

Java Multimedia Studio v1.0

Giancarlo Fortino*
fortino@icsi.berkeley.edu

International Computer Science Institute, Berkeley, CA

TR-97-043

November 1997

Abstract

Along with the emergence of a new generation of multimedia applications has come a need to facilitate real-virtual teleconferences and automatic generation of contents. In this direction Java Multimedia Studio, a tool completely java-based allowing to edit, record and playback multimedia sessions over the Internet Mbone, has been developed. Java Multimedia Studio is founded on a QoS centered Java and Actor-based Framework providing the management of local and distributed synchronization of streams by mixing, translating and filtering RTP packets. It not only enhances on-line and enables off-line multimedia conferencing but also gives a much more challenging opportunity to create multimedia sessions enriching their contents.

* The author was sponsored by the International Institute for Advanced Scientific Studies "Eduardo R. Caianiello", Vietri sul Mare (SA), Italy.

1. Introduction

The Internet MBone (Multicast Backbone) is being deployed over the Global Internet as part of real production network and many sites around the world are connected and many others are preparing to do it. MBone has evolved over the time and now is a stable platform for world wide delivery of seminars, conferences, working group meetings, virtual worlds and even games. A lot of stand-alone and distributed multimedia tools (e.g., vic [1], vat, wb, MBone VCR [2], Mvote, MVoD [3], mMOD [4]) exist enabling the users to handle several kinds of multimedia sessions consisting of video, audio, text, and graphical images. In the scenario of real-virtual conferencing and automatic content generation, Java Multimedia Studio enables the multimedia users to edit, record, playback, browse and filter multimedia sessions. It is completely written in Java to guarantee multi-platform portability. Java Multimedia Studio is founded on a QoS centered Java and Actor-based Framework (JAF) for real-virtual teleconferencing [5] providing the management of local and distributed synchronization of streams by the mixing, translating and filtering of RTP packets. JAF is a variant of the Actor model, which rests on timing predictability, customizable scheduling and a modular specification of QoS constraints [5,6,7,17]. A multimedia application is modeled as a collection of autonomous, distributed, concurrent and mobile multimedia and inter-media actors interacting one to another to achieve their common goal, e.g., real-virtual multimedia conferencing sessions [8], automatic contents generation or video-on demand services [9]. The tool handles multimedia sessions under the form of RTP packets [10,11]. All the recorded multimedia sessions constitute the basic content background. Java Multimedia Studio can operate on this background by using the concept of added content layer. A layer is a sort of index built both on the background and on another layer. From a user point of view, the layer is a multimedia session that can be handled (e.g., played-back, edited). The background and layer concepts enable to create multimedia sessions by using pre-recorded and live material. As an example, a part of the background consists of a multimedia session regarding to an audio/video teleteaching session in English language along with another session regarding to the translation of the same audio in Italian language. The layer could be a “new” teleteaching session in Italian language. The 2nd section describes the design

issues and the architecture of the multimedia toolkit. The 3rd section shows and comments some of the most important GUI features. Finally, conclusions are presented which clarify the overall status of the project and furnish an indication of some directions of further work.

2. Design issues and Architecture

Java Multimedia Studio is completely object-oriented and founded on Java and Actor-based Framework [5]. It is independent from tools originating the multimedia sessions under the form of RTP streams. The presentation of video and audio might be done by whatever tool understands RTP protocol (e.g., VIC/VAT, IP/TV, ICast Broadcaster, and so forth). It follows the characteristics of a typical MBone Session [2] consisting of a number of multicast channels on which a media stream (e.g., audio, video, text) might be present. The design is favored by the inherent simplicity of Java programming which allows to take great advantage of the object-oriented paradigm. In fact, in this context choosing Java as the object-oriented programming language and environment ensures the toolkit multi-platform portability, heterogeneous environment interoperability and natural extensibility.

The main characteristics are:

- support record/playback of multimedia sessions;
- support edit of multimedia sessions in accord to the Background/Layers concept;
- provide local and distributed synchronization of multiple media;
- provide navigation within sessions by using Java Session Directory;
- provide implicit and explicit indexing;
- GUI Java-like.

Java Multimedia Studio consists of several components (see fig.1): Player, Recorder, Editor, Session Directory, Filter. It also has built-in several protocols: RTP/RTCP, SAP, SDP.

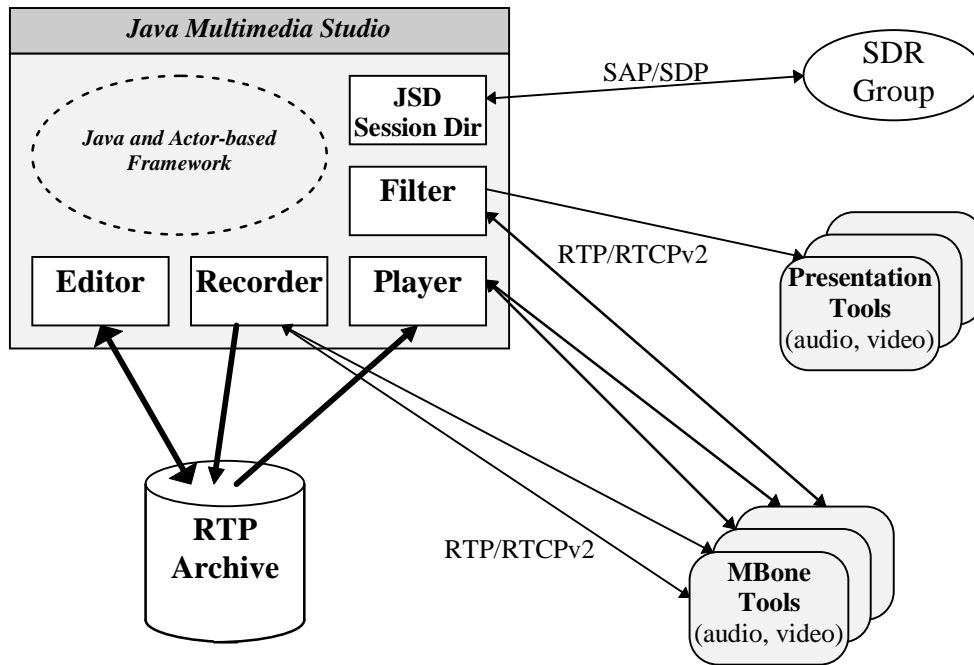


Figure 1: Java Multimedia Studio's Architecture.

2.1 Recorder

The recorder records IP-packets on RTP-level. Recording on RTP-level means that the VCR parses the RTP-header of each incoming packet and checks for duplicates and out-of-order packets. It can record data arriving both in unicast and in multicast way. Each recorded data-stream is stored in two files, a data-file and an index-file. The data-file contains the raw RTP packets as in the original session. The index-file provides indexing on the data-file to make random and fast access to the data. A recording can also include an optional third file, an option-file. This file contains markers provided by the user for book-marking purposes (e.g., jumping to certain points in the session, segmentation of the single media stream).

2.2 Player

The player plays RTP packets back in accord with their timestamps. The packets are scheduled according the following scheme: let T_a and T_b be the timestamps of two

contiguous packets of a same stream where a is the latest sent message. The schedule time T (e.g., in msec) of the next b message is computed as:

$$T = abs(T_a - T_b) * (1000 / ClockFreq) ,$$

where ClockFreq is a parameter of the adopted payload (e.g., H261, JPEG, PCMU, GSM). The temporal basis to schedule the delivery of the packets is provided by the JAF mechanisms.

2.3 Editor

The Editor allows to create new multimedia sessions by using the pre-recorded existing ones accordingly to the background/layers concept (see fig. 2). All the recorded multimedia sessions constitute the basic content background. Java Multimedia Studio can operate on this background by using the concept of added content layer. A layer is a sort of index built both on the background and on another layer. The editor [12] makes possible to graphically specify the points of synchronization or relations among the media streams constituting the multimedia session (layer) which is being edited.

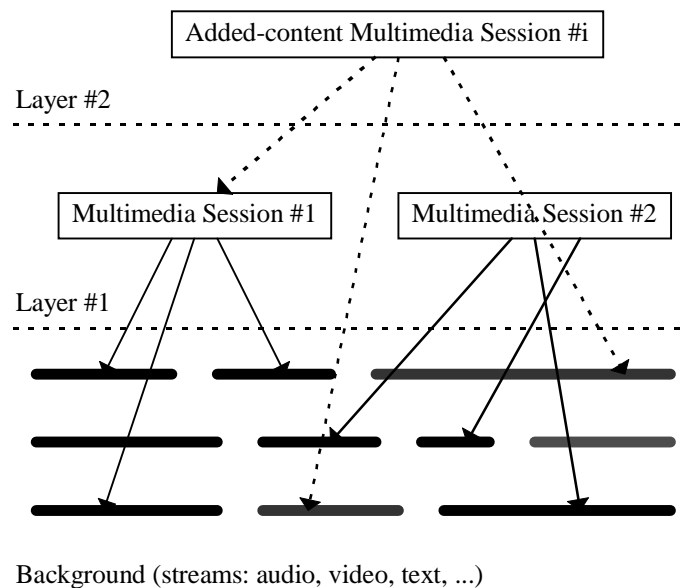


Figure 2: Background/Layers concept.

2.4 Java Session Directory

The Java Session Directory implements a SDR (Mbone Session directory tool) using the Session Announcement Protocol [13] and the Session Description Protocol [14] to listen to and announce multimedia sessions over Mbone.

2.5 Filter

The Filter implements a QoS Filter (see fig. 3) based on JAF enabling the filtering and translating of RTP packets. It allows the synchronization of media streams by deleting the jitter delay per stream basis and keeping the skew between audio and video under-control within a negotiable range. In the hypothesis of ignoring packet losses, the Filter can verify JITTER and SKEW (EED could be indirectly estimated by using RTCP report). On the basis of the arrival time of latest packets, stored into local variables, the observed arrival time of the current received packet, the available information on the JAF control channel, and the technique of buffer-delay adaptation [18], the Filter can calculate JITTER and SKEW values and apply a suitable policy to synchronize the media streams.

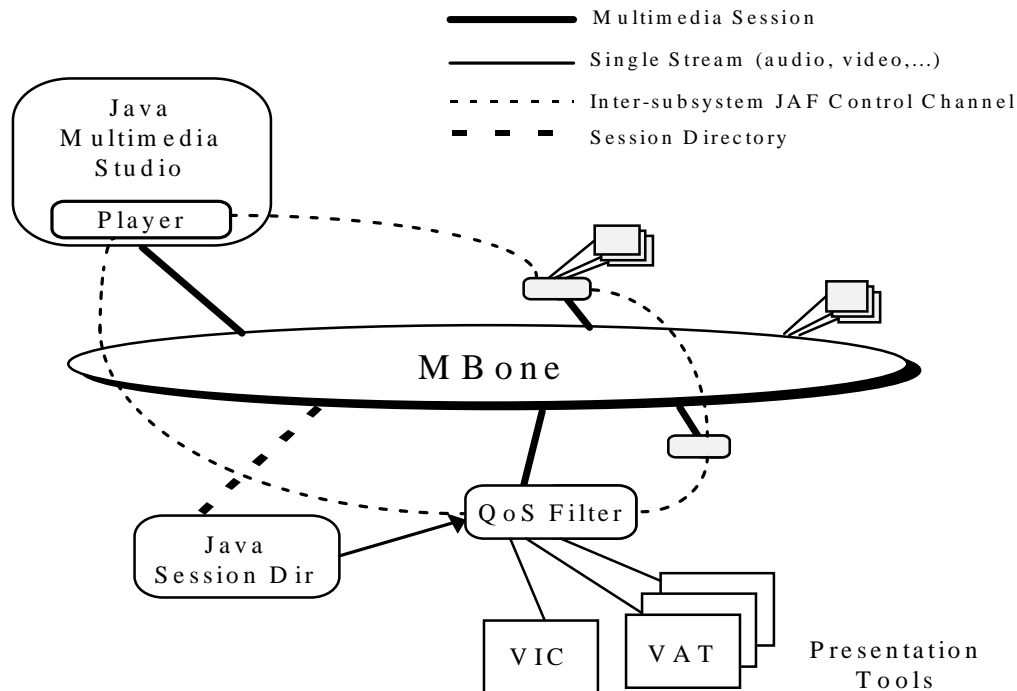


Figure 3: Distributed configuration of the QoS Filter.

An application which the JAF Filter is being tested in is the Lip Synchronization. The transmission of multimedia data over packet networks and the use of independent presentation media (audio, video), usually causes the receivers to experience a time lag between a remote speaker's speech and the associated lip movements. Subjective studies have shown that the two streams do not have to be exactly matched, but that a skew of 80-100 ms is below the limit of human perception. A desired lip synch skew should be ≤ 80 ms. Between 80 ms and 160ms the skew was found acceptable. However, beyond 160 ms skew is perceived as ANNOYING. By using the filter, it is possible to provide, according to certain QoS requirements, lip synchronization artificially delaying either audio or video output or both, so that words and sounds are matched in time. The scheme implemented is described in figure 4.

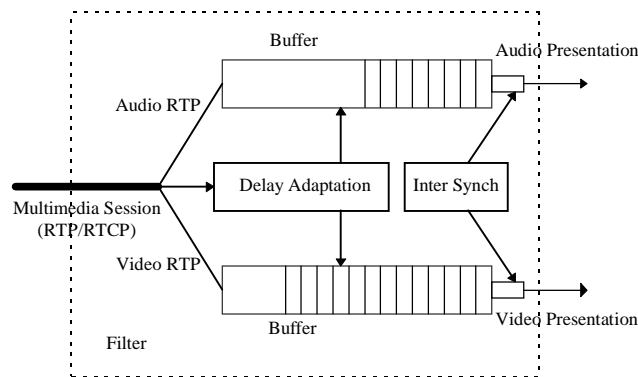


Figure 4: Filter synchronization scheme.

The filter plays the role of a synchronization filter/agent (fig. 5). It filters the multimedia session and handles the streams among the presentation tools (audio, video).

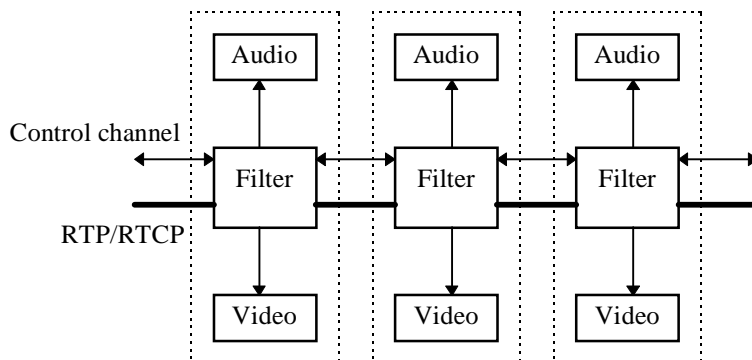


Figure 5: Synchronization Filter/Agent.

3. GUI description

The Java Multimedia Studio offers a graphical user interface user-friendly and organized in a main window with selectable menu items calling dialog boxes. As it is the version 1.0 α , the next versions might be different in some detail. The main window (see fig. 6) displays five menu: Session to create, open, list multimedia sessions and launch the SD; Option to set-up parameters such as presentation tools to use; Editor to run the Editor; VCR to enable the Player and Recorder; Help to have information about the toolkit.

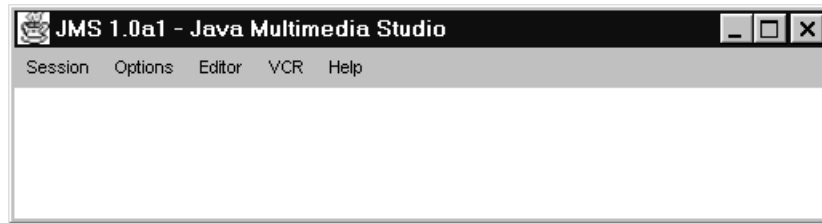


Figure 6: Main window of Java Multimedia Studio

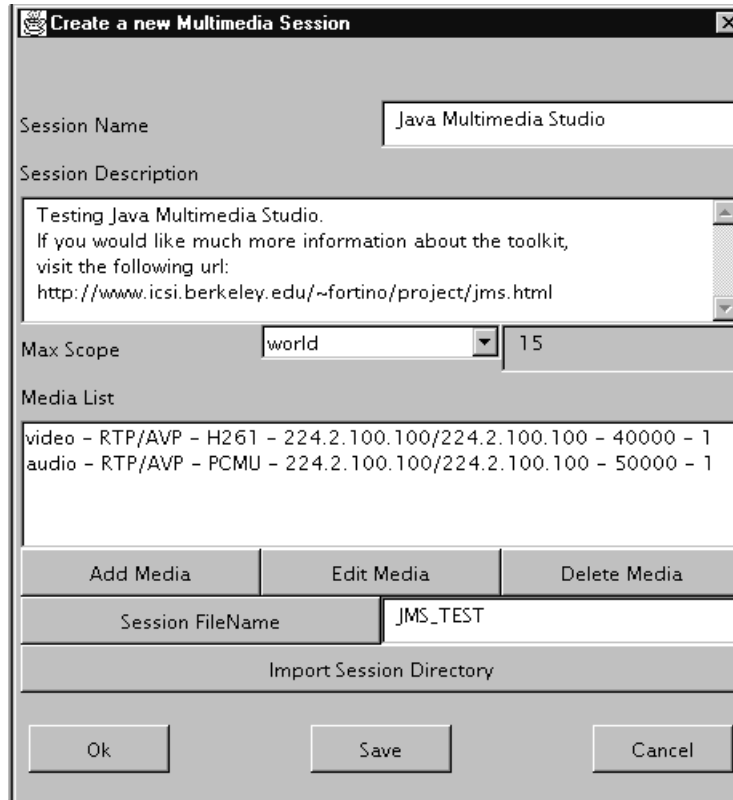


Figure 7: Creation of a Multimedia Session.

The creation of a multimedia session for recording and playing-back purposes is enabled by pressing the menu item New Session in the menu Session. A dialog box (see fig. 7) is displayed and allows to insert all the parameters characterizing a multimedia session such as Title, Description, Max scope, Media list, Filename of the session. The Media List displays all the media composing the multimedia session. The Add, Edit and Delete buttons respectively enable to add a media to the media list, edit and delete already inserted ones. By the Import Session Directory button a dialog listing all the multimedia sessions currently present over MBone is displayed. It is possible to select a session choosing to add either the whole multimedia session or only some media to the multimedia session is being created. The figure 8 shows the Add Dialog Box which allows to insert a new media in the media list. The parameters which can be set-up are: media type, media encoding, address (multicast or unicast), port, protocol and scope.

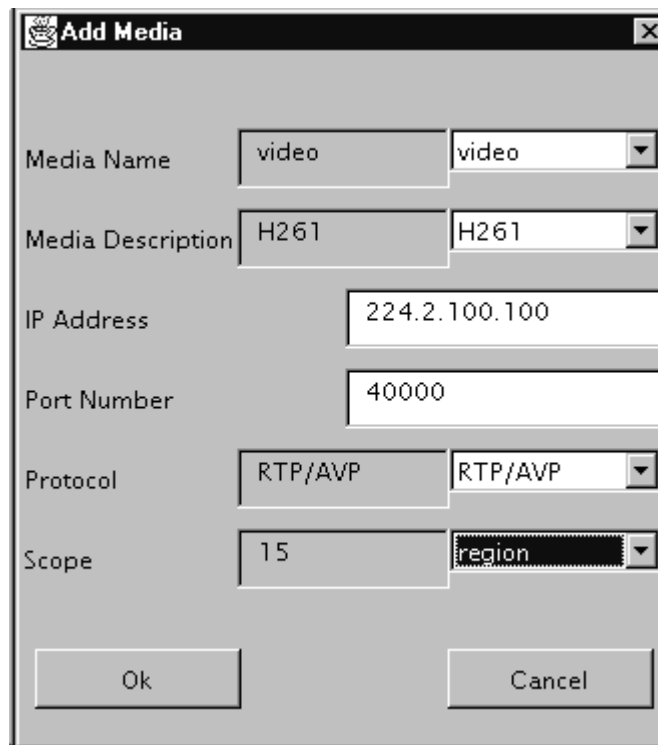


Figure 8: Add Dialog Box.

By pressing the menu item Set Playing-back Session in the menu VCR, the Player (see fig. 9) is enabled. It allows to select a multimedia session to be played-back and to launch a RTP pump thread. More than one multimedia session can be played-back at the same time.

The commands Start, Stop and Pause facilitate the user to control the play-back. The commands refer to the currently selected multimedia session in the list. The horizontal scroll bar serves to move on the multimedia session to provide with random timing access. The Total time displays the time of the recorded session and the Elapsed time reports the current time of the playback.

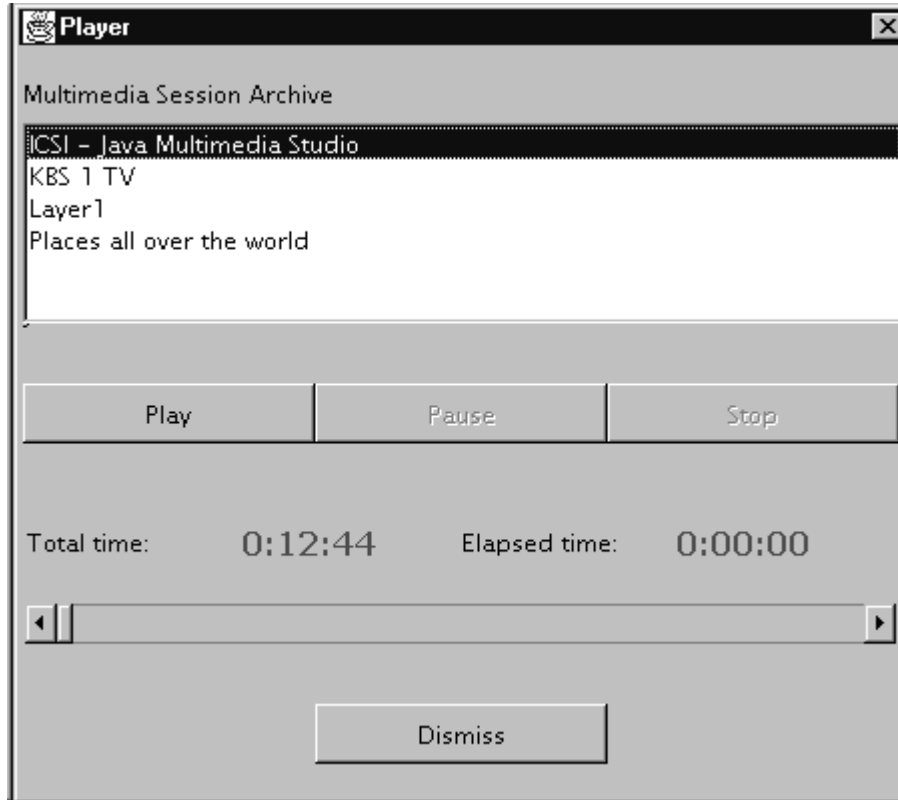


Figure 9: Player.

The Java Session Directory (see fig. 10) displays all the multimedia sessions currently present over MBone. By pressing the Start session button, the selected MBone session is joined and, accordingly to media type of the session media, the correct and set-up presentation tools (e.g., video, audio) are launched. By pressing the Start media Button, only the media selected in the session media list is joined. The Start Filter button enables the JAF Filter. JAF Filter can start only if the session is JAF controlled in accord to the schema in figure 3. The Stop Filter button stops the filtering of the RTP packets.

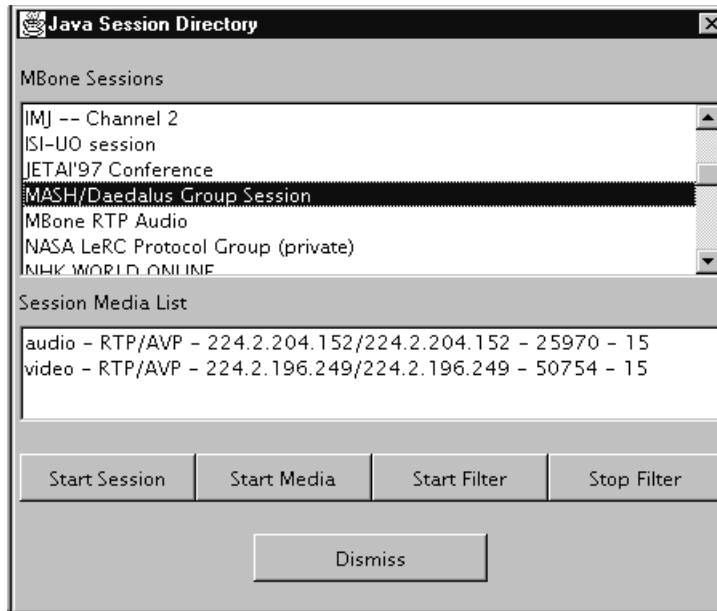


Figure 10: Java Session Directory and JAF Filter.

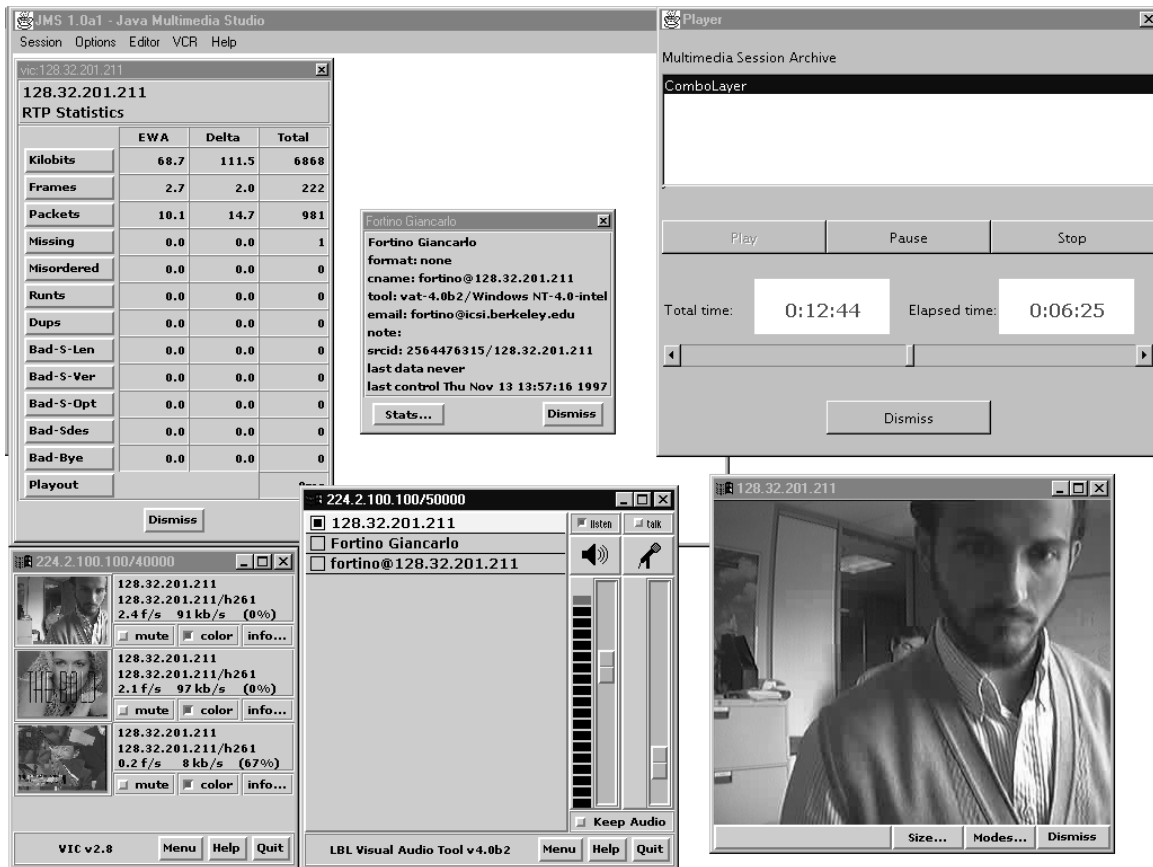


Figure 11: Java Multimedia Studio “in action” during a playback of a multimedia session.

Acknowledgments

The author wishes to thank Libero Nigro with whom is involved in the JAF project at the University of Calabria, Andres Albanese (Network Group Leader at the ICSI) for the helpful discussions, advise and contributions to the Java Multimedia Studio project.

Conclusions

This paper proposes Java Multimedia Studio founded on Java and Actor based Framework that provides a minimal and light-weight architecture which supports a modular construction of multimedia systems. JAF is based on the separation of concerns and clearly separates the specifications of functional from timing and more in general QoS constraints. All of this improves modularity and reusability. The toolkit consists of several components: Player, Recorder, Editor, Session Directory, Filter. A first implementation of the JAF Filter is under test. Prosecution of the research activity aims at extending the toolkit with a multicast service to restore “a posteriori” recorded multimedia sessions by using mobile agents [15], an implementation of the RTSP protocol and a program scheduler agent to automatically record and playback multimedia sessions. In addition the next version is being currently designed towards a collaborative media on demand (CMoD). It will be completely distributed by integrating MBone and WEB technologies [4,16].

References

- [1] McCanne S., Jacobson V., “vic: a flexible framework for packet video”, Proceedings of ACM Multimedia '95, ACM, pp. 511-522, Oct. 1995.
- [2] Holfelder W., “MBone VCR - Video Conference on the MBone”, Proc. Of ACM Multimedia '95, ACM, 237-238, 1995.
- [3] Holfelder W., “Interactive remote recording and playback of multicast videoconferences”, IDMS'97, 1997.
- [4] Parnes P., & others, “mMOD: the multicast Media-on-Demand system”,

<http://www.cdt.luth.se/~peppar/progs/mMOD/>.

- [5] Fortino G., Nigro L., "QoS centred Java and actor based framework for real/virtual teleconferences", accepted for SCS EuroMedia 98 (MEDIATEC), Leicester UK, Jan 5-7, 1998.
- [6] Kirk B., Nigro L., Pupo F., "Using real time constraints for modularisation", Springer-Verlag, LNCS 1204, 236-251, 1997.
- [7] Ren S., Venkatasubramanian N., Agha G., "Formalising multimedia QoS constraints using actors", Formal Methods for Open Object-based Distributed Systems, Vol. 2, Bowman H. And Derrik J. (eds), Chapman & Hall, 139-153, 1997.
- [8] Steinmetz R., Nahrstedt K., "Multimedia: computing, communications and applications", Prentice-Hall, 1995.
- [9] Rowe L. A., Berger D. A., and Baldeschwieler J. E., "The Berkeley Distributed Video-on-Demand System", Multimedia Computing, Proceedings of the Sixth NEC Research Symposium, Ed. T. Ishiguro, SIAM, 1995.
- [10] Schulzrinne H., "RTP Profile for Audio and Video Conferences with Minimal Control", IETF RFC1890, 01/25/1996.
- [11] Schulzrinne H., Casner S., Frederick R., Jacobson V., "RTP: a Transport Protocol for Real-Time Applications", IETF RFC1889, January 1996.
- [12] Pazandak P., Srivastava J., "Multi-User Interactive Presentation Environments on the Internet: An Overview of DAMSEL and It's Implementation", IEEE Conference On Multimedia Computing and Systems (IEEE MMS '96) June 1996, Hiroshima, Japan.
- [13] Handley M., "SAP: Session Announcement Protocol", Internet Draft, IETF, Multiparty Multimedia Session Control Working Group, 09/09/1997.
- [14] Handley M., Jacobson V., "SDP: Session Description Protocol", Internet Draft, IETF, Multiparty Multimedia Session Control Working Group, 09/09/1997.
- [15] Fortino G., Grimaldi D., Nigro L., "Multicast control of Mobile Measurement Systems based on JAF", *Submitted*, 1998.
- [16] Fortino G., Grimaldi D., Nigro L., "Distributed measurement patterns using Java and web tools", Proc. of IEEE AutoTestCon97, Anaheim (CA), 624-628, 1997.

- [17] Nigro L., Pupo F., “A modular approach to real-time programming using actors and Java”, Proc. Of 22nd IFAC/IFIP Workshop on Real-Time Programming, (to be published by Elsevier), Sep. 15-17, Lyon (Fr), 83-88, 1997.
- [18] Jacobson V., “Multimedia Conferencing on the Internet”, SIGCOMM '94 Tutorial, University College of London, England, 30 August 1994.

Acronyms

[CMoD]	Collaborative Media on Demand
[EED]	End to End Delay
[GUI]	Graphical User Interface
[JAF]	Java and Actor-based Framework
[JSD]	Java Session Directory
[MBone]	The Internet Multicast Backbone
[mMOD]	multicast Media-on-Demand
[MVoD]	MBone Video conference recording on Demand
[QoS]	Quality of Service
[RTCP]	Real-Time Control Protocol
[RTP]	Real-Time Protocol
[RTSP]	Real-Time Streaming Protocol
[SAP]	Session Announcement Protocol
[SDP]	Session Description Protocol
[SDR]	Session Directory