Controlling and Managing Distributed Multimedia Applications*

José F. Rezende† and Serge Fdida

Laboratoire MASI-CNRS
4, place Jussieu
75252 Paris Cedex 05 - France
e.mails: {Jose-Ferreira.Rezende, Serge.Fdida}@masi.ibp.fr

Abstract

This paper describes a service by which users will be able to select, configure and activate multimedia services in a multivendor broadband environment. The service interface hides the dificulties and details of handling these complex services from the applications. An abstract model is defined that accommodates an extensive set of services involving communication and end-systems. The paper defines several abstractions, describes these abstractions by an object-oriented approach and defines a set of primitives for the manipulation of these objects. To show the applicability of our service, a video-on-demand distribution scenario is used.

1 Introduction

The emergence of distributed multimedia applications on top of multiservice networks has led to the appearance of an important research area that deals with the control and management of multimedia services in a broadband network environment.

To meet the distributed multimedia application requirements for multiservice and multiparty communications, Quality of Service (QoS) and application context services, considerable changes must be done in the conventional services commonly used for this purpose. The manner in which these requirements are expressed using the existing services influences strongly the final quality of service obtained. Thus, we must find new ways for these applications to describe their needs.

An important number of studies has been carried out in communication and operating system support for distributed multimedia applications over high speed networks. Furthermore, few work has been done in the design of an efficient service to support the control and development of general purpose applications in a broadband environment.

The work presented in [1] provides an abstract service description of communication systems supporting multimedia multi-user applications. In [2], a signaling protocol to allow applications to invoke multimedia services in the context of a broadband ISDN testbed is presented. The preceding works present as drawback the fact that they concentrate merely on the capabilities of the communication systems. In [3], an open platform supporting multimedia multi-user communications has been defined. It provides an application programming interface (API) that makes available network-provided capabilities and core services to multimedia applications. Nevertheless, it lacks for specification of higher layer functions as inter-media synchronization and presentation. It takes into account only those aspects related to the physical resources for the information transport.

In this work, we define a service to be used in a heterogeneous multivendor environment that provides an application with the capability of specifying, setting up, modifying and terminating a complete multimedia session (MM-session).

In the definition of the service interface, our main concerns are to provide abstractions that remove from the user the burden of coping with complex services and to provide users with a maximum flexibility to invoke multimedia services. To reach this goal, we define an abstract model that takes into account a set of services

*This work has been supported by CNET-France Télécom under grant n°92 1B 178 as part of the CNET-CNRS CESAME project on the design of high speed multimedia cooperative systems. Any views expressed in this paper are those of the authors alone.

†Supported by a grant from CAPES/Brazil.

needed to support a distributed multimedia application. It integrates, at the same level of abstraction, services that belong to different domains.

This paper is structured as follows. In the next section, the multimedia application requirements are described. In section 3, we outline our service. Section 4 describes an abstract model and its related objects. In section 5, we define the set of primitives provided by the service. In section 6, a video-on-demand scenario is used to illustrate its applicability. Finally, section 7 presents our conclusions.

2 Requirements from distributed multimedia applications

A system supporting distributed multimedia applications must cope with a wide diversity of requirements. So, before to describe the capabilities provided by our service, we found necessary to describe these requirements in terms of both the communication and the end-system.

2.1 Communication system

The main requirement related to the communication system is the need for multiparty and multiservice communications, and all the others are somehow associated with it. It refers to the communication between multiple participants using multiple traffic streams (e.g., audio, video, text, graphics, etc.). Many capabilities are needed to cope with it:

1. support for dynamic association between multiple services - when an application communicates using multiple services, it exists the need for associating these different services within a same context to ease the joint control of related connections.
2. support for specifying relationships between multiple services - different kinds of relationship can exist between a set of traffic streams. One of the more important is the synchronization that can be: inter-stream (continuous and event-based synchronization), ordering and mixing.
3. support for connections with asymmetric requirements - generally, the requirements for the forward direction are different from those related to the backward direction. For example, the bandwidth requirement in the requests and responses from a kiosk system is completely asymmetric.
4. support for Quality of Service control of both individual and multiparty communications - it consists of providing applications with capabilities to specify, negotiate and obtain guarantees of QoS. When multi-receivers are involved the problem of QoS control is still more complex, since each receiver might be able to state its own QoS requirements.
5. support for group communication - it refers to the exchange of data between two or more communication entities that pertain to a group. The services related to the group communication are those of group management (membership administration, directory service, group administration) and reliability.

2.2 End-system

The end-system requirements are mainly related to the operating system support. The capabilities needed are:

1. support for device control independently of the system configuration - the system must provide the application with a way to control multimedia devices (camera, microphone, video window, etc.).
2. support for presentation control of each service - it is helpful that users be able to control the presentation of each media (continuous and discrete). For example, different coding formats and/or video windows size could be used in a video communication. In some cases, the user needs to manipulate multimedia information.
3. support for synchronization between multiple media at the presentation level - the synchronization at the presentation level is a complex task that needs OS support to be achieved. The platform must provide the user with the capability to define synchronization scenarios that describe all the synchronization requirements of a multimedia session.
4. support for the description, selection and activation of application service elements - they are application-context services specifically tailored to support multimedia applications, such as floor control service [4].

3 The Multimedia Signaling Service

In this work, our main objective is the specification of a service called Multimedia Signaling Service (MMSS). It specifies the procedures by which the applications will be able to describe, establish, maintain and clear multimedia sessions (MM-sessions) in a multivendor broadband network environment.

A MM-session represents a multiparty and multiconnection association among a set of users. It encompasses multiple channels of different services (continuous and discrete media and control data) and end-system components, such as multimedia devices, application processes, etc. Besides the MM-session control, this service provides the multimedia applications with the capability to describe, select and activate application service elements (ASEs) that can be viewed as user-selected elementary services.

Defining this service aims to provide an abstract description of the future capabilities needed to support multimedia applications. It shows how the communication systems and the end-systems must evolve to best support these applications. It allows multimedia applications development during the evolution of heterogeneous network technologies and end-terminals.

An important feature for adequately supporting distributed multimedia applications is the need to provide them with a flexible and unambiguous manner to specify their requirements. In our service, all requirements are specified by means of abstractions, described later. So, the application states a logical specification of a MM-session that encompasses the communication topology, end-system components of each involved user, set up policies and additional services needed in the MM-session.

4 The Abstract Model

In order to provide the application with the ability to specify its requirements without concern with details and complexities of the underlying system, we have defined a set of abstractions. These abstractions have been grouped in what we call an *abstract model*. Its main objectives are to integrate, at the same level of abstraction, the different services needed by the applications and to release the application from concern about their physical realization.

Our abstract model is composed of objects that are used as building blocks for specifying the services required by a MM-session. These objects are viewed as logical abstractions and are composed by their attributes, interfaces and relations with others objects. Therefore, the specification of a MM-session is realized by both the objects' instantiation by setting the appropriate parameters and attributes, and the assignment of these objects to indicate the existing relationships among them.

Two types of services are identified in this model: the communication and the end-system services.

4.1 Abstractions of communication services

The four objects described below allow the application to specify their needs in terms of communication services.

- **Link -** It is an object that defines a unidirectional logical association among a set of user endpoints for a given service. Thus, it is typed by the kind of media that it carries (audio, video, voice, graphics, text, etc.). It is related to the transport capabilities of an association. But, it does not define an end-to-end transport connection by itself, since the access capabilities are not defined. The choice of unidirectional associations is due to the asymmetric requirement of most applications.

- **LinkAccess -** this object defines an access point of a user interface that sends or receives information according to a specific encoding format. It is typed by the user address, the direction of information flow (sink or source) and the type of information that it generates or receives.
  The mismatch of the encoding format between the information treated by source and sink *LinkAccess* objects provides evidence for the need of data conversion. Presentation transparency requires that the involved applications should not have to handle this conversion and should not even need to know about the conversion and the location of the converter.

- **QosLink -** instantiation of *QoSLink* object allows the application to specify the quality of service associated with *Link* object. It is used for QoS negotiation between peer users, network resource allocation and monitoring, depending on the facilities provided by the underlying system.

- **Relationship -** the instantiation of this object allows the application to specify the existing relationships among a set of *Links*.
  We identify three types of relationships: *synchronization*, *static* and *dynamic*. The *synchronization* relationship encompasses *Links* with temporal synchronization requirements. The *static* relationship establishes a relation of existence among the *Link* objects. It relates *Links* whose realization depends on the other *Links*. Finally, the *dynamic* relationship establishes a relation of activation of *Link* objects in

the timescale. In other words, it specifies the maximum number of *Link* objects that will be activated simultaneously.
In the *synchronization* relationship, the application may choose within: intra-stream synchronization, inter-stream synchronization, multi-destination synchronization and ordering (partial, causal or total).

4.2 Abstractions of end-system services

The abstractions provided below are related to the specification of the services performed in the workstations as well as mainframes which have multimedia devices attached to them. They are associated with the processing, storage, generation and presentation of multimedia data.

- **MMDevice -** this object is an abstraction of physical devices that represent either sources or sinks of multimedia data. To each device is associated a device description that contains all information with respect to the use of the device including addresses of device drivers and presentation information. By the instantiation of this object, the operations allowed to be executed by the application will be available in the interface of this object. They allow the applications to make use of physical devices without concern about their heterogeneity and variety. The operations supplied for any device are: produce data, consume data, switch information flow on, switch information flow off. Thus, the *MMDevice* object allow application both to specify the devices that will be used in a MM-session, and to control these devices during the execution of the MM-session.

- **DeviceAccess -** this object is an abstraction of the access point to which a multimedia device may be attached. Instantiation of this object provides a way for the application to define a termination point to which will be associated input or output buffers for the storage of multimedia data before its transmission or presentation. It is typed by both the direction of flow and the kind of information that it handles. The assignment of a *MMDevice* object to a *DeviceAccess* object denotes the attachment of a device to a network access channel and allows the system to reserve the corresponding resources.

- **SoftHandler -** this object is an abstraction of a software process that actively manipulates real-time continuous media data in an operating system environment. It may be viewed as an operating system thread acting as component of a flow, and may represent user processes specially elaborated to handle multimedia data. Instantiation of this object provides a clear way to make available an application code that handles data and must be taken into consideration not to jeopardize the end-to-end QoS specified. Scheduling of multimedia processes [5] added to the operating system resources reservation are of primary concern in current multimedia systems in order to guarantee the end-to-end QoS.

- **ScriptPresentation -** this object allows the applications to define scenarios to the presentation of multimedia data on the corresponding device taking into account the synchronization problems related to the multiple media presentation. It is associated with the *DeviceAccess* objects in order to carry out the synchronization at the presentation level of the devices attached to them. To define the synchronization scenarios different tools are under study. For example, TStreamPN [6] is a model that allows one to formally describe the timed behavior of multimedia objects and streams using an extension of timed Petri Nets.

5 Service Interface

The Multimedia Signaling Service provides a set of primitives that allows the application to set up, modify and terminate MM-sessions (table 1).

For each MM-session set-up, a negotiation takes place among all involved users and service provider. It is up to the service provider to carry out the mapping between the MM-session specification to its physical realization according to the services provided by the underlying services.

The MMSsetup primitive allows the initiator user to create a MM-session. The MM-session components are specified by the objects defined in the abstract model. The MMSsetup parameters are constituted of five blocks: add_user, add_service, attachment, set-up and extras (table 2). In the first block the initiator describes each involved user with the corresponding end-system components. In the second one, the communication properties of the MM-session are described. In the attachment block the user defines the complete topology of the MM-session by the assignment of the objects earlier defined. In the setup block, the policy to be used in the realization of the MM-session shall be determined. In the extras block, the additional services needed, in terms of functional unity's, shall be specified.

primitives	*function*
MMSsetup	establish a MM-session
MMSmodify	modify an existing MM-session
MMSrelease	end a MM-session
MMSstatus	MM-session status changed
MMSaccept	accept the changes
MMSrefuse	refuse the changes
MMSresult	result of MM-session modification

Table 1: Multimedia Signaling Service interface.

If the service provider is able to support the new MM-session, each involved user will receive a MMSstatus primitive. Its parameters are similar to the MMSsetup primitive. The MMSaccept primitive indicates to what extent the user accepts the creation, modification or termination of a MM-session. The user could change its own end-system components and topology. After receive all answers or after a time-out expiration, the service provider issues a MMSresult primitive either to indicate the successful creation of the MM-session or to indicate that it was rejected. The contents of the MMSresult parameters can be altered for several reasons. The users block contains only the users that answered to the invitation and agreed to join the MM-session. The communication and the extras block could have been modified by the service provider according to its capability to support the MM-session. The topology may be changed by each involved user to accommodate their new requirements. The Multimedia Signaling Service time sequence diagram is shown in the (figure 1).

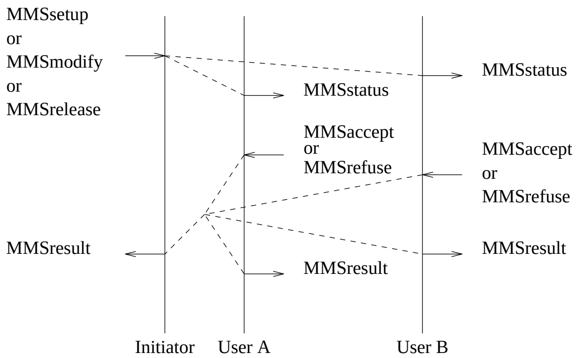


Figure 1: MMSS time sequence diagram.

6 VoD Distribution Scenario

Video-on-demand (VoD) services are likely to be the first of numerous multimedia services offered in the near future. This service provides customers with the ultimate flexibility of selecting movies for viewing at the time they wish. Therefore, to illustrate the use of the Multimedia Signaling Service, a video-on-demand distribution scenario is used.

In this scenario, a storage medium server (VoD server) delivers pre-scheduled programmes to clients in an *on-demand mode*. This means that it allows clients to receive just the desired programmes which are distributed according to a pre-arranged timetable. In addition, it allows some clients to control these programmes by the conventional VCR functions (pause, resume, fast-forward, or rewind). In this case, each viewer has a full interactivity with the service, i.e. Interactive VoD (IVoD).

Thus, each client may choose beforehand a number of desired programmes and for each available programme the server keeps a list of the clients that wish to obtain it. The quality of service required by each customer may differ - for example, some may require HDTV quality while other may need MPEG quality. After having subscribed to the service, a client may choose among the scheduled programmes those that he desires to acquire. The client also chooses the QoS for the programme between those proposed by the VoD service, and if he wants or not to make use of viewing control. Each available programme is distributed based

primitives	*parameters*
MMSsetup	@init_user, MMS_id,
	add_user(user_id): object_definition;
	add_service: object_definition;
	attachment: object_assignment;
	set-up: policy for MM-session set-up;
	extras: functional unity's selection;
MMSmodify	@init_user, MMS_id,action,
	[attachment: object_assignment;]
MMSstatus	@init_user, MMS_id,action,
	[attachment:object_assignment;]
	[extras: functional unity's selection;]
MMSaccept	@user, MMS_id, add_user or ch_user,
	attachment: object_assignment;
MMSrefuse	@user,MMS_id,reason
MMSresult	MMS_id,action,
	[attachment:object_assignment;]
	[extras: functional unity's selection;]

Table 2: Multimedia Signaling Service primitives.

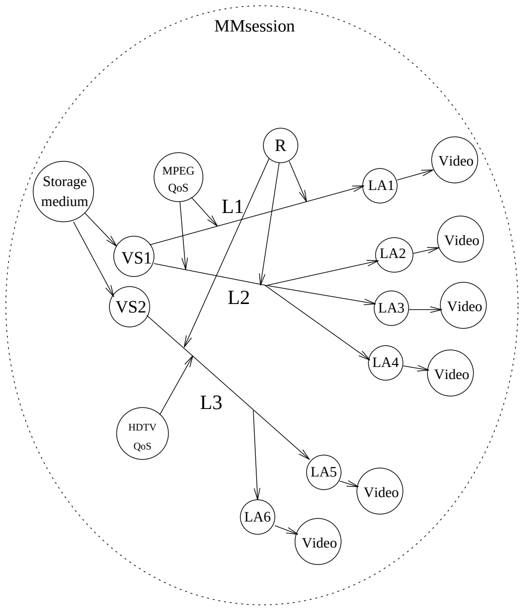
on a set of multicast address, one for each QoS provided. When a client chooses a programme, his address is automatically inserted in the address list associated with the QoS chosen.

For each programme to be distributed, the VoD server gathers the information supplied by all clients that wish to see the programme and builds the topology required for its distribution. Then, the VoD server sets-up a MM-session using these informations. Each MM-session (programme distribution) comprises a set of *Links* from the server to the customers to deliver the programme. There might be multiple *Links* for different QoS requirements. All these *Links* belong to the same MM-session and may be set-up and removed during the lifetime of the corresponding MM-session. Therefore, the VoD-server initiates a MMSsetup including the setup conditions and the required QoS (by the *QoSLink* object) for each *Link*. When individual and instantaneous control of the programme (virtual VCR functions) is required by a client, a *Link* dedicated to this client is necessary. For this reason, two different types of *Links* for the same MM-session may be used. The *Link* is used for all clients not using viewing control and a dedicated *Link* for each other client (figure 2).

Depending on the pricing policy used, the VoD server may express different constraints on the set-up of each MM-session. If we consider the case where only the clients that takes part in the MM-session will be charged (a "pay-as-you-watch" scheme), the VoD server may specify that "at least n clients" are needed in order to proceed, otherwise the MM-session does not take place.

It is possible for clients to join and leave on-going MM-sessions using the MMSmodify primitive. For example, clients who arrive late to the programme distribution may join the corresponding MM-session. It must be verified if MMSmodify leads to the creation of a new *Link* or if the new arriving may join an existing *Link*. In the same way, when a client leaves a MM-session, it must be observed if an existing *Link* must be removed from the MM-session.

In the example shown in figure 2, a video distribution is performed by the VoD server to six clients. As a result of the required QoS and viewing control, three *Links* (L1, L2 and L3) are necessary to carry it out. The *Links* L1, L2 and L3 are associated to one, three and two *LinkAccess*, respectively. Each *Link* has a *QoSLink* object associated to it. In the example, two *QoSLink* objects are defined. One that specifies the quality of service required to communicate using MPEG video and the other for HDTV. A *Relationship* (R) object is specified to establish the existing relations among these three *Links*. It establishes a *static* relation between them that may indicate, for example, that at least two of these *Links* must be established or else the



client_ID: <Addr.,VideoQuality,ViewingCtrl(Y/N)>

Programme A:

client1: @c1,MPEG,Yes
client2: @c2,MPEG,No
client3: @c3,MPEG,No
client4: @c4,MPEG,No
client5: @c5,HDTV,No
client6: @c6,HDTV,No

Figure 2: Programme Distribution.

communication doesn't make sense and the application doesn't want to carry it out. For each client, a video *MMDevice* object is defined. The storage medium device is also represented by this object.

7 Conclusions

In this paper we described a service that allows the multimedia applications to invoke multimedia services. We defined an object-oriented abstract model for the specification of distributed multimedia applications requirements. A video-on-demand scenario was used to illustrate the applicability of the service defined.

Emphasis was done on the integration of controlling and managing different services of different domains in a single service. Rather than separating the capabilities of the communication systems from those that are related to the end-systems, we argue for a complete service description, and hence, a unique interface used for this purpose.

The main contribution of our work is to provide a first step towards the definition of a general purpose interface able to hide the details of heterogeneous networks and end-terminals and also be independent of the system configuration up to a maximum extent.

References

[1] G. J. Heijenk, X. Hou and I. Niemegeers, "Communication systems supporting multimedia multi-user applications", *IEEE Network Magazine*, vol. 8, no. 1, pp. 34–44, Jan. 1994.

[2] S. Minzer, "A signaling protocol for complex multimedia services", *IEEE Journal on Selected Areas in Communications*, vol. 9, no. 9, pp. 1383–1394, Dec. 1991.

[3] M. Arango, L. Bahler, P. Bates, M. Conchinwala and D. Cohrs, "The Touring Machine system", *ACM Computer Communication Review*, vol. 36, no. 1, pp. 68–77, Jan. 1993.

[4] L. A.Crutcher and A. G. Waters, "Connection management for an ATM network", *IEEE Network Magazine*, vol. 6, no. 6, pp. 42–55, Nov. 1992.

[5] R. Govindan and D. P. Anderson, "Scheduling and IPC mechanisms for continuous media", in *13th ACM Symposium on Operating Systems Principles*, Pacific Grove, CA, pp. 68–80, Oct. 1991.

[6] P. Sénac, M. Diaz and P. Saqui-Sannes, "A formal environment for the specification and design of multimedia synchronization scenarios", in *presented at short Proc. Fourth International Workshop on Network and Operating System Support for Digital Audio and Video*, Lancaster, England, Nov. 1993.