

# An Interactive Object Based Multimedia System for IP Networks

Magda El Zarki, Liang Cheng, Haining Liu and Xiaoping Wei  
*Dept. of Information and Computer Science*  
*UC Irvine, Irvine, CA 92697*  
*{magda, lcheng61, haining, wei}@ics.uci.edu*

## Abstract

*In this paper, we present the design of a flexible framework for an IP-based network that will support an interactive multimedia (MM) system. The system will enable end users to: 1) author their own MM presentations, 2) control the delivery of the presentation, and 3) interact with the content (media objects) to make changes to the presentation in real time. To facilitate the design of the system we have introduced the concept of a meta object and have defined three levels of user interactivity: Presentation level interactivity, Session level interactivity, and Local level interactivity:*

## 1. Introduction

### 1.1. Background on MPEG 4 and Interactive Multimedia Services

MPEG-4, an ISO/IEC standard, provides a broad framework for the joint description, compression, storage, and transmission of natural and synthetic audio/visual data. MPEG-4 defines improved compression algorithms for audio and video signals, and efficient object-based representation of audio-video scenes [1]. It is expected that MPEG-4 will play an important role in multimedia applications over IP-based networks in the near future [4] because of the following reasons: First, MPEG-4 encompasses all types of media (video, audio, text, graphics, etc.). Second, MPEG-4 is fully object-oriented with scalability support for each object. And third, MPEG-4 supports end user interactivity.

In the MPEG-4 system standard, audio-visual objects are encoded separately into their own Elementary Streams (ES). The Scene Description (SD), also referred to as the Binary Format for Scene (BIFS), defines the spatio-temporal features of these objects in the final scene to be presented to the end user. The SD uses a tree-based graph and can be dynamically updated. The SD is conveyed between the source and the destination through one or more ESs and is transmitted separately. Object

Descriptors (ODs) are used to associate scene description components to the actual elementary streams that contain the corresponding coded media data. Each OD groups all descriptive components that are related to a single media object, e.g., an audio or video object, or even an animated face. ODs carry information on the hierarchical relationships, locations, and properties of ESs. ODs themselves are also transported in ESs. The set of ODs can be seen as a stream level session and resource description of an MPEG-4 presentation [5]. The separate transport of media objects, SD, and ODs enables flexible user interactivity and content management.

The object-oriented nature of MPEG-4 makes it possible for an end user to manipulate the media objects and create a multimedia presentation tailored to his or her specific needs. The multimedia content resides on a server (or bank of servers) and the end user has either local or remote access to the service. The end user can pick the desired language, the quality of the video, the format of the text, etc. This service model differs from the traditional streaming applications that we are used to because of its emphasis on end user interactivity. To date, nearly all the streaming multimedia applications that are running on the Internet are designed for simple remote retrieval or for broadcasting/multicasting services.

Interactive services are useful in settings such as distance learning, where the student can access a database of course content and tailor a presentation with the appropriate modules based upon the level and speed at which he or she wants to progress. It can also be envisioned for use in gaming environments, where players set up their own environments or for travelers planning a trip and selecting a variety of things they want to view and read. Special education is another area in which user interactivity can play an important role.

There are only a few MPEG-4 interactive client-server systems discussed in the literature. H. Kalva *et al.* describe an implementation of a streaming client-server

system based on an IP QoS Model called XRM [6]. As such it cannot be extended for use over a generic IP network (it requires a specific broadband kernel – xbind). Y. Pourmohammadi *et al.* propose a DMIF based system design for IP-based networks. However, their system is mainly geared toward remote retrieval and very little is mentioned in the paper regarding client interactivity with respect to the actual presentation playback [7]. In [6], the authors present the Command Descriptor Framework (CDF) which provides a means to associate commands with media objects in the SD for some user interactivity. EnvivioTV [33] is an MPEG-4 MM viewer which can playback streamed MP4 files while providing some session level interactivity with the content. The embedded interactivity includes switching multimedia content, changing rendering style (transparency and layout) and meta-information retrieval. The interactivity is session level because any user interaction that involves trying to change the rendering content within the current session will result in the interruption of the whole session, i.e., stop and reset of the entire session.

### 1.2. Overview of Work

Our MM system, motivated by the MPEG-4 standard, is **object-based** with a strong emphasis on **user interactivity**. The system is designed to enable two main functions: first, a user can author a MM presentation using media objects residing in a media repository at a server, and second, a user can view a MM presentation and interact with the objects in the presentation. The objects have associated with them descriptors, such as bit rate, resolution, etc. The presentation description therefore consists of a scene description relating objects temporally and spatially, and each object in the scene has associated with it an object descriptor that defines its attributes. A user can change the attributes of an object during presentation playback. A user also has VCR-like control (stop, pause, fast forward and fast rewind) over the objects in the presentation. These two features constitute the core aspects of the system. In Figure 1 and 2 below we show an example of a scene from a distance learning application with its corresponding PD.



Figure 1. A distance learning application

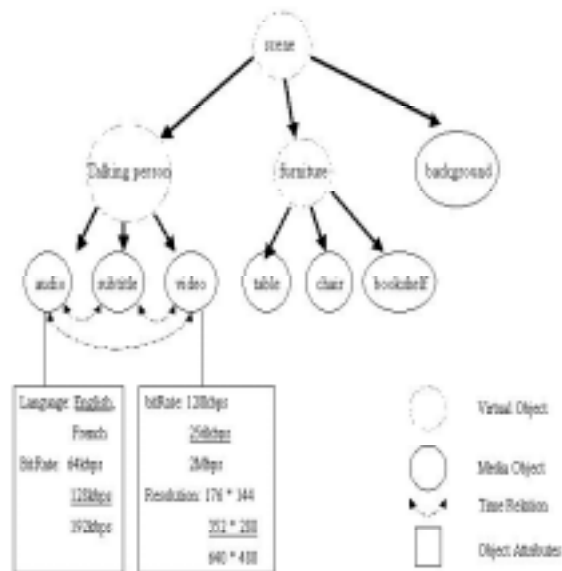


Figure 2. PD of distance learning application

**Object-based** - We have defined a new object type: a *meta object* to easily integrate and enable end user control over the media objects (video, audio, image, text, etc.) in a MM presentation. Some objects, such as a multi layer video object, will consist of several “sub” objects that correspond to the different layers, and may be linked to other “coupled” objects such as a set of associated audio tracks. A video meta object would then consist of all the layer objects and the corresponding audio track objects. A meta object is used to group the full set of “sub” and “coupled” objects associated with a particular object. The meta object therefore defines the attributes of a media object.

**User Interactivity** – In order to enable user interactivity during playback we have defined three types of interactivity levels: 1) *Presentation level interactivity*: in which a user makes changes to the scene by changing the characteristics of an object (size, resolution, sound, language track, etc.). The user can also add and delete objects in real time. 2) *Session level interactivity*: in which a user controls the playback process of the presentation (i.e., VCR-like control for the whole session and for objects (or group of objects if temporally linked)). 3) *Local level interactivity*: in which a user only makes changes that can be taken care of locally, e.g., changing the position of an object on the screen, volume control, etc.

Many issues must be considered for a MM system that enables end user interactivity at the object level during playback. To list a few: 1) transmission of end user interactivity commands to the server, 2) transport of

media content with quality of service considerations (scheduling, rate adaptation and buffer allocation), 3) real time session control based upon end user interactivity, 4) mapping of data streams onto Internet transport protocols, 5) extension of existing IETF signaling and control protocols to support multiple media streams in an interactive environment, etc. To accommodate these issues, our system design, although it draws on many of the features of MPEG-4, deviates from the published standard in several instances: 1) We do not make use of DMIF since our system is mainly targeted for interactive client-server applications over IP. DMIF is fairly complex and we do not need all of the generalities specified by the primitives. 2) We define a Presentation Description (PD), based on the Synchronized Multimedia Integration Language (SMIL) [26]. The PD incorporates not only the scene description but also object information. In other words, we have augmented the MPEG-4 defined SD based on BIFS to include object information. 3) We use neither the initial object descriptor (IOD) nor the OD streams as defined by the MPEG-4 standard, our system design uses available Internet signaling protocols to convey session information between the server and the client.

In the sections below we present the design of our object based interactive system that supports end user interactivity at the three levels defined above.

## 2. System Architecture

The system architecture is depicted in Figure 3. It consists of: 1) a MM server, which stores encoded MM objects and streams the MM content, 2) a MM client, which serves as the platform for the composition and playback of a MM presentation. The server and client communicate over an IP network.

The system consists of 3 logical components. Each component has a distinct role in the overall system, one deals with the media data and its transport over an IP based network, the second deals with the end user control of the media data and the third deals with the creation and presentation of MM material. The three components are: 1) The Data Plane: which comprises the media layer responsible for media coding/decoding and synchronization, and the delivery layer responsible for transport of the media data. 2) The Control Plane: which is responsible for enabling the three different levels of interactivity defined above. 3) The Application Modules: which consist of 1) an MM authoring tool for presentation creation, and 2) an interactive media player (IPlayer) for scene composition/rendering with a GUI that will allow interactivity during playback.

The data plane and a control plane include features to allow for dynamic service provisioning (reacting to end user actions). These two planes reside at the server and the client sides and communicate over the IP network. The control plane communicates the end user actions to the data plane for session initiation and termination. The authoring tool embeds the object information in the PD that is then used by the media player for user interactivity during playback.

The essence of our MM platform lies in its object-oriented structure and user interactivity. Each ES, which carries the content of a media object, is transmitted to the client over its own transport channel. This approach gives the end user at the client side tremendous flexibility to interact with the multimedia presentation and to manipulate the different media objects. End users can change the spatio-temporal relationships among media objects, turn on or shut down media objects, specify different perceptual quality requirements for different media objects, and have VCR-like control over the session and objects therein. This results in a complex session management and control architecture. For example, if a user decides to delete a sound track from the presentation, the control plane sends the action to the media layer, the media layer stops the flow and informs the delivery layer to terminate the associated transport channel. If the user chooses to change the resolution of an object, for example to a higher resolution, this will require adding one or more enhancement layers, which will need a new transport channel to be initiated for its transport. Our design is centered around the premise that end user interactivity is crucial to the service and therefore it targets a flexible session management scheme with efficient and adaptive encapsulation of data. In the next sections, we describe the functionality of the different planes and modules and show they interact to build a complete media delivery framework.

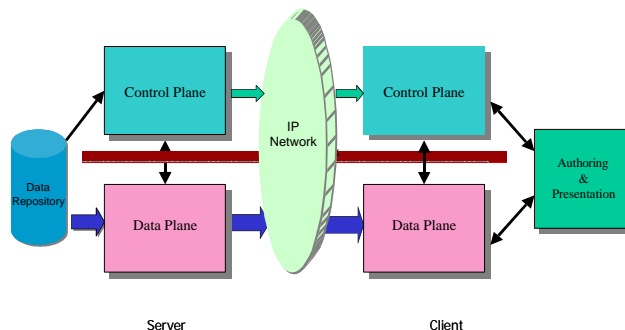


Figure 3. Overview of system architecture

### 2.1. The Data Plane

The data plane is responsible for all actions associated with the retrieval, streaming, and playback of the media data. The data plane's two main functions can be summarized as one that deals with the media itself and its synchronization (the media layer) and the other that deals solely with the transport of the media data (the delivery layer).

**The Media Layer** - This layer covers all aspects of the media data including creation of the media objects for the data repository, and intra/inter stream synchronization. The system design supports natural video (multi layered and shape coding), audio, text and graphics media. Below we show an example of layered video and one of arbitrary shaped coding produced in our lab. The data repository stores all the media objects.

**Synchronization Layer (SL)**- Functionally, the SL is designed to implement the synchronization and timing mechanisms in MPEG-4. Unlike its predecessor MPEG-2, MPEG-4 does not specify an associated transport layer, instead, it defines an SL that provides a standard interface to link the compression layer for all kinds of transport protocols: TCP, IP, RTP, MPEG-2TS, etc.



**Figure 4. A high resolution layered arbitrary shaped foreground object with a low resolution background. Snapshot taken in our lab.**



**Figure 5. Two arbitrary shaped foreground objects with a JPEG background. Streamed from a server to a client over an IP test bed in our lab.**

Our design provides a very flexible and configurable packetization scheme that allows the inclusion of timing,

fragmentation, error checking, and continuity information in the SL data packets.

**The Delivery Layer** - The design of the delivery layer poses some very challenging problems due to the multi-stream nature of the applications and the real-time user interactivity. The design incorporates a transport system that is fully compliant with the Internet yet provides some extra functionality at the server side (scheduling) to accommodate the service requirements of the different streams.

The delivery layer is responsible for interfacing with the media layer and translating the requirements of the different streams into transport services, such as scheduling, buffering requirements, rate adaptation, etc. The design consists of a Session Adaptor and sending and receiving modules that are largely based on Real-time Transport Protocol (RTP) [11-14]. A Session Adaptor along with the RTP Senders that pump the media data into the IP network, reside at the server side. Likewise, another Session Adaptor along with the RTP Receivers that retrieve the media data from the network reside at the client side. An RTP Sender has a Sending Module (SM) pumping media data and a Feedback Module (FM) collecting feedback data from the peer RTP Receiver. An RTP Receiver has a Receiving Module (RM) which reads and buffers the packets from the network and a FM that collects the status of the receiving process.

## 2.2. The Control Plane

In our system the control plane, at both the client and the server side, exchanges and processes control messages triggered by user interactivity events. The control plane interacts and cooperates with the data plane to support the three levels of user interactivity. The control plane has three distinct functional modules: presentation control, signaling, and session control, as illustrated in Figure 6.

The presentation control module at the client side sends to the server side control messages related to presentation level user interactivity, which could be a request for an authored or pre-formatted presentation, or changes to an ongoing presentation. The contents embedded in such presentation control messages will be a SMIL based PD or changes to the PD [26]. The server side presentation control module will then pass the PD, or updated information related to the PD, to the data plane, which will parse the information and take the appropriate steps with regard to scheduling the flows and manipulating the transport channels.

The signaling module at the server side parses the PD and communicates with its counterpart at the client side to

establish the necessary network connections. It informs the client side signaling module about media type and delivery encapsulation format (packet header configuration) for each ES. Modifying and terminating an existing network connection are also the responsibility of the signaling module if initiated by the end user. Note that the Media and Delivery layers react to network congestion and will take the necessary actions to maintain the “health” of the connection. Any actions they take will be indicated to the control plane so that it is aware of the state of the session and can inform the user should degradation of the presentation occur as a result.



**Figure 6. Control plane**

The session control module at the client side forwards to the server the control messages that result from session level user interactivities, i.e. VCR-like control. The server side session control module then relays the information to the data plane for appropriate actions. For example, if a user FF a presentation to a certain point, the data plane at the server side will be informed by the session control module about the new time instant from where the media data should be read from disk and sent to the client. Session control actions will affect buffer management and scheduling in the data plane.

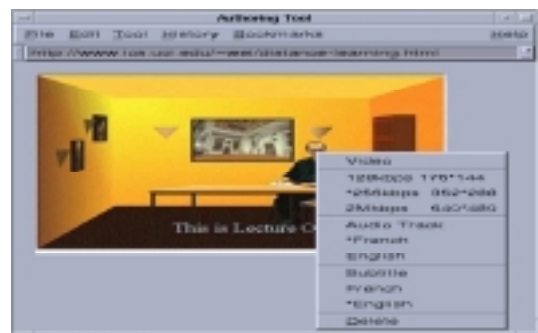
Control messages typically are critical, and require reliable delivery, therefore, we use the TCP/IP transport stack for control message transmission in the control plane of our system.

### 2.3. Application Modules

The system defines two modules that support an interactive MM application: one for authoring and the other for presentation and interactive playback. We will design a web based authoring tool that serves as the user interface to the data repository. The IPlayer we will design consists of the compositor (decoders), the renderer and the GUI which allows user interactivity during playback.

**Authoring Tool-** Several MM authoring tools exist [28-32] However none of them meet the specifications of our system. Therefore, we designed our own web based tool that embeds object information in the presentation for session setup and interactive playback. The tool is designed for remote access and presentation preparation. The user is able to browse the contents, i.e. the media objects, stored in the server database, pick the media objects in an authoring window, choose the preferred quality (bit rate, resolution etc.) of each individual object, and specify the temporal and spatial relationship between the objects. A preliminary implementation of the authoring tool is shown in Figure 7.

Once a user has finished authoring a presentation, a SMIL-based description of the presentation scene will be generated (the PD). The PD will contain a scene description that indicates the temporal and spatial relationship among all the media objects in the presentation, and an object descriptor for each individual media object in the scene as was illustrated in Figure 1 in the introduction. SMIL is an XML [27] based markup language that can be used to describe all the components in a multimedia presentation and their synchronization information. It offers great simplicity, flexibility and extensibility to describe a presentation. SMIL has been widely adopted and many media players such as Real Network’s Real Player G2 and Microsoft’s Windows Media Player supports SMIL. Building our presentation description on the basis of SMIL, we will achieve interoperability with other systems.



**Figure 7. Authoring Tool**

When the user wants to playback the presentation, the SMIL description of the presentation will first be sent to the server. The media layer then parses the description and initializes the delivery layer. At the client side the SL uses the PD for de-packetization and the IPlayer will use the PD to compose and render the presentation.

To facilitate authoring and user presentation control, we introduced a new object type: a *meta object*. We

describe the features of the meta object below and discuss its importance with respect to our system.

**Meta Object** - In our system, a meta object is associated with a media object, such as a video object, to describe the media format and “group” all related files. For example, a layered video in our system can have one video object for the base layer and a set of related video objects for the multiple enhancement layers. A meta object for this video then specifies how the base layer video object can be combined with one or more enhancement layer video objects to achieve a desired quality, i.e., a particular bit rate, a screen resolution, picture size, etc. A meta object is also used to describe the dependency relationship between a video object and several audio tracks. For example, a video clip of a lecture can be associated with multiple audio streams, each giving the lecture in a different language, i.e., an English, a Spanish or a Chinese audio track. The meta object for the video links the video to the multiple related audio objects.

Meta objects are designed to facilitate the authoring of a multimedia presentation and to support user interactivity during presentation playback. Our system provides a user with the control and freedom to author a presentation customized to his/her interest and environment. Take distance learning as an example, the video of a lecture is layer compressed and the audio is prepared in multiple languages. The video meta object, and the audio objects of the lecture are stored on the data repository at the server. A user can browse (using a web browser) the content of the server, pick a lecture and author a presentation with a particular video quality and a preferred language for the lecture. When the user clicks a meta object, a window will pop up and present to the user all the possible qualities, i.e. combinations of bit rate, resolution and picture size, of the video and the encoded language tracks. Based on the network connection and his/her language preference, the user can decide what the bit rate of the video should be and which language to listen to. Once the user has authored the presentation and submits the request to the server, the server then retrieves the quality specification of the requested video, looks it up in the meta object and determines what enhancement layer(s) should be sent along with the base layer to achieve that quality. This information is then included in the PD.

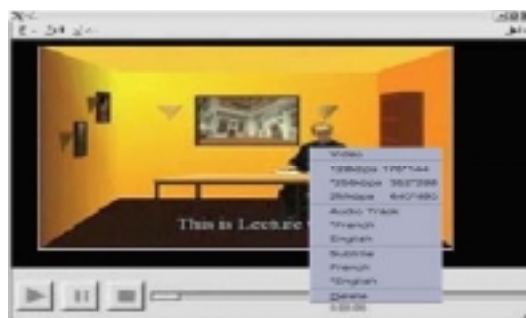
Similarly, meta objects can be used to support user interactivity in presentation playback. If in the middle of a presentation a user decides to change the quality of a particular video and/or the associated language, he can click on the video to pop up the menu window described in the previous paragraph, choose a different video quality and/or language. Such a change does not necessarily reset

the whole presentation, instead the presentation can proceed while the changes take effect on the fly. The user can opt to restart the session.

Furthermore, since a meta object contains the media format (codec) information about the object, it can be used to check the client's capability against the requested object when authoring a presentation. If a user request a video, for example in H.263 format, but does not have the capability, i.e. the H.263 codec is missing, he/she will be prompted with a warning message.

Conceptually, a video meta object is like a table, each entry of which specifies a quality and one or more enhancement layer video objects which are combined with the base layer video to achieve that quality. Currently a quality is designated by two attributes: resolution (CIF, QCIF etc) and bit rate (128kbps, 2Mbps etc). More quality attributes can be added in the future. Other linked objects are also listed in the table.

**The IPlayer** - According to the MPEG-4 System Decoder Timing Model (SDT), the inter-stream synchronization within one object can be established by extracting the Decoding Timing Stamp (DTS) and the Composition Timing Stamp (CTS) from SL-packet header. DTS and CTS act as a latch for the respective decoder and compositor buffers to control the data coming into decoder and compositor. Finally, the compositor uses the scene description contained in the PD to compose one Temporal-Spatial snapshot, which can be rendered by the renderer.



**Figure 8. IPlayer**

User interactivity is built into the GUI of the IPlayer. It consists of several buttons and sliders, as shown in Figure 8 below, that allows a user to click on an object and change its quality parameters or delete the object. The pop up menu used here is similar to the one used in the authoring tool. Objects may also be added. When “Add” is selected, the authoring tool is fired up and the user proceeds to pick from the data repository the required object(s). The changes to the presentation will be reflected in the PD with an update message. The

presentation proceeds with the new object added unless the user decides to restart the session.

### 3. Conclusions

In this paper we presented the design of an object based MM system that supports user interactivity at different levels, presentation, session and local. The system supports a layered and shaped based coder that is at the heart of object based system. Internet based protocols were implemented to provide the necessary control and signaling for the interactive system.

### 4. References

1. ISO/IEC JTC 1/SC 29/WG 11, "Information technology-Coding of audio-visual objects, Part1: Systems (ISO/IEC 14496-1)," Oct. 2001.
2. ISO/IEC JTC 1/SC 29/WG 11, "Information technology-Coding of audio-visual objects, Part6: Delivery Multimedia Integration Framework (ISO/IEC 14496-6)," Dec. 2000.
3. ISO/IEC JTC 1/SC 29/WG 11, "Information technology-Coding of audio-visual objects, Part8: Carriage of MPEG-4 contents over IP networks (ISO/IEC 14496-8)," Jan. 2001.
4. D. Wu, Y. T. Hou, W. Zhu, H. Lee, T. Chiang, Y. Zhang, H. J. Chao, "On End-to-End Architecture for Transporting MPEG-4 Video Over the Internet", *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 10, No. 6, Sep. 2000, pp. 923-941.
5. A. Basso, S. Varakliotis, "Transport of MPEG-4 over IP/RTP," *Proc. of ICME 2000*, Capri, Italy, June 2000.
6. H. Kalva, L. Tang, J. Huard, G. Tselikis, J. Zamora, L. Cheok, A. Eleftheriadis, "Implementing Multiplexing, Streaming, and Server Interaction for MPEG-4," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 9, No. 8, Dec. 1999.
7. Y. Pourmohammadi, K. A. Haghighi, A. Mohamed, H. M. Alnuweiri, "Streaming MPEG-4 over IP and Broadcast Networks: DMIF Based Architectures," *Proc. of Packet Video Workshop '01*, Seoul, Korea, May 2001.
8. A. Akhtar, H. Kalva, and A. Eleftheriadis, "Command Node Implementation", Contribution ISO-IEC JTC1/SC29/WG11 MGE99/4905, July 1999, Vancouver, Canada. ISO/IEC/SC29/WG11, "Text of ISO/IEC 14496-1/FPDAM 1(MPEG-4 System Version-2)," International Standards Organization, May 1999.
9. ISO/IEC/SC29/WG11, "Text of ISO/IEC 14496-1/FPDAM 1(MPEG-4 System Version-2)," International Standards Organization, May 1999.
10. A. Akhtar, H. Kalva, and A. Eleftheriadis, "Command Node Implementation in the IM1 Framework", Contribution ISO-IEC JT1/SC29/WG11 MGE00/5687, March 2000, Noordwijkerhout, Netherlands
11. Schulzrinne, Casner, Frederick, Jacobson, "RTP: A Transport Protocol for Real-Time Applications," *draft-ietf-avt-rtp-new-11.txt*, November 2001.
12. Avaro, Basso, Casner, Civanlar, Gentric, Herpel, Lifshitz, Lim, Perkins, van der Meer, "RTP Payload Format for MPEG-4 Streams," *draft-gentric-avt-mpeg4-multiSL-03.txt*, April 2001.
13. C.Roux et al, "RTP Payload Format for MPEG-4 Flexmultiplexed Streams," *draft-curet-avt-rtp-mpeg4-flexmux-00.txt*, Feb.'01.
14. Y. Kikuchi, T. Nomura, S. Fukunaga, Y. Matsui, H. Kimata, "RTP Payload Format for MPEG-4 Audio/Visual Streams," *RFC 3016*, Nov. 2000.
15. Handley, Schulzrinne, Schooler, Rosenberg, "SIP: Session Initiation Protocol," *ietf-sip-rfc2543bis-02.ps*, Nov.'00.
16. H. Schulzrinne, A. Rao, R. Lanphier, "Real Time Streaming Protocol (RTSP)", RFC 2326, April, 1998.
17. M.Handley, V. Jacobson, "SDP: Session Description Protocol", *RFC 2327*, April'98.
18. A. Johnston, S. Donovan, R. Sparks, C. Cunningham, D. Willis, J. Rosenberg, K. Summers, H. Schulzrinne, "SIP Call Flow Example," *draft-ietf-sip-call-flows-03.txt*, March 2001.
19. C. Herpel, "Elementary Stream Management in MPEG-4," *IEEE Trans on Circuits and Systems for Video Technology*, 9 (2) Mar. 1999, pp. 315-324.
20. C. Lin, J. Zhou, J. Youn, M. Sun, "MPEG Video Streaming with VCR-Functionality", *IEEE Trans. on Circuits and Systems for Video Technology*, 11 (3), Mar. 2001.
21. M.S. Chen, D.D. Kandlur, "Downloading and Stream Conversion: Supporting Interactive Playout of Videos in a Client Station", *Proc. 2<sup>nd</sup> Int'l IEEE Conf of Multimedia Computing and Systems*, 1995.
22. C. Yang, "User-interactionsupported data-retrieving engine for distributed multimedia presentations," *Communications*, Helsinki, Finland, 2001. vol. 10, pp.3244-3250.
23. K. Psannis, M. Hadjinicolaou, "Transmitting additional data of MPEG-2 compressed video to support interactive operations," *Intelligent Multimedia, Video and Speech Processing*, 2001, Hong Kong, China,
24. C. Lee, Y. Chang, W. Yang "An efficient strategy for supporting fully interactive display in a video-on-demand server," *Multi-Media Database Management Systems*, 1998, pp28-35.
25. Basso, Civanlar, Gentric, Herpel, Lifshitz, Lim, Perkins, Van Der Meer-Philips, "RTP Payload Format for MPEG-4 Streams", *draft-ietf-avt-mpeg4-multisl-04.txt*, Feb.'02.
26. W3C, Synchronized Multimedia Integration Language (SMIL) 2.0 Specification, <http://www.w3.org/TR/smil20/>
27. W3C, Extensible Markup Language (XML), <http://www.w3.org/XML>
28. <http://www.cnmtc.columbia.edu/flavor>
29. [MPEG4@ENST](mailto:MPEG4@ENST), <http://www.comelec.enst.fr/~dufourd/mpeg-4/index.html>
30. GSRT Authoring Tool, [http://uranus.ee.auth.gr/pened99/Demos/AuthoringTool/authoring\\_tool.html](http://uranus.ee.auth.gr/pened99/Demos/AuthoringTool/authoring_tool.html)
31. M.Y. Sung, S. J. Rho, J. H. Jang, "A SMIL-based Multimedia Presentation Authoring System and Some Remarks on Future Extension of SMIL", Packet Video 2002, Pittsburgh, PA.
32. M.-J. Shieh, K.-L. Peng, W.-C. Chen, "The Design and Implementation of a Visual MPEG-4 Scene-Authoring Tool", WEMP4, 2001.
33. <http://www.envivio.com>