An Intelligent Agent Based QoS Provisioning and Network Management System

ANGELOS MICHALAS[1], MALAMATI LOUTA[2], GEORGE KOUZAS[1]
[1]School of Electrical and Computer Engineering
National Technical University of Athens
9 Heroon Polytechneiou, Zografou, Athens, 15773
GREECE
michalas@central.ntua.gr
[2]Department of Business Administration
Technological Educational Institute of Western Macedonia
Koila, Kozani, 50100
GREECE
louta@telecom.ntua.gr

Abstract: - The DiffServ architecture provides a scalable mechanism for QoS introduction in a TCP/IP network. DiffServ model is based on the aggregation of traffic flows at an ingress (or egress) point of a network and the IP packet marking for different priority flows, according to several classification criteria. Two approaches exist in the DiffServ architecture: the *Absolute* and the *Relative*. In Absolute DiffServ, an admission control scheme is used to provide QoS guarantees as absolute bounds of specific QoS parameters. The Relative DiffServ model provides QoS guarantees per service class expressed with reference to guarantees given to other classes defined. Our study presents a QoS Provisioning & Network Management System. This system is based on an extension of the network management architecture followed and implemented within the scope of the European IST Project MANTRIP. MANTRIP network management system supports quality of service configuration and monitoring in IP networks. Our extension provides QoS Differentiation (Absolute and Relative) in IP DiffServ based Networks. The proposed system has been applied and performed well on a real network testbed.

Key-Words: - Quality of Service, Differentiated Services, Absolute and Proportional Relative Differentiation, Intelligent Mobile Agents.

1 Introduction

The research community has concentrated on two different techniques to provide QoS differentiation to customers of packet switched networks. First, the Integrated Services (*IntServ*) [1] and, second, the Differentiated Services (*DiffServ*) [2] approach. The major difference between *IntServ* and *DiffServ* architecture is the granularity of service differentiation. The *IntServ* concept lies on resource reservation notion per application flow, while in *DiffServ* model, IP traffic is classified into finite, predefined service classes (on the basis of demand requirements and characteristics) that receive different routing treatment. *DiffServ* provides quality assurances at traffic aggregate level and not at application flow level. This way, *DiffServ* achieves scalability and manageability, while on the other hand, *IntServ* approach faces potential bottleneck problems, since all routers must maintain information per flow state. There exist two directions in the *DiffServ* architecture: the *Absolute* and the *Relative*.

In *Absolute DiffServ* [3], an admission control scheme [4] is used to provide QoS guarantees as absolute bounds of specific parameters such as bandwidth, packet delay, packet loss rate, or packet delay variation (*jitter*). The *Relative DiffServ* model [5] supports QoS guarantees per service class with reference to guarantees given to other classes. The only assurance coming from the network is that high priority classes receive better (or at least not worse) service treatment than lower priority classes.

In the context of this paper, *Absolute* and *Relative Diffserv* Provisioning and Management are achieved through a distributed QoS System. This system is based on an extension of the network management architecture followed and implemented within the scope of the European IST (Information Society Technology) Project *MANTRIP* (*MANagement, Testing and Reconfiguration of IP based networks using Mobile Software Agents*) [6]. MANTRIP network management system supports quality of service configuration and monitoring in IP networks. Our extension provides QoS Differentiation (Absolute and Relative) in IP DiffServ based Networks.

The implementation of the QoS Provisioning & Management system is based on Intelligent Mobile Agent Technology (*MAT*), which has been considered as a paradigm that can help service designers handle the potentially increased functional complexity involved in service creation and deployment [7]. The main objective of our system is to provide and manage Quality of Service in IP DiffServ based Networks [8], with Mobile Agents (*MAs*) implementing specified Application Interfaces (*APIs*) for Absolute and Relative Differentiation. MAT has already been used for the implementation of network, resource and telecommunication management services [9]. Adoption of *MAs* as our basic architectural design and implementation technology allowed for decentralization of the QoS configuration and monitoring tasks. Additionally, *MAT* promoted good software design, added flexibility, manageability and scalability with a relatively easy implementation of such a complicated system. Furthermore, *MAs* allowed for the implementation of various Absolute and Relative Diffserv approaches without introducing any modifications to the main system architecture.

The rest of the paper is organized as follows. In Section 2, the *Absolute and Relative Diffserv* schemes are briefly revisited. Section 3, as a first step, provides the high level architectural description of the proposed QoS Provisioning and Management System and, as a second step, elaborates on its functional procedures. In Section 4 a set of results indicative of the proposed system operational efficiency on a real network testbed are given. Finally concluding remarks are made and issues for future study are provided.

2 DiffServ Architectural Schemes

The main strength of *DiffServ* model is quality assurance provision at traffic aggregate level, thus allowing for IP traffic classification into a finite number of service classes that receive different routing treatment. Specifically, routers at the network edges classify packets into predefined service classes based on demand requirements and characteristics of the associated application. Core routers forward each packet according to a class based scheduling policy. This way, the model provides service differentiation on each node for large aggregates of network traffic.

In *Absolute DiffServ* architecture, an admission control scheme is used to provide QoS guarantees as absolute bounds of specific parameters such as bandwidth, packet transfer delay, packet loss rate, or packet delay variation (*jitter*). A connection request is rejected if sufficient resources are not available in the network, so as to provide the desirable assurances. In case of acceptance, the appropriate resources are reserved, while monitoring procedures assure end to end performance of the specific connection. There are two basic approaches to admission control [10]. The first, which is called *Parameter-based approach*, computes the amount of network resources required to support a set of flows, given a priori flow characteristics. A key difficulty encountered in most parameter based approaches is their requirement for reserving resources and maintaining state information per traffic flow on each network node (e.g., traffic parameters, QoS class). The second, *measurement-based approach*, relies on measurement of actual traffic load in order to make admission decisions. Considering measurement-based admission control techniques based on probing, the traffic source or the ingress router probes the network by sending probe packets at the data rate it would like to reserve and recording the resulting level of packet delay or losses. The flow is accepted only if packet loss or delay is below a predefined threshold value.

In *Relative Diffserv* architecture, the QoS parameter values of a specific connection depend on the network load, since no admission control or resource reservation mechanism exists. Proposals for *Relative DiffServ* QoS define service differentiation qualitatively [11][12], in terms that higher classes receive lower delays and losses from lower classes. Recent research studies proposed a qualitative relative differentiation scheme, named *Proportional DiffServ* [5], which controls the ratios of delays or loss rates of successive priority classes in order to be constant. According to this scheme, given two consecutive priority classes, it can be guaranteed that the packet delays or the loss rate for the higher priority class can be a specified portion of the packet delays or the loss rate of the lower priority class.

Considering the *Proportional Delay Differentiation* (P*DD*) model [5], the ratios of packet delays of successive priority classes are equal to the ratio of their corresponding Delay Differentiation Parameters (*DDPs*) $\{\delta_i, i = 1,...N\}$. Thus, assuming that we have *N* classes of service and the average queueing delay of class-*i* packets is d_i , the ratio of average delays between two service classes *i*, *j* is fixed to the ratio of their corresponding *DDPs*:

$$\frac{d_i}{d_j} = \frac{\delta_i}{\delta_j} \qquad 1 \le i,\ j \le N \qquad (1)$$

In most research efforts carried out, proportional delay differentiation is achieved by employing *Priority Based* or *Link Sharing Schedulers* [13].

3 QoS Management System

3.1 High Level Architecture

The *QoS Provisioning and Network Management* Architecture is organized in four different layers. The ***Application Layer*** comprises the logic for performing the QoS provisioning and management tasks. Additionally, it includes the Graphical User Interfaces (*GUI*) developed in order to provide to the users access to the system in a quite friendly manner. The ***Service Layer*** contains the services that support the execution of the application. The ***Adaptation Layer*** is responsible for hiding the protocol details from the Service Layer and includes the network adapters and wrappers. The ***Network Layer*** includes the network resources. All network nodes support IP DiffServ, which is implemented by a *Class Based Queue* (*CBQ*) scheduler [13] on each network node interface.

The *QoS Provisioning and Network Management System* includes the following components and subsystems:

The ***QoS Provisioning & Management Application*** offers a *GUI* to both network administrators and users. The user is authorized for requesting a uni or bi-directional connection reservation across two *Service Access Points* (*SAPs*) by choosing a certain QoS class of service and giving certain value for the bandwidth required, less than or equal to the maximum bandwidth available for the specific class of service.

The ***QoS Connectivity Provisioning & Management Subsystem*** offers layer generic IP connectivity management capabilities to the QoS application. The Connectivity Provisioning & Management subsystem implements the *Connectivity Agents* software module as mobile agents that are sent as close as possible to the routers to perform the requested connectivity tasks.

The ***Monitoring Management Subsystem*** offers to both users and the administrator the capability to monitor the QoS parameters of the configured connections. The Monitoring Management subsystem exploits Active and Passive monitoring techniques [14] for obtaining delay/jitter and bandwidth utilisation/loss statistics, respectively. This subsystem implements the *Monitoring Agents* software module as mobile agents that are sent as close as possible to the routers to perform monitoring tasks.

The ***(Re)Configuration Provisioning & Management Subsystem*** allows the administrator to (re)configure certain QoS parameters of the network elements. On one hand, it caters for the initial configuration of the routers (i.e., define parameters for the service classes of the DiffServ model) and on the other hand, if a certain path flow is violating the thresholds defined by the QoS class of service parameters, the administrator may trace the route of the specific connection (if static routing is being used), get the queue load of the routers involved and reconfigure them in order to improve the end-to-end QoS. In essence, this subsystem allows for the configuration and reconfiguration of the queue size per class of service and of the maximum bandwidth allocated to each service class. This subsystem implements the *(Re)Configuration Agents* software module as mobile agents that are sent as close as possible to the routers to perform (re)configuration tasks. At this point it should be noted that the *(Re)configuration Agents* may estimate new bandwidth values taking into account both the queue load per class, and the required delay spacing among classes defined by the user.

Information concerning the network topology, the QoS class of service templates, the routers and their configuration parameter values (bandwidth given to each service class, queue size), the users and their accepted connections configurations (e.g., bandwidth) are stored in the system database. Concerning the implementation issues of the *QoS Provisioning and Network Management System*, all the subsystems have been implemented in Java in the context of [7]. Finally, *Connectivity Agents*, *Monitoring Agents* and *(Re)Configuration Agents* have been implemented as intelligent mobile agents based on the use of Voyager platform [15].

3.2 QoS Management System Functionality

QoS Provisioning & Management System caters for the initial configuration of the network routers (i.e., definition of the QoS parameters of the DiffServ service classes on the basis of past experience/ historical data) by means of the *Configuration Agents* sent from the *(Re)Configuration Provisioning & Management subsystem* (implemented on the management station) as close as possible to the routers in order to perform their mission.

Considering a connection reservation request for Absolute QoS across two *SAPs* issued by the user, as a first step, the QoS Provisioning & Management System performs Call Admission Control. In case creation of the connection is authorized (i.e., there are available network resources to serve the requested connection), the system proceeds with the establishment of the connection by configuring the routers associated with the specific path across the two *SAPs*. For parameter based approach, connection admission is based on the available bandwidth per link across the path from the source *SAP* to the destination *SAP*. Measurement based admission control is implemented using the *Monitoring Management subsystem*. To be more specific, as a first step, the connection is created, but is not set into active mode. As a second step, the *Monitoring Management subsystem* performs active and passive monitoring for this connection and according to the packet delay or packet loss experienced the connection is accepted or otherwise rejected.

In case the user requests Relative QoS (e.g., Proportional Delay Differentiation), the system does not perform admission control. Instead, *Reconfiguration Agents* are initiated by the administrator and sent on each router's outgoing network interface to estimate and allocate new bandwidth values (service rates) to service classes of the *CBQ* schedulers.

Taking into account the fact that in Absolute DiffServ most admission control schemes [16] consider average traffic arrival rate, in conjunction with the non static in general source's behaviour, congestion is likely to emerge particularly on the core network routers. Considering the case of higher service class overloading (e.g., such may be the case of simultaneous activation of many applications of a high rated service class), worse packet forwarding may emerge with respect to the lower service classes. In such a case, service rate reconfiguration of routers output link is required in order to provide the best QoS possible per flow. Consequently, assuming that users and applications cannot get the requested absolute service level assurance, such as an end-to-end delay bound or throughput due to network resources insufficiency, a consistent service differentiation on output links of core routers should be provided, so that most of the QoS required levels are satisfied. Similarly, Relative DiffServ could be achieved by reconfiguring the service rate of each class according to its packet arrival rate and buffer occupancy. Authors' previous work on these issues include [17][18][19][20][21].

In our system, the *Reconfiguration Provisioning & Management subsystem* is responsible for preserving dynamically the specified end-to-end Absolute Delay Constraints (*ADCs*) or/and Relative Delay Constraints (*RDCs*) for established connections. Specifically, a *Reconfiguration Agent* created by the *Reconfiguration Management subsystem* is sent to the Adaptation Layer of each core router of the Diffserv network. The *Reconfiguration Agent* retrieves regularly the number of packet arrivals $a_i(t)$, the number of packet departures $dp_i(t)$, and the number of packet drops $dr_i(t)$ per class of service. Thereafter, it finds the current router queue load: $q_i(t) = a_i(t) - dp_i(t) - dr_i(t)$ and estimates the current delay per class of service: $d_i(t) = \frac{q_i(t)}{r_i(t)}$, where $r_i(t)$ is the current service rate of class-*i*. The results are sent to the *Reconfiguration Manager*, which, in accordance to the $d_i(t)$ values, can predict if an absolute or relative delay constraint violation on a specific connection exists. In case of an *ADC* or *RDC* violation, it issues a reconfiguration request to the *Reconfiguration Agents* in order to dynamically reconfigure the service rates per class of service on each node involved in the connection.

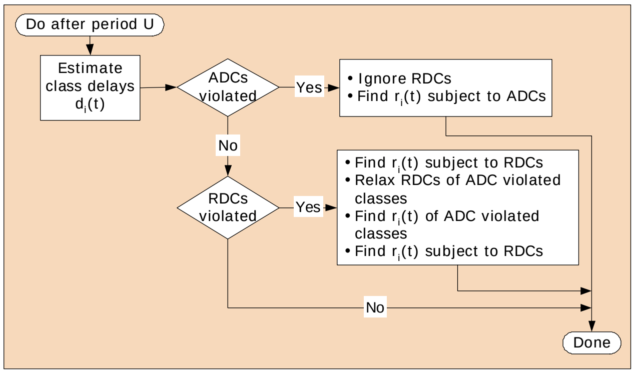


Figure 1. Service Rate Computation Algorithm for absolute and relative delay differentiation provision.

The service rate computation algorithm applied for absolute and relative delay differentiation is depicted in *Figure 1*. Specifically, in case of an *ADC* violation, the algorithm relaxes *RDCs* and tries to satisfy only the strict *ADCs*. To this respect, as a first step, the available resources of service classes are utilized in order to succeed in satisfying the *ADCs* of all service classes. The process is iterated until an *ADC* violation does not exist and all available resources have been redistributed. If after the completion of the reassignment process there are still classes that violate their *ADCs*, the strict *ADCs* of the lower priority service classes are relaxed in favour of the higher order classes. Thus, the service rates of the higher order classes may still be increased by reducing accordingly the rates of the lower priority classes, until a minimum predefined service rate has been reached. It should be noted that, in such a case, the *ADCs* of the lower service classes are violated. In case of an *RDC* violation, the service rates per class of service are re-estimated in accordance to (1). In the new solution violates an *ADC* defined for a service class, its corresponding *RDC* is relaxed.

4 Results

In this section, an indicative set of results regarding the proposed system operational efficiency on a real network testbed is given.

Figure 2 illustrates the network topology adopted in the context of our experiments. The QoS System provides the means for the initial configuration of both Linux and Cisco DiffServ routers according to the QoS parameters defined by the user, for monitoring the respective parameters and for reconfiguring the routers if and whenever needed.

The testbed considered consists of three routers: two Linux routers, one playing the role of an edge router and the other used as core router, and one Cisco router, used as edge router. Two Users, *User A* and *User B* are connected to each of the edge routers. Specifically, *User A* is connected to the Linux edge router and *User B* to the Cisco edge router. The administrator uses a different machine (management station), which comprises the *Application Layer* and the *Service Layer* related subsystems of the QoS Provisioning and Network Management system. However, these subsystems can be located at different machines.

The capacity of all the links in the network is taken equal to 10 Mbps. Users are connected to the network through links of 100 Mbps. Two service classes are considered in the CBQ scheduler of the routers: *Expedited Forwarding* (*EF-Class 1*) and *Assured Forwarding* (*AF-Class 2*) service classes, with bandwidth allocations 75% and 25% of the router's output link bandwidth, respectively. The queue size of each service class is taken equal to 20 packets. It is assumed that sharing between the two service classes is allowed.

SAP1 is the service access point to the network for *Traffic Generator* host and *SAP2* is the service access point to the network for *User B* host. Two QoS connections, *CON1* of *EF* class and *CON2* of *AF* class are configured both with origination the *SAP1* and destination the *SAP2* and with requested bandwidth equal to the available bandwidth of each class. Two sources generate *Constant Bit Rate* (*CBR*) *UDP* traffic through the *Traffic Generator* host to each configured connection. The packet length is taken equal to 1 Kbyte for both service classes.

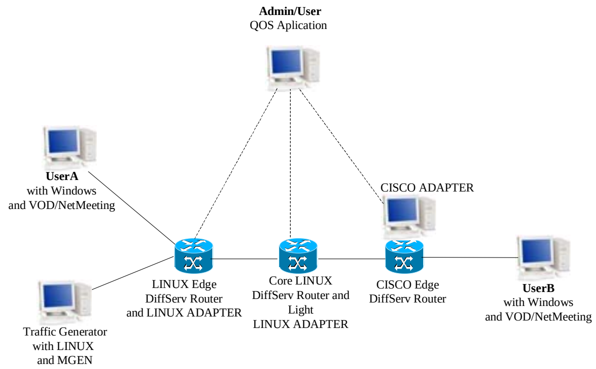


Figure 2. DiffServ Network Environment

We have conducted three different experiments. In the first experiment, we have not used any type of reconfiguration. In the second experiment, we have considered an *ADC* for the *EF* service class. Specifically, an absolute delay bound of 40 sec for packets of the *EF* service class has been posed. In the last experiment we have considered relative delay differentiation between *EF* and *AF* service classes. Relative delay factor is set equal to 2. The sources generate input traffic as shown in *Figure 3* for all the experiments considered.

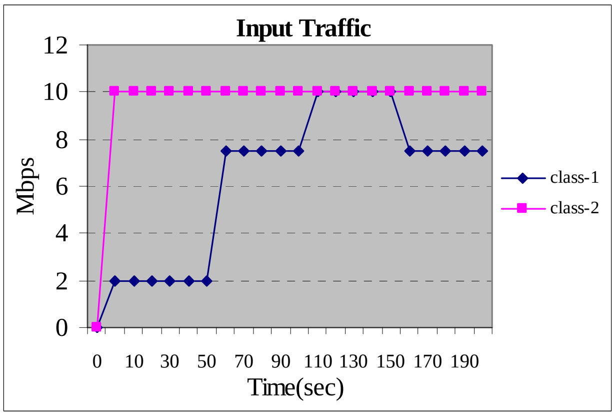


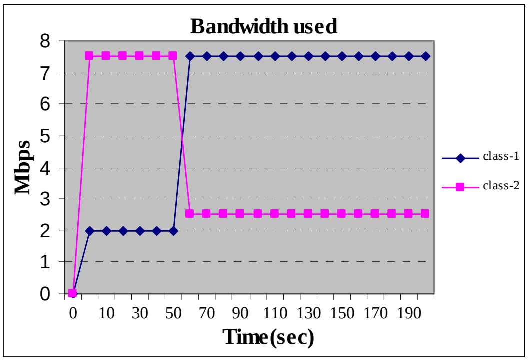
Figure 3. Offered Traffic

4.1 First Experiment

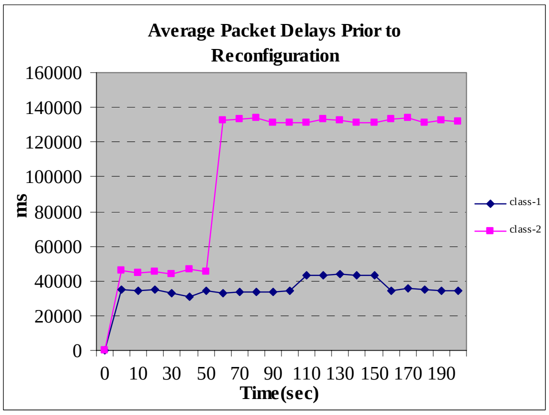
In the first experiment, reconfiguration is performed neither at the edge nor at the core routers of the DiffServ network. During the first 50 sec (*Figure 3*), the source of *EF* class transmits 2 Mbps traffic load and the source of *AF* class 10 Mbps.

The sharing of the *CBQ* mechanism at the network routers is exploited, since *EF* class does not utilize

fully its allocated bandwidth, whereas, the *AF* utilises more bandwidth than that allocated to it. As illustrated in *Figure 4(a)*, the throughput for *EF* service class is 2 Mbps (equal to the offered traffic) and *AF* service class borrows bandwidth portion from *EF* (5.5 Mbps), and has throughput of 8 Mbps.
During the next 50 seconds, from 50 to 100 sec, the source of *EF* service class changes its transmission rate to 7.5 Mbps, thus, *EF* class requires all its allocated bandwidth and sharing is no longer feasible. The throughput of *EF* and *AF* service classes become 7.5 Mbps and 2.5 Mbps, respectively. From 100 to 150 sec the *EF* source increases its rate even more to 10 Mbps. In this case, the bandwidth allocation to the two service classes does not change, since both sources utilize their allocated bandwidth. Hence, they continue to have the same throughput as before. However, during this time period, the *EF* class experiences packet losses. In the last 50 seconds, from 150 to 200 sec, the source of *EF* class decreases its rate to 7.5 Mbps. As expected, the throughput for each of the sources remains the same, equal to the bandwidth allocated to each service class. However, during this time period, the *EF* class stops having dropped packets.



(a)



(b)

Figure 4. The Reconfiguration scheme is not applied (a) Bandwidth used and (b) Average Packet Delays estimated regularly (10 sec time interval is considered).

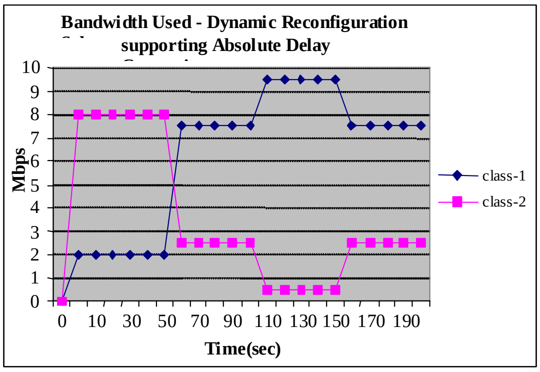
Figure 4(b) depicts the end-to-end average packet delay computed within time frames of 10 sec throughout the experiment. As it may be observed, packets of *EF* service class experienced delays above 40 sec during the time interval from 100 to 150 seconds, since the bandwidth allocated to *EF* is not enough to satisfy all the generated traffic. Also packet delays of *AF* class reach high values, close to 140 sec, when *AF* class becomes congested (50-200 sec).

4.2 Second Experiment

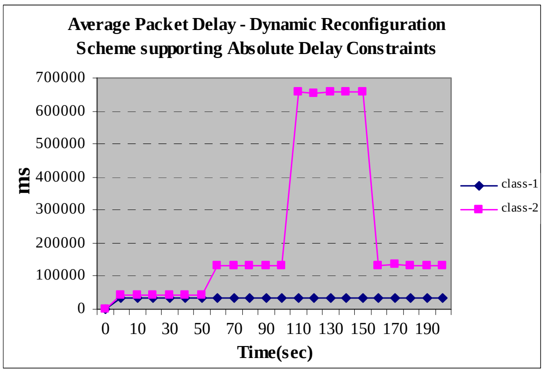
In the second experiment, dynamic reconfiguration is conducted in order to support absolute delay differentiation for packets of the *EF* service class. The *Reconfiguration Agent* located at each router estimates the current *EF* and *AF* packets delays experienced locally and readjusts their service rate when the *Reconfiguration Manager* detects that the *EF* absolute delay constraint is violated. The time interval between successive delay estimations is set equal to 1 sec. The absolute delay constraint for the *EF* classified packets on each router (local delay) was set to 10 sec (in order for end-to-end packet delays experienced to be under 40 sec). The minimum service rate of *AF* class was set equal to 0.5 Mbps.
Figure 5 illustrates in a graphical manner the bandwidth utilized and the average packet delay experienced for each service class. The difference with the first experiment due to the application of the dynamic reconfiguration scheme supporting *ADC* can be observed in the time interval from 100 to 150 sec. To be more specific, when no reconfiguration is performed, it can be observed (*Figure 3* and *Figure 4(a)*) that during this time period packets from the source of the *EF* class are transmitted with a higher rate than the allocated to this service class bandwidth. As a result, end to end delay experienced by *EF* service class packets (depicted in *Figure 4(b)*) exceeds the absolute delay constraint posed (40 sec).
However, with the application of the dynamic reconfiguration scheme, the *Reconfiguration Agent* estimates at t=100 that the delay threshold for *EF* packets is about to be exceeded. Since there is not any delay constraint for class *AF*, the *Reconfiguration Agent* assigns portion of the *AF* service class bandwidth to the *EF* service class, so as to satisfy the *ADC* constraint of the later. The bandwidth allocated to *EF* service class reaches 9.5 Mbps (*Figure 5(a)*) during this time interval. This value is dictated by the limit set to *AF* service class by the service rate reconfiguration algorithm applied in order to protect it from resource starvation. The

bandwidth reallocation process decongests *EF* service class as its excess traffic is effectively handled and its end to end packet delay falls below the 40 sec (as depicted in *Figure 5(b)*), which is the required end to end delay threshold.

In the following 50 seconds, from 150 to 200 seconds the input rate of *EF* service class is reduced and the algorithm gradually sets the bandwidth allocated to the *EF* service class equal to 7.5 Mbps, while the 2.5 Mbps are made available to the *AF* service class.



(a)

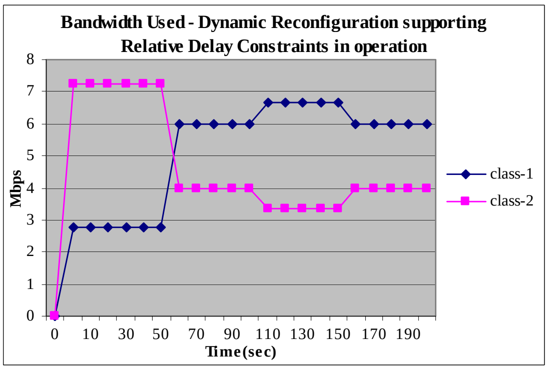


(b)

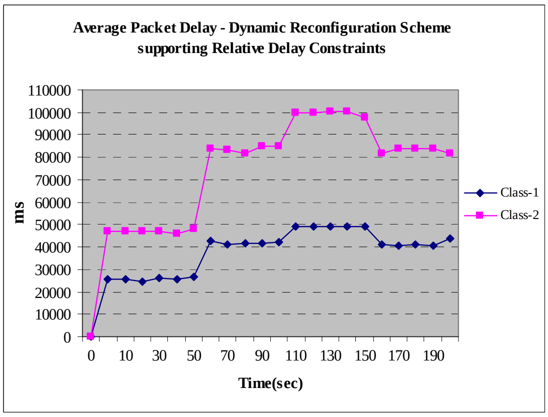
Figure 5. Dynamic reconfiguration scheme supporting Absolute Delay Constraints (a) Bandwidth used and (b) Average Packet Delays estimated regularly (10 sec time interval is considered)

4.3 Third Experiment

In the third experiment, dynamic reconfiguration is used in order to support relative delay differentiation between *EF* and *AF* service classes. The relative delay factor is set equal to 2. The *Reconfiguration Agent* located at each router estimates the delays experienced by *EF* and *AF* service class packets and readjusts the bandwidth in case it is required, so that the relative delay spacing locally at each router between the two service classes is satisfied. The time interval between successive delay estimations is taken equal to 1 sec. *Figure 6* depicts the bandwidth used and the end-to-end average packet delays experienced for the two service classes within time intervals of 10 sec. As illustrated in *Figure 6*, the bandwidth allocated to each class is adjusted appropriately with respect to the traffic load generated, so that the required relative delay differentiation between classes is achieved. It may be observed that the reconfiguration scheme achieves the relative differentiation in a quite accurate manner with only slight deviations of approximately 0.3%. These small deviations appear in cases where the traffic load of the high priority class (EF) is less than its allocated capacity.



(a)



(b)

Figure 6. Dynamic reconfiguration supporting Relative Delay Constraints (a) Bandwidth used and (b) Average Packet Delays estimated regularly (10 sec time interval is considered)

5 Conclusions

In the context of this paper, Absolute and Relative Diffserv provisioning and management in packet switched networks are achieved through a distributed QoS System. The implementation of the proposed QoS provisioning & management system is based on Intelligent Mobile Agent Technology (*MAT*), which added flexibility, scalability, manageability to our system, while at the same time allowed for a relatively easy implementation supporting even various configurations without introducing major modifications to the main architectural design.

As a first step, after briefly revisiting the Absolute and Relative DiffServ models, the high level architecture of the QoS Provisioning and Network Management System was presented. Thereafter, the authors elaborated on its operational procedures. Specifically, this paper presented the procedures followed for Absolute and Relative QoS Differentiation provision and management.

A set of results indicative of the effectiveness and efficiency of the QoS system performance in a real network testbed was given. Three experiments were conducted. The first one did not consider any reconfiguration of the router's output link, while the second and the third entailed application of the proposed Reconfiguration Scheme for Absolute and Relative DiffServ provision, respectively. It was shown that our system succeeds in satisfying the ADC and/or RDC related constraints posed each time. Directions for future work include but are not limited to the realization of further wide scale trials so as to experiment with the applicability of the framework presented herewith.

References:

[1] P.P White, "RSVP and Integrated Services in the Internet: A Tutorial," IEEE Comm. Mag., pp. 100-106, May 1997.

[2] K. Nichols, S. Blake, F. Baker, and D. Black, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers," RFC 2474, Dec. 1998.

[3] S. Blake, et al., "An Architecture for Differentiated Services," RFC 2475, Dec. 1998.

[4] I. Stoika and H. Zhang, "LIRA: An approach for Service Differentiation in the Internet," IProc. NOSS-DAV, 1998.

[5] C. Dovrolis, D. Stiliadis, and P. Ramanathan, "Proportional Differentiated Services: Delay Differentiation and Packet Scheduling," IEEE/ACM Transactions in Networking, Feb. 2002.

[6] MANTRIP Del. D2.1: System Requirements and High Level Architecture.

[7] M. Breugst and T. Magedanz, "Mobile Agents—Enabling Technology for Active Intelligent Network Implementation," IEEE Network, vol. 12, no. 3, Aug. 1998.

[8] B Teitelbaum, " QBone Architecture (v1.0)," Internet2 QoS Working Group Draft, Aug. 1999.

[9] W. Caripe, G. Cybenko, K. Moizumi, and R. Gray, "Network Awareness and Mobile Agent Systems," IEEE Comm. Mag., July 1998.

[10] S. Jamin, S. Shenker, and P. B. Danzig, "Comparison of Measurement-based Admission Control Algorithms for Controlled-Load Service," Proc. INFOCOM '97, April 1997.

[11] A.M. Odlyzko, "Paris Metro Pricing: The Minimalist Differentiated Services Solution," Proc. of IEEE/IFIP International Workshop on Quality of Service, June 1999.

[12] D. Clark and W. Fang, "Explicit allocation of best-effort packet delivery service," IEEE/ACM Transactions on Networking, vol. 6 no. 4, Aug. 1998.

[13] S. Floyd and V. Jacobson, "Link-sharing and resource management models for packet networks," IEEE/ACM Transactions on Networking, vol. 3, no. 4, Aug. 1995.

[14] ITU-T Rec. X.739, "Information Technology - Open Systems Interconnection, Systems Management Functions - Metric Objects and Attributes", 1992.

[15] ObjectSpace Inc. Web site, www.objectspace.com, May 2001.

[16] C. Cetinkaya, V. Kanodia, and E. Knightly, "Scalable Services via Egress Admission Control," IEEE TRANSACTIONS ON MULTIMEDIA: Special Issue on Multimedia over IP, vol. 3, no. 1, March 2001.

[17] M. Louta and A. Michalas, "Quality of Service Management in IP networks through Dynamic Service Rate Reconfiguration", Journal of Internet Technology, vol 7, no. 1, 2006.

[18] M. Louta, A. Michalas, E. Loutas, and V. Loumos, "Quality of Service Management in IP networks", Proc. of 7th International Conference on Telecommunications ConTEL IEEE 2003, Zagreb, Croatia, June 2003.

[19] A. Michalas, M. Louta and V. Loumos, "Efficient Call Admission Control Methods in DiffServ Architecture," Proc. 12th IEEE MELECON 2004, Dubrovnic, Croatia, May 2004.

[20] A. Michalas, P. Fafali, M.Louta, and V. Loumos, "Proportional Delay Differentiation Employing the CBQ Service Discipline,", Proc. 7th International Conference on Telecommunications ConTEL IEEE 2003, Zagreb, Croatia, June 2003.

[21] A. Michalas, M. Louta, P. Fafali, G. Karetsos, V. Loumos, "Proportional Delay Differentiation Provision by Bandwidth Adaptation of Class Based Queue Scheduling", International Journal of Communication Systems, Wiley InterScience, vol. 17, Issue 7, July 2004.