Adaptive Quality of Service Management

Piroska HALLER
“PETRU MAIOR” UNIVERSITY TIRGU-MURES ROMANIA
Str. N. Iorga nr 1. 4300 Tirgu-Mures, Tel/Fax: 40-65-162275
phaller@uttgm.ro

Abstract

The quality of services management in distributed application needs a lot of messages and complex reservation protocols at every communication unit. The proposed distributed and hierarchical architecture for different services offers more extensibility and efficiency to the application. But this architecture is part of the middleware and is transparent for the user level. Most of the systems use admission control policies and resource reservation for solve the QoS management. The new architecture and measurement based management policies, make possible the continuous adaptation of the QoS parameters to the modification of the requirement or the system resources. In the following we present an extended object model for the middleware platform and the case study for the design and implement an adaptive QoS management services based on the proposed model. The presented management system is designed to accommodate users with very different QoS requirement, which also can be changed in time. The resources managing system will be also implemented on several levels, in accordance with the needs in QoS on each specific level.

1. Introduction

In order to obtain performance it is necessary to create new techniques and specifications in designing, implementing and testing of the distributed multimedia systems. The defined extended model for distributed multimedia applications act as a middleware.

The middleware platform will contain a set of components which offer a set of services, while the creation of an application means the selection of a set of components and the description of the way they interact, not knowing the mode these components act internally. From the point of view of the application, its costs to obtain certain quality parameters are more important than the used algorithm. These services must have attached the cost parameter.

The advantage of the middleware platform consists in the global control assured on the applications, which use this platform. The bottleneck of the global control was solved by the distributed and hierarchical architecture for different services, that offer more extensibility and efficiency for the application. The conflict between the services for occupying the resources can be solved on an adaptive way, increasing this way the rate of using the resources. The algorithms used in the middleware platform can be changed dynamically according to the nature of the applications, and the system loading. The advantage of the adaptive algorithms against of the algorithms based on reservation is evident, for in most of the cases of distributed multimedia applications the variation of demands is very large and is not known beforehand. Most of the operating systems do not implement reservation techniques of the resources, but best effort techniques; therefore the using of these techniques on an intermediary level will certainly not lead to the results desired, but even in these cases the adaptive techniques will lead to both the improvement of the utilisation rate of resources and the maintaining the parameters of the system beyond the limits needed at least on the intervals. If the system implement some QoS management services on the communication level or operating system level, they can be integrated in the platform services. The QoS management services continuously monitor the system capacities, and distribute the requirements in optimal way, but not assure the resources reservation.

The platforms novelty consists in connecting two main aspects – the scalable architecture, and the managing of quality parameters of the system at the user level. Comparing with the existing platforms AGILOS[1], GOPI[2], Sumo-Orb[3], MCA[4], M3[5] we can emphasize some advantage of the presented platform, like support for group communication, the services are distributed, and hierarchically connected, the management decision are decomposed in different level, but assure the global optimality. The control of the QoS parameters at the operating system level or at the communication level is not the goal of the platform, but the open
architecture of the platform make possible the integration of the existing support.

2. The middleware platform

The whole system was built in order to run on different platforms and operating systems using only their minimal functionality. The computational viewpoint follows the Reference Model of Open Distributed Processing (RM-ODP) [6] and defines the system as a set of objects independent of location connected with explicit bindings to create communication groups. The objects are accessed through one or more interfaces that can be operational signal or stream interfaces.

The proposed extended model corresponds to the engineering viewpoint. On this level of abstraction the system may be considered as a set of terminal node objects which form communication groups. All the terminal objects will be registered at a central manager object to obtain a unique identifier, and to define those discussion groups it participates at. On this level are created multipoint communications between the terminals for different types of media. The terminal objects can vary considering granularity and can be an interactive user which presents the data asked from the server, a file server, a database server, an audio or video broadcasting source, a monitoring point, or any other combination of source and presentation object. An object will implement all the specific interfaces for the chosen media, moreover, new media may be added or others delete dynamically during the lifetime of a connection.

For each media of one terminal, QoS demands may be specified either in the form of admissible intervals, constant values, or in the form of priorities respectively in qualitative forms, but in the case where no value is explicated, the middleware platform will calculate the optimum value for each communication media, depending on both the system’s loading and the own transfer capacity of each channel.

The central manager object will comprise the managing of discussion groups, that is adding, deleting of terminal, creating, deleting discussion groups, control of the group’s parameters and rights. The manager object will assure in the same time the distribution of data with different QoS for each communicating channel of the group and each media of the channel. The manager object will abstract the middleware platform at this level.

The manager object seen to be as an object composed of many distributed active objects: group managers, user managers, QoS managers and stream managers; that communicate through messages. The active objects may be discomposed recursively in more simple distributed objects that communicate through messages. This hierarchical discomposing will be performed according to both the nature of the application and the complexity of the communication infrastructure.

Each object has a permanent link with the hierarchically superior object that has created it and

![Figure 1. The middleware platform](image-url)
registered it together with the existing interface types. This hierarchically superior object may be queried in order to localise another active object with which a communication channel will be created. Thus in case the loading of the system varies it is possible to move an active object or to dynamically replicate it. The hierarchical registration structure will be created according to the infrastructure of the system, but not being attached to the infrastructure of the system. The hierarchical organisation is fundamental for the scalability of the model. It provides a good response time, even if the inter-connection structure varies dynamically.

The central manager object acts as a root of the hierarchical registration structure. When the new terminal is connecting, it create a new terminal manager after the authentication and authorisation, for co-ordinate the message communication, and register the terminal at the required group manager. Due to the required media the terminal manager initiate the creation of explicit bindings with the stream managers. The stream managers represent a special set of active objects that create data links; they receive the data from the members of the groups and distribute it to the members that are listening for that media.

For each media we have a separate manager object that will be connected to the group manager through the message interface to receive the information from the modification in the structure or the group parameters. In order to reduce the number of the message transfers, each will contain a partial cache object with the group structure and with the user’s required parameters for that type of media. It may have a hierarchical structure, in case the dimensions of the group are large or the required parameters are very different and it may create controller objects for the subgroup.

3. The QoS management services

The monitoring, evaluation and control the QoS parameters in a distributed system require a very large number of messages. Create an hierarchical structure followed the proposed model reduce the number of messages, offer small response time for a new connection and make possible to renegotiate the QoS parameters without stop the data transfer. The QoS management service is distributed in every communication unit from terminals and stream managers. The central QoS manager unit has a general QoS controller unit that calculate the optimal distribution for the resources trying to satisfy the users’ QoS requirements. This module monitoring the QoS parameters for the communication group based on messages received from the distributed QoS monitors across the communication units. The parameter value represents an average and takes in consideration in negotiation or renegotiations phases.

The central object will generate at request periodical statistics about the state of communications that can be used to combine the platform with other platforms or control facilities of the quality parameters implemented at the level of the operating system.

When a new terminal connecting for group send a messages with required QoS parameters and guaranteed capacities for all communication media. The parameters are defined at minimum and maximum values. Due of the policies implemented in the QoS controller, it renegotiates the required parameters with the new user of with all members of the group in case of global QoS violation. When the user leave the group, or explicit change the required QoS the optimal value for QoS parameters for all media and for all members are recalculated.

Let’s note with m the number of the terminals, with n the number of media, the output rate to send date from terminal j to media i with r_{ij}, and the input rate to receive date in node j to media i with q_{ij}. If U_j is the total guaranteed transfer capacities for the terminal node j, and G_i the total guaranteed transfer capacities for the stream controller i, we can write the following inequalities, that should be satisfied:

\[ \sum_{j=1}^{m} r_{ij} + \sum_{j=1}^{m} q_{ij} \leq U_j, \quad j = 1, m, \]

\[ q_{ij} \leq \sum_{j=1}^{m} r_{ij} \quad j = 1, m \quad \text{and} \quad i = 1, n. \]

\[ \sum_{j=1}^{m} r_{ij} + \sum_{j=1}^{m} q_{ij} \leq G_i \quad i = 1, n. \] (1)
We can define the cost function, that minimise the difference between the prescribed input rate and the sum of the output rate from the other terminals (that affect also the delay, and the lost of the packets), respectively minimise the difference between the prescribed output rate \( r_{ij} \) and the required output rate \( r_{ij}^{*} \). To obtain a more equilibrated solution we introduce the weight coefficients for the terms of the cost function. In this way we can define priorities in the system and can attach cost for some strident requirement.

Lets note with \( p_{ij} \) the weight coefficients for input rate and with \( c_{ij} \) the weight coefficients for output rate, we can obtain the new cost function:

\[
f(r,q) = \sum_{i=1}^{n} \left( \sum_{j=1}^{m} p_{ij} \left( q_{ij} - \sum_{j=1}^{m} r_{ij} \right)^{2} \right) + \sum_{i=1}^{n} \sum_{j=1}^{m} 2mc_{ij} \left( r_{ij} - r_{ij}^{*} \right)^{2}
\]

(2)

The manager unit receive the required values, solve the quadratic optimisation problem with inequality constrains to obtain the prescribed optimal values for the input and output rate [7], then send the calculated parameters to the stream controllers and terminal nodes.

The presented mathematical model set out from the required output rates and capacities for all terminals, but they conflict with each other. The role of the central manager unit is to determine the optimal requirement for the whole system, starting with the user's demand, but take in consideration the constrains. The form of the cost function assure that the optimal point correspond with the required rates if the constrains are satisfied. In a distributed multimedia system all constrains can’t be satisfied simultaneously. The central QoS manager unit guarantees the global optimality and prevents the dominance of a group of terminals.

The determined parameters are averages values, for a period that take in consideration the maximal delay for the direct and feedback control messages. To prevent the system oscillation, the recalculation of the optimal parameters is initiated only if the modifications of the capacities are exceeding some threshold.

It exists a lot of situation, when the user can not specify the values the required parameters or the weight coefficients at the connection. The QoS manager unit offer a lot of services that accept the perceptive parameters like image dimension, image quality or audio quality level, and assign priorities for different media in accordance with the user choice. These qualitative parameters are transform in quantitative parameters need by the optimization problem. We can illustrate one strategy of transformation for n=3 media (video, whiteboard, file), m terminals, and p the level of priorities for video stream. Take in consideration that the delay for video streaming can not exceed the maximal value, the algorithm assure the minimal rate for video, even if the terminal bandwidth is low.

- if \( U_{i} \leq 3300 \) // video
  \( r_{1i} = U_{i}/2; \)
- else
  \( r_{1i} = U_{i}/2*(n+p); \)

- if \( U_{i} < 3300 \) // file
  \( r_{2i} = U_{i}/2*n; \)
- else if \( U_{i} < 15000 \)
  \( r_{2i} = U_{i}/6; \)
- else
  \( r_{2i} = U_{i}/4; \)

\( c_{ij} = \max\left\{ r_{ij}^{*}\right\}/r_{ij}^{*} \) and \( p_{ij} = \max\left\{ q_{ij}^{*}\right\}/q_{ij}^{*} \)

Other strategies can be also loaded statically or dynamically.

4. The stream controller

The determined optimal values for QoS parameters are distributed from the manager unit to the stream controllers and use them as a required value in the control algorithm. They monitor the local value of the parameters and receive the monitored values as feedback from the users. The monitoring information is globally accessible, via request, but at different level of granularity, favoring the implementation of local control.

The stream controller read data for the sources and delivery for the users at the required values of QoS parameters. Each data unit is associated with a time-stamp. The succession of the time-stamps describes the stream time. The scenario and synchronization relations are temporal relations defined over the time stamps as events, or over the intervals delimited at these time-stamps.

In this way the data generate rate can be adapt to the value of the best transfer rate, and different user can receive the same stream at different QoS values. The stream controllers also implement a scenario control and a primary synchronization mechanism. They support different mechanism of adaptation: filtering, frame reduction, hierarchical compression, mixing [8] or use the existing QoS techniques implemented at the communication level like Diffserv. The different media are routed in different way using separate controller to obtain the best usability for the network resources.

The controller receives a lot of messages that generates events due the transfer parameters from the external monitoring objects. The internal monitoring unit also generates a set of events. The monitoring system will hold information about the module as
well, module that varies the package number in the pipes. The pipes are used to equalize the jitter, but their dimension and read/write strategies can be adapt dynamically. There is a possibility to lose packages due to the fix length of the pipes, but this is signaled to the monitor.

The scheduler that deliver the data to the users, read the time-stamps of each data unit verify if they satisfy the temporal relations and send the data with the required transfer parameters. If the temporal relation can not be satisfied they also generate internal events. The received internal and external events are used to adapt the control and scheduler strategies. If the problem persists and the controller can not solve locally, send a message to the QoS manager unit and initiate a QoS parameter renegotiations. After the renegotiations phase receive the new required values that are used in control algorithm.

The proposed controller act as a classical numerical control unit, that generate an output at the required values, controlled the input and used the feedback values from the output. Under the assumption, that the network can not guarantee the required values for the delay, rate and jitter, and the operating system scheduling is non real-time, there is a need to accommodate this parameters in continuous way by the controller.

5. Conclusion

The presented extended model not describes a specific application. The objects are grouped in object types characterised by specific interfaces to create service. Together create a middleware service structure for multiparty communication, administration, authentication, and QoS management without depending on the type of the distributed multimedia application it participates at.

The hierarchical and distributed organisation of the manager units offer more dynamic aspects, the system can adapt to the required parameters, provide the scalability of the system. If the communication system has no support for QoS guaranteed transfer, the distributed monitoring and resources reservation modules offer possibilities for continuous media transfer. The QoS parameter violation can be localised and try to correct locally or the system can initiate renegotiations of the parameters if the problem persist in this way can adapt to the variation of the system's parameters.

The modules can adapt to various requirements for different user without change the structure. The central QoS management unit is used to co-ordinate the underlying resource management, in an integrated manner, but the distributed stream manager units make the whole system robust and assure the adaptive control of the QoS parameters. The resource management was solved like a classical hierarchical optimisation problem. The proposed cost function take in consideration the possibility of specify the user requirement in quantitative and qualitative manner, and the possibility to solve the problem in real time even if is a large system.

The case study focus on the QoS management service, but the presented methodology was used for the other services. The architecture of the proposed middleware platform is open, and new services can be added, if they are based on the extended model.

6. References