An example of traffic-accomodating application.

M. Podestà ***, S. Giordano**,.P. Cremonese*
*** m.podesta@tlcpi.finsiel.it Finsiel S.p.A via Matteucci 34 Pisa Italy
ph +39 050 968526 fax +39 050 968525
** silvia.giordano@epfl.ch EPFL Lausanne Switzerland
ph +41 21 6936748 fax +41 21 693 6141
*p.cremonese@tlcpi.finsiel.it Finsiel S.p.A via Matteucci 34 Pisa Italy
ph +39 050 968516 fax +39 050 968525

Keywords: RSVP, RVBR, QoS, MPEG-4

The paper, if accepted, will be presented by Piergiorgio Cremonese who will attend the conference.

The paper covers the Multimedia Communication Area.

Abstract

The traffic generated by multimedia applications presents a great amount of burstiness, which can hardly be described by a static set of traffic parameters. The dynamic and efficient usage of the resources is one of the fundamental aspects of multimedia networks: the traffic specification should first reflect the real traffic demand, but optimise, at the same time, the resources requested. This paper presents an example of application able to accommodate its traffic to managing QoS dynamically. The paper is focused on the technique used to implement the Dynamic Reallocation Scheme (RVBR) taking into account problems deriving from delay during the reallocation phase.

1 Introduction

Future applications will make use of different technologies as voice, data, and video. These multimedia applications require, in many cases, a better service than a best-effort service. This service is generally expressed in terms of Quality of Service (QoS), whereas network efficiency depends crucially on the degree of resources sharing inside the network.
To achieve both applications QoS requirements and network resources efficiency it is extremely important, for several reasons, network dimensioning or traffic charging.
The evolution of multimedia applications has pointed out how the QoS management must be supported by the network as well as at the application layer. The resource optimisation is possible only if requests for reservation fit as much as possible the effective resource occupation. It follows that applications must be enabled to directly manage the QoS in order to limit the resource lost.
The introduction of the renegotiable variable bit rate (RVBR) service [1], [2] at application layer is assumed to simplify and generalise this task. Whenever re-negotiation is underway, the RVBR scheme generates a traffic specification conforming to the real demand to reallocate the network resources in an optimal way while guaranteeing QoS to the traffic flows. The RVBR service uses the knowledge of the past status of the system and the profile of the traffic expected in the near future, which can be either pre-recorded or known by means of exact prediction.
We propose an example of a multimedia application (called Armida) supporting dynamic QoS management based on RSVP that integrates RVBR services. Armida provides MPEG4 streaming video over an IP network in Microsoft environment.
The rest of the paper is organised as follows. In the next section we provide a short overview of the RSVP protocol. Then we describe the Renegotiable Variable Bit Rate mechanism as defined in [2]. In the fourth section we introduce the Armida application pointing out the component implementing the signalling protocol. Finally we provide a set of results related to a real case in which we compared the required bandwidth (derived from generated traffic) and the reserved QoS, varying the number of performed re-negotiations. We provide also an analysis of re-negotiation cost in terms of time required to set up the new QoS.

2 QoS management via RSVP.

The QoS management in the Internet is performed via the Resource ReSerVation Protocol (RSVP) [6]. RSVP is the signalling protocol implementing the QoS management according to the model defined by the Integrated Services (IS) group within IETF [8].
RSVP allows the reservation of resources for a flow, seen as a sequence of datagrams.
The sender sends the characteristics of the traffic in the *Tspec* traffic descriptor, contained in the PATH message. The receiver tries to set up a reservation related to the received PATH message issuing a RESV message.
The reservation is periodically refreshed (suggested refresh period is currently 30 seconds), i.e. the PATH and the RESV messages are reissued.
IS defines three classes of services: Guaranteed Service [7], Control Load Service (CLS)[9] and Best Effort Service. In the rest of the paper we will focus only on the CLS.
CLS provides the client data flow with a quality of service closely approximating the QoS that the same flow would receive from an unloaded network element, but uses a capacity (admission) control to assure that this service is received even when the network element is overloaded. The end-to-end behaviour offered by the controlled-load service to an application, under the assumption of a correct functioning of the network, is expected to provide little or no delay and congestion loss.
The sender provides the information of the data traffic it will generate in the *Tspec*. The parameters specified by the *Tspec* are:

- peak rate *p*
- bucket rate *r*
- bucket size *b*

In addition, there is a minimum policed unit *m* and a maximum packet size *M.*
The service offered by the network ensures that adequate network resources are available for that traffic.

The controlled-load service is well suited to those applications that can usefully characterise their traffic requirements and are not too sensible to eventual delays or losses.

3 Dynamic QoS: RVBR

The RVBR service is based on a renegotiable VBR traffic specification, and offers a scheme to optimise the traffic specification in the next period of time during which this traffic specification is valid [2], [3]. RVBR service uses the knowledge of the past status of the system and the profile of the traffic expected in the near future, which can be either pre-recorded or known by means of exact prediction. This scheme suits perfectly the dynamics of the traffic generated by multimedia applications.
Moreover it naturally integrates with the soft state mechanism of RSVP, which allows for resources renegotiating.
We first recall the characterisation of the RVBR service in terms of input and output functions as given in [1].

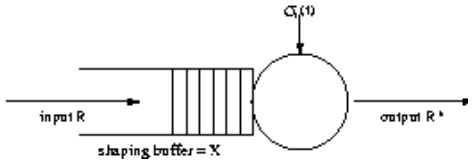

Figure 1 RVBR reference configuration

There is a renegotiable leaky bucket specification (with rate *r* and depth *b*) plus a fixed size buffer *X* drained at maximum at renegotiable peak rate *p*. The elements of a RVBR source, Figure 1**,** are a renegotiable leaky bucket specification (with rate *r* and depth *b*) plus a fixed size buffer *X* drained at maximum at renegotiable peak rate *p*. In **[2**] the RVBR service is described with two leaky bucket specifications. In the case of RSVP, the bucket associated to the peak *p* is the MTU size, hence it is fixed. We further assume it equal to zero to simplify the computation, given that this is not a limitation.
The observation time is divided into intervals, and $I_i = [t\_i, t_{i+1}]$ represents the i-th interval. Inside each interval the system does not change. The parameters of the RVBR service in I_i are indicated with (p_i, ri, b_i).
The RVBR service is completely defined by:

- the time instants t_i at which the parameters changes
- the RVBR parameters (p_i, r_i, b_i), for each interval *Ii*
- the fixed shaping buffer capacity *X*

In the following we will show how RVBR can be used to implement signalling adaptive applications.

4 ARMIDA description

ARMIDA is a MPEG-4 [5] compliant client-server platform. It provides Video Streaming feature, potentially requiring a large amount of bandwidth with strict QoS bounds to satisfy audio/video requirements. It provides features to manage several multimedia streams (named Elementary Stream) which are combined together by the client according to synchronisation requirements.
The ARMIDA architecture is characterised by the introduction of the intermediate DMIF (Delivering Multimedia Interface Framework)[4] layer to make the application independent from the network. The final architecture consists of three layers: the “real” application, the DMIF filter (the part independent of the delivery technology and the part dependent on the delivery technology), and the Daemon process.

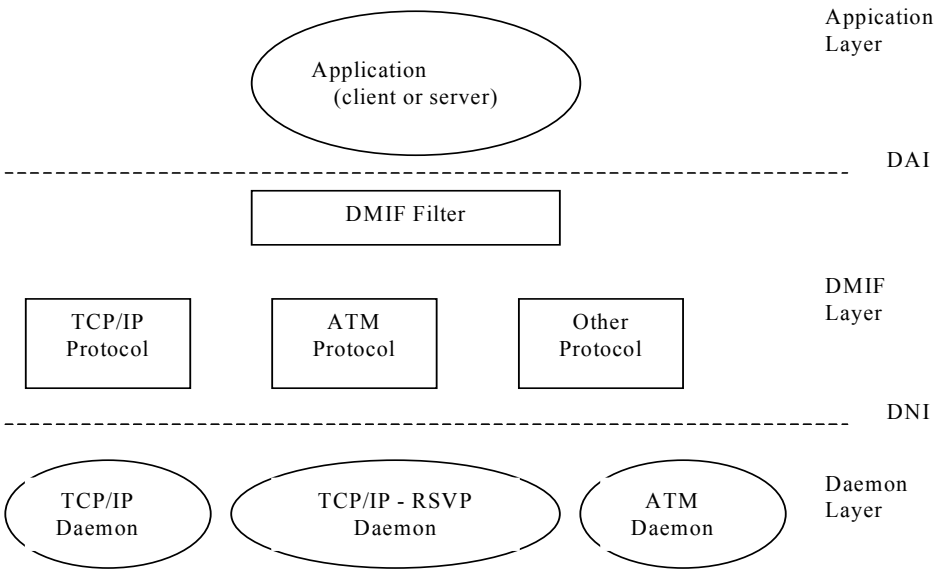

Figure 2: ARMIDA Architecture

DMIF provides QoS management aspects and mechanism to gather information about data transfer and resource utilisation. A standard definition of QoS format would be needed to guarantee a general mapping from user QoS (DMIF) to network QoS (RSVP, etc), providing information about data sources.
The following parameters are passed from the application to the DMIF:

- MAX_AU_SIZE: maximum size of an access unit. It is expressed in bytes;
- AVG_BITRATE: average bit rate. It is expressed in bytes/second;
- MAX_BITRATE: maximum bit rate. It is expressed in bytes/second;
- SERVICES_CONSTRAINT, which indicates if the traffic requires strict bounds of delay;
- TIME_LENGTH, which contains the whole duration of the stream to be transmitted by the application;
- BURST_SIZE: is the burst dimension, which the application foresees to send.

These values can be calculated because ARMIDA is a video-streaming application, where the traffic is known in advance and measurements can be made in order to get these values.

5 ARMIDA with RVBR

The introduction of RVBR within ARMIDA makes an impact on three layers:

1. Application Layer: several QoS descriptors must be handled. The current standard defines that only a single QoS descriptor can be associated to an Elementary Stream. In order to introduce the re-negotiation, it is necessary to define the association between several QoS descriptors and an Elementary Stream. Additionally, the structure for maintaining and managing several QoS descriptors for the same data flow is needed. It means that a data flow must store more than one QoS descriptor and a reference to the related portion of data.
2. DMIF layer: at each re-negotiation, it must provide the new QoS descriptors to Daemon layer.
3. Daemon layer: several re-negotiation phases must be managed. It must be able to
   - identify when an old QoS expires to start a new QoS
   - interact with the RSVP which must modify the *Tspec* sent with the PATH message asking for a new reservation.

The RVBR can be used only with the CLS. Unlike the GS, in the CLS reservation is always successful when the first resv message is received, even if the reservation characteristics are unknown, because the CLS does not require bounds on end-to-end datagram queueing: so reservation failure due to resource lack is not possible. Moreover, the *Tspec* in the CLS takes the form of a double bucket specification as given by the RVBR service: there is a peak rate p and a leaky bucket specification with rate r and bucket size b.
The video stream has to be divided into time intervals, each of which requires a homogeneous QoS descriptor, and its duration has to be greater than those of the soft state, so the application is protected from

the loss of RSVP packets. We can divide the total duration T of the stream in a set of intervals related to each required reallocation: $T=\{T_1,T_2,\ldots,T_n\}$.
RSVP provides features to ask for a new reservation conformant to traffic characteristics. The nature of the application, a streaming of pre-recorded video, allows to know all generated traffic information, e.g. rate, burst, peak, etc. It enables the implementation of a completely deterministic system based on the knowledge of parameters suiting perfectly the dynamics of generated traffic.
The reservation mechanism provided by RSVP are based on the re-sending of a control message (PATH) at every pre-defined time interval (default is 30 sec), carrying information about the required QoS. The usual behaviour foresees that RSVP daemon re-sends the same PATH until the end of data communication.
The idea is to exploit this mechanism also to change the reservation sending a new PATH message carrying the values related to the new traffic descriptor defined in the *Tspec*.
The re-negotiation must take into account the following:

- The new reservation must avoid removing resources when these are still needed (e.g. new reservation starts before the new interval)
- The new reservation must guarantee enough resources to the application according to requirements (e.g. a new reservation starts late)

The main problem is to determine when the new reservation phase must start to be sure that data sending and signalling are synchronised to guarantee always the requested resources.

5.1 Reallocation time evaluation.

To evaluate when the new reservation phase must start, we need to know the time spent to install a generic reservation, i.e. the time spent to send a path message and receive a resv message. The time or delay needed to install a generic reservation takes variable values. To calculate it, we assume a Poisson distribution of the delay where the probability that the delay x spent is less than T is y:

$$P\{x \le T\} = y$$
$$P\{x \le T\} = 1 - e^{-\lambda T}$$

Where $\lambda = 1/E\{x\}$.
The mean value $E\{x\}$, and then λ, is function of the network properties. In our experiments, server and client belong to the same network, which is half-duplex with a bandwidth of 10 Mbits. By considering two access time periods (needed to complete the reservation) and an elaboration time on the terminal of 20%, we have $E\{x\}$ = 240 milliseconds. This result is confirmed by experimental observations, where we have measured the delay spent to install the reservation for each request to change it.

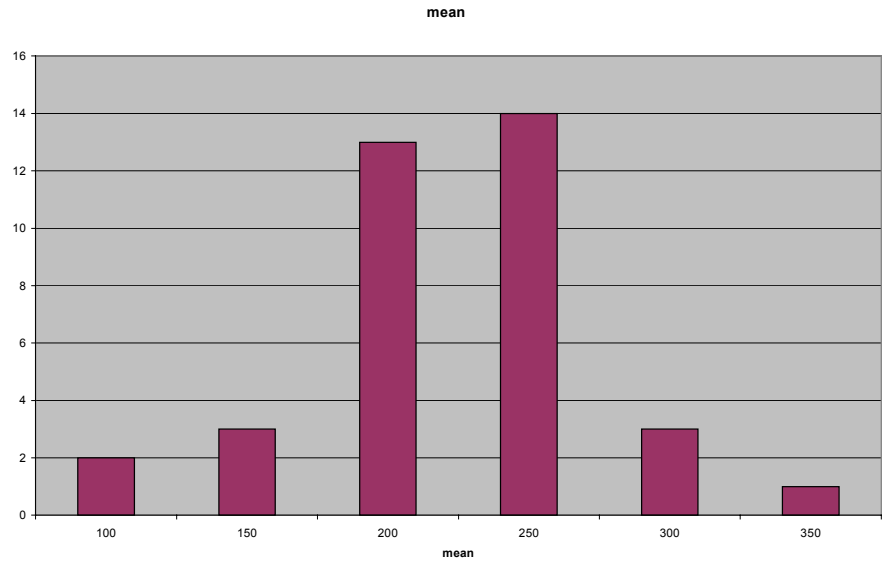

Figure 3 Behaviour of reallocation time

Then, with $E\{x\}$ known, it is possible to estimate a time needed to set up the reservation T observed with probability y:

– $y = 0.9 \Rightarrow T = 552.62$ msec.

Knowning T, we can decide when a new reservation phase must start by considering the *Tspec* associated to the actual time interval and to the previous and successive interval.

Let Θ be the set of QoS defined as follows:

$$\Theta=\{\theta\iota:\theta\iota \text{ is the QoS related to the i-th interval of the data sequence}\}$$

We can define on Θ a Total Order < as follows

Let $\theta1, \theta2 \in \Theta$ be defined as $\theta1 =[r_1, b_1, p_1]$, $\theta2=[r_2, b_2, p_2]$

$$\theta1 < \theta2 \text{ if } (r_1+0.5* b_1) < (r_2+0.5* b_2) \qquad [F1]$$

We can define a relationship between *Tspec* and Θ mapping each $\theta \in \Theta$ on the *Tspec* carrying related values.

In the case of the first time period, the actual *Tspec* has to be compared with the next one via [F1] in order to assure that at the end of the first interval, a correct reservation for the data associated to the second interval has been installed.

Let $Tspec_1$ and $Tspec_2$ be *Tspec* related to the first and the second interval

- if $Tspec_2 < Tspec_1$, the procedure needed to set up the second reservation can start at the end of the first interval, because the resources actually reserved are sufficient to assure a correct data sending.
- if $Tspec_1 < Tspec_2$ the procedure has to be anticipated in order to guarantee enough resources to the second group of data.

Let TR_i be the time needed to set up the reservation related to the i-th interval.

In the first case, the path message containing the new *Tspec* can be sent at the end of the first interval; in the second case, the path message has to be sent T seconds before the end of the time period.

In the case of the successive time periods, i.e. the general case, we need also to consider the previous reservation, because in order to evaluate the duration of the i-th reservation, we have to know when this reservation has take place.

We can define the starting time TSi of each reservation as follows:

- $Tspec_n<Tspec_{n-1}$ then: $TSn=T_{n-1}+TR_n$
- $Tspec_{n-1}<Tspec_n$ then: $TSn=T_{n-1}-(T-TR_n)$

We can generalise the procedure evaluating the starting time for each reservation related to each interval Ti as follows.

Let TEi be the time when the i-th reservation is replaced by the new one (i+1).

- $Tspec_n<Tspec_{n+1}$: $TEn=T_n-T+TR_{n+1}$
- $Tspec_{n+1}<Tspec_n$: $TEn=T_n+ TR_{n+1}$

Let TDi be the duration of reservation related to T_i.

We can summarise the behaviour as follows:

- $TDi=T_i - TR_i$ if $Tspec_i<Tspec_{i-1}$ and $Tspec_i>Tspec_{i+1}$;
- $TDi=T_i - T - TR_i$ if $Tspec_i<Tspec_{i-1}$ and $Tspec_i <Tspec_{i+1}$;
- $TDi=T_i - T + TR_i$ if $Tspec_i>Tspec_{i-1}$ and $Tspec_i >Tspec_{i+1}$;
- $TDi=T_i - T + (T - TR_i) = T_i - TR_i$ if $Tspec_i>Tspec_{i-1}$ and $Tspec_i < Tspec_{i+1}$;

The following pictures show this concept.

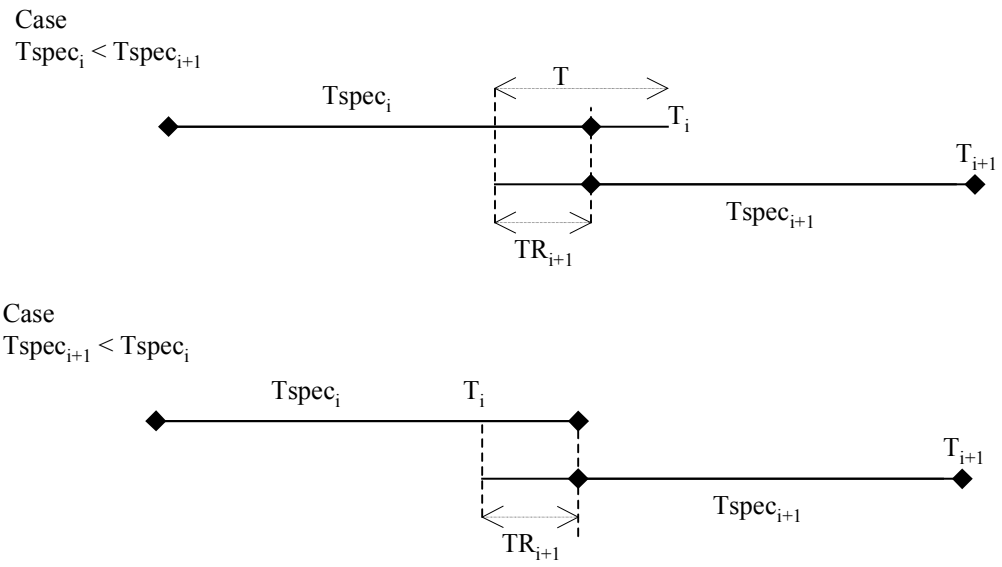

Figure 4: Resources overlapping

In Figure 5 we show a practical case referred to a video stream of 280 sec where reallocation actions are performed every 60 seconds. The allocated QoS is always greater or equal to the QoS required by the application. The trial has been performed on a LAN Ethernet.

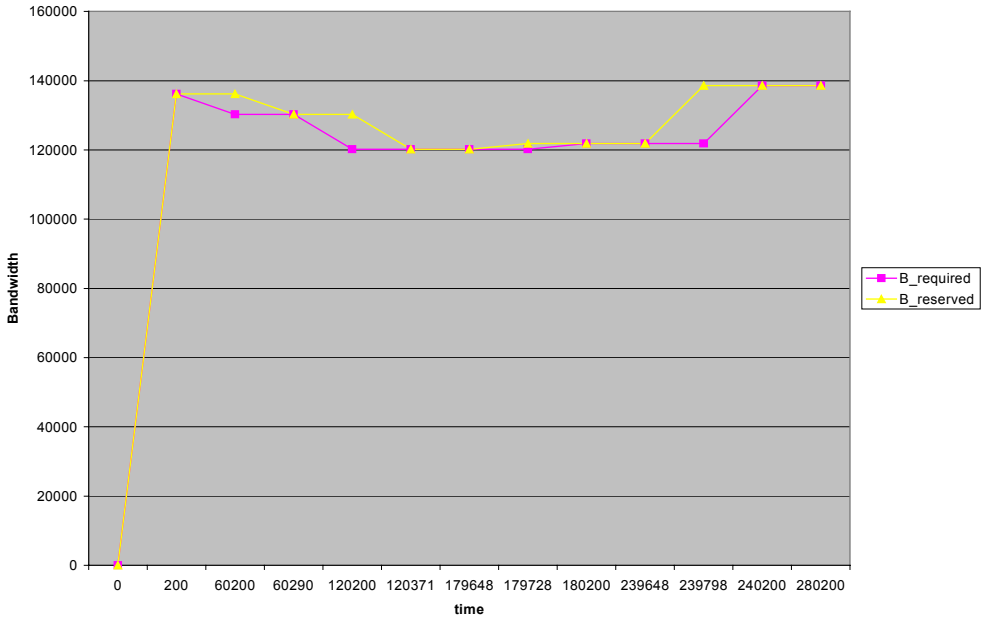

Figure 5 Example of reallocation time

5.2 QoS input parameters

We must distinguish between QoS defined at the network layer from a QoS defined at the application layer. RSVP provides features to set up a communication according to a defined QoS at the network layer; the same is provided by the DMIF. Because of the quite initial state of DMIF standardisation (related to QoS definition) and the presence of fields to be defined we propose to introduce the following parameters in order to support RSVP with RVBR:

- SERVICES_CONSTRAINT: it indicates if there are some restrictions for the traffic delay, it is a property of the entire ES.
- INTERVAL_NUMBER: it specifies the number of time intervals composing the stream.
- MAX_AU_SIZE: it is the maximum size of an access unit; it can be considered a global parameter.
- TIME_LENGTH: it is the total length of the stream.

Parameters listed above are common to the complete stream; there are also some parameters, which have to be replicated for each time interval:

- AVG_BITRATE: it is the average bit rate of each interval.
- MAX_BITRATE: it is the maximum bit rate of each interval.
- BURST_SIZE: it is the burst size of each interval.
- INTERVAL_TIME: it is the duration of each interval.

It follows that the QoS DMIF descriptor of each Elementary Stream is composed by two main parts:

- global parameters
- array of repeated parameters: the dimension is equal to $|\{T1,\ldots,Tn\}|$

The computing of the total QoS parameters is made for each interval, according to the previous rules; for this purpose, the interval lengths of Elementary Stream, which will be composed in the same TransMux, have to be equal to allow a right combination of them.

6 Conclusions

We have presented an example of a *traffic-accomodation application* able to manage dynamically QoS according to traffic requirements. Figure 6 and Figure 7 show the effective generated traffic and the allocated resources with reallocation intervals defined at 60 sec and 280 sec. It is obvious that increasing the number of segments with different QoS parameters, we need fewer resources with lower bandwidth lost (allocation greater than the real traffic) and limited amount of buffer for traffic shaping. We have also shown as the *reallocation activity* affects the communication.

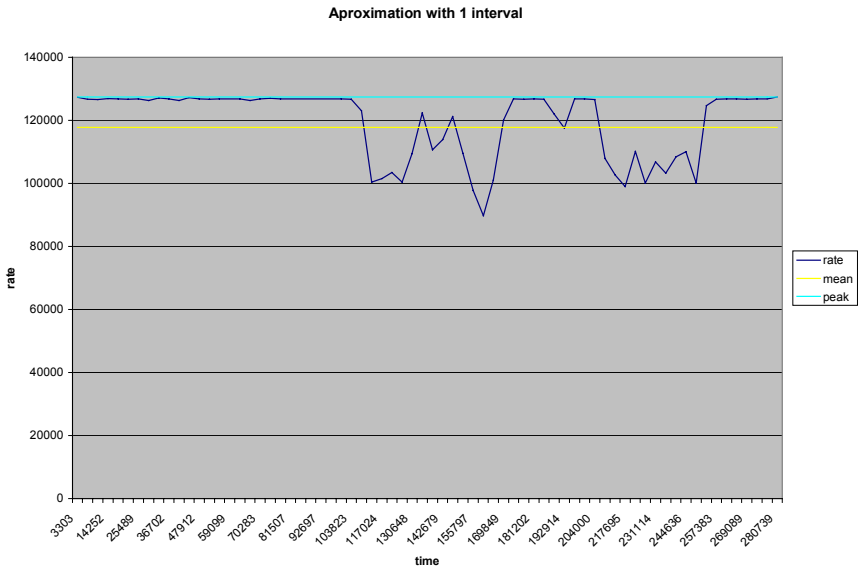

Figure 6 Example of reallocation every 280 sec

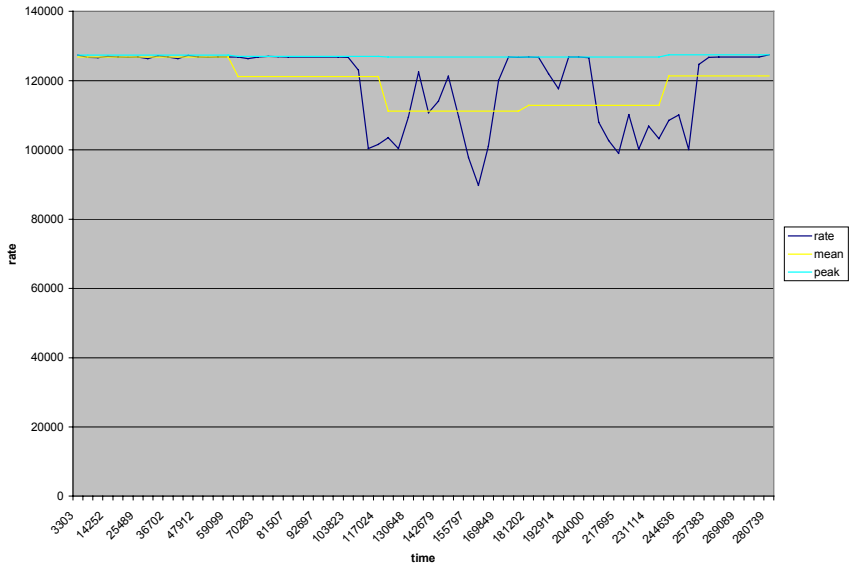

Figure 7 Example of reallocation every 60 sec

In Figure 8 we show the buffer utilisation for the same experiment. The comparison of these pictures confirms that the scheme we propose allows to exploit the available resources in terms of bandwidth and buffer occupation with a limited overhead deriving from the effort needed for reallocation.

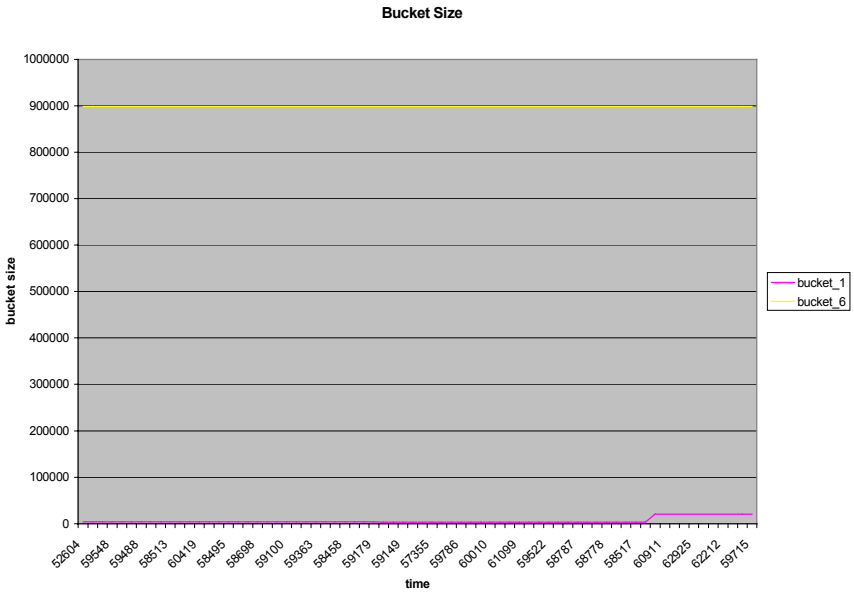

Figure 8 Buffer utilisation for reallocation time 60 and 280 sec.

7 References

[1] S. Giordano, J.-Y. Le Boudec: **"On a Class of Time Varying Shapers with Application to the Renegotiable Variable Bit Rate Service"** , SSC Technical Report no. 98/035, 1998.

[2] S. Giordano, J.-Y. Le Boudec **"The Renegotiable Variable Bit Rate Service"** SSC Technical Report no. 98/038, 1999.

[3] S. Giordano, J.-Y. Le Boudec **"QoS based Integration of IP and ATM: Resource Renegotiation"** , SSC Technical Report no. 98/030, in Proceedings of 13th IEEE Computer Communications Workshop 1998.

[4] Information technology, Generic coding of moving pictures and associated audio information, art 6: Delivery Multimedia Integration Framework - ISO 1998.

[5] Information technology, Generic coding of moving pictures and associated audio information, Part 1: System International Standard Organisation 1988.

[6] R. Braden, L. Zhang, S. Berson, S. Herzog, S. Jamin RFC2205: Resource ReSerVation Protocol (RSVP) – IETF 1997.

[7] S. Shenker, C. Partridge, R. Guerin RFC2212: Specification of Guaranteed Quality of Service IETF 1997.

[8] J. Wroclawski RFC2210: The Use of RSVP with IETF Integrated Services IETF 1997.

[9] J. Wroclawski RFC2211: Specification of Controlled-Load Network Element Service IETF 1997.

[10] S. Shenker, J. Wroclawski RFC2216: Network Element Service Specification Template IETF 1997.