A Model for Dynamic QoS Negotiation applied to an MPEG4 Application

S. Giordano*, P. Cremonese**, J-Y. Le Boudec*, M. Podestà ***

* EPFL Lausanne Switzerland
** Netikos S.p.A via Matteucci 34 Pisa Italy
*** Whitehead, Livorno Italy

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 chapter presents: a model for dynamically renegotiating the traffic specification (RVBR), how this can be integrated with the traffic reservation mechanism RSVP, and an example of application able to accommodate its traffic to managing QoS dynamically. The remaining of this chapter is focused on the technique used to implement RVBR) taking into account problems deriving from delay during the renegotiation phase and on the performance of the application with MPEG4 traffic. **Keywords**: RSVP, RVBR, QoS, MPEG-4

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 [Giordano,1999], [Giordano,2000] 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 renegotiate 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 chapter is organised as follows. In the next section we provide a short overview of the RSVP protocol. Then we describe the RVBR mechanism as defined in [Giordano,2000]. 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.

BACKGROUND

QoS management via RSVP.

The QoS management in the Internet is performed via the Resource ReSerVation Protocol (RSVP) [Braden,1997]. RSVP is the signalling protocol implementing the QoS management according to the model defined by the Integrated Services (IS) group within IETF [Wroclawski-2210,1997].
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 (GS) [Shenker,1997], Control Load Service (CLS) [Wroclawski-2211,1997] and Best Effort Service. In the rest of the chapter 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.
As this work was unfolding, the media was busily hyping RSVP as a panacea - the magic cure that would bring an end to all network woes. As is often the case with over-hyped technologies, RSVP and IS failed to deliver on the promises [DeSousa,1999]. However, RSVP found a new application for the configuration of traffic handling mechanisms. The ISSLL working group of the IETF has developed a model by which RSVP signalling is used with diffserv traffic handling in order to enable QoS in a scalable manner. In addition, by listening to RSVP signalling, network devices are more readily able to identify and classify traffic in order to determine the appropriate traffic handling mechanisms to apply [DeSousa,1999].

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 [Giordano,1999], [Giordano,1998].
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. 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*.

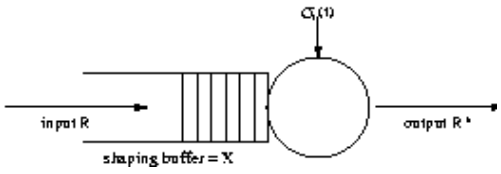

Figure 1 RVBR reference configuration

The elements of a RVBR source, as shown in 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*. Therefore, the RVBR service is described with two leaky bucket specifications [Giordano,2000], [Giordano,1999]. In the case of RSVP, the MTU size is the bucket associated to the peak *p*, 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 $T_i = [t_i, t_{i+1}]$ 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 T_i
- the fixed shaping buffer capacity *X*

The input-output characterisation of the RVBR service comes straightforward as a special case of the time varying leaky bucket shaper [Giordano,2000], that is defined by means of Network Calculus theory [LeBoudec,2000], [Thiran,2000] as :

$$R^*(t) = min \{ \sigma^0_i (t - t_i) + R^*(t_i), inf_{t_i <= s <= t} \{ \sigma_i(t-s) + R(s) \} \} \quad [eq1]$$

where σ^0_i, representing the service curve taking into account the initial conditions at time t_i , is defined as

$$\sigma^0_i(u) = min_j (r_i^j * u + b_i^j - q^j(t_i))$$

and $q^j(t)$ is the bucket level of the j-th bucket, defined as, at time t in T_I

$$q^j(t) = \max \{ sup_{t_i < s <= t} \{ R^*(t) - R^*(s) - r_i^j (t-s) \}, \{ R^*(t) - R^*(t_i) - r_i^j (t-t_i) + q^j(t_i) \} \} \quad [eq2]$$

Moreover, $w(t_i)$ is the backlog in the shaping buffer at time t in T_i

$$w(t) = max \{ sup_{t_i < s <= t} \{ R(t) - R(s) - \sigma_i(t-s) \}, \{ R(t) R(t_i) - \sigma^0_i (t-t_i) + w(t_i) \} \text{ for t in } T_i \quad [eq3]$$

An RVBR source is a time varying leaky-bucket shaper with two renegotiable leaky buckets (J=2); one with rate r_i and depth b_i and the second with rate p_i and depth always equal to zero, plus a buffer of fixed size X. Therefore, in the Equations [eq1], [eq2], and [eq3], σ_i and σ^0_i are given by

$$\sigma_i(u) = min (p_i * u + b_i^1, r_i * u + b_i^2)$$

$$\sigma^0_i(u) = min (p_i * u + b_i^1 - q^1(t_i), r_i * u + b_i^2 - q^2(t_i))$$

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

The application of the RVBR Service to RSVP

In real life, examples of this service are traffic shaping done at source sending over VBR connections as defined in [ITU,1998] and Internet traffic that takes the form of IntServ specification with RSVP reservation [Braden,1997], [Wroclawski-2210,1997].

In RSVP, the sender sends a PATH message with a *Tspec* object that characterises the traffic it is willing to send. When we consider a network that provides a service as specified for the Controlled Load service (CLS) [Wroclawski-2211,1997], the *Tspec* takes the form of a double bucket specification [Wroclawski-2210,1997] as given by the RVBR service. There is a peak rate *p* and a leaky bucket specification with rate *r* and bucket size *b*. Since, with RSVP as reservation protocol, the reservation has to be periodically refreshed, *p, r* and *b* need to be reissued at each renegotiation time. There is no additional signalling cost in applying a *Tspec* renegotiation at that point, even if there is some computational overhead due to the computation of the new parameters, or to the call admission control, etc. It is important to note here that, contrary to the negotiation of a new connection, with the renegotiation the reservation is never interrupted.

If the network cannot support the requested traffic specification, the old traffic specification is restored and the network may not be able to accommodate the next traffic. Mechanisms to prevent this failure from occurring are still under study. Here we assume that either (1) the *Tspec* is accepted all over the network, as well as at the destination, such that the source can transmit conforming to its desired traffic specification or (2) the source can adapt to transmit with the old *Tspec*, even if at the price of a reduced quality.

We assume that at any time t_i=30 * i the application knows (because pre-recorded or predicted) the traffic for the next 30 seconds. We further assume to know the cost to the network of the *Tspec* (indicated by the cost function (u * r + b) and the upper bound to the bucket size b_{max} and to the bucket rate r_{max}. The backlog $w(t_i)$ and the bucket level $q(t_i)$ can be measured in the system. In order to use RVBR service for RSVP with CL service scenario, we are faced with the problem of computing the leaky bucket parameters. Therefore, we describe the case of a source that wants to reserve the resources for the next interval. For the RVBR service, this is equivalent to the problem of computing the RVBR parameters for the next interval. Here we present the approach that we used in the simulations [Giordano,1998]. As we will see later, in real cases, this approach can require some modifications.
From Equations [eq1] and [eq2] it comes

$R(t)-R(s) <= \sigma_i(t-s) + X \qquad t \text{ in } T_i,\ t_i < s <= t$

$R(t)-R(t_i) <= \sigma^0_i(t-t_i) - w(t_i) + X \qquad t \text{ in } T_i$

These equations give a necessary and sufficient condition for a minimum p_i. This, in analogy to the work in [LeBoudec,1997] can be seen as the *effective bandwidth* of the arrival stream in I_i taking into account the backlog at time t_i. Therefore, given that p_i is computed independently from r_i and b_i, the problem of finding a complete optimal parameter set (p_i,r_i,b_i) for the RVBR service is reduced to the problem of finding the optimal parameters r_i and b_i.
This optimisation problem, when the cost function is linear: $c(r_i,b_i)=u * r_i+b_i$, for fixed values of u, is modelled in [Giordano,2000] in form of an algorithm (*LocalOptimm1*).

Algorithm 1 Local Optimum $\left(X, \{R(t)\}_{t\in I}, b_{\max}, r_{\max}, u, w(t_i), q(t_i), t_{i+1}\right)$

If $b_{\max} < \sup_{s\in I}\{\beta_i(s) - r_{\max}s - X\}$ then
there is no feasible solution;
Else
{

$$p_i = \max\left(\sup_{t,s\in I_i}\frac{R(t)-R(s)-X}{t-s}, \sup_{s\in I_i}\frac{R(t_i)-R(s)-X+\omega(t_i)}{t_i-s}\right)$$

If (*u≤0) then {*
X_0=min(v_{max},p_i);
}else {

$$x_0 = \sup_{s\in I}\frac{\beta_i(s)-\beta_i(u)}{s-u}$$

$$x_A = \sup_{s\in I, s>0}\frac{\beta_i(s)-X-b_{\max}}{s}$$

$$x_B = \sup_{s\in I, s>0}\frac{\beta_i(s)-X}{s}$$

if $(x0 \geq \min(x_b, r_{\max}, p_i))$ then $x0 = \min(x_b, r_{\max}, p_i)$
else if $(x_0 \leq x_a)$ then $x_0 = x_a$;

}

$r_i = x_0$;

$b_i = \sup_{s \in I} \{\beta_i(s) - X - sx_0\}$;

}

RVBR when the traffic is specified by its arrival curve

LocalOptimum algorithm uses functions that imply the exact traffic. In the real case of video stream applications, the module that implements RVBR has to access information related to the network and, therefore, even if the traffic is pre-recorded and stored, it is very unlikely to have access to the exact traffic.
To the aim of using the algorithm in a real application, we propose an approximation to some functions, which originally work with the exact traffic, in order to work with a smaller and less precise information: the exact traffic R(t) for t in I_i is substituted by upper bound functions. We introduce the function:

$$\alpha_i(u) = min\ (p^{\alpha}_i * u, r^{\alpha}_i * t\ u + b^{\alpha}_i)$$

where

$$p^{\alpha}_i = sup_{t,s}\ (R(t)-R(s))/\ (t-s)$$

$$r^{\alpha}_i = (R(t_i+1)-R(t_i))/(t_i+1\ -t_i)$$

$$b^{\alpha}_i = St\ [R(t)- r^{\alpha}_i * t]^{+}$$

and a second function that takes in account the traffic $q(t_i)$ that is the bucket at the transient period.

$$\alpha_i{}^0(u) = min\ (p^{\alpha 0}_i * u, r^{\alpha}_i * t\ u + b^{\alpha}_i - q(t_i))$$

where

$$p^{\alpha 0}_i = sup_t (R(t)-R(t_i))/(t- t_i)$$

These functions are arrival curves [LeBoudec,2000] of R(t), i.e. upper bounds to the traffic R(t). With the introduction of α_i and $\alpha_i{}^0$we can approximate the function b_i and the optimal peak rate p_i that in the RVBR are originally computed from the exact traffic R(t).
Therefore, indicating with $w(t_i)$ the backlog in the shaping buffer at time t_i , the function b_i is given by:

$$b_i(s) = max_s\ (\ (\alpha(s), \alpha_i{}^0\ (s) + w(t_i) + q(t_i)\ )$$

and the minimum p_i by

$$p_i = max\ (sup_s\ (\alpha\iota(s\ )-X)/s\ ,\ sup_s\ (\alpha_i{}^0\ (s\ )-X+w(t_i)\ )/s)$$

$\alpha\iota$ and $\alpha_i{}^0$ can be used in a real implementation, because computed with only four parameters (p^{α}_i, r^{α}_i , b^{α}_i, $p^{\alpha 0}_i$) that can be easily stored and passed from the application level to the RVBR module.

AN MPEG4 APPLICATION: ARMIDA

ARMIDA is an MPEG-4 [ISO-1,1998] 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.
In the version used in this work, the ARMIDA architecture is characterised by the introduction of the intermediate DMIF (Delivering Multimedia Interface Framework)[ISO,1998] 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.

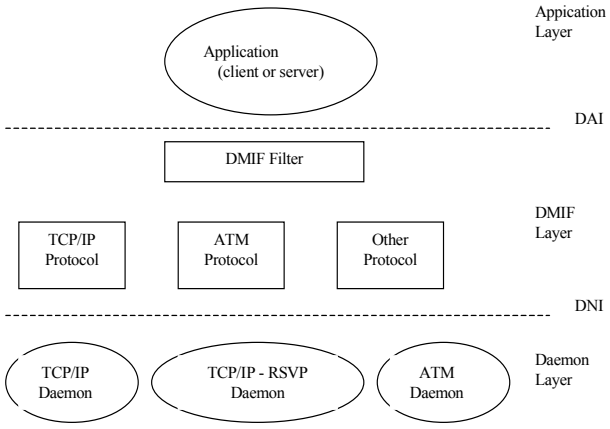

Figure 2: ARMIDA Architecture

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.

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 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 renegotiation: $T=\{T_1,T_2,\ldots,T_n\}$.
In ARMIDA we implemented both the GS and the CLS, but the RVBR is more suitable with the CLS. Unlike the GS, a CLS reservation is 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 queuing. Therefore, a reservation failure due to resource lack is not possible. In the case of GS reservation, when we introduce the renegotiation, we cannot assure anymore the service for the entire traffic. For

example, if at the interval I_i we require more resources than the resources used in the interval I_{i-1} there is not guarantee that the network can satisfy the request. Therefore, the resulting service is guaranteed only interval by interval (local guarantee), as opposite to the GS (global guarantee).
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.

DYNAMIC QOS FOR MULTIMEDIA TRAFFIC

In the following we present the performance of ARMIDA when used with the RVBR capability. We first introduce a more adaptive approach that is not constrained by a fixed renegotiation period and then we illustrate our results that were also shown in a real demonstration at Telecom99 exposition [DeSousa,1999].

Renegotiation 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:

$P\{ x \leq T\} = y$

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

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 Mbytes. 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.

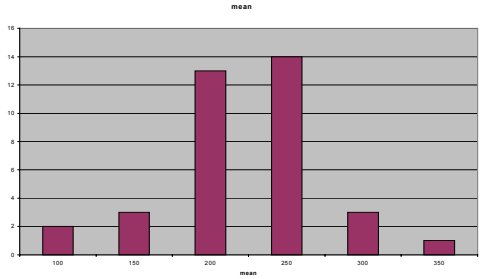

Figure 3 Behaviour of renegotiation time

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.

Knowing 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:

Θ={θι:θι is the QoS related to the i-th interval of the data sequence}

We can define on Θ a Total Order < as follows
Let θ1, θ2 ∈ Θ be defined as θ1 =[r_1, b_1, p_1], θ2=[r_2, b_2, p_2]

θ1< θ2 if (r_1+0.5* b_1) < (r_2+0.5* b_2) [eq4]

We can define a relationship between *Tspec* and Θ mapping each θ ∈Θ 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 [eq4] 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.

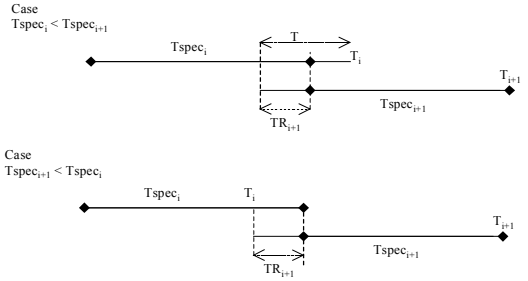

Figure 4: Resources overlapping

In Figure 5 we show a practical case referred to a video stream of 280 sec where renegotiation 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.

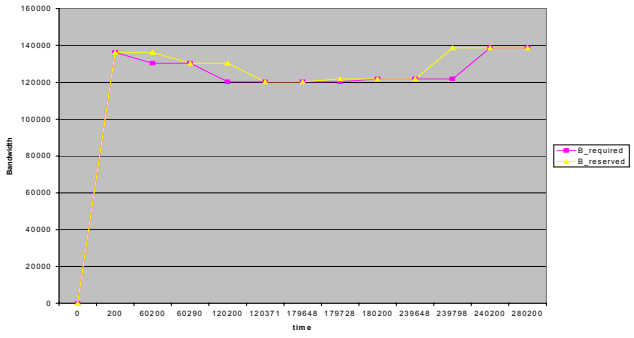

Figure 5 Example of renegotiation time

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 two main parts compose the QOS DMIF descriptor of each Elementary Stream:

- 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.

Traffic renegotiation

We present here some experiment with a real MPEG4 video of 2.2 Mbytes transmitted in 280 seconds over a LAN with a low-average load. Initially all the buffers and buckets are empty (zero initial conditions). The file is pre-recorded pro frames and, given that we do not use any scheduling or pre-fetch, we know R(t) over the entire interval. We also assume that the *Tspec* is accepted all over the network, as well as at destination, such that the source can transmit conforming to its traffic specification.

As it comes from the formulas given above, when we renegotiate, we also know R*(t) for any previous time, and we can measure the buffer and the leaky bucket content. We obtain the optimal shaper parameters by applying the algorithm *LocalOptimum*.

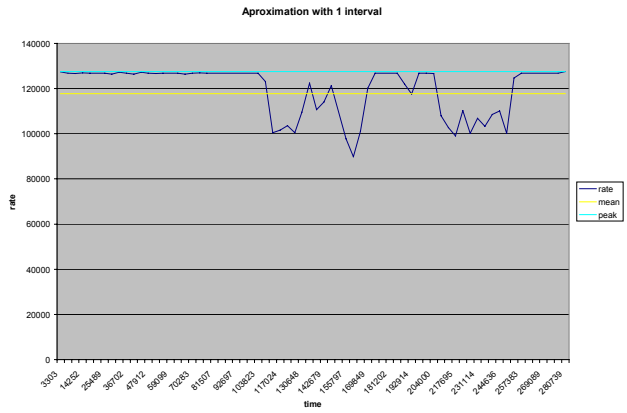

Figure 6 Example allocation without renegoziation

Figure 6 and Figure 7 show the effective generated traffic and the allocated resources with renegotiation intervals defined at 60 sec and with no renegotiation (280 sec.).
It is obvious that increasing the number of segments with different QoS parameters, we need fewer resources with a lower waist of bandwidth (allocation greater than the real traffic) and limited amount of buffer for traffic shaping. We have also shown as the *renegotiation activity* adds an insignificant communication overhead in the case of RSVP.
We have a relatively small improvement for the peak, because the input traffic is not very bursty in the initial and in the final parts. On the contrary, there is a real improvement for the mean rate (and also for the bucket size, as shown in Figure 9). The rate needed with renegotiation is much smaller, because it adapts to the current transmission, and it is not constrained by a fixed initial negotiation as in Figure 6.

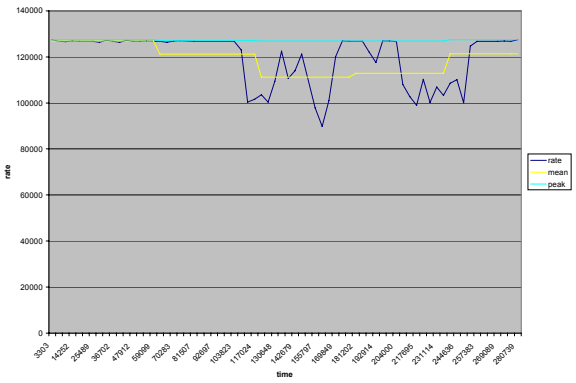

Figure 7 Example of renegotiation every 60 sec

In Figure 8 we compare the behaviour of the system with different reallocation intervals: 60sec, 120sec. and no reallocation. Here it is evident the benefit of renegotiation in terms of resources (that is beneficial to both the network, that has to allocate them) and the application (that, very likely, has to pay for them).

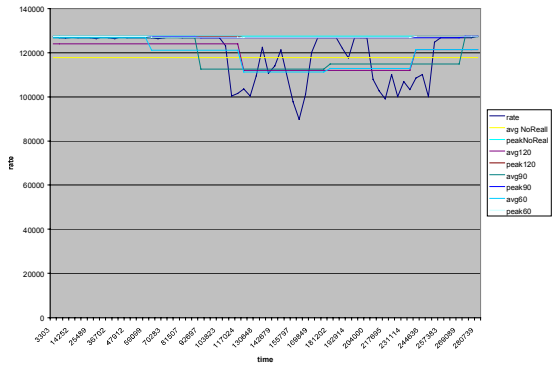

Figure 8 Comparison among different reallocation intervals

Figure 9 illustrates the setting of the bucket size for the same experiment. By the analysis of this figure it is evident that the renegotiation is effective for the, because the algorithm works for smaller interval. The comparison of these curves confirms that the scheme we propose allows to optimise the resources reserved to the network (expressed in terms of bucket), with a limited overhead deriving from the effort needed for renegotiation. These results indicate that renegotiation is an efficient mechanism for allowing the better use of network resources at the very low price of implementing a service like RVBR.

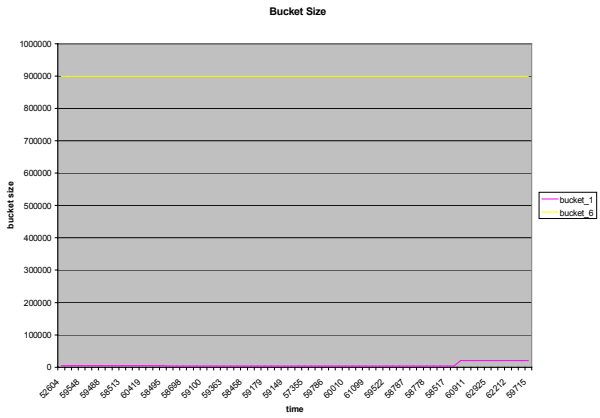

Figure 9 Bucket definition for renegotiation time equal to 60 (bucket_1) and with no renegotiation (bucket_6).

The initial conditions are: $q(0)=0$ and $w(0)=0$. The file is pre-recorded pro frames and, given that we do not use any scheduling or pre-fetch, we know R(t) over the entire interval. We also assume that the *Tspec* is accepted all over the network, as well as at destination, such that the source can transmit conforming to its traffic specification.
As noted above, when we renegotiate, we also know R*(t) for any previous time, and we can measure the buffer and the leaky bucket content. We obtain the optimal shaper parameters by applying the algorithm *LocalOptimum*.

CONCLUSIONS

We presented an example of a *traffic-accommodation application* able to manage dynamically the QoS according to the traffic requirements.
This is obtained with the introduction of the RVBR service, that allows to modify the traffic specification of a connection, while keeping the connection active, in order to support the traffic QoS requirements.

In this respect, we introduced the video on demand application RVBR-enabled ARMIDA, which also became the first instance of an application RVBR-enabled that renegotiates RSVP traffic specification
We solved several issues arising with the integration and the utilisation of this model and illustrated the performance of the application with MPEG4 traffic.
We carried this study on a real network with the client and server exchanging RSVP messages containing a Tspec renegotiated according to the RVBR service. We compared the results of transmitting a MPEG4 video trace with renegotiation (at different renegotiation period) and without renegotiation. The measurements performed, as well as in the real-time demonstration we conducted at Telecom99, showed that the renegotiation allows a better use of network resources and that, with protocols as RSVP, where there is no additional cost for signalling (or so we mainly assume), it is better to renegotiate.
We also presented an advanced study on the renegotiation period that takes in account the possibility of using different period or the same traffic.
Some important aspects (as, for example a recovery mechanism in case of fault) are still under study.

REFERENCES

[Giordano,2000] S. Giordano, J.-Y. Le Boudec: **"On a Class of Time Varying Shapers with Application to the Renegotiable Variable Bit Rate Service"** , Journal on High Speed Networks, 2000.

[Giordano,1999] S. Giordano, J.-Y. Le Boudec **"The Renegotiable Variable Bit Rate Service" ,** IFIP99, 1999.

[Giordano,1998] S. Giordano, J.-Y. Le Boudec **"QoS based Integration of IP and ATM: Resource Renegotiation"** , in Proceedings of 13th IEEE Computer Communications Workshop, 1998.

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

[ISO-1,1998] Information technology, Generic coding of moving pictures and associated audio information, Part 1: System International Standard Organisation, 1988.

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

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

[Wroclawski-2210,1997] J. Wroclawski RFC2210: The Use of RSVP with IETF Integrated Services IETF 1997.

[Wroclawski-2211,1997] J. Wroclawski RFC2211: Specification of Controlled-Load Network Element Service IETF, 1997.

[Shenker-2216,1997] S. Shenker, J. Wroclawski RFC2216: Network Element Service Specification Template IETF, 1997.

[Bernet,2000] Y. Bernet] The Complementary Roles of RSVP and Differentiated Services in the Full-Service QoS Network, Communication Magazine, February 2000.

[LeBoudec,1997] J.-Y. Le Boudec Network Calculus, Deterministic Effective Bandwidth, VBR trunks - IEEE Globecom 97 November.

[LeBoudec,2000] J-Y. Le Boudec, P. Thiran,"A short tutorial on network calculus I: fundamental bounds in communication networks", Proceedings of ISCAS 2000, Geneva, May 2000.

[Thiran,2000] J-Y. Le Boudec, P. Thiran, S. Giordano,"A short tutorial on network calculus II: min-plus system theory applied to communication networks", Proceedings of ISCAS 2000, Geneve, May 2000.

[ITU,1998] ITU-T Recommandation Q.2963.2. : Broadband integrated services digital network (B-ISDN) digital subscriber signalling system No. 2 (DSS 2) connection modification - Modification procedure for sustainable cell rate parameters, ITU Telecommunication Standardization Sector - Study group 13, 1998.

[DeSousa,1999] Paulo De Sousa, Global Communication Newsletter, Communication Magazine, 1999.