

Elementary Stream Management in MPEG-4

Carsten Herpel

Abstract—The forthcoming MPEG-4 standard specifies in its systems part an audiovisual scene description and functionality for the elementary stream management. The elementary stream-management functionality is introduced here. It consists of a media object description framework that describes the streaming resources that form part of an MPEG-4 presentation and of a synchronization syntax incorporated in a flexible sync layer with an underlying systems decoder model. The final section outlines the transport and session setup for MPEG-4 presentations on relevant transport media, namely, the Internet and in digital broadcast scenarios.

I. INTRODUCTION

ISO/IEC JTC1 SC29 WG11, better known as MPEG (Moving Pictures Experts Group), is currently in the process of finalizing the next MPEG standard, called MPEG-4, which will formally be labeled ISO/IEC IS 14496. The MPEG-4 standardization process reached final committee draft level in May 1998. This is a major milestone and therefore a good opportunity to highlight some major elements and properties of this forthcoming standard.

The major design goal of MPEG-4 is to define a framework for the joint description, compression, storage, and transmission of natural and synthetic audiovisual data. Similar to previous MPEG standards, MPEG-4 defines improved compression algorithms for audio and video signals, including specialized tools, e.g., for speech coding. In addition, means to efficiently describe synthetic audio and graphics exist. Again, specialized tools, e.g., for the animation of synthetic faces, are included. The data streams that correspond to such time-variant signals are transmitted or stored separately and are only composed into an integrated audiovisual presentation at the receiver. The format for the description of how the individual elements of a presentation are related in space and time is also defined by MPEG-4. This description is object oriented and identifies each semantically meaningful audiovisual entity as a media object.

The MPEG-4 standard comes in several parts. Compression techniques for visual and audible media objects are specified in parts 2 [2] and 3 [3] of MPEG-4, respectively. Part 1 [1] has itself two major topics: the definition of the binary scene description, which has its roots in VRML97 [4], and the management of the elementary streams that convey the coded representation of audiovisual data. Other parts of MPEG-4 contain reference software (part 5) [5] and conformance

procedures (part 4). The latter will be about one year delayed from the other parts. Part 6 [6], the delivery multimedia integration framework (DMIF), defines interfaces to arbitrary transport networks; hence, it standardizes transport service features that are available for MPEG-4 as well as other applications.

This paper surveys one of the major topics of MPEG-4 Systems, the management of elementary streams. Section II introduces the concept and content of object descriptors that serve to identify a related set of elementary streams. Section III describes how the synchronization of MPEG-4 data is taken care of in the context of the system decoder model and the corresponding sync layer syntax. Sections IV and V describe how it is envisioned to transport MPEG-4 elementary streams in some relevant transport layers and how to actually gain access to MPEG-4 content in a session setup process, respectively. Section VI contains some conclusions and description of future work.

II. MEDIA OBJECT DESCRIPTION FRAMEWORK

An MPEG-4 scene consists of any number of visual and audible media objects. Some of them are of a static nature, like background images or audio clips (jingles), while others (video, audio) are time variant, requiring continuous streaming data to convey them unless the time variance is deduced programmatically. Data for each streaming media object are conveyed within one or more elementary streams. Static images and finite audio clips are also conveyed in separate streams. This allows efficient interleaving of different streams for transport, taking into account their real-time constraints.

A major task of the MPEG-4 Systems architecture is to describe the relations of the various audiovisual components of an MPEG-4 scene. Fig. 1(a) shows an example scene. The description is done at two levels. On a structural level, it must be known how the content, the media objects, are arranged in space and time. This scene description is accomplished by the binary format for scene (BIFS), which is not further detailed in this paper. On a lower level, it is only of interest how the various streams that contain the compressed MPEG-4 data relate to each other, how they are configured and synchronized, and where they are located. In addition, some information about the content item, including intellectual property information, is desirable. This resource description is accomplished by the media object description framework.

Content and resource descriptions are separated to ease content management. While the content description will be generated by content authors, the stream resources may be independently relocated by resource-management tools. The link between both descriptions is established by object de-

Manuscript received October 1, 1997; revised November 1, 1998. This paper was recommended by Guest Editors H. H. Chen, T. Ebrahimi, G. Rajan, C. Horne, P. K. Doenges, and L. Chiariglione.

The author is with Corporate Research, Thomson Multimedia, Hannover 30625 Germany.

Publisher Item Identifier S 1051-8215(99)02335-6.

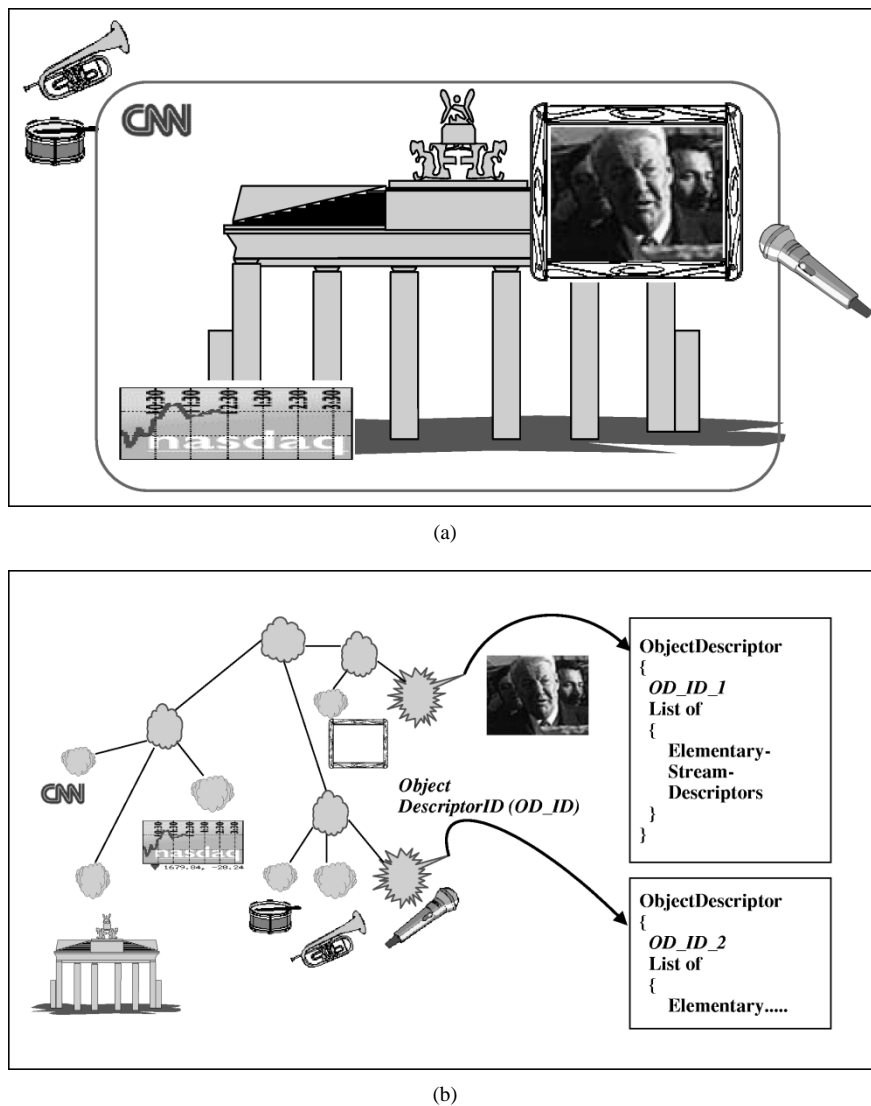


Fig. 1. (a) An example scene and (b) its symbolic representation as a scene tree (simplified) plus object descriptors.

descriptor ID's [Fig. 1(b)]. Both descriptions are highly dynamic, allowing time-stamped changes of object descriptors as well as of the scene description.

The components of the object descriptor (OD) and their transport encapsulation are introduced below. The section is concluded by a discussion of the usage of OD's.

A. OD Components

An object descriptor groups all descriptive components that relate to a single media object, e.g., a video object, an animated face, or an audio object. It is labeled by an ObjectDescriptorID. In particular, an OD contains a list of elementary stream descriptors that point to one or more streams with data or side information for this object (Fig. 2). Additional descriptors may carry additional information that refers to the whole object. Some of the individual descriptors used in this context are briefly described in the subsequent subsections. Instead of providing information about the media object in-place, the object descriptor may only contain a URL that points to another object descriptor at a remote location.

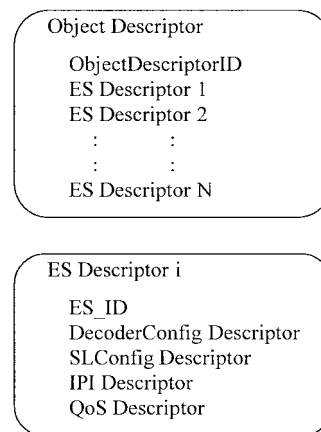


Fig. 2. Essential parts of an object descriptor and its components.

1) *Elementary Stream (ES) Descriptor*: Each ES descriptor refers to a single elementary stream and is identified by a numeric ES_ID or, alternatively, by a URL. ES_ID's allow locating of this stream by number. This is only possible within

a specific name scope. URL's are more flexible and allow referring to streams by a unique name. See Section II-D for further usage details.

A number of subdescriptors that are introduced below are part of the elementary stream descriptor.

2) *DecoderConfig Descriptor*: This descriptor contains all information that is needed to select and initialize a media decoder for this elementary stream. Since an elementary stream descriptor may also describe control streams with object-related information flowing upstream, in fact, it might also be the configuration of the associated control command *encoder* that is described here. Note, however, that back-channel signaling issues are not yet finalized.

The descriptor includes an indication of a predefined profile and level and a stream type, as well as information about basic traffic parameters of this stream, like average and maximum bitrate.

MPEG-4 defines object profiles that relate to individual media objects, defining the permissible tools and parameters of the compression scheme for this object. At this writing, the details of the MPEG-4 profile architecture are still being discussed, especially whether an object profile is to be attached to an individual stream rather than a media object.

The stream type identifies just the most basic aspect of the stream, e.g., visual stream, audio stream, scene description stream, object descriptor stream, clock reference stream, or object content information stream. An important task of the DecoderConfig descriptor is to carry additional decoder-specific configuration information that is passed on to the media decoder selected by stream type and profile and level indication.

3) *SLConfig Descriptor*: Similar to the previous descriptor, the SLConfig descriptor carries configuration information, this time for the sync-layer (SL) parser for this elementary stream. The SL and the elements of the SLConfig descriptor will be described in more detail in Section IV.

4) *Intellectual Property Identification (IPI) Descriptor*: The IPI descriptor is a vehicle to convey standardized identifiers for content like international standard book number, international standard music number, or digital object identifier if so desired by the content author. If multiple media objects within one MPEG-4 session are identified by the same IPI information, the IPI descriptor may consist just of a pointer to another elementary stream, using its ES_ID, that carries the IPI information.

5) *Quality of Service (QoS) Descriptor*: The QoS descriptor aims to qualify the requirements that this elementary stream has on the QoS of the transport channel for this stream. The most obvious QoS parameter is the stream priority and a possibility to signal predefined QoS scenarios, most notably "guaranteed" and "best effort" channels. Apart from that, unfortunately, it remains difficult to agree on a universal set of QoS parameters that is valid for a variety of transport networks. Therefore, a generic set of QoS_Qualifiers has been adopted that needs to be defined in the scope of specific applications.

It should be noted in this context that a QoS descriptor at the ES level rather than on the transport level is only

of interest in an interactive application, where the receiving terminal may select individual elementary streams based on their traffic (signaled in DecoderConfigDescriptor) and QoS requirements and the associated communication cost. It is also of interest in heterogeneous networks, if a network bridge shall be made aware of the requirements of individual elementary streams.

6) *Extension Descriptor*: A generic extension mechanism has been defined for future use by ISO or specific applications. It allows arbitrary descriptive information to be associated to the media object.

B. Auxiliary Object Information

Apart from grouping media streams, an object descriptor allows one to point to auxiliary streams associated to this media object. For example, semantic information about the content of the media object is conveyed by means of an object content information (OCI) elementary stream. An individual OCI stream may be associated to every media object, or it may be shared by multiple objects. It carries keywords, a textual description, language, and content rating information as well as creation dates and names of the content and OCI authors. OCI carried in a stream may be dynamically updated along with the content itself. Alternatively, single OCI descriptors embedded in an object descriptor or elementary stream descriptor can be used to convey static information like the language of an audio object.

It may be noted that OCI is a first step toward MPEG-7 that is going to provide much more extensive semantic information about media content in a standardized format.

Another example of very important auxiliary information is the optional clock reference stream that may be used to convey time base information (see Section III).

C. Transport Encapsulation of Object Descriptors

The knowledge of the scene description and the associated set of object descriptors is a prerequisite to access an MPEG-4 presentation. Care has been taken to ensure that object descriptors and scene description (BIFS) are easily identified and separated out of the complete MPEG-4 presentation. In particular, both are transported in individual elementary streams. For increased flexibility, there may be more than one elementary stream dedicated to either OD or BIFS transport.

A lightweight protocol has been defined to encapsulate object descriptors in messages. They allow for update or removal of a set of object descriptors or individual elementary stream descriptors at a specific point in time. The time stamp is associated with such a message on the sync layer (see Section III).

In unicast applications, it is recommended to use reliable transport channels for both OD's and scene description. In multicast applications, these are the data that need to be periodically repeated or conveyed out of band to enable random access to the multicast MPEG-4 session. In that respect, these data are comparable to program-specific information (PSI) or service information (SI) in MPEG-2 applications.

D. Usage of Object Descriptors

1) *Usage of Object Descriptor Information:* A set of object descriptors constitutes a description of the resources needed to access and process the media objects that form part of an MPEG-4 presentation. Access to an elementary stream is gained through either ES_ID or a URL. Both are symbolic names that need to be resolved to the actual transport channel that conveys the streaming data. The way this is done is specific to the underlying transport layer. For an example, see Section V.

ES_ID's are unique only within their scope, which is defined by the set of object descriptor streams that are themselves pointed to from a single object descriptor. Reference to a stream outside this scope is only possible by using a URL. This allows for an integrated description of content that originates from various locations but is intended for concurrent presentation.

Note that this descriptive framework deliberately ignores questions of how to treat delays induced by establishing access to various potentially distributed resources. It is assumed that with the evolution of the Internet and other suitable transport networks this issue will be solved.

As mentioned before, each object descriptor groups one or more elementary streams that are related to a single media object. These streams can be alternative representations, e.g., at different bitrates or resolutions, or they may have a hierarchical dependency, indicating a scalable encoding of a media object. The processing resources required for each stream are described by the respective DecoderConfig descriptors.

The evaluation of this decoder configuration, stream priorities, bitrate requirements, and the dependencies signaled in the ES descriptors allows the receiving terminal in an interactive application to request the delivery of a meaningful subset of streams for each media object, so that the computational resources of the terminal are not exceeded. Instead of evaluating individual entries of the object descriptors, the profile and level information present in each ES descriptor may be used. Apart from these resource-driven considerations, of course, the terminal needs to evaluate the scene description in order to decide which subset of the media objects advertised in an object descriptor stream it needs to process.

2) *Usage of Multiple Object Descriptor Streams:* An even more powerful grouping mechanism exists by distributing object descriptors for different subsets of media objects within a single presentation into separate elementary streams (Fig. 3). It is even possible to distribute the descriptive information for a single media object. As an example, one OD stream may be created that advertises ES descriptors with low-resolution versions of each media object of a presentation while a second OD stream conveys additional ES descriptors pointing to higher resolution versions of the same set of media objects (Fig. 4).

Note that the structural description of the content, the BIFS, also supports a distributed and scalable scene description conveyed in multiple streams, which is of course needed to exploit the flexibility offered by multiple object descriptor streams.

The grouping of descriptive (OD and BIFS) streams as well as associated data streams at the elementary stream level

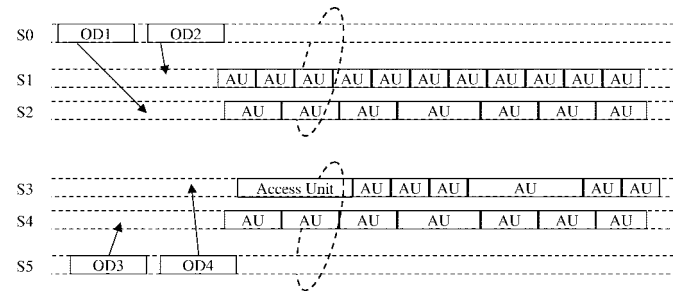


Fig. 3. Grouping media objects by using multiple OD streams.

allows, for example, for the transport of such stream groups on different transport links using different quality of service. In the example in Fig. 4, if it is assumed that streams S0, S3, and S5 carry the basic information of the presentation, they would have better QoS than the remaining streams.

While the logical groupings of streams are given by the list of ES_ID's, which are referred to from a given object descriptor stream, this is not sufficient for a content agnostic delivery layer to manipulate such stream groups. The grouping must be made visible, for example, by assigning different priorities to different stream groups. Such priority information is already present in the ES descriptors and may be attached to the lower layer transport packets if the chosen transport network supports this.

In that case, it is feasible for an MPEG-4 agnostic gateway to identify more or less important stream groups and to discard lower priority streams if necessary. Furthermore, if streams and their descriptive information are grouped according to the resolution or complexity of the encoded representation of the media object, this may correspond to a grouping for specific MPEG-4 terminal profiles. Therefore, a server can easily choose the right subset of streams for a given client terminal.

3) *Multicast Considerations:* The above-mentioned grouping of elementary streams is most relevant for multicast applications, since in point-to-point applications, it would also be possible to negotiate the desired subset of elementary streams between client and server.

In general, multicast, or broadcast, applications are characterized by the fact that clients may randomly tune into a running presentation. To accommodate this, usually all the crucial information for configuration of elementary stream decoders is periodically repeated. Such a classical (e.g., MPEG-2-like) scenario is depicted in Fig. 5(a) and shows that each stream follows its own schedule for inserting some configuration information.

In MPEG-4, the object descriptors convey the basic information that is required to gain access to and be able to process the media streams that form part of an MPEG-4 presentation. The mentioned configuration information is part of the DecoderConfig descriptor and therefore is automatically repeated if the OD's themselves are repeated, which is, of course, necessary in a multicast application as shown in Fig. 5(b).

Even in a multicast scenario, it is recommended to convey the object descriptor on a rather reliable channel, which has the side effect that crucial decoder configuration data might be received more reliably than the media data.

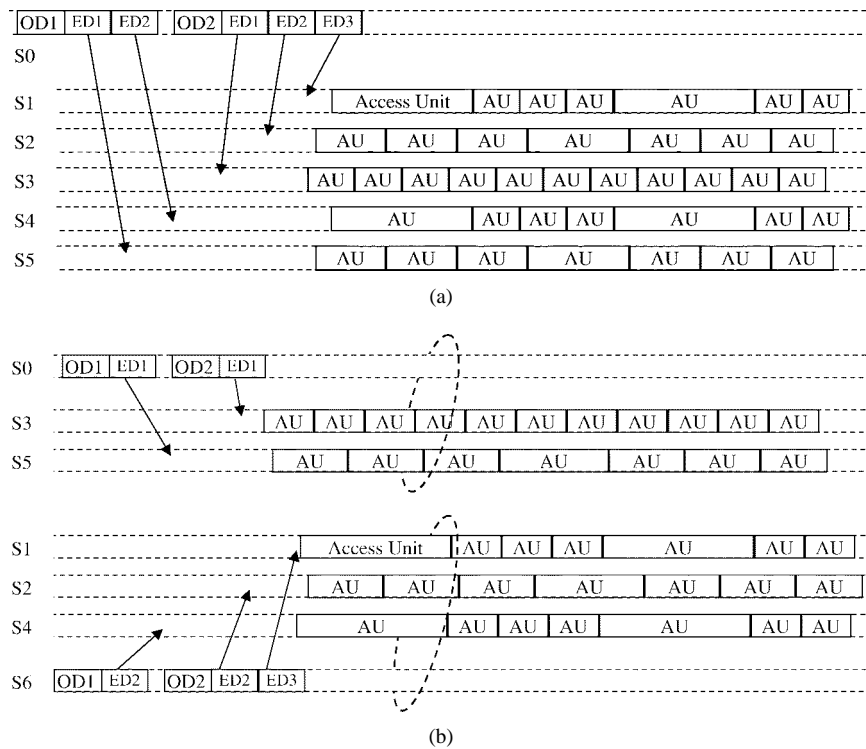


Fig. 4. Alternative representations ED1, ED2, ED3 (e.g., bit rate) for (a) a set of media objects OD1, OD2 conveyed in streams S1-S5 and (b) their grouping using multiple OD streams.

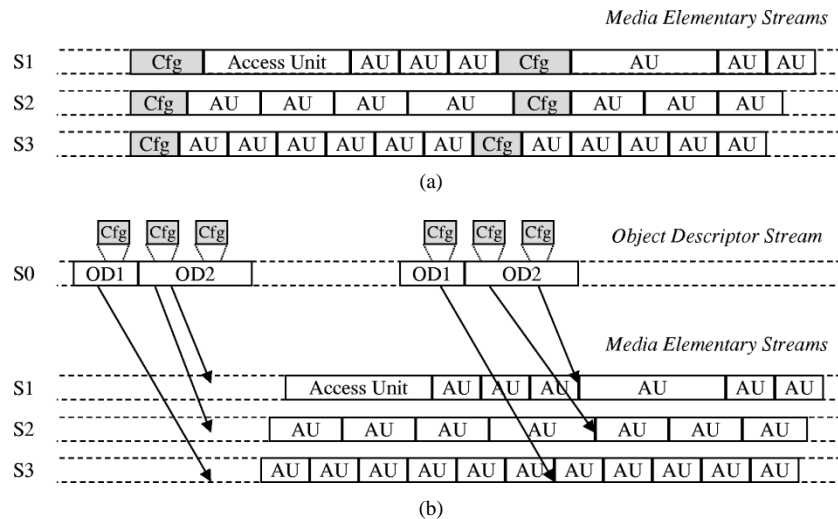


Fig. 5. Random access in (a) MPEG-2 using repetition within media streams and (b) MPEG-4 using repetition within OD stream.

III. SYNCHRONIZATION OF ELEMENTARY STREAMS

The synchronization of elementary streams is accomplished by the well-known concepts of time stamps and clock references, as used, for example, in MPEG-2 systems [7]. These concepts are summarized in Section III-A on the system decoder model, which models buffer and timing behavior of an ideal MPEG-4 terminal. Subsequently, the syntax to convey timing information, the sync layer, is introduced.

A. The System Decoder Model

A system decoder model (SDM) is used to specify the behavior of a receiving MPEG-4 terminal (Fig. 6). Different from MPEG-1 or MPEG-2, the SDM receives individual elementary

streams through a stream multiplex interface (SMI). There is no mandatory single protocol stack below the SMI that accomplishes the multiplexed transport of elementary streams. MPEG-4 has requirements on the end-to-end delivery of data through the stream multiplex interface, most prominently, a constant end-to-end delay. It is considered a task for the transport layer below the SMI to guarantee this constant delay by suitable means.¹ The SMI is not a complete application programming interface (API) specification whose functionality must be implemented in every MPEG-4 terminal for access to elementary streams. However, if it is exposed, such an inter-

¹It has been acknowledged that this is not realistic for the Internet as is. Therefore, a relaxation of this model is considered for such transport networks.

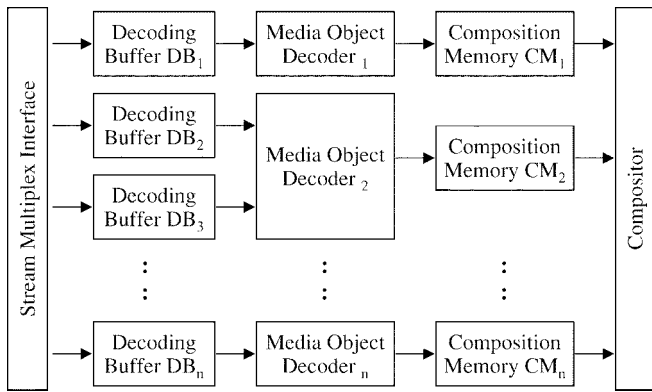


Fig. 6. System decoder model.

face must be able to deliver packets of data plus an indication of the packets' length. Additionally, reliable flagging of errors is expected.

The delivery multimedia integration framework (DMIF), which is specified as part 6 of MPEG-4, proposes a specific embodiment of such a stream multiplex interface, called DMIF application interface (DAI), that satisfies these minimal requirements and incorporates additional control functionality.

The system decoder model consists of the stream multiplex interface, a number of decoding buffers, media object decoders, composition memories, and the compositor.

The syntax of the SL, described in Section III-B, allows one to packetize elementary streams into access units or parts thereof. Such SL-packetized elementary stream data enter the decoding buffers through the SMI. Access units (AU's) are the smallest entities to which timing information can be associated, while the semantic meaning of access units is opaque at this layer.

The decoding buffer serves to store access units of elementary stream data until the decoding time. At that point in time, the SDM assumes instantaneous decoding of the access unit, removal of the access unit data from the decoding buffer, and appearance of the decoded representation of the access unit in the associated composition memory. With this model, it is possible for the encoding terminal to know how much decoding buffer resources are available for a specific stream at any point in time at the receiving terminal.

Media object decoders may be connected to multiple decoding buffers in case of hierarchically encoded media objects, but only to one composition memory for the decoded data. The decoded data are grouped into composition units. The relation between access units and composition units need not be one to one but is assumed to be known for each specific decoder. Each composition unit of decoded data is available for composition starting at an indicated composition time, known either implicitly or through an explicit composition time stamp, and ending at the composition time of the subsequent composition unit. The amount of composition memory required for specific media objects is currently not modeled by the SDM.

Only a minimum of assumptions is made on the compositor. This includes that the compositor instantaneously samples the content of each composition memory of the MPEG-4 terminal.

B. Transport of Timing Information—The Sync Layer

A flexible syntax has been created that describes all relevant properties of access units and allows one to map integer or partial access units into a lower layer protocol. This layer is designed under the assumption that the lower layer is responsible for framing the created data units, called SL packets.

The flexibility of the syntax is achieved by means of the SLConfig descriptor, a part of the ES descriptor that exists for each elementary stream. This SLConfig descriptor allows one to select the length of many syntax fields according to the requirements of this individual stream. For example, a very low-bitrate audio stream may require time stamps that consume few bits or omit them altogether, while a higher bitrate video stream needs very precise decoding time stamps. This implies that an MPEG-4 media encoder is aware of the subsequent SL encapsulation and in many implementations will probably directly create SL-packetized media streams.

Each SL packet may contain a sequence number to discover lost SL packets, a clock reference time stamp, and various flags to indicate presence of padding, the boundaries of access units in the packet, and the possibility of random access. SL packets that contain the beginning of an access unit may have additional information about this AU, namely, decoding and composition time stamp as well as the length of the AU or the instantaneous bitrate of this stream.

Clock reference time stamps are special in that they convey the speed of an encoder's clock. In many applications, however, multiple encoders will actually use the same clock. Therefore, referencing another ES that conveys the clock reference for the current stream is possible. It is even possible to create an elementary stream for the sole purpose of conveying clock references and no further media payload in order to separate the two functionalities completely.

IV. TRANSPORT OF ELEMENTARY STREAMS

In general, MPEG decided not to venture too much into the domain of multiplexing, since during the initial phase it became clear that an abundance of digital multiplexing infrastructure is already available, most notably based on MPEG-2 as well as the Internet protocols. Therefore, MPEG proposes to specify the small adaptations necessary to define the encapsulation of MPEG-4 SL-packetized elementary streams in such protocols.

In some environments, the MPEG-defined FlexMux tool described subsequently may be of value to reduce multiplex overhead or delay. With the exception of FlexMux streams, one elementary stream is mapped to one transport channel. This section discusses approaches to the adaptation of MPEG-4 streams to such transport channels on the example of the Internet and MPEG-2 transport environments. The last example discusses storage formats that are seen as conceptually very similar to transmission environments.

All such adaptations eventually require standardization within the community responsible for a transport network, e.g., the IETF in the case of the Internet. In general, identifiers for MPEG-4 presentations as such are needed, but also,

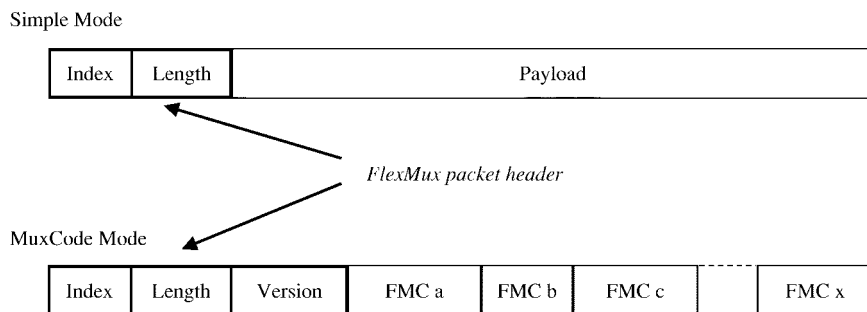


Fig. 7. Structure of a FlexMux packet.

specifically, a format for the mapping between ES_Id values and the number of their transport channels is required in order to locate the streams. Payload format specifications are needed to describe how SL packets would fit into the packets of the transport network.

A. The FlexMux Tool

The FlexMux is a simple syntax created for low-bitrate, low-delay streams, such as, for example, object descriptor, scene description, animation, or speech streams. It is not regarded as a true transport-level protocol but rather as a tool that should be used if the cost in terms of management load, overhead, or delay to set up transport channels for each individual elementary stream would be too high. This may be the case if a presentation consists of dozens of objects.

The two variations of FlexMux packet structure are depicted in Fig. 7. In Simple Mode, the header consists of an Index that corresponds to the FlexMux channel, or stream number, and the packet length in bytes, both 8 bit, limiting the number of streams and payload size to 256 each. The MuxCode mode is active for $\text{Index} \geq 240$. It further reduces the overhead by conveying *a priori* how the payload of a single FlexMux packet is shared among multiple streams. This mode comes at the cost that each of the 16 possible templates, called MuxCode table entries, needs to be conveyed before it can be used. To maintain correct state and to discover transmission errors, a version number is used that must match between the current packet and the MuxCode table entry.

The sequence of FlexMux packets that flows through one transport channel is termed FlexMux stream. The protocol that sets up this transport channel must also provide a means to convey the MuxCode table entries.

B. Transport on Real-Time Transport Protocol (RTP)

The RTP [8] is evolving as one of the major protocols for transport of real-time data on the Internet. It does not address issues of organizing real-time content into sessions or their control. Such functionalities are addressed by accompanying protocols that are still works in progress [9], [10].

Note that transport of MPEG-4 data on hypertext transfer protocol (HTTP) is not considered here, as HTTP is deemed to be a download protocol rather than a real-time transport protocol. Of course, however, a self-contained “MPEG-4 file,” as described in Section IV-D, could be conveyed via HTTP.

1) *Single Stream Encapsulation*: Both RTP and the MPEG-4 SL syntax serve the major purpose of labeling a payload with a time stamp and enabling loss detection through sequence numbers. While RTP has been designed for computational efficiency by respecting byte, word, or even 32-bit boundaries for its syntax elements, the SL syntax is focused on flexibility and coding efficiency. Both expect the payload to be a semantically meaningful application data unit to facilitate stream-specific loss recovery according to the application-level framing (ALF) principle [11]. Therefore, wherever possible, one elementary stream should be mapped into one RTP stream. In this case, one SL packet will be conveyed per RTP packet. The redundancy between the RTP time stamp and sequence number fields and the SL packet header elements `decodingTimeStamp` and `sequenceNumber` can be reduced by removing them from the SL packet header. This, however, requires processing of the SL packets during the RTP encapsulation process.

2) *FlexMux Stream Encapsulation*: Since an MPEG-4 presentation may well consist of dozens of media streams, an additional mapping of multiple elementary streams into one RTP stream is proposed, using FlexMux. Both the overhead for individual RTP streams but even more the management load induced by the parallel RTP control protocol streams and by the increased number of lower-level (Internet protocol) protocol packets suggest that the ALF principle should optionally be ignored in this case by bundling multiple MPEG-4 elementary streams in one RTP session. This is achieved by mapping a complete FlexMux stream in one RTP session. The encapsulation process becomes simpler in this case, since only the fixed FlexMux packet header needs to be parsed to ensure that an integer number of FlexMux packets are put in each RTP packet. This, of course, comes at the cost that time stamps and sequence numbers can no longer be shared between SL and RTP layers. Another disadvantage that needs to be considered is that loss of one RTP packet might result in lost data from multiple elementary streams.

C. Transport in MPEG-2 Transport Streams

MPEG-2 transport streams (TS's) provide the transport encapsulation for real-time digital broadcast data. The framework includes as well all necessary descriptors (PSI) to identify the services communicated within a transport stream. This section focuses on the several options to encapsulate MPEG-4 streams

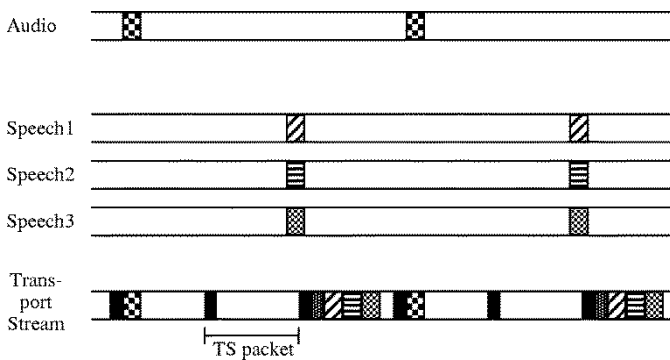


Fig. 8. Multiplexing MPEG-4 elementary streams in an MPEG-2 TS using single stream or FlexMux stream mapping to a PID.

in an MPEG-2 transport stream, while the description of and access to MPEG-4-based services is addressed in Section V.

1) *Single Stream Encapsulation*: A single SL-packetized stream may be mapped to a sequence of transport packets labeled by a specific packet identifier (PID). The TS packets have a fixed size. Since SL packets have variable size, some means of adaptation is needed. The MPEG-2 packetized elementary stream syntax could be used; however, it is not very efficient, since each SL packet would have at least a 6-byte overhead. Instead, each SL packet may be prefixed by a 1-byte length indication. A pointer mechanism may be used to indicate the start of the first SL packet within a transport packet, in the same way as is done for PSI sections.

This method may be inefficient if an SL-packetized stream has a rather low bitrate. To minimize the transmission delay, the sender may need to send SL packets before an amount of 183 bytes has been collected, which is the payload length of a transport packet minus 1 byte for the mentioned pointer. This results in application of padding, i.e., bandwidth waste. Such a situation is illustrated in Fig. 8.

Fig. 8 shows a visualization of a number of audio and speech objects and how these objects produce SL packets of data over time. These SL packets are multiplexed in a sequence of transport packets shown at the bottom. The data packets of the single audio stream on top occur at regular intervals but have much less than 183 bytes in length. To keep the delay low, each audio SL packet is immediately put in a transport packet with a given PID, and padding is applied.

2) *FlexMux Stream Encapsulation*: To address the previously mentioned inadequacy, a FlexMux stream can be mapped to a PID in a similar way. FlexMux is particularly useful if the data production for the streams multiplexed into the FlexMux stream is well correlated in time, as indicated for the three speech streams in the middle of Fig. 8. The bandwidth waste can be reduced enormously using FlexMux MuxCode mode.

First, one SL packet of each stream is concatenated in one FlexMux packet, i.e., only one FlexMux packet header is needed for all three SL packets. Then each FlexMux packet is put in one transport packet (with a different PID than the audio stream, of course), leading to much less padding compared to the transport packets of the single audio stream. Again, the pointer mechanism mentioned above could be used

to indicate the start of the first FlexMux packet within the transport packet, if it is desired to have them unaligned to TS packet boundaries.

3) *Digital Storage Media—Command and Control (DSM-CC) Data Carousel Encapsulation*: The DSM-CC data carousel specification is intended for embedding download data into MPEG-2 broadcast applications. It supports the multiplexing of several data entities, termed modules, in one MPEG-2 TS PID. Originally, these modules were supposed to be time invariant or slowly changing, as would be the case for digital teletext, program guides, or download applications in interactive TV services. The modules would be regularly repeated in order to allow random tune-in.

In the meantime, it has been proposed to extend this specification to support the delivery of lower bitrate real-time data. The advantage of this approach is that the multiplex of several SL-packetized elementary streams into one PID would be feasible in this framework; however, it would be at the price that each SL packet to be encapsulated may be preceded by a message header that has a length of 18 bytes.

A practical advantage of DSM-CC data carousel is that next-generation settop boxes will support it; however, the acceptance of such an approach will surely depend on whether this enormous bandwidth waste will be tolerated by service providers.

D. Storage Format for MPEG-4 Presentations

The storage of MPEG-4 presentations seems to be an issue that is not directly related to the previously described mappings to transport formats. Conceptually, however, it is not much different. MPEG-4 presentations may consist of a large number of elementary streams, and, similar to the above, it has to be decided whether each of the streams is to be stored in a separate file or whether streams share one file, potentially through the use of FlexMux.

The issue of storage of MPEG-4 presentations has been deemed very important, so that a call for proposals has been issued. A number of responses have been received after this call that may be categorized into two vastly different approaches to the issue. The first approach proposes a simple storage format that consists just of FlexMux streams, preceded by a map to associate the ES_ID's of the elementary streams to the numbers of the FlexMux channels carrying them. At the beginning of the file, a single object descriptor either identifies the object descriptor stream(s) and scene description stream(s) within the file or, if only one elementary stream is stored in the file, identifies this stream. This approach deliberately ignores the fact that such a storage format is not particularly optimized for application in a media server or editing environment. Such a storage format is rather thought of as a pure interchange format that may be converted to more specialized storage formats in specific applications.

A second proposal attempts to cure this situation by adding more side information—for example, for indexing—to the file, trying to define a more versatile file format based on MPEG tools. In general, however, the second approach proposes to take a single, existing multimedia file format and adapt it to the needs of MPEG-4. In fact, this approach has been chosen,

so that by now an MPEG-4 intermedia format, as it is called, is being defined based on Apple's QuickTime architecture.

It is important to note that it is not intended to directly use the file format for transmission of MPEG-4 streams. The format should be translated to a transmission format that is appropriate for the target transport network. The work on the storage format is ongoing, however, and is to be included in the second version of the MPEG-4 standard that is due at the end of 1999, so that this can only represent the current status.

V. ACCESSING MPEG-4 CONTENT

The access to MPEG-4 content may occur in different ways, depending on the application context. A part of this process is standardized by MPEG in an abstract way. This abstraction relies on the availability of an interface that can establish a session context and open and manage transport channels. Such an interface is embodied by the DMIF application interface specified in [6].

The assumption is that a DMIF implementation for interactive applications will either translate the messages that are needed in the setup process to native signaling of the respective transport layer or use its own default signaling. Of course, in a scenario without back channel, no messages are generated at all, and instead broadcast or locally available information is exploited. Still, the content-access process is described as if it were a client-server process, with the aforementioned understanding.

It is assumed in this MPEG-4 content-access process that an initial communication path between client and server is available. This initial communication path may be identified in an Internet scenario by a URL that requests that HTTP or any other suitable Internet protocol be used. In a broadcast scenario, it could correspond to a special MPEG-4 descriptor in an MPEG-2 program-specific information table that points to the PID with further information.

From that point on, the following conceptual steps have to take place.

- The initial communication path allows access to a single object descriptor, called initial OD.
- The ES descriptors in this initial OD identify the primary OD stream(s) and scene description stream(s) for this service.
- The client selects the ES_ID's of those OD and scene description streams that it requires.
- The client requests delivery of the streams with these ES_ID's through the DAI.
- Handles to the respective streams are returned.
- In interactive scenarios, the client confirms that it is ready to receive data.
- Stream delivery starts.
- The client parses arriving BIFS and OD data.
- The client selects ES_ID's of those media streams that it requires.
- The clients requests delivery of the streams with these ES_ID's through the DAI (same as above).

As can be seen, the MPEG-4 content acquisition may require multiple stages between client and server, which in practice may lead to the definition of "well-known" ES_ID's, so that some streams can be acquired without having to go through the entire procedure. In any case, the abstract procedure described above needs to be mapped into concrete application scenarios, two of which are outlined below.

A. Content Access in an Internet Scenario

As far as the author may foresee right now, given the fast evolution of the Internet protocols, the access to MPEG-4 content in the Internet may be controlled via the session description protocol (SDP) [10]. SDP, which is still at the status of Internet draft, is itself comparable to some extent to the object-descriptor framework of MPEG-4. Therefore, it is worth analyzing briefly whether SDP can take the role of the object descriptors in an Internet scenario.

SDP allows one to advertise media streams that belong to a multimedia session, including some level of descriptive information about the stream and an indication of through which transport channel (user datagram protocol port) this media stream is conveyed. However, there is no explicit advertisement of hierarchical or alternative representations of a single media object, and, of course, there is no predefined way to associate specific streams to such media objects at all. There is also no means to associate intellectual property rights information or object content information to individual media objects, nor can the number of streams in the session be dynamically modified. Of course, it is conceivable to extend SDP in this direction but, assuming that the Internet will not be the only medium on which MPEG-4 content will be released, it seems preferable to stick to the MPEG-defined object descriptors to describe the elements of an MPEG-4 session and to use SDP at a higher level to simply point to this session as a whole.

In any case, SDP can be used to convey the mapping from RTP sessions, identified by their synchronization source label to the ES_ID's of the elementary streams carried in them.

Using SDP as a descriptor for an MPEG-4 presentation allows one to make use of other infrastructure that is currently being defined for content access. Multimedia content will be announced using real-time streaming protocol (RTSP) [9] with an "rtsp://" URL that establishes connection to a session control unit that allows stream control (PLAY, PAUSE, TEARDOWN, etc.). The applicability of the stream control for MPEG-4 may depend on the concrete application. However, RTSP also has a DESCRIBE command that conveys an SDP descriptor, which may in turn contain the necessary initial object descriptor to access the MPEG-4 content. Similarly, other draft protocols like session initiation protocol [12] and session announcement protocol [13] may be used, since they also use SDP descriptors to advertise a session's content.

B. Implementation in a Digital TV Broadcast Scenario

An MPEG-4 application within a digital TV broadcast scenario will not use client-server interaction, as is suggested by the abstract content-access procedure above to gain access

to the session data. Still, this procedure can conceptually be followed, *de facto* reading the initial object descriptor for an MPEG-4 session from a specific MPEG-4 descriptor in a program association table (PAT) or program map table (PMT) entry or, in the case of the DSM-CC data carousel approach, from the respective DSM-CC messages. An additional descriptor, or stream map table, is needed to establish the correspondence between the ES_ID's that identify elementary streams within the session and the transport channel consisting of either a PID, a PID plus a FlexMux channel number, or a DSM-CC data carousel module ID.

Acquisition of the MPEG-4 session continues by identifying and retrieving the object descriptor and scene description streams indicated in this initial object descriptor. After finding the next random access point in this information, it can be parsed in order to identify the media elementary streams that are required additionally. Again, at the next random access point, decoding of these media elementary streams may begin. In principle, this process is identical to MPEG-2, where access to a program is gained through the subsequent evaluation of PAT and PMT before the streams that form the program itself can be decoded.

VI. CONCLUSION

The framework of the elementary stream management in the forthcoming MPEG-4 standard has been described. Its basic building blocks are the object-description framework and the means to synchronize elementary streams. MPEG-4 content should be conveyed in existing multimedia infrastructures; therefore, approaches to embed the streaming data in such transport and storage infrastructures have been proposed. The steps that are needed to access the MPEG-4 content via its descriptive information have been outlined. The specification of the content embedding, for example, in Internet data transfer and control protocols cannot be standardized within MPEG-4, since MPEG in most cases does not have normative power to accomplish this. However, MPEG members actively pursue such standardization in the respective groups.

The scope of MPEG-4 that addresses object-based multimedia applications, possibly with multiple users in a heterogeneous network environment, makes it evident that it is not easy to define it in a way that satisfies the needs of different application domains simultaneously. Therefore, it can be expected that especially the MPEG-4 Systems specification will receive more modifications to come closer to this ambitious goal before finally the international standard status is reached, which was scheduled for the end of 1998.

A second version of the standard is already anticipated for the end of 1999, featuring, among other things, the aforementioned storage format, a framework for intellectual property management and protection, and a set of API's that will render the specification more flexible in a broader application context.

ACKNOWLEDGMENT

This paper reflects the work of many people dedicated to the success of MPEG-4. The author would like to thank all of them for their active involvement and only mention those that have the (not always pleasant) task of managing these activities, specifically O. Avaro, Chairman of the MPEG-4 Systems Group, and L. Chiariglione, Convener of ISO/IEC JTC1/SC29/WG11, better known as MPEG.

REFERENCES

- [1] Coding of audio-visual objects: Systems, ISO/IEC 14496-1; final committee draft. ISO/IEC JTC1/SC29/WG11 N2201. (May 1998). [Online]. Available WWW: http://www.csel.it/mpeg/public/mpeg-4_fcd/w2201.zip.
- [2] Coding of audio-visual objects: Visual, ISO/IEC 14496-2; final committee draft. ISO/IEC JTC1/SC29/WG11 N2202. (May 1998). [Online]. Available WWW: http://www.csel.it/mpeg/public/mpeg-4_fcd/w2202.zip.
- [3] Coding of audio-visual objects: Audio, ISO/IEC 14496-3; final committee draft. ISO/IEC JTC1/SC29/WG11 N2203. (May 1998). [Online]. Available WWW: http://www.csel.it/mpeg/public/mpeg-4_fcd/w2203.zip.
- [4] The virtual reality modeling language—International standard ISO/IEC 14772-1:1997. (1997). [Online]. Available WWW: <http://www.vrml.org/Specifications/VRML97/>.
- [5] Coding of audio-visual objects: Reference software, ISO/IEC 14496-5; final committee draft. ISO/IEC JTC1/SC29/WG11 N2205. (May 1998). [Online]. Available WWW: http://www.csel.it/mpeg/public/mpeg-4_fcd/w2205.zip.
- [6] Coding of audio-visual objects: Delivery multimedia integration framework, ISO/IEC 14496-6; final committee draft. ISO/IEC JTC1/SC29/WG11 N2206. (May 1998). [Online]. Available WWW: http://www.csel.it/mpeg/public/mpeg-4_fcd/w2206.zip.
- [7] "Generic coding of moving pictures and associated audio: Systems, ISO/IEC 13818-1," ISO/IEC JTC1/SC29/WG11 N0801, Nov. 1994.
- [8] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. (1996). RTP: A transport protocol for real-time applications, RFC 1889. [Online]. Available: <http://ds.internic.net/rfc/rfc1889.txt>.
- [9] H. Schulzrinne, A. Rao, and R. Lanphier. Real time streaming protocol (RTSP), Internet draft. [Online]. Available FTP: <ftp://ietf.org/internet-drafts/draft-ietf-mmusic-rtsp-09.txt>.
- [10] V. Jacobson and M. Handley. SDP: Session description protocol, Internet draft. [Online]. Available FTP: <ftp://ietf.org/internet-drafts/draft-ietf-mmusic-sdp-06.txt>.
- [11] D. D. Clark and D. L. Tennenhouse, "Architectural considerations for a new generation of protocols," in *Proc. SIGCOMM Symp. Communications Architectures and Protocols*, Philadelphia, PA, Sept. 1990, pp. 200–208; also *Comput. Commun. Rev.*, vol. 20, p. 4, Sept. 1990.
- [12] E. Schooler, H. Schulzrinne, and M. Handley. SIP: Session initiation protocol, Internet draft. [Online]. Available FTP: <ftp://ietf.org/internet-drafts/draft-ietf-mmusic-sip-07.txt>.
- [13] M. Handley and V. Jacobson. SAP: Session announcement protocol, Internet draft. [Online]. Available: [draft-ietf-mmusic-sap-00.txt](http://www.csel.it/mpeg/public/mpeg-4_fcd/w2206.zip).



Carsten Herpel was born in Cologne, Germany, in 1962. He received the diploma in communication engineering from RWTH Aachen, Germany, in 1988.

He joined Thomson Multimedia's research facility in Hannover, Germany, and developed video coding algorithms with a focus on scalable coding tools. His research interests then moved to multimedia systems, with a current focus on MPEG-4. He is a Coeditor of the MPEG-4 Systems specification.