Real-Time Databases and Multimedia Systems

Quality of service management in distributed multimedia systems

Claude Duvallet

University of Havre
Faculty of Sciences and Technology
25 rue Philippe Lebon - BP 540
76058 LE HAVRE CEDEX, FRANCE
Claude.Duvallet@gmail.com
http://litis.univ-lehavre.fr/~duvallet/index-en.php

Who am I?

- Claude Duvallet
  - Associate Professor in computer science since september 2003.
  - PhD obtained in october 2001 at the University of Le Havre, France
- Where do I com from? University of Le Havre (France).
- My topics of interest:
  - Teachings: Programmation (Java, C/C++,...), Operating Systems (Linux, Unix), Distributed System (CORBA, RMI, RPC, EJB, LDAP, etc.), Network Protocols and Architecture, Network and System Administration.
  - Research: Real-Time Databases, Multimedia Systems, Quality of Service Management, Distributed Systems, etc.
- Current PhD supervising: Nizar Idoudi, Emna Bouazizi and Bechir Alaya.
- My homepage in english:
  `http://litis.univ-lehavre.fr/~duvallet/index-en.php`

Le Havre in France

Outline

- Introduction and context.
- Management of the quality of service in RTDB.
- Management of the quality of service in multimedia systems.
- Limitations of existing models.
- Contributions.
- Conclusion and futures works.

Introduction

- Computer software are designed to help users:
  - they are more and more sophisticated in terms of treatments and data.
  - some of them uses data with a validity period limited in time.
  - some of them must answer to the users requests before a deadline and using fresh data (recent data).
  - some of them must manipulate large amounts of data.

$\Rightarrow$ Due to the increasing of requests for real-time data, the use of real-time database becomes necessary.

- The charge of the systems is unpredictable and can quickly become very important (overload).

$\Rightarrow$ Missing deadline and use of non fresh data during these overload periods.

Multimedia applications

- They are more and more present in our daily lives:
  - They can be enriched information (image, sound, video) rather than just text.
  - They facilitate the understanding of the information.
- Example: video on demand system
  - A server provides multimedia contents (movies, etc.).
  - Customers make requests and view these contents in real-time without downloading them, but through a client (player) that supports the streaming video.
- Two class of applications:
  - Those type of conference: stream are sending both in the two directions with a certain interaction.
  - Those type of streaming: they deliver multimedia streams from a server to a client (such as RealPlayer).

Real-time problems

- The multimedia content which represent large amount of data.
- Real-Time constraints:
  - pieces of films that are transmitted through the network, which must be played in timeliness,
  - keep a certain fluidity of the streams,
  - avoid the problems of non synchronization between sound and video.
- Users demands are unpredictable in time.

$\Longrightarrow$ Some similarities with the management of the quality of service in real-time databases (RTDB).

Mains goals of our works

- Using a quality of service based approach to ensure the requirements on the behavior of systems even in case of overload.
- It is important for applications based on real-time treatment and where it is not possible to have an accurate analysis of arrivals and runtime.
- Our approach:
  - using imprecise computation: make a compromise on the resources necessary to get the quality of service required.
  - varying quality of video (QoV) in order to be closer to the requested quality of service.

  ⇒ Using dynamic load balancing algorithms.

Real-Time Database

- New generation of database.
- Efficient management of great quantities of data.
- Satisfaction of real-time constraints on data and on transaction.

Real-Time Data (RTD)

- They have a timestamp, an absolute validity interval and a value.

Real-Time Transactions

- Classification according to the operations:
  - update transactions (only write real-time data),
  - user transactions (read or write non real-time data, read real-time data).
- Classification according the deadline:
  - transaction with hard deadline,
  - transaction with firm deadline,
  - transaction with soft deadline.

Some definitions (1/2)

Non Real-Time Data:

- It is the classical data of the computer software.

Real-Time Data:

- This is data which have a limited validity time beyond which they become obsolete (not fresh). Their validity time most often corresponds to the frequency with which they are updated.

Some definitions (2/2)

Update Transaction:

- These are transactions (in general periodicals) which are updating the databases with the value read from sensors.
- They only write real-time data.

User transactions:

- These are the transactions made by users, i.e. the requests for information or operations on the data. They are characterized by both operations of reading and writing in the database.
- They read and write non real-time data and they read real-time data.

Quality of service management in Real-Time Databases

Definition of the quality of service

- Collective measure of the service level provided to the client. QoS can be characterized by different benchmarks base which include availability, the rate of errors, the response time, the rate of successful transactions that met their deadline, and so on.

Quality of service management in Real-Time Databases

Definition of the quality of service

- Collective measure of the service level provided to the client. QoS can be characterized by different benchmarks base which include availability, the rate of errors, the response time, the rate of successful transactions that met their deadline, and so on.

Quality of Data (QoD) et Quality of Transaction (QoT)

- QoD: It corresponds to the accuracy or error on the stored value (in database) compared to the current value in real world.
- QoT: It corresponds to the accuracy of the results provided by the transaction compared with an optimal precision which would be obtained in a complete calculation.

  $\Longrightarrow$ This is only true for user transactions.

Quality of Service based approach

- Definition of references parameters.
- Measure of the performances of the system (miss ratio for the transaction, quality of the transaction,...).
- Changing operating parameters in order to be closer to the reference values.

⇒ Using a feedback loop based on the principle of observation and adaptation.

- Goal: obtain more robust system that automatically adapt to the load conditions of the system.

The parameters of quality of service

- Temporal Parameters:
  - transfer time, response time, time for the establishment of a connection.
- Density Parameters:
  - Amount of data that an application could send per unit of time.
- Error Parameters:
  - Lost packets, late or disorderly deliveries packets, error for the establishment or the closing of a connection.
- Reliability Parameters:
  - Availability and fault tolerance.
- Security parameters:
  - Degree of protection, access control, authentication and confidentiality.
- Cost parameters:
  - The price that the user is ready to pay to get the requested quality of service.

Resume

- Real-Time Data that are updated by update transactions.
- User transaction that are making complex operations of read and write.
- Imprecise computing on the data and on the transactions.
- The quality of data increases as the imprecision on the data decreases.
- The quality of transactions increases as the imprecision of the results produced by transactions decreases.
- The quality of user transactions is linked to the quality of data (if the quality of data is decreasing then the quality of the results provided by the transactions decreases).

First solution

- The approach:
  - Distribute the load between the update transactions and the users transactions.

  ⇒ Reduce the load of user transactions leads to increase the resources available for the update transactions.

  ⇒ Reduce the load of update transactions leads to an increase of the quality of user transactions.
- Imprecise data:
  - Admitting a difference between the data stored in the database and its value in the real world,
  - If a such difference could be tolerated then we can ignore some of the update transactions,
  - Measuring the quality is effected by the means of the concept of Data Error (DE).
  - Using a Maximum Data Error (MDE) to control the Data Error.
  - Adjustment of the load of the updating transaction according to the data error measured (taking into account the MDE).

The problem of rejection of the update transaction

- Rejection or not ?
  - T_j is rejected if the error on the data (d_i) which must be updated by the transaction T_j is $\leq$ than MDE ($DE_i \leq MDE$).
  - T_j is executed and committed if $DE_i > MDE$.
- Consequences of the changes of the value of MDE:
  - MDE $\nearrow \Rightarrow$ More transactions are rejected (data error higher and so quality of data is worse).
  - MDE $\searrow \Rightarrow$ Fewer transactions are rejected (data error lower and so the quality of data greater).
- Objective: To design algorithms to adjust the data error so that the quality of data and the quality of users transactions satisfy the quality of service specified.

Transaction model (1/3)

- Update transactions and user transactions:
  - Update Transactions are periodic and write real-time data,
  - Users transactions are aperiodic read real-time data and read/write non real-time data.
- A transaction T_i is composed of:
  - a mandatory sub-transaction M_i,
  - $\#O_i$ optional sub-transactions, noted $O_{i,j}$ where $1 \leq j \leq \#O_i$
    $t_i \in \{M_i, O_{i,j}, \ldots, O_{i,\#O_i}\}$
- Decomposition of transactions according to the Milestone approach:
  - Transactions can provide imprecise results of which the precision will increase according to the number of sub-options transactions executed.

Transaction model (2/3)

- A mandatory sub-transaction is said to be complete if it is in the traditional sense (completed before its deadline).
- Sub-transactions depend on the mandatory sub-transaction and are executed only if they still have time.

$M_i \prec O_{i,1} \prec O_{i,2} \dots \prec O_{i,\#O_i}$

- All sub-transactions have the deadline of the main transaction:
  - A sub-transaction is finished if it is complete or has missed its deadline.
  - A transaction (T_i) is completed if its mandatory sub-transaction is completed.
  - A transaction (T_i) is finished when its last optional sub-transaction is finished.

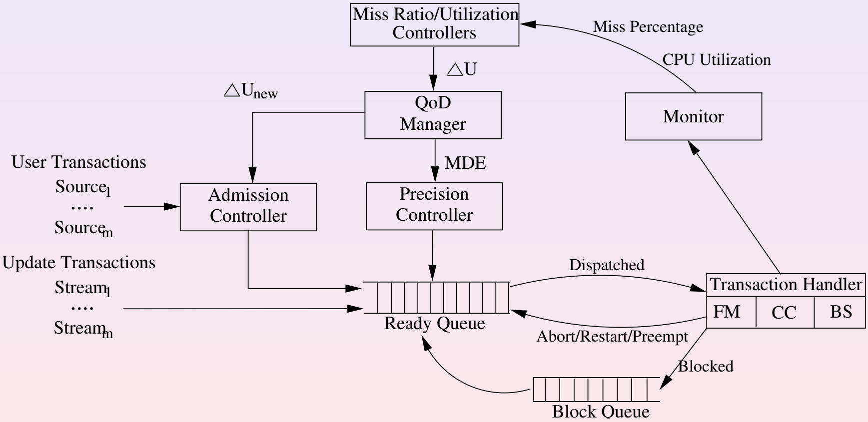
Transaction model (3/3)

- An update transaction has not many sub-transactions.
- An update transaction is only composed of one mandatory sub-transaction because it does not make complex operations but only an operation of write.
- It has an execution time fewer than the user transaction.
- The average time of execution is known but the current time isn't. We are in a model where the use of the processor is unpredicted and not known from the sheduler.

Quality of transaction and imprecise results

- Objectives:
  - to increase the availability of the resources,
  - to promote the respect of deadlines at the expense of the quality of results,
  - to allow the greatest number of transactions to be executed.
- Employed method:
  - to design transactions that are able to provide intermediate results that are exploitable,
  - to make use of algorithms which compute imprecise results but faster,
  - to increase the accuracy of a result as time.
- Remarks:
  - the quality of the results is a function that takes as input parameter the time,
  - more a transaction has time, the better the quality of its results will be.

Feedback Control Scheduling Architecture



The transaction manager (1/4)

- The transaction handler is composed of:
  - a freshness manager (FM) which check the freshness of the real-time that will be acceded using the timestamp of the data and absolute validy interval: it blocks the transactions which want to access to non fresh data,
  - a concurrency control (CC) protocol which is most of the time 2PL-HP,
  - a basic scheduler (BS) which is most of the time EDF.
- Two queues for the transactions:
  - update transactions and mandatory (users) sub-transactions are placed in the highest queue priority,
  - optional (users) sub-transactions (users) are placed in the lower queue priority,
  - taking into account the transactions of these two queue is decided at the transactions handler level.

The transaction manager (2/4)

- The admission controller (AC):
  - it controls the flow of input transactions,
  - it decides whether a transaction can be accepted or not in the system,
  - it uses parameters such as the importance of transactions (priority), the load of the system (resource use).
- The precision manager:
  - it eliminates update transactions which try to write data (d_i) with an error $DE_i \leq MDE$,
  - otherwise the new value of d_i is updated,
  - in all cases the timestamp of d_i is updated,
  - its goal is to reduce the load of the system in terms of execution of update transactions,
  - it increases or decreases the value of MDE depending on the ΔU returned by the controller use.

The transaction manager (3/4)

- The monitor:
  - it measures the number of transactions that ended before theirs deadline, ending prior to maturity or that fail to meet their deadline,
  - it take its measure from the transaction handler, Item it sends the measures it has done to utilization controller.
- The utilization controller:
  - available information provided by the instructor,
  - it makes computations on the use of the system that allows it to detect transients overload (too many transactions that fail to meet their deadline, for example)
  - it looks at CPU load of the system,
  - it makes a final computation to determine ΔU (the difference between the current utilization and the reference value) that will affect the quality of data manager.

The transaction manager (4/4)

- The quality of data manager:
  - it will increase or decrease the quality of the data based on the use of the system (in overload periods, it will decrease the quality of data)
  - it affects user transactions admitted in the system by the admission controller but also on the execution or not of the update transactions,
  - it recomputes MDE in order to decrease or increase the number of update transactions that will be executed,
  - it calculates a new ΔU from that's one provided by the utilization controller and its own internal changes,
  - the new value of ΔU is transmitted to the admission controller.

Resume about the Feedback Control Scheduling Architecture

- In input, we have parameters of quality of service specified by the DBA.
- Recomputing the parameters of the quality of service according to the runtime and the references parameters of the systems.

$\Rightarrow$ Creating a feedback loop to control the behavior of the RTDB during overload period of the systems.

$\Rightarrow$ It is not necessary to have a specific model of the load of the system over time.

$\Rightarrow$ It leads to a dynamic stabilization system according to the load and the available resources.

Applying to the multimedia systems

- How to apply the work done on the management of quality of service in RTDB to the distributed multimedia systems?
- How to be sure that the overload periods does not failure applications and decreases the performance of systems too much?
- How to take advantage of periods during which applications and networks are no longer used with intense? under utilization periods?
- How to design multimedia systems that take into account the dynamics of the applications?
- The number of users is not known in advance.

First solutions

- To decrease the quality provided during the overload period or the period of high demands from the users.
- To take advantage of periods of under-utilization.

⇒ So, the approach to use is base on the management of the quality of service.

⇒ Objective: to guarantee a certain quality of service to all users only if it is possible.

- Used method:
  - Using an admission controller in order to adjust the number of admitted user in the system according to the capacity from the network and from the video server.
  - Using the advantage offered by the imprecise computing techniques and the feedback control scheduling.
  - Use the particularity of MPEG to reduce the quality of MPEG stream.

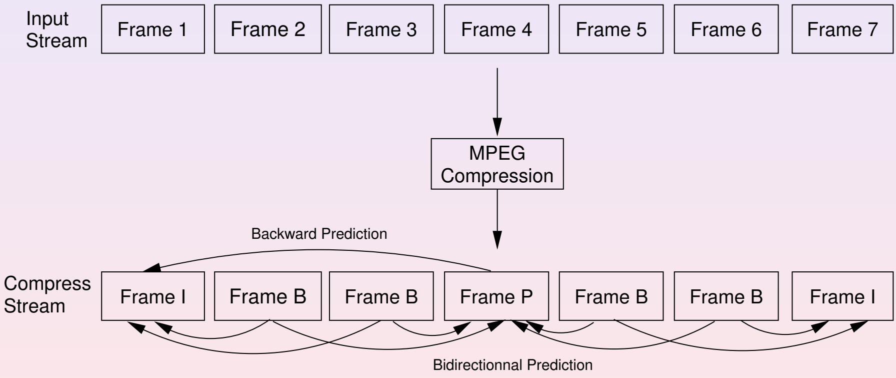
Video compression according to MPEG (1/3)

- A video is just a succession of still images that are also called it frames.
- Each image/frame may be compressed in the JPEG format
- In a video sequence, we have a succession of images in which there is a redundancy particularly if there is few movement.
- The MPEG takes into account this redundancy inter-frames to build its video.
- MPEG takes, in input, a sequence of frames and compress them into three types of frames:
  - Frame I (Intra-Frame),
  - Frame P (Predictive Frame),
  - Frame B (Bidirectional Frame).

Video compression according to MPEG (2/3)

- Frame I (Intra-Frame):
  - Independent from other frames.
  - A JPEG image.
  - They are mandatory for the decompression.
- Frame P (Predictive Frame):
  - They depends from other frames.
  - It specifies the differences with the previous frame I.
- Frame B (Bidirectional Frame):
  - They depends from other frames.
  - It specifies an interpolation between previous and next frame of type I and P.

Video compression according to MPEG (3/3)

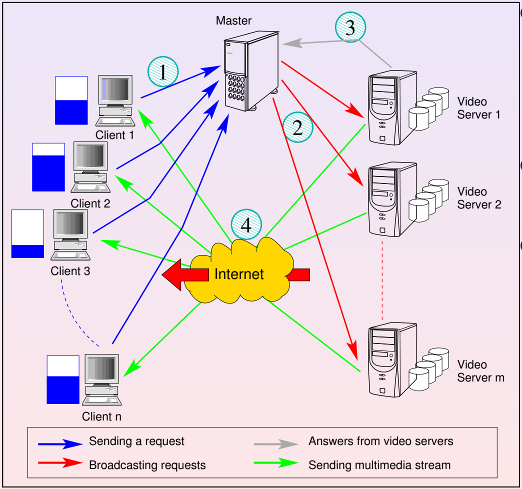


Reducing the quality of service in MPEG

- Let's take a Group of Picture (GoP) according to the format GoP (12.3).
- We would like to reduce the number of frames sent by decreasing the quality of the GoP.
- Objectives:
  - Send less data over the network when it is congested.
  - Focus on important frames like frame I.

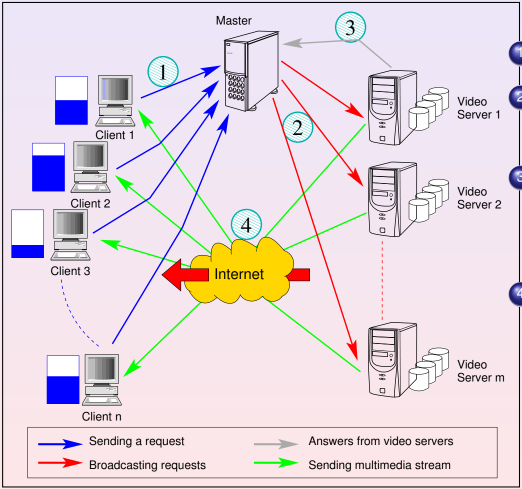
```
IBBPBBPBBPBB
IBBPB.PBBPBB
IBBPB.PBBPB.
IBBPB.PB.PB.
IB.PB.PB.PB.
IB.PB.P..PB.
IB.PB.P..P..
IB.P..P..P..
I..P..P..P..
I..P..P.....
I..P........
I...........
```

Distributed Multimedia Systems Architecture



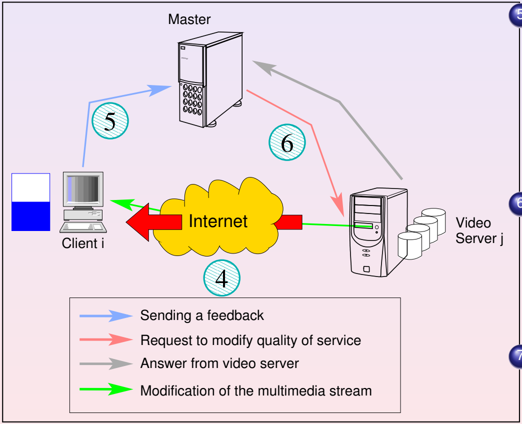
- The master server receives requests from the clients and allocates to them a video server.
- The video servers send to the client the media stream.
- The clients receive the stream from video server and play it. They also have a buffer (limited) to lessen/decrease the consequences of changes in the network.

Distributed Multimedia Systems Architecture



1. A client is sending a request.
2. The master is broadcasting the request to all video server.
3. Connection established between a client and a video server.
4. The video server is sending the multimedia stream.

The feedback loop in the architecture



5. The client are sending a feedback to the master server in order to report on the quality of service currently obtained.
6. The master server may ask the video server for an adaptation of the quality of service.
7. The stream and the quality of service is changing and then we start again at the step 5.

Replication of the most popular multimedia contents

When a video server is overloaded, it may use the following strategy for the replication of the most requested contents:

1. It sends a request to all video server available in the system.
2. Three kinds of answers are possible:
   a. It exist a video server that have the content et which is not overloaded. So here, we have no problems and do nothing.
   b. Some video server do not have the content and they are overloaded.
   c. Some video server do not have the content and they are not overload but we are not in the case of a.
3. In case a and b, the replication strategy is not taking into account.
4. In the case c, we can replicate the content on a video server that is not overload.

$\Longrightarrow$ Election of a video server that can provide the best QoS. (i.e. the video server the less overloaded).

An (m,k)-firm classification of the *frames*

Définition

(m,k)-firm model: at least m operation among k consecutive operations must meet their deadline.

Adaptation

Adaptation of the (m,k)-firm model to MPEG: at least, m frames among k must be received by the client.

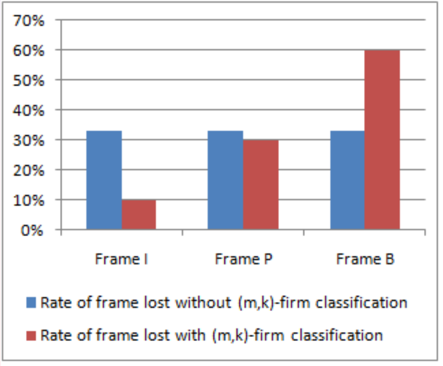
$\Longrightarrow$ k would be different according to the type of frames : I, P, B.

Simulations and results

Simulation parameters:

- (10,10)-firm for the frames I,
- (7,10)-firm for the frames P,
- (4,10)-firm for the frames B.

Results : Decreasing the rate of lost frames for the frames I.



Conclusion and future work

- A first architecture that permit to make hypothesis.
- A simulator currently under construction and will provide more results.
- Problems remains:
  - Taking into account the fault tolerance problem about the master server $\Rightarrow$ we are looking at a solution similar as peer-to-peer architecture.
  - Taking into account the problems of mobility both the client and server.
  - What about servers that act as a server at a given moment and then client at another time?
  - Adaptation of this architecture for embedded applications in communicating vehicles. Is it possible to have vehicles which exchange multimedia information at the end of games, or for road safety?

References

- Krithi Ramamritham, Sang Hyuk Son, Lisa Cingiser DiPippo: Real-Time Databases and Data Services. Real-Time Systems 28(2-3): 179-215 (2004)
- Mehdi Amirijoo, Jörgen Hansson, Sang Hyuk Son: Specification and Management of QoS in Real-Time Databases Supporting Imprecise Computations. IEEE Trans. Computers 55(3): 304-319 (2006).
- Emna Bouazizi, Claude Duvallet and Bruno Sadeg. Improvement of QoD and QoS in RTDBS. Proceedings of 14th International Conference on Real-Time and Network System (RTNS'2006), Poitiers, France, May 30-31, pages 87-95, 2006.
- Emna Bouazizi, Claude Duvallet and Bruno Sadeg. Multi-Versions Data for improvement of QoS in RTDBS. Proceedings of 11th IEEE International Conference on Real-Time and Embedded Computing Systems and Applications (IEEE RTCSA'2005), Hong Kong, China, pages 293-296, August 17-19, 2005.

My current topics of research

- Quality of service in Real-Time Database.
  - Use of Multi-Version Data to improve Quality of Service in RTDB (with a PhD student: Emna Bouazizi).
  - Management of Real-Time Derived Data in Feedback Control Scheduling (currently a graduate student is developing a simulator).
- Quality of service in Multimedia Systems (with a PhD student: Bechir Alaya and a graduate student is developing a simulator).
- Structural Model for Real-Time Databases (with a PhD student: Nizar Idoudi).