STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **XLVIII**, Number 1, 2003

SCALABLE PLATFORM FOR MULTIMEDIA GROUP COMMUNICATION

PIROSKA HALLER

ABSTRACT. The main aim of this paper is the development of a middleware platform, which will create the common abstracting level for different distributed media forming a communication group. The innovation of the presented platform consists in connecting two major aspects – the scalable architecture and the management of the quality parameters. In the same time the middleware platform has to be independent from the multimedia application that use it, and can run distributed on different platform.

We define an extended object model following the recommendation of the reference model, as basis for middleware service level. 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.

The proposed new architecture and measurement based management policies, make possible the continuous adaptation of the quality parameters to the modification of the requirement or the system resources at the application level, even if the network level not support guaranteed services.

Keywords: Distributed multimedia system, middleware platform, group communication, quality of services.

1. Introduction

The development of the interactive and distributed multimedia applications has introduced two new requirements, support for group communication and quality of services (QoS). The existing solutions on the level of the communication system cannot be adapted to this kind of multimedia applications. On the one hand the structure of the communication group may continuously modify, and the modifications of a terminal will affect all the members of the group. On the other hand the reservation-based quality of services will continuously block the unused resources, as it is very hard to estimate the requirements from the beginning. The advantage of the adaptive algorithms against of the algorithms based on reservation is evident.

Received by the editors: March 15, 2003.

2000 *Mathematics Subject Classification.* 68M14.

1998 *CR Categories and Descriptors.* D.2.11 [**Software engineering**]: Software architectures – *domain specific architectures* .

Presently the solutions follow two different directions. On the one hand the creation of several adaptive applications which would assure the transfer of the multimedia data even if the communication subsystem does not guarantee the quality of services[2,8], using different buffering and differentiated encoding techniques, but unfortunately they work on a relatively homogenous infrastructure. The other direction represents the extending of the communication subsystem with quality parameter maintaining services (Diffserv, Intserv), but this requires the modification of the infrastructure. Generally there are no integral solutions for maintaining the quality parameters for end-to-end connections on the application level. The effective usage of the services newly offered by the operation systems and by the communication subsystem is very difficult on the application level and it is necessary to create middleware platforms.

We suggest the creation of an middleware platform with a hierarchical and distributed structure, that will manage the communication groups for multimedia applications, providing the real time data access and the synchronous presentation of the data resulted from a distributed system. The proposed platform was implemented in collaboration with INTEGRASOFT in the HERMIX project.

The fundamental requirements that are impose for this platform represent the scalability of the system and the quality parameter management for end-to-end connections. The quality parameters management will control the distribution of the resources depending on the requirements, but based on globally optimal criteria, assuring in the same time the auto adaptation of the whole system at the modifications in a single node.

2. The Object Model

The computational point of view of the proposed platform corresponds to the RM-ODP[1] standard and defines the system as a set of location independent objects connected through explicit connections[4]. The objects can be accessed through the interfaces that can be message or stream type.

From engineering point of view the system may be considered as a set of terminal node objects forming communication groups. The terminal objects represent location independent entities that will be treated uniformly as encapsulated objects. These may vary in terms of granularity from a single communication media (for ex. text), to more types of communication media (for ex. video, audio, text, graphics, file), or to any number of identical media (for ex. 20 video streams). In fact a terminal node may be an interactive user presenting data required 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 objects.

The terminal objects represent a set of source objects, presentation objects, timer objects and a coordination object that is capable of generating, of processing and of interpreting scenarios[fig 1]. These scenarios represent temporal constrains between different types of data[7]. The terminal objects may implement two types

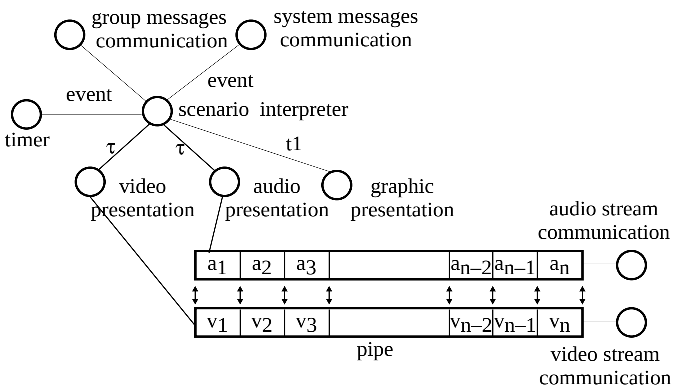


FIGURE 1. The structure of the terminal node

of communication: stream or message. The stream communication class may transmit notification events when receiving or transmitting a package based one the frame's time stamp, to the classes that have been registered for a certain event. Through this mechanism the synchronization or the presentation is supported according to certain scenarios. Each media will have its own communication object that will be dynamically created by the middleware platform.

Each terminal object has to contain message type communication objects representing the link to the middleware platform that provide the support for the management of the communication. The messages may be of different type: administration, system, error or group and will be transmitted asynchronously.

3. The Internal Structure

The middleware platform represents a set of services grouped in different modules: authentication, group management, quality parameters management, stream management.

The proposed platform has an open architecture, allowing for the integration of new services in case these correspond to the extended model. These modules may be created and dynamically loaded by the central manager object depending on the connection parameters specified in the terminal objects.

The central manager object is an object having a hierarchical structure, including different distributed active objects (group manager, terminal manager, stream manager, and QoS manager), communicating through messages[fig 2].

The active objects will include processing elements that support the parallel processing through the threads and will manage the communication channels for a subgroup of nodes, but only for a single media. Each active object will have a control object that will manage the rest and have a link to the hierarchically superior active object. The management consists in the creation, suspending, resuming or destroying of threads according to the occurrence of an internal activation condition, with the received message or the notification event received. In case the resources are no longer sufficient or the capacity of a resource is exceed, it may initiate the creation of a new active object similar or hierarchically subordinated anywhere in the distributed system.

Each object will have a permanent connection with the hierarchically superior object where it will be registered together with the type of interface it implements. This hierarchically superior object will be polled in order to localize another active object to which it is to establish the connection. Thus there is the possibility that an object to be moved or replicated depending on the dynamic modification of the loading of the system. This hierarchical structure will be created according to the requirements of the infrastructure, but it will not be physically related to it. Unlike the models in which the controllers are to be distributed in the network nodes[9], the suggested system will be distributed depending on the load and the capacity of the resources, providing thus the scalability of the system. It will allow a best response time for the system, even if the interconnection system varies dynamically.

3.1. **Active objects.** Considering the requirements of the multi-point communication for the management of the quality parameters, for the real time synchronization and their insufficient implementation on the operating system level we propose the usage of the active objects for the creation and the controlling of the multi-point communication. In the followings we will define the internal behavior of the active objects that compose the middleware platform, their distribution and the relationship between them.

3.1.1. *Access manager.* The security services (authentication and authorization) will be handled by the access management object, which is a centralized object linked to a database and which will be created together with the central manager object wherever in the distributed system and with which communicate trough asynchronous messages. It is a two level security system, each group having attached a set of rights related to the administration of the group and to the access modes for different types of media, and similarly each registered terminal may have a set of rights. The module may use the security system of another platform or of the operating system. In case the access rights for the terminal are dynamically changed the access management object will send notification messages to the terminal manager.

3.1.2. *Terminal manager.* The central manager as a result of the connection request received from the terminal node dynamically creates the terminal manager and a message type bi-directional communication will be established with it for the transfer of the system messages. At the connection of a terminal the central

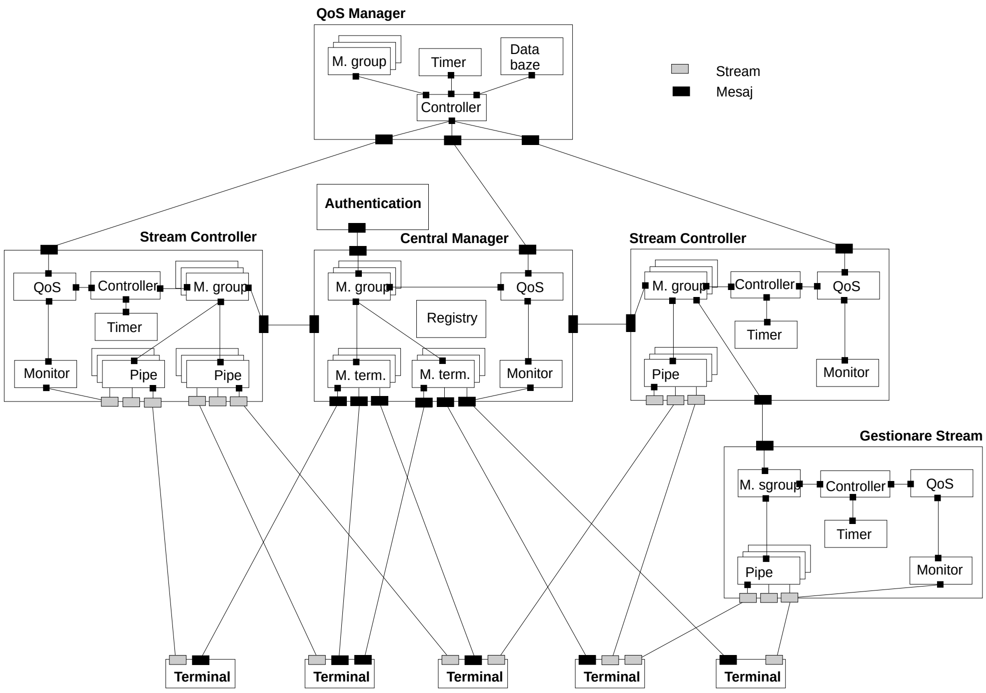


FIGURE 2. The middleware platform

manager object will initiate its authentication and all the data related to the rights and the terminal access will be stored locally in the terminal manager so that it should take over the verification task for the connected terminal. The terminal node will present a set of requested operations together with the requested transfer rate for these operations, and this value may be specified as an admission interval. The terminal manager will store the data related to the selected connection media together with the communication parameters. In case there are no explicit values specified for the quality parameters the terminal manager will generate these values based on certain qualitative options or priorities expressed by the terminal node and based on the monitored capacity. In order to implement the quality management system each channel will be characterized by the quality parameters such as the transfer rate, the delay, the jitter and the error rate. This transformation function of the quality parameters (linguistically expressed) in values or admissible intervals may influence the performance of the whole system.

3.1.3. *Group manager.* As the terminal nodes will be organized on discussion groups, for each of them a group manager object is created dynamically, that will administer and will distribute the group messages as text, whiteboard and graphics, transmitted on connections different from the ones used by the system messages. The group manager will have separate data connections implemented through message communication objects for each terminal with traffic monitoring. Even for asynchronous message traffic there are prescribed transfer rates. The group manager will have handled the administration of the members of the list, respectively the verification functions of the rights for the group. It will communicate with the access manager object in order to obtain the list of rights and the eventual dynamical alterations that occur. It will send notification messages to the stream manager objects in case the structure of the group changes, in case a member is added to the group or is removed or in case a terminal changes its options related to the communication media.

3.1.4. *Stream manager.* The stream manager objects represent a set of special active objects handling the creation and the control of the stream data transfer among the terminals for a certain type of media.

The creation of different subtypes for different media is necessary as the control algorithms and the distribution of the data differs very much depending on the media type. In case the large number of communication groups requires it, more similar distributed stream manager object instances may be created, being registered at the central manager.

The stream manager object will have a bi-directional connection with each terminal implemented through stream communicators, through which it will receive data from the terminals and will distribute them to the members of the group. The data transfer has to respect the quality parameters established by local quality manager system, so that embedded controller and monitoring objects will be used. The timer objects will correlate the activity of the monitors and controllers, respectively will guarantee for the temporal constrains imposed for the continuous communication media[fig 2]. The controller objects will implement different techniques in order to reduce the sending rate depending on the attributes of the media, such as: filtering, reducing the frames, mixing, differentiated compressions. Each stream manager object is directly connected to the quality parameter manager object, where the monitoring data is regularly sent to and where the prescribed values are obtained for the data-receiving rate from the terminal nodes respectively for the data sending rates. The local quality parameter manager objects will handle the parameter administration inside a group, creating thus a hierarchical coordination structure. In the same time they will communicate with the quality parameters manager object, in order to send feedback messages in case the prescribed values cannot be satisfied with implemented algorithms. These messages initiate the re-negotiation of the prescribed values and the modification of the control algorithm parameters.

3.2. **Quality parameters management.** The architecture of the QoS management services includes the central QoS manager, the monitors and controllers distributed in the terminal objects and stream controllers. The role of the central manager unit is to determine the optimal requirement for the whole system, starting with the terminal's demand, but take in consideration the system constrains. 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.

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 manager, it renegotiates the required parameters with the new user of with the 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.

Lets 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:

$$\sum_{i=1}^{n} r_{ij} + \sum_{i=1}^{n} q_{ij} \leq U_j, \quad j = \overline{1,m},$$
$$\sum_{j=1}^{m} r_{ij} + \sum_{j=1}^{m} q_{ij} \leq G_i \quad i = \overline{1,n} \quad (1)$$

We define the cost function, that minimize the difference between the prescribed input rate q_{ij} and the sum of the output rate q^*_{ij} from the other terminals (that affect also the delay, and the lost of the packets), respectively minimize 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.

$$f(r,q) = \sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}\left(q_{ij} - q^*_{ij}\right)^2 + \sum_{i=1}^{n}\sum_{j=1}^{m} c_{ij}\left(r_{ij} - r^*_{ij}\right)^2 \quad (2)$$

The central QoS manager unit guarantees the global optimality and prevents the dominance of a group of terminals solving the quadratic optimization problem minimizing the cost function (2) with inequality constrains (1)using inertia controlling methods [5]. When the structure of the group are changed, or the monitoring units detects change in the network capacities, the manager unit recalculate the optimal prescribed values. To prevent the system oscillation, the recalculation of the optimal parameters is initiated only if the modifications of the capacities are exceeding some threshold.

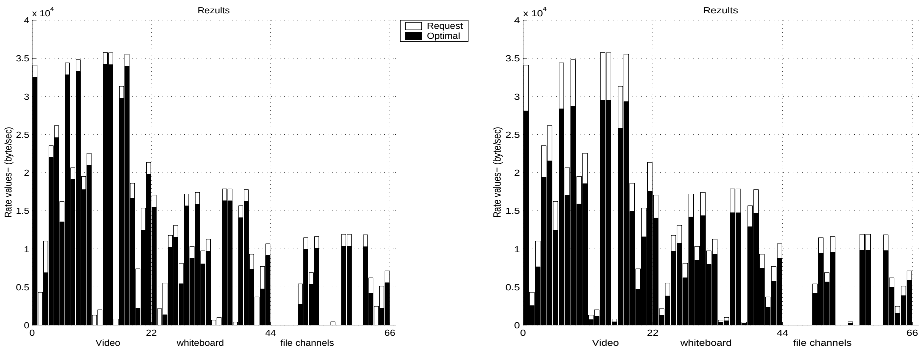
Using weight coefficients can obtain a more equilibrated requirement, at optimal cost. The weight coefficients can be defined by the user as extra cost to obtain the prescribed value near the required value, or can be generated by the manager unit according with the system loading. The user can modify the required values, or specify new priorities in system, or new qualitative option and the system should be adapt to the modification of the parameters.

We present some results, for one group with 22 users and 3 type of media (video/audio, shared graphics, file). Starting from the measured capacities and required transfer rate we calculate the optimal output transfer rate for each users on each media. The left diagram (figure 3) presents the results if the weight coefficients are equal. We can remarque the dominance of the high capacities channels. The slow channels can't send data, they only receive. Using the weight coefficients, where the costs are proportional with the capacities also the weak channels can send data (right diagram of figure 3). But the weight coefficient could be modified dynamically by the users with administrative rights.

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. The introduction of the weight coefficient proportional with the costs in the criteria function allows for a better equilibrium of the system.

The calculated values will then be transmitted to the quality parameters manager objects distributed in the stream manager objects which will treat them as prescribed values and following it there will be selected the control algorithms corresponding to each local controller in part in order to maintain the rate and the delays in the prescribed limits.The proposed stream 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, the presented solution assure the adaptation of the parameters in continuous way[6].

The most important role besides the controllers pertains to the monitoring objects that will be distributed in the terminal nodes as well. In case it is necessary the monitoring nodes may be distributed in the physical communication nodes as well, following the communication infrastructure.



user capacities in kbyte/sec:

822	172	441	806	810	649	834	809	835	780	805
52	80	837	836	6	818	836	744	296	614	810

FIGURE 3. The calculated optimal rate values

The monitoring of the system will include the classical communication parameters such as the transfer rate, delay, jitter, and the parameters specific to the active objects, such as the package loss due to the pipes dimension, compression factor, the temporary deviation from the prescribed value.

4. CONCLUSION

The proposed scalable architecture allows for the creation of certain application that will handle the connection of certain distributed terminal nodes – the number of which varies dynamically on a large scale, which will form the communication groups that will be configured and reconfigured dynamically. The management of the quality parameters will include the possibility to specify these requirements deterministically, probabilistically and stochastically. It will also include a support for the static and dynamical administration of the quality parameters, including the specification, admission negotiation and re-negotiation of the parameters, the controlling and the allocation of the resources, the monitoring of the system, independently of the degree it is guaranteed on the operating system's level.

On the other hand considering the platform as a classical distributed control system in which each type of media will be controlled by a controller object that will maintain the outgoing rate prescribed by the quality management system, controlling the incoming rate and monitoring the perturbations in the system, represents a very new approach. The quality management system will dynamically modify the temporal constrains for an object in order to maintain the optimality of the whole system, and in case it is necessary it will initiate the replication of the respective object. The proposed mathematical model allows the continuous adapting of the parameters to the modifications of the system.

References

[1] G. Blair, J. B. Stefani: Open Distributed Processing and Multimedia. Addison Wesley, England, 1998.

[2] G. Coulson: A Distributed Object Platform Infrastructure for Multimedia Application. Computer Communications Vol 21, No 9, July 1998, 802–818, 1998.

[3] R. Davison, J. Hardwicke: A New Architecture for Open and Distributed Network Management. H. Zuidweg et al. (Eds.): IS&N'99, LNCS 1597, pp. 25-37. Springer-Verlag Berlin Heidelberg 1999.

[4] D. Duke, I. Herman, M. S. Marshall: PREMO A Framework for Multimedia Middleware, Specification, Rationale, and Java Binding. LNCS nr 1591, Springer Verlag 1999.

[5] P. Gill, W. Murray, M. Saunders, M. Wright: Inertia Controlling Methods for General Quadratic Programming. SIAM Review, vol 33, No. 1, 1–36, 1991.

[6] P. Haller: Contributii la implementarea sistemelor multimedia distribuite, PhD thesis University of Cluj-Napoca, 2002 Cluj-Napoca Romania.

[7] S. Ing, S. Rudkin: Simplifying Real-Time Multimedia Application Development Using Session Descriptions. H. Zuidweg et al. (Eds.): IS&N'99, LNCS 1597, pp. 305-314, Springer-Verlag Berlin Heidelberg 1999

[8] B. Li: AGILOS: A Middleware Control Architecture for Application-Aware Quality of Service Adaptations, phD thesis University of Illinois at UrbanaChampaign, 2000 Urbana, Illinois

[9] B. Li, W. Kalter, K. Nahrstedt. A Hierarchical Quality of Service Control Architecture for Configurable Multimedia Applications. in Journal of High-Speed Networks, Special Issue on QoS for Multimedia on the Internet, IOS Press, Vol. 9, pp. 153–174, 2000.

Petru Maior University Târgu-Mures, Romania, 1, Nicolae Iorga St., 4300 Târgu-Mures, Romania

40-265-211838

E-mail address: `phaller@upm.ro`