(m,k)-frame: A Method to Control the Quality of Service in Distributed Multimedia Systems

Bechir Alaya, Claude Duvallet, Bruno Sadeg
LITIS, UFR des Sciences et Techniques
25 rue Philippe Lebon, BP 540
F-76058 Le Havre Cedex
Firstname.Lastname@univ-lehavre.fr

Abstract

Multimedia applications usually manage large quantities of data in the form of frames of certain types. To ensure a good trafic of these frames through the network, temporal constraints must be respected when sending and receiving these frames. If the temporal constraints are not met, then the quality of service (QoS) provided to users decreases. In this paper, we exploit some results obtained in QoS management in Real-Time Database Systems (RTDBSs), and we apply them to multimedia applications, because of similarities existing between these two fields. We propose a new method allowing to control the QoS provided to clients according to the network congestion, by discarding some frames when needed.

1 Introduction

The recent improvements in networks area allow to consider the large exploitation of new services in many applications, particularly in multimedia applications. These applications deal with large volumes of data and require real-time processing, i.e., they must be completed before fixed dates, to guarantee an acceptable quality of service (QoS) in the streams presented to the users. Systems adapted to the management of these kinds of data with QoS guarantees are real-time database systems (RTDBSs) [13][14].

Many distributed multimedia applications must face to the unpredictable loads that cause the system overload. For example, user-demands may arrive in a bursty manner during a short period. Currently, all applications need to provide a good QoS to the users (a good flow of video frames). To this purpose, it will be interesting to adapt the existing techniques to multimedia applications in order to obtain more reliable and efficient transfer of the video packets, without modifying the initial infrastructure. The main problems are related to the adaptation of available resources (bandwidth, buffer size, video servers, etc.) and to the proposition of new techniques which deal with system instability periods (overload or under-utilization). The proposition must allow to ensure an acceptable QoS while respecting the multiple requirements of the video streams.

Many works on QoS management in RTDBSs have been done [1][10]. Almost all these works are based on a feedback control scheduling architecture (FCSA) that controls the system behavior thanks to a feedback loop.

The feedback loop begins to measure the performances of the system in order to detect overload periods. Then, according to the results observed, the values of the parameters are modified to adjust the system load to the real conditions. As these conditions always vary, this process is repeated indefinitely.

Because of the similarities existing between RTDBSs and multimedia applications [6], in this paper, we propose to apply the results obtained on the QoS management in RTDBSs to multimedia applications. The main objective is to allow to design multimedia applications that will be able to provide the QoS guarantees and a certain robustness when user's demands quickly grow up or when the network becomes congested. These works are especially applied to video-on-demand (VoD) applications.

In this article, we present a method allowing to take into account the network congestion in order to increase the QoS provided to the users, especially, how to achieve an optimal value of frames in a MPEG stream [6][12]. In Section 2, we present the multimedia system architecture that we use. In Section 3, we describe our approach which allows to increase the applications QoS during the overload periods (network congestion). Then, in Section 4, we present the simulations results that we have done to test the validity of our approach. Finally, we conclude this article and we give some perspectives.

2 A quality of service approach

2.1 Quality of service in distributed multimedia systems

The commonly used definition of the QoS in a multimedia application is the whole of requirements in terms

of bandwidth, quality of visualization, delay and rate of video packets loss. Our approach consists in taking into account researches already done on the management of QoS in RTDBSs [11][2] and their adaptation to multimedia systems. To this purpose, we propose an adaptation of a method based on feedback control architecture to distributed multimedia systems [6]. We exploit, to this purpose, the notion of (m,k)-firm constraints used in real-time sytems [5] and in RTDBSs [8][3].

This adapted method is called FCA-DMS (Feedback Control Architecture for Distributed Multimedia Systems). We will apply a control of the network congestion by discarding or not some multimedia frames of certain types according to the network state, notably to the shared bandwidth. This will increase the QoS provided to the users.

2.2 Feedback control architecture

In a previous work, Natalia Dulgheru has proposed an architecture, named QMPEGv2 [6] which deals with distributed multimedia systems (cf. Figure 1). The architecture proposed contains three main components:

- ***Master server:*** it accepts requests from clients, chooses the video servers able to serve the demand, supervises the system state and adjusts the video streams in order to maintain the QoS initially fixed.
- ***Video servers:*** They run under the master server control and send the video packets to the clients.
- ***Clients:*** they send requests to the master server and receive the video frames from the video server. When a state change occurs, they send a feedback report to the master server.

In the following, we describe briefly a typical procedure which is executed when a video on demand is requested, based on FCA-DMS architecture:

1. A client sends a request to the master server to get a video, with a certain level of QoS.
2. The master server broadcasts the request to the video servers available in the system.
3. The video servers send back their response to the master server, which chooses one among them.
4. A stream is opened between the chosen video server and the concerned client.
5. The master server asks the video servers to adapt their QoS, when necessary.

The feedback loop consists on adapting the QoS according to the load system conditions (servers and network congestion). The system observes the QoS obtained

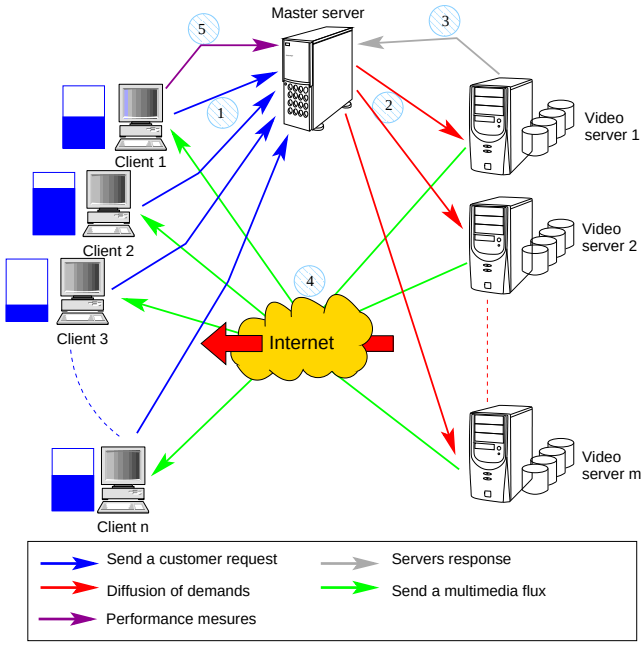


Figure 1: Functional model of the FCA-DMS architecture.

by the client and, if necessary, asks the concerned video server to improve it.

In order to improve the QoS, the system increases or decreases the number of transmitted frames of certain types. To this purpose, we based our action on the characteristics of the standard MPEG format [9], that defines a mechanism to code frames at the time of the video compression. A video sequence enters the system. It's then compressed and coded according to three types of frames: *Intra frames* (I), *Predicted frames* (P) and *Bidirectional frames* (B). *I* frames are references frames. *P* frames allow to rebuild a frame using an *I* frame. *B* frames use *I* frames and *P* frames to rebuild a sequence. Therefore, I frames are the most critical. To decrease the eventual network congestion, it is necessary to remove some frames from a video sequence, but these suppressions must be done in a controlled manner. We propose in the following section a method based on the controlled frames suppression in order to control the QoS provided to users.

2.3 Feedback control loop

Using the feedback loop allows to stabilize the system during the instability periods [4]. It is based on the two principles: observation and auto-adaptation. The principle consists of observing the results obtained by the system and checking if the current QoS observed is consistent with the QoS initially required, e.g. in VoD application, the system checks if the videos sequences

are presented to users without interruptions. The auto-adaptation consists for the system to adapt the results according to the QoS needed par the clients, by adjusting some network and video parameters, e.g. the system increases or decreases the number of accepted frames[1]. In this way, the feedback loop ensures the stability of the system.

3 A new method to control the network congestion

3.1 The (m,k)-frame method

According to certain conditions, the system load varies from overload state to under-utilization state and vice-versa. Indeed, since the number of video servers sending the video packets is unknown, sometimes this causes severe damages on the service level provided to clients. Consequently, the number of transmitted packets is also unknown and can be important. Moreover, when a high number of video packets access to network resources, it is necessary to keep a high priority level for more critical packets (I frames, then B frames, then P frames) [6][12].

We propose an approach based on (m,k)-firm method [7]. The (m,k)-firm model is characterized by two parameters m and k. An application is said under (m,k)-firm real-time constraint if at least m operations among k consecutive operations meet their deadlines. We adapt this method to the context of multimedia applications. An multimedia operation consists of sending/receiving a video frame. To adapt this method, we consider that m video packets among k must be correctly sent. To this purpose, we propose a new technique of video packets management crossing the network, called (m,k)-frame.

A video stream is decomposed into several classes according to their tolerance to the loss of frames characteristics, i.e. each class contains the video packets of similar (m,k)-frames constraints. The three classes, we consider, refer to the three types of frames: I, B and P. With this technique we realize a trade off between the shared resources and the QoS granularity in the same class of a video stream.

3.2 The quality of service adaptation

In this work, we focus on the adaptation of the video stream to the network state. We assume that measures of the network capacity are available, in one hand, and that we have an important number of frames to send, on the other hand.

The three classes of frames (I, B and P) are used to adapt the quality of stream sent to the network capacity. We consider the following constraints: (m_I, k_I)-frame, (m_P, k_P)-frame and (m_B, k_B)-frame, i.e, m_i frames of a certain type must be received among the k_i frames sent. Then the network capacity is measured by the formula: $m_I + m_P + m_B$. Recall that I frames are the most critical. The parameters are ordered in the following manner: $m_I > m_P > m_B$. We usually have $m_I = k_I$, i.e., I frames are critical and it is forbidden to remove them.

We assume the situation where the network, whose current capacity is N, is congested. We also assume that QoS_{max} is the quality of the stream to send including M frames. To be consistent with the network capacity, it is necessary to remove $(M - N)$ frames. Therefore, we have to degrade the quality of the MPEG stream. When we apply no method of congestion control, frames will be randomly removed, i.e. they are lost by the network, causing the degradation of the video presentation, notably if some I frames are removed. Here, we apply our (m,k)-frame method, which consists of removing frames in an intelligent manner. We have: (1) M=$k_I + k_P + k_B$, and (2) N=$m_I + m_P + m_B$. The number of frames to remove is then: $M - N = (k_I - m_I) + (k_P - m_P) + (k_B - m_B)$, where $k_I = m_I$ (I frame are the most critical, and are not to remove).

3.3 Bandwidth fair sharing

With the previous assumptions, we deal with the broblem of sharing bandwidth between servers in case of network congestion phases. In the previous section, we have seen how to reduce the QoS at the stream level, according to the available capacity of the network. Here, we need to share fairly the bandwidth between all sources that wish to send a stream. We compute the total capacity needed by all servers. Then, we compute R, the ratio between the needed capacity and the available network capacity (N).

$$R = \frac{N}{\sum_{i=1}^{m} RC_i}$$

such as :

- m : the number of video server.
- RC_i : The Required Capacity of the video server i.

Example: let 3 video servers wishing to send flows of 40, 30 and 20 frames per second respectively. The total capacity of the network needed to answer to this demand must be 40+30+20=90. If, however, the network only arranges a capacity of 75 frames per seconds, it is not able to sent all the frames. The ratio R is computed as follows: (75/90)*100 = 83.33%. Then, we apply this rate

[1] note that I frames (critical) are not removed

to each of the three required capacities 40*83.33%=33, 30*83.33%=25 and 20*83.33%=17. If we sum the three obtained numbers, we find 75 frames per second. This corresponds to the actual capacity of the network.

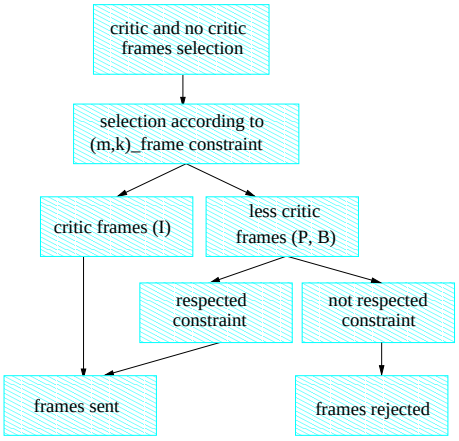


Figure 2: The (m,k)-frame algorithm.

In the following, are listed some advantages of the bandwidth fair sharing:

- To control of the congestion of the network by fair sharing resources between all streams. A bad stream doesn't affect the service provided to the other streams. Only this service will be concerned if a stream wants to consume more resources than available.
- To guarantee an acceptable bandwidth and delay.
- To guarantee a link sharing between the different classes of service.

4 Simulations and results

To test the validity of our proposition, we have developed and implemented a simulator. To verify the behavior of system and its adaptation to different load variation.

The simulator is based on the architecture components presented in Section 2. A master server serves all video servers participating to the video diffusion. Moreover, the master server allows to add new video servers, when necessary. A new server is assigned a number and a list of accessible objects. After starting the master server, the video servers who want to participate in the video distribution, refer themselves to the master server, and then get a number.

In order to make a request, a client dialogs with the master server, that distributes the request to the available video servers. When the master references a video server able to satisfy the client request, i.e. the server is able to provide the required QoS, it sends the video server reference to the client. Then, the broadcast from the video server can begin. After some time, the client sends to the server the QoS level he obtained.

The three parts of the architecture of this simulator are modeled thanks to an object modeling language and realized in JAVA language.

4.1 Simulations objectives

The objective of the simulations is to demonstrate how our method, called (m,k)-frame, is able to adapt the QoS to the real conditions of a multimedia application, according to the current system load. Notably, the system must adjust the QoS when the client number that carries out requests varies (dynamic arrived of clients). The network congestion can have different sources:

- *internal:* when there is a large number of clients doing requests in the system. We can limit this client number by using an admission controller located at the master server level.
- *external:* when the network is used by other applications that can cause the congestion.

Our architecture must adjust the QoS by reducing the number of frames broadcasts in the network.

4.2 Simulation results

We have tested two possibilities of frames removal in our simulator. So we have implemented two versions:

- *Version 1:* here, we assume that there exist dependencies between P frames and B frames. When a P frame is removed then all B frames that depend on this P frame must be removed because they become useless.
- *Version 2:* in this case, the removal of B frames doesn't depend on P frames. The removal of P and B frames is randomly done, i.e, the probability to remove a P frame is equivalent to that of removing a B frame because they are independent.

The results (see Figure 4) obtained in *version 1* are better than those obtained in *version 2*.

In Figures 4(a) and 4(d), we note that the loss rate of I frames is less important than other types of frames. We particularly distinguish the following cases:

1. when no frame management method is used: we note that in Figure 4(a), only ∼62% of I frames sent are received and usable. The other frames are lost. For P frames (Figure 4(b)) and B frames (Figure 4(c)),

Frames I for 9000 sent frames	Usables Frames	Lost Frames	Received Frames	Sent Frames
Without any management selection of the stream	67.73	32.27	67.73	100.00
According to the (m,k)−frames selection method (version 1)	100.00	0.00	100.00	100.00
According to the (m,k)−frames selection method (version 2)	100.00	0.00	100.00	100.00

(a) For I frames.

Frames P for 9000 sent frames	Usables Frames	Lost Frames	Received Frames	Sent Frames
Without any management selection of the stream	44,22	34.27	65.73	100.00
According to the (m,k)−frames selection method (version 1)	60.00	40.00	60.00	60.00
According to the (m,k)−frames selection method (version 2)	60.00	40.00	60.00	60.00

(b) For P frames.

Frames B for 9000 sent frames	Usables Frames	Lost Frames	Received Frames	Sent Frames
Without any management selection of the stream	32.98	33.12	66.88	100.00
According to the (m,k)−frames selection method (version 1)	45.57	35.00	65.00	65.00
According to the (m,k)−frames selection method (version 2)	45.90	54.10	45.90	45.90

(c) For B frames.

Frames I,P and B for 9000 sent frames	Usables Frames	Lost Frames	Received Frames	Sent Frames
Without any management selection of the stream	38.69	33.33	66.67	100.00
According to the (m,k)−frames selection method (version 1)	53.71	33.33	66.67	66.67
According to the (m,k)−frames selection method (version 2)	53.93	46.07	53.93	53.93

(d) For I, P and B frames.

Figure 3: The simulation results

almost the same rate of frames are received (62%), but the rate of used frames is only about 40%, because P and B might be removed randomly.
In Figure 4(d), when we consider all types of frames, we obtain approximatively the same results as in Figures 4(b) and 4(c).

2. when (m,k)-frames method is used: we note, in Figure 4(a), that all I frames sent are received without any loss, in the two versions(*V1* and *V2*), and they are all usable. In Figures 4(b), 4(c) and 4(d), we note that only m frames among k are sent. However, we note also that all frames sent, are received and used. Consequently, frames selection is made at the time where the stream enters the system.

Finally, we note, in Figure 4(d), the rate gap between I, P and B frames with and without (m,k)-frames model: there exists a clear difference between the three types of frames : the selective packets reject is very efficient to maintain an acceptable QoS to the client while minimizing the critical packets loss, i.e. I packets.

In case of network overload, the QoS degradation of the MPEG stream is acceptable with (m,k)-frame, since I frames are more significant than both B and P frames for the decoding process, to efficiently rebuild the original video.

5 Conclusion and futures works

In this work, we have proposed an adaptation of the feedback control architecture for distributed multimedia systems (FCA-DMS). Our objective is to provide a deterministic temporal guarantee according to the temporal constraints to real-time video streams. Our main contribution concerned the adaptation of (m,k)-firm constraints for the video packets and the establishment of a video distribution strategy.

A possible extension of this work consists on enhancing the architecture presented, notably, in order to bring some breakdowns tolerance because of the presence of only one master server. We also have presented the importance of the bandwidth sharing and given a certain priority to the video packets in the network resources level, in order to increase its reliability and robustness and to converge towards the QoS specified by the client.

The simulator design allowed to validate the feasability of our approach and should permit to provide results demonstrating the real contribution of this new approach.

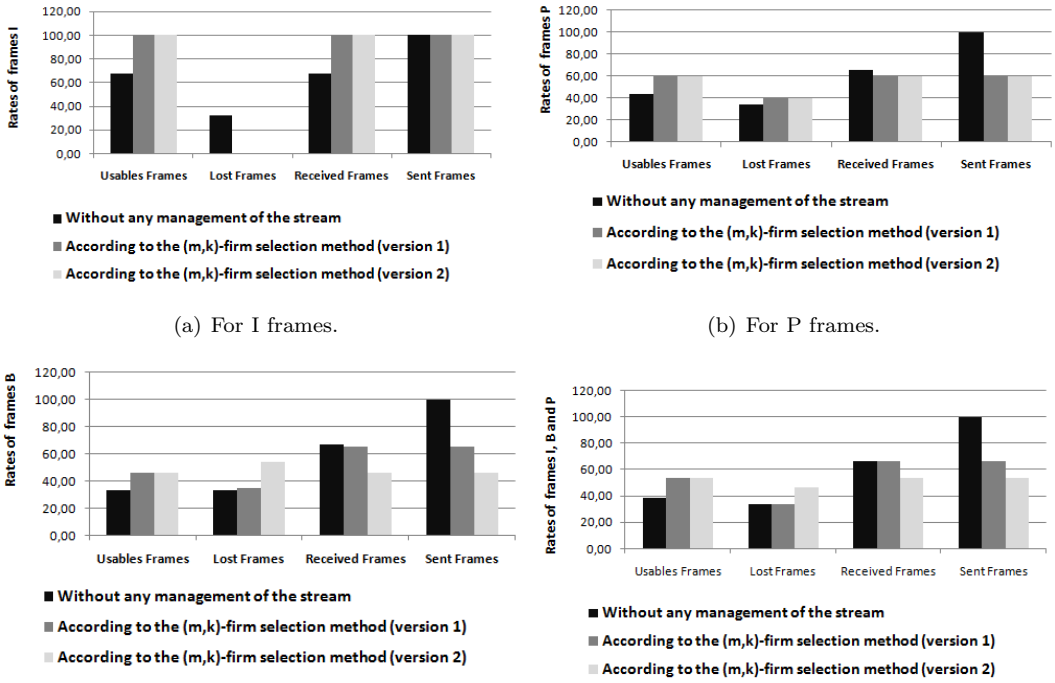
A possible future work would consist also of building a real video on demand server based on the architecture that we proposed.

References

[1] M. Amirijoo, J. Hansson, and S. H. Son. Specification and management of QoS in real-time databases supporting imprecise computations. *IEEE Transactions on Computers*, 55(3):304–319, 2006.

[2] M. Amirijoo, J. Hansson, and S. Song. Error-driven QoS imprecise management in imprecise real-time databases. In *Proc. of Euromicro Conf. on Real-Time Systems (ECRTS'03)*, pages 63–72, Portugal, 2003.

[3] S. Andler and J. Hansson, editors. *Active, real-Time and Temporal Database Systems (ARTDB'97)*,



(a) For I frames.

(b) For P frames.

(c) For B frames.

(d) For I, P and B frames.

Figure 4: Utilization of (m,k)-frames method.

Como, Italy, 1997. Proceedings of the Second International Workshop on Active, Real-Time and Temporal Database Systems, Springer.

[4] E. Bouazizi, C. Duvallet, and B. Sadeg. Management of QoS and data freshness in RTDBSs using feedback control scheduling and data versions. In *Proceedings of* 8^{th} *IEEE International Symposium on Object-oriented Real-time distributed Computing (ISORC'2005)*, pages 337–340, Seattle, United State, May 18-20 2005.

[5] G. Bucci, M. Campanai, and P. Nesi. Tools for specifying real-time systems. *Journal of Real-Time Systems*, 8:117–172, 1995.

[6] N. Dulgheru. Management of QoS in Distributed MPEG Video. Master's thesis, University of Linkoping, 2004.

[7] M. Hamdaoui and P. Ramanathan. A Dynamic Priority Assignment Technique for Streams with (m, k)-*Firm* Deadlines. *IEEE Transactions on Computers*, 44(4):1325–1337, 1995.

[8] J. Hansson and S. Son. Overload Management in RT-DBs. In *Real-Time Database Systems: Architecture and Techniques*, chapter 10, pages 125–140. Kluwer Academic Publishers, 2001.

[9] ISO/IEC 13818-2. Information Technology-Generic Coding of Moving Pictures and Associated Audio, Part 2: Video. Recommendation ITU H.262, International Standardization Organization, November 1994.

[10] K. Kang, S. Son, J. Stankovic, and T. Abdelzaher. A QoS-Sensitive Approach For Timeliness and Freshness Guarantees in Real-Time Databases. In *Proceedings of the Euromicro Conference on Real-Time System*, pages 203–212, 2002.

[11] K.-D. Kang, S. Son, and J. Stankovic. Service Differentiation in Real-Time Main Memory Databases. In 5^{th} *IEEE International Symposium on Object-Oriented Real-Time Distributed Computing (IEEE ISORC'02)*, pages 119–128, Washington D.C, 2002.

[12] J. Ng, K. Leung, and W. Wong. Quality of Service for MPEG Video in Human Perspective. Technical report, Hong Kong Baptist University., 2000.

[13] K. Ramamritham. Real-time databases. *Journal of Distributed and Parallel Databases*, 1(2):199–226, 1993.

[14] K. Ramamritham, S. Son, and L. DiPippo. Real-Time Databases and Data Services. *Real-Time Systems*, 28:179–215, 2004.