# JMFMoD: A New System for Media on Demand Presentations on Education


## Authors:

Ángela Belda, Universidad Politécnica de Valencia, Camino de Vera s/n, Valencia, Spain, anbelor@doctor.upv.es
Juan José Cermeño, Universidad Politécnica de Valencia, Camino de Vera s/n, Valencia, Spain, juacerpo@teleco.upv.es
Juan Carlos Guerri, Universidad Politécnica de Valencia, Camino de Vera s/n, Valencia, Spain, jcguerri@dcom.upv.es
Ana Pajares, Universidad Politécnica de Valencia, Camino de Vera s/n, Valencia, Spain, apajares@dcom.upv.es
Carlos Palau, Universidad Politécnica de Valencia, Camino de Vera s/n, Valencia, Spain, cpalau@dcom.upv.es



***Abstract*** — *This paper describes the main challenges of implementing a Java Media on Demand (MoD) system. The objective of this system is to offer a complete solution for creating multimedia presentations (audio, video, text, slides) and their synchronized reproduction on the client side through Internet. This system consists of the following elements: an authoring presentation tool, a stand-alone MoD server application, a web integrated MoD client, a database containing all the available media presentations, and a proprietary protocol that provides all the functionality of a VCR stream control mechanism. At present, the system is been using to complement lectures in the Technical University of Valencia.*



***Index Terms*** — *MPEG-4, multimedia,real-time system, synchronization, system on demand.*


## INTRODUCTION

Among the different multimedia systems currently in use, *Video on Demand systems* (VoD) cover a wide range of applications. However, in most of these applications, the transmitted information is a synchronized combination of a video-only flow and an audio-only flow. In contrast to these traditional systems, the system (JMF Media on Demand, JMFMoD) presented in this paper allows *on-demand transmission and synchronized rendering of multiple media: audio, video, text and slides*, without limiting the number of flows of each type. This system has been developed in Java making use of the *JMF 2.1.1* Sun libraries. It offers an end-to-end solution, starting with the creation of multimedia presentations and allowing their on-demand transmission with RTP. The session is controlled through a proprietary protocol which works over TCP. Tele-teaching, tele-marketing, help-on-demand systems and tutorials could become typical applications of the developed system.

Some works and references related with the main aspects of this paper are analysed below. The issues are mainly: on-demand based diffusion systems; temporal synchronization specification; available libraries for the development of multimedia systems; and suitable protocols for real-time applications.

Most *on-demand* applications like RealNetwork's RealSystem, Nullsoft's SHOUTcast, Windows Media, Apple's Quicktime, or Cisco's IPTV [11], [10], [5] implement systems that are only based on Video-on-Demand (VoD). VoD systems allow clients to select movies from a large menu, and play them back on their own stations. The main problems a VoD system has to deal with are: storage space, bandwidth, synchronization, transport protocols, playback control protocols, and scalability. Among on-demand systems, we can distinguish between near-on-demand (NoD) and true-on-demand (ToD) systems. The former makes use of multicast technologies to enable multiple users to share a single channel and reduce system cost. ToD systems allocate a dedicated channel for every user in order to minimize the latency and allow interactive operations. The system presented in this work is a ToD system and uses a previously developed VoD system, the JMFVoD [1], as starting point.

With regard to the temporal specification of multimedia presentations, there are many works [7], [6], [2], [3] that deal with intra and inter-frame synchronization and propose various solutions: interval-based method, axe synchronization, control flow, timed petri nets, event-based synchronization or scripts. Our proposal is based on the use of a temporal axe (axe synchronization) with all the inherent advantages of this method: easy compression and the capacity for integrating time independent objects as text and slides.

Among the different alternatives to implement our MoD system, we decided to use Java for its current importance in web-based multimedia applications and the availability of Java Media Framework (JMF) development library. JMF is an API defined jointly by Sun MicroSystems and IBM to develop multimedia applications. The main aim of this product is the provision of real-time delivery and multimedia data control (video and audio) by means of streaming techniques. The API can be used in applets and in stand-alone applications developed in Java. There are other possibilities for developing multimedia applications, such as Windows Media [4]. Nevertheless, the ease of use offered by the JMF library and the availability of the source code, led us to choose this library. The JMF 2.1.1 implementation provides a variety of audio and video codecs and some packetizers that allow the transmission of the coded content over RTP. For audio transmitting over Internet purposes there already exist low bit rate codecs, and its corresponding packetizers, that fit the requirements of the application (G-723 audio codec, i.e.). Nevertheless there exist only three possibilities for transmitting codec video over RTP: JPEG, H.263 or MPEG-1 (only for content already coded in this format). Obviously, under the bandwidth restriction of Internet there is only one possibility left and that is H.263. This codec achieves low bit rates for low motion videos but this may still not be enough for certain channels. In order to allow a better adaptation to the channel changing parameters and to improve quality and error resilience, we have added the capability to encode, decode, transmit and receive over a RTP channel MPEG-4 streams to the JMF library.

Nowadays, most multimedia applications use the Real-Time-Transport Protocol, RTP [8]. RTP provides end-to-end delivery services for data with real-time characteristics. These services are suitable for distributed applications that transmit real-time data, such as interactive audio and video. As for the session control, the Real Time Streaming Protocol, RTSP [9] is most used protocol for on-demand applications. Nevertheless, the current version of JMF supplies the RTSP client but does not include the RTSP server. For this reason, the decision was taken to implement a proprietary client-server control protocol that was able to offer this functionality in a simple way.

This paper is structured as follows: Section 2 describes briefly MPEG-4 codec and some results about XVID implementatin. Section 3 presents an illustrative example for the utilization of the JMFMoD system. In Section 4, an overview of the proposed MoD system, JMFMoD, is introduced. The different elements that constitute the complete system. Finally, the conclusions and future work are analysed in Section 6.

## MPEG4 AND MULTIMEDIA IN INTERNET

MPEG-4 provides very efficient video coding covering the range from the very low bit rates of wireless communication to bit rates and quality levels beyond high definition television. In contrast to the "frame-based" video coding of MPEG-1 and H.263, MPEG-4 is object based. Each scene is composed of Video Objects (VOs) that are coded individually. Very low bitrate (VLBV) coding with rates of 5-64kbps is provided for wireless and PSTN applications. MPEG-4 can encode low motion video contents such as talking heads with long predictive sequences to substantially reduce the bitrate. Compression of information inherently removes redundancies making it more difficult to decode information under error conditions. To address these issues the MPEG-4 standards define a set of special error resiliency tools. The standard supports flexible re-synchronization markers, data partitioning to separate motion and header information from texture information, reversible variable length coding (RVLC), and forced intra-frame refresh [12].

The XVID implementation of the MPEG-4 simple video profile is an open-source project. It is written in C and can be compiled on different platforms. It provides an API to access the core library. The video sequences that have been used in the codec characterization were taken from three popular films, each of them with different general color and movement characteristics: Blade Runner, Ice Age and Bridget Jones's Diary. These films were saved to disk from DVD (MPEG-2). From the original data, we grabbed 10001 frames at 25 fps out of the film and encoded it in YUV 4:2:0 QCIF format. For example, we can focus on one options level (quality level =6 ) and see how the different types of sequence affect the resulting average sizes for different values of quantizer parameter (Figure 1, Table 1).

We have created some JNI methods to access the XVID API from Java and its integration in JMF. These methods allow the programmer to initialize the encoder/decoder, code/decode one image and finalize the encoder/decoder from any Java code. Additionally we provide controls for live changing encoding parameters like the target bit rate, options parameters, quantizer and maximum key frame interval, by implementing de Control interface defined in JMF.

## A REAL EXAMPLE

In this section, we present a real example that illustrates the necessity of incorporating the various elements included in our system.

Let's suppose that an architecture teacher wishes to create a lecture presentation and store it in a media server so that the students can study the lecture whenever they wish through the Internet. The topic chosen by the teacher is the architectural style of Valencia city. At the time of creating the presentation, the teacher decides to organize the lecture in the following sections: (a) introduction, (b) emblematic buildings of Valencia , and (c) summary.

During its presentation, the teacher wants to use various resources:

a) video and audio (teacher), text (transcribed audio) and slides to outline the main aspects that will be explained during the presentation.
b) video, audio, music, text and slides for showing the emblematic buildings of Valencia city.
c) video, audio, text and slides to conclude the lecture.

For creating this presentation, the teacher needs a tool with some basic features:

- flexibility – that is, the teacher wants to be able to easily extend or modify the presentation;
- modularity – the teacher must be able to, for example, give the same lecture in another language by simply changing the audio, text and slides files and saving the presentation with another name;
- graphic interface
- ease of use.

Once the presentation is created, the media files that take part in the presentation must be stored in a media server and the presentation itself has to be published on a web server. To do this, a database stores descriptions of the available presentations – organized in a hierarchical scheme of categories and subcategories Finally, the students will be able to access the web server by a single request to an URL address, load an applet, choose the lecture and control the presentation.

## JMFMoD System

Figure 2 shows the architecture of the JMFMoD system and the communication flow between the different elements. Basically, five elements can be distinguished: the authoring presentation tool, the client, the web server, the database and the media server.

- Authoring presentation tool: the developed stand-alone tool which allows the creation of multimedia presentations. The user has various options available for including text, slides, audio, and video files – together with their temporal relationships – as well as menus for selecting audio and video compression formats.
- Client: the client application is based on a Java applet which is started when the correspondent HTML page is loaded from the web server. Through the graphic interface, the users can select from the database the presentation they want to playback.
- Web server: stores the initial page and the Java applet. It also holds the authentication procedures and the database access management.
- Database: keeps the information catalogue of the available multimedia presentations.
- Media server: stores the media files (text, slides, audio, video) and sends them to the clients when requested.

In the developed system, the web server and the media server run over the same station (a Windows 2000 server), whereas the client application and the authoring presentation tool run over Windows 98, or Windows 2000. A Microsoft Access database was used.

With regard to the generic system operation, four stages can be considered:

- Stage 1) Deals with the creation of multimedia presentations (media specification and their temporal relationships), the database updating and also the storing of the media files in the media server.
- Stage 2) The clients access the web server using an http connection to load the initial page and java applet. After selecting the media presentation, the users run an authentication procedure to cipher the information. The client's personal information is validated by the web server. Without authentication, a client will only be able to visualize the presentations catalogue.
- Stage 3) At this point, the client starts the reception and synchronized playback of the multimedia files.
- Stage 4) The client has to be able to control the media reproduction (stop, rewind, pause, etc.) through a TCP connection using an easy proprietary control protocol.

Finally, the last step involves disconnection from the system, closing RTP sessions, and TCP control connections.

## JMFMoD Implementation

This section describes the global system operation, taking into account the real scenario presented in Section 3 and following the stages enumerated in Section 4.

### Creation of multimedia presentations

Figure 3 shows an example of the temporal relationships between the different media that appear in our real scenario. As explained earlier, the presentation starts with an introduction of the lecture (A) and involves the synchronization of the teacher's video, (v1), together with the audio (a1), the text (t1) (which corresponds to the transcribed audio), and the slides (s1,s2,s3,s4). In the next step, (B), three videos of architectural structures are presented (v1,v2,v3) each being complemented with a specific music (m1,m2,m3). Some related comments are made via audio (a2) together with transcribed text (t2). Specific data related with the monuments appear as text and slides (t3, s5). Finally, the teacher concludes the lecture with a summary (C) making use of video (v5), audio (a5), text (transcribed audio) (t5) and slides (s6,s7,s8).

The creation of presentations is carried out in two steps which allow the specification of intra and inter-frame synchronization. To achieve this objective, two tools have been developed: the *IntraMediaStamper* and the *InterMediaStamper*. These synchronization tools are integrated in the *Authoring Presentation Tool* application (Figure 4).

**Client access and presentation selection**

To attend clients requests, a server application, *MoDServer* has been developed. This application has a graphic interface for accessing the application that maintains the database, and also for changing the configuration parameters and managing the users. The *MoDServer* application is in charge of providing information about all the available multimedia presentations. This application listens to the client requests in a specific port and creates an independent thread for each new established connection. The client accesses the server by using an http connection in order to load the page and configuration applet. From the client applet, *MoDApplet*, the user can choose the presentation he wants to playback.

Once the presentation is selected, the client must run an authentication algorithm. To do this, the applet requires from the client his identification, i.e., a login name and a password.

**Media playback**

Once the client has selected the presentation and has run the authentication process, the server sends to the client the content of the *ConfigPresentation* file. In this way, the client can create the proper player for each flow that takes part in the presentation. Two types of media flows can be distinguished: rtp and http. Audio and video are encapsulated in rtp, whereas text and slides are transmitted through http. The player is integrated in an applet. For each rtp flow, a new rtp session is created for handling the rendering of the corresponding flow. Http flows (text and slides) are reproduced locally. So, the player has to previously obtain the text and the slide files through an http connection.

As well as creating a player for each multimedia flow, the client also creates a controller for synchronizing all the players. This controller is again a specific implemented player with the peculiarity of having no content to playback.

The bandwidth needed by a particular presentation basically depends on the number of rtp flows and the compression formats of each flow. In the latest version of the JMF library, it is possible to use JPEG, H.263 and MPEG4 for transmitting video over rtp.

**Control**

The system control protocol consists of all the exchanged messages between the client and the MoD server (stream control meesages). The stream control messages gives local control of the remote transmission to the client (Figure 5). These messages are sent over the TCP control connection to the server so that it can start, or stop, the transmission and set the media time of the RTP streams. There are five stream control messages:

- PLAY: starts or restarts the stopped streams.
- PAUSE: pauses the stream reproduction at the current time.
- REWIND: stops the reproduction and sets it to the beginning of the media file.
- POSITION: changes the stream media time to that specified by the parameter sent immediately after this message (in the same line) in nanoseconds.
- EJECT: ends the stream reproduction and releases any exclusive resource used in the transmission and reproduction.

## CONCLUSIONS AND FUTURE WORK

The developed system allows the creation and on-demand synchronized distribution of multimedia presentations. These presentations can include audio and video as well as text and slides. Presentations are created through an easy to use and intuitive graphic application.

The main contributions of the presented MoD system are:

- The possibility of creating customized presentations that include as many audio, video, text and slides flows as the author wishes.
- The development of synchronization tools that allow both intra and inter-frame temporal specification.
- The synchronized playback of all the media, for which the functions that provides the JMF library have had to be extended.

The only requirements are that the client station must have a browser and the JMF libraries available. Audio and video flows are transmitted by RTP, whereas text and slides are downloaded by http and then reproduced locally and simultaneously with the RTP flows. Access to the database is achieved through a proprietary control protocol which establishes a TCP connection.

## ACKNOWLEDGEMENT

This work has been developed as part of the CICYT TIC 2001-2786-C02-01 Spanish Project.


## REFERENCES


[1] Belda A., Pajares A., Guerri J.C., Palau C., Esteve C., "JMFVoD: design and implemetation in Java of a media streaming on-demand system", *In Proceedings of SCS Euromedia*, Modena, Italy, 2002, April.

[2] Blakowski G., Steinmetz R., "A Media Synchronization Survey: Reference Model, Specification, and Case Studies," *IEEE Journal on Selected Areas in Communications*, vol. 14, no. 1, 1996, January, pp. 5-35.

[3] Cruz I.F., Mahalley P.S., "Temporal Synchronization in Multimedia Presentations", *International Conference on Multimedia Computing and Systems*, vol 2, Florence, Italy, 1999, June, pp. 851-856.

[4] Huang S. and Hu, H., "Integrating Windows streaming media technologies into a virtual classroom environment*", Proceedings of the International Symposium on Multimedia Software Engineering*, 2000, pp: 411 -418.

[5] Lee J., 1999, "UvoD: An Unified Architecture for Video-on-Demand Services", *IEEE Communications Letters*, vol. 3, no. 9, 1999, September, pp.277-279.

[6] Little T.D.C., Ghafoor A., "Synchronization and Storage Models for Multimedia Objetcs," *IEEE Journal on Selected Areas in Communications*, vol. 8, no. 3, April, 1990, pp. 413-427.

[7] Nicolau C., "An Architecture for Real-Time Multimedia Communication Systems," IEEE *Journal on Selected Areas in Communications*, vol. 8, no. 3, 1990, April, pp. 391-400.

[8] Schulzrinne H., S. Casner, R. Frederick and V. Jacobson., "RTP: A Transport Protocol for Real-Time Applications". *RFC1889*, 1996, January.

[9] Schulzrinne H., Rao A. and Lanphier R., "Real Time Streaming Protocol". *RFC2326*, 1998, January.

[10] Wright W., "An Efficient Video-on-Demand Model", *IEEE Computer*, vol. 34, no. 5 , 2001, May, pp. 64-70,.

[11] Zink M., Griwodz C. and Steinmetz, R., 2001. "KOM Player – A Platform for Experimental VoD Research", *ISCC 2001*, Hammamet (Tunisia)

[12] Pereira F., Ebrahimi T. , "The MPEG-4 Book", *Prentice Hall PTR*, 2002.


## FIGURES AND TABLES

TABLE I
RATIO COMPRESSION VALUES

| | Alien | Blade Runner | Bridget Jones | Ice Age |
|---|---|---|---|---|
| q=4 | 65.443 | 61.952 | 54.596 | 42.612 |
| q=31 | 441 | 507.2 | 478.25 | 350.5 |

FIGURE. 1
AVERAGE SIZE (I FRAMES).

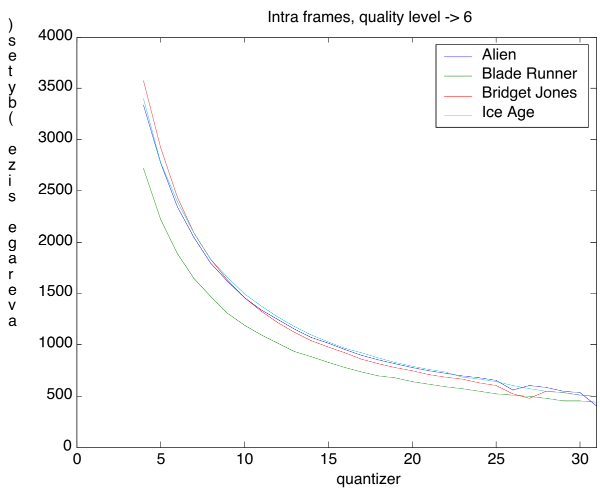

FIGURE. 2
JMFMOD ARCHITECTURE.

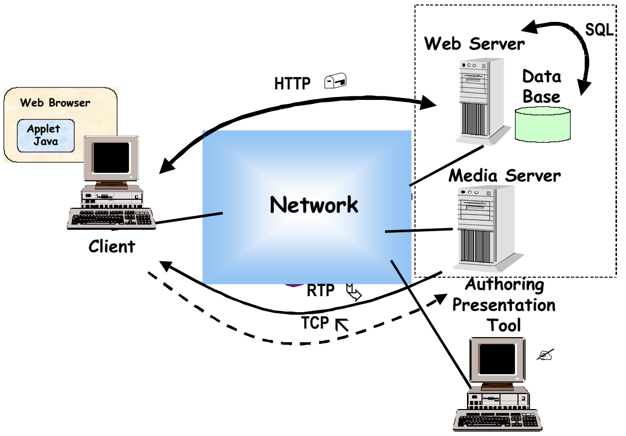


FIGURE. 3
TEMPORAL RELATIONSHIPS.

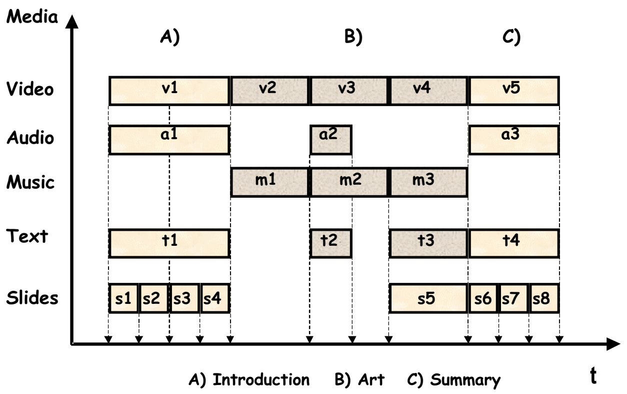


FIGURE. 4
AUTHORING TOOLS.

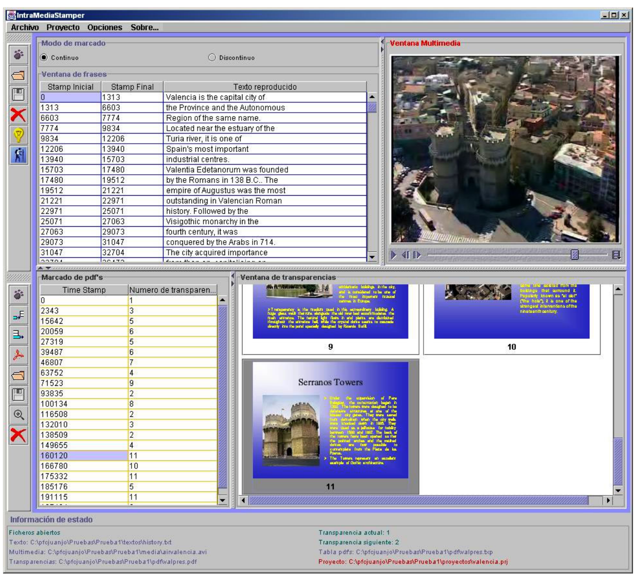

FIGURE. 5
USER INTERFACE.