INTERNATIONAL STANDARD

ISO/IEC 14496-12

Fourth edition
2012-07-15

Corrected version
2012-09-15

AMENDMENT 2
2014-01-15

Information technology — Coding of audio-visual objects —

Part 12: ISO base media file format

AMENDMENT 2: Carriage of timed text and other visual overlays

Technologies de l'information — Codage des objets audiovisuels —

Partie 12: Format ISO de base pour les fichiers médias

AMENDEMENT 2: Transport de texte temporisé et autres recouvrements visuels

Reference number
ISO/IEC 14496-12:2012/Amd.2:2014(E)

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

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.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

Amendment 2 to ISO/IEC 14496-12:2012 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 29, *Coding of audio, picture, multimedia and hypermedia information*.

Information technology — Coding of audio-visual objects —

Part 12:
ISO base media file format

AMENDMENT 2: Carriage of timed text and other visual overlays

In subclause 6.2.3, Table 1, add a new row for sthd as follows (the other rows of the table are provided here to show the position but are unchanged):

			minf			*		*media information container*
				vmhd				*video media header, overall information (video track only)*
				smhd				*sound media header, overall information (sound track only)*
				sthd			8.4.5.6	*subtitle media header, overall information (subtitle track only)*
				hmhd				*hint media header, overall information (hint track only)*
				nmhd				*Null media header, overall information (some tracks only)*

In section 8.4.3.1, replace

This box within a Media Box declares the process by which the media-data in the track is presented, and thus, the nature of the media in a track. For example, a video track would be handled by a video handler.

with

This box within a Media Box declares media type of the track, and thus the process by which the media-data in the track is presented. For example, a format for which the decoder delivers video would be stored in a video track, identified by being handled by a video handler. The documentation of the storage of a media format identifies the media type which that format uses.

In section 8.4.3.1, replace

There is a general handler for metadata streams of any type; the specific format is identified by the sample entry, as for video or audio, for example. If they are in text, then a MIME format is supplied to document their format; if in XML, each sample is a complete XML document, and the namespace of the XML is also supplied.

with

There is a general handler for metadata streams of any type; the specific format is identified by the sample entry, as for video or audio, for example.

and add the following before the final Notes

The timed text media type indicates that the associated decoder will process only text data. The subtitle media type indicates that the associated decoder will process text data and possibly images.

In 8.4.3.3, add the following lines to the list of handler_types:

`'text'`	Timed text track
`'subt'`	Subtitle track

Add to 8.4.5.1, before the end

Which type of media header is used is determined by the media handler:

— video track	VideoMediaHeaderBox
— audio track	SoundMediaHeaderBox
— timed metadata track	NullMediaHeaderBox
— timed text track	NullMediaHeaderBox
— subtitle track	SubtitleMediaHeaderBox
— hint tracks	HintMediaHeaderBox

Change subclause 8.4.5.5 as follows:

Streams for which no specific media header is identified use a null Media Header Box, as defined here.

Add a new subclause 8.4.5.6 as follows:

8.4.5.6 Subtitle Media Header Box

The subtitle media header contains general presentation information, independent of the coding, for subtitle media. This header is used for all tracks containing subtitles.

8.4.5.6.1 Syntax


```
aligned(8) class SubtitleMediaHeaderBox
   extends FullBox ('sthd', version = 0, flags = 0){
 }
```

8.4.5.6.2 Semantics

`version` is an integer that specifies the version of this box.

`flags` is a 24-bit integer with flags for future use (currently all zero)

In 8.5.2.1, replace the paragraph

For video tracks, a VisualSampleEntry is used**,** for audio tracks, an AudioSampleEntry and for metadata tracks, a MetaDataSampleEntry. Hint tracks use an entry format specific to their protocol, with an appropriate name.

with

Which type of sample entry form is used is determined by the media handler:

— video track	VisualSampleEntry
— audio track	AudioSampleEntry
— timed metadata track	MetaDataSampleEntry
— timed text track	PlainTextSampleEntry
— subtitle track	SubtitleSampleEntry
— hint tracks	an entry format specific to their protocol, with an appropriate name.

In 8.5.2.1 replace the paragraph

The samplerate, samplesize and channelcount fields document the default audio output playback format for this media. The timescale for an audio track should be chosen to match the sampling rate, or be an integer multiple of it, to enable sample-accurate timing. `ChannelCount` is a value greater than zero that indicates the maximum number of channels that the audio could deliver. A `ChannelCount` of 1 indicates mono audio, and 2 indicates stereo (left/right). When values greater than 2 are used, the codec configuration should identify the channel assignment.

with

The samplerate, samplesize and channelcount fields document the default audio output playback format for this media. The timescale for an audio track should be chosen to match the sampling rate, or be an integer multiple of it, to enable sample-accurate timing. `ChannelCount` is a value greater than zero that indicates the maximum number of channels that the audio could deliver. A `ChannelCount` of 1 indicates mono audio, and 2 indicates stereo (left/right). When values greater than 2 are used, the codec configuration should identify the channel assignment.

When it is desired to indicate an audio sampling rate greater than the value that can be represented in the samplerate field, the following may be used:

— an AudioSampleEntryV1 is used, which requires that the enclosing Sample Description Box also take the version 1;

— a Sampling Rate box may be present only in an AudioSampleEntryV1, and when present, it overrides the samplerate field and documents the actual sampling rate;

— when the Sampling Rate box is present, the media timescale should be the same as the sampling rate, or an integer division or multiple of it;

— the samplerate field in the sample entry should contain a value left-shifted 16 bits (as for AudioSampleEntry) that matches the media timescale, or be an integer division or multiple of it.

An AudioSampleEntryV1 should only be used when needed; otherwise, for maximum compatibility, an AudioSampleEntry should be used. An AudioSampleEntryV1 must not occur in a SampleDescriptionBox with version set to 0.

A TextSubtitleSampleEntry, TextMetaDataSampleEntry, or SimpleTextSampleEntry, all of which contain a MIME type, may be used to identify the format of streams for which a MIME type applies. A MIME type applies if the contents of a set of samples, starting with a sync sample and ending at the sample immediately preceding a sync sample, are concatenated in their entirety, and the result meets the decoding requirements for documents of that MIME type. Non-sync samples should be used only if that format specifies the behaviour of 'progressive decoding', and then the sample times indicate when the results of such progressive decoding should be presented (according to the media type).

NOTE The samples in a track that is all sync samples are therefore each a valid document for that MIME type.

In 8.5.2.2 add the subt and text cases to the SampleDescriptionBox

```
aligned(8) class SampleDescriptionBox (unsigned int(32) handler_type)
   extends FullBox('stsd', version, 0){
   int i ;
   unsigned int(32) entry_count;
   for (i = 1 ; i <= entry_count ; i++){
      switch (handler_type){
         case 'soun': // for audio tracks
            AudioSampleEntry();
            break;
         case 'vide': // for video tracks
            VisualSampleEntry();
            break;
         case 'subt': // for subtitle tracks
            SubtitleSampleEntry();
            break;
         case 'text': // for plain text tracks
            TextSampleEntry();
```



```
            break;
         case 'hint': // Hint track
            HintSampleEntry();
            break;
         case 'meta': // Metadata track
            MetadataSampleEntry();
            break;
         }
      }
   }
}
```

In 8.5.2.2 add the following after AudioSampleEntry

```
aligned(8) class SamplingRateBox extends FullBox('srat') {
   unsigned int(32) sampling_rate;
}

class AudioSampleEntryV1(codingname) extends SampleEntry (codingname){
   const unsigned int(16) audioentryversion = 1;
   const unsigned int(16) reserved = 0;
   const unsigned int(32) reserved = 0;
   template unsigned int(16) channelcount = 2;
   template unsigned int(16) samplesize = 16;
   unsigned int(16) pre_defined = 0;
   const unsigned int(16) reserved = 0 ;
   template unsigned int(32) samplerate = {suitable rate from timescale << 16};
   SamplingRateBox();   // optional but normally present
}

   // Timed Text Sequences

class PlainTextSampleEntry(codingname) extends SampleEntry (codingname) {
}

class SimpleTextSampleEntry(codingname) extends PlainTextSampleEntry (codingname) {
   string   content_encoding; // optional
   string   mime_format;
   BitRateBox ();             // optional
}

   // Subtitle Sequences

class SubtitleSampleEntry(codingname) extends SampleEntry (codingname) {
}

class XMLSubtitleSampleEntry() extends SubtitleSampleEntry ('stpp') {
   string   namespace;
   string   schema_location;    // optional
   string   auxiliary_mime_types;
                  // optional, required if auxiliary resources are present
   BitRateBox ();             // optional

}

class TextSubtitleSampleEntry() extends SubtitleSampleEntry ('sbtt') {
   string   content_encoding; // optional
   string   mime_format;
   BitRateBox ();             // optional
}
```


In 8.5.2.3 replace or add the following definitions:

`version` is set to zero unless the box contains an AudioSampleEntryV1, whereupon version must be 1

`SampleRate` when a SamplingRateBox is absent is the sampling rate; when a SamplingRateBox is present, is a suitable integer multiple or division of the actual sampling rate. This 32-bit field is expressed as a 16.16 fixed-point number (hi.lo)

`sampling_rate` is the actual sampling rate of the audio media, expressed as a 32-bit integer

`namespace` is a null-terminated field consisting of a space-separated list, in UTF-8 characters, of one or more XML namespaces to which the sample documents conform. When used for metadata, this is needed for identifying its type, e.g. gBSD or AQoS [MPEG-21-7] and for decoding using XML aware encoding mechanisms such as BiM.

`schema_location` is an optional null-terminated field consisting of a space-separated list, in UTF-8 characters, of zero or more URL's for XML schema(s) to which the sample document conforms. If there is one namespace and one schema, then this field shall be the URL of the one schema. If there is more than one namespace, then the syntax of this field shall adhere to that for xsi:schemaLocation attribute as defined by [XML]. When used for metadata, this is needed for decoding of the timed metadata by XML aware encoding mechanisms such as BiM.

`mime_format` - provides a MIME type, in null-terminated UTF-8 characters, which identifies the content format of the samples. Examples for this field include 'text/html' and 'text/plain'.

`auxiliary_mime_types` indicates the media type of all auxiliary resources, such as images and fonts, if present, stored as subtitle subsamples. If there is more than one mime_type, then this field shall be a space-separated list. This field is null-terminated in UTF-8 characters.

In 8.5.2.3 add before the end:

All string fields shall be null-terminated, even if unused. "Optional" means there is at least one null byte.

The namespace and schema_location are used both to identify the XML document content and to declare "brand" or profile compatibility. Multiple namespace identifiers indicate that the track conforms to the specification represented by each of the identifiers, some of which may identify supersets of the features present. A decoder should be able to decode all the namespaces in order to be able to decode and present correctly the media associated with this sample entry.

NOTE Additionally, namespace identifiers may represent performance constraints, such as limits on document size, font size, drawing rate, etc., as well as syntax constraints such as features that are not permitted or ignored.

Add to subclause 8.9.3.2 as follows before the definition of SampleGroupDescriptionBox:

```
abstract class SubtitleSampleGroupEntry (unsigned int(32) grouping_type) extends Sample-
GroupDescriptionEntry (grouping_type)
{
}

abstract class TextSampleGroupEntry (unsigned int(32) grouping_type) extends SampleGroup-
DescriptionEntry (grouping_type)
{
}
```

and add the subt and text cases to the SampleGroupDescriptionBox:

```
aligned(8) class SampleGroupDescriptionBox (unsigned int(32) handler_type)
   extends FullBox('sgpd', version, 0){
   unsigned int(32) grouping_type;
   if (version==1) { unsigned int(32) default_length; }
   unsigned int(32) entry_count;
   int i;
   for (i = 1 ; i <= entry_count ; i++){
      if (version==1) {
         if (default_length==0) {
            unsigned int(32) description_length;
         }
      }
      switch (handler_type){
         case 'vide': // for video tracks
            VisualSampleGroupEntry (grouping_type);
            break;
         case 'soun': // for audio tracks
```

