

- good survey paper
- Issues for MHEG
- why an standard?
- MM/HM Model . OSE (open system environment)

MM/HM in open distributed environments.

June, 1994

COLA94: MHEG.HP

LOC: MHEG

REV: Feb/95

read 2/2/95

10/10

The MHEG standard principles and examples of applications

Françoise COLAITIS, Convener of ISO/IEC JTC1/SC29/WG12, MHEG
colaitis@ccett.fr

Florence BERTRAND, Co-Editor of the MHEG Standard
bertrand@ccett.fr

France Telecom CCETT, 4 rue du Clos Courtel, BP 59,
35512 Cesson Sévigné Cedex, FRANCE.

Abstract. This paper presents the main concepts of the ISO International Standard/ITU-T Recommendation for Multimedia/Hypermedia objects, prepared by the ISO/IEC JTC1/SC29/WG12 known as MHEG, as well as examples of the use of MHEG in different communicating environments. It presents the different approaches in the multimedia/hypermedia domain, leading to different possible levels of standardization. Each level corresponds to a different balance between Genericity and Specificity or adequation to a given application domain. The level of MHEG is a compromise between genericity across various application types, and support for a minimum set of basic, essential functionalities. The requirements are introduced and expressed in terms of synchronization/linking between elementary units of information, real time interchange and presentation. The MHEG standard is then introduced, and the way the MHEG specifications meet the application requirements is discussed. The specification process is accompanied by experiments and prototype developments.

1. Introduction

This paper introduces the main concepts and field of applications of the emerging MHEG standard prepared by the Working Group 12 of ISO/IEC JTC 1/SC 29, known as MHEG (Multimedia and Hypermedia information coding Expert Group).

This presentation provides the reader with clear insights into the technical context of the MHEG standard (development of open multimedia applications and underlying requirements), and into the consequences of the application requirements on the design of a generic interchange format for multimedia/hypermedia information. In this perspective, several issues are discussed in this paper:

- ① Is there a need for a standard defining the representation and encoding of data structures (objects) suited to multimedia applications or services? and if yes, what are the required features for such objects?
- ② How the design of multimedia/hypermedia objects can meet the requirements of their different types of using applications?
- ③ How multimedia/hypermedia objects may be used in real applications?

2. Why a standard for multimedia/hypermedia Objects

2.1. General context of multimedia/hypermedia applications development

In recent years, an explosion of multimedia applications has been seen in many domains such as: education, training systems, office and business systems, information and point of sales systems, digital television, etc...

In all these three fields, multimedia interactive programs have been developed, providing their users with pictures, video sequences and audio sequences, presented in interactive situations.

This major trend can be explained as follows:

- a- The availability of multimedia resources on many computers. Compression standards, such as JPEG or MPEG have allowed the development of dedicated chips, installed on computer motherboards as basic system resources.
- b- The increasing availability of optical disks and digital transmission media such as ISDN, digital broadcasting networks and broadband telecommunication networks, which allow the exchange of very large amounts of data.

Many multimedia applications will be designed to run on heterogeneous workstations, or to be interconnected to offer a multimedia service: computer supported multimedia cooperative work, multimedia messaging systems, electronic publishing and electronic books, audiovisual telematic systems for training and education, simulation and games, sales and advertizing, TV guides and entirely new classes of multimedia applications.

The production of the large quantities of multimedia/hypermedia information necessary for the expansion of such services represents a significant investment, and it is vital that this information remains available in a world of rapidly evolving systems and technologies, and is not lost because of incompatibilities in the data structures supported by the applications. This is why a "common language" is required for all those multimedia/hypermedia communicating systems which will appear in the forthcoming years is required. An important issue remains: what "level" of interchanged information should be standardized? How to define the appropriate "unit of information" to be interchanged between heterogeneous multimedia/hypermedia systems?

2.2. Multimedia/hypermedia application model

An OSE (Open System Environment) reference model is proposed for heterogeneous multimedia/hypermedia technologies presently being developed. This model is composed of:

- a- application software (specific programs and data);
- b- application platform (the actual hardware and software);
- c- platform external environment (system elements such as communication services, peripheral devices, ...);

and interfaces between these entities:

- 1- API (Application Programming Interface) between the application software and the application platform (man/machine interaction services, information interchange services, communication services, internal system services);

- 2- and EEI (External Environment Interface) between the platform and the external environment (protocols and data formats).

This type of model, which basically consists in a clear separation between platform-dependent and platform-independent modules, is very commonly used in many delivery systems architectures (e.g CDI players, MHEG terminals as described in section 6, etc ...).

In this OSE model, a number of basic multimedia/hypermedia services are described as the sets of functionalities necessary for the support of multimedia/hypermedia applications. These services are summarized in figure 1.

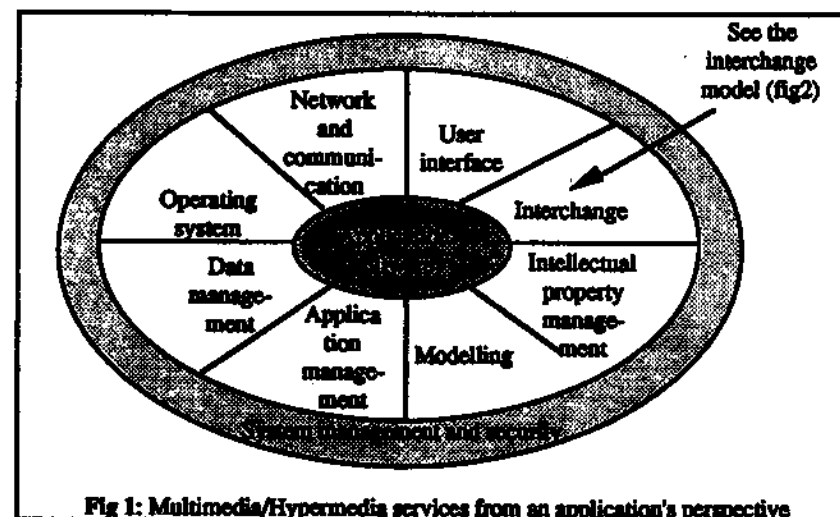


Fig 1: Multimedia/Hypermedia services from an application's perspective

The figure 2 shows the position of MHEG with respect to other hypermedia data interchanges:

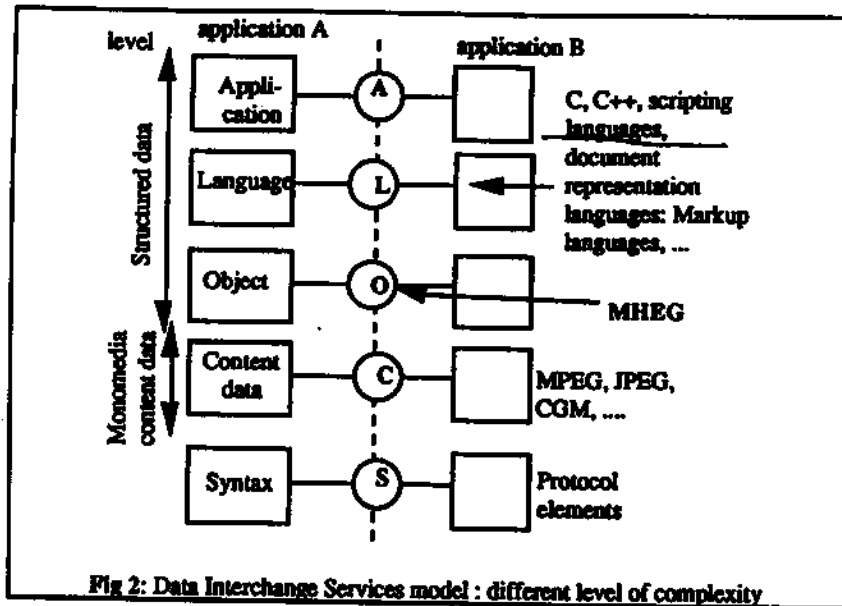


Fig 2: Data Interchange Services model : different level of complexity

In this figure, some of the data interchange levels correspond to existing or emerging standards: monomedia contents (JPEG, JBIG, MPEG, MIDI, CGM, ...), languages and scripts (international standards such as C, C++, SGML, HyTime and proprietary scripting languages such as Hypertalk, ToolBook, ScriptX, ...). The Object level defines representation for multimedia and hypermedia objects and is addressed by the MHEG standard.

From the above model, we see that there are various levels of data interchange between applications, according to the level of genericity which is targeted:

Genericity: If communicating applications interchange low level unstructured data, they do not need to share any high level representation of the information that they are handling. In this way, standards such as JPEG or MPEG define algorithms suited to data compression independently of the application domain.

No genericity: The interchange of high level structured data (e.g hypertext documents) assumes that communicating applications share the same representations and semantics of a hypertext document. The interchanged information is specific to a given context.

Future hypermedia applications will go far beyond the old limits between categories and will use cross-domain concepts and facilities, e.g multimedia document retrieval and consultation associated to hypertext navigation, groupware associated with multimedia messaging, training program plus direct interpersonal communication. It is essential to find the adequate unit of information communication to reconcile the required genericity with a correct level of structuration.

The MHEG committee has adopted the following methodology:

- 1- Start from the hypermedia application requirements and define the level of genericity required to meet the needs of a broad range of communicating applications;
- 2- Deduce the features of the generic structures from the application requirements.

2.3. Hypermedia application requirements

From the list of typical applications enumerated in paragraph 2.1, a number of underlying requirements can be identified:

- Portability in a multi-vendor environment, i.e the possible use of a wide range of terminals and workstations;
- Multimedia information, i.e to be able to group several monomedia entities into a single "container";
- Information structured in such a way that real-time interactivity, including acquisition of multimedia data, as well as real-time interchange can be ensured;
- Composition and Synchronization in space and time;
- Definition of links between data elements;
- Definition of interaction with the user;
- Reusability of data in other documents;
- Ability to update the data, and manipulate acts of data elements.

These requirements call for the design of structured entities called multimedia/hypermedia objects or MHEG objects in the standard, able to encapsulate a limited set of functionalities while remaining sufficiently generic to be usable by a range of applications:

- Standardization only at the level of monomedia information is not enough to guarantee applications portability, because applications do not use the monomedia data individually
- Monomedia information is not well suited to the design and interchange of hypermedia information
- The design of distributed hypermedia applications will be eased if the internal details of the information presentation are masked from the application.

The MHEG abstractions are communication oriented and are suited to:

- Real-time presentation
- Final form representation: the objects do not require additional processing of their structure.

3. The main concepts of the MHEG standard

3.1. Methodology

The object-oriented approach was chosen for the design of the standard because it fits the requirements of active, autonomous and reusable objects. The standard focuses on the generic structuring aspects of the objects.

The base representation (Part I of the standard) uses ASN.1, and the Part II of the standard will provide an isomorphic representation in SGML.

These alternate structure representations are equivalent, as shown in Figure 3.

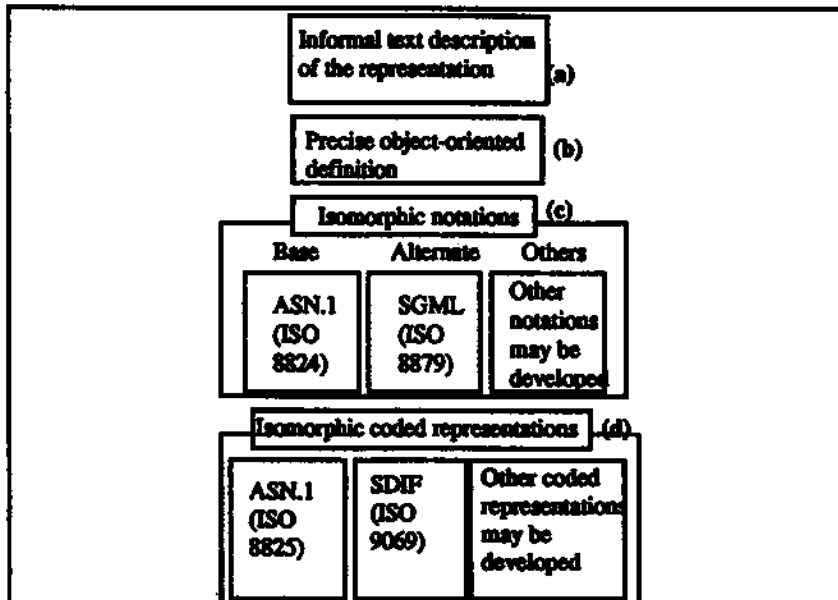


Fig 3: Methodology used to develop the MHEG standard

For each class, the standard provides (a) an informal textual description (for intuitive understanding), (b) an object-oriented definition expressed in a formal context free grammar, (c) a set of equivalent notations of the formal definition, and the corresponding encodings (d).

3.2. MHEG object classes

The MHEG standard defines classes of objects, instances of these classes will be interchanged between using applications. Figure 4, shows the MHEG inheritance tree:

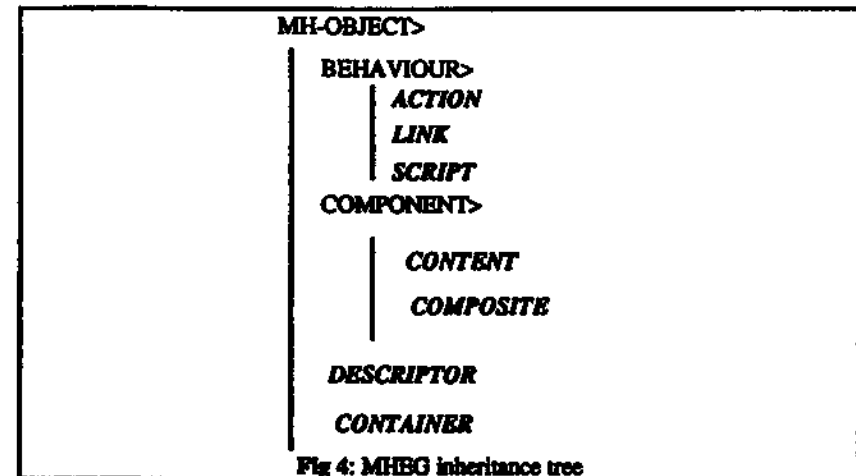


Fig 4: MHEG inheritance tree

MH-Object class. The standard provides the required information for the identification and addressing of MHEG Objects.

Action Class. The action class defines a set of actions to be applied to a specified target. Different types of actions can affect the following behaviours of an MHEG object, presentation and script instances:

- **Preparation:** controls the availability of the object in the system. For example, "Prepare" and "Destroy" actions may be applied to add and remove an object from the system.
- **Creation:** creation of presentation instances from a component object or script instances from a script object.
- **Presentation:** controls the progress of the presentation instances in the system. For example, "Run" and "Stop" actions may be applied to control the progression of a time-based presentation instance.
- **Rendition:** controls the rendition of the presentation instance on the system. These actions vary according to the media type, for example, "Set Speed" for time-based media and "Set Size" for visible media.
- **Interaction:** controls the results of interaction with a presentation instance in the system. For example, "Set Selectable" specifies the selectability of a presentation instance and "Set Modifiability" specifies the modifiability of a presentable.
- **Activation:** controls the activation of the script instances in the system. For example, "Run" and "Stop" actions may be applied to control the progression of a time-based presentation instance.
- **Getting value:** get the attribute, status or behaviour value of MHEG object, presentation instance and script instance. These actions are used to express the link condition in a link object.

Link Class. The link class specifies sets of relationships between a "source" and a list of "targets". The relationship is composed of conditions associated with the source and actions to be applied to the targets when the conditions are satisfied.

Script Class. The script class defines a container for *complex* relationships between objects, defined by a non-MHEG language.

Content Class. The content class contains or refers to the coded representation of monomedia information together with a parameter set which identifies the coding method, and application-oriented parameters such as fonts, colour table.

Composite Class. The composite class provides the support for associating multimedia and hypermedia objects. This mechanism provides a consistent approach to the synchronisation in time and space and linking of a set of objects. This class provides also the logical structure to describe the list of possible interactions offered to the user but does not define the interaction facilities provided by the user interface. Such interaction may be achieved in a variety of ways, for example, Graphical User Interfaces, keyboards, etc. This standard does not define the "look and feel" of multimedia interactive presentations, neither does it propose to change or add concepts to those that exist in typical Graphical User Interfaces. As this standard is generic and independent of platform and implementation, it describes interaction at a virtual level. It is for a using application to apply these mechanisms using its specific "look and feel".

Descriptor Class. The descriptor class defines the interchange of resource information about sets of interchanged objects and is used to facilitate installation and resources negotiation.

Container Class. The container class provides a container for regrouping multimedia and hypermedia data interchange.

3.4. Presentation and script instances

For the purpose of reusing component objects or script objects in different presentations or activations, a clear separation has been made between the MHEG content interchanged object which contains the original reusable media data and the presentable corresponding to a specific presentation of the original media data. The same distinction has been made between the MHEG composite object which contains the original arrangements of information and the tree corresponding to a specific presentation of the original arrangements. The same distinction has been also made between the MHEG script object which contains the original script data and the script instance corresponding to a specific activation of the original script data.

The presentation of a presentable or a tree does not affect the original component object, this allows the reuse of the same original component object in different presentation instances. The activation of a script instance does not affect the original script object, this allows the reuse of the same original script object in different script instances.

A component object is considered as an 'original' or 'model' object, any number of presentation instances may be created based on instructions given by the author. An instance of presentation made from a content object is called a presentable. An instance of presentation made from a composite object is called a tree. A script object

is considered as an 'original' or 'model' object, any number of script instances may be created based on instructions given by the author.

This standard defines an explicit action 'make' to create an instance of presentation from a component object or 'make_script' to create a script instance from a script object

This standard defines also an initial expected behaviour of the presentables, trees, and presentation instances it defines also actions that will modify this behaviour.

The internal representations of the presentable, tree and script instance are not defined by the standard, each MHEG engine will have its own internal representation technique. However, this standard defines the overall structure and the identification techniques to reference a presentation instance in order to modify its behaviour.

3.5. Schedule

Part I: "Information Technology - Coding of Multimedia and Hypermedia Information - MHEG object representation - Base notation (ASN.1)"
CD approved: November 93, Start of DIS ballot: Mid 94, Planned IS: 1994

Part II : "Information Technology - Coding of Multimedia and Hypermedia Information - MHEG object representation - Alternate notation (SGML)"
Planned CD ballot: End 94, Planned DIS ballot: 1995, Planned IS: 1995

Part III : "Information Technology - Coding of Multimedia and Hypermedia Information - MHEG-S : MHEG extensions for scripting language support"
Planned CD ballot: End 94, Planned DIS ballot: 1995, Planned IS: 1995

4. Uses of MHEG Objects in communicating environments

The standard defines an encoded format for the interchange between applications. Once interchanged, the MHEG objects are handled by the MHEG engine in an internal representation. There is only one representation defined in the MHEG standard for the interchange of objects, but there will be many possible MHEG engines, each with their own internal format.

The purpose of the MHEG standard is not the specification of the MHEG engine, however, the standard makes some assumptions on different processes that may exist in an MHEG engine.

4.1. The interchange process

The protocol used for the interchange is not described in the MHEG standard. The interchange process may have two sub processes: the formatter and the parser.

- When an MHEG engine in a using application sends MHEG objects to another using application, a formatter may be used to convert the internal format used in this mheg engine to the encoded format defined by the MHEG standard.

- When an MHEG engine in a using application receives an MHEG object, the object is decoded by a parser. The values within the object are passed to the MHEG engine which may convert them to an internal format.

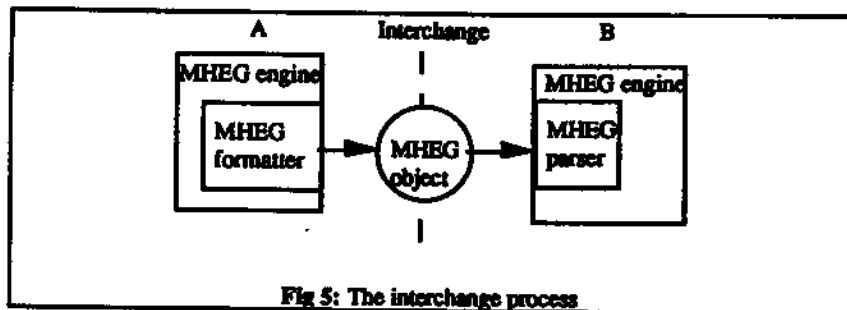


Fig 5: The interchange process

If the exchange between two using applications is bidirectional, the two interchange processes may have both a formatter and a parser.

4.3. The preparation process

This process prepares MHEG objects for processing by the MHEG engine. For example, retrieval of an audiovisual sequence content data from a disk may require so much time that it may be efficient to start loading it before it is needed.

Specific MHEG actions invoke the preparation process: *prepare* and *destroy*. The *prepare* action makes the object available to the MHEG engine, the *destroy* action removes the MHEG object from the MHEG engine.

4.4. The make process

This process make presentation instances or script instances from MHEG original model object and render them available for presentation or activation by the MHEG engine.

Specific MHEG actions invoke the make process: *make* and *kill*. The *make* action makes the presentation or script instance from the object model, and render it available to the MHEG engine, the *kill* action removes the presentation or script instance from the MHEG engine.

4.5. The presentation process

This process presents the presentation instances to the user.

The *run* action places the presentation instance under control of the presentation process, the *stop* action removes this control. The presentation status of a presentation instance in an MHEG engine shows whether the presentation instance is running or not running.

The *rendition* actions may change the rendering of the presentation instance to the user dynamically. Parameters may modify the volume of an audible media data, the size of a visible media data or the presentation of a selection item as a menu item or as a button.

An application may extend the actions with its specific renditions, e.g. colour, character fonts.

4.6. The interaction process

This process enables interactions with the user. The MHEG standard does not aim at defining the look and feel of multimedia presentation, it interworks with graphical user interface tools existing on the system. The interaction process is composed of two sub-processes: selection and modification of presentation instances.

4.7. The synchronization and linking process

This process enables the dynamic variation of presentation of a presentation instance, the spatial and temporal synchronization of a set of objects and the presentation of objects after an interaction. The information interchanged in action, link and script objects are handled in this process.

4.8. The activation process

This process activate a script instance. This may have user effect or not, depending of the content of the script data.

The *activate* action places the script instance under control of the activation process, the *deactivate* action removes this control. The activation status of a script instance in an MHEG engine shows whether the script instance is activated or not activated.

4.9. Interface between MHEG objects and using applications

The MHEG standard has been designed to allow simple relationships between an MHEG object and a user application. The interface between an application and an MHEG engine is not defined by the standard, it is defined by each MHEG engine. In the case of an object oriented MHEG engine, the interface is based on sending messages and receiving results.

Multimedia applications may manipulate MHEG objects at different levels of details, the object (as a whole or attribute per attribute). The standard does not define the structure of data passing across the interface.

However, the following facilities are suggested features which may be provided by an MHEG engine interface :

- Start, stop the MHEG engine
- Supply objects (or references) to the engine, accept objects (or references) from the engine
- Start, stop the processing of an object
- Facilities equivalent to those provided by the action class
- Facilities to access to data within the MHEG engine (attributes, statuses, ...)
- System management information (error indications, ..)

5. Examples of models of user applications

5.1. Model of an interactive telematic training service

The figure shows an example of an interactive telematic service in the educational environment described in CCITT Recommendation F.740.

It shows the possible actions of a student terminal which is in session with a central server, for example the school system, and is receiving MHEG objects. The student is studying a hypermedia document in an interactive manner, and as the MHEG engine analyses the responses of the student, it may request either further MHEG objects or the content data required by the objects it has already received from the server.

The formal responses of the student are returned to the server as MHEG objects which may in turn be received by the more sophisticated system used by the tutor. Note in that case the requirement to code these "answers" using the MHEG standard, since the tutoring and student equipments may be the subject of separate procurement policies.

The student system has the "look and feel" defined by the interaction process in conjunction with the graphical user interface.

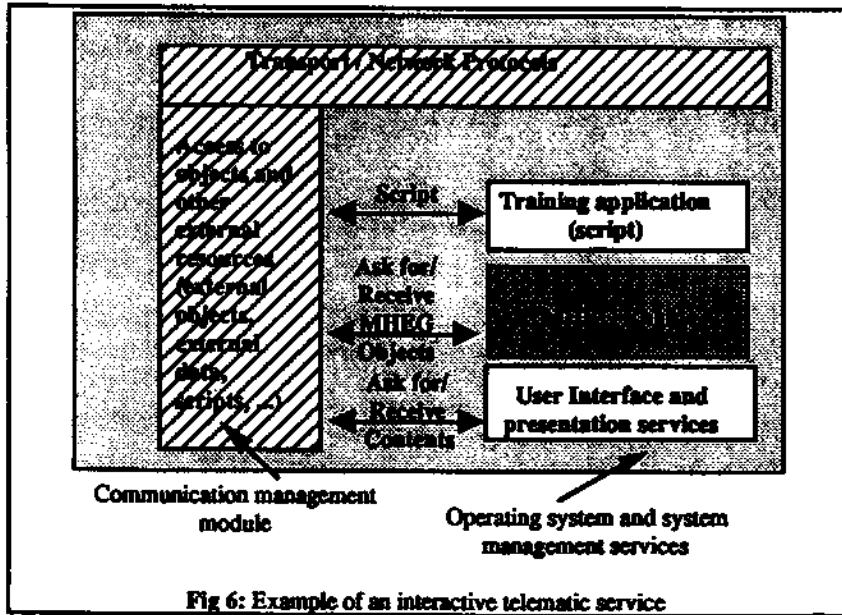


Fig 6: Example of an interactive telematic service

5.3. Example of an authoring application

The authoring application creates MHEG objects and relies on the services of the

MHEG engine for the dialog with the author, the allocation of identifiers to the objects and the exchange of MHEG objects.

In this generic functional model, the MHEG engine as well as the application can access a specific module to allocate unique identifiers to each MHEG object at creation time.

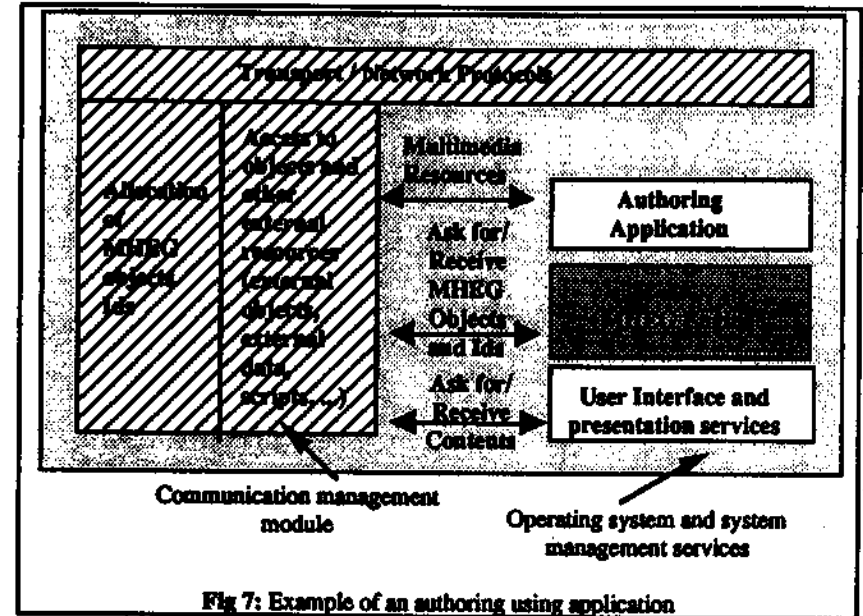


Fig 7: Example of an authoring using application

6. COMIS : An implemented example of the use of MHEG objects and MPEG video sequences

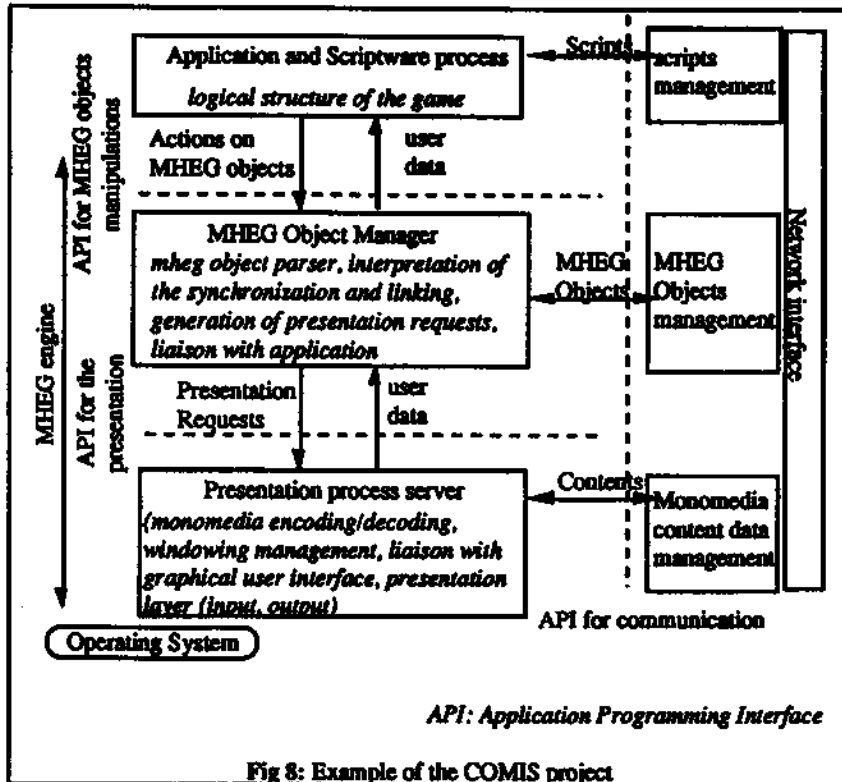
In 1992, CCETT participated in a CEC (Commission of the European Communities) ESPRIT project "COMIS" (Coding of Moving Images on digital Storage media) which developed an application showing the interworking of MHEG, JPEG and MPEG standards. This is an interactive game consisting of a tour of different capitals in Europe.

For this project, CCETT has developed an initial version of MHEG engine, integrated into a multimedia platform according to the software architecture outlined below:

The platform consists of three software components, each of which corresponds to a logical layer and implemented as a separate Windows application, which communicates by messages with its neighbouring layers:

- The windowing presentation server: manages windowing resources, deals with user-generated events, displays visual and auditive information.

- The MHEG object manager: accesses, decodes and analyses MHEG objects, interprets the associated methods as series of presentation requests or feedback answers.
- The game scriptware: executes the algorithmic structure of the scenario.



In a communicating version, the voluminous MPEG video sequences are located in a remote server. After a user selection, the interchange process of the MHEG engine accesses the corresponding video sequences via a broadband network for real-time presentation.

Following the achievement of the COMIS project, the MHEG engine has been continuously improved to stick to the progress of the MHEG standard specifications. A complete toolkit, composed of an Object Editor, a MHEG Class Library, an ASN.1 Codec, and an Interpreter, is now available in CD version.

7. Other ongoing projects

Since the beginning of the MHEG standard development, the MHEG partners have been collaborating in experimental programs based on interchange and validation of

test objects and pilot implementations.

By the end of 1993, the MHEG specifications have stabilized, allowing for more significant developments and full size experiments. Early products and licensing announcements are expected as the MHEG standard progresses to the DIS and IS stages.

A European initiative funded by the EEC, European Economic Commission, known as "OMHEGA" (Open MHEG Applications) has been launched at the beginning of 1994, with the following objectives:

- Elaboration of a generic architecture and API specifications by a set of industry partners and telecommunication operators, to be available and widely published as the reference architecture of a MHEG system;
- Design and realization of MHEG prototypes;
- Promotion of the MHEG standard through regular information exchange, conferences.

Other european projects, using MHEG objects in Video-On-Demand applications, and in multimedia TV guides for digital television broadcasting, have also begun in January 1994.

8. Conclusion

The MHEG standard is a key to the development of communicating multimedia applications and services. The standard supports the interchange of multimedia information, and it specifically meets the needs of applications which run under the following constraints:

- Need for multimedia synchronisation between the components of a single and easily addressable "composite" object;
- Need for specific structures to support interactivity with the end-user;
- Need for real time interchange, through simple and efficient mechanisms to optimize interchange of composite objects;
- Final form representation of information, without additional processing needed to restructure the information.

Acknowledgments

The authors wish to thank all the MHEG members for their contribution to this international effort.

References

1. Information Technology - Coded Representation of Multimedia and Hypermedia Information Objects (MHEG), Part I : Base Notation (ASN.1), ISO/IEC CD 13522-1.
2. Colaitis Françoise, Kretz Francis: "Standardizing Hypermedia Information Objects", IEEE Communications Magazine, May 1992.
3. Colaitis Françoise: "MHEG, an emerging standard for interchange of multimedia and hypermedia objects", ICC'93 Conference and Proceedings, Geneva, May 23-26, 1993.
4. Price Roger: "An introduction to the future MHEG international standard for Hypermedia Object interchange", ACM 93 Conference and Multimedia Proceedings, Anaheim, August 93.