INTERNATIONAL STANDARD

ISO/IEC 14496-14

First edition
2003-11-15

Information technology — Coding of audio-visual objects —

Part 14: MP4 file format

Technologies de l'information — Codage des objets audiovisuels —

Partie 14: Format de fichier MP4

Reference number
ISO/IEC 14496-14:2003(E)

PDF disclaimer

This PDF file may contain embedded typefaces. In accordance with Adobe's licensing policy, this file may be printed or viewed but shall not be edited unless the typefaces which are embedded are licensed to and installed on the computer performing the editing. In downloading this file, parties accept therein the responsibility of not infringing Adobe's licensing policy. The ISO Central Secretariat accepts no liability in this area.

Adobe is a trademark of Adobe Systems Incorporated.

Details of the software products used to create this PDF file can be found in the General Info relative to the file; the PDF-creation parameters were optimized for printing. Every care has been taken to ensure that the file is suitable for use by ISO member bodies. In the unlikely event that a problem relating to it is found, please inform the Central Secretariat at the address given below.

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

Page

Foreword iv

Introduction v
0.1 Derivation v
0.2 Interchange v
0.3 Content Creation v
0.4 Streamed presentation vi

1 Scope 1

2 Normative references 1

3 Storage of MPEG-4 1
3.1 Elementary Stream Tracks 1
3.2 Track Identifiers 3
3.3 Synchronization of streams 4
3.4 Composition 5
3.5 Handling of FlexMux 5

4 File Identification 6

5 Additions to the Base Media Format 6
5.1 Object Descriptor Box 7
5.2 Track Reference Types 7
5.3 Track Header Box 8
5.4 Handler Reference Types 8
5.5 MPEG-4 Media Header Boxes 8
5.6 Sample Description Boxes 8
5.7 Degradation Priority Values 10

6 Template fields used 10

Annex A (informative) **Patent statements** 11

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

ISO/IEC 14496-14 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 29, *Coding of audio, picture, multimedia and hypermedia information*.

ISO/IEC 14496 consists of the following parts, under the general title *Information technology — Coding of audio-visual objects*:

Part 1: Systems

Part 2: Visual

Part 3: Audio

Part 4: Conformance testing

Part 5: Reference software

Part 6: Delivery Multimedia Integration Framework (DMIF)

Part 7: Optimized reference software for coding of audio-visual objects

Part 8: Carriage of ISO/IEC 14496 contents over IP networks

Part 9: Reference hardware description

Part 10: Advanced Video Coding (AVC)

Part 11: Scene description and application engine

Part 12: ISO base media file format

Part 13: Intellectual Property Management and Protection (IPMP) extensions

Part 14: MP4 file format

Part 15: Advanced Video Coding (AVC) file format

Part 16: Animation Framework eXtension (AFX)

Introduction

0.1 Derivation

This specification defines MP4 as an instance of the ISO Media File format [ISO/IEC 14496-12 and ISO/IEC 15444-12].

The general nature of the ISO Media File format is fully exercised by MP4. MPEG-4 presentations can be highly dynamic, and there is an infrastructure — the Object Descriptor Framework —, which serves to manage the objects and streams in a presentation. An Initial Object Descriptor serves as the starting point for this framework. In the usage modes documented in the ISO Media File, an Initial Object Descriptor would normally be present, as shown in the following diagrams.

0.2 Interchange

The following diagram gives an example of a simple interchange file, containing two streams.

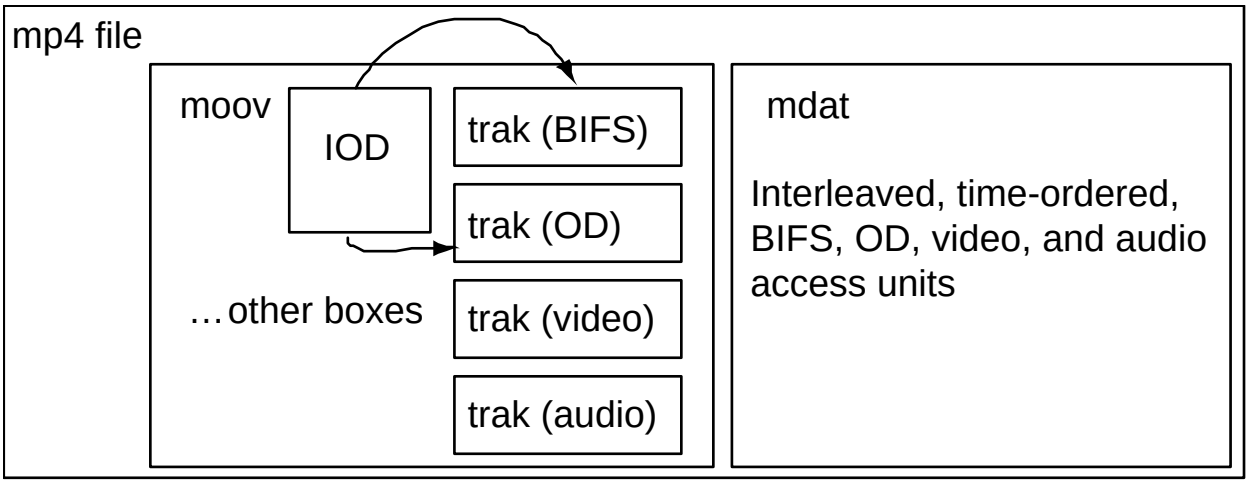

Figure 1 — Simple interchange file

0.3 Content Creation

In the following diagram, a set of files being used in the process of content creation is shown.

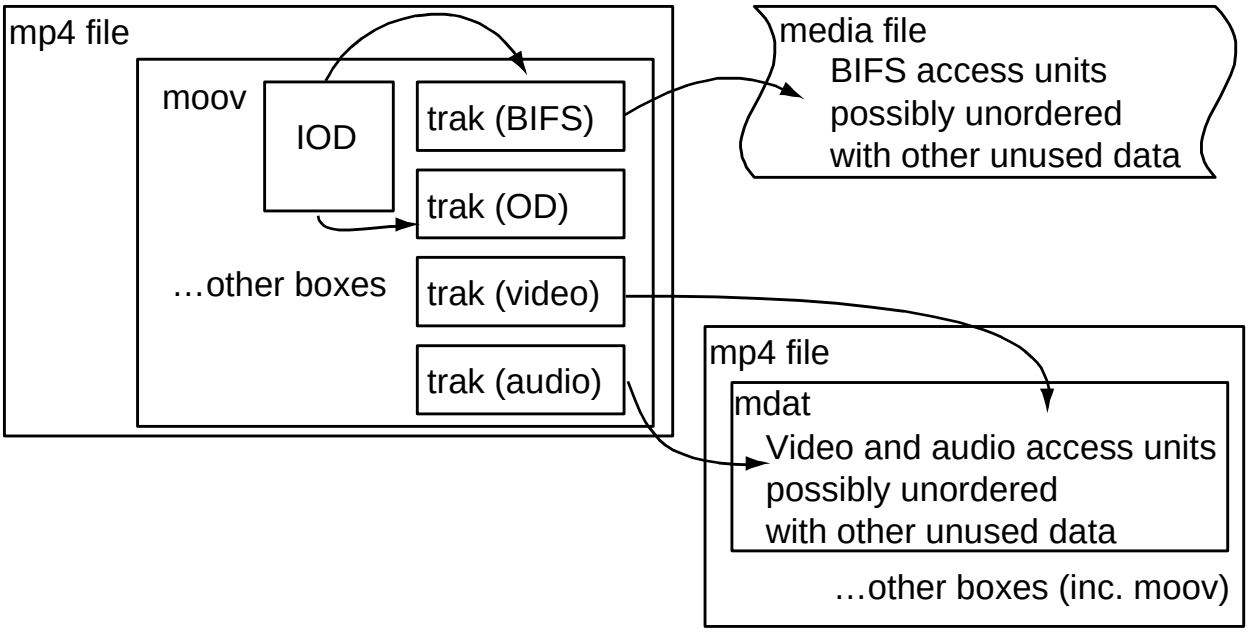

Figure 2 — Content Creation File

0.4 Streamed presentation

The following diagram shows a presentation prepared for streaming over a multiplexing protocol, only one hint track is required.

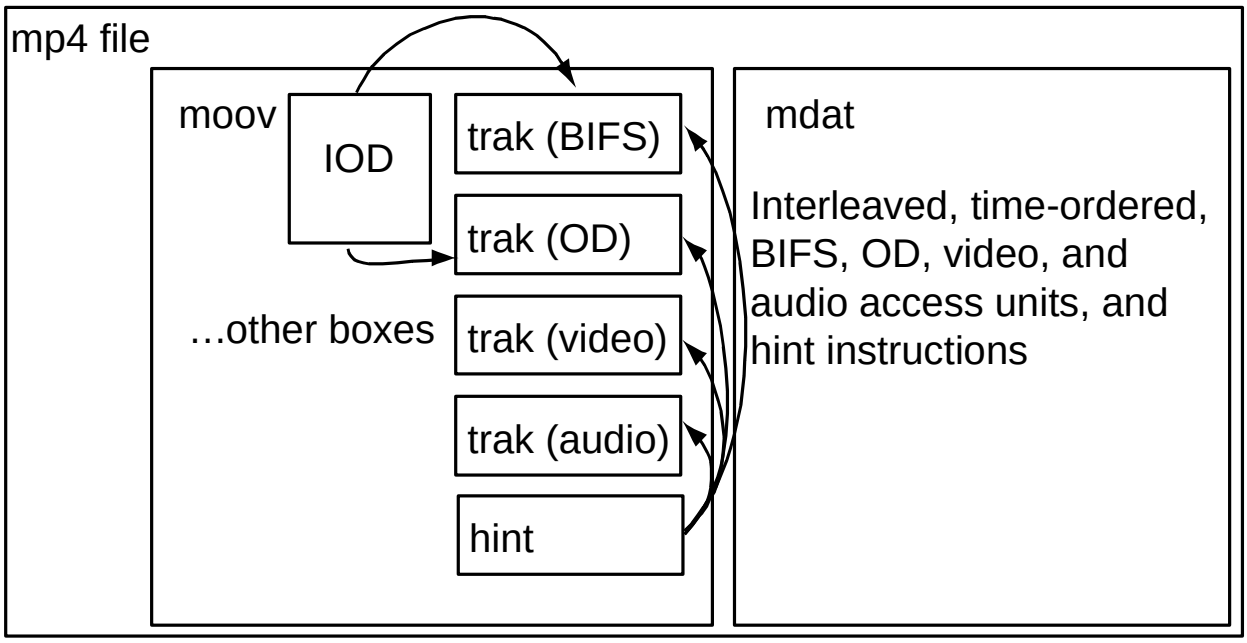

Figure 3 — Hinted Presentation for Streaming

Information technology — Coding of audio-visual objects —

Part 14:
MP4 file format

1 Scope

This International Standard defines the MP4 file format, as derived from the ISO Base Media File format.

2 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 14496-1:2001, *Information technology — Coding of audio-visual objects — Part 1: Systems*

ISO/IEC 14496-12: *Information technology — Coding of audio-visual objects — Part 12: ISO base media file format* (technically identical to ISO/IEC 15444-12)

3 Storage of MPEG-4

3.1 Elementary Stream Tracks

3.1.1 Elementary Stream Data

To maintain the goals of streaming protocol independence, the media data is stored in its most 'natural' format, and not fragmented. This enables easy local manipulation of the media data. Therefore media-data is stored as access units, a range of contiguous bytes for each access unit (a single access unit is the definition of a 'sample' for an MPEG-4 media stream). This greatly facilitates the fragmentation process used in hint tracks. The file format can describe and use media data stored in other files, however this restriction still applies. Therefore if a file is to be used which contains 'pre-fragmented' media data (e.g. a FlexMux stream on disc), the media data will need to be copied to re-form the access units, in order to import the data into this file format.

This is true for all stream types in this specification, including such 'meta-information' streams as Object Descriptor and the Clock Reference. The consequences of this are, on the positive side, that the file format treats all streams equally; on the negative side, this means that there are 'internal' cross-links between the streams. This means that adding and removing streams from a presentation will involve more than adding or deleting the track and its associated media-data. Not only must the stream be placed in, or removed from, the scene, but also the object descriptor stream may need updating.

For each track, the entire ES-descriptor is stored as the sample description or descriptions. The SLConfigDescriptor for the media track shall be stored in the file using a default value (predefined = 2), except when the Elementary Stream Descriptor refers to a stream through a URL, i.e. the referred stream is outside the scope of the MP4 file. In that case the SLConfigDescriptor is not constrained to this predefined value.

In a transmitted bit-stream, the access units in the SL Packets are transmitted on byte boundaries. This means that hint tracks will construct SL Packet headers using the information in the media tracks, and the hint tracks will reference the access units from the media track. The placement of the header during hinting is possible without bit shifting, as each SL Packet and corresponding contained access unit will both start on byte boundaries.

3.1.2 Elementary Stream Descriptors

The ESDescriptor for a stream within the scope of the MP4 file as described in this document is stored in the sample description and the fields and included structures are restricted as follows.

- `ES_ID` — set to 0 as stored; when built into a stream, the lower 16 bits of the TrackID are used.
- `streamDependenceFlag` — set to 0 as stored; if a dependency exists, it is indicated using a track reference of type 'dpnd'.
- `URLflag` — kept untouched, i.e. set to false, as the stream is in the file, not remote.
- `SLConfigDescriptor` — is predefined type 2.
- `OCRStreamFlag` — set to false in the file.

The ESDescriptor for a stream referenced through an ES URL is stored in the sample description and the fields and included structures are restricted as follows.

- `ES_ID` — set to 0 as stored; when built into a stream, the lower 16 bits of the TrackID are used.
- `streamDependenceFlag` — set to 0 as stored; if a dependency exists, it is indicated using a track reference of type 'dpnd'.
- `URLflag` — kept untouched, i.e. set to true, as the stream is not in the file.
- `SLConfigDescriptor` — kept untouched.
- `OCRStreamFlag` — set to false in the file.

Note that the QoSDescriptor also may need re-writing for transmission as it contains information about PDU sizes etc.

3.1.3 Object Descriptors

The initial object descriptor and object descriptor streams are handled specially within the file format. Object descriptors contain ES descriptors, which in turn contain stream specific information. In addition, to facilitate editing, the information about a track is stored as an ESDescriptor in the sample description within that track. It must be taken from there, re-written as appropriate, and transmitted as part of the OD stream when the presentation is streamed.

As a consequence, ES descriptors are not stored within the OD track or initial object descriptor. Instead, the initial object descriptor has a descriptor used only in the file, containing solely the track ID of the elementary stream. When used, an appropriately re-written ESDescriptor from the referenced track replaces this descriptor. Likewise, OD tracks are linked to ES tracks by track references. Where an ES descriptor would be used within the OD track, another descriptor is used, which again occurs only in the file. It contains the index into the set of mpod track references that this OD track owns. A suitably re-written ESDescriptor replaces it by the hinting of this track.

The ES_ID_Inc is used in the Object Descriptor Box:

```
class ES_ID_Inc extends BaseDescriptor : bit(8) tag=ES_IDIncTag {
  unsigned int(32)   Track_ID;   // ID of the track to use
}

ES_ID_IncTag = 0x0E is reserved for file format usage.
```

The ES_ID_Ref is used in the OD stream:

```
class ES_ID_Ref extends BaseDescriptor : bit(8) tag=ES_IDRefTag {
  bit(16)   ref_index;  // track ref. index of the track to use
}

ES_ID_RefTag = 0x0F is reserved for file format usage.
MP4_IOD_Tag = 0x10 is reserved for file format usage.
MP4_OD_Tag = 0x11 is reserved for file format usage.
IPI_DescrPointerRefTag = 0x12 is reserved for file format usage.
ES_DescrRemoveRefTag = 0x07 is reserved for file format usage (command tag).
```

NOTE The above tag values are defined in 8.2.2.2 Table 1 and 8.2.3.2 Table 2 of the MPEG-4 Systems Specification, and the actual values should be referenced from those tables.

A hinter may need to send more OD events than actually occur in the OD track: for example, if the ES_description changes at a time when there is no event in the OD track. In general, any OD events explicitly authored into the OD track should be sent along with those necessary to indicate other changes. The ES descriptor sent in the OD track is taken from the description of the temporally next sample in the ES track (in decoding time).

3.2 Track Identifiers

The track identifiers used in an MP4 file are unique within that file; no two tracks may use the same identifier.

Each elementary stream in the file is stored as a media track. In the case of an elementary stream, the lower two bytes of the four-byte `track_ID` shall be set to the elementary stream identifier (ES_ID).; the upper two bytes of the track_ID are zero in this case. Hint tracks may use track identifier values in the same range, if this number space is adequate (which it generally is). However, hint track identifiers may also use larger values of track identifier, as their identifiers are not mapped to elementary stream identifiers. Thus very large presentations may use the entire 16-bit number space for elementary stream identifiers.

The next track identifier value, found in next_track_ID in the MovieHeaderBox, as defined in the ISO Base Media Format, generally contains a value one greater than the largest track identifier value found in the file. This enables easy generation of a track identifier under most circumstances. However, if this value is equal to or larger than 65535, and a new media track is to be added, then a search must be made in the file for a free track identifier. If the value is all 1s (32-bit maxint) then this search is needed for all additions.

If it is desired to add a track with a known track identifier (elementary stream identifier) then the file must be searched to ensure that there is no conflict. Note that hint tracks can be re-numbered fairly easily while more care should be taken with media tracks, as there may be references to their ES_ID (track ID) in other tracks.

If hint tracks have track IDs outside the allowed range for elementary stream tracks, then next track ID documents the next available hint track ID. Since this is larger than 65535, a search will then always be needed to find a valid elementary stream track ID.

If two presentations are merged, then there may be conflict between their track IDs. In that case, one or more tracks will have to be re-numbered. There are two actions to be taken here:

- Changing the ID of the track itself, which is easy (track ID in the track header).
- Changing pointers to it.

The pointers may only occur in the file format structure itself. The file format uses track IDs only through track references, which are easily found and modified. Track IDs become ES_IDs in the MPEG-4 data, and ES_IDs occur within the OD Stream. Since all pointers to ES_IDs in the OD stream are replaced by means of track references, there is no need to inspect the OD stream for cross-references within MPEG-4 streams.

In the file format, ES_DescriptorRemove command and IPI_DescrPointer descriptor are converted to ES_DescrRemoveRef and IPI_DescrPointerRef by:

- changing the tag value to ES_DescrRemoveRefTag or IPI_DescrPointerRefTag respectively;
- changing any ES_ID to the appropriate track reference index (using references of type mpod and ipir respectively – see 5.2).

When hinting or serving, the tag value and track reference index changes shall be reversed.

3.3 Synchronization of streams

In the absence of explicit declarations to the contrary, tracks (streams) coming from the same file shall be presented synchronized. This means that hinters and/or servers must either pick one of the streams to serve as the OCR source for the others or add an OCR stream to associate all the streams with it. Track references of type 'sync' may be used in the file to defeat the default behavior. In MPEG-4 the OCRStreamFlag and OCR_ES_ID fields in the ESDescriptor govern the synchronization relationships. The mapping of MP4 structures into those fields shall obey the following rules.

- The MPEG-4 ESDescriptor, as stored in the file, usually contains OCRStreamFlag set to FALSE, and no OCR_ES_ID. If an OCR_ES_ID is set, it is ignored.
- If a track (stream) contains a track reference of type 'sync' whose value is 0, then the hinter or server shall set the OCRStreamFlag field in the MPEG-4 ESDescriptor to FALSE and shall not insert any OCR_ES_ID field. This means that this stream is not synchronized to another, but other streams may be synchronized to it.
- If a track (stream) contains a track reference of type 'sync' whose value is not 0, then the hinter or server shall set the OCRStreamFlag field in the MPEG-4 ESDescriptor to TRUE and shall insert an OCR_ES_ID field with the same value contained in the 'sync' track reference. This means that this stream is synchronized to the stream indicated in the OCR_ES_ID. Other streams may also be synchronized to the same stream, either explicitly or implicitly.
- If a track (stream) does not contain a track reference of type 'sync', then the default behavior applies. The hinter or server shall set the OCRStreamFlag field in the MPEG-4 ESDescriptor to TRUE and shall insert an OCR_ES_ID field with a value selected based on the rules below. This means that this stream is synchronized to the stream indicated in the OCR_ES_ID. The rules for selecting the OCR_ES_ID are as follows.
    - If no track (stream) in the file contains a track reference of type 'sync', then the hinter picks one TrackId and uses that value for the OCR_ES_ID field of all ESDescriptors. There is one possible exception where the ESDescriptor of the stream which corresponds to that TrackId, for which the OCRStreamFlag may be set to FALSE.
    - If one or more tracks (streams) in the file contain a track reference of type 'sync', and all such track references indicate consistently a single TrackId, then the hinter uses that TrackId. In a track reference of type 'sync' the value 0 is equivalent to the TrackId of the track itself.
    - If two or more tracks (streams) in the file contain a track reference of type 'sync', and such track references do not indicate a single TrackId, then the hinter cannot make a deterministic selection and the behavior is undefined. In a track reference of type 'sync' the value 0 is equivalent to the TrackId of the track itself.

3.4 Composition

In MPEG-4 both visual and aural composition are done using the BIFS system. Therefore structures marked as “template” in the ISO Base Media Format which pertain to composition, including fields such as matrices, layers, graphics modes (and their opcolors), volumes, and balance values, from the MovieHeaderBox and TrackHeaderBox, are all set to their default values in the file format. These fields do not define visual or audio composition in MPEG-4; in MPEG-4, the BIFS system defines the composition.

The fields width and height in the VisualSampleEntry and in the Track Header Box shall be set to the pixel dimensions of the visual stream.

3.5 Handling of FlexMux

An intermediate, optional, fragmentation and packetization step, called FlexMux, has been defined in this document. Some streaming protocols may carry a FlexMux stream rather than packetized elementary streams. Flexmux may be employed for a variety of purposes, including, but not limited to:

- reducing wasted network bandwidth caused by SL Packet header overhead when the payload is small;

- reducing required server resources when providing many streams, by reducing the number of disk reads or network writes.

The process of building FlexMux PDUs is necessarily aware of the characteristics of the streaming protocol into which the FlexMux must be placed. It is not therefore possible to design a streaming protocol-independent handling of FlexMux. Instead, in those streaming protocols where FlexMux is used, the hint tracks for that protocol will encapsulate and include the formation of FlexMux packets. It is expected that the design of the hint tracks will, in this case, closely reflect the way that FlexMux is used. For example, a compact table resembling the MuxCode (a method used to associate the payload to FlexMux Channels) mode may be needed if the interleave offered by that mode is needed.

In some cases, it may not be possible to create a static FlexMux multiplex via a hint track. Notably, if stream selection is dynamic (for example, based on application feedback) or the choice of muxcode modes or other aspects of Flexmux is dynamic, the FlexMux is therefore created dynamically. This is a necessary cost of run-time multiplexing. It may be difficult for a server to create such a multiplex dynamically at runtime, but with this cost comes added flexibility. A server that wished to provide such functionality could weigh the costs and benefits, and choose to perform the multiplexing without the aid of hint tracks.

Several ISO/IEC 14496 structures are intrinsically linked to FlexMux, and therefore must be addressed in the context of a FlexMux-aware hint track. For example, a stream map table must be supplied to the receiving terminal which maps FlexMux channel IDs to elementary stream IDs. Similarly, if the MuxCode mode of FlexMux is used, a MuxCode mode structure for each MuxCode index used must be defined and supplied to the terminal.

These mappings and definitions may change over time, and there is no normative way in ISO/IEC 14496 to supply these to the terminals; instead, some mechanism, associated with the overall system design or protocol used, must be employed. The hinter must store the mappings and definitions. Because they are intimately associated with a particular time-segment of a particular hint track, it is recommended that they be placed in the sample description(s) for that hint track. This description would normally be in the form of:

- a table mapping FlexMux channels to elementary stream IDs;

- a set of MuxCode mode structure definitions.

It is recommended further that a format such as that in subclause 12.2.5, be used for the MuxCode mode definitions.

```
aligned(8) class MuxCodeTableEntry {
   int    i, k;
   bit(8) length;
   bit(4) MuxCode;
   bit(4) version;
   bit(8) substructureCount;
   for (i=0; i<substructureCount; i++) {
      bit(5) slotCount;
      bit(3) repetitionCount;
      for (k=0; k<slotCount; k++){
         bit(8) flexMuxChannel[[i]][[k]];
         bit(8) numberOfBytes[[i]][[k]];
      }
   }
}
```

Special attention must also be taken when pausing or seeking a stream that is being transported as part of a FlexMux stream. Pausing or seeking any component stream of a FlexMux must necessarily pause or seek all the streams. When seeking, care must be taken with random access points. These may not be aligned in time in the streams which form the FlexMux, which means that any seek operation cannot start them all at a random access point. Indeed, the random access points of the FlexMux itself are necessarily rather poorly defined under such circumstances.

It may be necessary for the server to:

- examine the track references to determine the base media tracks (elementary streams) which are formed into the FlexMux;
- find the latest time before the desired seek point such that there is a random access point for all the streams between that time and the seek point, by examining each stream separately;
- transmit the FlexMux stream from that time.

This will ensure that the terminal has received a random access point for all streams at or prior to the desired seek time. However, it may have to discard data for those streams which had data received before the random access points.

4 File Identification

The brand ‘mp41’ is defined as identifying version 1 of this specification (ISO/IEC 14496-1:2001), and the brand ‘mp42’ identifies this version of the specification; at least one of these brands shall appear in the compatible-brands list in the file-type box, in all files conforming to this specification.

The preferred file extension is ‘.mp4’. The MIME types video/mp4, audio/mp4 are used as defined in the appropriate RFC.

5 Additions to the Base Media Format

This section defines the boxes, and track reference types, which are defined for use in this file format and are not defined in the ISO Base Media File Format.

5.1 Object Descriptor Box

Box Type: 'iods'
Container: Movie Box ('moov')
Mandatory: No
Quantity: Zero or one

This object contains an Object Descriptor or an Initial Object Descriptor.

There are a number of possible file types based on usage, depending on the descriptor:

- Presentation, contains IOD which contains a BIFS stream (MP4 file);
- Sub-part of a presentation, contains an IOD without a BIFS stream (MP4 file);
- Sub-part of a presentation, contains an OD (MP4 file);
- Free-form file, referenced by MP4 data references (free-format);
- Sub-part of a presentation, referenced by an ES URL.

NOTE The first three are MP4 files, a file referenced by a data reference is not necessarily an MP4 file, as it is free-format. Files referenced by ES URLs, by data references, or intended as input to an editing process, need not have an Object Descriptor Box.

An OD URL may point to an MP4 file. Implicitly, the target of such a URL is the OD/IOD located in the 'iods' atom in that file.

If an MP4 file contains several object descriptors, only the OD/IOD in the 'iods' atom can be addressed using an OD URL from a remote MPEG-4 presentation.

5.1.1 Syntax

```
aligned(8) class ObjectDescriptorBox
  extends FullBox('iods', version = 0, 0) {
  ObjectDescriptor OD;
}
```

The syntax for ObjectDescriptor and InitialObjectDescriptor is described in 8.6.2 through 8.6.4.

5.1.2 Semantics

The semantics for ObjectDescriptor and InitialObjectDescriptor are described in 8.6.2 through 8.6.4. The contents of this box are formed by taking an object descriptor or initial object descriptor and:

- changing the tag to MP4_OD_Tag or MP4_IOD_Tag as appropriate for this object;
- replacing the ES descriptors with ES_ID_Inc referencing the appropriate track

5.2 Track Reference Types

MP4 defines the following additional values for `reference-type`:

- dpnd — this track has an MPEG-4 dependency on the referenced track;
- ipir — this track contains IPI declarations for the referenced track;
- mpod — this track is an OD track which uses the referenced track as an included elementary stream track;
- sync — this track uses the referenced track as its synchronization source.

The reference type 'cdsc' (content describes) is the way within an MP4 file that description streams (such as MPEG-7) are linked to the content they describe; when the file is streamed or hinted, these track references are used to form an ObjectDescriptor describing the content and the description, or the DescriptionDescriptionDescriptor as appropriate.

5.3 Track Header Box

The track header box documents the track duration. If the duration of a track cannot be determined, then the duration is set to all 1s (32-bit maxint): this is the case when an Elementary Stream Descriptor contains a ES_URL, since the media content is outside the MP4 file and its partitioning into samples is not known.
The track header flags track_in_movie and track_in_preview are not used in MP4 and shall be set to the default value of 1 in all files.

5.4 Handler Reference Types

The following additional values for handler-type, in the Handler Reference Box ('hdlr') of the ISO Base Media File Format, are defined:

'odsm'	ObjectDescriptorStream
'crsm'	ClockReferenceStream
'sdsm'	SceneDescriptionStream
'm7sm'	MPEG7Stream
'ocsm'	ObjectContentInfoStream
'ipsm'	IPMP Stream
'mjsm'	MPEG-J Stream

5.5 MPEG-4 Media Header Boxes

ISO/IEC 14496 streams other than visual and audio currently use an empty MPEG-4 Media Header Box, as defined here. There is a set of reserved types for media headers specific to these ISO/IEC 14496 stream types.

5.5.1 Syntax

```
aligned(8) class Mpeg4MediaHeaderBox extends NullMediaHeaderBox( flags ) { };
```

5.5.2 Semantics

`version` - is an integer that specifies the version of this box.
`flags` - is a 24-bit integer with flags (currently all zero).

The following box types are reserved as potential Media Header box types, but are currently unused:

ObjectDescriptorStream	'odhd'
ClockReferenceStream	'crhd'
SceneDescriptionStream	'sdhd'
MPEG7Stream	'm7hd'
ObjectContentInfoStream	'ochd'
IPMP Stream	'iphd'
MPEG-J Stream	'mjhd'

5.6 Sample Description Boxes

Box Types: 'mp4v', 'mp4a', 'mp4s'
Container: Sample Table Box ('stbl')
Mandatory: Yes
Quantity: Exactly one

For visual streams, a VisualSampleEntry is used; for audio streams, an AudioSampleEntry. For all other MPEG-4 streams, a MpegSampleEntry is used. Hint tracks use an entry format specific to their protocol, with an appropriate name.

For all the MPEG-4 streams, the data field stores an ES_Descriptor with all its contents. Multiple entries in the table imply the occurrence of ES_DescriptorUpdate commands. In case an ES_Descriptor references the stream through an ES URL (thus outside the scope of the MP4 file as described in this document) only one entry in this table is allowed, i.e. the occurrence of ES_DescriptorUpdate commands is not supported. The ES_Descriptor as stored within the file format is constrained by the rules set in 3.1

For hint tracks, the sample description contains appropriate declarative data for the protocol being used, and the format of the hint track. The definition of the sample description is specific to the streaming protocol. However, note the discussion of FlexMux above, and the need for a Stream Map table, and MuxCode mode format definitions.

For visual streams, Annex K subclause 3.1 of the video specification requires that configuration information (e.g. the video sequence header) be carried in the decoder configuration structure, and not in stream. Since MP4 is a systems structure, it should be noted that that means that these headers (video object sequence, and so on) shall be in the ES_descriptor in the sample description, and not in the media samples themselves.

5.6.1 Syntax

```
aligned(8) class ESDBox
   extends FullBox('esds', version = 0, 0) {
   ES_Descriptor ES;
}
   // Visual Streams

class MP4VisualSampleEntry() extends VisualSampleEntry ('mp4v'){
   ESDBox ES;
}
   // Audio Streams

class MP4AudioSampleEntry() extends AudioSampleEntry ('mp4a'){
   ESDBox ES;
}
   // all other Mpeg stream types
class MpegSampleEntry() extends SampleEntry ('mp4s'){
  ESDBox ES;
}

aligned(8) class SampleDescriptionBox (unsigned int(32) handler_type)
  extends FullBox('stsd', 0, 0){
  int i ;
  unsigned int(32) entry_count;
  for (i = 0 ; i < entry_count ; i++){
    switch (handler_type){
      case 'soun': // AudioStream
        AudioSampleEntry();
        break;
      case 'vide': // VisualStream
        VisualSampleEntry();
        break;
      case 'hint': // Hint track
        HintSampleEntbry();
        break;
      default :
        MpegSampleEntry();
        break;
   }
 }
```

5.6.2 Semantics

`Entry_count` — is an integer that gives the number of entries in the following table.

`SampleEntry` — is the appropriate sample entry.

`width` in the VisualSampleEntry is the maximum visual width of the stream described by this sample description, in pixels, as described in ISO/IEC 14496-2, 6.2.3, video_object_layer_width in the visual headers; it is repeated here for the convenience of tools;

`height` in the VisualSampleEntry is the maximum visual height of the stream described by this sample description, in pixels, as described in ISO/IEC 14496-2, 6.2.3, video_object_layer_height in the visual headers; it is repeated here for the convenience of tools;

`compressorname` in the sample entries shall be set to 0

`ES` — is the ES Descriptor for this stream.

5.7 Degradation Priority Values

In the Degradation Priority Box, the maximum size of a degradation priority in the SL header is 15 bits; this is smaller than the field size of 16 bits. The most-significant bit is reserved as zero.

6 Template fields used

In the section “Data Types and Fields” of the ISO Base Media File Format, the concept of “template” fields is defined. This specification derives from the base, and it is required that any derived specification state explicitly which template fields are used. This format uses no template fields.

When a file is created as a pure MPEG-4 file, those fields shall be set to their default values. If a file is multi-purpose and also complies with other specifications, then those fields may have non-default values as required by those other specifications.

When a file is read as an MPEG-4 file, the values in the template fields shall be ignored.

Annex A
(informative)

Patent statements

The International Organization for Standardization and the International Electrotechnical Commission (IEC) draw attention to the fact that it is claimed that compliance with this part of ISO/IEC 14496 may involve the use of patents.

ISO and IEC take no position concerning the evidence, validity and scope of these patent rights.

The holders of these patent rights have assured the ISO and IEC that they are willing to negotiate licences under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statements of the holders of these patents right are registered with ISO and IEC. Information may be obtained from the companies listed below.

Attention is drawn to the possibility that some of the elements of this part of ISO/IEC 14496 may be the subject of patent rights other than those identified in this annex. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

	Company
1.	Apple
2.	IBM
3.	Matsushita Electric Industrial Co., Ltd.
4.	Mitsubishi Electric

ICS 35.040

Price based on 11 pages