Editor: Peiya Liu
Siemens Corporate Research

MHEG-5—Aims, Concepts, and Implementation Issues

Marica Echiffre, Claudio Marchisio, Pietro Marchisio, Paolo Panicciari, and Silvia Del Rossi

CSELT—Centro Studi e Laboratori Telecomunicazioni

The ISO MHEG-5 standard owes its existence to the increasing convergence of broadcast and interactive technologies. It specifies an encoding format for multimedia applications independently of service paradigms and network protocols. This article introduces the MHEG-5 standard, describes examples of functionality, and illustrates an implementation that covers both runtime presentation and multimedia authoring applications.

Motivation for MHEG-5

Every design generally represents a compromise between conflicting goals. MHEG-5's design is no exception, especially if you consider that MHEG-5 targets a continuously growing and fiercely competitive market where broadcast and interactive technologies converge almost daily.

Converging technologies have often stimulated adopting standard solutions. Multimedia applications standards provide more than just the obvious objectives of portability and interoperability. A good multimedia standard can become a reference solution for system developers and application programmers. It also promotes the use of modular architectures that rely on common components that accomplish a specific functionality, such as interpreting and presenting MHEG-5 applications to users. This task is performed by a compliant runtime engine (RTE), a resident software component for the OEM market that schedules delivery of an application to the user. It's aimed at a wide installation base within complete solutions, like a VoD or an ITV system. RTEs help improve a product's return on investment, abate a product's per-unit costs, and provide high-quality, robust products due to extensive product testing.

The family of MHEG standards

MHEG encompasses the family of standards issued by the ISO/IEC JTC1 joint technical committee's working group WG12—information technology, subcommittee SC29, coding of audio, picture multimedia, and hypermedia information. See Table 1 for the complete list of MHEG standards.

Since it was introduced first, MHEG-1[1] received the most attention. It's the generic standard for encoding multimedia objects without specific assumptions on the application area or on the target platform used for delivering and rendering these objects to the user. MHEG-3 provides a

Acronyms

API	application programming interface
Armida	applications retrieving multimedia information distributed over ATM
ASCII	American Standard Code for Information Interchange
ASN	abstract syntax notation
ATM	asynchronous transfer mode
CCETT	Centre Commun d'Etudes de Télédiffusion et Télécommunications
Davic	Digital Audio Visual Council
DSMCC	Digital Storage Media Command and Control
DVB	Digital Video Broadcasting
DVB-SI	Digital Video Broadcasting Service Information
HTML	Hypertext Markup Language
HTTP	hypertext transfer protocol
ITV	interactive TV
KDD	Kokusai Denshin Denwa
MHEG	Multimedia and Hypermedia Expert Group
MPEG	Moving Picture Expert Group
MUG	MHEG-5 Users Group
OEM	original equipment manufacturer
RTE	runtime engine
URL	uniform resource locator
VCR	videocassette recorder
VoD	video-on-demand

1070-986X/98/$10.00 © 1998 IEEE

Table 1. MHEG standards.

Nickname	Complete Name	Status
MHEG-1	MHEG object representation—base notation (ASN.1)	International standard
MHEG-2	MHEG object representation—alternate notation (SGML)	Withdrawn
MHEG-3	MHEG script interchange representation	International standard
MHEG-4	MHEG registration procedure	International standard
MHEG-5	Support for base-level interactive applications	International standard
MHEG-6	Support for enhanced interactive applications	Draft international standard
MHEG-7	Interoperability and conformance testing for ISO/IEC 13522-5	Working draft

script extension to MHEG-1. MHEG-4 specifies a registration procedure for identifiers used by the objects to identify, for example, a specific format for content data.

MHEG-5[2] can conceptually be considered a simplifying profile of MHEG-1. It addresses terminals with limited resources, like the set-top unit. Actually, an MHEG-1 decoder can't decode MHEG-5 applications due to some slightly different provisions added to optimize performance in VoD/ITV environments.

MHEG-6[3] extends the "declarative" MHEG-5 approach with procedural code capabilities typical of a scripting language. It defines the interface (MHEG-5 API) and a script engine's runtime environment on top of an MHEG-5 engine using the Java virtual machine to provide a complete solution for application representation.

MHEG-7, a new standard, addresses the conformance and interoperability of MHEG-5 engines and applications.

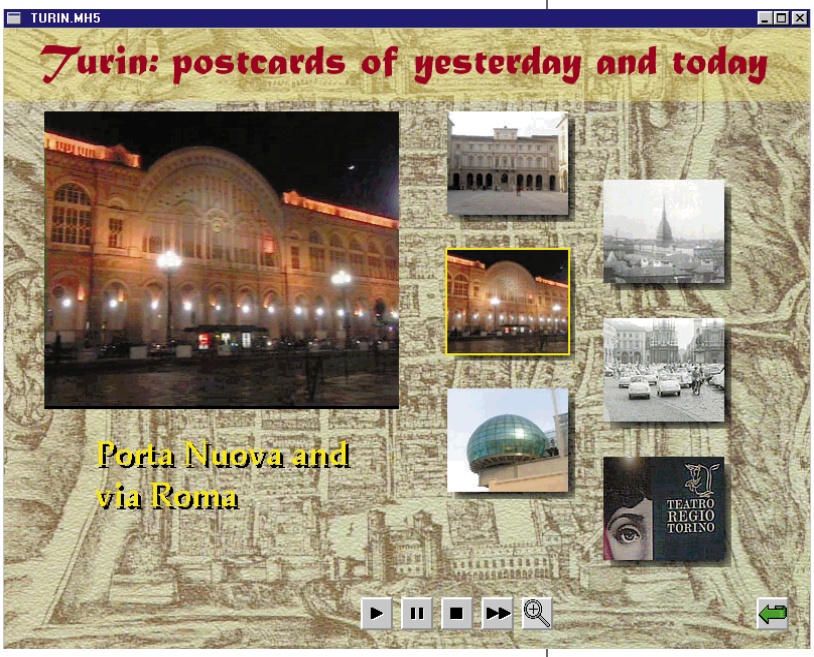

Figure 1. The look and feel of the Turin application.

MHEG-5 overview

A multimedia application can be conceived as a set of self-contained objects based on synchronization and spatial-temporal relationships of multiple media formats, structural composition, "event-action" associations, navigation, and user interaction capabilities. Controlling the playback of time-dependent contents, like streams of multiplexed audiovisual data requires specific support. These streams demand VCR control functions (play, pause, fast forward, and so on), as well as the capability to manage events generated during their presentation. For example, rendering text subtitles can be synchronized with time-code events generated during the playback of a stream.

MHEG-5 represents an application, as a set of scenes, which contain objects common to all scenes. A scene supports the spatially and temporally coordinated presentation of audiovisual content consisting of graphics, bitmaps, text, and streams (based on the multiplex of audio and video components). Interaction can be performed via graphic elements like buttons, sliders, text entry boxes, and hypertext selections. Every scene, as well as an entire application, is a self-contained entity that can represent its localized behavior by links that are event-action associations. Events can be generated by users, expiration of timers, playback of streams, and other conditions within the RTE.

For example, the "Turin" application lets users retrieve tourist information about the city of Turin, Italy. Figure 1 shows a screen shot of the "main_scene" object. This scene consists of

- a text title,
- a bitmap background,
- a video clip, which is playing a segment of

Porta Nuova, the main railway station, enriched with a text subtitle object (left side),

- some interactive thumbnails (right side),
- a set of buttons that provide VCR controls,
- functions to switch between viewing the video in "normal" and "zoom" modes, and
- a "return to previous screen" capability.

The playback of the video clip synchronizes with the subtitle's content. Selecting an interactive thumbnail constrains the playback to the associated temporal segment.

The MHEG-5 model is object-oriented. The actions are methods targeted to objects from different classes to perform a specific behavior: preparation, activation, controlling the presentation, user interaction, getting the value of attributes, and so on. To allow interoperability across heterogeneous systems, MHEG-5 specifies the precise encoding syntax. Two notations are possible: the ASN.1 notation, which MHEG-1 also adopts, and a textual notation. Figure 2 shows an excerpt from the Turin application's textual notation.

In a client-server architecture, MHEG-5 applications are stored on the server and downloaded to the terminal for the RTE to interpret.

This model is not limited to storage and retrieval services. In the broadcast environment, for example, the set of channels transmitted on a broadcast network can be considered a virtual server, where the download mechanism relies on cyclic rebroadcast of all portions of an application.

Major MHEG-5 classes

MHEG-5 adopted the object-oriented approach, since it provides suitable abstractions for managing active, autonomous, and reusable entities. A class is specified by three kinds of properties:

- attributes that make up an object's structure,
- events that originate from an object, and
- actions that target an object to accomplish a specific behavior or to set or get an attribute's value.

Here we'll provide an overall description of only the most significant classes.

Root

A common Root superclass provides a uniform object identification mechanism and specifies the general semantics for preparation/destruction and activation/deactivation of objects, including notification of changes of an object's availability and running status. These general provisions are further specialized moving downwards through the inheritance tree, which first branches into the Group and Ingredient classes.

Group

This abstract class handles the grouping of objects in the Ingredient class as a unique entity of interchange. In fact, Group objects can be addressed and independently downloaded from the server. A Group can be specialized into Application and Scene classes.

Application. An MHEG-5 application is structurally organized into one Application and one or more Scene objects. The Application object represents the entry point that performs a transition to the presentation's first Scene. Generally, this transition occurs at startup (see Figure 2) because a presentation can't happen without a Scene running.

The Launch action activates an Application after quitting the active Application. The Quit action ends the active Application, which also terminates the active Scene's presentation. The Ingredients of an Application are available to the different Scenes that become active, thereby allowing an uninterrupted presentation of contents (for example, a bitmap can serve as the common background for all Scenes in an Application).

Scene. This class allows spatially and temporally coordinated presentations of Ingredients. At most, one Scene can be active at one time. Navigating within an Application is performed via the TransitionTo action that closes the current Scene, including its Ingredients, and activates the new one.

The SceneCoordinateSystem attribute specifies the presentation space's 2D size for the Scene. If a user interaction occurs in this space, a UserInput event is generated. A Scene also supports timers. A Timer event is generated when a timer expires.

Ingredient

This abstract class provides the common behavior for all objects that can be included in an Application or a Scene.

The OriginalContent attribute maps object and content data. It contains either included content

```
{:Application ("turin.mh5" 0)
  :OnStartUp ( // sequence of initialization
   actions
    :TransitionTo (("main_scene.mh5" 0)) //
     activation of the first scene
  )
}

{:Scene ("main_scene.mh5" 0)
  :OnStartUp ( // sequence of initialization
   actions
    preload (2) // the connection to the
     source of the video clip is set up
    ...
    setCounterTrigger (2 3 190000) // book a
     time code event at 190000 msec
    ...
  )
  :Items ( // both presentable ingredients and
   links
    {:Bitmap 1 // background bitmap
      :InitiallyActive true
      :CHook 3 // JPEG
      :OrigContent
        :ContentRef ("background.jpg")
      :OrigBoxSize 800 600
      :OrigPosition 0 0
    }
    {:Stream 2 // video clip
      :InitiallyActive false
      :CHook 101 // MPEG-1
      :OrigContent
        :ContentRef ("turin.mpg")
      :Multiplex (
        {:Audio 3 // audio component of the
         video clip
          :ComponentTag 1 // refers to audio
           elementary stream
          :InitiallyActive true
            }
          {:Video 4 // video component of the
           video clip
            :ComponentTag 2 // refers to
             video elementary stream
            :InitiallyActive true
            :OrigBoxSize 352 288
              :OrigPosition 40 80
            }
        )
      }
      {:HotSpot 20 // "Porta Nuova" hotspot
        :InitiallyActive true
        :OrigBoxSize 120 100
        :OrigPosition 440 214
      }
      ... // 25 more presentable ingredients
      {:Link 30 // selecting an "interactive"
       thumbnail
        :EventSource (20) // "Porta Nuova"
         hotspot
        :EventType IsSelected
        :LinkEffect (
          :SetSpeed (2 1 1) // video clip:
           speed is set to 1/1 (normal)
          :SetCounterPosition (2 190000) //
           initial point of the segment
          :SetCounterEndPosition (2 246500) //
           end point of the segment
          :Run (2) // activate playback with
           the above settings
        )
      }
      {:Link 49 // the video clip crosses a pre
       defined time code position
        :EventSource (2) // video clip
        :EventType CounterTrigger
        :EventData 3 // booked at startup by
         setCounterTrigger (2 3 190000)
        :LinkEffect (
          :SetData (5 // text subtitle is set
           to a new string, that is
            :NewRefContent ("st9.txt")) //
             "Porta Nuova and via Roma"
          :SetHighlightStatus (20 true) //
           hotspot 20 is highlighted
        )
      }
      ... // 58 more links
    )
  :SceneCS 800 600 // size of the scene's
   presentation space
}
```

Figure 2. The textual notation for the "Turin" application.

or a reference to an external data source (such as a URL or a DSMCC file name). The ContentHook attribute specifies the encoding format for the content. However, MHEG-5 does not list the supported encoding formats. See Figure 2 for the use of content references and hooks.

The action Preload gives hints to the RTE for making the content available for presentation. Especially for streams, this action does not completely download the content, it just sets up the

proper network connection to the site where the content is stored. The action Unload frees allocated resources for the content.

The Presentable, Stream, and Link classes are subclasses of the Ingredient class.

Presentable

This abstract class specifies the common aspects for information that can be seen or heard by the user. The Run and Stop actions activate and terminate the presentation, while generating the IsRunning and IsStopped events.

Visible

The Visible abstract class specializes the Presentable class with provisions for displaying objects in the active Scene's presentation space. The OriginalBoxSize and OriginalPosition attributes respectively specify the size and position of the object's bounding box relative to the Scene's presentation space. The actions SetSize and SetPosition change the current values of these attributes.

Here we'll discuss the specialized objects in the Visible class.

- *Bitmap*. This object displays a 2D array of pixels. The Tiling attribute specifies whether the content will be replicated throughout the BoxSize area. The action ScaleBitmap scales the content to a new size.

- *LineArt, DynamicLineArt*. A LineArt is a vectorial representation of graphical entities, like polylines and ellipses. DynamicLineArt draws lines and curves on the fly in the BoxSize area.

- *Text*. This object represents a text string with a set of rendition attributes. Essentially, these attributes specify fonts and formatting information like justification and wrapping.

Stream

This class (a subclass of Ingredient) controls the synchronized presentation of multiplexed audiovisual data (such as an MPEG-2 file). A Stream object consists of a list of components from the Video, Audio, and RTGraphics (animated graphics) classes. The OriginalContent attribute of the Stream object refers to the whole multiplex of data streams.

When a Stream object is running, its streams can be switched on and off independently. This lets users switch between different audio trails (different languages) or choose which video stream(s) to present among a range of available ones. For example, Figure 2 contains an MPEG-1 Stream composed of one audio and one video component. These components automatically activate when a run action targets the whole Stream because their InitiallyActive attribute is set to "true."

Specific events are associated with playback: StreamPlaying/StreamStopped notifies the actual initiation/termination and CounterTrigger notifies the system when a previously booked timecode event occurs. Figure 2 shows how the CounterTrigger event can be used to synchronize text subtitling and also illustrates the SetCounterPosition and SetCounterEndPosition actions to specify a temporal segment for presentation.

Interactible

This abstract class provides a way for users to interact with objects within the following subclasses:

- *Hotspot, PushButton,* and *SwitchButton*. These subclasses implement button selection capability and generate the IsSelected event.

- *Hypertext*. This class extends the Text class with "anchors." When selected, these anchors link text content to associated information.

- *Slider* and *EntryField*. Respectively, these objects let users adjust a numeric value (such as the volume of an audio stream) and edit text.

Link

The Link class (another special class of Ingredient) implements event-action behavior by a condition and an effect. The LinkCondition contains an EventSource—a reference to the object on which the event occurs—an EventType that specifies the kind of event and a possible EventData that is a data value associated with the event. In Figure 2, "Link 49" triggers only if a CounterTrigger event on the video clip occurs with "EventData = 3." Specifically, this lets you associate a different effect with every booked value of the CounterPosition.

The LinkEffect comprises a set of actions that are executed in sequence when an event that matches with the LinkCondition occurs. Every action specifies the target object and, possibly, other parameters depending on the type of action. MHEG-5 specifies more than 100 kinds of actions.

Relationships to major standards

Important relationships exist between MHEG-5 and other standards and specifications.

Davic

Davic[4] aims to maximize interoperability across applications and services for the broadcast and interactive domains. It selects existing standards and specifies their interfaces in the realm of a complete reference model. This comprises the content provider system, the service provider system, and the service consumer system. This forum has prepared a series of specifications: Davic 1.0 selected MHEG-5 for encoding base-level applications and Davic 1.1 relies on MHEG-6 to extend these applications in terms of the Java virtual machine that uses services from the MHEG-5 RTE.

DVB

DVB provides a complete solution for digital television and data broadcasting across a range of delivery media where audio and video signals are encoded in MPEG-2. The specification includes an open service information system, known as DVB-SI,[5] which provides the elements necessary for developing a basic electronic program guide (EPG) to support navigation amongst the new digital television services.

MPEG

MPEG is the name of the well-known family of standards used for coding audiovisual information (such as movies, video, and music) in a digital compressed format. MPEG-1 and MPEG-2 streams are likely to be used by MHEG-5 applications, which can easily control their playback through the facilities provided by the Stream class.

DSMCC

DSMCC[6] specifies a set of protocols for controlling and managing MPEG streams in a client-server environment. The user-to-user protocol (both the client and server are users) consists of VCR commands for playback of streams stored on the server, as well as commands for downloading other data (bitmaps, text, and so on). The playback of MPEG streams from within MHEG-5 applications has a counterpart at the DSMCC layer.

Implementation

We implemented both an RTE and an authoring tool for MHEG-5 applications. These tools operate in Armida, a client-server system for interactively retrieving multimedia applications from remote databases via broadband networks. Armida aims to comply with Davic 1.0's specifications for systems supporting interactive multimedia services and applications. Armida also receives, decodes, and displays TV programs conforming to the DVB standard. It adopts MPEG-2 for high-quality audiovisual streams, the DSMCC user-to-user protocol for client-server interaction, and MHEG-5 for application encoding.

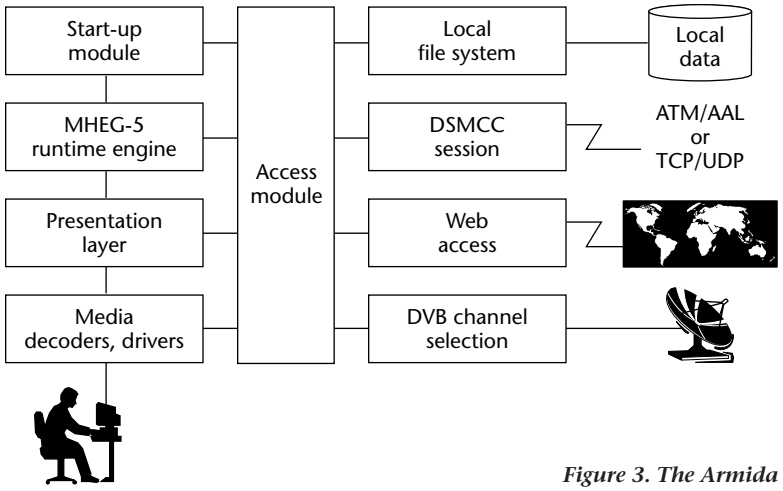

Figure 3. The Armida client architecture.

MHEG-5 runtime engine

Figure 3 shows the software architecture of the client (a PC with Microsoft Windows95/NT). The Start-up Module performs the general initialization: The client can be launched either as an autonomous Windows application or as a plug-in by an HTML browser, allowing seamless navigation between the World Wide Web and the webs of MHEG-5 applications. See Dal Lago[7] for more details.

The MHEG-5 RTE is the kernel of the client's architecture. It performs the pure interpretation of MHEG-5 objects and, as a platform-independent module, issues I/O and data access requests to other components that are optimized for the specific runtime platform.

The RTE performs two main tasks. First, it prepares the presentation and handles accessing, decoding, and managing MHEG-5 objects in their internal format. The second task is the actual presentation, which is based on an event loop where events trigger actions. These actions then become requests to the Presentation layer along with other actions that internally affect the engine.

Presentation layer. The presentation layer (PL) manages windowing resources, deals with low-level events, and performs decoding and render-

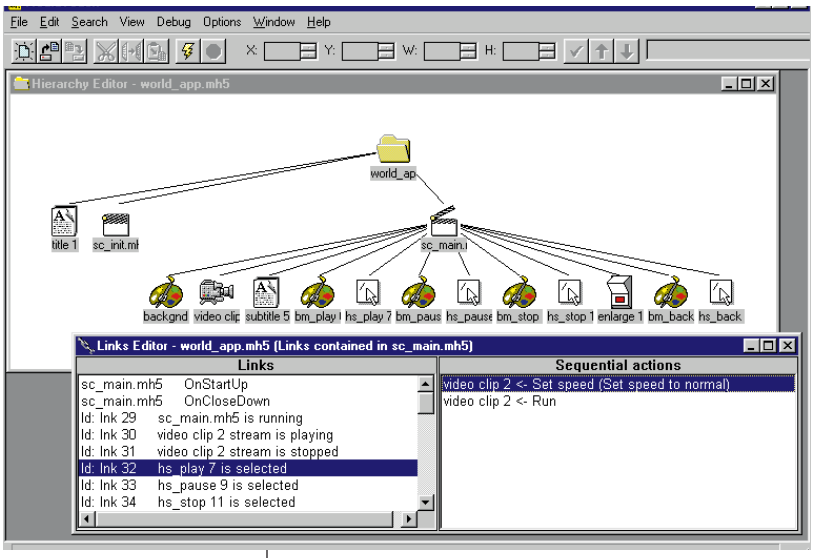

Figure 4. MediaTouch Hierarchy and Links Editor windows.

ing of contents from different media to the user. This functionality is available to the RTE via an object-oriented API that encapsulates all I/O platform-specific aspects. The basic MHEG-5 Presentable classes have counterparts at this API level, which makes provisions for initialization/termination, data access and decoding, setting specific attributes (such as text font, color, and so on), and performing spatial and temporal controls. In addition, an informative flow exists from the PL back to the RTE, which notifies user interaction and stream events.

Access module. This module provides a consistent API for accessing information from different sources. It's used by the RTE to get objects and the PL to access content data (either downloaded or streamed). Note that the selection of a particular delivery strategy is out of MHEG-5's scope, and hence remains an implementation issue. Our implementation supports

- bulk download for bitmaps, text, and MHEG-5 objects; and
- progressive download for audio and audiovisual streams.

The implementation of these mechanisms occurs via the DSMCC interface. The user has full interactive control of the data presentation, including playback of higher quality MPEG-2 streams delivered through an ATM network.

Object and content access requests can also be issued to the Web via HTTP. However, this widespread protocol still suffers from limitations when high-quality video clips are progressively downloaded, since the underlying networks may not yet provide adequate quality-of-service (QoS).

When accessing the broadcast service, the Access module requires the DVB channel selection component to select the program referred to by a Stream object. To achieve this, you set the ContentHook to a value that means "Switched DVB" and the OriginalContent to the identifier of the channel to select. The MPEG-2 stream transported by the selected channel is displayed in the active Scene's presentation space.

MediaTouch

The availability of an adequate authoring tool is mandatory to create the applications delivered by Armida. We developed MediaTouch, a visual-based authoring tool, to create these applications. It's based on the native approach, which lets the author operate at the level of MHEG-5 objects.

MediaTouch provides authors with the following functions:

- *Hierarchy Editor*. Supports the interactive creation of the structure of Applications and Scenes objects. The author operates by adding, moving, and deleting Ingredients on the tree that represents the hierarchy's current status. Several Scenes from different Applications can be edited at one time and you can copy and move Ingredients between Scenes and Applications.
- *Properties Editor*. The author sets the properties of Presentable Ingredients via an input form that is specific to each object class. The author does not need to type a value directly, since the system adopts a menu-based input interface that aims to minimize errors and inconsistencies.
- *Layout Editor*. A Scene's layout can be set by interactively adjusting the position and size of the bounding box for every Visible Ingredient. An Ingredient's content data, when available, is also displayed within its bounding box to make the layout match the actual presentation.
- *Links Editor*. This function lets authors add, modify, and delete links and actions to Application and Scene objects. The author sets the link conditions, actions, and referenced objects via a menu. Links and actions can also be copied between different scenes and applications.

Figure 4 shows a screen shot from MediaTouch where an Application composed of two Scenes and one shared Text is open within the Hierarchy Editor window. One of these Scenes shows its Presentable Ingredients, and the author is editing its Links. The author can then launch the RTE from within MediaTouch to check the outcome of his work.

Future plans

Several companies and research institutes are currently developing MHEG-5 tools and applications and conducting interoperability experiments for international projects and consortia.

The MHEG Support Center is a European project that hopes to implement and operate an MHEG support and conformance testing environment for developers of multimedia systems and applications. Its partners include CCETT, Cril Ingenierie (France), De teBerkom, and GMD Fokus (Germany).

IPA, the Information-technology Promotion Agency of Japan, is developing an MHEG-5 system with an authoring tool and viewer. The project members consist of ASCII, KDD, NEC, NTT Software, the Oki Electric Industry, Pioneer, Sony, and Toppan Printing. The purpose of this project is to do a profiling study and conduct interoperability testing of MHEG-5 systems.

Also, the following European achievements are worth mentioning:

- a Java implementation of an MHEG-5 engine, available from Philips;
- MhegDitor by CCETT, an authoring tool based on Macromedia Director; and
- the results of the Jupiter project that addresses usability, performance, and interoperability of Davic-compliant services.

At CSELT, a partner of Jupiter, we're making more complete versions of MediaTouch and Armida. We've demonstrated these systems at a number of events on behalf of Telecom Italia and showed such applications as movies-on-demand, news-on-demand, tourist information, and interactive TV.

A list containing the above applications and other initiatives is maintained by the MHEG-5 Users Group (MUG), an unofficial forum begun by people from the standardization body to disseminate information and enable discussion on MHEG-5 (see http://www.fokus.gmd.de/ovma/mug).

It's evident that the initial interest in MHEG-1 and MHEG-3 has recently decreased due to the incoming ITV/VoD profile, which is based on MHEG-5 and MHEG-6. The specification of this "complete solution," urged and endorsed by Davic, will be finalized by April 1998 when MHEG-6 is scheduled to become an International Standard. Further efforts will be devoted to MHEG-7, which is expected to provide the final specification of conformance and interoperability aspects for MHEG-5 by January 1999. **MM**

Resources on the Web

For more information, please visit
MHEG-5 User Group
www.fokus.gmd.de/ovma/mug
Moving Pictures Expert Group
www.cselt.it/mpeg
www.mpeg.org
Digital Audio Visual Council
www.davic.org
Digital Video Broadcasting
www.dvb.org
CSELT, Multimedia and Video Services
www.cselt.it/ufv

References

1. T. Meyer-Boudnik and W. Effelsberg, "MHEG Explained," *IEEE Multimedia*, Spring 1995, Vol. 2, No. 1, pp. 26-38.
2. ISO 13522-5, *Information Technology—Coding of Multimedia and Hypermedia Information, Part 5: Support for Base-Level Interactive Applications*, Int'l Org. for Standardization/Int'l Electronics Comm., Geneva, 1996.
3. ISO Draft International Standard 13522-6, *Information Technology—Coding of Multimedia and Hypermedia Information, Part 6: Support for Enhanced Interactive Applications*, Int'l Org. for Standardization/Int'l Electronics Comm., Geneva, 1997.
4. Davic 1.0 Specifications, Revision 5.0, Imprimeries de Versoix, Versoix, Switzerland, Dec. 1995.
5. ETSI 300 468 2d ed.: *Digital Broadcasting Systems for Television, Sound, and Data Services, Specification for Service Information (SI) in Digital Video Broadcasting (DVB) Systems*, ETSI publications dept., Sophia Antipolis, France, 1997, http://www.etsi.fr.
6. ISO 13818-6, *Information Technology—Generic Coding of Moving Pictures and Associated Audio Information, Part 6: Digital Storage Media Command and Control*, Int'l Org. for Standardization/Int'l Electronics Comm., Geneva, 1996.
7. S. Dal Lago et al., "Armida: Multimedia Applications Across ATM-Based Networks Accessed via Internet Navigation," *Multimedia Tools and Applications*, Vol. 5, No. 2, Sept. 1997.

Contact Pietro Marchisio at CSELT, Multimedia and Video Services, Via G. Reiss Romoli 274, I-10148 Torino, Italy, e-mail pietro.marchisio@cselt.it.