MASTER IN COMPUTER SCIENCE

Database Queries and Quality of Service Management

Minculescu, Anca

Award date:
1998

FACULTES UNIVERSITAIRES NOTRE-DAME DE LA PAIX

NAMUR

INSTITUT D'INFORMATIQUE

Rue Grandgagnage, 21, B-5000 Namur (Belgium)

Database Queries and Quality of Service Management

Anca Minculescu

Mémoire présenté en vue de l'obtention du grade de
Licencié et Maître en Informatique

Année Académique 1997 - 1998

1. Table of Contents

Table of Figures

Table of Tables

2. Abstract

The notion of Quality of Service (QoS) has evolved rapidly over the past few years. Until recently, the term "QoS" referred to certain characteristics of network performances outside the control or influence of the end user. Recent years have seen great advances in QoS research, mainly due to the emergence of multimedia networking and computing. New user perspectives and emergence of QoS demanding, multimedia applications complete these technological developments.

The concept of QoS has moved from where the end user had no influence to the delivered quality. Today, ATM networks not only have the capability of transmitting information at high speed, but they have the potential to offer end-to-end QoS-configurable communications under the influence of the end user.

The Canadian Institute for Telecommunications Research (CITR), in collaboration with the IBM Toronto Laboratory Center for Advanced Studies (CAS) initiated in 1993 a major project on Broadband Services. The goal of this project is to provide the software technologies required to support the development of distributed multimedia applications. Several issues such as multimedia data management, continuous media file servers, multimedia synchronization and QoS management are addressed in this project. To validate approaches proposed in the different areas, a *News-on-demand* prototype has been developed.

In the framework of our training period at UQAM, we focus on QoS management in the context of the CITR Broadband Services project. Our objectives are to investigate the interactions between database management systems and QoS manager. We aim at proposing a new approach where the database system executes some QoS management functions for the QoS manager. This proposal is implemented in a prototype built using relational database technology.

3. Résumé

Le concept de Qualité de Service (QoS) à évolué rapidement les derniers années. Dernièrement, la notion de QoS faisait référence à certaines caractéristiques de performance de réseaux en dehors du contrôle ou de l'influence de l'utilisateur final. Les dernières années ont connu des grands mouvements au niveau des recherches en QoS, principalement à cause de l'utilisation des multimédia. Nouvelle perspective des utilisateurs et des demandes en termes de QoS, les applications multimédia sont la complétion de cet développement technologique.

Le concept de QoS a migré vers la qualité délivrée à l'utilisateur final. De nos jours, les réseaux ATM n'ont pas seulement la capacité de transmettre de l'information à grande vitesse, mais ils ont aussi la possibilité d'offrir des communications QoS-configurables « end-to-end » par l'utilisateur.

Le CITR, en collaboration avec IBM Toronto CAS à initié en 1993 un projet principal sur "Broadband Services". Le but de ce projet est d'offrir les technologies software nécessaires pour aider le développement des applications multimédia distribuées. Certains aspects, comme le management de données multimédia, les serveurs "continuous media file", la synchronisation multimédia et le management de la QdS sont traités dans ce projet. Pour valider les différentes propositions, le prototype "News-on-demand" a été développé.

Durant notre stage à l'UQAM, nous nous sommes focalisés sur le management de la QoS dans le contexte du projet Broadband Services du CITR. Nos objectifs sont d'étudier les interactions entre les systèmes de gestion de bases de données (SGBD) et le gestionnaire de QoS. Nous souhaitons proposer une nouvelle solution où le SGBD exécute certaines fonctionnalités du gestionnaire de QoS. Cette approche sera implémentée dans un prototype basé sur la technique des bases de données relationnelles.

4. Acknowledgements

The research works we are reporting on have been developed in the frame of CITR at the Université du Québec à Montréal, at the Computer Science Department.

This research would not have been possible without the support of Professor Brigitte Kerhervé that guided and supported me with patience through all the work, providing advice and encouragement, valuable text materials and making helpful and constructive suggestions. It is my pleasure to acknowledge her scientific and personal support during the five months training period.

I am grateful to the FUNDP and especially to Professor Jean Ramaekers that gave me the opportunity to work on such an interesting project. The training period was a constructive and educational experience.

In that light, I am deep indebted to CERUNA, CITR and UQAM for their important financial support, without whom the training period wouldn't be possible. I would also like to thank the "Laureat Program" from "Province de Namur" which facilitate the contacts needed for this training period.

I would also like to express my appreciation to Prof. Gilles Gauthier and Prof. Robert Godin for providing access to their laboratories when I needed most.

The Université de Montréal research team, working on the same CITR project, provided a great deal of help with important references.

I am thankful to my colleagues at UQAM – especially Annick – who have known to create such a pleasant work atmosphere. I will like also to thank my Mother, Vlad, my friends, as well as my colleagues from FUNDP for their valuable support.

5. Introduction

Distributed multimedia systems require the integration of various services for multimedia object creation, storage, access, transfer and presentation [KERH96]. Such systems integrate different components among which the database system plays a dominant role in providing a persistent and reliable environment to store and access multimedia objects.

Research in the field of multimedia databases has essentially focused on multimedia information (the objects stored and accessed by the applications) and led to various propositions for multimedia data modelling [MEGH91], for data manipulation languages, as well as for strategies for multimedia object storage [RANG93]. Recently Klas and Sheth [KLAS94] have brought to the fore the necessity of defining and efficiently managing meta-data within the framework of distributed multimedia systems.

In order to process a search request, the various components of the distributed multimedia system require the use of meta-data. Meta-data describing the content of multimedia objects is used in the multimedia document searching process to select pertinent documents. Meta-data concerning the representation can be used for quality of service management, distribution management, and data administration. Much attention has been paid to the use of meta-data for multimedia document searching, but less to its use with other functions of multimedia systems.

Among these functions, Quality of Service (QoS) management is essential to efficiently access pertinent information at the required level of quality. This function aims to control and guarantee the level of quality that the system is able to offer to the user. While integrating such a function in a distributed multimedia system, it is necessary to consider the user's requirements regarding the quality of service provided by the system. All the components of a distributed multimedia system should be involved in the QoS management process [GUOJ96]. The users should describe their requirements related to QoS, but these requirements must be adapted to the various constraints supported by the distributed multimedia system components: client machines, database systems, server machines and transport system.

Thus, the QoS manager is in charge of managing all this information in order to provide the user with an offer which might satisfy his requirements. It then becomes essential to properly define the meta-data needed for QoS management, to integrate them within the multimedia document model and to take them into account when processing a user's request. In that light, defining the exact task of the QoS manager is one of the most important steps. In order to provide efficiency, eventual task delegation has to be discussed.

This paper investigates the interactions between database systems and QoS managers. We examine the possible role of database systems in QoS negotiation and adaptation and we propose different approaches to use database internal mechanisms to efficiently support QoS management functions.

This document is structured as follows. Chapter I introduces the context of our research. In that light, this chapter sets out the description of Canadian Institute for Communication Research (CITR) research project and a framework for discussing QoS support in distributed multimedia systems (DMS). QoS terminology and principles are introduced. The objectives of this thesis will follow.

Next – in Chapter II – the QoS Management is introduced in the context of the Broadband Services Project and especially News-on-demand Prototype. Following, the relational approach is introduced by addressing QoS manager and database manager system (DBMS) interactions.

Chapter III includes the different solutions regarding our new approach of the prototype. Three variants are discussed and the concluding choice is presented.

The implementation details presented in Chapter IV correspond to the new prototype architecture. The database creation, the user interface and the other implementation details are presented.

Chapter V regroups the general conclusions and presents areas where further work should be performed.

Chapter I: General Context

A. The Canadian Institute for Telecommunication Research

Established in 1990, the Canadian Institute for Telecommunication Research (CITR) is a federally incorporated not-for-profit research company, devoted to enhancing the competitiveness of the Canadian telecommunications and software industries through university-based research and postgraduate studies target on the research projects. Initiation and collaboration in different projects are its goals. [CITR97]

The CITR research program is defined through six Major Projects, each one a large, multi-faceted enterprise with a single unifying theme, a coherent set of objectives and involving a geographically distributed team of university-based researchers [WANG97]. They are as follows: Broadband Network Architecture, Broadband Services, Photonic Devices and Systems, Mobile and Personal Communication, Broadband Wireless Communications, Broadband Satellite Communications. Each one has been formulated in collaboration with industrial partners so as to respond to particular needs of Canadian industry.

1. The Broadband Services Project

In the light of flexibility and efficiency, the project aims at offering software technologies required to support the development of distributed multimedia applications. We speak here about such applications as electronic news, digital library, tele-medecine, remote consultation, tele-learning or collaborative work. From the industrial partnership point of view, IBM Toronto Laboratory Center for Advanced Studies (CAS) collaborates in the « Broadband Services Project ». The project goal is thus to "gain new knowledge and/or validate proposed approaches through prototyping, demonstration and evaluation on an ATM test bed".[VELT96]

Among its objectives, the "Broadband Services" project aims to integrate solutions into a prototype of distributed multimedia systems.[CITR97-98] Thus, the project leads to the development of an integrated prototype of a News-On-Demand service. This prototype integrates a continuous-media database and file server capable of supporting a remote-access, news-on-demand network service, implemented on a broadband ATM-based network platform. As reported in [VELT96] three such prototypes were build and demonstrated, the last one in March 1996.

The project has been divided into four sub-projects investigating specific issues: Multimedia Data Management, Distributed Multimedia File Service,

Quality of Service Negotiation and Adaptation, Synchronization of Multimedia Data.

The Multimedia Data Management project addresses the issue of providing data management support for distributed multimedia applications. The general approach incorporates object-orientation and distributed database functionality. The specific objectives are: logical modelling of multimedia objects, modelling of the presentation synchronization requirements of multimedia objects using a temporal object model, developing the user access primitives and developing distribution and data localization strategies for multimedia objects [CITR97-98].

The major objectives of the Distributed Multimedia File Service project are to stress test and fix problems resulting from the tests, to provide a service on the Internet which permits users around the world to use the Continuous Media File System (CMFS), to complete the performance analysis and theoretical work of the admission and the network control algorithms as well as to complete PC support for Windows 95/NT by supporting a client using MPEG hardware card [CITR97].

The CMFS is distinguished from other conventional network file servers most notably in that it is capable of handling continuous-media documents, but there other features as well, made necessary by the scale and quality objectives of the envisioned service, that are novel. We are lading at huge volume of data retrieved and transmitted at gigabit/sec speeds to tight timing constraints and responding with imperceptible delay to hundreds of different, concurrent, client requests. The various data streams involved in the transmission of a multimedia document are aligned (synchronized) at the transmission to avoid the need for large playback buffers at the subscriber. It supports concurrent read/write and QoS can be user controlled [CITR97].

The objective of the Quality of Service Negotiation and Adaptation project is to investigate the impact of dynamically changing QoS design of applications and to develop methods for the management of the quality of service related resources within a distributed environment. It aims at developing a methodology for the design of distributed multimedia applications that can take advantage of the QoS actually available in the system and optimize the user satisfaction in terms of delivered QoS and incurred costs [CITR97].

Synchronization of Multimedia Data project is concerned with the temporal synchronization of different media streams over an ATM network. [WANG97] Both intra-stream and inter-stream synchronization are considered. An important design objective is to keep buffering at the client to a minimum. [CITR97]

The objective of the Project Integration is to coordinate the milestones of the constituent projects, such that the shared goals of the Broadband Services major project are achieved. Table 1 presents each sub-project with its respective principal and co-investigators.

Sub-project	Principal Investigator	Co-investigators
Multimedia Data Management	M. T. Ozsu *(University of Alberta)*	• D. Szafron *(University of Alberta)*
Distributed Multimedia File Service	G. Neufeld *(University of British Columbia)*	• N. Hutchinson *(University of British Colombia)* • R. Ng *(University of British Colombia)* • M. Ito *(University of British Colombia)*
Quality of Service Negotiation and Adaptation	G. v. Bochman *(Université de Montréal)*	• R. Dssouli *(Université de Montréal)* • J. Gecsei *(Université de Montréal)* • B. Kerhervé *(UQAM)*
Synchronization of Multimedia Data	N. D. Georganas *(Univeristy of Ottawa)*	
Project Integration	J. W. Wong *(University of Waterloo)*	

Table 1: Broadband Services Investigators and Co-investigators [WANG97].

In order to support conversational capabilities, an extension of the prototype is taken into consideration. Its target is to suit applications where two or more users are engaged.

During our training period, we were involved in the Quality of Service Negotiation and Adaptation sub-project. Our work was supervised by Prof. Brigitte Kerhervé, from Université du Québec à Montréal, co-investigator in this sub-project.

2. New extensions of the project

The applications mentioned above are related to conferencing or collaborative work, a topic that have received much attention in recent years. 1996 brought two new areas into attention: the electronic commerce and the Internet. As far as we are concerned, the first area is concentrating under the heterogeneous repositories, QoS management, scalability and system performances. On the other hand, a sub-project will pay attention to the Internet Protocol in addition to ATM.

Table 2 [BROAD97] presents a mapping between the two new sub-projects and its investigators or co-investigators.

Sub-project	Principal Investigator	Co-investigators
Electronic Commerce – Heterogeneous Repository and Performance Issues	J. W. Wong *(University of Waterloo)*	• G. v. Bochmann *(Université de Montréal)* • M. T. Ozsu *(University of Alberta)* • B. Kerhervé *(UQAM)* • K. Lyons *(IBM Canada Ltd)*
Multimedia Support in an Integrated Services Internet	G. Neufeld *(University of British Columbia)*	• J. Wong *(University of Waterloo)* • M. Ito *(University of British Columbia)*

Table 2: New extensions investigators.

B. Distributed Multimedia Systems and Applications

This section presents an overview of distributed multimedia systems and applications. We first introduce multimedia technology concepts and then we present specificity of distributed multimedia systems.

1. Multimedia Technology Concepts

A multimedia object is an object combining a variety of information types, such as image (as graphics, for example), audio (as voice, for example) and video (as animation, for example). More specifically, a multimedia object is a set of representations of media coding information in a format that presents strong characteristics such as temporal and volume constraints, continuity and dynamism [MIS88].

An entity-relationship schema is providing in Figure 1 the dependencies between different multimedia classes and categories. In fact, a multimedia object is related to a number of temporal constraints and is composed of a set of monomedia objects. Each monomedia object is surrounded by a set of

parameters, as for example the volume of the information or the type of object. Those characteristics are not going to be detailed.

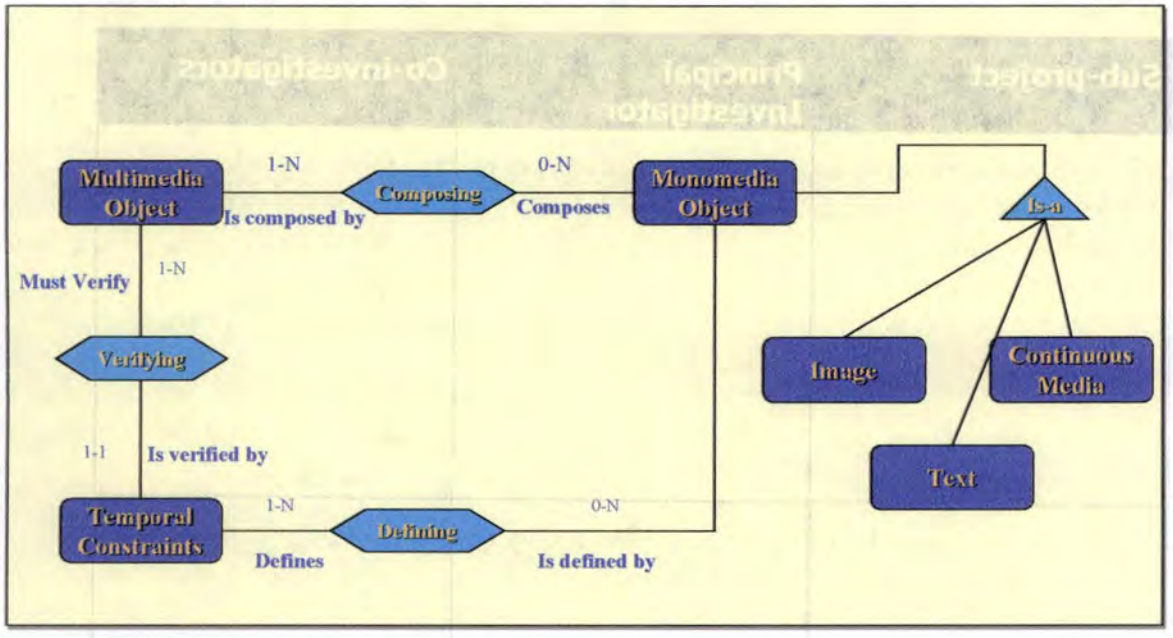

Figure 1: Multimedia Document E/R Model.

Multimedia systems are composed of hardware and software allowing the storage, transfer, manipulation and retrieval of multimedia object. Basically, multimedia systems comprise the following functional units: networks, end-to-end protocols, data management systems, applications, human-computer interfaces (HCI), operating systems, as shown in Figure 2.

Working with a multimedia system does not only mean to manipulate and transmit multimedia objects. It additionally means storing, retrieving and presenting multimedia data. These operations are thus strongly dependent on the environment. In that order we introduce hereafter the distributed multimedia systems, a context where manipulation of multimedia objects is obvious.

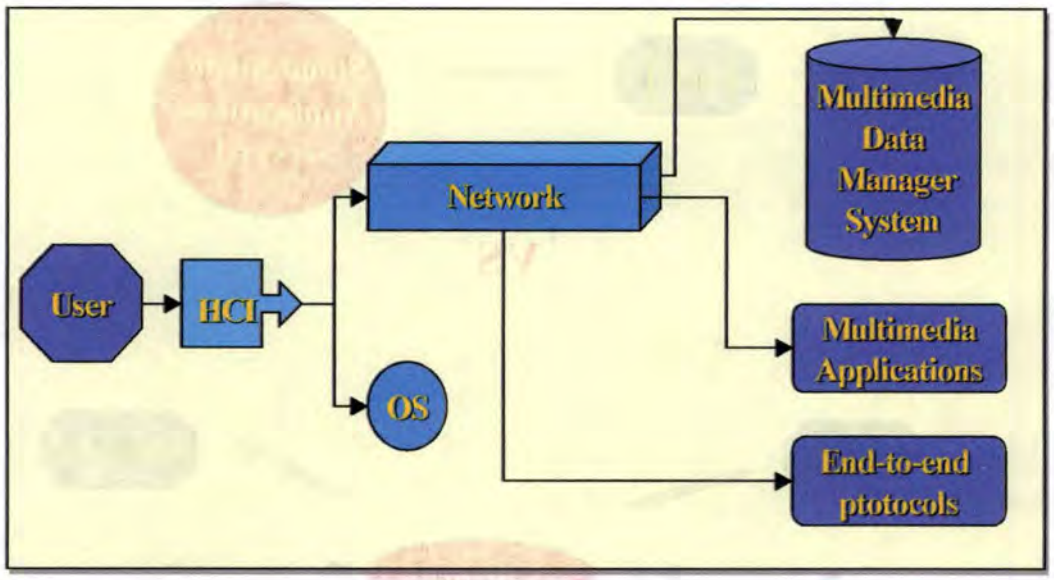

Figure 2: Multimedia Systems.

2. Distributed Multimedia Systems

Research and development efforts in multimedia computing fall into two groups [FURT94]. One group centers efforts on the stand-alone multimedia workstation and associated software systems and tools, such as music composition, computer-aided learning or interactive video, while the other combines multimedia computing with distributed systems. This last area includes multimedia information systems, collaboration and conferencing systems, on-demand multimedia services and distance learning.

While stand-alone applications principal need is high performance local resources, the distributed ones are involving factors like network speed capabilities, resource reservation all over the net, adequate communication protocols and information systems.

Both approaches are centered on real-time constraints that any multimedia application requires. But if a stand-alone application has to justify the delays « in private », the distributed ones need to know the characteristics of all involved devices. In that order, high capacity storage - as well as the possibility of integration of various services - is imposed.

Figure 3 visualises the comparison between stand-alone and distributed applications.

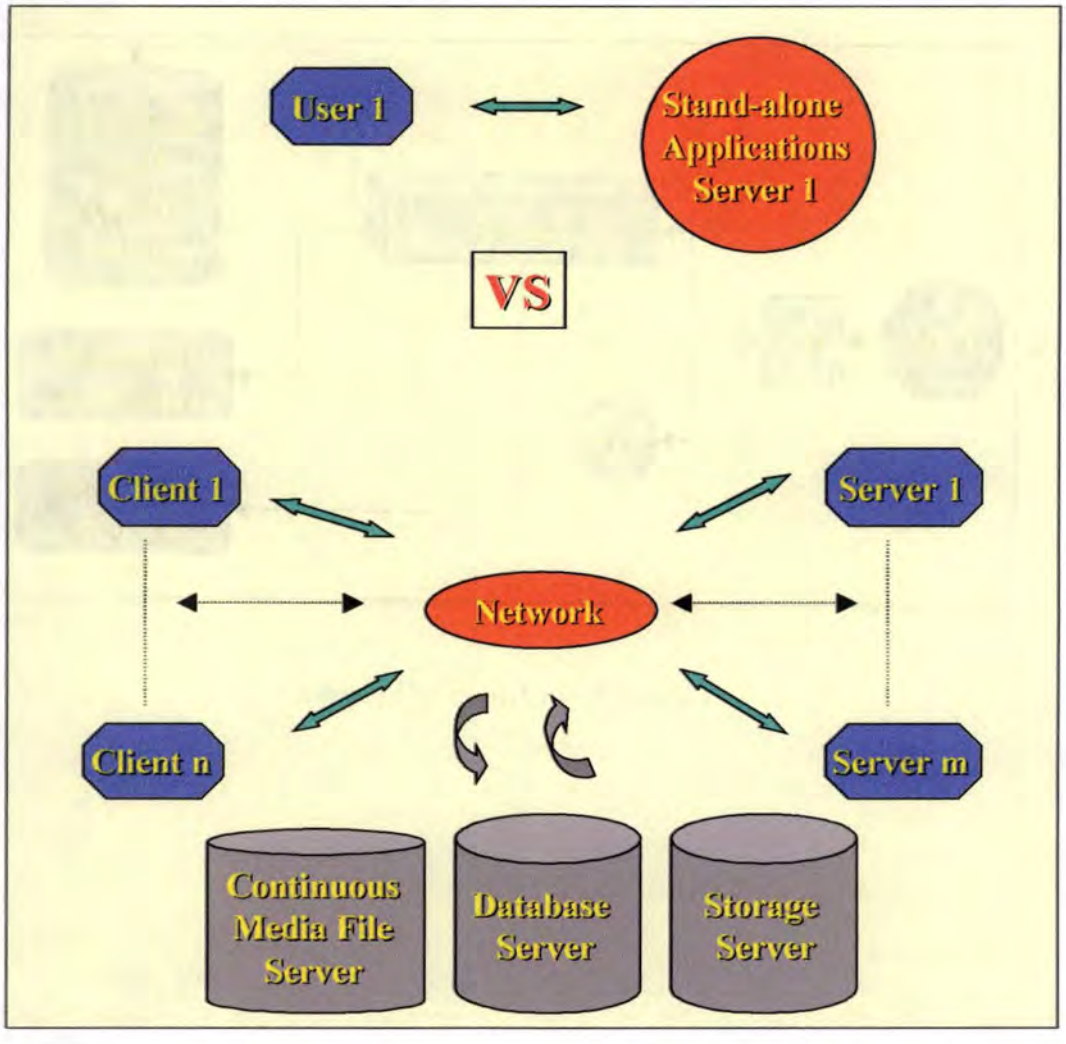

Figure 3: Distributed Multimedia Applications vs. Stand-alone.

In different words, a distributed application is based on a sharing service while the stand-alone application is independent. Obviously, a gap between functionality offered by operating systems and the specific needs of distributed multimedia applications appears. To the date, software components are a possible solution to fill that gap.

As mentioned before, networking is very important in a distributed multimedia environment. The network choice is capital, for the traditional ones are trying to provide a free-error service that is not always desirable. In fact, in order to maintain synchronization, delayed packets are even rejected. In this case a low jitter is more important than the recover of a lost packet. According to those issues, Asynchronous Transfer Mode (ATM) networks seem to regroup the main characteristics to suit multimedia traffic.

In fact, the basic idea behind ATM is to transmit all information in small fixed-size packets called cells [TAN96]. There is a variety of reasons why cell-switching was chosen, among them are the following. First, cell-switching is highly flexible and can handle both constant rate traffic (audio, video) and variable rate traffic (data) easily. Second, at the very high speeds environment

(gigabit/sec are within reach), digital switching of cells is easier than using traditional multiplexing techniques, especially using fiber optics. Third, broadcasting is essential; cell-switching can provide this and circuit switching cannot.

It is commonly accepted that a major advantage of multimedia computing in distributed environments is the possibility to share resources among users and applications, where the shared resource might be a multimedia object. The concept of distributed multimedia applications combine thus the advantage of distributed computing with the capability of processing discrete and continuous media in an integrated fashion.

But speaking about distributed multimedia applications means talking about video-on-demand as well as video-conferencing. From that point of view, [GOLD] shows another classification:

- presentational applications
- conversational applications.

We can call conversational applications whatever multimedia application involving a conversation, involving different partners during the communication or involving real-time conferences. Presentational applications provide remote access to multimedia documents. The conversational applications focus on the real-time multimedia communication. Logically, we therefore deduce that properties such as delay are more important in conversational applications than in the presentational ones. The real-time constraints are thus stronger.

C. QoS in DMS Environment

This section is dedicated to an overview of the QoS concepts and major characteristics. We present QoS parameters and we introduce the concepts of QoS management in a distributed environment.

1. The QoS Concept: Semantics and Parameters

With the emergence of distributed multimedia applications, however, QoS has become a major issue in distributed systems research. To introduce the QoS, a very first definition is clarifying the context:

*"**QoS** represents the set of those quantitative and qualitative characteristics of a distributed multimedia system that are necessary to achieve the required functionality of an application." [HAFI96]*

The concept of QoS induces to its main function: "to control and guarantee the level of quality that the system is able to offer to the user"[GUOJ96].

Approaching QoS develops different viewpoints: from the network provider, the customer and the regulator perspectives. We therefore provide different definitions in order to hinge on each point of view. Moreover, it is realistic to imagine that intersection of the three viewpoints might be empty.

Regarding the problem as a customer, we can interpret the network as a market place dominated by concurrence. In that case, each customer is in competition for resources. Meanwhile, he will try to minimize its costs and maximize the QoS guarantees. Therefore, the next definition is more appropriate:

Customer definition: *QoS is described in terms of a set of user-perceived characteristics of the performance of a service; it is expressed in user understandable language and manifests itself as a number of parameters, all of which have either subjective (principally not directly measurable, e.g. depending upon user equipment) or objective (directly observed, e.g. delay, throughput) values.[HAFI96]*

Turning to the network provider side, we find a maximization problem concerning the revenue over a short-term period, so far a pricing decision. A possible definition is:

Network provider definition: *QoS is a set of parameters that express the system behaviour performance during a certain period of time and parameters that express other service characteristics such as security or priority.[HAFI96]*

Finally, the regulator position is similar to a supervising one: he needs to know the supplier's strategy so that he can determine fairness, but he also might force suppliers to agree on different prices. In that conception, the definition of QoS may sound as:

Regulator definition: *QoS is a collective effect of service performance that determines the degree of satisfaction of a user of the service. [HAFI96]*

An exhaustive classification of QoS parameters is non-existent to date. In that light, this section focuses on the state of the art in this domain. We will therefore try to define a set of necessary QoS parameters.

Several hierarchies have been world-wide integrated: Open Systems Interconnection (OSI) Performance-oriented, OSI Non-performance oriented, ATM Forum QoS Attributes, etc. Each attribute is dependent on an **activity**. This term is introduced to refer to the system aspects for which it is useful to describe QoS characteristics. Examples of activities are processes, communications, complete computer systems. One particularly important type of activity in the context of multimedia and real-time distributed applications is the concept of "flow" that [VOGE95] defined it as:

"The production, transmission and eventual consumption of a single media source as an integrated activity governed by a single statement of QoS; flows generally require end-to-end admission control and rssource reservation and support heterogeneous QoS demands."

In this optic, a possible result is the union of different QoS requirements. Under these circumstances, updating the result of that union is a capital operation. A non-exhaustive list of parameters, whose activity is mainly situated at the transport layer, is hereafter available in Table 3:

	Parameter	Description[1]	Standard
1	**Throughput**	The maximum number of bytes, contained in Service Data units (SDUs) that may be successfully transferred in unit time by the service provider over the connection on a sustained basis.	**OSI Performance-oriented**
2	**Transit delay**	The time delay between the issuing of a *data.request* and the corresponding *data.indication*. The parameter is usually specified as a pair of values, a statistical average and a maximum. Those data transfers where receiving service exercises flow controls are excluded. The computations are all based on SDUs of a fixed size.	**OSI Performance-oriented**
3	**Residual error rate**	The probability that an SDU is transferred with error, or that it is lost, or that a duplicate copy is transferred.	**OSI Performance-oriented**
4	**Establishment delay**	The delay between the issuing *connect.request* and the corresponding *connect.confirmation*.	**OSI Performance-oriented**
5	**Establishment failure probability**	The probability that a requested connection is not established within the specified maximum acceptable establishment delay as a consequence of actions that are solely attributable to the service provider.	**OSI Performance-oriented**
6	**Transfer failure probability**	The probability that the observed performance with respect to transit delay, residual error rate or throughput will be worse than the specified level of performance. The failure probability is, as such, specified for each measure of performance of data transfer.	**OSI Performance-oriented**
7	**Resilience**	The probability that a service provider will, on its own, release the connection, or reset it, within a specified interval of time.	**OSI Performance-oriented**
8	**Release delay**	The maximum delay between the issuing of a *disconnect.request* primitive by the service user and a corresponding *disconnect.indication* primitive issued by the service provider.	**OSI Performance-oriented**
9	**Release failure probability**	The probability that the service provider is unable to release the connection within a specified maximum release delay.	**OSI Performance-oriented**

[1] Descriptions are drown out from [GUOJ96], [MADJ97] and [HAFI96].

	Parameter	Description	Standard
10	**Protection**	The extent to which a server provider attempts to prevent unauthorized monitoring or manipulation of user data.	**OSI Non-performance Oriented**
11	**Priority**	High-priority connections are served before lower ones. Lower-priority connection packets will be dropped before high-priority packets if the network becomes congested.	**OSI Non-performance Oriented**
12	**Cost**	Define the maximum acceptable cost for a network connection; final actions of this parameter are left to the specific network providers.	**OSI Non-performance Oriented**
13	**Peak cell rate**	The maximum instantaneous rate at which the user can transmit. For bursty traffic the inter-cell interval and the cell rate varies considerably. The peak cell rate is the inverse of the minimum inter-cell interval.	**ATM Forum**
14	**Sustained cell rate**	This is the average rate measured over a long time interval.	**ATM Forum**
15	**Cell loss ratio**	The percentage of cells that are lost in the network because of error or congestion and are not delivered to the receiver.	**ATM Forum**
16	**Cell transfer delay**	The delay experienced by a cell between entry and exit points is called the cell transfer delay. This includes the propagation delays, queuing delays at the various intermediate switches and service times at queuing points.	**ATM Forum**
17	**Cell delay variation**	This is a measure of variance of the cell transfer delay. High variation implies large buffering for delay sensitive traffic such as voice and video.	**ATM Forum**
18	**Burst tolerance**	This determines the maximum burst size that can be sent at the peak rate. This is a bucket size parameter for a leaky bucket algorithm. This is also a product of the maximum burst size.	**ATM Forum**
19	**Minimum cell rate**	This is a minimal cell rate desired by the application.	**ATM Forum**

Table 3: Parameter Description.

2. Integrating QoS on a Distributed Multimedia System

Let's have a look at the "multimedia document" flow "travel". From media source devices down to the source protocol stack, across the network and up through the receiver protocol stack to the end-system devices, flow crossbar different resources. That implies specifying different levels of QoS. At each of the hereafter layers a QoS module is thus required:

- User
- Application
- System
- Network.

Quality of Service management on an end-to-end basis means that: (1) the user expresses his wishes on the quality he wants and the cost he is willing to pay, and (2) the system transparently operates in order to deliver the requested level of quality. This approach requires to consider QoS not only at a specific layer, such as the network or the operating system layers, but at all the layers of the distributed multimedia system. As described in [KERH96], Figure 4 places QoS management across the different layers of distributed multimedia systems.

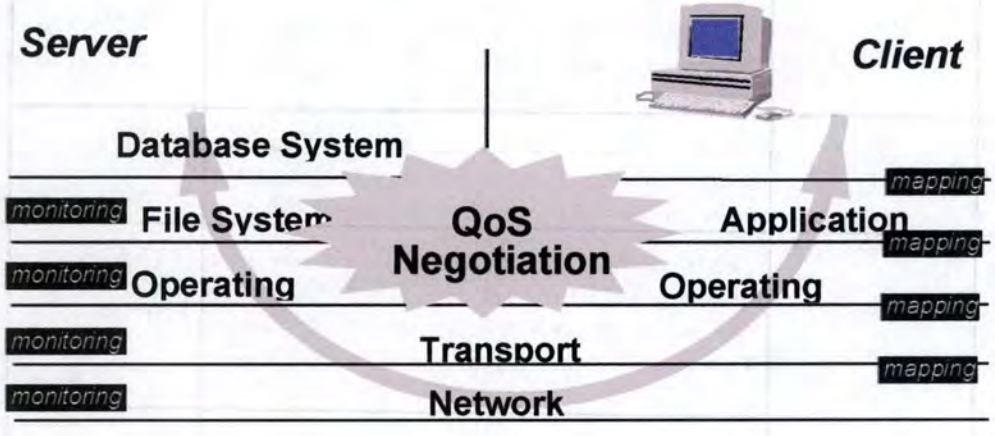

Figure 4 : Multimedia System Layers and QoS Management [KERH96].

Our distributed multimedia system follows a multi-client multi-server architecture. The client offers search and access to the multimedia database and the server provides reliable storage for multimedia objects.

Figure 5 illustrates the physical architecture of our distributed multimedia system. The server is composed of a database server that provides access to specialized file servers such as continuous media file servers or archival storage servers.

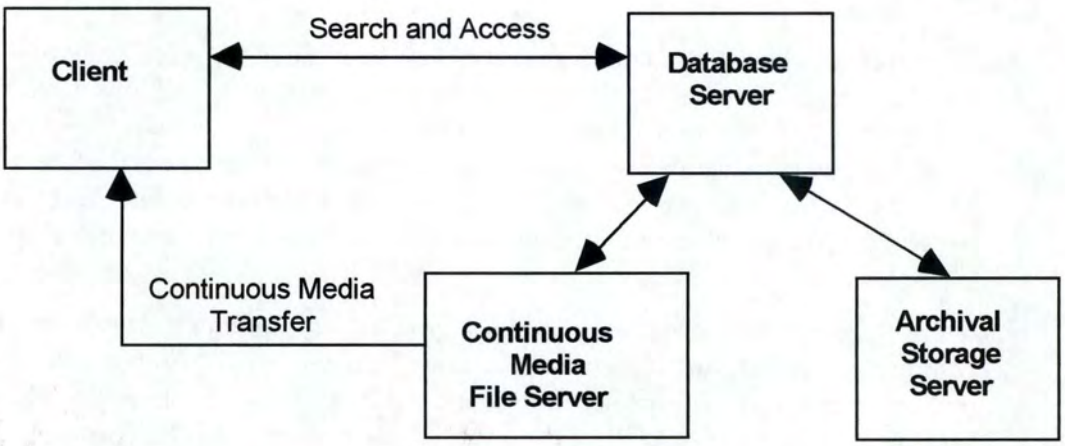

Figure 5: Physical Architecture [KERH96].

D. Research Goals

QoS, as viewed by a particular user, is determined by a combination of individual taste, terminal capabilities and network operating conditions. Our idea was to build into the user interface a mechanism by which the user, reacting to these various considerations, can explicitly select or modify quality of service. The framework will provide the user a way in which to define and select particular QoS profiles, with the option of specifying QoS (with the help of examples).

Regarding the idea of proposing enabling technologies for electronic commerce systems, the "Electronic Commerce: heterogeneous repositories and performance issues" sub-project was launched by CITR under the "Broadband Services" co-ordination. One of the goals of the research on QoS management is to examine the possible role of database systems in QoS negotiation and adaptation. Efficiently supporting QoS management functions will be a target. In that light, specific QoS management tasks can be delegated to different components of the distributed multimedia system. During our training period, we focused on the delegation of QoS management tasks to the database system components and on the use of database internal mechanisms for implementing these tasks.

The sub-project Quality of Service Negotiation and Adaptation conducted at Université de Montréal (UdM) and Université du Québec à Montréal (UQAM) investigates the impact of dynamically changing QoS on the design of the application. In this framework, a QoS negotiation protocol has been prototyped at University of Montreal and is currently analyzed and enhanced. Recently, a new sub-project has been integrated in the Broadband Services major project. This sub-project, entitled "Electronic Commerce: heterogeneous repositories and performance issues" aims at identifying and proposing enabling technologies for electronic commerce systems. We presently investigate QoS issues in this context where a large-scale distributed architecture involving heterogeneous components is provided. Among the essential functionality supported by large-scale distributed multimedia systems, QoS negotiation and adaptation are of prime interest and require the involvement of the different system components such as database systems, file servers or client machines.

During the five months project in this context, we investigate interactions between database systems and QoS managers. We examine the possible role of database systems in QoS negotiation and adaptation. We propose different approaches to use database internal mechanisms to efficiently support QoS management functions and then revisit QoS algorithms, especially specification, and negotiation. This work leads to the development of a simplified prototype of a multimedia application where QoS functions are implemented on top of a relational database system.

Chapter II: QoS Management in the Broadband Services Project

In previous sections, we introduced the general context in which our research project is conducted. We focused on distributed multimedia system and presented concepts and approaches in these systems.

In this chapter we present QoS management in distributed multimedia systems and we detail the approaches and implementations of QoS management that have been done in the framework of the Broadband Services, a major project of the CITR.

This chapter first presents a study on QoS management and defines the principal steps illustrated in a task diagram. Next, the existing QoS architectures are introduced. Later we give a detailed presentation of the QoS manager implemented in the News-on-demand Prototypes. The different prototype implementations of the News-on-demand service are then discussed. Finally, we describe the interactions between the QoS manager and the database management system.

A. QoS Management

In a distributed multimedia system, QoS management is the component responsible for the delivery of multimedia documents at a guaranteed level of quality. This quality can be expressed in terms of system performance, quality of information, financial costs or document duration. A precise definition is:

> ***QoS management*** *is a set of activities and decisions needed to perform a QoS guaranteed exchange, i.e. to obtain an optimal system configuration. [VOGE95]*

Based on [urlUNIK] and [HAFI96b] analysis, we describe the QoS Manager as activating the next several steps:

1. QoS specification
2. QoS mapping
3. QoS negotiation
4. QoS renegotiation
5. Resource reservation
6. QoS monitoring
7. QoS adaptation
8. QoS accounting

9. Admission control
10. QoS policing
11. QoS close-down procedure

A QoS manager, supporting the previously defined QoS management steps, is implemented inside this architecture both on the server and the client sides. QoS management takes place during the different phases of the user's request processing search, access and display. [KERH96] presents, as in Figure 6, the functional view of our system in OMT object model notation, this view focuses on QoS management and shows the successive steps that are processed to deliver multimedia objects to the users. Ideally, QoS management should become completely transparent to the user.

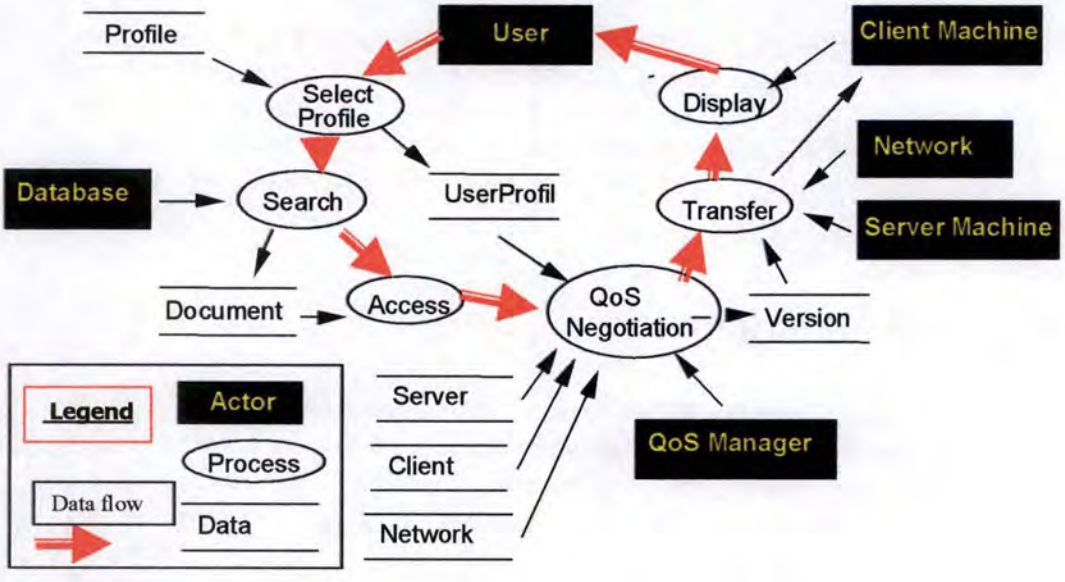

Figure 6 : Functional View of QoS Management [KERH96].

This functional view –presented in OMT object model notation– focuses on QoS management and shows the successive steps that are processed to deliver multimedia objects to the user. Ideally, QoS management should become completely transparent to the user. In this prototype this goal is achieved by QoS specification through the definition of QoS profiles. these profiles are defined for particular applications and users. At the beginning of the session, the user sets his preferred profile through the dedicated user interface. For the applications, we have defined an API allowing the specification of their behaviour concerning QoS characteristics and management. That becomes the responsibility of application programmers to specify applications QoS requirements and their reaction to QoS modification or degradation.

While querying multimedia database, we distinct two different phases: the search and the access phases. The search phase consists in isolating a set of documents of potential interest for the user. Among these documents only those specified by the user during the access phase will be completely delivered. These different steps are required to avoid the delivery of large amount of data that are not of real interest to the user. During the search phase, the information delivered to the user are the meta-data describing the content of the document. These meta-data allow the user to select the document he wants to access. [KERH96]

For the chosen multimedia document, the negotiation protocol is initiated to select the variant of each composed monomedia object that matches the requirements of the user given by his user profile and the constraints of all the involved actors. These constraints are identified for each actor by analyzing its QoS information. Some of the QoS parameters, such as the client machine environment are static, while other parameters are dynamic, such as the load of the server machine or the available memory resources.

In order to obtain a qualitative result, we can logically group the steps presented here-before by task or functionality. We first describe the different steps and then we group them into tasks.

1. Step definitions

We first define each step of a so-called "QoS transaction". Let assume the next hypothesis:

- Let "U" be a user asking for a multimedia service.
- U is initiating its requested QoS, in a user-comprehensible language;
- Let "ReqQoS" be the values given to the needed QoS parameters. ReqQoS must be translated into a detailed syntax understandable and useful for the distributed multimedia system ("DMS");
- Let "MapQoS" be the new set of QoS parameters. Next, MapQoS must be guaranteed by the system.
- Let "SysRes" be the set of system components finally required.

1. QoS specification

An ergonomically HCI with easy-understandable options –as for a novice user– should offer the possibility of choosing or introducing all parameters the user can desire. Minimizing the cognitive charge is a must. In that optic, a list of relevant parameters could eliminate misunderstandings. A non-neglected aspect should thus be the background of the user. In conclusion, the interface could be distinct according to U's level. Constraints as measure unity, parameters name or description have to be taken into consideration.

In order to provide more flexibility, the user should be allowed to specify the importance of different parameters. Once ReqQoS is established, the specification phase is finished.

2. QoS mapping

This step realizes the mapping between ReqQoS and MapQoS. We encounter this phase at different levels, practically between every layer of the QoS model. We then speak about a QoS-QoS mapping. In fact, each level describes the QoS parameters in a specific language, proper to its level. The translation from one language to the other is done by the QoS mapping.

Once MapQoS is established, another task should be accomplished: mapping MapQoS and the resources as well as the services into service components.

3. QoS negotiation

QoS negotiation goal is to reach an agreement by which user and system will cooperate on the required QoS values. At this step, the cost parameter is playing a big role. It might be unlikely, and especially inefficient, to arrive at a re-negotiation just because omitting this parameter. In that light, an important remark is to impose the user to negotiate the cost at the beginning of this step. In Message Sequence Chart (MSC)[2] representation, the cost constraint may be modelled as hereafter in Figure 7:

[2] As far as negotiation is based on a message exchange between user and service provider, MSC notation seems to be the adequate choice of representation.

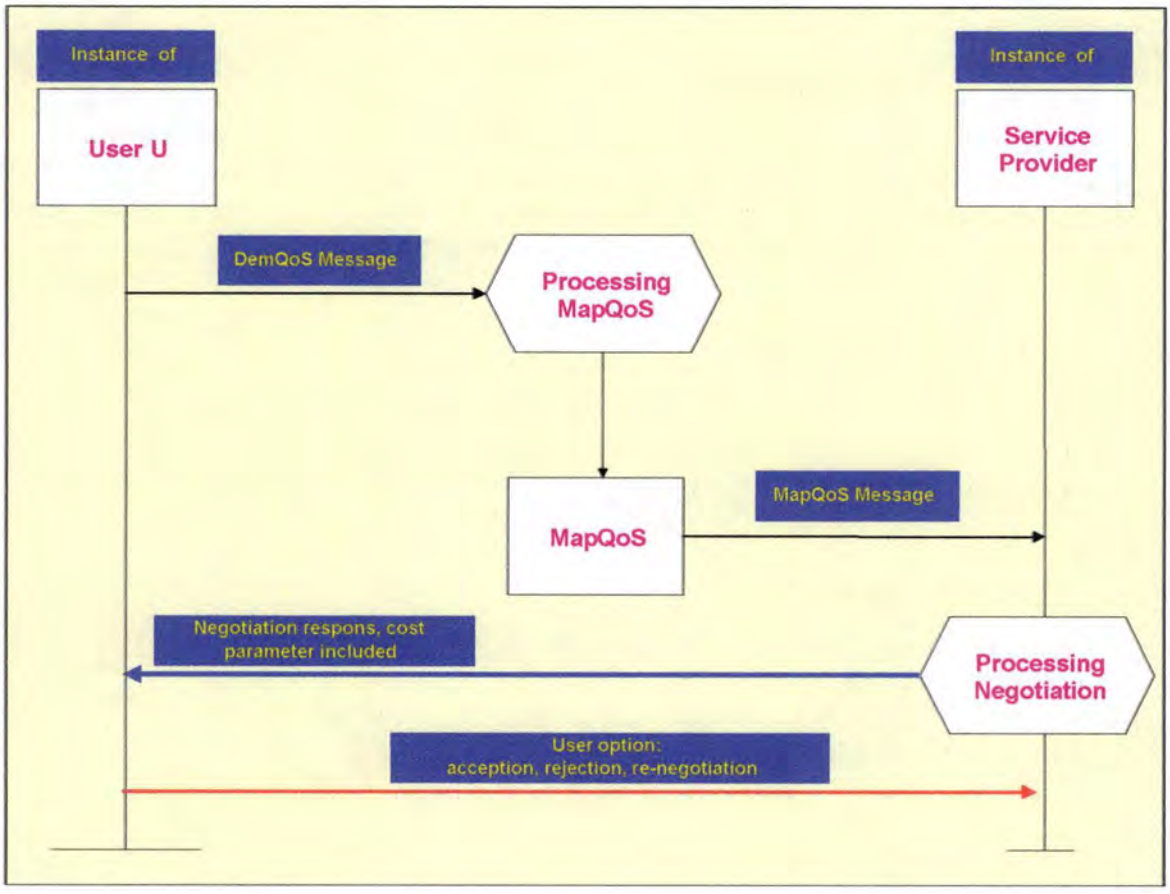

Figure 7: MSC Model of a Cost Constraint [urlObjectGeode].

4. QoS re-negotiation

Three reasons can activate a re-negotiation: a user request for better quality, a resource lack signalized by the system or a failed negotiation.

The big lines of the re-negotiation mechanism are similar to the negotiation ones; hoping results are better. If any accord is settled, the re-negotiation phase may loop back to its beginning.

5. Resource reservation

Inasmuch as multimedia applications require an end-to-end QoS guarantee, network resources are substantially demanded. Therefore, the resources reservation phase is pointed. As far as "the" perfect solution is obviously not determined, a risk has to be taken: over-appreciate or under-appreciate the network. Nevertheless, the reservation act is done and a mechanism of reservation is needed. Protocols as RSVP[3] or ST-II[4] can be used in order to complete this task.

[3] Resource Reservation Protocol

6. QoS monitoring

The QoS monitoring step is more a supervisor task. During this stage, statistics are performed and QoS controls are made in order to examine system state concerning QoS guarantees or violations. To obtain all needed data, measurement procedures and methods should be defined.

7. QoS adaptation

The QoS adaptation goal is to maintain the service on depend of its quality. That means QoS parameters should be modified on the way. Techniques such as scalable encoding, interpolation or dynamic compression will thus be employed.

8. QoS accounting

We can consider QoS accounting step much closer to QoS negotiation and re-negotiation than a stand-alone step. In fact its target is to determine the cost of the service. The "cost components" concept is largely enough and will not be discussed in this chapter.

9. Admission control

This step consists in examining if the user request can be admitted by the system, which means that the system has to check if it can support this new request without disturbing the other request currently executed. This is a step to be considered in close relation with the negotiation. In fact, a negotiation can only start after the success of the admission control.

10. QoS policing

Up to the time that we want to provide guarantees we have to assure non-existence of any violation of those guarantees. Here is the goal of QoS policing: preventing and obstructing misbehaviours.

11. QoS close-down procedure

By the time a service is finished, no more resource reservation is needed. More than that, other users should be allowed to benefice of those resources. In that light, a resources liberating procedure is indispensable.

[4] Stream Transport Protocol Version Two

2. Task decomposition

We thus obtain a task decomposition of the QoS management that stands as follows:

1. Identifying the user needs in terms of QoS.
   - QoS specification
   - QoS mapping
2. Establishing a contract between user and service provider on the QoS guarantees to be provided.
   - QoS mapping.
   - Admission control.
   - QoS negotiation.
   - QoS accounting.
   - QoS re-negotiation.
3. Providing the QoS guarantees.
   - Resource reservation.
4. Controlling the required QoS.
   - QoS monitoring.
   - QoS policing.
5. Finding a go-between solution in critical situations.
   - QoS adaptation.
   - QoS re-negotiation.
   - QoS accounting.
6. Ending the established contract.
   - QoS close-down procedure.

3. Task and sub-task diagram

Starting with the main aim of **performing a QoS guaranteed exchange**, we design hereafter in n different tasks and sub-tasks.

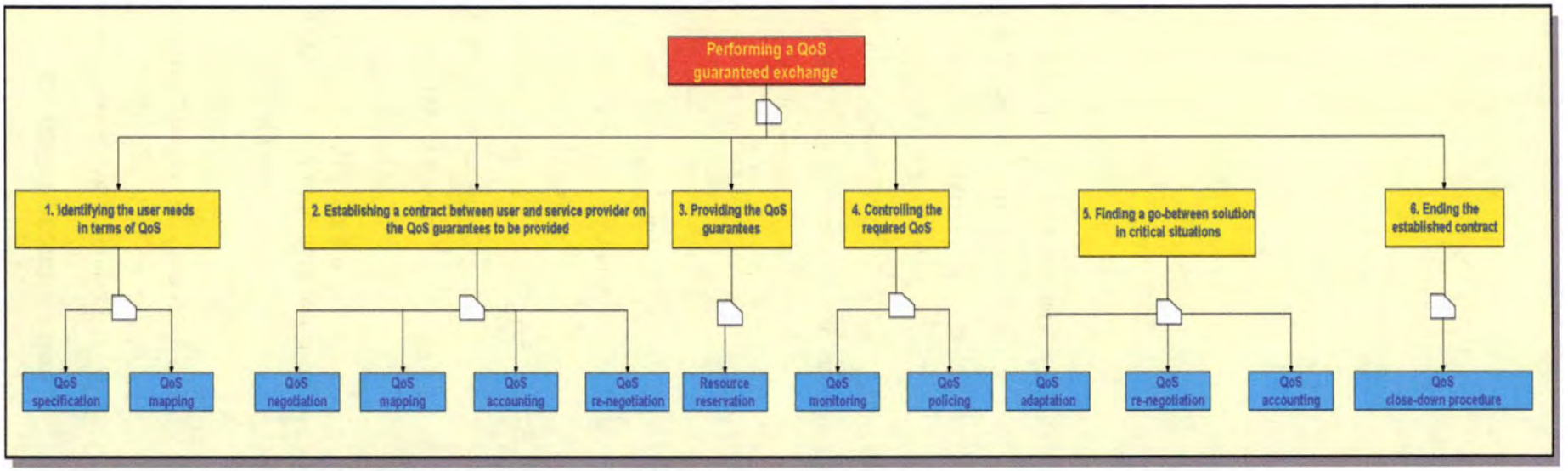

Figure 8 the decomposition of this goal in different tasks and sub-tasks.

Figure 8: Task diagram.

B. QoS Architecture

Until recently, research in providing QoS guarantees has mainly focused on network oriented traffic models and service scheduling disciplines. These guarantees are not, however, end-to-end in nature, rather they preserve QoS guarantees only between network access points to which end-systems are attached. Work on QoS-driven end-system architecture must be integrated with network configurable QoS services and protocols to meet application-to-application requirements. In recognition of this, researchers have recently proposed new communication architectures which are broader in scope and cover both network and end-system domains. This section presents an overview of the most representative approaches.

In order to obtain a non-redundant QoS Architecture a few principles are needed. Thus, [CAMP96] defines four concepts as majors:

- Separation principle
- Management principle
- Performance principle
- Transparency principle.

In order to obtain better results media transfer, control and management should be separated activities. In fact, flows or signals don't have to be constrained to the same parameters.

The management principle refers to the asynchronous resource and treats the scheduling operations, the flow control, the routing, etc. The performance principle is to be applied at a design and implementation level of communications. Rules in structuring communication protocols are a perfect example. Transparency principle puts applications into a shelter, in order to protect them of QoS monitoring and maintenance.

We can now introduce the major QoS Architectures: OSI QoS Framework, OMEGA Architecture, Heilderberg QoS model, QoS-A or Tenet QoS Architecture. Tenet Suite or Heidelberg Protocol (HeiTP), both connection-oriented protocols are providing QoS guarantees at the network and transport layers [GUOJ96]. The OMEGA architecture is more complex, in the way that it also concerns the application layer.

The Tenet protocol suite can be visualized as in Figure 9. Real-time Message Transport Protocol (RMTP) and Continuous Media Transport Protocol (CMTP) represent the transport protocol and run over the Real-Time Internet Protocol (RTIP). The Real-Time Channel Administration Protocol (RCAP) provides resource reservation, admission and QoS network handling, while the Real-time Control Message Protocol (RTCMP) is generating error detection.

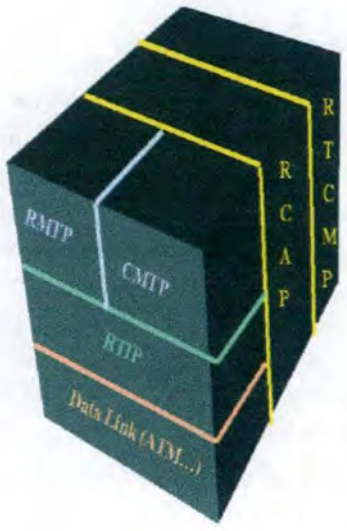

Figure 9: Tenet Protocol Suite Architecture [urlFUNDP].

Permitting multicast, HeiTP is transmitting continuous media data streams to one or multiple targets. The Heidelberg System (HeiTS) is based on the ST-II protocol and is initiating a large QoS negotiation phase.

Both HeiTS and Tenet Suite are providing real-time guaranteed services up to the transport layer. Or, real-time applications require end-to-end guarantees. The OMEGA architecture is offering application-to-application QoS guarantees; it relies on network management and transmission protocols for provision of guarantees in intermediate nodes.

All QoS architectures mentioned consider QoS specification (e.g. contracts, flow specifications, service and traffic classes, etc.) as fundamental to capture user level QoS requirements. Some architectures puts QoS specification at different logical layers or planes in the end-system and network – as in the case of the "QoS –A". In the latter one, QoS mapping is used to translate QoS specifications between logical layers/planes. Developing a new architecture should thus take into consideration the existing models and especially the lack of guarantees in these last ones.

C. The QoS Manager in the News-On-Demand Prototypes

We now integrate the concept of QoS management and architecture into the Broadband Services Project, more precisely, into the News-on-demand Prototypes.

News-on-demand prototype was created for the storage, retrieval and presentation of multimedia news information. The prototype draws on broadcast news from a radio and a TV station (CBC) and from newspaper publishers (the University of Waterloo Gazette).[HAFI94]

The first News-on-demand prototype was implemented in 1995. This first complete version (Proto1), running on older than Unix4 versions, is available. Proto1 is incorporating a QoS Manager and was designed for one client and one server. Due to last year's developments in communication field, an extension of the Proto1 was proposed in 1996 in order to accept several clients and one server (e.g. teleconference, real time data transfer, etc.). This new prototype (Proto2) does not include a new version of the QoS Manager module since other developments, such as integration in the Web environment have been performed during that period.

Figure 10, using OMT notation, shows the abstract architectural design of a remote access to a MM databases system, such as described in Proto1. The design focuses on the aspects of the system that are essential for QoS negotiation and adaptation, two important steps in the QoS manager activities. The main components are presented as objects along with the methods they support.

The application starts by executing a *SearchDocument()* operation on the Database object with search information obtained from the user. When a document is selected by the user, the application determines the QoS user profile to be used during the session; the profile manager is involved at this point by executing either *GetDefaultProfile()* or *GetCurrentProfile()* functions. *GetDefaultProfile()* sets a default profile, while *GetCurrentProfile()* sets the profile defined by the user. Then the application calls *NegotiateMMPresentation()* with the document and user profile as parameters.

The *NegotiateMMPresentation()* operation of the QoS manager allows the selection of a suitable configuration which might support the delivery of the document while satisfying the user profile. For this propose, the QoS manager will consult the meta-data of the document and determine –for each monomedia component- which of the available variants[5] is the most appropriate to be used. For each monomedia, once the best variant is chosen, the QoS manager asks the network to open (or activate, if the connection is already open) a real-time transport connection (methods *Connect()* or *Activate()* are used). Once a variant

[5] Each monomedia may exist in different physical representations, which we call variants. The variants of a given monomedia may differ in the format of coding, the size of file, the quality of media, etc. and may be located in different file servers.

has been activated for each monomedia component of the document, the QoS manager starts the playout of the document by calling the *Play()* operation. *QoSViolation()* or *AdaptMMPresentation()* are called if the network monitor detects a QoS violation. [HAFI94]

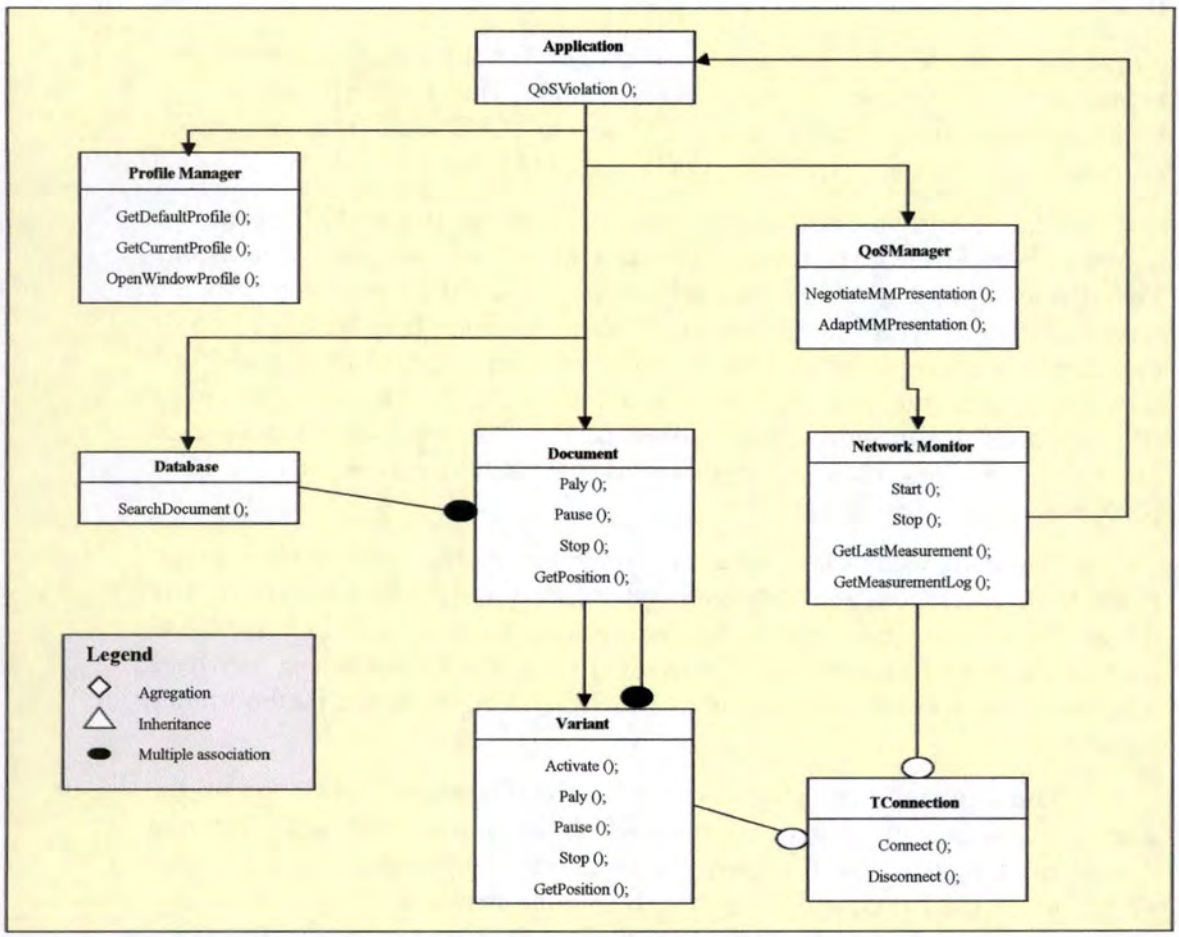

Figure 10: QoS Manager Architecture [HAFI94].

D. QoS Meta-data Modelling – Existent Approach

For the purpose of QoS management, it was essential to introduce quality meta-data associated to documents as well as to system components. In this section, we describe the QoS meta-data that were introduced in the prototype.

A description of the meta-data model used in Proto1 is hereafter introduced. In order to discuss the different meta-data types, a classification is presented:

1. Document information
2. User information
3. System information

1. Document information

A multimedia document object contains meta-data about all different types of documents. This meta-data is used to describe the characteristics of each variant of each document and in order to select which variant is best for a given user profile.

The QoS Manager manipulates document's meta-data. We present in Figure 11 the current structure of a multimedia document; meta-data will thus be extracted from this model that was developed in [HAFI96] in order to validate the next structure:

Document : *Author, Description*

Monomedia : *Type, Price*

Multimedia : *Presentation Scenario, Spatial relationships*

Variant : *Format, Size, Localization*

Image : *Color, Height, Width*

Continuous Media : *Duration*

Text : *Language*

Audio : *Sample Rate, Bits per sample, Language, Number of channels*

Video : *Color, Frame rate, Height, Width*

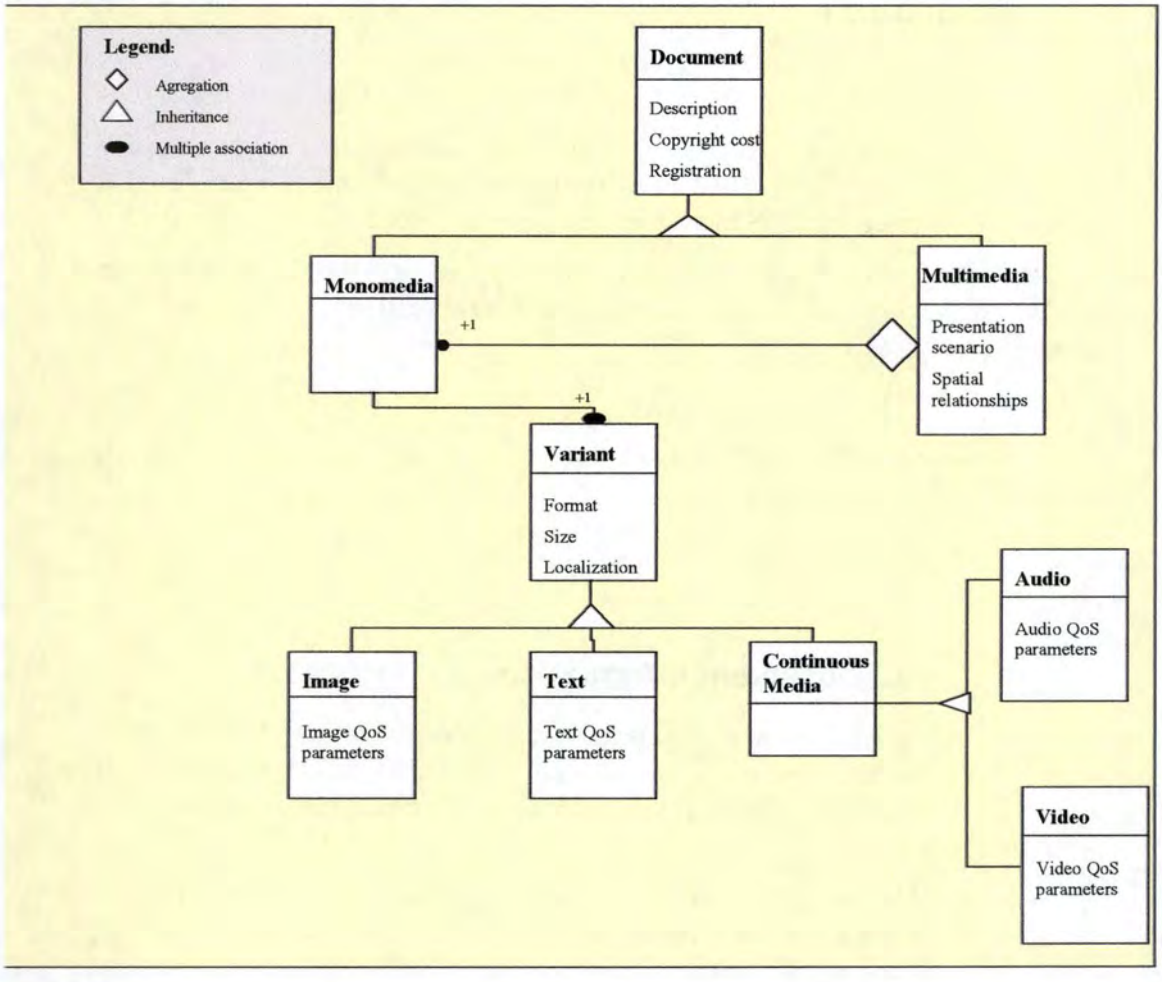

Figure 11 : Multimedia Document Model[6].

A document can be either monomedia or multimedia. A multimedia document is composed of several monomedia and has attributes, which consist of spatial and temporal synchronization constraints. Several physical representations –called variants- may exist for a monomedia object; each variant having different degrees of quality. A monomedia, and thus a variant, is defined in a particular medium: a text, a still image, an audio, graphic or video sequence.

[6] The model is presented in OMT Notation.

2. User information

An objective notification and presentation of QoS parameters is probably not a user meaningful one. From this point of view, user-friendly descriptions should be employed. In order to perform this task, we introduce the «User Profiles» that take into consideration the user perception of QoS. A User Profile is thus a set of values associated to QoS parameters, personalized by the user itself. A hierarchical schema of a user profile comprehension is hereafter presented in Figure 12:

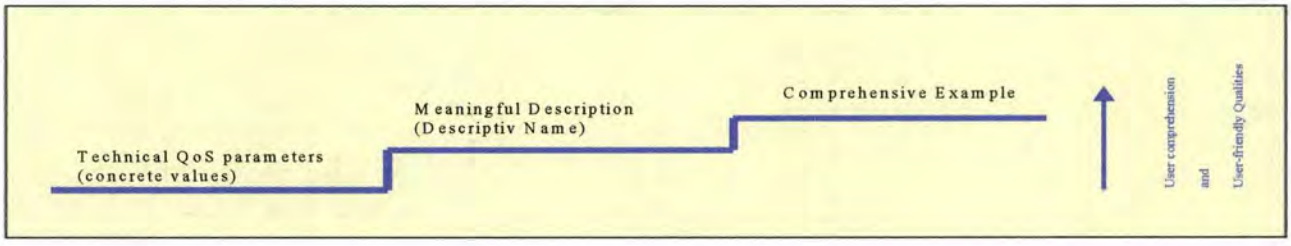

Figure 12: QoS User Profile.

Thus the aim of introducing user profiles is to efficiently permit the recognition and selection of QoS parameters. Aspects as default profiles, preference parameters or profile browser possibilities should be taken into consideration. One solution is to create several abstraction levels, which means to define different user abilities and goals in terms of QoS.

Two existing prototypes are dealing with User Profiles. Both are composed of a set of media, cost and time profiles. They differ by the way of interpreting the user profile. Let first watch the user profile model in Figure 13 and then discuss the differences between the two prototypes.

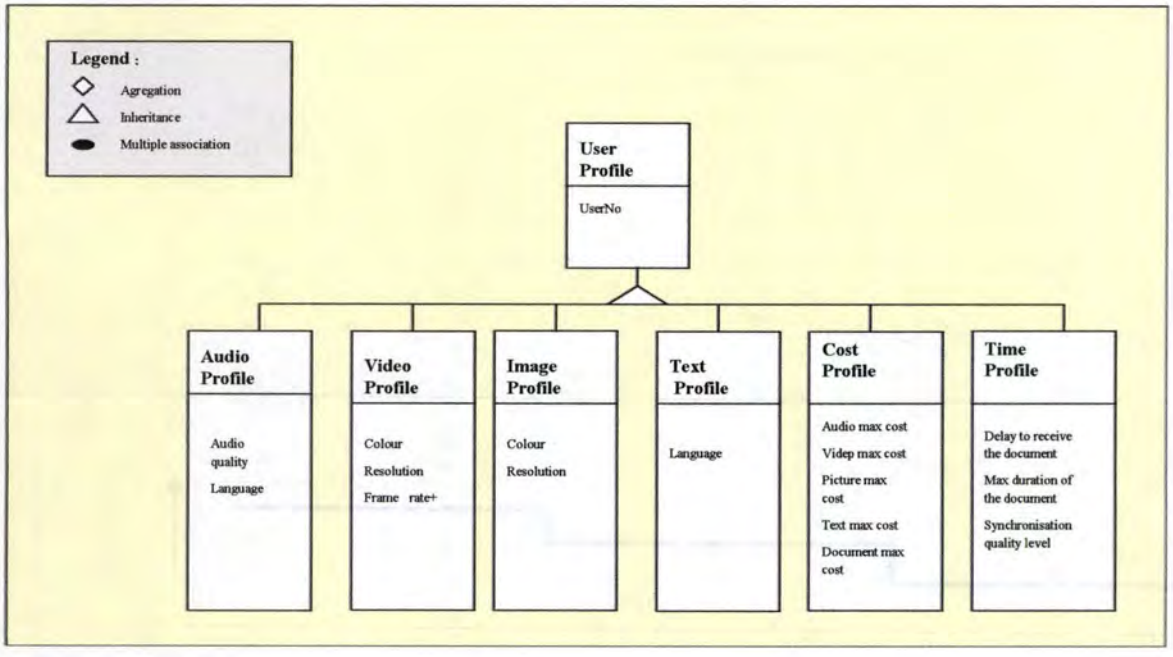

Figure 13: User Profile Model.

a) Proto1 approach

This first approach is based on an "importance factor" (IF) technique. An IF is the importance of one QoS parameter in the set of QoS parameters. The aim of an IF is to induct interactions between different QoS parameters or between the different values of a same parameter. The offer importance is the sum of importance of each IF.

The advantage of this approach [HAFI96] is that it permits to obtain the demanded QoS as well as the less acceptable offer. The user profile is thus composed of desired values, acceptable values and IF values.

b) Proto2a approach

Proto2a [MADJ97]is part of the News-on-demand prototype. Its aim is to make an extension of Proto1 for QoS management in the Internet environment (World Wide Web).

This second approach is based on a "priorities system": a priority is associated to each QoS parameter. Each medium has a set of specific QoS parameters. For example, for video media, color, frame rate and resolution, or for audio media, language and sound quality.

The user associates a value to each QoS parameter of the different media and priorities between the different parameters for a given medium. He may define different profiles for each medium or create multiple-profiles combining the values of QoS parameters for different media.

The advantage of this approach is that the priorities are globally defined on the set of QoS parameters. Classifying and selecting offers is thus much easier. The best offer between the existing ones will be then obtained through the user profile and more specifically through the priorities between the different QoS parameters.

3. System QoS meta-data

System QoS meta-data, which means system's QoS parameters, is associated to client and server machines, as well as to the transport system. Client and server machines most concern the available hardware on those machines; QoS meta-data for such components is thus considered as ***static information***. The transport system is taken into consideration especially during the negotiation phase – and particularly during monitoring. Is hence evaluated as a ***dynamic information***. Basically, the major characteristic for dynamic information is its temporal rise, function of the system charge.

In order to concretize those principles, a list of examples [GUOJ96], [FISC97] and [BOCH96] is presented hereafter in Figure 14. As far as the static characteristic is concerned, parameters such as screen device, audio device, formats, compression and costs are affected. As for the dynamic one, throughput, delay, guarantee, disk storage and costs should be assumed. Semantics of those concepts are introduced.

System Information Type	Parameters	Semantics	Examples
Static information	Screen Device	Monitor type and characteristics	• Black & White/Color • Synchronization rate
	Audio Device	Audio device type and characteristics	
	Formats	Formats supported by the component	
	Compression	Data compression is the result of redundancy elimination and human perception tolerance.	• Symmetric compression techniques (similar compression and decompression techniques) or Asymmetric compression techniques • Lossless compression techniques, eventually combined with lossy techniques • Digital audio compression techniques • Digital image and video compressing techniques • Multimedia compression standards (JPEG, CCITT, MPEG, ISO JBIG, ITU-TS H.263…)
	Costs	Costs represent the costs required to obtain the document.	• $2.4, $4, etc.

System Information Type	Parameters	Semantics	Examples
Dynamic Information	Throughput	Throughput associated with the component.	
	Delay	Transfer delay.	
	Guarantee	Guarantee represents the level of guarantee given by the system. Cost may vary in function of that level of guarantee.	• Deterministic • Statistical • Best effort • Predictive
	Disk Storage	Available space on disk.	
	Costs	Costs represent the costs required to obtain the document and it may vary in time.	• $2.4, $4, etc.

Figure 14: Static and Dynamic Components of the System Information Model.

E. QoS Manager and DBMS Interactions

This section is intended to present the current design of the QoS Manager. As developed in [HAFI96] its principal actions will be described; possible interactions with the DBMS will be commented.

1. QoS Manager Primitives

QoS Manager basic actions are here presented by the following primitives.

1. **GetAllMultimedia** (Document, List of Monomedia): Status

This routine returns a list of monomedia components of the input document.

2. **GetQoSMonomedia** (Monomedia, QoS Parameters): Status

This routine returns the QoS parameters of the input monomedia, e.g. the type of monomedia, etc.

3. **GetAllVariants** (Monomedia, List of variants): Status

This routine returns a list of variants for the input monomedia.

4. GetQoSVariant (Variant, QoS Parameters)

This routine returns the QoS parameters associated with the input variant.

5. **GetSize** (Variant, Size): Status

This routine returns the size of the input variant.

6. **GetFormat** (Variant, Format): Status

This routine returns the format of the input variant.

7. **GetSite** (Variant, Location): Status

This routine returns the location of the input variant.

8. **GetPresentationScenario** (Document, Presentation scenario): Status

This routine returns the presentation of the input document.

In order to alert any external problem, and thus to avoid a bad accomplishment of the action, each primitive is supposed to have a boolean out parameter – «*Status*» – indicating if the operation could take place and correctly achieves its target. This variable is introduced especially from a technical viewpoint.

2. QoS Manager Activities

To understand the possible interactions between the QoS Manager and the DBMS, we explain hereafter the QoS manager actions during the negotiation and adaptation steps, respectively step 3 and 7 described in Chapter II:A.1, Step definitions. The QoS manager algorithm permits us to better focalize the DBMS intervention. Then, we introduce the negotiation procedure's main steps.

Step 1: Static Local Negotiation

Step 2: Static Compatibility Checking

Step 3: Computation of System Offers

Step 4: Classification of System Offers

Step 5: Resources Commitments

Step 6: User Confirmation

The QoS manager's application programming interface consists of the following operations:

- **NegotiateMMPresentation** (Document, User Profile, Negotiation Status, Renegotiation, MM Profile) : Status

This routine determines all possible system configurations and selects an optimal one, which might support the delivery of the input document, while satisfying the user requirements.

- **AdaptMMPresentation** (Document, List of connections, Adaptation Status, MM profile) : Status

This routine allows, when QoS violation occurs, to find an alternate configuration that might support the initially agreed QoS.

The following flow diagram offers us a better understanding of the problem. Four actors are involved: the client machine, the system, the QoS manager and the DBMS.

Three preliminary hypotheses have to be mentioned:

- **H1:** "Machine Characteristics" information is supposed to be known.
- **H2:** the system is supposed to offer guarantees concerning its performances.
- **H3:** we divide the entire process into three steps: user static negotiation, negotiation and confirmation.

According to H3, the flow diagram is presented hereafter. We divided it in three phases:

Phase I: User Static Negotiation, corresponding to step 1;

Phase II: Negotiation, regrouping steps 2, 3, 4 and 5;

Phase III: Commitment Confirmation, corresponding to step 6.

a) Phase I: User Static Negotiation

This Phase I (Figure 15) deals with the very first negotiation steps. We speak here about the client machine characteristics and the user document request.

At this level we observe that the client document request is independent of user profiles. While the aim is to access the chosen document, the user profile is compared with machine characteristics in order to obtain an agreement between them.

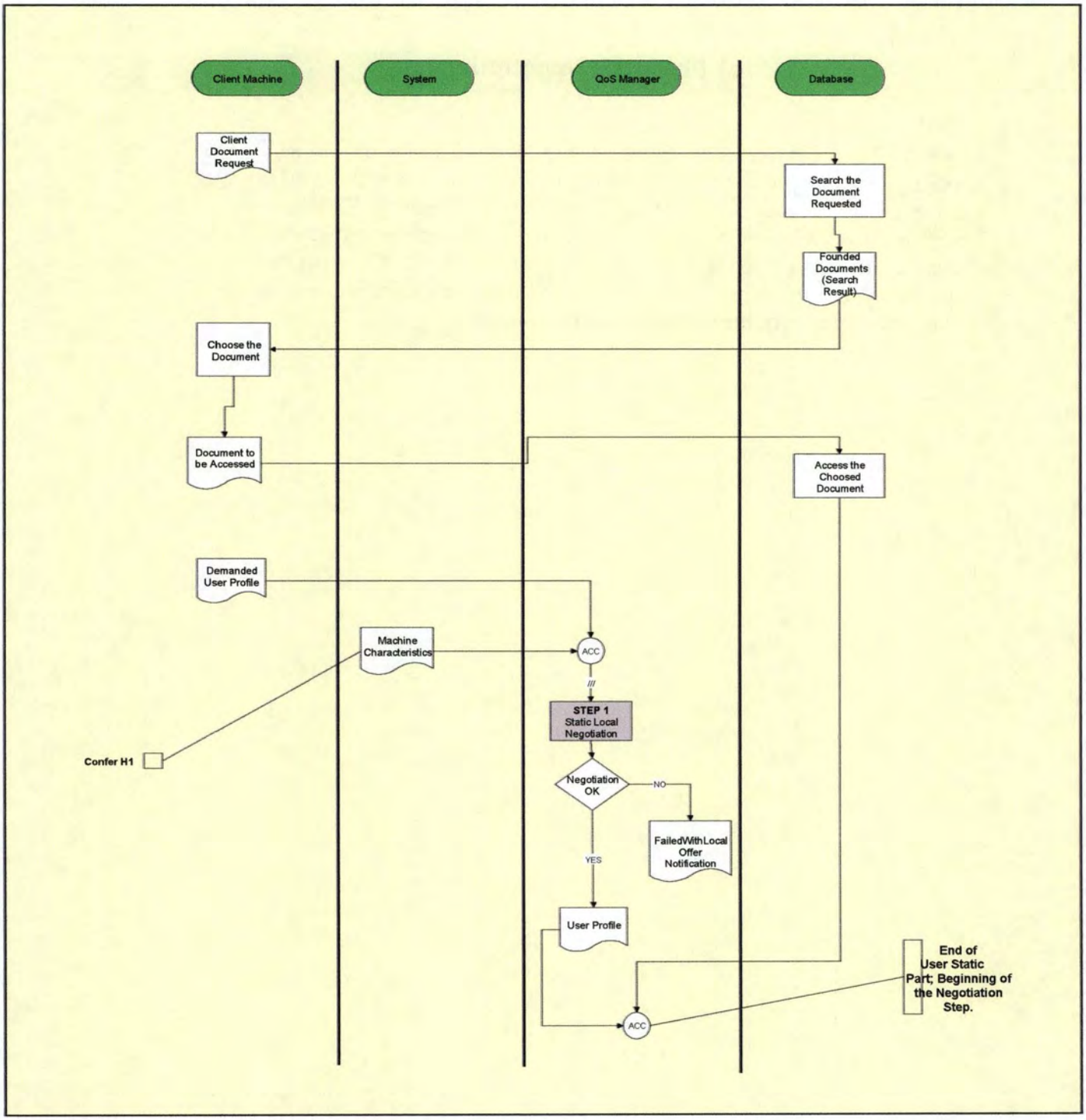

Figure 15: Phase I – User Static Negotiation Flow Diagram.

b) Phase II: Negotiation

The main effort is performed during Phase II (Figure 16), while the QoS manager tries to present a satisfying offer. The entire job is done by the QoS manager during four steps: Static Compatibility Checking, Computation of System Offers, Classification of System Offers, Resources Commitments. Once Step 2 successfully passed, the importance factor of QoS parameters is introduced and the step 3 is proceeded. Then suits classification and ordering. This phase ends once the resources commitment starts.

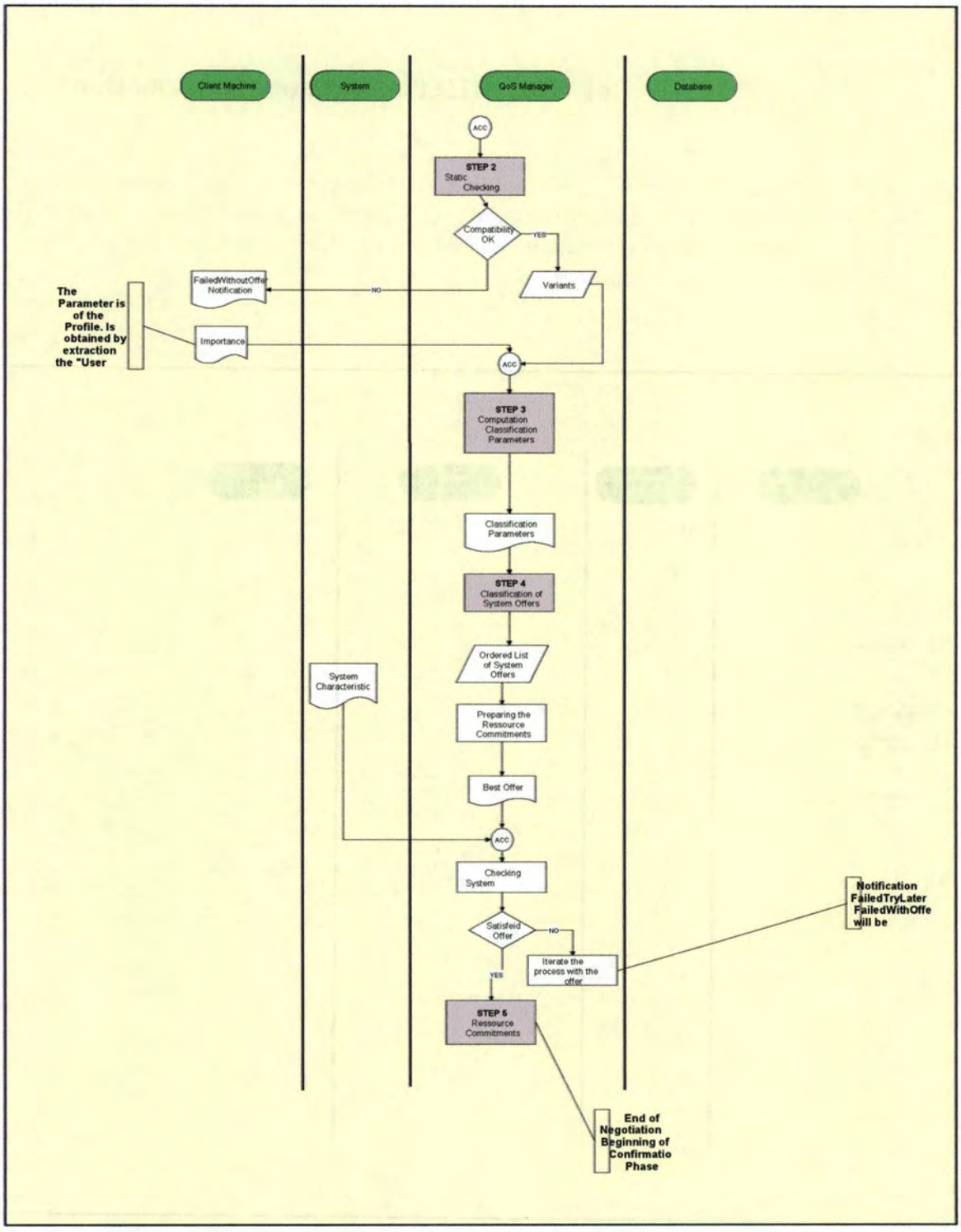

Figure 16: Phase II – Negotiation Flow Diagram.

c) Phase III: Commitment Confirmation

Phase III (Figure 17) corresponds to the last part of the negotiation step. Once the negotiation succeeded, the user is asked for a confirmation. In case of a positive answer, the document will be played, else, a rejection takes place and the resources are deallocated.

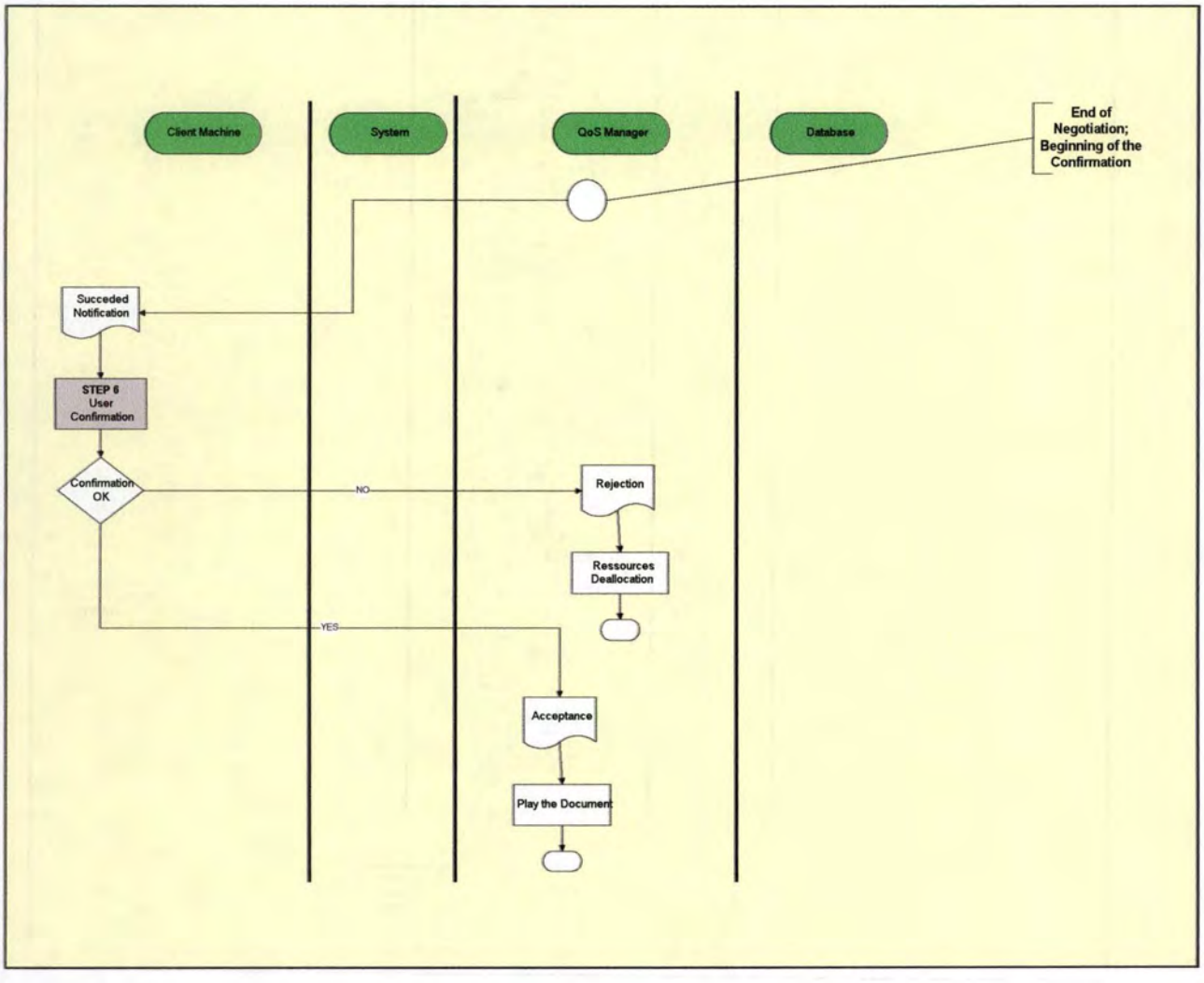

Figure 17: Phase III – Commitment Confirmation Flow Diagram.

F. Proposals

In this section we are presenting different possible enhancements for the existing prototypes. Problems and insufficiencies at levels of document information, user profiles and interactions between DBMS and QoS manager are discussed.

1. Document Meta-data

Regarding the document meta-data, there is no major improvement to be performed. A possible amelioration is to increase the number of QoS parameters taken into consideration.

2. User Profiles

Concerning the user profiles, a major problem can be solved. In fact, the user encounters a comprehension problem in "decoding" the significance of each QoS parameter. The degree of correct comprehension depends on the experience level of the user. In that light, different levels can be introduced in a way that each level detains different explanations for a same QoS parameter. For example, a high experienced level may give only the name of the QoS parameter, while the less experienced might even show an example.

Still regarding the use, the guidance of the user should be considered as highly important. In both actual prototypes, the user is provided an exhaustive list of parameters which he must order either by the importance factor, either by priority. Dividing this QoS parameters list into one "Important List" and one "Auxiliary List" allows the user to advance faster in his research: only needed parameters will be chosen.

3. DBMS Interactions

The QoS manager is having a heavy job in order to classify and order the search results. Using relational DBMS permits some internal database mechanism to deal with the sort and classification procedures and thus facilitates the QoS manager algorithms. The database systems can thus process QoS-based queries.

Chapter III: General Presentation of our Solution

This chapter presents three solutions we propose as amelioration for the present prototype. The first one is a "Client QoS-based queries" solution, while the next ones are "QoS-based queries" and a "View creation according to QoS profiles". We discuss principles, advantages and drawbacks of each solution, the final goal being to demonstrate the possibilities of those approaches. A prototype implementing one of those solutions is also developed. We finally justify the choice of the implemented solution.

A. Database internal mechanisms

While querying multimedia databases, we separate two distinct phases: the *search* and the *access* phases. The *search* phase consists in isolating a set of documents of potential interest for the user. Among these documents only those specified by the user during the *access* phase will be completely delivered. [HERH96] These different steps are required to avoid the delivery of large amount of data that are not of real interest to the user. During the search phase, the information delivered to the user is the meta-data describing the content of the document. These meta-data allow the user to select the document he wants to access.

While the search phase major goal is isolating a set of documents, we discuss the introduction of the "view" concept. We can think of a view as a way of specifying a table that we need to reference frequently, even though it may not exist physically.

A view in SQL terminology is a single table that is derived from other tables [ELMA94]. These other tables could be base tables or previous defined views. A view does not exist in physically form; it is considered in a virtual table, in contrast to base tables that can be applied to views, but it does not provide any limitations on querying a view.

B. Solution 1: Client QoS-based Queries

1. Principle

Phase I of the flow diagram, presented in Figure 15, shows that the first search result depends on the client machine characteristics. In fact, during the search phase, we only search for documents compatible with the Client machine QoS constraints. During the access phase, for the selected document, the QoS manager asks for QoS meta-data sorted according to user QoS profile. The QoS manager will then inspect this meta-data set in order to elect the best variant for each monomedia.

We then propose that the user query is enriched with client QoS information. By client QoS information we understand all parameters that may concern the client machine. In that light, the concept of view is interesting and we may define one corresponding to the client machine QoS constraints. The resulting set of documents satisfies though the client QoS constraints. This excludes the first step of the existent prototype, the "Static Local Negotiation".

The next step consists of selecting all documents satisfying the search request criteria and having at least, for each monomedia, one variant satisfying the QoS client-machine constraints. We enter now the phase where the user chooses the document to be accessed. Once this document selected, the proper negotiation (Phase II of the flow diagram) begins. At this level the QoS manager queries the database to get QoS information on all variants of all monomedia of this document according to sort criteria. The sort criteria express some constraints described in the user QoS profile like color, resolution, format, size, language, frame rate, bit rate, etc.

We now arrive at step 3 of the negotiation. At this moment, the principal process to do is the classification. In fact, the user has already selected a document for access and transfer; it rests only to select all QoS meta-data for all variants of all monomedia. Those variants should also satisfy the QoS client machine constraints. The result is sorted by monomedia and by the QoS constraints expressed in the user profile and presented to the user for confirmation.

2. Advantages

The major advantage of this solution is that the static local negotiation is processed by the database system. Exonerate the QoS manager of that first step permits the user to start the access phase on a smaller set of documents.

Besides this advantage, which is clearly perceptive from the user point of view, the "efficiency" point of view has to be regarded. While classification is directly performed on the database, the QoS manager is free to deal with other jobs.

3. Drawbacks

There is no known major drawback but for the fact that a user cannot search a document for another machine.

C. Solution 2: QoS based Queries (Client and User profile)

1. Principle

This second approach is based on introducing more complete queries in order to better focalize the user request. Adding QoS information to the queries restrains the result domain and gives the user a better offer. We regard here principally phases I (Figure 15) and II (Figure 16) of the flow diagram. The document list – result of the search phase – satisfies the client machine constraints and is ordered according to QoS parameters expressed in the user profile. While preparing the result, the database system sorts documents to facilitate further work done by the QoS manager.

Enriching the user queries with client machine QoS information as well as with sort criteria corresponding to user QoS profile permits to facilitate the negotiation task. The first consequence of that solution is similar of the one of the precedent solution: the resulting set of documents satisfies the client machine QoS constraints. Thus, static local negotiation is no more required. Also, the resulting set of documents is sorted according to QoS constraints expressed in the user profile, i.e. color, resolution, format, size, language, frame rate, bit rate, etc.

The amount of constraints is quickly increasing. A logical implication is that it requires expressing and executing more than one SQL query. In this solution, the principle is to give the more information to the database system to allow to efficiently process queries based on QoS characteristics.

The DBMS first selects all documents satisfying the search request criteria and having at least, for each monomedia, one variant satisfying the QoS client machine constraints. Next, the DBMS sorts the documents according to the QoS criteria expressed in the user QoS profile.

2. Advantages

The exoneration of the static negotiation permits to facilitate the QoS manager task. In the meantime, the result is ordered and then presented to the user. That means the user has QoS information to select the document he wants to access. In that way the database system prepares part of the work for the QoS manager and the user can benefit of better selections.

3. Drawbacks

The efficiency of this approach can be slowed down in some circumstances. When the user request is not very selective – which means the result contains a lot of documents – this approach is not very efficient since the query is complex and requires a lot of process-time.

D. Solution 3: View Creation according to QoS profiles

1. Principle

This approach deals principally with user profiles. Its basically idea is that a user profile is to be reused – at least all along the current session. Speaking of reusability implies using views. When the user defines a QoS profile, the QoS manager asks for a view creation. This relational view virtually contains all the documents satisfying the QoS constraints expressed in the user QoS profile.

A sort criterion is also defined in order to sort the documents according to the priorities specified between parameter values. When the user expresses a query on the document database, the QoS manager modifies this query in order to process it on the corresponding relational view. The flow diagram suffers then a major task delegation from the QoS Manager to the DBMS.

This solution is thus a two-step one. First, the user QoS definition determines the creation of a "personal" relational view, as well as a query transformation. In fact, the query transformation corresponds to a query on the relational view and the client machine constraint criteria. The second step concerns the access. From that point of view, the result is sorted according to the QoS.

2. Advantages

The advantage of this solution consists in the simplicity of the query modifications. In fact, there is no necessity to execute complex query modification for each query formulation.

3. Drawbacks

The major contribution of this solution is the profile views creation. Or, that implies that one view per profile must be created. The higher the users number is, the less interesting our solution is. We conclude that this last solution can be outstanding for "stable" users, users that reutilize theirs profiles. In the meantime, the average number of user has to be taken into consideration while accepting this solution.

E. The final selection

At this moment, we introduce a new presentation of our prototype. This time, it is more concrete and detailed. Our Prototype (Proto3) takes into consideration only one of these solutions. Its goal is to implement this solution and to practically prove the acceptance of this principle. Further work will demonstrate and compare the performances of two other solutions.

Between solutions presented above, the first one seems to be more productive. In fact, we consider that the drawbacks of solutions two and three might be so important in a distributed system and drawback of solution one is pretty rare, that the Client QoS-based queries (first solution) was finally chosen to be implemented. Our approach is purely theoretical and the detailed demonstration can be obtained only after implementation of the three solutions. A practical comparison in terms of performances is thus needed.

This proposal implies major improvements in the flow diagram. We propose hereafter the reviewed flow diagram of the negotiation process. The three phases – user static negotiation, negotiation and commitment confirmation – compose the negotiation algorithm.

1. Phase I: User Static Negotiation

During this Phase I, the negotiation is situated at the user level. The user request and the client machine characteristics are taken into consideration. The major change presented at this level stands at the very first database request. In this approach, the client machine characteristics are already taken into consideration.

Comparing with Proto1 solution, Step 1 is almost insignificant now.

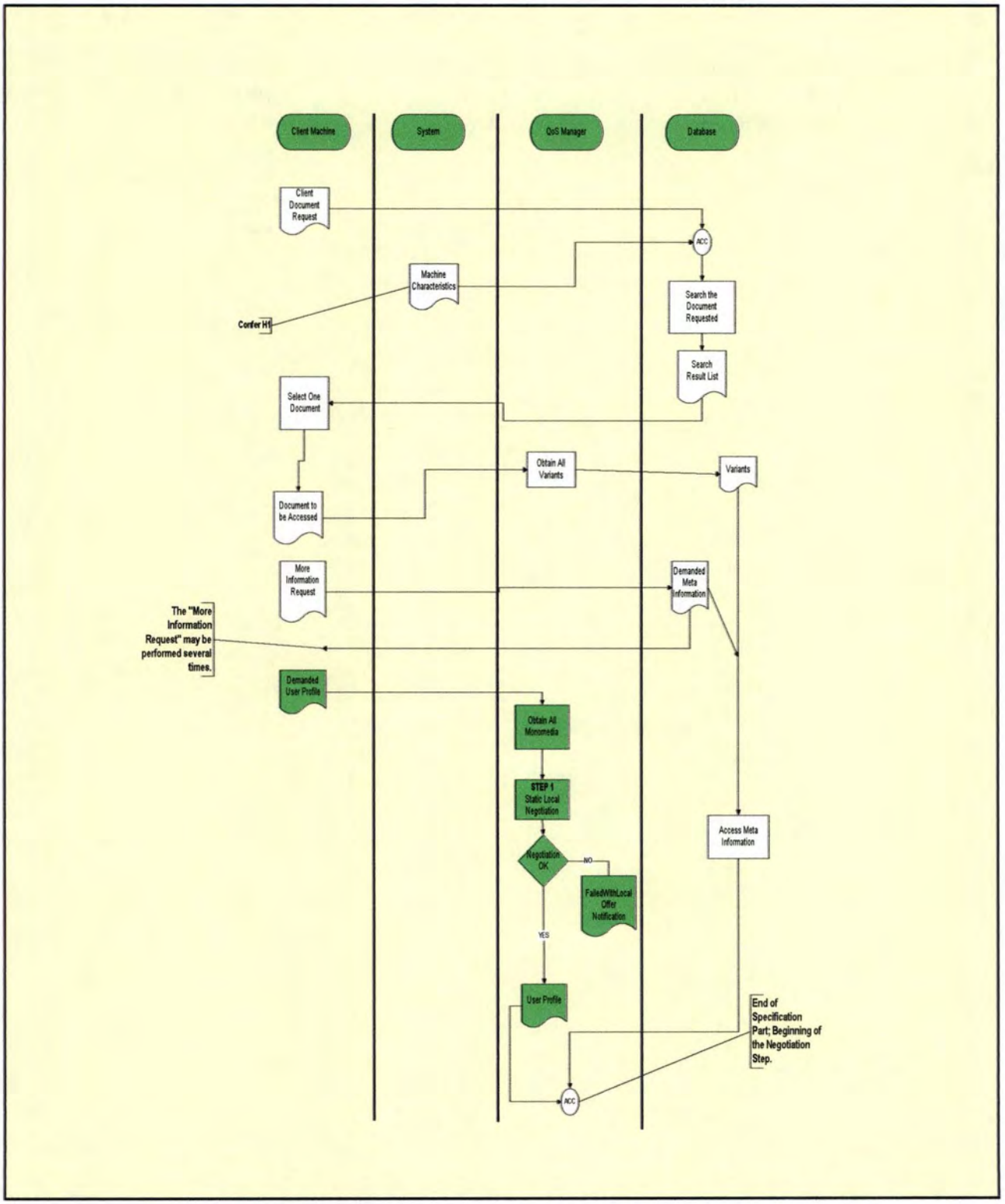

Figure 18: Phase I – New User Static Negotiation's Flow Diagram.

2. Phase II: Negotiation

The major goal of this Phase II is to classify the offered variants. In that light, the database will realise steps 3 and 4, which are mainly classification procedures. In that way, the most important part of this phase was transferred to the DBMS.

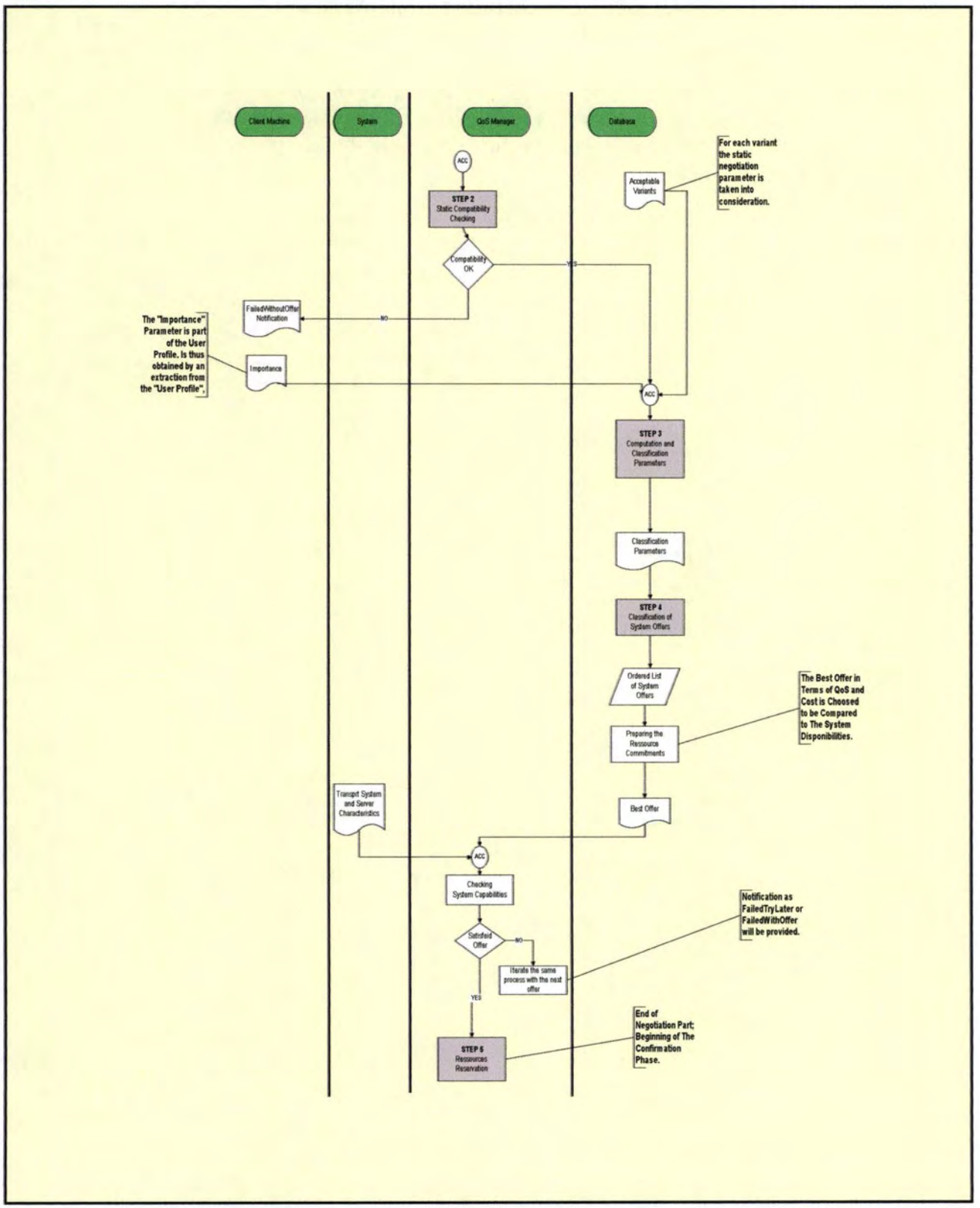

Figure 19: Phase II – New Negotiation Flow Diagram.

3. Phase III: Commitment Confirmation

This last phase, Phase III, is principally the same as in the previous version. No interaction between the DBMS and the QoS manager had been identified. At this level, only the acceptance of the user is interesting.

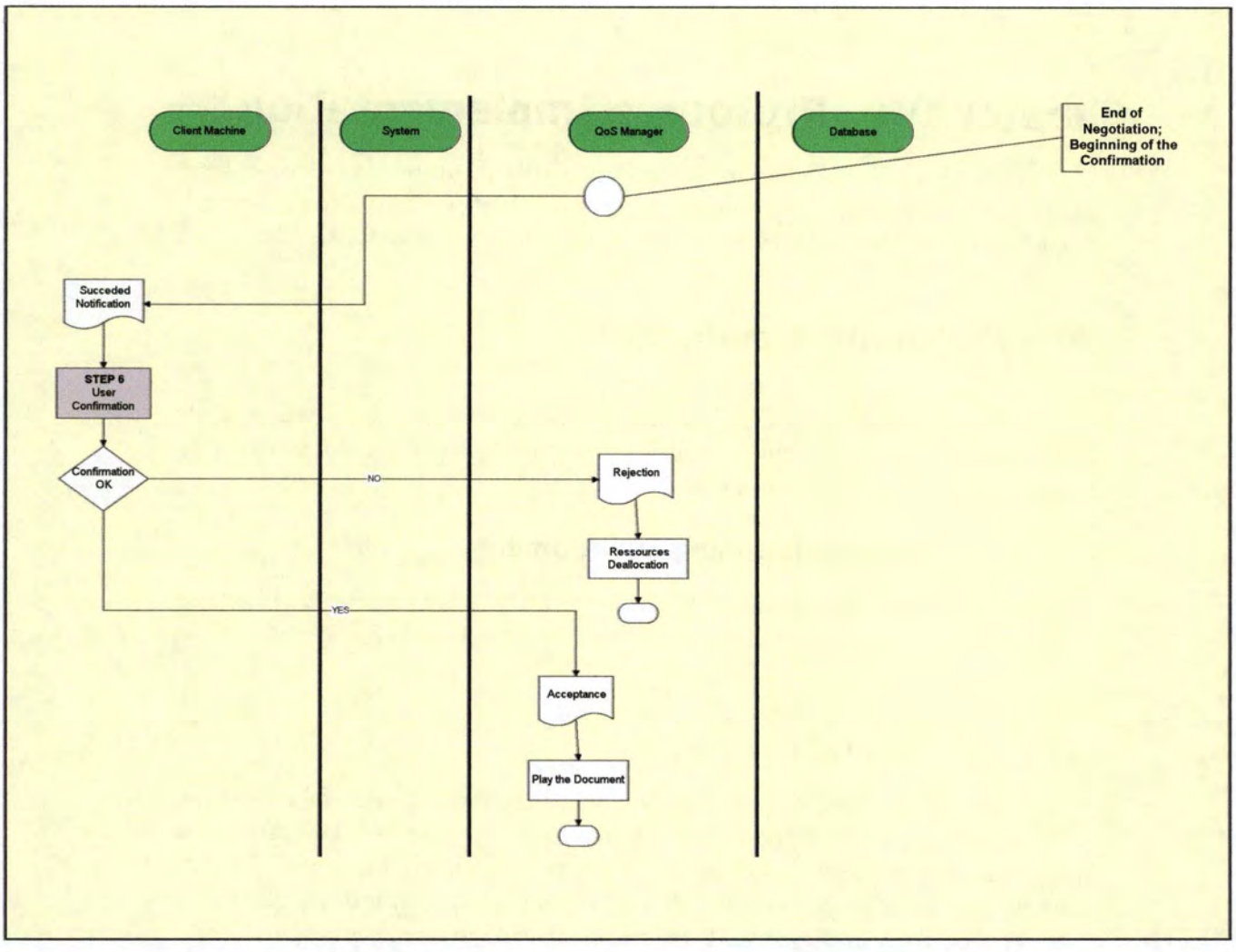

Figure 20: Phase III - Commitment Confirmation Flow Diagram.

Our prototype is developed as corresponding to those phases.

Chapter IV: Prototype Implementation

This chapter is dedicated to the implementation features of our prototype. First, the architecture is presented; next, the database creation is detailed. Implementation comments as well as tests procedures are described.

A. Prototype Architecture

Speaking about program architecture means directly speaking about module design. We introduce at this step the programming environment in order to familiarize the reader with the work context.

1. Programming Environment

The principal actor in our prototype is the DBMS. We choose the Interbase SQL version 2.0 and the programming environment is Delphi 2.

2. Modules

The architecture of our prototype is mainly divided into eight modules. There is one coordinator module which manages – via the user interface – the three treatment modules: UserProfiles, DocumentSearch and Priorities. The data level is represented by SystemInfo, ProfileInfo and DocumentInfo modules. Use relation described in **Figure 21** by arrow assures the chaining up of other modules.

A briefly description of each module is hereafter presented. In order to preserve comprehension, we only describe the general lines of each module; parameters and additional procedures are not detailed.

The UserInterface module aims to introduce interface primitives permitting opening, closing and changing from a window to another. It permits links between the user and the DBMS and it covers all HCI actions.

The DocumentSearch module is referring to three important steps: the search of the document, the selection and final description of the request's (access') result.

The Priorities module is divided in four different profiles: audio, video, text and image and is managing the priorities between the different QoS parameters.

The UserProfiles module regroups all process concerning the four types of profiles.

At the data level, the SystemInfo module concerns data about the client machine, the continuous-media file and the network. Methods dealing with profiles are concerted into the ProfileInfo module, while information about document is in DocumentInfo module.

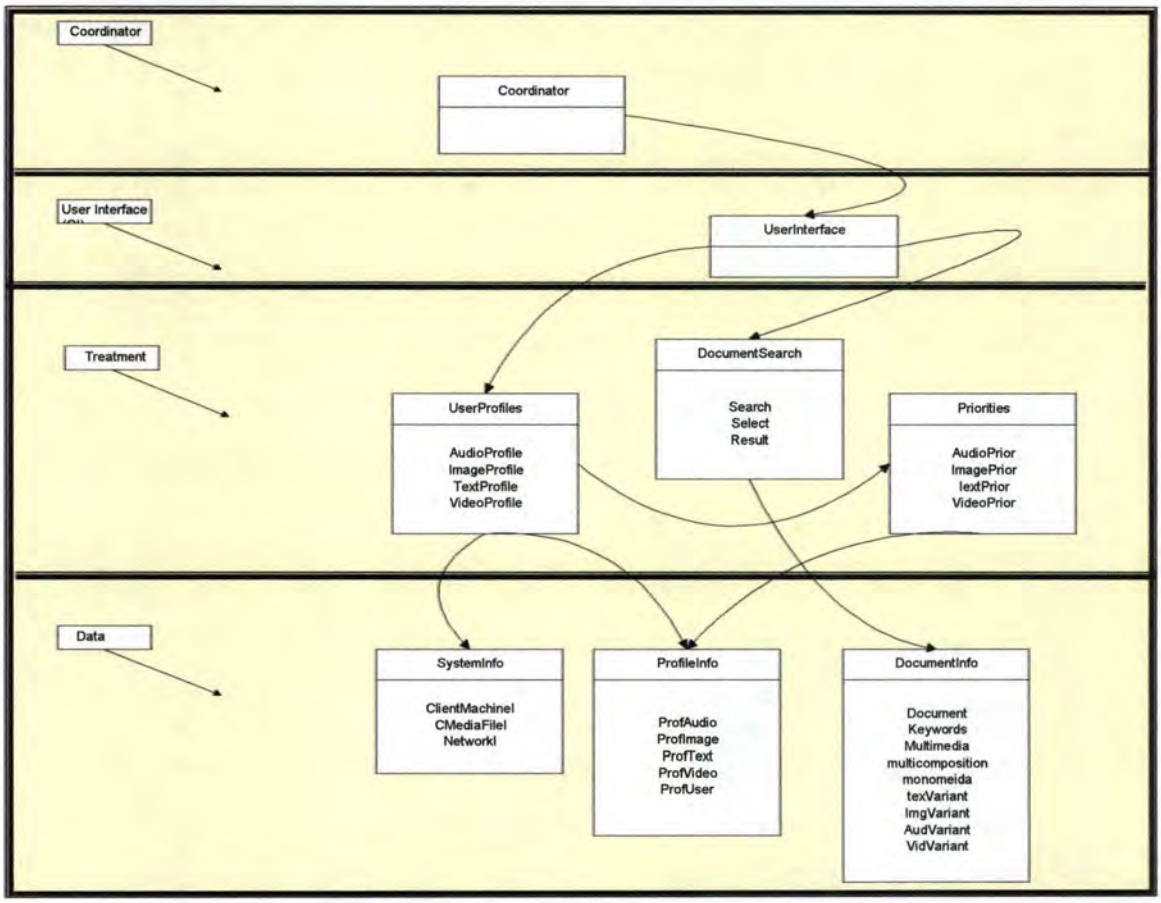

Figure 21: Architecture.

B. Database Scheme for the Prototype

In this section we present the design of the simulation's prototype database. We begin by describing all data types. Next, document's, profile's and system information's databases are presented. Finally, mappings and priorities between enumerative types are discussed.

A briefly reminder about the needed database structure is hereafter presented. The QoS manager is concerned to deal with QoS parameters at different levels. First of all the principal user goal is to retrieve a document. In that light a defined structure of a document is needed. In our context, the concept of QoS management is an end-to-end concept so that any system

information is pretty important. The next important level of information is concentrated around the profiles: audio, video, text and image.

1. Data Types

The data types described hereafter correspond to the data information contained in the database. The enumerative types are the most interesting at this moment because they permit to show the range of values that a data can have.

```
TColor       = SET {super_color, color, black_white,
               grey_scale, no_color}
TFormatTxt   = SET {no_format, word95, word97, wordperfect,
               wps, asc, ans, mcw, dot, txt, rtf, ms-dos, htm,
               html, htx}
TFormatIm    = SET {no_format, gif, jpg, pcd, pcx, png, raw,
               tif, wmf, wpg}
TFormatAud   = SET {no_format, mp3, wav}
TFormatVid   = SET {no_format, mpg, avi, vdo, mov, ra}
TGuarantee   = SET {guarantee, best_effort}
TLanguage    = SET {no_language, english, french, german,
               spanish, dutch, arabic}
TLocalisation      = INT
TName              = CHAR
TPriority          = INT
TProfType          = SET {TextPr, ImagePr, AudioPr, VideoPr}
TQuality           = SET {no_audioQ, cdQ, phoneQ}
TResolution        = INT x INT
TType              = SET {MonoMedia, MultiMedia}
```

2. Document's Database

The **Document** contains information on its name, date, author, type – monomedia or multimedia – and abstract.

```
Document ( DocId: INT,
DocName: CHAR,
DocDate: DATE,
DocAuthor: CHAR,
DocType: TType,
DocAbstract: CHAR) 0
```

The **Keywords** structure is dedicated to permit the user a more detailed search. Each keyword is directly dependent of a document.

```
Keywords ( DocId: INT,
Kwd: CHAR)
```

Monomedia is the linker between a document and the monomedia contained in.

```
Monomedia ( MonoId: INT,
DocId: INT)
```

Multimedia and **MultiComposition** are two structures that are not implemented in the present prototype. We introduce them, so that they show one of the further steps of this project, the simulation for several monomedia objects, i.e. a multimedia object. **MultiComposition** is supposed to make the link between different monomedia and a multimedia, while **Multimedia** contains information on the spatial and temporal organisation of the multimedia, as well as the link to the document containing this object.

```
MultiComposition (MultiId; INT,
MonoId: INT)
```

```
Multimedia (      MultiId: INT,
DocId: INT,
SpatialOrg: CHAR,
TemporalOrg: CHAR)
```

Each monomedia might have different variants. Four types of variants are taken into consideration: audio, image, text and video. For each one of them we consider the most important parameters.

```
AudVariant (      AudVarId: INT,
Language: TLanguage,
SampleRate: INT,
BitsPerSample: INT,
SoundQuality: TQuality,
Format: TFormatAud,
Size: INT,
FileName: TName,
IdSite: TLocalisation)
```

```
TexVariant (      TexVarId: INT,
Language: TLanguage,
Format: TFormatTex,
Size: INT,
FileName: TName,
IdSite: TLocalisation)
```

```
ImgVariant (      ImgVarId: INT,
MonoId: INT,
Color: TColor,
Resolution: TResolution,
Format: TFormatIm,
Size: INT,
FileName: TName,
IdSite: TLocalisation)

VidVariant (      VidVarId: INT,
Color: TColor,
Resolution: TResolution,
FrameRate: INT,
BitRate: INT,
Format: TFormatVid,
Size: INT,
FileName: TName,
IdSite: TLocalisation)
```

3. Profile's Database

The four profiles – audio, image, text and video – are described hereafter. User profiles collect user's preferences [ISNA95]. In that light, all parameters regarding profiles are treated in this section. However, they will be implemented in two different steps. The first step is taking into consideration the most significant QoS parameters, while the second one offers a detailed variant in terms of parameters. Our prototype is dealing with the first step. The most significant parameters are taken into consideration. Our choice in determining those parameters is based on the frequency of utilisation. While introducing the second step (parameters presented in italic characters), the order should thus also be based on a frequency of utilisation.

Each parameter has its proper priority that serves to offer a better result to the user demand.

The **ProfAudio** is dealing with the QoS parameters concerning the audio variants. Thus quality of sound and language were decided as the most important ones.

```
ProfAudio( IdProf: INT,
QualityDesired: TQuality,
QualityWorst: TQuality,
QualityPriority: TPriority,
LanguageDesired: TLanguage,
LanguagePriority: TPriority,
FormatDesired: TFormatAud,
FormatPriority: TPriority,
BpsDesired: INT,
BpsWorst: INT,
BpsPriority: TPriority,
SampleRateDesired: INT,
SampleRateWorst: INT,
SampleRatePriority: TPriority)
```

ProfImage concerns the image QoS parameters. In that light, color and resolution are the most significant ones.

```
ProfImage( IdProf: INT,
ColorDesired: TColor,
ColorWorst: TColor,
ColorPriority: TPriority,
ResolutionDesired: TResolution,
ResolutionWorst: TResolution,
ResolutionPriority: TPriority,
FormatDesired: TFormatIm,
FormatPriority: TPriority)
```

ProfText deals with text parameters. We describe here the language and the format.

```
ProfText(   IdProf: INT,
LanguageDesired: TLanguage,
LanguagePriority: TPriority,
FormatDesired: TFormatTxt,
FormatPriority: TPriority)
```

ProfVideo is dealing with parameters as color, resolution and frame rate.

```
ProfVideo(  IdProf: INT,
ColorDesired: TColor,
ColorWorst: TColor,
ColorPriority: TPriority,
ResolutionDesired: TResolution,
ResolutionWorst: TResolution,
ResolutionPriority: TPriority,
FrameRateDesired: INT,
FrameRateWorst: INT,
FrameRatePriority: TPriority,
BitRateDesired: INT,
BitRateWorst: INT,
BitRatePriority: TPriority,
FormatDesired: TFormatVid,
FormatPriority: TPriority)
```

The user profile **ProfUser** links a user to a type of profile.

```
ProfUser(   IdUser: INT,
IdProf: INT,
ProfType:TProfType,
Date: DATE)
```

4. System Information's Database

To provide QoS for a given activity in the context of an application involving remote access to a multimedia database, a set of system components are involved [BOCH96]:

- ⇨ The User Machine and the Presentation Device
- ⇨ The Continuous Media File Server
- ⇨ The Network

The **ClientMachineI** characteristics are very important during the first phase of the negotiation. Parameters as screen size, resolution, color, sound quality, delay, throughput, etc. are important in permitting a better target offer. In the meantime, the network performances are needed (**NetworkI**) as well as the continuous media file server ones (**CMediaFileI**).

```
ClientMachineI (IdClient: INT,
ScreenCharactSize: INT ,
ScreenCharactResolution: TResolution,
ScreenCharactColor: TColor,
SoundCharact: TQuality,
Delay: INT,
Reliability: BOOL,
Jitter: INT,
Throughput: INT,
GuaranteeType: TGuarantee)
```

```
CMediaFileI(IdServer: INT,
CmFileReliability: BOOL,
CmFileThroughput: INT)
```

```
NetworkI (Site1: TLocalisation,
Site2: TLocalisation,
NetReliability: BOOL,
NetThroughput: INT)
```

Annex 1: Database Creation Script presents the complete script for the database creation.

5. Mappings

This section describes basic correspondences between the meanings, importance, order relations and links concerning different parameters.

a) Order relation for different parameters

In order to define a strict order relation between the different values of one parameter, an importance factor is allocated to each value of each type of parameter.

We thus present the order relation for color and sound quality:

(1) The "***color***" parameter (Table 4)

Colour Quality	Level of Quality
	(best quality)
Super_Color	5
Color	4
Grey_scale	3
Black_white	2
No_color	1
	(lowest quality)

Table 4: Color Parameter.

(2) The "***sound_quality***" parameter (Table 5)

Sound Quality	Level of Quality
	(best quality)
CdQ	3
PhoneQ	2
No_audio_quality	1
	(lowest quality)

Table 5: Sound Parameter.

The "***format***" and respectively "***format_aud***", "***format_img***", "***format_txt***", "***format_vid***" parameters do not have a specific order relation; there is only a compatibility semantics.

Identically, the "***language***" and respectively "***language_desired***", " ***language_worst*** " parameters do not have a specific order relation; there is only compatibility semantics.

b) Mapping Client-Machine / Variant

Table 6 is aimed to help the construction of views while we introduce a new client machine in the network. This table represents the correspondences between the client machine parameters and the document characteristics and is helpful in creating the comparison criteria.

Client Machine SI_CLIEN_MACHINE	Text Variant TXT_IMG	Image Variant VARIANT_IMG	Audio Variant VARIANT_AUD	Video Variant VARIANT_VID
ID_CLIENT				
SCREN_SIZE				
RESOLUTION1		RESOLUTION1		RESOLUTION1
RESOLUTION2		RESOLUTION2		RESOLUTION2
COLOR		COLOR		COLOR
SOUND_QUALITY			SOUND-QUALITY	
FORMAT_AUD			FORMAT	
FORMAT_IMG		FORMAT		
FORMAT_TXT	FORMAT			
FORMAT_VID				FORMAT
THROUGHPUT	non-pertinent	non-pertinent	non-pertinent	non-pertinent
RELIABILITY	non-pertinent	non-pertinent	non-pertinent	non-pertinent

Table 6: Mapping Client-Machine / Variant.

6. Mapping User Profiles / Variant

Concerning the User Profiles' specification, it is required to establish correspondence criteria between the profile constraints and the document characteristics, here represented by variant's parameters. This correspondence is inevitable due to the need of sort criteria.

The four profiles: text, image, audio and video are discussed hereafter.

(1) Text (Table 7)

User Text Profile	Comparison Criteria	Text Variant
{Language_desired ∪ Language_worst}	⊇	Language
{Format_desired ∪ Format_worst}	⊇	Format
[VarSize_worst…VarSize_desired[7]]	⊃	Varsize
Id_prof		Non-pertinent
Non-pertinent		Filename
Non-pertinent		Site-Id
Non-pertinent		Tex_var_id

Table 7: Text Profile.

We visualise this parameter by presenting (Figure 22) the Text Profile's HCI as developed in our prototype.

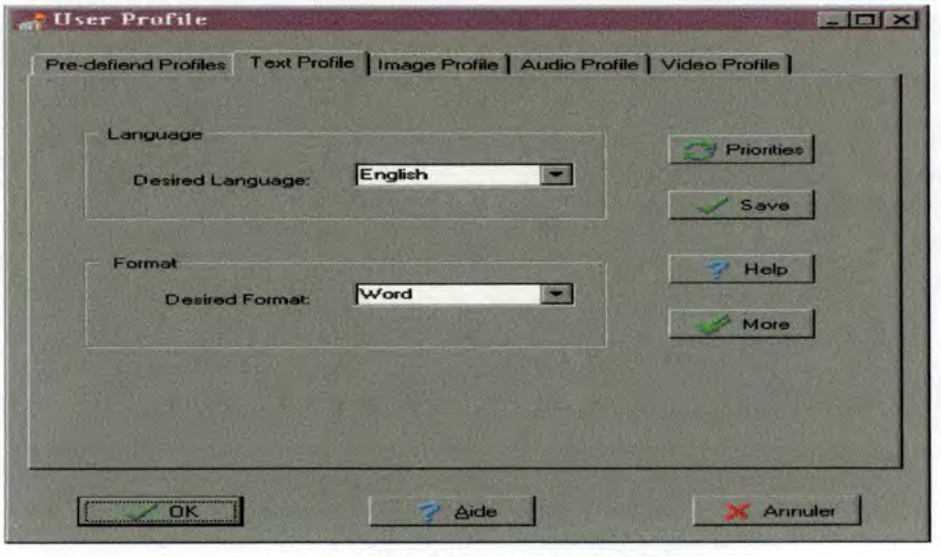

Figure 22: Text Profile HCI.

[7] There is one of the "supplementary" parameter –including the "desired" and the "worst" values- that could be found by a detailed profile description (i.e. the "MORE" function-button).

(2) Image (Table 8)

First of all, a brief description of image's characteristics is necessary. We propose to take into consideration the following image parameters: resolution, colour, size and format. In order to fix the importance, the utility and the use of each parameter, different kinds of "images" will be studied. To date, two kinds of images can be identified: *Vector graphics* and *Bitmap graphics*.

The first ones are not resolution dependant and thus can be scaled to any size without degrading image quality. In that light, the "resolution" side of Profiles or Variants is not interesting and does not take a place in the mapping between "User profiles" and "Document variants".

As far as the Bitmaps Images are of interest, we can speak about "pixel based" images. That means, the colour range is broken down into square pixels of a single colour. The combination of these pixels – side by side – is what creates the appearance of continuous tone. These images are resolution dependent and the image quality will degrade if they are scaled too large.

The "Resolution" parameter needs more explanation. Bitmap image resolution refers to the spacing of pixels in an image and is measured in pixels per inch (p.p.i.); the printing resolution is measured in dots per inch (d.p.i.[8]). We cannot establish a relation between p.p.i.'s and d.p.i.'s in practical uses. The higher the resolution is, more pixels in the image are. Different sides of the resolution concept can be treated at this point.

We basically speak about an "absolute resolution" [urlHYPERSTAND]; it represents the resolution since it is independent of how it is displayed. So, an image with 384 x 280 pixels "absolute resolution" represents that the image consists of 384 pixels by 280 pixels.

But we can also speak about a "scanning resolution" or a "display resolution"[urlHYPESTAND]. In that case the size of the image is an active factor in calculating its resolution. Image size refers to the physical dimensions of an image. Because the number of pixels in an image is fixed, increasing the size of an image decreases its resolution and vice-versa, decreasing its size increases its resolution.

Let us concretise these notions by an example [urlPRICETON]:

[8] The optimal printing resolution (dpi) is twice the line screen (lpi); as far as the size can activate on the resolution too, the next formula should be used: dpi = 2 x lpi x % of the original size.

1. The initial size of an image is: 8 x 10 (like in photography).
2. The scanning resolution is at, for example, 300 dpi, so the resulting number of pixels in the digital image is: 8 x 300 by 10 x 300, that means 2400 x 3000 pixels.
3. The display resolution, when we use a 72 dpi computer screen, is: 2400/72 by 3000/72, that means 33.33 by 41.66 inches.

We conclude that, the higher the resolution is, the higher the quality of image is. To "count" the resolution of an image, we consider the absolute resolution and we multiply the two values. For example, our 384 x 280 image would gave us a "comparison resolution value" of 384 * 280 = 107 520 pixels.

The "Colour" parameter is represented by the next values: Super_Color, Color, Grey_scale, Black_white, No_color. As far as this parameter is concerned, the set of possible values is ordered. That means the next order relation can be used as comparison criteria:

`Super_Color > Color > Grey_scale > Black_white > No_color.`

We suppose that the user worst desired colour value is the inferior limit of the possible value and in the meantime the desired value gives only an information on the quality attended, but there is not a superior limit. Thus, quality better then desired one can be offered to the user. An inequality is thus imminent.

User Image Profile PROFIMG	Comparison Criteria	Image Variant
[Quality Level of Color_worst … Quality Level of Color_desired]	$\supset$	Color
Resolution1_desired * Resolution2_desired	$\geq$	Resolution 1 * Resolution 2
Resolution1_worst * Resolution2_worst	$\leq$	Resolution 1 * Resolution 2
{Format_desired ∪ Format_worst [9]}	$\supseteq$	Format
[VarSize_worst … VarSize_desired [9]]	$\supset$	VarSize
Non-pertinent		Filename
Non-pertinent		Site_id
Id_prof		Non-pertinent

Table 8: Image Profile.

[9] There is one of the "additional" parameter – including the "desired" and the "worst" values – that could be found by a detailed profile description (i.e. the "MORE" function-button).

(3) Audio (Table 9)

Using the same justification as for the image colour parameter, the audio quality one will give place at two inequalities.

As far as the other sound parameters are concerned, briefly definitions are necessary. Taking a number of samples at discrete intervals digitises sound. The quality of the sound depends on two factors: the sampling rate and the sample size.

The sample rate is the number of samples per second, expressed in kHz. Common sample rates are 44 kHz (CD quality), 22 kHz and 11 kHz [urlKEYBOARDMAG]. The higher the sample rate, the better the sound quality. The number of binary digits (bits) used to quantify each sample is called the sample size (here, Bits_per_sample parameter). Larger sample sizes give better sound quality.

User Audio Profile PROFAUD	Comparison Criteria	Audio Variant
[Quality Level of Quality _worst … Quality Level of Quality_desired]	⊃	Quality
{Language_desired ∪ Language_worst}	⊇	Language
[Sample_rate_worst …Sample_rate_desired[9]]	⊃	Sample_rate
{Format_desired ∪ Format_worst [9]}	⊇	Format
[VarSize_worst … VarSize_desired[9]]	⊃	VarSize
[Bits_per_sample_worst … Bits_per_sample_desired[9]]	⊃	Bits_per_sample
Non-pertinent		Site_id
Non-pertinent		Aud_var_id
Non-pertinent		Mono_id
Non-pertinent		Filename
Id_prof		Non-pertinent

Table 9: Audio Profile.

We present hereafter a screen (**Figure 23**) of our prototype that offers the user the possibility of installing preferences priorities between different parameters of the audio profile.

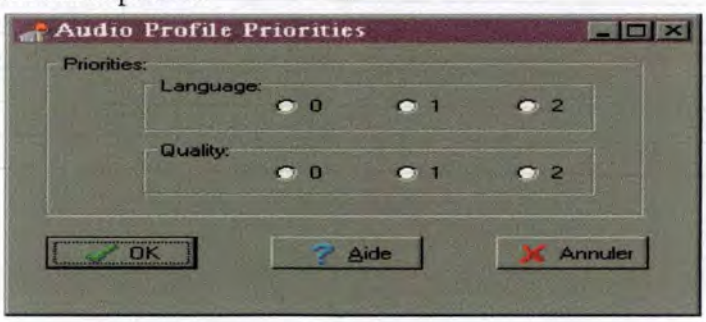

Figure 23: Audio Profile Priorities HCI.

(4) Video

The video profile is similar to the audio one, except some different parameters. We present hereafter all the parameters (Table 10) and the user interface (Figure 24).

User Video Profile PROFVID	Comparison Criteria	Video Variant VARIANT_VID
[Quality Level of Color_worst … Quality Level of Color_desired]	⊃	Color
Resolution1_desired * Resolution2_desired	≥	Resolution1 * Resolution2
Resolution1_worst * Resolution2_worst	≤	Resolution1 * Resolution2
[Framerate_worst … Framerate_desired]		Framerate
{Format_desired ∪ Format_worst [9]}	⊇	Format
[VarSize_worst … VarSize_desired [9]]	⊃	VarSize
[Bitrate_worst… Bitrate_desired[9]]	⊃	BitRate
Non-pertinent		Filename
Non-pertinent		Site_id
Non-pertinent		Mono_id
Non-pertinent		Vid_id
Id_prof		Non-pertinent

Table 10: Video Profile.

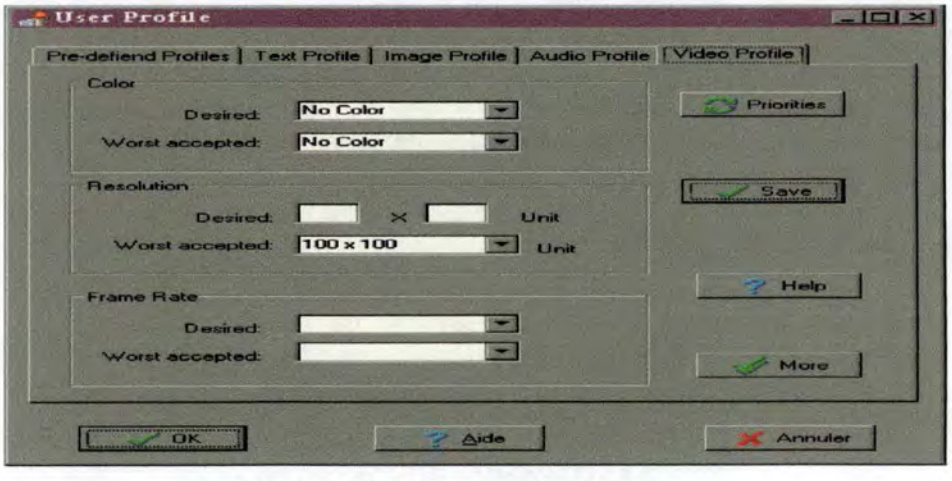

Figure 24: Video Profile HCI.

7. Views creation

We discussed in former chapters the interest of using views in our prototype. This section presents the concrete representation of those views.

View creation for client machine QoS constraint, correspond to the mapping between Client machine QoS constraints and Variant characteristics. The audio monomedia documents satisfying the QoS client machine constraints should have at least one variant having a better sound quality; or the image monomedia documents should have at least one variant having the color and resolution acceptable by the screen characteristics; or the video monomedia documents should have at least one variant having color, resolution and format acceptable by the client machine.

We present hereafter the SQL request, creator of this view.

(1) SQL Request

The following SQL request corresponds to the view creation:

```
CREATE VIEW MACHINE_DOCUMENT AS
SELECT *
FROM   DOCUMENT     D,
       MONOMEDIA    M
WHERE  DOC_TYPE= "M"
AND(   D.DOC_ID=M.DOC_ID
AND(   (    M.TYPE="A"
       AND EXISTS (   SELECT *
                      FROM        VARIANT_AUD   VA,
                                  SI_CLIENTMACHINE SI
                      WHERE M.MONO_ID=VA.MONO_ID
                      AND   VA.QUALITY<=SI.SOUND_QUALITY
              AND   VA.FORMAT<=SI.FORMAT_AUD))
OR     (    M.TYPE="V"
       AND EXISTS (   SELECT *
                      FROM        VARIANT_VID        VV,
                                  SI_CLIENTMACHINE SI
                      WHERE  M.MONO_ID=VV.MONO_ID
                      AND    VV.COLOR<=SI.COLOR
                      AND VV.RESOLUTION1<=SI.RESOLUTION1
```

```
                AND   VV.RESOLUTION2<=SI.RESOLUTION2
          AND   VV.FORMAT<=SI.FORMAT_VID))
OR    (   M.TYPE="I"
     AND EXISTS (  SELECT *
                FROM       VARIANT_IMG   VI,
                           SI_CLIENTMACHINE SI
                WHERE  M.MONO_ID=VI.MONO_ID
                AND        VI.COLOR<=SI.COLOR
                AND VI.RESOLUTION1<=SI.RESOLUTION1
                AND VI.RESOLUTION2<=SI.RESOLUTION2
          AND  VI.FORMAT<=SI. FORMAT_IMG))
OR    (   M.TYPE="T"
     AND EXISTS (  SELECT *
                FROM   VARIANT_TEXT   VT,
                           SI_CLIENTMACHINE SI
                WHERE  M.MONO_ID=VT.MONO_ID
          AND   VT.FORMAT<=SI.FORMAT_TXT))
))
```

C. Implementation

The implementation details, which concern the code and the user interface aspects, are not presented in this report.

D. Tests and Comments

Tests were developed in several steps. The first step was the database creation. From that point of view items were inserted into the database. Annex 2 shows an example of the populated database.

Queries were tested individually and gradually introduced into the program.

In a first period, only the audio monomedia and profiles were tested. Gradually, text, video and image profiles, as well as monomedia are introduced for testing. We conclude emphasising that tests were done gradually, using an increasingly integrity checking.

Chapter V: Conclusion and Future Work

"The whole is more than the sum of the parts." (Aristotle, Metaphysica)

In this Chapter V, conclusions of this thesis are presented. This chapter begins with an overview of the argument of this thesis. Following it, the research concludes by providing some indicators for future work in the area of database queries and QoS management.

Chapter I reported on the evolving notion of QoS in a DMS environment as well as CITR project goals. That section argued that for applications relying on the transfer of multimedia information, in particular continuous media, it is important that QoS is configurable, predictable and maintainable on an end-to-end basis – that is system-wide, including the distributed system platform, operating system, transport system, the underlying network and – last but not least – the user need and request. In recognition of this, the thesis has argued for the need for an integrated end-to-end QoS architecture. Our idea was to enrich the user interface by a user-determined mechanism by which we can explicitly select or modify QoS. We all know till which point "specs" are important. We consider then, that user should be able to propose or produce an "as good as possible" request, i.e. a request containing all QoS parameters needed, used or visualised by the user.

Chapter II indicated the importance of QoS control, maintenance and management in distributed multimedia systems and showed how these functions are presented in the "Broadband Services" project and the News-on-demand prototype. At that level author's contribution is focused on the QoS manager's tasks division. The aim is to introduce the delegation of some of these tasks to the database system.

Chapter III described the author's contribution to the QoS end-to-end systems, QoS management and interactions with a DBMS. Our research leads to three theoretical solutions, presented and discussed in that chapter. The possible role of database systems in QoS negotiation and adaptation is examined. Different approaches of using database internal mechanisms are presented. This research ends with the development of a prototype whose implementation details are presented in Chapter IV. This latter chapter concerns especially database creation and internal techniques used to accomplish some of the QoS manager activities.

The major aim of this project is thus concluded. The important contributions and realizations of the author's work are:

- **An adaptation of the initial prototype**

We develop our prototype and our research in the light of previous realizations of the main Broadband Services Project. We thus have to reuse

different data structure or modules, but also to point out the different obsolete aspects.

- **A research contribution concerning the relational DBMS solutions.**

Author's research concludes to three theoretical solutions. According to those solutions, different QoS manager tasks can be performed by the DBMS. After studying different database internal mechanisms, we present the Client QoS-based Queries Solution, the Client & User QoS-based Queries Solution and the QoS Profiles Solution. A first theoretical evaluation of these solutions is performed in view of realization of our first prototype.

- **Performing a prototype.**

In the light of demonstrating the feasibility of the chosen solution, a prototype was performed.

This represents a first step concluded in this domain and regarding the goals of our project.

There remain many specific areas of research in the field addressed by our prototype that should be developed in the future work. The major areas requiring investigation are presented hereafter.

In our prototype we proved the feasibility of transferring activities from the QoS Manager to the DBMS. In Chapter III we propose three solutions dealing with this kind of interactions. Presently, there is only one implemented. The other two solutions should be implemented too, so that a comparison in terms of performance and efficacy can be developed. The aim of these two prototypes would be to prove the advantage of using internal database techniques in manipulating QoS parameters.

Once the most efficient prototype decided, integration must be proceeded This best solution should be integrated in the News-on-demand prototype. In that light, the prototype must change of work platform, from a stand-alone environment to on ATM one. The aim is to validate the whole prototype, which means providing end-to-end QoS.

Validating the prototype can push the research further to extensions of the actual solutions that can be implemented and tested.

The context of working in the CITR projects, emphasised the importance of coordination, planning, respecting dead-lines and integration features. Organization issues appeared while research and prototype development was performed between several distinct institutions.

Associating industrial requirements and educational methods offers to a still-student a first constructive view of what a further carrier might represents.

6. References

[BOCH96] Gregor v. Bochmann, Brigitte Kerhervé, Abdelhakim hafid, Petre Dini, Anne Pons, *" Architectural Design of Adaptive Distributed Multimedia Systems"*, IEEE Workshop on Multimedia Software Development, Berlin, Germany, March 1996.

[BOCH] Gregor v. Bochmann, *"Architectural Design for Adaptive Distributed Multimedia Systems", article.*

[BOUR] Patrice Boursier, Pierre-Antoine Taufour, *"La technologie multimédia"*, Hermes.

[BROAD97] *"Draft Research Proposal"*, Broadband Services, 1997-98.

[CAMP96] Andrew Campbell, *"A QoS Architecture"*, PhD Lancaster University, 1996.

[CITR96] The CITR Broadband Services Major Project, *The March 1996 Demo presented by R.J. Velthuys,* May 1996.

[CITR97] CITR, *Annual Report* 1996-1997.

[CITR97-98] DRAFT CITR Research Proposal for 1997-98, *Broadband Services*, 25 February 1997.

[ELMA94] Elmasri, Navathe, *"Fundamentals of Database Systems"*, Benjamin Cummings, 1994.

[FISC97] Stephane Fischer and al, *"Cooperative QoS Management for Multimedia Applications"*, IEEE, 1997.

[FURT94] Borko Furth, *"Multimedia Systems: An Overview"*, IEEE, 1994.

[GARD96] Olivier Gardarin, *"Using Priority-based Specification for Efficient Quality of Service Negotiation"*, Université de Québec à Montréal et Université de Montréal, 1996.

[GOLD] German Goldszmidt, *"On Distributed System Management"*, article.

[GUOJ96] Guojun Lu, *"Communication and Computing for Distributed Multimedia Systems"*, Artech house, 1996.

[HAFI96] Abdelhakim Hafid, *"Gestion de la Qualité de Service dans les Applications Multimédia"*, Université de Montréal, 1996.

[HAFI94] A. Hafid, A. Bibal, G.v. Bochman, T.Burdin, R.Dssouli, J. Gecsi, B. Kerhervé, Q.Vu, *"On News-on-demand Service Implementation"*, Université de Montréal, 1994.

[HAFI96a] Abdelhakim Hafid and al, *"A QoS Negotiation Procedure for Distributed Multimedia Presentational Applications"*, IEEE, 1996.

[HAFI96b] Abdelhakim Hafid, Gregor v. Bochmann, Brigitte Kerhervé, *"A quality of Service Negotiation Procedure for Distributed Multimedia Presentational Applications"*, IEEE International Symposium on High Performance Computing, Syracuse, USA, August 1996.

[ISNA95] Benjamin Isnard, *"Gestion des profiles de qualité de service dans une application multimédia distribuée"*, Université de Montréal, 1995.

[KERH] Brigitte Kerhervé and al, *"Database Queries and QoS Management", 1996.*

[KERH96] Brigitte Kerhervé and al, *"Meta-data Modelling for Quality of Service Management in Distributed Multimedia Systems"*, IEEE 1996.

[KLAS94] W. Klas & A. Sheth, *"Meta-data for Digital Media: Introduction of the Special Issue"*, ACM Sigmond Record, 23, No. 4, December 94.

[MADJ97] Erika Madja, *"Le Web et la Qualité de Service"*, Université de Montréal, 1997.

[MEGH91] C. Meghini and al., *"Conceptual Modelling of Multimedia Documents"*, IEEE, 1991.

[MIS88] *"Multimedia and Related Technologies - A Glossary of Terms"*, Monitor Information Services, 1988.

[NAHR95] Klara Nahrstedt, *"An Architecture for End-to-end Quality of Service Provision and its Experimental Validation"*, University of Pensylvenia, 1995.

[RANG93] T.V. Rangan, *"Efficient Storage Techniques for Digital Continuous Multimedia"*, IEEE 1993.

[TAN96] A.S. Tanenbaum, *"Computer Networks"*, Third Edition, Prentice Hall International, 1996.

[VELT96] R.J.Velthuys, *"The CITR Broadband Services Major Project"*, the March 1996 Demo.

[VOGE95] Andreas Vogel and al, *"Distributed Multimedia and QoS : A Survey", PhD thesis,* 1995.

[WANG97] J.W. Wang and al., *"Enabling Technology for Distributed Multimedia Applications"*, Broadband Services Project, 1997.

[WONG] J.W. Wong and al., *"An MBone-based Distance Education System", NCE program, 1996.*

Internet references:

[urlFUNDP][10] La *Téléphonie sur Internet*

[http://www.info.fundp.ac.be/~amincule/telecom/avenir.htm]

[urlHYPERSTAND] *"Hypermedia Communications"*

[http://www.hyperstand.com]

[urlKEYBOARDMAG] *20 THINGS YOU MUST KNOW TO MAKE ELECTRONIC MUSIC*

[http://www.keyboardmag.com/reference/20things.html]

[urlObjectGEODE] ObjectGEODE – MSC Notation

[http://www.verilogusa.com/]

[urlPRINCETON] *Sound and Music*

[http://www.princeton.edu/Workshops/daveh/sound.html]

[urlUMN] *Relevant Work*

[http://www.cs.umn.edu/~wijesek/publications/qos/full/node6.html]

[urlUNIK] *Application Requirements and QoS Negotiation in Multimedia Systems*

[http://janus.unik.no/~plageman/fm2html/PROMS-preprint.html]

[10] All Internet references are indexed as: references concerning the Web site "SITE" is mentioned as: [urlSITE].

7. Annex 1: Database Creation Script

```
CREATE DATABASE "c:\Mes
Documents\Anca\Memoire\Demo\Zip\Simul0612.gdb" PAGE_SIZE 1024
;


/* Table: DOCUMENT, Owner: SYSDBA */
CREATE TABLE DOCUMENT (DOC_ID INTEGER NOT NULL,
        DOC_NAME CHAR(50),
        DOC_DATE DATE,
        DOC_AUTHOR CHAR(50),
        DOC_TYPE CHAR(2) NOT NULL,
        DOC_ABSTRACT CHAR(100),
PRIMARY KEY (DOC_ID));

/* Table: FORMAT, Owner: SYSDBA */
CREATE TABLE FORMAT (ID_FORMAT INTEGER NOT NULL,
        ID_CLIENT INTEGER NOT NULL,
        FORMAT_SUPPORTED CHAR(11) NOT NULL,
PRIMARY KEY (ID_FORMAT));

/* Table: KEYWORDS, Owner: SYSDBA */
CREATE TABLE KEYWORDS (DOC_ID INTEGER NOT NULL,
        KWD CHAR(50),
PRIMARY KEY (DOC_ID));

/* Table: LOGINS, Owner: SYSDBA */
CREATE TABLE LOGINS (LOGIN_ID CHAR(25) NOT NULL,
        PASSWD_ID CHAR(8) NOT NULL);

/* Table: MONOMEDIA, Owner: SYSDBA */
CREATE TABLE MONOMEDIA (MONO_ID INTEGER NOT NULL,
        DOC_ID INTEGER,
        TYPE_MONO CHAR(1) NOT NULL,
PRIMARY KEY (MONO_ID));

/* Table: M_D, Owner: SYSDBA */
CREATE TABLE M_D (DOC_ID INTEGER NOT NULL,
        DOC_NAME CHAR(50),
        DOC_DATE DATE,
        DOC_AUTHOR CHAR(50),
        DOC_TYPE CHAR(2) NOT NULL,
        DOC_ABSTRACT CHAR(100),
PRIMARY KEY (DOC_ID));

/* Table: M_DOCUM, Owner: SYSDBA */
CREATE TABLE M_DOCUM (NUM CHAR(25) NOT NULL);

/* Table: PROFAUDIO, Owner: SYSDBA */
CREATE TABLE PROFAUDIO (ID_PROF INTEGER NOT NULL,
        QUALITY_DESIRED CHAR(6) NOT NULL,
        QUAL_D SMALLINT NOT NULL,
        QUALITY_WORST CHAR(6) NOT NULL,
```

```
        QUAL_W SMALLINT NOT NULL,
        LANGUAGE_DESIRED CHAR(11) NOT NULL,
        PRIORITY CHAR(50) NOT NULL,
PRIMARY KEY (ID_PROF));

/* Table: PROFIMAGE, Owner: SYSDBA */
CREATE TABLE PROFIMAGE (ID_PROF INTEGER NOT NULL,
        COLOR_DESIRED CHAR(11) NOT NULL,
        COLOR_WORST CHAR(11) NOT NULL,
        COLOR_PRIORITY SMALLINT NOT NULL,
        RESOLUTION1_DESIRED SMALLINT NOT NULL,
        RESOLUTION2_DESIRED SMALLINT NOT NULL,
        RESOLUTION1_WORST SMALLINT NOT NULL,
        RESOLUTION2_WORST SMALLINT NOT NULL,
        RESOLUTION_PRIORITY SMALLINT NOT NULL,
PRIMARY KEY (ID_PROF));

/* Table: PROFTEXT, Owner: SYSDBA */
CREATE TABLE PROFTEXT (ID_PROF INTEGER NOT NULL,
        LANGUAGE_DESIRED CHAR(11) NOT NULL,
        LANGUAGE_WORST CHAR(11) NOT NULL,
        LANGUAGE_PRIORITY SMALLINT NOT NULL,
        FORMAT_TXT CHAR(11) NOT NULL,
        FORMAT_PRIORITY SMALLINT NOT NULL,
PRIMARY KEY (ID_PROF));

/* Table: PROFUSER, Owner: SYSDBA */
CREATE TABLE PROFUSER (ID_USER INTEGER NOT NULL,
        ID_PROF INTEGER NOT NULL,
        PROF_TYPE CHAR(1) NOT NULL,
        DATE_PROF DATE,
PRIMARY KEY (ID_PROF, ID_USER));

/* Table: PROFVIDEO, Owner: SYSDBA */
CREATE TABLE PROFVIDEO (ID_PROF INTEGER NOT NULL,
        COLOR_DESIRED CHAR(11) NOT NULL,
        COLOR_WORST CHAR(11) NOT NULL,
        COLOR_PRIORITY SMALLINT NOT NULL,
        RESOLUTION1_DESIRED SMALLINT NOT NULL,
        RESOLUTION2_DESIRED SMALLINT NOT NULL,
        RESOLUTION1_WORST SMALLINT NOT NULL,
        RESOLUTION2_WORST SMALLINT NOT NULL,
        RESOLUTION_PRIORITY SMALLINT NOT NULL,
        FRAMERATE_DESIRED SMALLINT,
        FRAMERATE_WORST SMALLINT,
        FRAMERATE_PRIORITY SMALLINT NOT NULL,
PRIMARY KEY (ID_PROF));

/* Table: SI_CLIENTMACHINE, Owner: SYSDBA */
CREATE TABLE SI_CLIENTMACHINE (ID_CLIENT INTEGER NOT NULL,
        MACHINE_NAME CHAR(30) NOT NULL,
        SCREEN_SIZE SMALLINT,
        RESOLUTION1 SMALLINT NOT NULL,
        RESOLUTION2 SMALLINT NOT NULL,
        COLOR CHAR(11) NOT NULL,
        SOUND_QUALITY CHAR(6) NOT NULL,
        S_QUAL SMALLINT NOT NULL,
        RELIABILITY SMALLINT,
        THROUGHPUT SMALLINT,
```

```
PRIMARY KEY (ID_CLIENT));

/* Table: SI_CMEDIAFILE, Owner: SYSDBA */
CREATE TABLE SI_CMEDIAFILE (ID_SERVER INTEGER NOT NULL,
        RELIABILITY SMALLINT,
        THROUGHPUT SMALLINT,
PRIMARY KEY (ID_SERVER));

/* Table: SI_NETWORK, Owner: SYSDBA */
CREATE TABLE SI_NETWORK (SITE1 INTEGER NOT NULL,
        SITE2 INTEGER NOT NULL,
        RELIABILITY SMALLINT,
        THROUGHPUT SMALLINT,
PRIMARY KEY (SITE1, SITE2));

/* Table: VARIANT_AUD, Owner: SYSDBA */
CREATE TABLE VARIANT_AUD (AUD_VAR_ID INTEGER NOT NULL,
        MONO_ID INTEGER,
        LANGUAGE CHAR(11) NOT NULL,
        SAMPLE_RATE SMALLINT,
        BITS_PER_SAMPLE SMALLINT,
        SOUND_QUALITY CHAR(11) NOT NULL,
        S_QUAL SMALLINT NOT NULL,
        FORMAT CHAR(11) NOT NULL,
        VARSIZE SMALLINT,
        FILENAME CHAR(50),
        SITE_ID INTEGER NOT NULL,
PRIMARY KEY (AUD_VAR_ID));

/* Table: VARIANT_IMG, Owner: SYSDBA */
CREATE TABLE VARIANT_IMG (IMG_VAR_ID INTEGER NOT NULL,
        MONO_ID INTEGER NOT NULL,
        COLOR CHAR(11) NOT NULL,
        RESOLUTION1 SMALLINT NOT NULL,
        RESOLUTION2 SMALLINT NOT NULL,
        FORMAT CHAR(11) NOT NULL,
        VARSIZE SMALLINT,
        FILENAME CHAR(50),
        SITE_ID INTEGER NOT NULL,
PRIMARY KEY (IMG_VAR_ID));

/* Table: VARIANT_TEXT, Owner: SYSDBA */
CREATE TABLE VARIANT_TEXT (TEX_VAR_ID INTEGER NOT NULL,
        MONO_ID INTEGER NOT NULL,
        LANGUAGE CHAR(11) NOT NULL,
        FORMAT CHAR(11) NOT NULL,
        VARSIZE SMALLINT,
        FILENAME CHAR(50),
        SITE_ID INTEGER NOT NULL,
PRIMARY KEY (TEX_VAR_ID));

/* Table: VARIANT_VID, Owner: SYSDBA */
CREATE TABLE VARIANT_VID (VID_VAR_ID INTEGER NOT NULL,
        MONO_ID INTEGER NOT NULL,
        COLOR CHAR(11) NOT NULL,
        RESOLUTION1 SMALLINT NOT NULL,
        RESOLUTION2 SMALLINT NOT NULL,
        FRAMERATE SMALLINT,
        BITRATE SMALLINT,
```

```
        FORMAT CHAR(11) NOT NULL,
        VARSIZE SMALLINT,
        FILENAME CHAR(50),
        SITE_ID INTEGER NOT NULL,
PRIMARY KEY (VID_VAR_ID));


ALTER TABLE KEYWORDS ADD FOREIGN KEY (DOC_ID) REFERENCES
DOCUMENT(DOC_ID);
ALTER TABLE MONOMEDIA ADD FOREIGN KEY (DOC_ID) REFERENCES
DOCUMENT(DOC_ID);
ALTER TABLE FORMAT ADD FOREIGN KEY (ID_CLIENT) REFERENCES
SI_CLIENTMACHINE(ID_CLIENT);
ALTER TABLE SI_NETWORK ADD FOREIGN KEY (SITE1) REFERENCES
SI_CMEDIAFILE(ID_SERVER);
ALTER TABLE SI_NETWORK ADD FOREIGN KEY (SITE2) REFERENCES
SI_CMEDIAFILE(ID_SERVER);
ALTER TABLE VARIANT_AUD ADD FOREIGN KEY (MONO_ID) REFERENCES
MONOMEDIA(MONO_ID);
ALTER TABLE VARIANT_AUD ADD FOREIGN KEY (SITE_ID) REFERENCES
SI_CMEDIAFILE(ID_SERVER);
ALTER TABLE VARIANT_IMG ADD FOREIGN KEY (MONO_ID) REFERENCES
MONOMEDIA(MONO_ID);
ALTER TABLE VARIANT_IMG ADD FOREIGN KEY (SITE_ID) REFERENCES
SI_CMEDIAFILE(ID_SERVER);
ALTER TABLE VARIANT_TEXT ADD FOREIGN KEY (MONO_ID) REFERENCES
MONOMEDIA(MONO_ID);
ALTER TABLE VARIANT_TEXT ADD FOREIGN KEY (SITE_ID) REFERENCES
SI_CMEDIAFILE(ID_SERVER);
ALTER TABLE VARIANT_VID ADD FOREIGN KEY (MONO_ID) REFERENCES
MONOMEDIA(MONO_ID);
ALTER TABLE VARIANT_VID ADD FOREIGN KEY (SITE_ID) REFERENCES
SI_CMEDIAFILE(ID_SERVER);


ALTER TABLE DOCUMENT ADD
        check (doc_type in ("M", "Mm"))
;
ALTER TABLE MONOMEDIA ADD
        check (type_mono in ("t", "i", "a", "v"))
;
ALTER TABLE PROFAUDIO ADD
        check (quality_desired in ("NoAudioQ", "CD", "Phone"))
;
ALTER TABLE PROFAUDIO ADD
        check (quality_worst in ("NoAudioQ", "CD", "Phone"))
;
ALTER TABLE PROFAUDIO ADD
        check (language_desired in ("No_Language", "English",
"French", "German", "Spanish", "Dutch", "Arabic"))
;
ALTER TABLE PROFAUDIO ADD
        check (priority in ("Language, Sound_Quality",
"Sound_Quality, Language"))
;
ALTER TABLE PROFIMAGE ADD
        check (color_desired in ("Super_color", "Color",
"Black
_white", "Grey_scale", "No_color"))
```

```
;

ALTER TABLE PROFIMAGE ADD
        check (color_desired in ("Super_color", "Color",
"Black_white", "Grey_scale", "No_color"))
;
ALTER TABLE PROFTEXT ADD
        check (language_desired in ("No_Language", "English",
"French", "German", "Spanish", "Dutch", "Arabic"))
;
ALTER TABLE PROFTEXT ADD
        check (language_worst in ("No_Language", "English",
"French", "German", "Spanish", "Dutch", "Arabic"))
;
ALTER TABLE PROFTEXT ADD
        check (format_txt in ("No_Format", "word95", "word97",
"wordpf", "wps", "asc", "ans", "mcw", "dot", "txt", "rtf",
"ms_dos", "htm", "html"))
;
ALTER TABLE PROFUSER ADD
        check (prof_type in ("t", "i", "a", "v"))
;
ALTER TABLE PROFVIDEO ADD
        check (color_desired in ("Super_color", "Color",
"Black_white", "Grey_scale", "No_color"))
;
ALTER TABLE PROFVIDEO ADD
        check (color_worst in ("Super_color", "Color",
"Black_white", "Grey_scale", "No_color"))
;
ALTER TABLE FORMAT ADD
        check (format_supported in ("NoFormat", "mp3", "wav",
"No_Format", "gif", "jpg", "pcd", "pcx", "png", "raw", "tif",
"wmf", "wpg", "word95", "word97", "wordpf", "wps", "asc",
"ans", "mcw", "dot", "txt", "rtf", "ms_dos", "htm", "html",
"mpg", "avi", "vdo", "mov", "ra"))
;
ALTER TABLE SI_CLIENTMACHINE ADD
        check (color in ("Super_color", "Color",
"Black_white", "Grey_scale", "No_color"))
;
ALTER TABLE SI_CLIENTMACHINE ADD
        check (sound_quality in ("NoAudioQ", "CD", "Phone"))
;
ALTER TABLE VARIANT_AUD ADD
        check (language in ("No_Language", "English",
"French", "German", "Spanish", "Dutch", "Arabic"))
;
ALTER TABLE VARIANT_AUD ADD
        check (sound_quality in ("NoAudioQ", "CD", "Phone"))
;
ALTER TABLE VARIANT_AUD ADD
        check (format in ("NoFormat", "mp3", "wav"))
;
ALTER TABLE VARIANT_IMG ADD
        check (color in ("Super_color", "Color",
"Black_white", "Grey_scale", "No_color"))
;
ALTER TABLE VARIANT_IMG ADD
```

```
        check (format in ("No_Format", "gif", "jpg", "pcd",
"pcx", "png", "raw", "tif", "wmf", "wpg"))
;
ALTER TABLE VARIANT_TEXT ADD
        check (language in ("No_Language", "English",
"French", "German", "Spanish", "Dutch", "Arabic"))
;
ALTER TABLE VARIANT_TEXT ADD
        check (format in ("No_Format", "word95", "word97",
"wordpf", "wps", "asc", "ans", "mcw", "dot", "txt", "rtf",
"ms_dos", "htm", "html"))
;
ALTER TABLE VARIANT_VID ADD
        check (color in ("Super_color", "Color",
"Black_white", "Grey_scale", "No_color"))
;
ALTER TABLE VARIANT_VID ADD
        check (format in ("NoFormat", "mpg", "avi", "vdo",
"mov", "ra"))
;
```

8. Annex 2: Acronyms List

ATM = Asynchronous Transfer Mode

CAS = Center for Advanced Studies

CITR = Canadian Institute for Telecommunication Research

CMTP = Continuous Media Transport Protocol

CMFS = Continuous Media File Server

DMS = Distributed Multimedia Systems

DMSA = Distributed Multimedia Systems Application

HCI = Human-Computer Interfaces

HeiTP = Heidelberg Protocol

HeiTS = Heidelberg System

MSC = Message Sequence Chart

OSI = Open Systems Interconnection

QoS = Quality of Service

RCAP = Real-time Channel Administration Protocol

RMTP = Real-time Message Transport Protocol

RSVP = Resource Reservation Protocol

RTCMP = Real-time Control Message Protocol

SDU = Service Data Units

ST-II = Stream Transport Protocol Version Two