[Won 97] J.Wong, K.Lyons, R.Velthuys, G.Bochmann, E.Dubois, N.Georganas, G.Neufeld, T.Ozsu, J.Brinskelle, D.Evans, A.Hafid, N.Hutchinson, P.Inglinski, B.Kerherve, L.Lamont, D.Makaroff, and D.Szafron, *Enabling Technology for Distributed Multimedia Applications*, IBM Systems Journal, 1997 (to appear)

[Zha 95] L.Zhang, et al., *RSVP Functional Specification*, Working Draft, draft-ietf-rsvp-spec-07.ps, 1995

[Haf 96a] A.Hafid and G.v.Bochmann, *A QoS Negotiation Procedure for Distributed MM Presentational Applications*, In Proceedings of the Fifth IEEE International High Speed Distributed Computing (HPDC-5), Syracuse, New York, 1996

[Haf 96b] A.Hafid and G.v.Bochmann, *Quality of Service Negotiation in News-on-Demand Systems: An Implementation*, In Proceedings of the Third International Workshop on Protocols for Multimedia Systems, Madrid, Spain, 1996

[Haf 97a] A.Hafid and G.v.Bochmann, *Quality of Service Adaptation in Distributed Multimedia Applications*, ACM Multimedia Systems Journal, 1996 (to appear)

[Haf 97b] A.Hafid, G.v.Bochmann and R.Dssouli, *A Negotiation Approach with Future Reservation (NAFUR): A Detailed Study*, Computer Networks and ISDN Journal, 1996 (to appear)

[Haf 97c] A.Hafid and S.Fischer, A Multi-Agent Architecture for Cooperative Quality of Service Management, In Proceedings of IFIP/IEEE Conference on Management of Multimedia Networks and Services (MMNS'97), Montreal, Canada, 1997 (to appear)

[Han 91] J.Hanko, E.Kuerner, D.Northcut and G.Wall, *Workstation Support for Time-Critical Applications*, Second International Workshop Heidelberg, Germany, November 1991

[Kes 95] S.Keshav and H.Saran, *Semantics and Implementation of a Native-Mode ATM protocol Stack*, Internal Technical Memo, AT&T Bell Lab, Murray Hill, NJ, January, 1995

[Nah 93] K.Nahrstedt and J.Smith, *Application-Driven Approach to Networked Multimedia Systems*, The 18th Conference on Local Computer Networks, 1993

[Nah 95] K.Nahrstedt, *An Architecture for End-to-End Quality of Service Provision and its Experimental Validation*, Ph.D. Thesis, University of Pennsylvania, 1995

[Ram 91] J.Rambaugh & al., *Object Oriented Modeling and Design*, Prentice Hall, 1991

[Fis 97] S.Fisher, A.Hafid, G.v.Bochmann and H.de Meer, *An Approach to Cooperative QoS Management for Multimedia Applications*, In Proceedings of the IEEE Multimedia Systems Conference, Ottawa, 1997

[Top 90] C.Topolcic, *Experimental Internet Stream Protocol: Version 2 (ST-II)*, Internet RFC 1190, 90

[Vel 95] R.Velthuys, CITR News-on-Demand Demo, CITR Intenal Report, March 1995

[Vol 95] C.Volg, L.Wolf, R.Herrtwich and H.Wittig, *HeiRat -Quality of Service Management for Distributed Multimedia Systems*, Multimedia Systems Journal, November 1995

multimedia databases) can be applied to applications which have both presentational and conversational characteristics, such as computer supported cooperative work and tele-teaching application [Fis 97, Haf 97c].

Acknowledgment

We would like to thank our colleagues of the CITR QoS subproject, Rachida Dssouli, Jan Gecsei, Brigitte Kerhervé, Ramesh Somalingam, Jun Zhang, Benjamin Isnard, and Quoc Vu, for their contribution to the work presented here. We would like also to thank David Evans (University of Waterloo), Jeff Brinskelle (University of Ottawa) and Rolf Velthuys (IBM CAS, Toronto) who participated in the integration of the QoS functions into the news-on-demand prototype. This work was supported by a grant from the Canadian Institute for Telecommunication Research (CITR), under the Networks of Centres of Excellence Program of the Canadian Government.

References

[And 91] D.Anderson, R.Herrtwich and C.Schaefer, *SRP: A resource Reservation Protocol for Guaranteed-Performance Communications in the Internet*, The International Computer Science Institute, Berkeley, 1991

[Boc 96] G.v.Bochmann, B.Kerherve, A.Hafid, P.Dini and A.Pons, *Functional Design of Adaptive Systems*, In Proceedings of the IEEE Workshop on Multimedia Software Development, Berlin, Germany 1996

[Cam 94] A.Campbell, G.Coulson and D.Hutchison, *A Quality of Service Architecture*, ACM Computer Communication Review, April 1994

[Fer 92] D.Ferrari, A.Banerjea and H.Zhang, *Network Support for Multimedia*, Technical report 92-072, International Computer Science Institute, Berkeley, November 1992

[Fer 95] D.Ferrari, *The Tenet Experience and the Design of Protocols for Integrated Services Internetworks*, Multimedia Systems Journal, November 1995

[Gop 94] G.Gopalakrishna and G.Parulkar, *Efficient Quality of Service in Multimedia Computer Operating Systems*, Department of Computer Science, Washington University, Report WUCS-TM-94-04, 1994

[Haf 95] A.Hafid, *A Hierarchical Negotiation for Distributed Multimedia Applications in a Multi-Domain Environment*, In Proceeding of the Second International Workshop on Protocols for Multimedia Systems, Salzburg, Austria, 1995

and the corresponding cost. The user should press OK, within a given amount of time, to start the delivery of the document. If the given time is exceeded, the session is simply aborted and a new negotiation is required if the user wants to play the document.

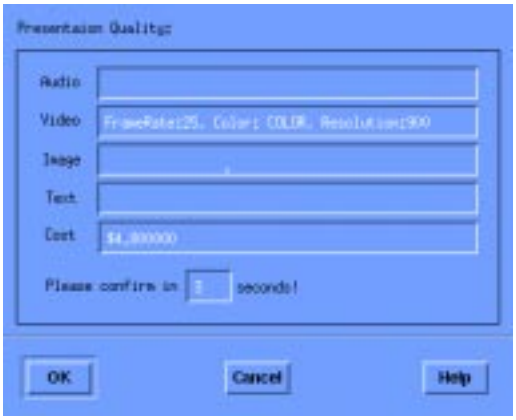

Figure 8. Information window

4. Conclusion

The paper describes work on a new QoS management framework which includes the dynamic choice of a configuration of system components to support the QoS requirements of the user. The management process, especially the QoS negotiation process, is considered as an optimization problem and not only as a resource reservation and scheduling problem: several system configurations which might support the requested service are considered and the most appropriate one is selected. This is not considered in other existing QoS architectures which assume a static system configuration, that is, the system components involved in QoS provision are know a priori. The proposed framework allows to improve the utilization of system resources, and thus to increase the system availability; it also allows to recover automatically, if this is possible, from QoS degradations. Furthermore, it provides the flexibility to incorporate different resource reservation schemes and scheduling policies, and to accommodate new system component technologies

To show the feasibility of our proposals, we implemented a QoS manager for access to distributed multimedia databases, which is an instantiation of the proposed framework. This QoS manager provides smart negotiation, in opposition to the basic negotiation provided by existing QoS architectures, which is based on the specified user preferences and resource availability. Furthermore it supports the negotiation of a multimedia object (audio and video) as an single object, that is, the optimization is performed taking into account all the different monomedia components of the document, while most existing approaches consider a single monomedia object. More generally, the negotiation procedure can be used for the initial QoS negotiation, QoS renegotiation, and QoS adaptation [Haf 97a] with almost no modifications.

Currently we study to which extent our negotiation procedure for presentational applications (access to

those profiles that cannot be satisfied by the system are high-lighted in red color.

If the user wants to edit a profile, he/she must double-click on this profile and the corresponding profile window appears (Figure 7). Each profile, namely video, audio, text, image, time, and cost profiles, has an associated customized profile window which allows the user to specify his/her requirements. Through scaling bars, and predefined values the user is able to set the values for QoS parameters, cost, time and importance. For each QoS parameter the user is able to set the desired value and the minimum value acceptable.

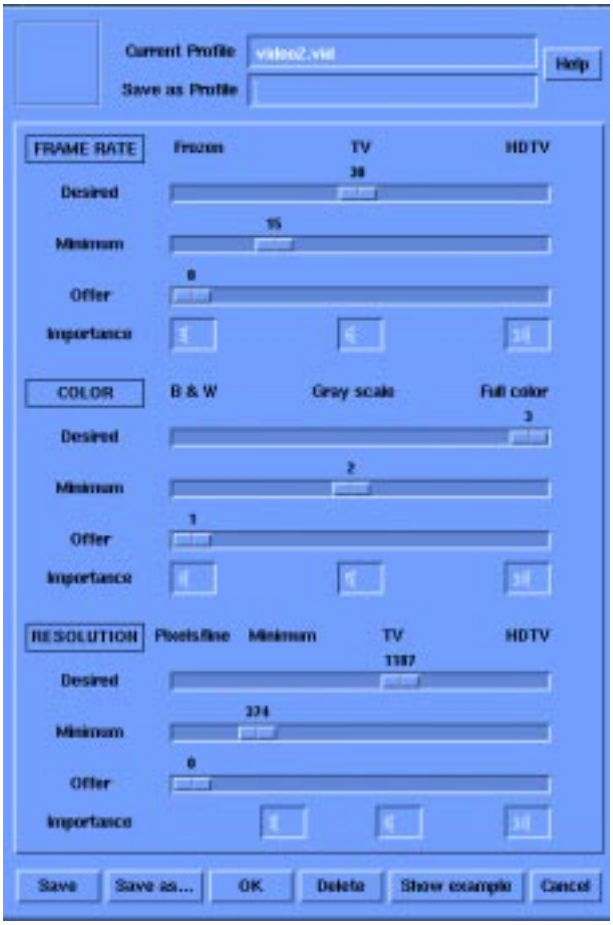

Figure 7. Video profile

If the constraint button corresponding to the profile is red (see profile component window), the offer provided by the system is also displayed for each QoS parameter (on the offer scaling bar). If the user accepts the offer, he/she should push OK; otherwise he/she should push CANCEL to reject the offer, or modify the offer and then push OK to initiate a renegotiation.

At any time the user is able to cancel his/her profile settings using the button CANCEL.

The profile manager uses the information window (Figure 8) to display the results of the negotiation process. If the negotiation failed, the profile manager displays the negotiation status, namely FAILEDTRYLATER or FAILEDWITHOUTOFFER; otherwise the profile manager displays the values of the different QoS parameters

negotiation for the selected document.

Figure 5. Main window

If the user double-clicks on a given user profile, the profile component window (Figure 6) appears displaying the list of monomedia, time, and cost profiles (see Section 3.2). The profile components of the user profile selected are highlighted. To modify or create a new user profile, the user selects the profiles of interest (monomedia, time, and cost profiles) and pushes on Save or Save, respectively.

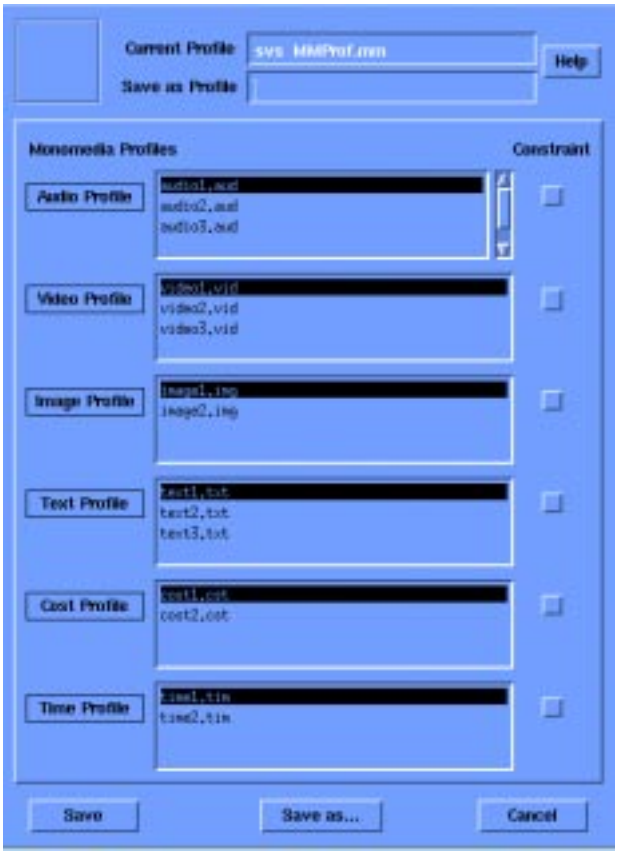

Figure 6. Profile component window

The profile component window appears also when the negotiation fails. In this case the constraint buttons of

classification parameter, and the OIV as the secondary classification parameter. Thus the OIV allows to classify these offers which have the same static negotiation status.

We note that in the case that no offer satisfies the static negotiation status values DESIRABLE or ACCEPTABLE, the best offer will have the status CONSTRAINT and the QoS negotiation will lead to lead to SUCCEED, but could lead to FAILEDWITHOFFER which implies that the user is asked whether the offer should be accepted or not.

Example

In this example we consider the offers described in the example of Section 3.4.2. To demonstrate the impact of the overall importance value (OIV) on the classification of offers, we present the results of the classification for three different settings of importance values.

(1) Let us assume that we have the following importance values: color: 9, grey: 6, black&white: 2, TV resolution: 9, 25 frames/s: 9, 15 frames/s: 5, and cost factor: 4. The QoS manager computes the OIV for each offer and produces the following results: offer1: 10, offer2: 7, and offer3: 12, and offer4: 7. Taking into account the static negotiation status (see the example of Section 3.4.2), the offers are classified as follows: offer4, offer3, offer1, and offer2.

(2) Let us assume that we have the following importance values: color: 9, grey: 6, black&white: 2, TV resolution: 9, 25 frames/s: 9, 15 frames/s: 5, and cost factor: 0. This means that the cost is not of no importance; the quality is the main concern. The OIV values are: offer1: 20, offer2: 23, and offer3: 24, and offer4: 27. Taking into account the static negotiation status, the offers are classified as follows: offer4, offer3, offer2, and offer1.

(3) Let us assume that we have the following importance values: color: 0, grey: 0, black&white: 0, TV resolution: 0, 25 frames/s: 0, 15 frames/s: 0, and cost factor: 4. This means that the quality is of no importance; the cost is the main constraint. The OIV values are: offer1: -10, offer2: -16, and offer3: -12, and offer4: -20. Taking into account the static negotiation status, the offers are classified as follows: offer1, offer3, offer2, and offer4.

3.5. QoS user graphical interface (QoS GUI)

As described in Section 3.2, the QoS GUI allows the user to negotiate and renegotiate his/her requirements in terms of quality and cost. The profile manager is responsible for supporting these facilities. In this subsection we give an overview of the user interface implemented in our prototype.

The main window (Figure 5) of the QoS GUI appears as soon as the user pushes on Play-with-QoS or Open-Window in the news-on-demand prototype user interface [Vel 95]. The main function of this window is to allow the user to select, edit or delete a user profile, or to select a default user profile. If the user wants to leave the QoS GUI, he/she must push EXIT. When the user selects the desired user profile, he/she clicks on OK to start the QoS

get the requested quality; while for another user, cost is more important than quality, that is, he/she is willing to accept a degraded quality in order to pay not more than the cost specified in the user profile. The importance values allow the user to specify his/her specific preferences (which are different from user to user) regarding quality characteristics and cost.

For each QoS parameter, the user specifies the importance values for only a specific set of values. For example, the user sets the importance values only for HDTV rate, TV rate and frozen rate when dealing with the frame rate parameter. If the user selects a frame rate (specified in the user profile) different from these specific values, the corresponding importance values is computed by linear interpolation from the neighbouring values. In the context of the news-on-demand prototype, the user specifies the importance values, for a given QoS parameter, for the QoS parameter values shown in Figure 4 (resolution: HDTV resolution, TV resolution, minimum resolution; audio quality: CD, telephone; color: super-color, color, gray, black&white; etc.); other values are calculated by interpolation.

The user profile associates a default importance value with each QoS parameter value (in Figure 4). However, at any time during the negotiation phase, the user may modify these values to meet his/her goals. Indeed, for some user the video color parameter is more important than the video resolution, while it is the opposite for other users. The profile manager provides the user with facilities to set importance values for the QoS parameters of interest.

Similarly, the user sets an importance factor for the cost. This importance factor indicates the importance of a cost of 1 $. A small importance factor indicates that the cost is not very important, while a high importance factor indicates that the cost is a very important parameter to consider.

Given an offer, the following steps are performed to compute the overall importance value of that offer:

- compute the importance of the provided quality, which is the sum of the importance values of all QoS parameters (except cost) of all selected monomedia variants.
- compute the importance of the cost which is the product of the cost of the offer and the importance factor of the cost.
- substract the importance of cost from the importance of the provided quality to obtain the effective importance value of the offer.

3.4.4. Classification of offers

In order to satisfy the user requirements, it is not sufficient to classify the offers according to the overall importance value (OIV). Indeed the QoS of the best offer (with the highest OIV) may be very different from the QoS requested by the user: The best offer is effectively the best in terms of cost/QoS, however, it may not correspond to the user wishes. To overcome such a problem we use the static negotiation status as primary

situation, particularly if it happens frequently.

(2) Many offers may be produced for a given request, and it is not appropriate to present a large number of offers to the user. We think that the user, especially a novice, may become confused. Furthermore, the presentation of several offers will complicate the QoS GUI.

(3) The user may select an offer which does not make an optimal usage of system resources.

Given these drawbacks, we decided to define a classification procedure that allows to automatically classify the offers. Thus only one offer, for which the resources are reserved, will be presented to the user. The latter may accept or reject the offer, or initiate a renegotiation. Consequently when the user accepts an offer, he/she is certain that the offer will be supported by the system.

3.4.2. The static negotiation status

The static negotiation status indicates the degree of satisfaction of the user profile by a given system offer. We have considered three values for the static negotiation status, however, additional values may be considered:

DESIRABLE: the QoS satisfies the QoS desired by the user (we note that this implies ACCEPTABLE);

ACCEPTABLE: the QoS is better than the worst acceptable QoS values accepted by the user.

CONSTRAINT: the QoS of the offer does not meet the worst acceptable QoS values requested by the user (for at least one monomedia and some of its characteristics).

Given a system offer, the computation of the static negotiation status consists of a simple comparison between the QoS associated with the offer and the user profile.

As an example, let us assume that (1) the user asks to play a news article with (color, TV resolution, 25 frames/s) as desired QoS and also as the worst acceptable QoS, and 4 $ as the maximum cost to pay, and (2) the QoS manager produces, after the first two steps, the following offers (after the mapping activity):

- offer1: (black&white, TV resolution, 25 frames/s) with a cost of 2.5$,
- offer2: (color, TV resolution, 15 frames/s) with a cost of 4$,
- offer3: (grey, TV resolution, 25 frames/s) with a cost of 3 $, and
- offer4: (color, TV resolution, 25 frames/s) with a cost of 5$.

The QoS manager computes the static negotiation status for each offer. The results are: offer1: CONSTRAINT, offer2: CONSTRAINT, offer3: CONSTRAINT, and offer4: ACCEPTABLE.

3.4.3. The importance of different qualities

As mentioned above, the importance values indicate the relative importance between quality characteristics and cost. For some user, for instance, quality is more important than cost, that is, he/she is willing to pay "any" cost to

of the document will continue using the services of the alternate components (which support the new system offer). In order to perform this transition in the current implementation, the QoS Manager stops the presentation of the document after having obtained the current position of the display, and restarts the presentation (using the alternate components) from the position parameter determined earlier. This transition procedure is a simple one; more sophisticated procedures may be used.

3.4. The classification of system offers

The classification (in terms of "goodness") of a set of system offers in respect to the quality and cost desired by the user is not at all obvious. To the best of our knowledge there is no existing literature on this topic. To classify system offers in respect to cost is obvious, since the cheapest system offer is the best (a description of schemes to compute the cost to support a system offer can be found in [Haf 96a]). To classify system offers in terms of quality and cost, we proposed the notion of weighted average [Haf 97b]: for each system offer we compute the weighted average of the QoS parameter values associated with the system offer, and the system offer with the highest average is the best. In the following we use the terms system offer and user offer interchangeably. For sake of clarity we present the following examples in terms of user offers instead of system offers; a single system offer corresponds to a single user offer.

We note that a classification in terms of only QoS, or only cost, is neither optimal nor suitable for "smart" negotiation. If we classify the offers in terms of cost (resp. QoS), it is likely that the "best" offer (which has the lowest cost (resp. best QoS) does not meet the user QoS requirements.

3.4.1. Motivating example

Let us assume that the user asks to play a video news article with a quality (color, 25 frames/S, TV Resolution) and 6 $ as a maximum cost (the desired and the worst acceptable values are the same). The QoS manager makes use of our negotiation procedure, and finds the following user offers (after mapping the corresponding system offers): (Color, 15 frames/s, TV Resolution) at 5$, (GREY, 25 frames/s, TV resolution) at 4 $, and (Color, 25 frames/s, TV resolution) at 6 $.

The QoS manager should be able to classify these three offers: Which is the best offer and which is the worst offer?

A simple solution is to present the possible offers to the user, and it is up to him/her to select the best. However, this solution has several drawbacks:

(1) Since we have to present the all offers to the user, we cannot make any resources reservation. It does not make sense to reserve resources for all the offers. Then when the user selects an offer, we are not certain that the offer will be supported by the system, e.g. in the case of resources shortage; the user will not be happy with this

(3) Computation of classification parameters:

Given the requested QoS/cost (user profile) and the available audio and video variants of the document, two parameters are computed for each feasible system offer: the static negotiation parameter and the importance, which are used in the step below and explained in more detail in Section 3.

(4) Classification of system offers:

Step 4 sorts the different system offers (including the video, audio, image, text variants) in the order defined by the classification parameters, thus producing an ordered list of system offers. We decided to consider all feasible system offers, even those which do not satisfy the requested QoS and/or cost. This decision is motivated by the following points: (1) If the requested QoS/cost cannot be supported by the system, the best feasible system offer is produced, even if it does not satisfy the user requirements. (2) During the active phase some QoS violations may occur; in this case, the adaptation procedure can make use of the whole set of feasible system offers to perform an automatic adaptation [Haf 97a].

(5) Resources commitments:

The QoS manager considers the best system offer that satisfies the QoS/cost requested by the user, and asks the transport system and the media file servers to reserve the corresponding resources. If these reservations succeed, the negotiation completes successfully and the notification SUCCEEDED is sent to the user; otherwise, the next acceptable system offer is considered. If there are not enough resources to support any of the acceptable system offers, the same procedure is applied on the feasible (not acceptable) system offers. If the resource reservation succeeds for one of these system offers, the negotiation completes with the notification FAILEDWITHOFFER, which is sent to the user via the profile manager. If the resource reservation fails for all feasible system offers, the notification FAILEDTRYLATER is sent to the user via the profile manager.

(6) User confirmation:

Once the resources are reserved for a given system offer (Step 5), a notification is sent to the user, that is, the user offer derived from the system offer is displayed. The user must confirm the offer (by rejection or acceptance) within a limited amount of time since the resources are already reserved. If the offer is accepted, the system starts playing the document; otherwise the reserved resources are de-allocated.

If the network or/and the server machine become congested during the playout of the document, thus leading to lower presentation quality, the QoS manager makes use of the adaptation procedure. In this case, the QoS manager considers the ordered set of system offers, except the current one (which is in difficulty), and executes again Step 5. If an alternate system offer is selected and the required resources are reserved, the QoS manager automatically performs a transition from the current system configuration to the new one. This means, the delivery

Definition 1: A system offer consists of a set of variants (a variant for each monomedia component of the document) and the cost the user should pay.

Definition 2: A user offer represents the service quality the system promises to provide and the cost the user should pay for the document delivery. A user offer is specified as a MM profile (see Section 3.2).

The input of the negotiation procedure consists of the document to be played and the user profile selected by the user, while the output (output) consists of the negotiation status and possibly a user offer.

The negotiation status may take one of the following five values:

(1) SUCCEEDED: This indicates that the requested QoS and the maximum cost the user is willing to pay are satisfied by the system. A user offer (which does not violate the worst acceptable values contained in the user profile) is returned.

(2) FAILEDWITHOFFER: This indicates failure. A user offer is returned which can be supported, but does not satisfy the user requirements.

(3) FAILEDTRYLATER: This indicates failure due to resources shortage. The user may try later and his/her request may eventually be accepted.

(4) FAILEDWITHOUTOFFER: This indicates failure because no possible instantiation of the functional configuration (of the news-on-demand service) to a physical configuration exists, e.g. the client machine does not support a suitable decoder to decode the requested data.

(5) FAILEDLOCALLY: This indicates failure because the client machine does not support the QoS requested by the user, e.g. the user asks for a color video, while the client machine screen is black and white.

The negotiation procedure, performed by the QoS manager, consists of six main steps.

(1) Static local checking:

During this step, the characteristics of the client's machine are checked, such as the screen size and the screen color. If the client's machine does not support the QoS requested by the user, a notification FAILEDLOCALLY is sent to the user via the profile manager.

(2) Static compatibility checking:

This step checks the format compatibility of the variants of the selected document with the decoder(s) supported by the client machine. For example, if the client machine supports only MPEG decoding and the video variant Variant1 is coded in MJPEG then Variant1 will simply not be considered as a feasible system offer. If such an activity produces no feasible system offer, a notification FAILEDWITHOUTOFFER is sent to the user, via the profile manager.

A user profile consists of (1) a MM profile which indicates the desired values, (2) a MM profile which indicates the worst acceptable values, and (3) the importance profile which indicates the values of importance factors. A MM profile consists of video, audio, text, and image profiles, cost profile and time profile (Figure 4). For instance, the frame rate is specified by any integer value between HDTV rate (60 frames/s) and frozen rate (1 frame/s), and the resolution is defined by any integer values between HDTV resolution (1920 pixels/line) and minimal resolution (10 pixels/line). The cost profile attributes should be specified in terms of total $, or $ per second.

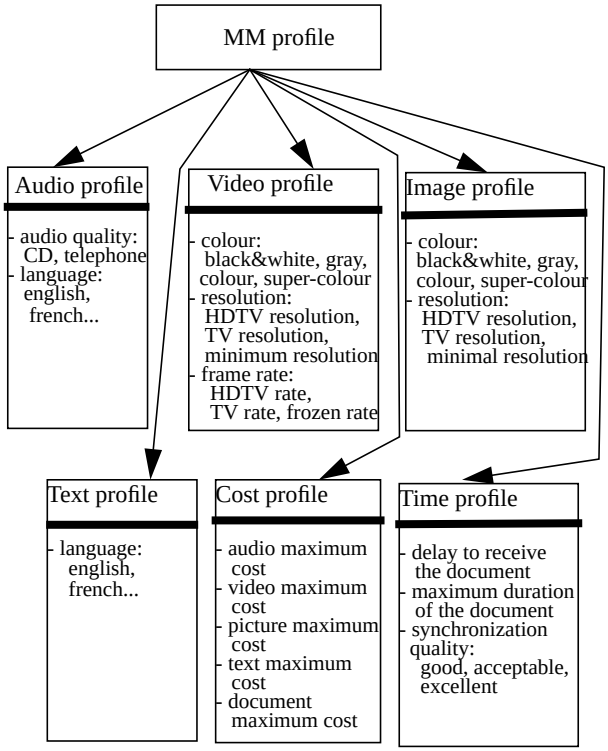

Figure 4. MM profile

The system component responsible for user profile management via the QoS GUI is called the profile manager.

3.3. The QoS Manager

The component which implements the QoS management functions, namely QoS negotiation and adaptation, is called the QoS manager. The role of the QoS manager is to determine an optimal system (physical) configuration (server(s), network(s), and the client machine) which might support the user requirements. In the context of the news-on-demand prototype, this activity consists of determining, for each monomedia of the selected document, which of the available variants is the "best". Since a given variant is stored on a specific server machine which is connected to the client machine through a specific network, the choice of the monomedia variants implies the choice of the system components which are involved in the configuration.

The QoS manager considers possible system configurations, which we will call system offers (see Definition 1), selects an optimal one, and makes an offer to the user, called user offer (see Definition 2). It is worth noting that a user offer can be derived from a system offer using appropriate mapping functions [Haf 96a].

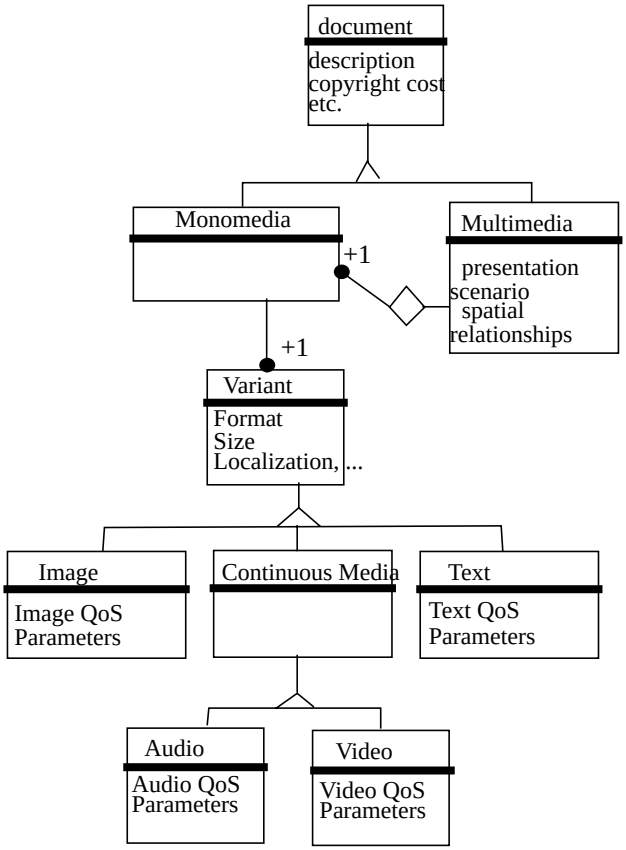

Figure 3. MM document model

3.2. QoS Specification and User Profiles

The user specifies his/her requirements via a graphical user interface (GUI), which we call here the QoS GUI (see Section 3.5). The QoS GUI should hide, as much as possible, the internal QoS parameters, e.g. throughput and jitter, and provide facilities to describe the requested QoS in terms of a set of user-perceived characteristics. In addition to service quality and cost requirements, the user should be able to specify some optimization criteria in terms of weights, which we will call importance factors. These factors indicate the degree of importance of different QoS characteristics, including the cost. Examples of the utilization of the importance factors are the following: (1) the user specifies that the QoS is more important than the cost, (2) the user specifies that the audio is more important than the video, (3) the user specifies that video frame rate is more important than video resolution, and (4) the user specifies that french is more important than english.

To facilitate the QoS specification by the user, we introduce the notion of user profiles. A user profile describes user preferences in terms of (1) QoS setting for video, audio, still images and text, (2) the cost he/she is willing to pay to play the requested document with the desired quality, (3) time constraints, such as the delivery time, and (4) the importance factors. To decrease the system blocking probability, the QoS GUI provides facilities to set not only the desired QoS but also the worst acceptable QoS values.

3. An Implementation

In the context of a canadian collaborative research project [Won 97], we have designed and implemented QoS negotiation for the application of access to distributed multimedia databases, especially a news-on-demand demonstration prototype [Haf 96b]. The system allows the user to select a multimedia (MM) document and specify his/her requirements in terms of quality and cost for playing the document. At any time, the user may initiate a renegotiation to change his/her preferences, e.g. ask for better video quality or for lower cost. When QoS violations occur, the prototype performs an automatic adaptation whenever possible. A QoS manager performs the functions of negotiation and adaptation by interacting with various system components. In this section we describe the design of these functions which follow the principles of the general QoS management framework discussed in Section2. More specifically, we are concerned with the negotiation steps (1) and (3); step (2) is not considered since we assume a fixed functional configuration for the remote database access service.

This section is organized as follows: Section 3.1 describes the model used to specify a MM document. Section 3.2 presents the notion of a user profile which enables the (human) user to specify his/her QoS requirements. The main steps of the negotiation procedure are described in Section 3.3, and Section 3.4 defines the optimization scheme used during the negotiation phase.

3.1. Multimedia Document

A multimedia (MM) document may be composed of several monomedia, or consist of a single monomedia object. Figure 3 shows the structure of a MM document using the OMT notation for object-oriented analysis and design [Ram 91]. It shows that a document is either a monomedia or a multimedia, and that a multimedia is composed of one or more monomedia (aggregation links), and has attributes describing the spatial and temporal synchronization constraints.

The properties of a monomedia object depend on the underlying medium, which could be text, a still image, an audio sequence, or a video sequence. Each monomedia object may exist in several variants, which correspond to a format variants or multiple identical copies in different servers. Each variant is characterized by its QoS parameters. More generally, variants of the same monomedia may differ in terms of some static parameters which may concern the coding scheme, the size of the file, the associated QoS parameters, e.g. image resolution or color and audio quality, and the physical localization of the file. For example, two variants of the same video sequence could offer different color qualities; Variant1 may be a super-color variant of the video sequence, while Variant2 could be the black and white variant of the same video. A more detailed description of the MM document model used can be found in [Boc 96].

useful configurations corresponding to the requested service, and perform a simple negotiation with the components of one of these configurations.

2.3.2 Dynamic Management Information

The dynamic information describes the current state of system components, network paths performance, reliability measures, and accounting management. It also concerns information about the QoS aspects of the active (operational) configurations. To each managed object, e.g. system component, is attached a set of attributes which describe its performance state and attributes concerning the supported active applications. These attributes are updated by the QoS agent which periodically gets the QoS parameters of the component, such as transit delay and loss rate. This dynamic management information provides a global view of the system and the performance of the active applications. The data stored may be used to compute statistics such as the number of violation of type1, the time of the last violation of type1, the mean time between two violations, the time spent to recover from a violation, the number of times that renegotiation is initiated because of QoS violation, or the most frequent cause of QoS violation. Such statistics may be used by the QoS manager when establishing a configuration for a new instance of an application or when dealing with a QoS violation.

2.4 Negotiation Scenarios

On the receipt of a service request (Step1), the QoS manager translates this request into one or more functional configurations (Step2), and produces the corresponding physical configurations (Step3) by accessing the static management information. To decide about the possibility of the establishment of a physical configuration, the QoS manager asks for dynamic information about the current states of the system components, and runs some algorithms to select an optimal configuration. The activity of the QoS manager results in the establishment of a new physical configuration which supports the user request or simply the rejection of the request with preferably an alternative proposal. During the active phase of the application, the QoS manager monitors the QoS of the established configuration, that is, it compares the agreed QoS against the QoS currently provided. When the agreed QoS is violated, the QoS manager tries to establish a new configuration that meets the original user requirements, based on the static information and an updated version of the dynamic information. This activity is called "renegotiation initiated by the system". If the QoS manager is unable to find a suitable new configuration, it does one of the following (depending on the established policy): ignore the violation, initiate a renegotiation with the user about the required QoS, or terminate the application. At any time, the user may take the initiative to renegotiate the currently provided QoS. If the new QoS requested is less restrictive than the original one, then the components may simply relax their commitments. In the case where the new QoS requested is more restrictive than the original one, a new negotiation (Steps 2 and 3) is performed before a new configuration is established.

indicates the amount and direction of the QoS change which is requested. In the case of an increase of the requested QoS, admission tests may be run before accepting such a request.

2.3 Information for the Negotiation Process

Different types of information must be available for the negotiation process described above. We distinguish two types of information, described in more detail below: (1) static information, which does not depend on the actual system load (e.g. maximum packet size supported by a given network, maximum bandwidth supported by a network connection, or a compression scheme supported by an end-system) and (2) dynamic information, which depends on the system state and its load (e.g. currently available bandwidth). These types of information should be readily available for the negotiation process, possibly at the different sites where the QoS management decisions are made. We assume in the following that the information is available in some abstract management information base (MIB) which may be accessed as required. A possible implementation of this MIB would be within the QoS agents which each could be responsible for providing the pertinent information concerning its own physical component.

2.3.1 Static Management Information

For a given service, at least the following information must be stored: (1) the identifier of the service, (2) a description of the service, its parameters and its return type, (3) a description of the service functions and their performance, such as the response time, etc., (4) the precedence relation between the service functions (the order between the service functions‘ executions), and (5) the locations throughout the system where the different service functions can be offered. The static information concerning the whole system includes a description of the system components and the topology of the system. For a network, for instance, the following information could be stored: (1) maximum packet size which must be known for estimating the end-to-end delay, (2) maximum throughput supported, depending on the links between the network nodes, (3) the charging scheme, which is required to compute the communication cost for providing the requested service, (4) the type of guarantees provided by the network, e.g. best-effort or guaranteed QoS, (5) the support of constant bit-rate and/or variable bit-rate, (6) the suitability of the transport for services with bit-rates varying from tens to tens of millions of bits per second, (7) the support of connection and/or connection-less data services, and (8) the support of multicast and/or broadcast services.

The static information may also include some "useful" physical configurations for a given service. We call a configuration useful if it is relatively optimized and has often been used successfully in the past. We think that a learning facility should be supported, especially for the more popular services which are often requested by the users. On the receipt of a service request from a client, a QoS manager could simply get the information about the

by the functional characteristics of the application, the user requirements, and the constraints imposed by the QoS characteristics of the given system components, such as the client's workstation and the requested document and its server station in the case of the News-on-Demand application. At each of the steps and sub-steps, the negotiation process may come to the conclusion that the assumptions made to this point cannot lead to an acceptable configuration for the application in question. In this case, the negotiation process should in general backtrack to a previous step and select an alternative proposal, such as an alternative compression scheme, or an alternate physical allocation of certain functions in Step 2, or an alternate coding format, implying an alternate functional configuration (with a different decoding functional unit) and in the case of the News-on-Demand application, possibly a different server station in which the document version is stored.

In the case that backtracking is necessary up to the first step, we have to introduce the scenario of renegotiation with the user, telling him/her that the requested QoS is not available and proposing possibly several available alternatives from which the user may choose.

2.2 A Component Model

In order to support the choice of appropriate QoS parameters for the different functional units of the application and the effective allocation of the corresponding resources, we associate a QoS agent with each physical system component (Figure 2). The QoS agent of a component is an object which provides operations to handle all the information about the performance and the functional possibilities of the component. Basically, two operations are foreseen: an inquiry function (service_request) by which another entity, such as the QoS manager, may obtain information about the currently available QoS parameters, and a reservation function (reserve_request) by which a specified local function can be reserved at a specified QoS level. The execution of a reservation request implies the allocation of the corresponding resources to the requesting application.

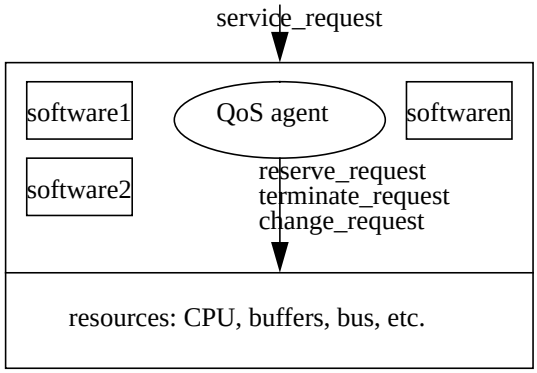

Figure 2. A Model of System Component

The QoS agent may run admission tests [Fer 92, Top 90, Han 91, And 91] in order to determine which QoS values can be currently promised, or in order to determine whether a given service request can be granted. In the case of renegotiation, the requesting entity may invoke a reservation change request (change_request) which

end delay but require huge bandwidth, while another one may require less bandwidth but does not provide a high quality of the image. Aware of these possible differences, one may associate a QoS profile with each functional configuration. A profile can be defined as a set of abstract related QoS requirements. We may for instance adopt a classification by delay and loss as described in [Nah 93], which considers four general profiles: loss not allowed and delay guarantees required, loss not allowed and delay guarantees not required, loss allowed and delay guarantees required, loss allowed and delay guarantees not required.

2.1.3 Selecting a Physical Configuration

During this step, an operational configuration for the application is selected. This process may be subdivided into four sub-steps as follows:

(a) Functional allocation to physical system components: Based on the knowledge of the available physical system components, the availability of functional software units on these components and their approximate performance characteristics, an allocation of the functional units identified in Step 2 onto the physical components is established during this step. This includes the choice of appropriate networks which provide for the data transmission functions specified by the links of the functional configuration if the interconnected functional units reside on different physical components. In certain cases, intermediate system components may be introduced, such as in the case where a stand-alone conversion service is used in order to convert the data from the file server into a format acceptable by the client workstation. If possible, this step should take into account the current availability of resources on these components.

(b) Refinement of functional configuration: Examples of such refinements are the choice of a compression scheme, which must satisfy the required delay and reliability, or the choice of a transport protocol which may be based on error control functions, performance, and other QoS parameters.

(c) Optimizing the resource allocations: During this sub-step, a set of QoS parameters for the functions provided by the different system components will be established within the limits of the available resources and in such a manner as to minimize the costs and maximize the service quality.

(d) Resource commitment: During this sub-step, commitment for the resources identified during sub-step (c) will be obtained from all involved system components. This completes the negotiation process and the active phase of the application may begin or, in the case of renegotiation, may proceed.

2.1.4 Optimization and Backtracking

It is clear that Step 2 and especially each of the first three sub-steps of Step 3 involve many opportunities for performance optimization. However, this is outside the scope of this paper.

It is important to note that the negotiation process described above evolves within the QoS constraints provided

The QoS requirements are initially defined in terms of user-level parameters and must be translated into the corresponding internal QoS parameter values in preparation for the subsequent step.

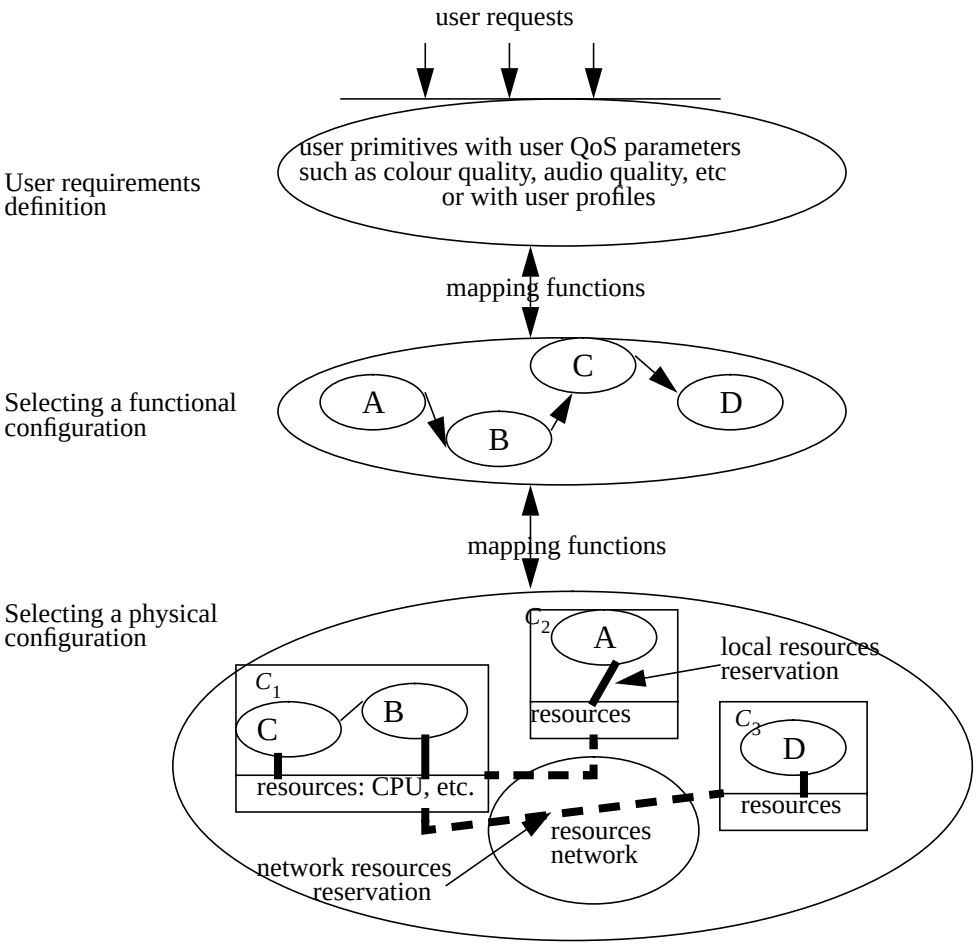

Figure 1. Steps of Negotiation

2.1.2 Selecting a Functional Configuration

Based on the requested service and the specified QoS parameters, there are usually a large number of functional configuration that could be considered. The purpose of this step is to propose a suitable functional configuration. We assume that a functional configuration can be represented as a graph, where each node represents a functional unit, and an edge represents a link between such functions. To each node of the graph is associated a set of attributes which characterize the corresponding function. Apart from the definition of the logical function provided by the unit, the attributes may characterize the unit in terms of resource requirements, such as buffers and CPU slots. Furthermore, each link is associated with attributes that characterize the requirements for data transmission between the nodes.

It is obvious that for a given service, different functional configurations may have different characteristics. Depending on the QoS parameters requested by the user, and depending on the presence of a synchronization scheduler and the coding format in which the document is stored, some configuration may allow for larger end-to-

2. A General Quality of Service Management Framework

During the negotiation or renegotiation of the QoS parameters of an application, a system configuration should be found, if possible optimized, which provides the requested service quality at an acceptable cost. It is the task of the QoS manager, possibly through the execution of a distributed QoS negotiation protocol, to determine such an acceptable configuration. The choice of the configuration has to take into account the different functions that must be provided for the application, such as video decoding, inter-stream synchronization, and data transport. But most important is the allocation of these functions to specific hardware components within the distributed system, such as the client workstation, the server station, or the network, and the allocation of the necessary local resources to guarantee the required service quality.

It is the task of the QoS manager to find such an acceptable configuration involving the functional and physical system components. In this section we describe a general architectural framework for the realization of this process which should lead to an acceptable configuration, if such a configuration exists. In this section we make abstraction from any particular negotiation protocol which may be used in the distributed system. A particular negotiation protocol is described in Section 3. Here we consider the QoS negotiation process from a global perspective.

2.1 Negotiation Steps

Within our negotiation framework, we distinguish three main steps for the negotiation process, as shown in Figure 1: (1) defining the user requirements, (2) selecting a functional configuration, and (3) selecting a physical configuration. During the first step, the user requirements are established through interactions with the user. During the second step a possible functional configuration is established. For certain applications, this may be a pre-established, fixed configuration of functions. During the third step, these functions are allocated to suitable physical system components, taking into account the available and required resources.

Figure 1 shows a case where the satisfaction of the user request requires four functional units: A, B, C, and D. During the third step, they are allocated to the physical system components as follows: C and B to C1, A to C2 and D to C3.

2.1.1 Defining the User Requirements

The user is the main actor for the specification of the required QoS. This specification may take place during the establishment of a new session, at the beginning of the presentation of a a new document, or in case of renegotiation with the user when the originally defined QoS cannot be attained by the system. The selected QoS parameters usually involve qualitative and quantitative parameters. In the case of the News-On-Demand service, typical qualitative parameters are color quality, frame rate, and the quantitative parameters may be cost and presentation delay. Some of these parameters may be automatically included from the user profile information.

play and pause a multimedia presentation.

- QoS violations, and the corresponding adaptation activities, should be dealt with automatically, if possible, and only require user/application intervention when built-in strategies fail.

- Meta-information on multimedia documents is required for efficient QoS management, especially QoS negotiation. Such meta-information may be used to find the most suitable system components to support the requested QoS.

- Significant progress has been made to support a guaranteed level of QoS by a single system component. However, to support the requirements of distributed multimedia applications, there is a need for an overall integration of the QoS management functions of all system components.

We believe that an ideal QoS architecture should support the following functions:

(1) To support a dynamic choice (in opposition to the static choice provided by current QoS architectures) of a suitable configuration of system components to support the QoS requirements of the user of a specific application; optimization criteria are the cost for the user and the overall resource utilization for the system.

(2) To support more complex flow services which may require specific processing functions, e.g. the synthesizing function.

(3) To be essentially independent of the mechanisms used within each individual system component for providing QoS guarantees.

(4) To provide a hierarchical management process (instead of an approach based on the sender or/and receiver) which naturally reflects the hierarchical structure of distributed systems.

(5) To return, in response to the user request, some useful information, especially when the request is rejected. Examples of such information are: the QoS currently available and the QoS which will be available at some specific future time.

(6) To provide QoS adaptation mechanisms which allow to automatically maintain the initially agreed QoS (instead of initiating the renegotiation of a degraded QoS, or of aborting the session).

In this paper, we present a general framework for QoS management for distributed multimedia applications which has the above attributes. We concentrate on the first three attributes. A more detailed description of the issues related to the attributes (4), (5), and (6) can be found in [Haf 95], [Haf 97a], and [Haf 97b], respectively. This framework is presented in Section 2. How such a framework can be realized is described in Section 3, where a presentational multimedia application is considered which provides access to remote multimedia databases, such as News-on-demand. The ideas presented have been implemented in a prototype system of a News-on-demand application [Won 97].

provided either using a separate protocol for each service, (e.g., the Tenet approach: a protocol for real-time message transport and another protocol for continuous media transport), or a single protocol which can be parameterized to allow for a flexible adaptation to the user's QoS requirements, e.g., XTP.

The second class of architectures concentrates on providing service guarantees with respect to network and end-system resources. They integrate the support of resource management in the network and in the end-systems. The main activity of end-system resource management concerns the scheduling of multimedia streams in the end-systems based on application requirements, specified as transport QoS parameters such as delay and throughput. To achieve this goal, real-time scheduling policies and memory (buffer) reservation schemes are used to perform such functions as transport protocol processing, e.g., fragmentation and error control, stream synchronization, and data movement.

The third class of architectures provides QoS guarantees on an application-to-application basis. This implies that the resource management for both end-systems and the network is based on requirements provided by the application/user in terms of high level QoS parameters, such as frame rate and video color (and not the QoS requirements of a single connection, stream or flow). For example the OMEGA architecture [Nah 95] provides mechanisms to negotiate, during the call establishment phase, QoS guarantees for all incoming and outcoming connections of the application.

In summary, all the architectures presented above focus on communication systems and operating systems, that is, they concentrate on resource reservation and real-time scheduling issues. The mechanisms and schemes proposed are used in a rather static manner, since the entities involved in QoS support, e.g. the network, sender and receiver, are known before the connection (call) set-up phase. The negotiation mechanisms provide the user with the QoS that can be supported at the time the service request is made, and assumes that the service is requested for indefinite duration. Last but not least, the provided adaptation schemes react to QoS violations by renegotiating a degraded service or simply by aborting the session.

We conclude that a number of important issues have not been sufficiently addressed by the reviewed approaches, such as the following:

- Any QoS management function, especially QoS negotiation and renegotiation, should take into account the cost to be charged to the user. This issue has been raised by a number of researchers, but no specific proposal has been made.

- The user interface is an important component for the success of multimedia applications. An appropriate interface should provide facilities allowing the user to specify his/her QoS and cost requirements, to perform QoS management functions such as (re-)negotiation, and to control the execution of the provided service, e.g. to stop,

schemes and scheduling policies, and to accommodate new system component technologies.

Keywords: multimedia applications, quality of service, management, negotiation, news-on-demand, multimedia document, cost

1. Introduction

Most work related to quality of service (QOS) is concerned with individual system components, such as the operating system or the network. However, to support distributed multimedia applications, the entire distributed system must participate in providing the guaranteed performance levels. In recognition of this, a number of QoS architectures have been proposed. These architectures can be classified into the following three classes.

(1) QoS guarantees at the network and transport level: The QoS architectures of this classes consider network QoS requirements of the application and concentrate on providing guarantees by making use of resource reservation protocols within the network along the path between the communication entities. These protocols are sender or receiver oriented. Examples of architectures which provide QoS guarantees at this level are Tenet protocol suite, the Internet protocol stack and the Native-Mode ATM protocol stack [Fer 95, Zha 95, Kes 95].

(2) QoS guarantees at the network, transport and end-system level: The QoS architectures of this classes take into account, in addition to network resource reservation, the resources in the end-systems, such as memory, CPU processing, and network interfaces. The services considered concerns mainly data flows; that is, the goal is to establish connections with QoS guarantees between two or more fixed application entities. Examples of QoS architectures which provide QoS guarantees at this level are QoS-A, End-System QoS Framework and Heidelberg QoS Model [Cam 94, Gop 94, Vol 95].

(3) QoS guarantees at the network, transport and end-system level including the application layer: These architectures take into account the application requirements in terms of high-level QoS parameters, such as image size and audio quality, and make use of resource management to provide end-to-end guarantees. Negotiation between the source and the sink end-systems is provided as part of the session establishment procedure. A significant example of this category is the OMEGA architecture which is the result of a research effort which examines the ability of local and global resource management to satisfy QoS requirements of applications [Nah 95].

The first class of architectures consider network QoS requirements from the application and concentrate on providing guarantees with respect to the network resources. The main activity consists of evaluating the capacity of network resources to accommodate a new connection without affecting the guarantees provided to the other existing connections. A connection is characterized by its traffic model, and its end-to-end performance requirements. These architectures provide deterministic, statistical or best-effort guarantees. The services are

An Approach to Quality of Service Management in Distributed Multimedia Application: Design and an Implementation

Abdelhakim Hafid[1] and Gregor v. Bochmann[2]

[1]Computer Research Institute of Montreal, 1801 McGill College Avenue,

Suite 800, Montreal (Qc) Canada H3A 2N4

E-mail: ahafid@crim.ca

[2]Université de Montréal, Dept. d'Informatique et de Recherche Operationnelle (DIRO),

Montréal, H3C 3J7, Canada

E-mail: bochmann@iro.umontreal.ca

Abstract: Most work related to quality of service (QOS) is concerned with individual system components, such as the operating system or the network. However, to support distributed multimedia applications, the entire distributed system must participate in providing the guaranteed performance levels. In recognition of this, a number of QoS architectures have been proposed to provide QoS guarantees. The mechanisms and schemes proposed by those architectures are used in a rather static manner since the involved entities, e.g. the network, sender and receiver, are known before the connection (call) set-up phase. In contrast to these architectures, we propose a general QoS management framework which supports the dynamic choice of a configuration of system components to support the QoS requirements for the user of a specific application. We consider different possible system configurations and select the most appropriate one depending on the desired QoS and the available resources. In this paper we present an overview of this general framework; especially, we concentrate on QoS negotiation and adaptation mechanisms. To show the feasibility of this approach, we designed and implemented a QoS manager for distributed multimedia presentational applications, such as news-on-demand. The negotiation and adaptation mechanisms which are supported by the QoS manager are specializations of the general framework. The proposed framework allows to improve the utilization of system resources, and thus to increase the system availability; it also allows to recover automatically, if this is possible, from QoS degradations. Furthermore, it provides the flexibility to incorporate different resource reservation