DMIF based QoS Management for MPEG-4 Multimedia Streaming: ATM and RSVP/IP Case Studies

Victor Marques[1], Ricardo Cadime[2], Amaro de Sousa[1], A. M. Oliveira Duarte[1]

[1] Instituto de Telecomunicações, Universidade de Aveiro, P-3810-193 Aveiro, Portugal

[2] Portugal Telecom Inovação, P-3810 Aveiro, Portugal

Abstract

This paper addresses the issue of QoS management under the framework of Delivery Multimedia Integration Framework (DMIF) protocol stack for the support of MPEG-4 multimedia streaming. In this paper, we describe how QoS was dealt in the implementation of a DMIF protocol stack over two different network technologies: IP networks with RSVP signaling and Asynchronous Transfer Mode (ATM) with Q.2931 signaling. In the RSVP/IP case, we address a strategy where the Token Bucket model is used to declare the traffic source as a variable bit rate source. In this case, we explore the fact that the network can support the delivery of variable bit rate data. In the ATM case, we address a strategy based on CBR connections where data bit rate variations are accommodated in the sending host. Particular attention is given to a video-on-demand service scenario where a server delivers MPEG-4 multimedia objects, each one composed by a video stream and an audio stream. Two key aspects are investigated. The first aspect is the mapping strategy of QoS DMIF parameters into the two different network QoS parameters. We propose two mapping methods that require pre-processing of stored media streams. The second aspect is the influence of multiplexing and packing streams in the efficiency of the mapping methods. The mapping strategies were tested on multimedia objects composed by H.263 video streams and G.723 audio streams.

I. INTRODUCTION

Networked multimedia applications are not widely commercially available because it is difficult to combine the knowledge of multimedia technologies and network technologies in the same experts. In the past, multimedia development experts were used to develop applications for stand-alone hardware platforms. More recently, with the advent of World Wide Web, these experts started developing services to be provided through "best effort" IP networks for remote retrieval of data. With the new developments on networks that support QoS, the use of network services is no longer simple. Multimedia application developers are reluctant to develop applications for a particular network technology because they do not know if this investment will have return in the future.

The support of application level QoS requirements through QoS enabled networks has been an investigation topic in the recent past under different frameworks [1-4]. One key aspect is how to map the application level QoS into network level QoS parameters. This topic was firstly addressed for ATM networks [5-6] and more recently for the IP based QoS enabled protocols [7]. The definition of an Application Programming Interface (API) that separates the application role from the delivery technology role benefits the market of networked multimedia applications. With such API, application developers can begin to invest in commercial multimedia applications with the assurance that the investment will not be made obsolete by new delivery technologies. The work carried on in ISO/IEC 14496 (MPEG-4) goes in this direction [8-10].

MPEG-4 defines a generic layered model, comprising a Compression Layer, a Sync Layer and a Delivery Layer. The Compression Layer processes individual audio-visual media streams without taking into account the delivery technologies. The MPEG-4 compression methods, defined in ISO/IEC specifications 14496-2 [11] and 14496-3 [12], achieve efficient encoding over a wide range from Kbps to multiple Mbps. The media content at this layer is organized in Elementary Streams that are composed of several Access Units (AU). An AU is the smallest piece of information possible. Each audio-visual object is composed by one or more associated Elementary Streams, each one with its own QoS requirements. The MPEG-4 Systems specification, ISO/IEC 14496-1 [13] defines the concepts needed to describe the relations between Elementary Streams in a way that allows the creation of distributed, yet integrated, content presentations and to synchronize the streams. This part of the specification, which defines the Sync Layer, is both media unaware and delivery technology unaware. The Delivery Layer in MPEG-4 is called Delivery Multimedia Integration Framework (DMIF) and is defined in ISO/IEC 14496-6 [14-15]. This layer is media unaware but delivery technology aware. It provides transparent access to the delivery of content irrespective of the technologies used through the interface between the Sync Layer and DMIF, called DMIF Application Interface (DAI). DAI represents the API that aims to fulfill the requirements described above.

Thus, DMIF is a general application and transport delivery framework aiming to hide the details of the transport network from the application, as well as to ensure signaling and transport interoperability between end systems. Each audio-visual object might have one or more associated Elementary Streams, each one with its own QoS requirements. It is the role of DMIF to set up the appropriate connections (in the underlying network technology) for the Elementary Streams that are requested from above through the DAI. MPEG-4 also defines a multiplexing tool, named FlexMux, that can be used to aggregate different Elementary Streams into the same network connection.

This paper refers to an implementation of a DMIF protocol stack over two different network technologies: IP networks with RSVP signaling and ATM with Q.2931 signaling. In this paper, we describe how QoS at DMIF level is mapped over the QoS parameters that are specific of each network technology. The DMIF over RSVP/IP protocol stack was developed for Windows 2000 operating system using WinSock 2 and its QoS SP supporting RSVP. The DMIF over ATM protocol stack was developed for Windows NT 4.0 operating system using Winsock 2 and the FORE Systems ATM Specific Extension SP. Particular attention is given to a video-on-demand service scenario where a server delivers MPEG-4 multimedia objects, each one composed by a video stream and an audio stream.

The paper is structured as follows. Section II presents the RSVP and ATM (using Constant Bit Rate – CBR – Connections) models and the relevant parameters that must be calculated. Section III shows how QoS is defined in the DAI level and presents the major constrains to map these parameters into RSVP and ATM parameters. Section IV shows the results of the methods for some sample streams.

II. RSVP AND CBR ATM MODELS

We have used the two different network technologies to implement two different QoS management strategies. In the RSVP/IP case, we address a strategy where the Token Bucket model is used to declare the traffic source as a variable bit rate source. In this case, we explore the fact that the network can support the delivery of variable bit rate data. In the ATM case, we address a strategy based on CBR connections where bit rate variations of data generation are accommodated in the sending host.

In the RSVP/IP implementation, the key aspect investigated is the role of sender shaping process (based on Token Bucket) and how a correct mapping can minimize network resources utilization without introducing shaping delays. In the ATM implementation, the key aspect investigated is on the determination of buffering size that can minimize the network resource utilization even though assuring the required maximum delay variation. Both aspects are particularly critical for video streams that have a bursty behavior.

Exploiting the fact that in video-on-demand scenarios the streams are stored on disk, we propose a method for QoS mapping that processes in advance the appropriate rate Token Bucket parameters (RSVP) and CBR Rate and Buffer Size (ATM) for each media stream.

A. The RSVP Case

RSVP is a signaling protocol used in IP networks that enable the host to characterize its data sources and request QoS for each individual source. The source characterization is based on Token Bucket concept and the QoS is based on maximum loss probability and maximum delay variation of each IP packet. The Token Bucket Model, used by RSVP is shown in figure 1. In order to optimize the network resources, the size of the Token Bucket and the Token Bucket rate must be optimized to comply with the application QoS demand and simultaneously using the minimum network resources. It is possible to calculate the values of b and r in order to guaranty that no delay [7] is introduced at the traffic shaper. Therefore, all the delay will be introduced by the network and it is given by the expression in equation (1)[21].

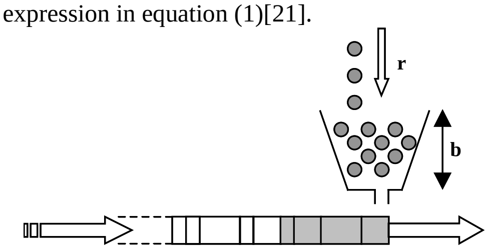

Fig. 1 Token Bucket model

$$Delay_{\max} = \frac{b + C_{tot}}{r} + D_{tot} \cong \frac{b}{r} \qquad (1)$$

B. The CBR ATM Case

ATM is a network technology capable of providing point-to-point connections with assured QoS parameters. It supports different connection types depending on the application interest. In the present work, we have addressed the QoS management issue based on the use of Constant Bit Rate (CBR) type of connections even for the cases where traffic sources (Elementary Streams) have variable bit rates. We choose CBR connections for two main reasons: (1) the SPI used to develop the ATM drivers only allows CBR or UBR type of service; (2) VBR, the most adequate type of service to transmit bursty contents like video, failed to be a commercially available type of service. In the CBR ATM case considering that the network introduces a constant delay to all cells, the delay variation will be given by the buffer size at the sender. In this case, a CBR Rate and a buffer size must be chosen in order to accommodate the bit rate variations of data generation in the sending host (figure 2). Thus, the maximum delay variation will be given by equation (2).

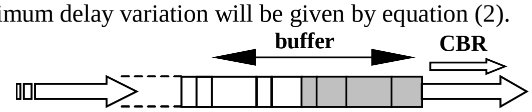

Fig. 2 CBR model

$$\text{Delay} = \frac{\text{buffer}}{\text{CBR\_Rate}} \qquad (2)$$

III. DAI LEVEL QOS PARAMETERS

The complete set of DAI level QoS parameters defined in the standard [14-15] is presented in Table I. These parameters can be conceptually divided in two sets: the flow descriptors (MAX_AU_SIZE, AVG_BITRATE and MAX_BITRATE) and QoS descriptors (all other parameters).

Table I
QoS specification at DAI level

DAI Parameters	Description
MAX_AU_SIZE	Maximum size of an individual AU
AVG_BITRATE	Average bit rate
MAX_BITRATE	Maximum bit rate
MAX_DELAY	Maximum delay per AU (µs) measured over 1 sec.
AVG_DELAY	Average delay of an AU (µs) measured over 1 sec.
LOSS_PROB	Loss probability of an AU (0-1) over 1 sec.
Dejitter Buffer	Bytes reserved for jitter removal

A major concern in DAI parameters is the lack of burstiness information of the stream. For example, the two streams shown in figure 3 have the same average and peak rates. However, the second stream generates traffic above average rate during a longer period of time. In terms of RSVP this means that the token bucket descriptors must consider a higher token bucket size to accommodate this burst. In terms of ATM, the CBR chosen must be such that that the delay introduced in the queue while the data is not sent (during bursts) is lower than the maximum allowed by the application. In this figure, *T* is the maximum time interval of a data burst.

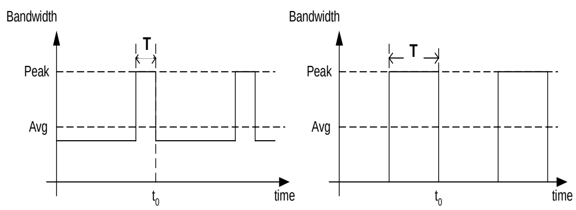

Fig. 3 Streams with the same peak and average rates

For the correct mapping into RSVP parameters and simultaneously to achieve high network resources utilization, the specification of maximum and average rates is not enough since a correct token bucket parameter set is only possible if we know how long the source will be generating packets above average rate. In the ATM case, and considering the use of a CBR connection, we will have a similar problem: to achieve high resources utilization, we need to use the lowest CBR that still complies with the QoS characteristics of the flows at the destination. Our approach is to pre-process the relevant network parameters and store them in a DMIF stack database. Whenever, the application requests connections for some Elementary Streams, their ES ID is used to retrieve from the database the appropriate network parameters (Token Bucket parameters in the RSVP case, or CBR capacity in the ATM case).

IV. CASE STUDY

Particular attention was given to a multimedia streaming provision scenario where a server delivers multimedia objects, each one composed by two Elementary streams: a video stream and an audio stream. The results were taken based on H.263 video streams and G.723 audio streams. We have studied the two possible multiplexing strategies: joint transmission of multiplexed audio and video through a single network connection and separate transmission of each stream in different network connections. Each stream is composed by a sequence of AUs (Access Units). We have also considered two other options: sending each AU individually or joining AUs in groups (packing) that are sent in the same network PDU. In pre-recorded multimedia contents, this option has no performance impact because the packed groups of AUs can be already stored on disk. In our implementation, all options were computed in advance and stored on disk. For the packetized streams, a maximum PDU of 1500 bytes was considered. We have used two multimedia contents, here named A and B with different characteristics.

A. Streams Characterization

Table II shows the characteristics in terms of maximum and average bit rate of the streams used. The FlexMux Streams refer to the Audio and Video streams multiplexed in one single stream. Unpacketized streams refer to the characteristics of data packets without packing and Packetized streams refer to the characteristics of data packets with packing.

Table II
Streams Characterization (kbps)

	Unpacketized		Packetized	
	Max Rate	Avg Rate	Max Rate	Avg Rate
H.263 Video (A)	254	111	270	111
G.723 Audio (A)	10	7.2	13	7.2
FlexMux Stream (A)	258	118	283	118
H.263 Video (B)	65	34	98	34
G.723 Audio (B)	11	8.3	18	8.3
FlexMux Stream (B)	66	44	99	44

Table III shows the UDP/IP overhead effect and Table IV shows the AAL5/ATM on each of the previous streams. In the UDP/IP case, this overhead is calculated considering the 28 bytes header for each packet sent. In the ATM case, the overhead is given by the following rule: 8 bytes of AAL5 overhead plus the padding in the last ATM cell to fulfill the 48 bytes. The 5 bytes of each ATM cell header are not taken into account in this table.

Table III
Streams Characterization with UDP/IP overhead (kbps)

	Unpacketized		Packetized	
	Max Rate	Avg Rate	Max Rate	Avg Rate
H.263 Video (A)	283	124	277	114
G.723 Audio (A)	21	14.7	17	8.7
FlexMux Stream (A)	293	139	291	122
H.263 Video (B)	72	39	101	36
G.723 Audio (B)	20	15.7	17	10
FlexMux Stream (B)	79	56	103	45

Table IV
Streams Characterization with AAL5/ATM effects (kbps)

	Unpacketized		Packetized	
	Max Rate	Avg Rate	Max Rate	Avg Rate
H.263 Video (A)	287	126	276	113
G.723 Audio (A)	18	12.8	15	8.2
FlexMux Stream (A)	296	138	293	122
H.263 Video (B)	73	39	102	36
G.723 Audio (B)	16	12.8	18	10.2
FlexMux Stream (B)	76	53	102	45

B. RSVP Requisites

The system shown in figure 1 was simulated [7] and the minimum value of Token Bucket Rate and the value of the Token Bucket Size were achieved in order to comply with the maximum delay (400ms) calculated according to the equation (1). The results were calculated for both Unpacketized and Packetized streams and are presented in Tables V and VI.

Table V
RSVP Unpacketized Streams Requisites

	Token Bucket Rate (kbps)	Token Bucket Size (bits)	Max Network Delay (ms)
H.263 Video (A)	164	63560	387
G.723 Audio (A)	15	3313	220
FlexMux Stream (A)	177	67955	383
H.263 Video (B)	55	19159	348
G.723 Audio (B)	16	3464	216
FlexMux Stream (B)	68	13404	197

Table VI
RSVP Packetized Streams Requisites

	Token Bucket Rate (kbps)	Token Bucket Size (bits)	Max Network Delay (ms)
H.263 Video (A)	151	58886	389
G.723 Audio (A)	9	2701	300
FlexMux Stream (A)	158	63038	398
H.263 Video (B)	51	19699	386
G.723 Audio (B)	10	3104	310
FlexMux Stream (B)	58	22350	385

C. ATM CBR Requisites

The system shown in figure 2 was also simulated and the minimum value of buffer size and the value of the CBR Rate were achieved in order to comply with the maximum delay (400ms) calculated according to the equation (2). The results were calculated for both UnPacketized and Packetized streams and are presented in Tables VII and VIII.

Table VII
UnPacketized Streams Requisites with AAL5/ATM effects

	CBR (kbps)	Buffer (bits)	Delay (ms)
H.263 Video (A)	166	63246	381
G.723 Audio (A)	13	2899	223
FlexMux Stream (A)	177	69384	392
H.263 Video (B)	56	16408	293
G.723 Audio (B)	13	2795	215
FlexMux Stream (B)	66	4818	73

Table VIII
Packetized Streams Requisites with ATM and ALL5 effects

	CBR (kbps)	Buffer (bits)	Delay (ms)
H.263 Video (A)	150	58800	392
G.723 Audio (A)	9	2691	299
FlexMux Stream (A)	160	61280	383
H.263 Video (B)	52	20020	385
G.723 Audio (B)	11	2816	256
FlexMux Stream (B)	59	16815	285

D. Results Discussion

The results of Tables II and III when compared with table I show that the overhead introduced by the two technologies is quite significant, particularly in the case of Unpacketized Streams. This effect is determinant on all sub-sequent results since the Token Rates for RSVP and the CBR rates for the ATM are always higher for these streams. Therefore, one of the main conclusions is that packing streams minimizes network overhead and should be used for streaming transmission.

Note that the Token Bucket rate is the bandwidth reservation in the RSVP case. Comparing this parameter with the CBR rate, we see that these values are quite similar for all considered streams. In fact, this was expected since equation (1) that specifies the maximum delay variation for the RSVP case is similar to equation (2) for the ATM case. Therefore, in terms of network resource utilization, both technologies are similar. However, using ATM with CBR connections requires buffering capabilities on server side, which can be the most limiting factor for the total number of clients that a single server can be serving at the same time.

Concerning the comparison between multiplexing the Video and Audio in a single FlexMux stream, or sending them separately, the results show that in terms of network resources, the results are similar. There are small gains between 2 and 3 kbps when audio and video are sent in the same FlexMux both for the RSVP [23] and ATM case.

V. CONCLUSIONS

DMIF defines an API (called DMIF Application Interface) that can be used by applications to request network connections for Elementary Streams. DAI defines a unique

QoS parameter set, which is independent of the network technology that is below DMIF. Besides the QoS parameter set, MPEG-4 defines a tool (called FlexMux) that can be used by applications to multiplex different Elementary Streams into the same network connection. However, the standard does not specify how to map DAI level QoS parameters into each existing network QoS parameters or how to decide when different Elementary Streams should be multiplexed in the same network connection (using a FlexMux). This paper has described an implementation of the DMIF protocol stack over two different network technologies (IP networks with RSVP signaling and ATM with Q.2931 signaling) where these issues were addressed.

We first discussed how to map DAI level QoS parameters into network level parameters. We showed that the current DAI QoS parameter set is not enough to calculate the appropriate network parameters that fulfill delay requirements. Then, exploiting the fact that in video-on-demand scenarios the media streams are stored on disk, appropriate mapping methods were proposed that require pre-processing of stored media streams. With the pre-processed values, a file-based database is built and used in our DMIF protocol stack implementation. With this approach, we developed a protocol stack that is able to set-up appropriate RSVP reservations (in RSVP) or CBR connections (in ATM) without making changes to the standardized DAI implementation.

We then studied the influence of multiplexing and packing streams on network resource utilization. The packing process is feasible in a video-on-demand scenario since the streams can be stored already packed. Multiplexing, alone, does not result in better network resources utilization, but minimize the number of required connections. When both packing and multiplexing are used, we were able to obtain significant network resources utilization gains

VI. REFERENCES

[1] C. Aurrecoechea, A. T. Campbell, and L- Hauw, "A Survey of QoS Architectures" (invited paper), *Proc. 4th IFIP International Workshop on Quality of Service (IWQoS'96)*, Paris, Mar. 1996

[2] D.G.Waddington, G.Coulson, and D.Hutchison, "Specifying QoS for Multimedia Communications within Distributed Programming Environments", *Proc. of the 3 rd International COST237 Workshop*. LNCS 1185, Barcelona, Nov. 1996

[3] J. F. Huard, and A. Lazar, "A Programmable Transport Architecture with QoS Guarantees", IEEE Communications Magazine, 36, pp 54-62, Oct 1998

[4] O. Verscheure, X. Garcia, G. Karlsson and J.P. Hubaux, "User-Oriented QoS in Packet Video Delivery", IEEE Network, 12, pp 12-21, Nov./Dec. 1998

[5] J. F. Huard, and A. Lazar, "On QoS Mapping in Multimedia Networks", *21th IEEE Annual International Computer Software and Application Conference (COMPSAC'97)*, Washington D.C., Aug. 1997

[6] S. Gringeri, K. Shuaib, R. Egorov, A. Lewis, B. Khasnabish, and B. Basch, "Traffic shaping, bandwidth allocation, and Quality Assessment for MPEG Video Distribution over Broadband Networks", IEEE Network, 12, pp 94-107, Nov./Dec. 1998

[7] Puqi P. Tang, and Tsung-Yuan C. Tai, "Network Traffic Characterization Using Token Bucket Model", *Proc. IEEE INFOCOM'99*, 1, pp. 51-62, New York, Apr. 1999

[8] Rob Koenen (Ed.), ISO/IEC JTC1/SC29/WG11 N3156, "MPEG-4 Overview – (Maui version)", Dec. 1999

[9] B. Haskell, A. Puri, and L. Rabiner, "Image and video coding – Emerging Standards and Beyond", *IEEE Transactions on Circuits and Systems for Video Technology*, 7, pp 814-837, Nov. 1998

[10] T. Sikora, "MPEG Digital Video-Coding Standards", IEEE Signal Processing Magazine, 14, pp 82-100, Sept. 1997

[11] ISO/IEC 14496-2, "Information technology – Coding of audio-visual objects: Video", International Standard, 1999

[12] ISO/IEC 14496-3, "Information technology – Coding of audio-visual objects: Audio", International Standard, 1999

[13] ISO/IEC 14496-1, "Information technology – Coding of audio-visual objects – Part 1: Systems", International Standard, 1999

[14] ISO/IEC 14496-6, "Information technology – Very-low bitrate audio-visual coding – Part 6: Delivery Multimedia Integration Framework (DMIF)", 1999

[15] ISO/IEC 14496-6, "Information technology – Very-low bitrate audio-visual coding – Part 6: Delivery Multimedia Integration Framework (DMIF)", Amendment 1: Extensions for Version 2, 1999

[16] N. Delgado, G. Franceschini, A. de Sousa and A. Oliveira Duarte, "An MPEG-4 DMIF Implementation over IP Networks", *7th IFIP/ICCC Conference on Information Networks and Data Communications (INDC'98)*, pp 281-290, Jun. 1998

[17] J. F. Huard and G. Tselikis, "An Overview of the Delivery Multimedia Integrated Framework for Broadband Networks", IEEE Communications Surveys, 2, pp 34-43, Fouth Quarter 1999

[18] J.F. Huard, A. Lazar, K.S. Lim, and G. Tselikis, "Realizing the MPEG-4 Multimedia Delivery Framework", IEEE Network, 12. pp 35-45, Nov./Dec. 1998

[19] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin, "Resource reSerVation Protocol (RSVP) – Version 1 Functional Specification", RFC 2205, Sept. 1997

[20] J. Wroclawski, "The use of RSVP with IETF Integrated Services", RFC 2210, Sept. 1997

[21] S. Shenker, C. Partridge, and R. Guerin, "Specification of Guaranteed Quality of Service", RFC 2212, Sept. 1997

[22] J. Wroclawski, "Specification of the Controlled-Load Network Element Service", RFC 2211, Sept. 1997

[23] Ricardo Cadime, Amaro de Sousa, A. M. Oliveira Duarte, "A DMIF over RSVP/IP Protocol Stack: Experimental Results for MPEG-4 Multimedia Streaming", SPIE Information Technologies 2000, Photonics East, Nov. 2000