation requests. The network may find that it can meet its commitment at a given average transmission rate, but only if it is allowed to adjust the token bucket depth. In a service model that includes bounds on delay variation, peak transmission rate may become a critical parameter. In order to prevent jitter, a network node may need to respond to input bursts, up to the specified peak rate, by creating a corresponding output burst. Peak transmission rate would therefore become a critical resource to be managed on outbound interfaces, in addition to average transmission rate. In situations such as these, it may still be possible to simplify the model by assuming that only one reservation parameter (probably still average transmission rate) should be specified as a range, while the other values are specified as scalars. The network would have flexibility to adjust the allocation of one parameter within the reserved range, in order to maintain its commitments for all parameters. For example, jitter could be prevented by limiting average flow transmission rates to ensure sufficient spare transmit bandwidth to allow transmission of a burst on one flow without affecting other flows. Application of the concept of reservation ranges to other service models remains a topic for further study. Throughout the remainder of this paper, we make the assumption that a reservation can be considered to be a bandwidth range (minimum to maximum usable bandwidth) and that other reservation parameters are specified as scalars.

3.2. Implications for Applications

Obviously, treating reservations as ranges has implications for applications. First, the application must have some notion of the QoS range within which it can operate. At first glance, this may seem to add complexity, but in fact we believe that in most cases this actually simplifies the task of the application programmer. When a QoS request must specify a scalar value, the application programmer has no way to know what this value should be. The solutions to this problem are not very good. For example, the application could be programmed to start out requesting the maximum level of service that the application might want to use, and if this request fails admission control, continue to make requests for successively lower levels of service until a request succeeds. Another approach might be to have the application prompt the user for the speed at which the end system is connected to the internet (dial-up modem, ISDN line, DSL, cable modem, etc.)[3] and then, assuming that this will be the bottleneck, base the requested reservation request on this value. Clearly, both of these approaches have disadvantages, and neither is suitable for use in a mobile ad hoc network. On the other hand, if reservations are treated as ranges, the application programmer has a reasonable chance of being able to determine a priori the minimum service level at which the application can reasonably operate and the maximum service level that it could utilize. These values can then be programmed in (or configured by the user according to his intended use of the application) as the QoS range that application will use in

[3] This is a fairly common practice, for example, web-based servers for downloading application packages servers often prompt the user for the connection speed, and then customize the package to be downloaded to provide the maximum functionality within an "acceptable" download time.

reservation requests. Then, at run time, when a QoS request is made, the network will provide feedback on the current allocation within this range that can be supported, and the application can react accordingly.

The second implication is that applications should be capable of adapting their behavior at run time based on the current allocation, which must be provided as feedback from the network to the application. This does require additional sophistication on the part of the application, but we do not believe this is an unreasonable task. In Section 6 we will discuss how we added adaptive behavior to streaming audio and video applications, which dynamically adjust the encoding scheme (for audio) and frame rate (for video) in response to changing allocations within the reservation range. We are also currently collaborating with another group to implement an adaptive web server which provides content that is appropriate to the available network resources between browser and server.

Finally, in order to support the inclusion of range-based reservations, and adaptation to changing network allocation, the application program interface (API) must be extended to support this new paradigm.

4. The Dynamic RSVP (dRSVP) Protocol

We created a distributed network protocol for the "Dynamic QoS" concept by extending the RSVP protocol. We then implemented this protocol by modifying and extending ISI's implementation of RSVP [18]. In this section we describe our new protocol, with the assumption that the reader has some familiarity with RSVP and the intserv architecture. We made the following extensions and modifications to standard RSVP:

a) We added an additional flow specification (flowspec) in Resv messages, and an additional traffic specification (tspec) in Path messages, so that they describe ranges of traffic flows,

b) We added a "measurement specification" (mspec) to the Resv messages, which is used to allow nodes to learn about "downstream" resource bottlenecks,

c) We created a new reservation notification (ResvNotify) message, which carries a "sender measurement specification" (smspec) that is used to allow nodes to learn about "upstream" resource bottlenecks,

d) We changed the admission control processing to deal with bandwidth ranges,

e) We added a bandwidth allocation algorithm that divides up available bandwidth among admitted flows, taking into account the desired range for each flow as well as any upstream or downstream bottlenecks for each flow,

f) We extended SCRAPI, the simplified RSVP API [19], to deal with bandwidth ranges.

An overview of the protocol is provided below. Detailed descriptions of the dRSVP messages, dRSVP message processing rules, and the new API are provided in a separate paper. *(Note to MobiHoc reviewer: the paper containing the protocol details has been written and will be submitted for publication. Before this paper goes to "press" we expect to be able to provide a full reference here to the paper containing the protocol details.)*

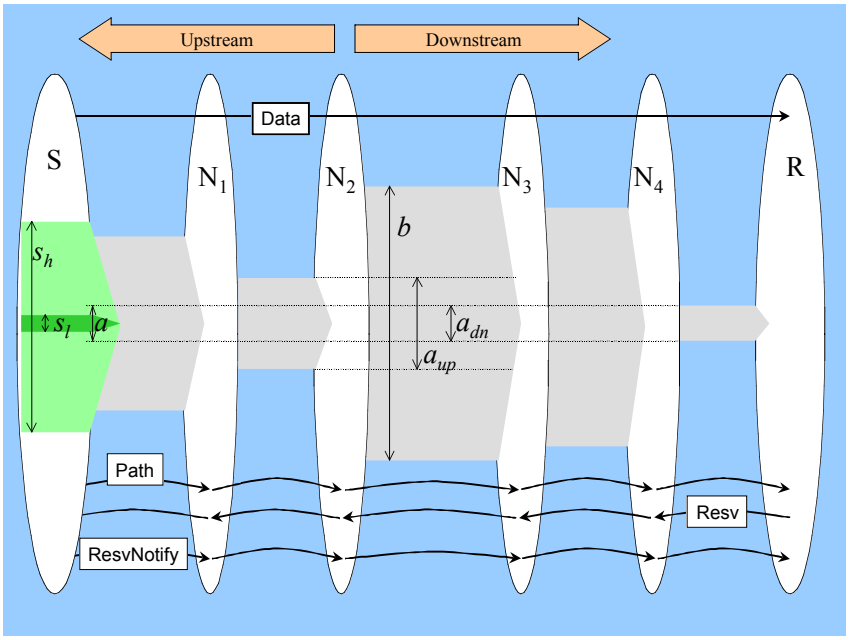

Figure 5-1. dRSVP Protocol Overview

Figure 5-1 illustrates a simple network in which node S sends data to node R through intermediate nodes N_1, N_2, N_3, and N_4. The nodes are connected by links, shown in the figure as wide bars, with the width of the bar corresponding to the bandwidth available on the link. The adaptive application running on node S can generate data at rates within the range from s_l to s_h. These values are communicated in Path messages, which flow through the network hop by hop, following the same route as the data messages, to the receiver R. Upon receipt of the Path messages, the receiving application on R requests a reservation for this flow, with QoS range (s_l, s_h). The request is carried through the network in Resv messages which reverse the route followed by the Path messages. Each node, upon receiving the Resv message, performs an admission control check and computes the bandwidth, within the range (s_l, s_h), that it can allocate to the flow. Assuming the admission control test passes, the Resv message is propagated upstream towards S. The Resv message also contains a "receiver measurement specification" value, denoted m_r. The value of m_r is initialized at R to s_h, but as each node propagates the Resv message upstream, if the bandwidth allocation that it is able to give the flow is less than the received m_r value, it reduces the m_r value to the allocation. S applies similar logic to the Application Program Interface (API), treating as if it were an upstream link, so the application receives a, as shown, indicating the bandwidth that has been reserved end-to-end through the network. The application must adapt to a in order to receive the agreed-upon service (controlled-load, in our implementation). If the reservation successfully propagates all the way through the network, node S initiates a ResvNotify message. This contains a "sender measurement specification", denoted m_s. initialized at S to s_h. This propagates toward R and in a similar fashion each node limits the value of m_s to the bandwidth that it is able to allocate for the flow.

Since reservation requests are for ranges, rather than specific values, a key issue in dRSVP is what level of resources to allocate within the requested range. Let us consider how this decision is made at node N_2, which must determine how much bandwidth, out of a total of b on the link to N_3, to allocate to the flow. Within one round trip (after the reservation was issued at R), N_2 has learned that there is a bottleneck somewhere downstream (on the link from N_4 to R) with a value of a_{dn}, and that there is a bottleneck upstream (on the link from S to N_1) with a value of a_{up}. Node N_2 has total bandwidth b available on the link, so if this is the only flow present, it can allocate the maximum requested, s_h, but it need not reserve more than a_{dn}. If other flows are present, the dRSVP algorithm divides the available bandwidth among the flows, attempting to give each flow its minimum requested bandwidth, plus an equal fraction of its requested bandwidth range. This algorithm takes into account upstream and

downstream bottlenecks for every flow, so that it avoids reserving resources for a flow that could not be utilized due to bottlenecks elsewhere in the network.

Now consider what happens when the bandwidth on the link from N_4 to R increases, as shown in Figure 5-2. The next

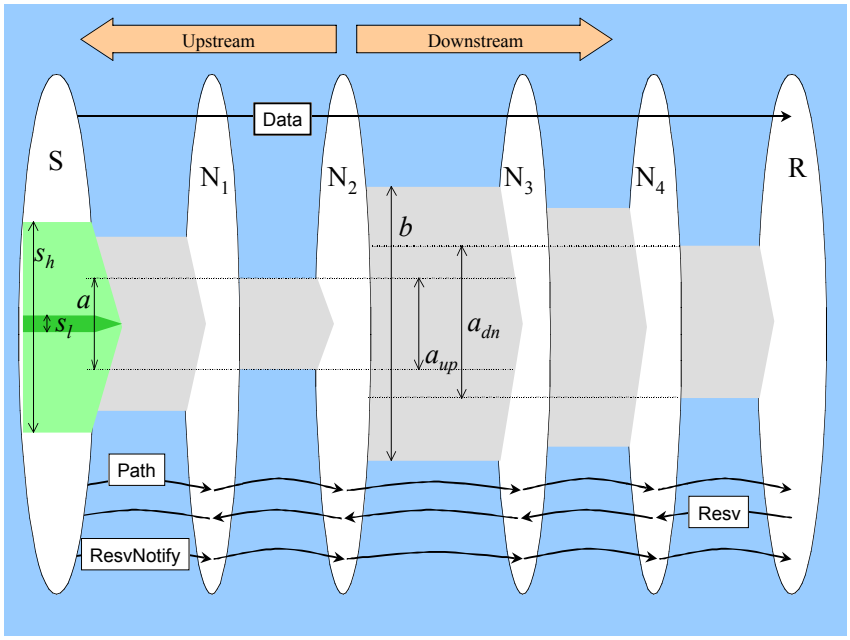

Figure 5-2. dRSVP Response to Link Bandwidth Changes

Resv message propagated by N_4 will allow all upstream nodes to learn that the bottleneck has been removed. The bottleneck is now the link between N_1 and N_2. The figure shows the new values of a_{dn} and a_{up} that will be computed at node N_2. N_2 now knows that this flow could now utilize up to a_{up}, as shown; remaining bandwidth can be freely allocated to other flows. Assuming that this is the only flow present, the bandwidth allocation provided to the application at S is now a, as shown.

The processing rules for the dRSVP protocol also include aggregation of Path and Resv messages for multicast flows.

5. Related Work on QoS in a Dynamic Network Environment

A survey of the literature reveals that there are three problems that are often brought up when attempting to provide QoS in a wireless and/or mobile network environment:

1) *The QoS routing problem*: How to find a route through the network that is capable of supporting a requested level of QoS.

2) *The QoS maintenance problem*: How to ensure that, when network topology changes, new routes that can support existing QoS commitments are available, or can be quickly found.

3) *The variable resource problem*: How to respond to changes in available resources, either as the result of a route change, or as the result of changes in link characteristics within a given route.

These issues are obviously very closely related, but we have found it useful to distinguish among them when examining the work in this area. In particular, we believe that the QoS routing problem should be decoupled from the other problems, and that a solution to the variable resource problem can obviate the need to solve the QoS maintenance problem. We will elaborate on these points as we examine related work in this area by other researchers.

Work on the QoS routing problem in mobile ad hoc networks has been documented by Chen and Nahrstedt [18]; Lin and Liu [21]; and Sivakumar, Shiha, and Bharghavan [22]; among

others. We do not discuss these papers here, since our work has not centered on this problem.

Solving the QoS maintenance problem in a general mobile ad hoc network is extremely difficult, and the published body of work in this area has generally assumed networks in which only the end systems move. The QoS maintenance problem is then reduced to a QoS handoff problem; that is: how to maintain QoS when an end system node is handed off from access node to another. The QoS handoff problem, though still challenging, is more tractable, and has been tackled by several researchers. One significant effort tackling the QoS maintenance problem (for mobile end systems) is that of Talukdar, Badrinath, and Acharya, at Rutgers University [23]. The solution proposed by these authors assumes that, at least for some mobile users, mobility will be predictable. That is, the mobile end systems will be able to provide a mobility specification that defines which access points they will visit. Three new mobility-related service models are proposed: Mobility Independent Guaranteed (MIG), Mobility Independent Predictive (MIP), and Mobility Dependent Predictive (MDP) service. With MIG and MIP service, mobile end systems are provided guaranteed or predictive service[4], respectively, as long as they move in accordance with their mobility specification. With MDP service, mobile end systems are provided a "weak" form of predictive service as long as they move in accordance with their mobility specification. The MIG and MIP service are provided by preallocating resources on all links that may be needed by a mobile host, wherever it might move in the network, as predicted by its mobility specification. In order to avoid dramatic underutilization, the MDP service is provided using resources that have been allocated to other flows, but which are currently being underutilized. This means that a mobile node receiving MDP service may experience QoS degradation when the node moves into another cell, or when other nodes receiving MIG or MIP service move into the same cell. An implementation of the MIG, MIP and MDP service models, based on modified versions of Mobile IP and RSVP, is described in [24].

We observe that the MDP service suffers from the variable resource problem, and no solution for this problem is proposed in [23] or [24]. That is, there is no provision for varying link bandwidths, and no flexibility to adjust QoS levels to deal with changes in resources. An interesting and potentially valuable approach might be to combine the concept of adaptive QoS ranges with the MDP service model, creating a Mobility Dependent Predictive service with Adaptive QoS.

In [4] and [5], Lu, Lee, and Bharghavan propose a system for solving both the QoS maintenance problem (for the handoff case only) and the variable resource problem. The QoS maintenance problem is tackled by performing advance resource reservation in cells that are neighbors of the cell currently occupied by a mobile end system. The variable resource problem is tackled by adaptively re-adjusting the quality of service within prenegotiated bounds. We believe that this concept (treating reservations as ranges and allowing the network the flexibility to adjust QoS within this range) is crucial to solving the resource change problem, and we have adopted this concept as the basis for our work. However, we have interpreted the concept differently. In [4], the concept of QoS ranges is interpreted to mean that a QoS "guarantee" is provided for the low end of the range, and best effort service is provided beyond that point:

> *"...service is guaranteed in the sense that each connection is guaranteed its minimum QoS requirement; the service is best effort in the sense that the network provides best-effort service beyond the minimum QoS support."*

[4] Here the terms guaranteed and predictive are as defined by Clark, Shenker, and Zhang [23].

In contrast, our interpretation of the notion of QoS ranges is that the network provides service at a signaled QoS allocation point within the range requested in QoS reservation request. Service is "guaranteed"[5] at the allocated point, however the network may change this allocation point at any time. The application should be capable of adapting its transmission characteristics to make use of the allocated QoS. Our implementation provides a controlled load service, which means that if the application adjusts its transmission to stay within the current allocation it will not experience congestion. In this regard, our approach could be considered a form of congestion control with explicit rate indication, as described by Charny, Clark, and Jain [26]. However, our approach could also be extended to other service models, for example, Guaranteed QoS [27].

In addition to a somewhat different interpretation of the concept of reservation ranges, the protocol mechanism and algorithms we have used to realize this concept are different from those presented in [4] and [5]. One significant difference is that our system supports multicast flows, which creates a requirement for aggregating information at branch points. Another major difference comes in the distributed bandwidth allocation algorithm. The bandwidth allocation algorithm presented in [4] is an adaptation of the explicit rate indication congestion control algorithm first presented by Charny et. al. in [26]. Our algorithm was developed independently as part of our research. Whereas the Lu-Bharghavan algorithm attempts to give each flow the minimum bandwidth requested, plus an equal amount of any "excess" bandwidth, our algorithm attempts to give each flow the minimum bandwidth requested, plus an equal fraction of the requested bandwidth range. Both algorithms have the concept of flow "bottlenecks", but use different protocol mechanisms for identifying the bottlenecks. Preliminary analysis and lab experience indicate that our algorithm converges quickly (in one round trip), but more work remains to rigorously analyze the convergence performance of our algorithm.

A significant difference between our work and that presented by Lu, and Bharghavan in [4] is that our approach does not include an explicit solution to the QoS maintenance problem. The approaches presented in [4] and in the other papers discussed below may offer promise for solving this problem in the limited case where only end systems move (the QoS handoff problem). However, due to our special interest in military applications, we have targeted our work for use in networks where intermediate system nodes may move, that is: mobile ad hoc networks. In these networks, routing is a difficult problem, QoS routing is even more difficult, and a practical solution to QoS routing which will ensure that QoS can be maintained as new routes are selected is perhaps intractable. Therefore, we believe that it is important to decouple these problems. We do not attempt to tackle the QoS routing problem or the QoS maintenance problem. Rather, our approach relies on "soft state" to reestablish QoS along the new route when a change occurs. Then, we rely on our concept of adaptive QoS to deal with the fact that a different level of resources may be available along the new route. Thus we avoid the QoS maintenance problem, and our solution to the resource change problem nevertheless allows relatively seamless provision of QoS. However, the reliance on soft-state mechanisms means that, when a route changes, there will be a period during which the traffic receives only best-effort service. State refresh messages must propagate end to end (round trip) before QoS will be reestablished, probably at a new allocation level, along the new route. Nevertheless, we believe that there is a large class of applications which can tolerate transient periods of degraded service, yet benefit from the Dynamic QoS mechanisms we have proposed.

The INSIGNIA in-band signaling system for supporting QoS in mobile ad hoc networks [7] also relies on soft-state and

[5] We use the term "guarantee" loosely here; the exact meaning of the term "guarantee" depends on the service model.

adaptability for dealing with the resource change problem. As with our Dynamic QoS approach, INSIGNIA includes mechanisms for signaling QoS requirements along the new route, and adapting to changes in available resources. By incorporating its signaling into the data packet headers, INSIGNIA has the significant advantage of potentially very rapid reestablishment of QoS along the new path. The concept of adaptability in INSIGNIA hinges on the data flow being hierarchical – consisting of a "base layer" and an "enhancement layer". This may be suitable for certain important applications (in particular video), but not all application data lends itself to this kind of layering. We believe that the concept of bandwidth ranges, as proposed by Lu and Bharghavan and further developed in our work as well as in the Mobiware effort (discussed below), offers considerably more generality. An interesting possibility would be to combine the in-band signaling of INSIGNIA with a bandwidth allocation algorithm and more general adaptive applications.

The Mobiware effort [6] has created an open and active programmable mobile testbed for experimenting with mobile and wireless networking. In the Mobiware testbed CORBA is used to signal handoffs of WaveLAN-equipped mobile end systems connected to an IP-over-ATM wired network. The Mobiware work does not explicitly tackle the QoS maintenance problem; that is, the pre-allocation approaches discussed earlier are not applied, so when a handoff occurs there is no guarantee that QoS will be maintained. However, the Mobiware work does tackle the resource change problem. If available resources change (either because of a handoff between cells or because of transmission quality changes within a cell), there is a mechanism for dealing with this change. Application adaptation to varying QoS in the experiments described in [6] was performed by passing the application flows, which consisted of hierarchically encoded video streams, through a filter deployed at the access point. The filter will drop or add layers containing more or less of the video flow in response to changing QoS. We should also note that the Mobiware work appears to be extremely general; the mechanism described above is just one approach that could be implemented in the Mobiware testbed.

The Mobiware work also explored the notion of a utility-fair bandwidth allocation algorithm, proposed by Bianchi, Campbell, and Liao [28]. In our work, as in that of Lu, Lee, and Bharhavan [5], an application defines QoS in terms of a bandwidth range. Allocating bandwidth within this range can be done in different ways. The bandwidth algorithm described in [5] allocates bandwidth by giving each flow the minimum plus an equal amount of the "excess" bandwidth. Our algorithm allocates bandwidth by giving each flow the minimum plus an equal percentage of its requested bandwidth range. The approach proposed in [28] and utilized within Mobiware allows each flow to specify how much "utility" it will receive from each unit of extra bandwidth. "Excess" bandwidth is then allocated so that each flow receives an equal "utility". This approach could have benefit, in particular for applications whose utility functions are step functions (a multi-resolution hierarchical flow, for example, can not make use of extra bandwidth unless it is sufficient to add another layer to the flow). However, whereas the CORBA-based signaling used in Mobiware can support the complexity of including an arbitrary function definition as part of an application's reservation request, it is unclear whether this level of complexity could be supported with a more lightweight signaling mechanism. Furthermore, we should point out that the Mobiware work appears to deal only with the case where the bottleneck location in the network is known to be at the last hop from the access point to the mobile end system. This will clearly not be the case in a mobile ad hoc network, and it may not be the case in other network architectures, for example, a fixed topology network that includes a number of variable-quality wireless links internal to the network.

6. Implementation Experience

6.1 The Dynamic Networking Testbed

In order to test and evaluate our implementation of the adaptive QoS protocol described above, we have created a testbed in which we can vary resources available in a network of routers. These routers can be configured in a variety of complex network topologies. The routers are PCs, running the FreeBSD operating system, with CPU speeds from 90 to 300 MHz. We implemented the dRSVP protocol described in this paper by modifying a version of ISI's rsvpd (RSVP daemon) code [18] together with class based queueing (CBQ) code from the Alternate Queueing (ALTQ) package [16], [17].

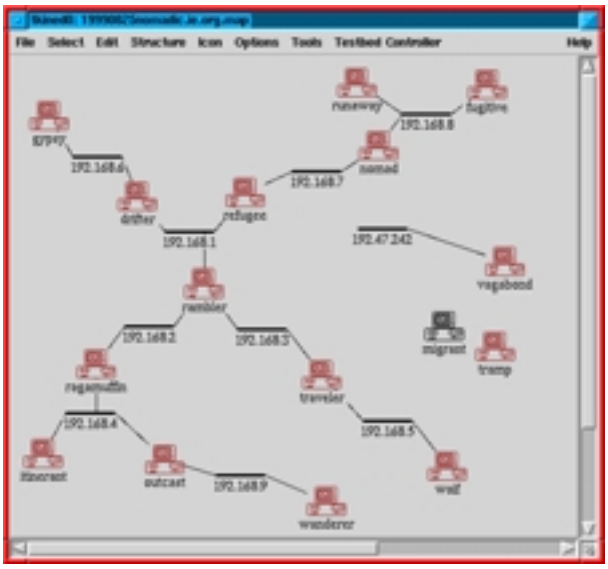

Figure 6-1. Testbed Controller Application

Our testbed does not yet include mobile nodes, but it does include the ability to emulate the effects of dynamic link characteristics. To accomplish this we created a centralized testbed controller application, which was built using the Tkined/Scotty package [29]. The testbed controller provides a GUI as well as a scripting facility from which we can set the speed of any of the links in the testbed. The testbed controller sends commands to a bandwidth manager daemon resident on each router in the testbed, which then implements the link speed change command by interacting, via rsvpd, with the ALTQ package. The link speed change is effected by modifying the queue service parameters used in CBQ. By examining the link traffic, we verified that this strategy is an effective way to vary the effective link speed seen by the network layer. The only problem we have observed with this technique is that the interface bandwidth actually consumed by CBQ is somewhat sensitive to packet size. Flows with small packets tend to be under-served; that is, they actually receive less bandwidth than specified.

Figure 6-1 is a screenshot of the Testbed Controller application. The screenshot shows the routers and links of the testbed, in a configuration we use for running demonstrations of the dRSVP protocol.

6.2 Test Applications

In order to verify the dynamic QoS mechanisms added to RSVP and the adaptive QoS API, we have created a set of adaptive applications. First, we modified the RSVP-aware multicast data generator and receiver applications mgen and drec, created by Naval Research Labs (NRL). These applications provide a nice tool for generating test traffic for laboratory experiments, however they do not provide much insight into the value of dynamic QoS for a realistic application. For that, we selected the streaming video and applications vic and vat, which were created by LBL, and modified to be RSVP-aware by ISI. We applied some rather extensive modifications to these applications to make these applications adaptive and to work with our extended API, called dSCRAPI.

Figure 6-2 shows a screenshot of the user interface of our modified version of the video tool vic. A key feature of this interface is the "Rate Control" slider. In the normal version of vic, this slider enables the user to control the maximum rate, in kilobits per second, at which video data is transmitted. When a new transmission rate is selected, the application will adjust the frame rate as necessary to stay within this limit. Those familiar with the standard version of this application will notice that we have added an "Adaptive Transmit" button alongside the standard "Transmit" button. The "Transmit" button invokes normal vic operation, in which the transmission rate slider is controlled manually by the user. The new "Adaptive Transmit" button invokes the new adaptive QoS behavior. In this mode, the transmit rate slider is no longer controllable by the user. Instead, the transmission rate is set to the rate allocated by the network in response to reservation requests from receivers. The end points of the transmission rate slider (10 Kbps and 1 Mbps in Figure 6-1) determine the QoS range that will be requested; this range is configurable by the user at application start-up time from the command line or a configuration file. In "Adaptive Transmit" mode, our version of vic automatically adapts to changes in transmission rate signaled by the network (via the dSCRAPI API) by adjusting the frame rate. The slider on the GUI moves to show the current transmission rate, and the color of the slider shows dRSVP reservation status – pending, reserved, or failed. Image quality and other video parameters (encoding format, color versus monochrome, etc.) can be manually adjusted by the user, using the sliders and buttons available in the GUI. These parameters will affect the frame rate that can be achieved at a given transmission rate. For example, at slow transmission rates reasonable frame rates can be obtained by decreasing image quality. While our implementation's QoS adaptation mechanism consists simply of adjusting the frame rate, a more sophisticated application could use some heuristics to choose a complete set of transmission parameters appropriate to the network conditions.

We also modified the audio tool vat. Again, we added an "Adaptive Transmit" button, which causes the application to transmit at a rate that can be reliably received by all destinations. In this case, the application adapts to changes in network bandwidth by changing the encoding algorithm to one which requires no more than the available bandwidth signaled by dRSVP via the dSCRAPI API.

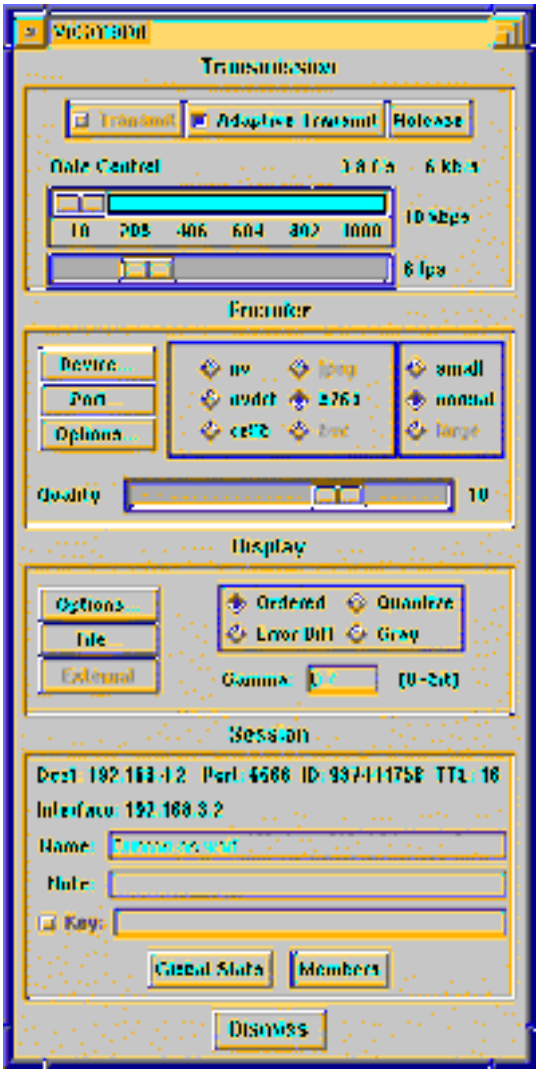

Figure 6-2. Screenshot of "vic" Video Application

While our version of vat worked well, and provided a good demonstration of the value of dynamic QoS, results with vat were less impressive than with vic, and quality was not always excellent even with reservations in place. Part of the reason for this is that the controlled load service model is not as successful for audio as it is for video. Another reason is that the CBQ implementation underlying the QoS mechanisms in our routers is sensitive to packet size, and under-serves queues with many small packets, as is the case with audio flows. A possible avenue for further experimentation would be to experiment with an audio tool such as vat, enabled with dynamic QoS together with an implementation of the Guaranteed-Load service model and a more refined CBQ implementation.

The modifications we made to vic and vat illustrate the suitability of the adaptive QoS paradigm for programming streaming applications. The source knows the range of network bandwidth it is capable of utilizing, the actual transmission rate is selected dynamically according to current network conditions, and the application adjusts its transmission rate to match what the network can currently support.

7 Conclusions

With our testbed, applications, and dRSVP implementation, we are able to demonstrate a complete system in which QoS support is maintained even while link bandwidths vary within the network. The dRSVP protocol functions correctly, and divides bandwidth among multiple applications, which are able to adapt to varying QoS levels provided by the network in response to link variations. Our experience in developing this capability has convinced us that an adaptive QoS approach is both feasible and potentially valuable. We are in the process of gathering quantitative results on various aspects of our implementation; for example we plan to gather data on protocol overhead under various conditions in order to analyze scalability. We also would like to add wireless hardware into our testbed, and experiment with node movement and a dynamic topology. Many interesting possibilities remain open for investigation. One possible area would be to apply the notion of reservation ranges to other service models, for example Guaranteed QoS [27]. Another area for further study is the interaction between adaptive QoS and a variety of different link layers, in particular a shared media link layer with a subnet bandwidth manager for link layer resource management. We would also like to investigate how an adaptive QoS approach such as that proposed in this paper would fit into an overall intserv/diffserv environment. Other possible areas for further research include integrating an adaptive QoS approach with a (separate) QoS routing solution, and applying the notion of bandwidth ranges together with a lightweight QoS signaling mechanism such as INSIGNIA, in a mobile ad hoc network environment. Our hope is that the concepts and experience documented in this paper will encourage further research into the concept of dynamic QoS and adaptive applications for mobile ad hoc networks.

Acknowledgements

The authors would like to acknowledge the contributions of Bob Moose, who led this project during its initial phase.

References

[1] R. Braden, L. Zhang, S. Berson, S. Herzog, S. Jamin. *Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification*, IETF RFC 2205, September 1997.

[2] R. Braden, L. Zhang; *Resource ReSerVation Protocol (RSVP) -- Version 1 Message Processing Rules*, IETF RFC 2209, September 1997.

[3] M. Stemm, R. Katz; *Vertical Handoffs in Wireless Overlay Networks*; ACM Mobile Networking (MONET), Special Issue on Mobile Networking in the Internet; Winter 1998.

[4] S. Lu, V. Bharghavan; *Adaptive Resource Management Algorithms for Indoor Mobile Computing Environments*; Proceedings of ACM SIGCOMM '96, Univ. of Illinois at Urbana-Champaign, Stanford, CA; August 1996.

[5] S. Lu, K. Lee, V. Bharghavan; *Adaptive Service in Mobile Computing Environments*; from *"Building QoS into Distributed Systems"*; Campbell and Nharstedt (editors); Chapman & Hall; 1997.

[6] O. Angin, A. Campbell, M. Kounavis, R. Liao, *The Mobiware Toolkit: Programmable Support for Adaptive Mobile Computing*, IEEE Personal Communications Magazine, August 1988.

[7] S. Lee and A. Campbell, *INSIGNIA: In-band Signaling Support for QOS in Mobile Ad Hoc Networks*, Proc of 5th International Workshop on Mobile Multimedia Communications (MoMuC,98), Berlin, Germany, October 1998.

[8] R. Bagrodia, W. Chu, L. Kleinrock, C. Popek, *Vision, Issues, and Architecture for Nomadic Computing [and Communications]*, IEEE Personal Communications Volume: 2 6 , Page(s): 14 –27, December 1995.

[9] ITU Telecommunication Standardization Sector of ITU, *V.90 - A Digital Modem and Analogue Modem Pair for Use on the Public Switched Telephone Network (PSTN) at Data Signalling Rates of up to 56000 Bit/S Downstream and up to 33600 Bit/S Upstream*, Section 9.6 – Rate Renegotiation, September 1998.

[10] Scientific Research Corp, *Turbolink™ White Paper*, January 2000. http://www.scires.com/turbo.htm

[11] D. Beyer, T. Frivold, J. Hight, M. Lewis, *API Framework for Internet Radios*, September 1998. ftp://ftp.rooftop.com/pub/apis/api_framework.pdf

[12] Yavatkar, Hoffman, Bernet, Baker, Speer, *SBM (Subnet Bandwidth Manager): A Protocol for RSVP-based Admission Control over IEEE 802-style Networks*; IETF Internet Draft (work in progress); January 2000. http://www.ietf.cnri.reston.va.us/internet-drafts/draft-ietf-issll-is802-sbm-10.txt

[13] E. Horlait, M. Bouyer, *CLEP (Controlled Load Ethernet Protocol): Bandwidth Management and Reservation Protocol for Shared Media*, Internet Draft (work in progress): draft-horlait-clep-00.txt, July 1999.

[14] J. Wroclawski, *Specification of the Controlled-Load Network Element Service*, Internet Engineering Task Force Request For Comments Number 2211, September 1997.

[15] S. Floyd, V. Jacobson; *Link-Sharing and Resource Management Modules for Packet Networks*; IEEE/ACM Transactions on Networking, Vol. 3, No. 4, pp 365-386; August 1995.

[16] K. Cho, *A Framework for Alternate Queueing: Towards Traffic Management by PC-UNIX Based Routers*, Proceedings of USENIX 1998 Annual Technical Conference, New Orleans LA, June 1998. www.csl.sony.co.jp/~kjc/kjc/papers.html

[17] K. Cho, *Managing Traffic with ALTQ*; Proceedings of USENIX, 1999 Annual Technical Conference: FREENIX Track, Monterey CA; June 1999. www.csl.sony.co.jp/~kjc/kjc/papers.html

[18] ISI, The RSVP Implementation developed by the University of Southern California (USC) Information Sciences Institute (ISI). http://www.isi.edu/div7/rsvp/rsvp.html

[19] B. Lindell, ISI; *SCRAPI – A Simple 'Bare Bones' API for RSVP*; Work in progress (draft-lindell-rsvp-scrapi-00.txt), August 10, 1998. http://www.isi.edu/rsvp/DOCUMENTS/draft-lindell-rsvp-scrapi-02.txt

[20] S. Chen, K. Nahrstedt, *Distributed Quality-of-Service Routing in Ad Hoc Networks*, IEEE Journal on Selected Areas in Communications, Vol 17, No. 8, August 1999.

[21] C. Lin, J. Liu, *QoS Routing in Ad Hoc Wireless Networks,* IEEE Journal on Selected Areas in Communications, Vol 17, No. 8, August 1999.

[22] R. Sivakumar, P. Sinha, V. Bharghavan *CEDAR: A Core-Extraction Distributed Ad Hoc Routing Algorithm*, IEEE Journal on Selected Areas in Communications, Vol 17, No. 8, August 1999.

[23] A.Talukdar, B. Badrinath, Rutgers University, and A. Acharya, C&C Research Labs, *On Accommodating Mobile Hosts in an Integrated Services Packet Network*, NEC USA, Princeton, NJ Proceedings of INFOCOM '97, Kobe, Japan, April 1997.

[24] A. Talukdar, B. Badrinath, *MRSVP: A Reservation Protocol for an Integrated Services Packet Network with Mobile Hosts*, Department of Computer Science

Technical Report DCS-TR-337, Rutgers University, July 1997.

[25] D. Clark, S. Shenker; *Supporting Real-Time Applications in an Integrated Services Packet Network: Architecture and Mechanism;* Proceedings of IEEE SIGCOMM; 1992.

[26] A. Charny, D. Clark, R. Jain, *Congestion Control with Explicit Rate*, Indication Proceedings, ICC, June 1995.

[27] S. Shenker, C. Partridge, R. Guerin, *Specification of Guaranteed Quality of Service*, Internet Engineering Task Force Request For Comments Number 2212, September 1997.

[28] G. Bianchi, A. Campbell, R. Liao, *On Utility-Fair Adaptive Services in Wireless Networks,* Proceedings IEEE Sixth International Workshop on QoS, pp 256-267, 1998.

[29] J. Schönwälder, H. Langendörfer; *Tcl Extensions for Network Management Applications*, Proceedings of the 3rd Tcl/Tk Workshop; Toronto (Canada); July 1995. http://wwwhome.cs.utwente.nl/~schoenw/scotty/