Managing QoS for Multimedia Applications in a Differentiated Services Environment

Manish Mahajan and Manish Parashar
The Applied Software Systems Laboratory, Dept. of Electrical and Computer Engineering, Rutgers University.
94 Brett Road, Piscataway, NJ-08855-1390
{manishm, parashar}@caip.rutgers.edu

(Extended Abstract)

1. Introduction

This paper presents design, implementation and evaluation of a content-aware bandwidth broker (CABB) that manages QoS of multimedia applications in a differentiated services (diffserv) environment. CABB controls multimedia flows between diffserv domains using service policies defined based on the requirement, nature and adaptability of the flow.

The overall quality of network connections (e.g. link capacity, available end-to-end bandwidth, congestion, etc) has a significant impact on the performance of networked applications. These applications generate traffic at varying rates and have a varying ability to tolerate delays and jitter in the network. Networked multimedia applications, which form a large part of today's Internet traffic, present a significant challenge, as they require quality guarantees from the network. Internet Protocol (IP) however, is best effort and does not provide any Quality of Service (QoS) guarantees [4] – that is, there are no mechanisms in IP for policing or controlling unresponsive and high bandwidth flows that can cause congestion in the network. As a result, all QoS management is left to the application [9]. Multimedia applications have very limited feedback control to stop them from causing congestion in the network. Consequently, QoS management for networked multimedia applications over IP is a significant and immediate challenge.

Different mechanisms such as Integrated Services (e.g. RSVP [19]), Differentiated Services [2], and Multi-Protocol Label Switching (MPLS) [20], have been proposed for providing some level of QoS over the unpredictable IP network [4]. These mechanisms provide improved network QoS to the applications at the edges of the network by identifying, handling and controlling traffic using scheduling and resource reservation mechanisms. The Differentiated Services (diffserv) network architecture provides these QoS guarantees in the most scalable and least complex manner [2]. A diffserv domain provides different levels of service (e.g. premium, assured) to meet client's request for guarantees. Network resources are partitioned between these levels of service. Packets belonging to client flows are marked with specific code points (DSCP) that map these packets to a particular level of service depending on the service level agreement (SLA) between the client and the network service provider. The bandwidth broker manages resources in a diffserv domain. It tracks available network resources and classifies flows using service polices defined based on client requirements and service classes offered by the diffserv domain [16]. In the diffserv architecture, flows are allocated resources without any understanding of the nature of information being transmitted. As a result, the broker statically overallocates resources so as to meet guarantees made to the client. This over allocation wastes resources and causes future flow requests to be rejected. Furthermore, the broker does not consider the nature of the flow and may allocate resources for rogue flows that can exceed their allocations and hog resources, causing congestion, and severely affecting the QoS of conforming flows. Thus, while diffserv does provide a sense of resource allocation and QoS, it does not guarantee QoS or eliminate the possibility of congestion.

In this paper we present a *content aware bandwidth broker* (CABB) that provides adaptive brokering for networked multimedia applications to allocate network resources among responsive (TCP) and unresponsive (UDP) flows. It builds on the observation that multimedia applications are flexible with respect to network parameters such as packet loss, delay and jitter. For example, a multimedia application flow may be tolerant or intolerant to packet loss [5]. CABB exploits this flexibility of multimedia flows to network level parameters to adapt the flows based on the state of network resources to maintain some level of QoS despite of unfavorable network conditions. For example, when the application's demand for resources exceeds availability, rather than refuse allocation to the application, CABB may admit and maintain the flow at a reduced QoS until the required resources become available. This can be significant for time critical applications. Furthermore, in case of network congestion, CABB can adapt to the network state and reduce QoS rather than completely disrupting the flow. CABB also prevents non-confirming (or rogue) flows from affecting the performance of conforming flows by constantly monitoring and gradually degrading the level of service for the rogue flows. Thus it provides the incentive in support of end-to-end congestion control for best effort traffic.

CABB is implemented and evaluated using the NS-2 simulator toolkit. Results presented in this paper show that by exploiting flexible nature of multimedia flows, CABB improves network resource allocations. The results also show that

multimedia flows are better managed and controlled, thereby improving perceived QoS and avoiding possible congestion. Flow throughputs also increase as CABB enables more flows to be admitted.

2. Content-Aware Bandwidth Broker: Architecture, Polices, Operations

The content-aware bandwidth broker (CABB) is an intelligent agent that is configured with service provisioning policies and associated with a particular diffserv domain. It builds on the bandwidth broker architecture defined by the Qbone team [16]. In a diffserv environment, there are two defined per hop behaviors (PHB): expedited forwarding (EF), and assured forwarding (AF) [2]. EF PHB supports low loss, low delay, and low jitter and serviced by a priority queue [3]. AF PHB defines four relative classes of service with each class supporting three levels of drop precedence and serviced by weighted fair queuing (WFQ). The CABB consists of four components, viz. resource allocator, a database to store all the parameters required to make the reservation decisions, a policy engine to create a particular policy for EF, AF or best effort flows, and a broker manager for inter-broker communication. CABB uses the policy engine to configure the functionality of routers in a diffserv domain. Changes made to a particular policy are reflected in all edge routers in the domain. The policies for EF, AF and best effort services calls consider the nature of the request along with the flexibility of application with respect to network level parameters of delay, jitter, and loss to resolve resource requirements.

As described above networked multimedia applications are flexible in the sense that they can tolerate resource scarcity (within certain limitations) by reducing performance, and can utilize additional resources to improve performance. Furthermore, they can sacrifice the quality/performance of some parameters in order to preserve (or improve) the quality/performance of more critical parameters [1]. CABB maps this flexibility into variations network level parameters to maintain an acceptable QoS range for an application. When an application requests the CABB for network resources, it passes application specific QoS requirements and constraints. These parameters can include quality descriptions for the specific media characteristics (e.g. height, width, and color specification in a video stream), media sample rate, priority/criticality and transmission characteristics requirements for end-to-end delivery (e.g. end-to-end delay bounds). On receiving such a application flow request along with QoS parameters, CABB translates the parameters into network resource requirements such as rate usage profiles (i.e. chief and peak information rate), burst, delay, flexibility to delay, jitter, loss, and determines a time for which the created profile is to be active. CABB then tracks current allocation of marked traffic, and interprets the new requests in light of defined policies and available resources. It looks up its policy table for the existing policy that governs the requesting host and contains the relevant service level agreements (SLA), service/DSCP mappings, management information, current reservations/allocations, and edge router configurations. These parameters along with the application's flexibility [1, 6] are passed to the policy engines defined by the SLA. These policies further translate the parameters into specific network actions such as bandwidth management (allocated transmission rate), queuing (per hop behavior), buffer space for queuing, network monitoring and accounting. The policies determine the users' bandwidth requirements (after translating flexibility to map to minimum resource requirement [1]) and allocate resources accordingly. Resource allocation is made in a succinct and organized fashion treating all flows fairly. Furthermore, the allocation is based on the client's needs, as determined from its QoS parameters, rather than its peak request that may be larger. This not only conserves resources (which can be allocated to later requests and would otherwise be wasted) but also allows requests to served when requested resources are not fully available.

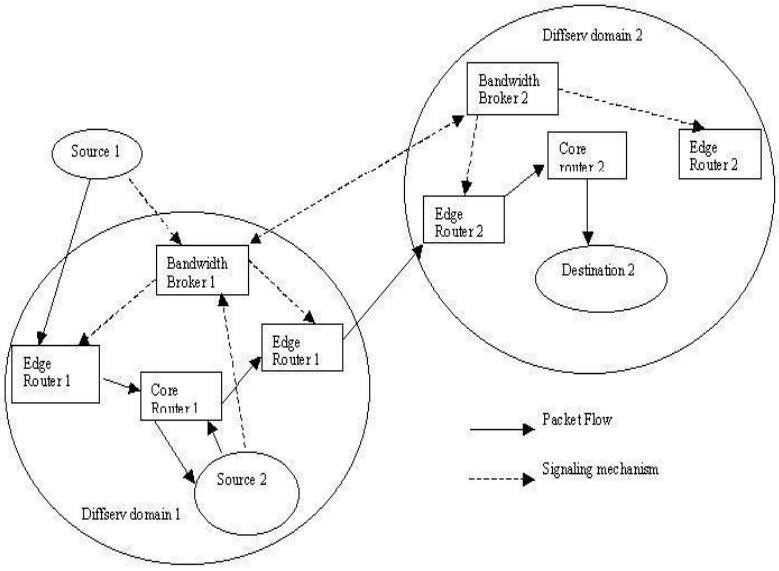


Figure 1: Test Network with Different Scenarios

This is possible because the CABB policy engines understand applications requirements and can effectively map them to available service levels to achieve better admission control. A flow is rejected only if resources remain insufficient even after reducing the level of service. For successful admission, the CABB makes appropriate reservations using the resource allocator (i.e. usable bandwidth, buffers) and assigns it a DSCP for that service. It then schedules the flow to a particular queue manager (Priority or Weighted Fair Queue). Diffserv's internal policing mechanisms then

force the flows to adhere to the agreed policy. Furthermore, with the help of the broker manager, CABB can generate additional request messages to reserve resources downstream [16]. Each downstream CABB takes into account the inter-domain traffic SLA before allocating resources. There is a limit on the number of flow reservations allowed per service class, which ensures good bandwidth utilization [2]. More information on CABB implementation can be found in [12]. The CABB may also maintain a log of a flow's history allowing it to identify (and curb) flows with a rogue nature (e.g. a flow that is continuously bursty after promising constant rate).

For example, consider Figure 1. Source 1 in diffserv domain 1 (DS1) requests a flow setup to destination 2 in diffserv domain 2 (DS2). It provides its requirements such as average and peak transmission rates, delay and jitter along with its type (e.g. multimedia audio) to CABB1 in DS1. CABB1 uses its policy engines to determine the required bandwidth and PHB based on the agreed upon SLA, the current available resources, and the flexibility of the application. The flexibility for various multimedia applications is encoded into the policy engine based on their tolerance to the above mentioned network level parameters. CABB1 passes the flow's parameters (as is or reduced) to downstream broker, CABB2 in DS2. On receiving a positive acknowledgement from CABB2, it assigns a DSCP to the traffic flow between this source-destination pair in its policy table. If available resources are insufficient or CABB2 returns a negative acknowledgement, CABB1 retries to set up of the flow by assigning a set of lower DSCPs, which define slightly lower requirements along with lower but still acceptable quality. It also informs the application of this adaptation. Source 2 in DS1 requests a flow (multimedia video) also to destination 2 and provides its parameters such as peak transmission rates, delay, jitter, packet loss and required resources. As before, CABB1 invokes its policy engines to determine the resources to be allocated. Note that the flexibility number will be different in this case - multimedia video is more flexible to packet loss than audio. It sets up the flow after receiving a positive acknowledgement from CABB2. In this example, source 1 is given EF PHB and source 2 is given AF PHB.

3. Experimental Evaluation

As mentioned earlier, CABB is implemented using the NS-2 simulator toolkit on top of Nortel Networks diffserv implementation. We evaluate CABB using three different network topologies with two sets of experiments in each case. We describe the first set of experiments below. The other two sets of experiments are summarized in Table 2 and will be described in detail in the final paper. The network topology is shown in Figure 2. The diffserv domain includes three routers and has one CABB that manages and configures the edge routers. The experiments consist of the three flows: (1) *S0-dest*: audio-EF PHB, (2) *S1-dest*: video – AF PHB, (3) *S2-dest*: video conference – AF PHB. *Core-E2*: is the bottleneck link with a 5 Mb capacity. *CABB* manages and configures routers *E1*, *E2* each with two queues, one priority and one WFQ. Following notations are used in the discussion below: Peak Information Rate (PIR), Committed Information Rate (CIR), Committed Burst Size (CBS), Packet Transmission Rate (RATE), Available Bandwidth (ABW), and Usable Bandwidth (UBW) which is the actual allocated bandwidth. Early drops follow the RED algorithm [5], while late drops occur when packet arrival exceeds buffer size. We define DSCP 10 for EF PHB and DSCP 21 for AF PHB. For non-conforming flows DSCP 10 can be downgraded (allocate reduced bandwidth) to DSCP 11. Similarly DSCP 21 is downgraded to DSCP 22. Simple policies for EF and AF flows are show in Table 1.

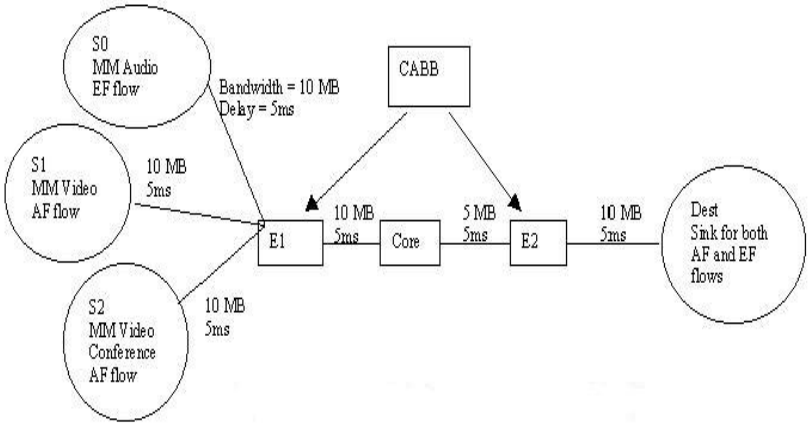


Figure 2: Topology (A mixture of one EF and two AF flows)

EF Policy	AF Policy
If ABW > PIR, UBW = f (PIR) If CIR < ABW < PIR, UBW = f (flexibility, CIR, PIR), If ABW < CIR || UBW retry with reduced parameters	If ABW > CIR, UBW = f (CIR, CBS) If ABW < CIR, UBW = f (flexibility, CIR, CBS). If ABW < UBW, retry with reduced parameters

Table 1: EF and AF Policies

The overall goal of this experiment was to demonstrate the operation of CABB and its ability to effectively manage resource allocation and limit the effect of unresponsive EF and/or AF flow. The two experiments for the topology show in Figure 2 are described below. The results are plotted in Figures 3 and 4. The legend for Figures 3 and 4 is as follows: cp – Codepoint; TotPkts – Total packets, TxPkts – Transmitted packets; edrops – early drops and ldrops – late drops at the core router (Core). Graphs 2 and 3 show the EF profile (CP 10 and 11). Graph 4 and 5 show the AF profile (CP21 and 22). The tables in these figures show transmission rates and parameters used by the CABB policy manager. These statistics were calculated for 80.0 time steps of the simulation.

Experiment 1: In this experiment, S0, S1 and S2 demand 4mb (PIR), 1mb (CIR), and 1mb (CIR) and transmit at 2.5mb, 1mb, and 1mb respectively. The results from this experiment are plotted in Figure 3. As can be seen from these plots (graphs 2 and 3), S0 is allocated reduced bandwidth (2 MB), i.e. its packets are downgraded to CP 11 since $Rate_{S0} > UBW_{S0}$. S1 and S2 are allocated their requested resources and are never downgraded as there are in-profile (see graphs 3 and 4). This shows that CABB does not allow the out-of-profile EF flow to downgrade or drop in-profile AF flow from S1 and S2.

Experiment 2: In this experiment, S0, S1, S2 demand 3mb, 1mb, and 1mb and transmit at 2.5mb, 2mb, and 1mb respectively. The results from this experiment are plotted in Figure 4. As can be seen from these plots, S0, S1 and S2 are allocated full resources. Graphs 2 and 3 show that S0 is in-profile and is never downgraded. S1 goes out of profile and the AF aggregate packets are downgraded to CP22 and are eventually dropped (see in graphs 4 and 5). As seen in graph 5, edrops are less than corresponding ldrops. Finally, we observe that the out of profile AF flow does not affect the in-profile EF aggregate.

Topology	Experiment	Observations and Results
Topology 2: Builds on Figure 2 but without source S2. *S1-dest* (video-EF PHB) Core-E2 bottleneck link: 7mb.	Observe CABB decisions regarding flow allocations and flow control for unresponsive EF flows. 1) S0, S1 demand 8mb and 2.6mb (PIR) and transmit at 8mb and 9mb respectively. 2) S0, S1 demand 1.5mb and 2mb (PIR) and transmit at 1.5mb and 1mb respectively.	**Result 1**: S0 was rejected after interbroker communication failed (congested link) for less flexible audio application. S1 was allocated full resources (PIR) but $Rate_{S1} >> UBW_{S1.}$ Thus one flow is rejected despite retrial and another allocated full resources but policed and forced to adhere to SLA. **Result 2**: S0 was allocated full resources. S1 was allocated reduced resources since video being more flexible than audio. We observed that CABB increased flow allocations with some flows getting reduced resources arrived at from their flexibility number.
Topology 3: Same as Fig 2 with the following changes. *S2-dest* (FTP on TCP-AF PHB). Core-E2 bottleneck link: 7mb. Three queues: priority and 2 WFQ	Observe CABB's admission and control for UDP and TCP flows and behavior between EF, AF aggregates. 1) S0, S1 and S2 demand 2.6mb, 1mb, and 3mb. S0 and S1 transmit at 2.6mb and 1mb. S2 produces packets at regular intervals. 2) S0, S1 and S2 demand 5mb, 7mb and 1mb respectively. S0 and S1 transmit at 2mb and 4mb respectively	**Result 1**: S0, S1 are allocated full resources while S2 has a reduced allocation. Out-of-profile S0 incurs packet drops while S1 is downgraded. In-profile S2 is unaffected by rogue S0. **Result 2**: S0 given reduced resources, S1 is rejected and S2 allocated full resources after interbroker communication and taking flexibility into account Out-of-profile S2 is downgraded while in-profile S1 is unaffected. Thus CABB allocated only those flows that would not load the network giving reduced resources to some. We see that TCP flow is not starved but is penalized for not conforming to SLA. Furthermore it does not affect in-profile audio (UDP).

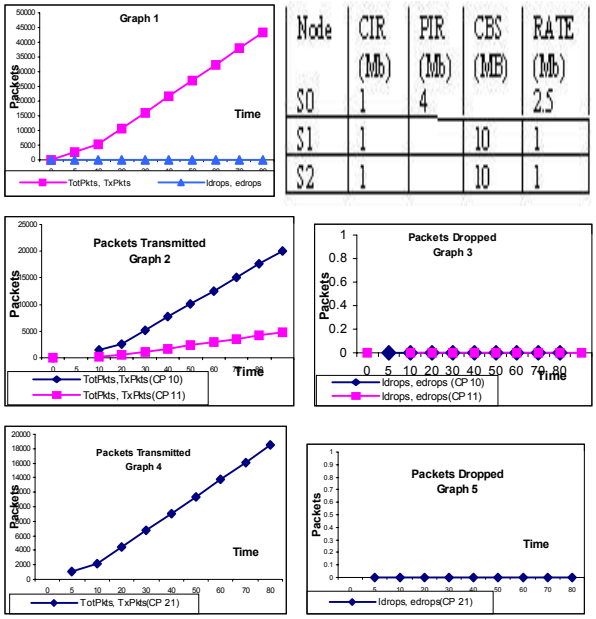
Table 2: Topologies 2 and 3 – Observation and Results

4. Summary of Results

In the CABB-based diffserv system, multimedia flows are analyzed, allocated resources and regulated before being allowed to use the network. CABB effectively orders flows so that they confirm to the traffic profile as agreed in the original SLA or the reduced profile arrived at using the applications's flexibility number. Furthermore, queue management and diffserv policing work such that when a flow goes out-of-profile its packets are downgraded and eventually dropped thus regulating a flow. EF flows serviced by priority queues are allowed to go through with minimal drop. Among the flows, the downgraded DSCP faces harsher penalty as compared to the initially allocated one. The out-of-profile AF flows

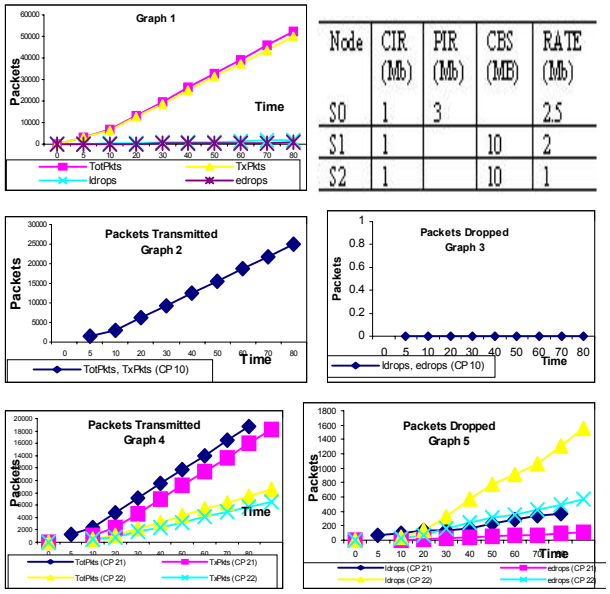
do not affect the throughput of EF flows and vice versa. Multimedia applications, due to their flow requirements, are usually assigned to EF flows. However those flows that can tolerate losses may be assigned to AF flows due to adaptation by the broker. Hence, these flows go through a congested network in some form or other, with the CABB doing coarse tuning of bandwidth requirements (avoiding over allocation) and the application level adaptive QoS doing the fine-tuning of the applications' response/sensitivity to network changes [9]. This is done such that the end user perceives a quantifiable QoS even with reduced resource allocation. CABB ensures higher flow throughput by identifying and controlling rogue flows. Furthermore, CABB eliminates the need for applications to have to deal directly with the diffserv router for resources [7]. Both TCP and UDP flows get a fair share of the network. TCP flows are not starved by rogue UDP flows and at the same time are regulated to confirm to SLA.

The normal broker does not account for application's adaptability leading to overallocation of resources. It allocates flows based on available resources and hence cannot ensure that a rogue flow will not affect flows belonging to same or other service classes. Consequently, when compared to a normal broker the CABB efficiently utilizes the network by allowing only those flows that do not congest it.



Node	CIR (Mb)	PIR (Mb)	CBS (MB)	RATE (Mb)
S0	1	4		2.5
S1	1		10	1
S2	1		10	1

Figure 3: Out-of-profile S0 gets downgraded not affecting in-profile S1 and S2



Node	CIR (Mb)	PIR (Mb)	CBS (MB)	RATE (Mb)
S0	1	3		2.5
S1	1		10	2
S2	1		10	1

Figure 4 S1 goes out-of-profile causing AF aggregate to downgrade

5. Conclusions

A reservation-based environment requires end-to-end multi-resource reservation plans. In this paper we discussed the CABB architecture for managing the QoS of multimedia applications in a diffserv environment. CABB determines the policy for a particular flow based on the end user service level agreement, the flow's characteristics (flexibility), network resource availability and interbroker communication, to provide application level end-to-end QoS under (un)favorable network states. Experimental results show that CABB's policy decisions prevent congestion in the downstream network. These policy decisions are simple, unbiased and effective to allow enough concurrent flows such that the network is not unduly loaded and hence controls traffic at the edge router keeping the core simple. This design is easily scalable since no state is maintained in routers. Multimedia applications or those that use UDP for data transmission are now coarsely controlled by the broker's policy decision. The broker considers the applications demands before allocating resources ensuring that the flow will confirm to the profile else be downgraded. Thus it intelligently allows flows to utilize resources and maximizes the flow throughput without congestion. This work can be extended further. Our next step is to enhance the CABB's intelligence with a better understanding of the distribution profile of the usage of multimedia applications or flows and/or users and also to make the policy decisions more dynamic.

References

1) B. Li, D. Xu, K. Nahrstedt, J. W. S. Liu, "End-to-End QoS support for Adaptive Applications Over the Internet," SPIE International Symposium on Voice, Video and Data Communications, pp 147 – 161, November 1998.
2) K. Nichols, V. Jacobson, L. Zhang, "A Two-bit Differentiated Services Architecture for the Internet," RFC 2638, July 1999, see http://www.cs.wisc.edu/~cs640-1/papers/jacobson.qos.ps
3) T. Bonald, A. Proutière, J. W. Roberts, "Statistical Performance Guarantees for Streaming Flows using Expedited Forwarding," INFOCOM 2001, 26-28, Tallinn, Estonia, April, 2001.
4) X. Xiao, L. M. Ni, "Internet QoS: The Big Picture," IEEE Network Magazine, March/April, pp. 8-18, 1999.
5) L. Peterson, B. S. Davie, "Computer Networks - A Systems Approach," pp. 446-514, 2nd edition, 2000, Morgan Kaufmann Publishers.
6) D. D. Clark, S. Shenker, and L. Zhang, "Supporting Real-Time Applications in an Integrated Services Packet Network: Architecture and Mechanisms," Proc. ACM Sigcomm 92, pp. 14-26, ACM Press, New York, Aug. 1992.
7) J. Shin, J. W. Kim, and C. C Jay Kuo, "Content-Based Packet Video Forwarding Mechanism in Differentiated Services Networks," Proceedings of the Packet Video Workshop'2000, May 2000
8) P. Chandra, A. L. Fisher, C. Kosak and P. Steenkiste, "Network Support for Application-oriented QoS," pp. 187-195, Sixth International Workshop on Quality of Service (IWQoS' 98).
9) N. Shaha, A. Desai, M. Parashar, "Multimedia Content Adaptation for QoS Management over Heterogeneous Networks," to appear in International Conference on Internet Computing 2001, Las Vegas, USA
10) L. W. I. Andrikopoulos and G. Pavlou, "A fair traffic conditioner for the assured service in a differentiated service internet," in *Proceedings of IEEE International Conference on Communications (ICC 2000)*, New Orleans, LA, June 2000.
11) Implementation of the Two Tier Differentiated Services Architecture, Computer Science, University of California, Los Angeles, see http://irl.cs.ucla.edu/twotier/
12) M. Mahajan, M. Parashar, "Content Aware Bandwidth Broker," Technical Report, Department of Electrical And Computer Engineering, Rutgers University, December 2001 (Available at http://www.caip.rutgers.edu/~manishm/bandwidthbroker/CABB.PDF)
13) A. Ramanathan, M. Parashar, "Active Resource Management for The Differentiated Services Environment," Third Annual International Workshop on Active Middleware Services, Network Services Session, San Francisco, CA, August, 2001
14) D. Xu, K. Nahrstedt, A. Viswanathan, D. Wichadakul, "QoS and Contention-Aware Multi-Resource Reservation," In Proceedings of the 9th IEEE International Symposium on High Performance Distributed Computing (HPDC-9), Pittsburgh, PA, August 2000.
15) IETF "Differentiated Services" Working Group, see http://www.ietf.org/ids.by.wg/diffserv.html
16) QBone, Simple Inter-domain Bandwidth Broker Signaling, Internet 2, see http://qbone.internet2.edu/bb/
17) The Network Simulator - NS-2, see http://ww.isi.edu/nsnam/ns/
18) Information and Telecommunication Technology Center (ITTC) in IP QoS research, see http://qos.ittc.ukans.edu
19) Resource ReSerVation Protocol (RSVP), RFC 2205, Sep 1997.
20) Multiprotocol Label Switching Architecture (MPLS), RFC 3031, Jan 2001