Distributed Environment for Control and Management of Multimedia Data

Łukasz Czekierda, Mariusz Michalski,
Piotr Nawrocki, Krzysztof Zieliński
Dept. of Computer Science, University of Mining & Metallurgy
Al. Mickiewicza 30, Cracow, Poland
tel:+48 (12) 617 39 82, fax:+48 (12) 617 39 66
{luke, cmich, piter, kz}@cs.agh.edu.pl
http://www.cs.agh.edu.pl

October 8, 1999

1 Introduction

The multimedia applications still represent important area of system research which addresses increasing demand for expanded functionality, higher quality, portability and flexibility. These requirements have a novel aspects in the age of IT society and meet with common acceptance of new paradigms of distributed computer systems design and implementation built around distributed objects and thin clients dynamically loadable on demand. The distributed systems technology break-through appeared at great extent thanks to OMG efforts of CORBA standardization, and the spread of SUN-Soft Java. Both technologies despite theirs definition of general information processing architectures offer some support for multimedia. The OMG specification "Control and Management of Audio/Video Streams" (OMG A/V) [4, 5] defines a model for implementing an open distributed multimedia streaming framework while SUN-Soft supports JMF (Java Media Framework) [10] dedicated for multimedia GUI building. Neither of these proposals provides complete system support for implementation of distributed multimedia applications.

The existing multimedia frameworks like for example Continuous Media Framework [3] provide different models and paradigms of multimedia applications construction and employ more traditional RPC-based control mechanisms [2] or non-CORBA compliant techniques of stream flows establishment [13]. Examples of such applications may be found elsewhere as for instance, Internet-based streaming and session establishment mechanisms, such as SDR [12] used for popular MBone programs VIC or RAT [12].

This paper presents an open multimedia framework, called MMFramework, which may be used for dynamic creation of various multimedia applications and which could be extended by new multimedia devices. The proposed framework architecture consists of six layers. Its definition results from decomposition of the system into components with well-defined interfaces and internal implementation dedicated to the given hardware usage or applied policy of the system control and management. Each layer represents a collection of components which are characterized by similar functionality.

The system has been designed with existing CORBA Services and OMG specifications related to multimedia applications in mind. The multimedia streams control has been implemented based on an OMG document [5, 4, 8] using partly IONA's OrbixMX [15]. The system has been integrated using Name Service [16]. The multimedia devices and streams are characterized by a large number of parameters what justified the Property Service [19] usage.

MMFramework has been constructed taking into account the distributed system scalability. The conventional request/reply synchronous client–server paradigm has been replaced, where appropriate, with event-driven asynchronous communication [18]. The publish/subscribe patterns are widely applied with unicast and reliable multicast communication protocols when a device state or property changes have to be reported to a group of clients. This style of information dissemination and event notification has been implemented with the support of CORBA Event Service and Notification Services. Resulting MMFramework has been structured as collection of observable distributed objects what is the characteristic feature of the proposed architecture.

The novel aspect of MMFramework is the definition of mobile multimedia devices. The background of this study originates from location-aware computational systems such as Active Badge next generation (ABng) [1]. This system is a CORBA-compliant implementation of the Active Badge System developed at Olivetti & Oracle Research Laboratory (ORL) [9] allowing to situate people and equipment within a building determining the location of their ABng transmitters.

MMFramework has been also equipped with integrated graphical interfaces built in Java that represent in a compact, user-friendly form configuration, state and control of complex distributed systems. The system exploits Java Applets communicating via IIOP protocol with suitable CORBA servers of the framework. The graphical elements of these interfaces may by connected in run-time to call-back functions which generate suitable events or perform control activity. A lot of effort has been put into invention of a uniform graphical form of the system components' representation for the global system state visualization.

The structure of the paper is as follows. In section 2 the multilayer structure of the MMFramework has been presented. Section 3 concentrates on the Management and Access Control Layer functionality as the only layer of the defined framework directly visible by the user. Section 4 elaborates the example of GUI for dynamic construction of multimedia applications based on the functionality of the proposed architecture. In section 5 possible scenarios of using the system have been analyzed in detail. The paper ends with conclusions.

The paper addresses the requirements of modern multimedia networking application. The MMFramework has been proposed as a system which provides a suitable solution. MMFramework presents a novel approach to the multimedia applications design and implementation built around observable objects paradigm with event-based notification a principal communication mechanism. Based on the powerful OMG Control and Management A/V streams model, it unifies traditionally separate applications such as Video on demand, Videoconference and Live TV & video into one system. This is possible thanks to flexibility of the object oriented distributed programming provided by CORBA, which introduces clear conceptual model of the systems, mechanisms of dynamic binding and services. These features help to overcome inherent complexity of the multimedia systems control and management.

Although the basic functionality of the system has been reached, new challenging problems such as interoperability with the existing MBone audio/video tools, security in the system and control of concurrent access to the multimedia devices by a group of users, still have to be solved. Very interesting problems not addressed in this paper concern mechanisms of preferences negotiation between groups of users sharing common resources and definition of the system usage policy by the multimedia services providers. They open up new opportunities for research.

2 MMFramework architecture

The proposed decomposition of the MMFramework into layers is depicted in Fig. 1. The detachment of the three bottom layers is straightforward. These layers provide full access to the multimedia devices configuration and usage. The complexity of the second and third layer is covered by layers four and five. These layers construct a uniform view of the system providing an easy to use abstraction of system components seen from the top application layer.

The first layer called **MMHardware and System Software Layer** deals with multimedia hardware and software provided by vendors. This layer is represented by a wide spectrum of devices such as: video cameras, computers, audio/video encoders/compressors (such as our NPoint device), media servers (for example Sun MediaCenter server), etc. These devices are usually equipped with proprietary control software. From the user's point of view, this layer looks like an inconsistent collection of different devices.

The second layer — **MMHardware CORBA Server Layer** packs up the vendor-provided software with CORBA interfaces. This layer introduces a uniform abstraction of the service provided by a given device from the lower layer. This abstraction is defined by an interface specified in IDL and standard communication mechanisms provided by the IIOP protocol. The IDL interfaces defined in this layer support all operations provided by the native software. The main goal of introducing this layer is to establish a common base for the system development. The choice of CORBA distributed objects model was one of the most significant decisions for the MMF architecture.

The third layer — **A/V Streams Control Layer** is dedicated to multimedia streams creation, control, and destruction. This layer implements the OMG specification [4, 5] and provides software objects which expose functionality of the lower layer CORBA servers in a standard form most suitable for audio and video streams control. It provides an abstraction of a stream encapsulated in the form of a CORBA object which represents its parameters and control operations. This layer deals not only with the streams control activity but also provides mechanisms for streams parameters negotiation between source and destination multimedia devices and provides streams addressing and QoS control.

The fourth layer — **Presentation Layer** resolves the problem of different types of data used for the parameters of multimedia devices and streams repre-

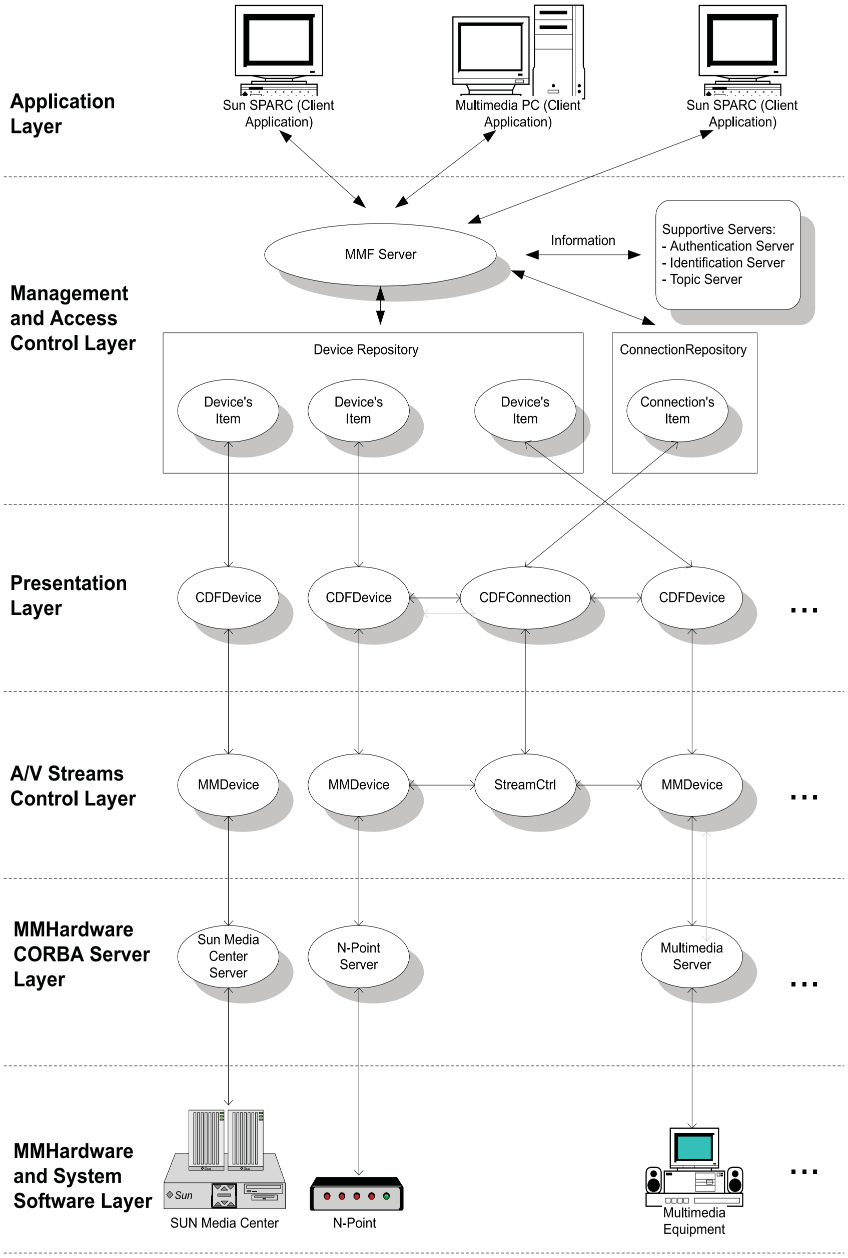

Figure 1: Multilayer MMFramework architecture

sentation. The main goal of this layer is to translate the parameters types from their actual values to especially constructed CDF (Common Data Format). This format is used above the Presentation Layer to simplify presentation of the system's state and to provide a uniform view of the system components for control and visualization purposes. This layer supports users with a powerful mechanism of forms what makes the system configuration simple and less susceptible to errors.

The **Management and Access Control Layer** provides a uniform view of the MMF components' state and a set of functions for manipulating and accessing them. The most important design decision about the construction of this layer is the choice of a model for representation of the system's state components. The most natural representation of the system is a repository of observable objects. They are objects with their own interfaces and notification mechanism. There are two types of items: devices and connections.

A top layer of the MMF architecture is called **Application Layer**. The entities of this layer are collections of user interfaces that provide access to control and visualization of the system state in the most convenient form. The objects defined on this level act as observers of the system components and combine them in the given application scenario. They may also perform the MMF clients' role actively changing the system's state by operations invocations on the devices and connections abstraction provided by the lower layer.

3 Management and Access Control Layer

The Management and Access Control Layer is the only layer visible by the user applications. This influences the tasks of the layer. It has to deliver a compact and uniform interface for end users, that offers access to the whole functionality of the system. The next its task is to provide methods of informing about the actual state of the system (see next sections).

Figure 1 illustrates the key components of the layer. It consists of two types of the repositories:

- Device Repository, which collects objects representing multimedia devices, and
- Connection Repository, containing objects which describe and control the state of multimedia streams connections,

and so called Supportive Servers.

An item from the Device Repository is variously comprehended. This may be (i) 'real' multimedia device, e.g. video server or hardware MPEG coder, (ii) multimedia workstation with its hardware and software enabling sending/receiving audio and video streams, (iii) *mobile device*, which is ascribed to a system's user and associated at the given moment with one of the workstations. Despite this diversity, the items are treated uniformly from the Management Layer's point of view.

Items from the Connection Repository represent the state of the connection between two or more devices as well as allow performing operations on the streams, like `start` or `stop`.

The main objective during the Management Layer's design was to create one logically centralized object, MMFServer, having knowledge about the whole system. That makes it possible to efficiently organize (i) an access to the system functions, (ii) monitoring its state, and (iii) making this state available to users in case of their synchronous invocations. The client never has direct access to the repository item. The only object the client has access to is MMFServer. It is necessary to point out that notification information is sent to observers by each of repository items separately. The notification information is passed directly to the observers in the Application Layer, hence the MMFServer does not represent the centralized point of the system what could made a bottleneck.

As has been already mentioned, there is a number of so-called supportive servers also considered as a part of the Management and Access Control Layer. They are designed in such a way as to give an external support for items' management and creation and to provide to the Connection and Device Repositories some external services described below.

One of supportive servers is an Identification Server. Items are externally identified by theirs identifiers. The id parameter is translated by the Identification Server to the item's reference which is used for further operation processing.

External support is also needed for the access control function which is performed by an Authentication Server. Operations requiring authentication have an extra parameter — a user's key. The key is used by the Authentication Server to check user's permission to perform a given operation. Thus the repository items don't need to have knowledge about the users and their rights.

As can be seen, it is a role of the MMFServer to contact the Authentication Server and the Identification Server and then invoke an operation on the repository item.

The third supportive server — Topic Server is to handle topics used by OrbixTalk [18] — IONA's implementation of OMG's Event Service. These attributes concern the control of notification event channels' space. A number of such channels can be associated with each item. Using the channels the receivers can be precisely informed about actual state of the system taking advantage of the fact that each topic represents some aspect of the item's state. The used *push* model of information dissemination allows the receiver to implement the response to the notification message in the suitable way.

4 MMFramework client

This section describes the architecture of the MMFramework client. A graphic interface of client's program consists of two main types of components designed for it. The first component, called **NetChart**, represents a topology of the multimedia system i.e. devices and connections between them. The next components, called **Form**s, are used for presenting parameters and states of multimedia devices and connections, as it is shown in Fig. 2. For the communication of the client with the MaACL layer the programming environments such as OrbixWeb [17] and OrbixTalk [18] are used.

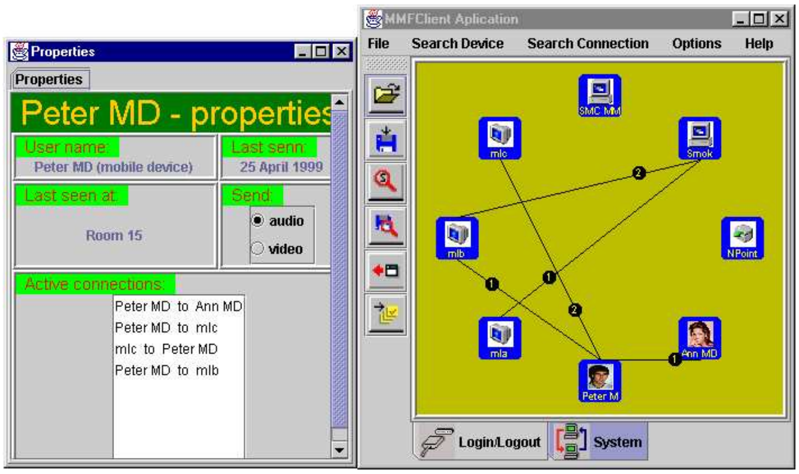

Figure 2: NetChart component and Form component associated with Peter MD device

4.1 NetChart component

The main goal while creating the NetChart component was to achieve a compact graphic representation of the whole system, devices and connections what provides access to their state and control functions. Thus, it allows users for a fast orientation in the system's state, making a control decisions and managing the system. Multimedia devices in this component are represented by rectangles placed on a circle as it is shown in Fig. 2. An appropriate icon and description of the device are placed inside each rectangle. Connections between devices are represented by lines drawn from one rectangle to another. Each line may indicate many connections between the same devices, therefore a label with the number of connections is plotted on each of them. This component also provides many options designed for customizing its appearance according to individual preferences. To each device and connection in the component is assigned a dedicated manager. The manager notifies them about the information changes.

4.2 Form component

The next component is a dynamically built GUI interface. It consists of simpler components called **Elem**s, which are automatically laid out on the Form in the way best fitting its size. Each elem possesses the main component which is one of six graphic Swing components such as JList, JComboBox, JLabel, JTextField, JCheckBox group and JRadioButton group [21]. These Swing components are sufficient for good representation of all possible parameters of devices and connections. Each elem has additional title and border for good appearance and also possesses its manager which delivers events connected with the elem.

The Forms are used for presentation of the parameters of devices and connections or can be edited to introduce new values of parameters. These values are used for setting devices and connections parameters. The example of graphical representation of the Form is shown in Fig. 2.

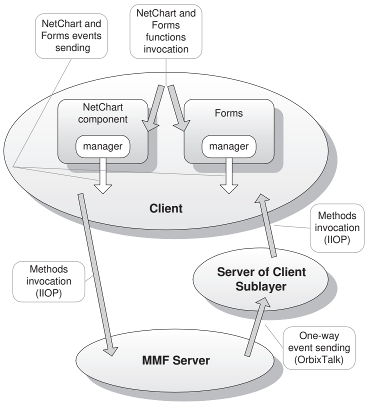

Figure 3: The communication mechanism

4.3 Client communication mechanism

A communication from the client to the MMF Server is done using the IIOP protocol. General assumption is that the items from the MaACL layer communicate with the client using one-way multicast message propagation mechanism used by OrbixTalk. A communication from the server to the client is more sophisticated because there is no an appropriate OrbixTalk implementation for the Java platform. Therefore the client program has a sublayer in the form of a server written in C++ language, which receives multicast messages from the MMF Server and propagates them to the client using IIOP protocol. This is schematically presented in Fig. 3.

5 MMFramework Application Scenarios

The aim of this section is to present the area of applications possible to realize using the MMFramework and available equipment. The constructed multimedia system is built universally, so it is possible to use different devices, including these which will be accessible in the future. Both user control application and management in lower layers of the architecture won't need any changes.

The multimedia system is built basis of the given hardware and software components what influences the possible applications. All the multimedia devices' functionality has been wrapped by CORBA servers realizing functions resulting from the architecture's needs. These hardware devices are:

Sun MediaCenter server, a combination of standard server hardware and special software that is optimised for and dedicated to the storage and delivery of video streams in the MPEG-1 format. The software consists of two RPC-based APIs: Content Manager (CM) allowing to store a multimedia content on the server and Media Stream Manager (MSM) allowing to play streams from SMC.

Multimedia workstations Sun-Ultra equipped with camera, microphone and speaker device, a software controling them and a player application visualising video streams.

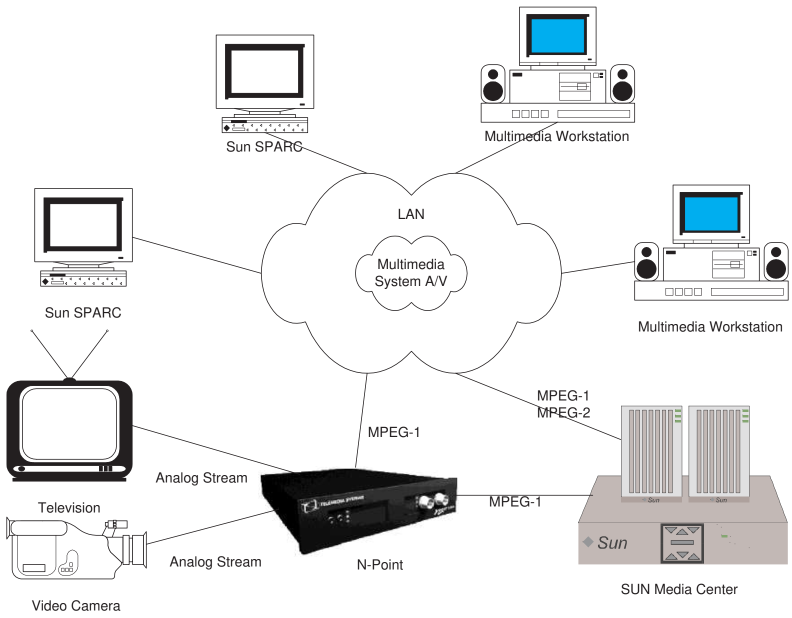

Figure 4: MMFramework Application Scenarios - Video on Demand and Live TV & video

NPoint device allowing to digitize in real time an analog stream taken from camera or TV. The data is then distributed to the network in the MPEG-1 format.

The novel aspect of MMFramework is the definition of *mobile multimedia devices*. ABng system integrated with the multimedia system allows to locate people and equipment within a building determining the location of their Active Badges. These small devices worn by personnel and attached to equipment periodically transmit infra-red signals detected by sensors which are installed in the building. Hence, source or sink of video or audio stream may be associated logically to a locatable user and follow him.

Even in such a small multimedia system as existing in our DCS there are some interesting scenarios possible to implement and exploiting the full potential of the system.

The MMFramework usage will be illustrated by a set of typical multimedia applications. In either example a connection is established with particular set of parameters, e.g. protocol for transferring data, transport addresses and QoS parameters defining properties of multimedia stream. After connection is realised there is some standard operations possible to perform on the stream: start, stop, destroy, modifyQoS whose semantics seems to be clear enough.

The following scenarios have been implemented:

5.1 Video on Demand

The investigated scenario is shown in Fig. 4. This application requires the following devices:

- Sun MediaCenter video server,
- one or more multimedia workstations receiving multimedia stream.

The user setting a connection can choose any of items stored on the video server. The hardware is so flexible that other users' activity practically never doesn't influence the request. Depending on the number of receivers, an unicast or multicast stream is set.

5.2 Live TV & video

This scenario (also shown in Fig. 4) requires the following devices:

- NPoint device,
- one or more multimedia workstations receiving multimedia stream,
- (optionally) Sun MediaCenter server.

The example is quite similar to the above one. The only difference is that the source of data is the NPoint device. Alternatively one can realize the scenario in a different way – taking advantage from the *playthrough* property of Sun MediaCenter (giving the possibility to store and transmit multimedia stream at the same time). Sun MediaCenter is then the only consumer of the unicast stream and the producer of the multicast or unicast stream.

5.3 Videoconference

This application requires only the multimedia workstations being capable of sending and receiving audio and video data streams. There is a possibility to set both a full duplex point-to-point and a simplex point-to-multipoint connections. One of the problems which needs still to be solved is the mixing of audio streams. This is why it is not possible for us at the moment to have the full duplex multipoint-to-multipoint connections.

5.4 Mobility facility

In each of the above examples a very interesting modification could be introduced, since the connection could be associated with the live user instead of the workstation. Initially, the connection will be established with the workstation which is the nearest to the chosen person. The person's movements results with the redirection of the attached multimedia stream to the another nearest workstation. The explanation of this "follow me" effect is presented in Fig. 5.

6 Evaluation of the system status

All layers of the architecture have been designed and successfully implemented. Of course, the system still needs further refinement, testing and validation; its basic functionality has been reached, however.

Even in such a small multimedia system as the one existing in the DCS at the UMM, it was purposeful to enrich its functionality with a scalable and

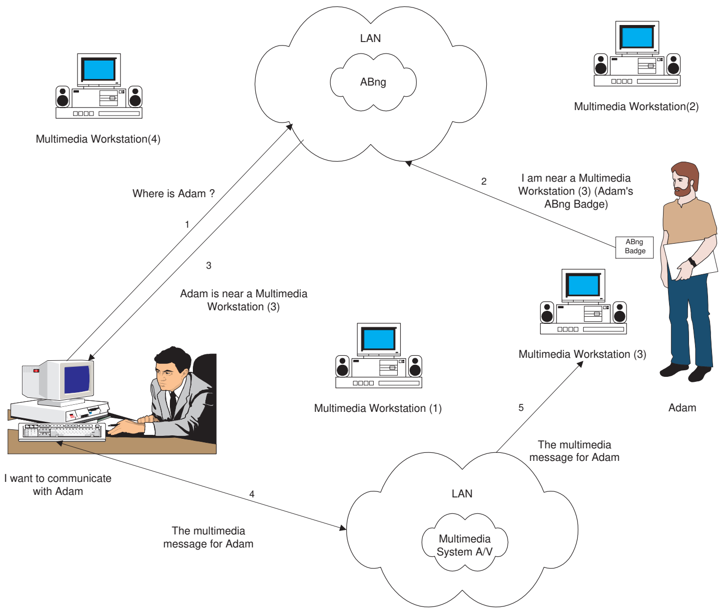

Figure 5: Videoconference scenario

efficient mechanism of notification about changes in the system's state allowing at the same time to monitor and archive events in the system.

One of the most important advantages of the MMFramework architecture is the implementation of the worked out CDF format. This powerful mechanism makes it possible to dynamically build applications. Taking advantage of CORBA architecture it has been possible to integrate different systems into one working framework.

7 Conclusions

MMFramework presents a novel approach to the multimedia applications design and implementation built around observable objects paradigm with event-based notification a principal communication mechanism. Based on the powerful OMG Control and Management A/V streams model, it unifies traditionally separate applications such as Video on demand, Videoconference and Live TV & video into one easy configurable system. This is possible thanks to flexibility of the object oriented distributed programming provided by CORBA, which introduces clear conceptual model of the systems, mechanisms of dynamic binding and services. These features help to overcome inherent complexity of the multimedia systems control and management.

References

[1] *Active Badge System*, The Olivetti & Oracle Research Laboratory, 1993

[2] *ARM Architecture Overview*, `http://www.arm.com/Architecture`

[3] Continuous Media Toolkit, `http://bmrc.berkeley.edu/projects/cmt`

[4] Object Management Group, *Control and Management of A/V Streams*, OMG Document telecom/97-05-07 ed., October 1997.

[5] Object Management Group, *Control and Management of A/V Streams, Revised Submission*, OMG Document telecom/98-10-05 ed., October 1998.

[6] Object Management Group, *CORBA Components*, Joint Initial Submission by IONA Technologies et al., 1997, November, OMG Report.

[7] T. J.Mowbray and R. C.Malveau, *CORBA Design Patterns*, Willey, 1997.

[8] S. Mingee, N. Surendran, D. Schmidt, *The Design and Performance of a CORBA Audio/Video Streaming Service*, Washington Univ. TR#WUCS-98-15.

[9] A. Harter, A. Hopper, *A Distributed Location System for the Active Office*, IEEE Network, Special Issue on Distributed Systems for Telecommunications, Vol. 8, No. 1, January, 1994

[10] SUN Microsystems, *Java Media Framework*, `http://www.sun.com/nab/java.framework.html`

[11] Czekierda Ł. Michalski M., Nawrocki P., Zieliński K., *MMFramework – Distributed Environment for Control and Management of Multimedia Data*, September 1999

[12] *MBone Conferencing Applications*, `http://www-mice.cs.ucl.ac.uk/multimedia/software/`

[13] *Multiparty Multimedia Session Control*, `http://www.ietf.org/`

[14] *Notification Service*, Joint Revised Submission by NEC System Laboratory et al., 1997.

[15] Iona Technologies Ltd. *Orbix MX, A Distributed Framework for Telecommunication Service Development and Deployment, White Paper*, IONA Technologies PLC., April 1998.

[16] Iona Technologies Ltd., *OrbixNames Programmers Guide* `http://www-usa.iona.com/support/docs/orbixnames/prog/guide.pdf`

[17] Iona Technologies Ltd., *OrbixWeb Programmer's Guide* `http://www.iona.com`

[18] Iona Technologies Ltd., *OrbixTalk White Paper*, April 1996, `http://www.iona.com`

[19] *CORBA services: Property Service*, `ftp://ftp.omg.org/pub/docs/formal/97-12-20.pdf`

[20] *Stream-oriented Information Flow based on CORBA Services in Large Distributed Systems*, ESPRIT Project 20288-CRIT-2, March 1997

[21] Sun Microsystems *Swing-1.0.1*, `http://www.javasoft.com/`