INTERNATIONAL STANDARD

ISO/IEC 23000-10

Second edition
2012-12-15

AMENDMENT 1
2014-08-15

Information technology — Multimedia application format (MPEG-A) —

Part 10: Surveillance application format

AMENDMENT 1: Conformance and reference software

Technologie de l'information — Format pour application multimédia (MPEG-A) —
Partie 10: Format pour application à la surveillance
AMENDEMENT 1

Reference number
ISO/IEC 23000-10:2012/Amd.1:2014(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.

ISO/IEC 23000-10:2012 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 — Multimedia application format (MPEG-A) —

Part 10:
Surveillance application format

AMENDMENT 1: Conformance and reference software

after 6.4.3.3 add the following two new clauses:

7 Conformance

7.1 File conformance

A file is said to be conforming when the following conditions are all satisfied:

— Brand consistency: the major_brand matches the declared brand name.

— Format conformance: the structure and syntax of the boxes included in the file accomplishes the conformance procedures defined below.

— Component conformance: all the included components in the file such as video data and audio data conform to the corresponding coding and file format specification.

— Additional boxes beyond of those required by this specification might be present but are not required to interpret conforming components correctly.

7.2 Parser conformance

A file parser is said to be conforming when the following condition is satisfied:

— The parser can correctly parse any conforming file and subsequently an appropriate player play the media components included in the file.

7.3 Creator conformance

A file creator (or authoring tool) is said to be conforming when the following condition is satisfied:

— The creator can produce files that a conforming player can play.

7.3 Conformance points

For the file format, the following conformance points shall be checked for conformance testing.

C1 – conformance point 1

— Check that the major_brand identifier is 'sff1'.

— Check that there is at least one fragment contained.

— Check that at least one metadata box is present

  — at file level and

  — for each media track at track level.

C2 – conformance point 2

— Check that each “AF Identification Box” contains a valid UUID identifying the fragment.

— Check that each “AF Identification Box” contains a predecessor UUID referencing the predecessor fragment or the current fragment if no predecessor exists.

— Check that each “AF Identification Box” contains a successor UUID referencing the successor fragment or the current fragment if no successor exists.

— Check that the “Camera/microphone Identification Box” for each track contains a valid UUID.

C3 – conformance point 3

— Check that all fragments use the same number of tracks.

— Check that for each media track a timed metadata track exists linked using a track reference of type ‘vsmd’ for video tracks and ‘asmd’ for audio tracks.

— Check that media tracks using identical media coding parameters exist in all fragments.

— Check that all media tracks for each fragment start with a random access point.

— Check that each fragment start with an instantaneous decoding refresh access unit if the sample entry type avc1 is present.

C4 – conformance point 4

— Check that the duration parameter given in the “AF Identification Box” coincides with the duration parameter in the “Movie Header Box”.

— Check that boxes of version “1” being used for “Track Header Box”, “Media Header Box” and SurveillanceMetadataSampleConfigBox.

— Check that following box types are not used: shadow sync sample, SampleGroupBox (‘sbgp’, ‘sgpd’), SampleGroupDescriptionBox, ‘subs’, ‘stsl’, ‘imif’, ‘ipmc’, ‘avcp’, ‘iods’, ‘mp4v’, ‘mp4s’, MPEG4BitRateBox, and MPEG4ExtensionDescriptorsBox.

7.5 Conformance files

Table 1 — Conformance files with conformance points

	File name	Conformance points
No		
1	Saf_C1C2_conf	C1 and C2
2	Saf_C1C2C3C4_conf	C1 and C2 and C3 and C4

8 Reference software description

The reference software is split into three separate modules. A creator module allows for compliant file generation, a parser module interprets given files and a framework module embraces the core libraries required.

8.1 Architecture

The architecture of the software is depicted in Figure 1. The software has been designed as standalone implementation, independent on any SDK and additional tools, and multiplatform capable of compilation. The design principle used is object oriented programming.

Each box derived from the framework has an auto validity check.

8.1.1 SAF Framework

A generalized function module, configurable and extendible.

Main features provided are:

— Instantiation of boxes conforming to the ISO Base Media FF and the Surveillance AF

— DOM structured containers

— Validity checking of all levels in the DOM

— Functionality for inserting and removing boxes, containers or root containers (DOM tree size is automatically adapted)

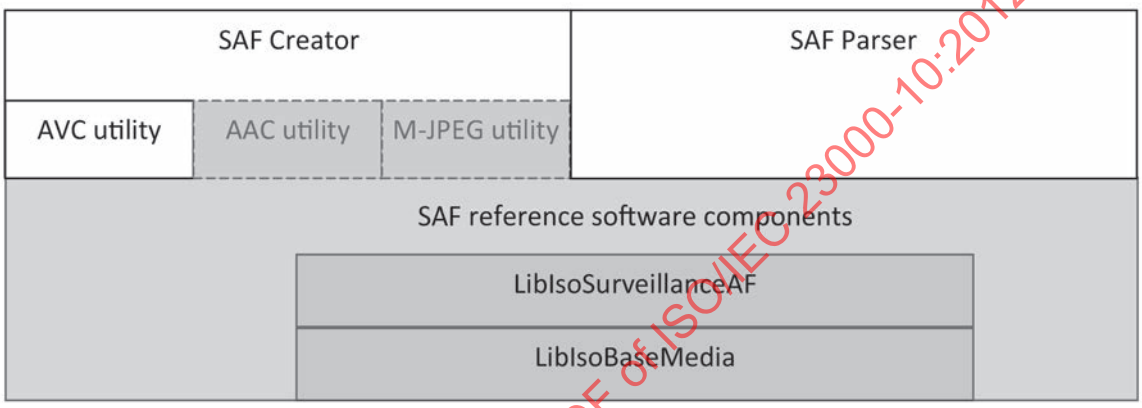

Figure 1 — SAF reference software components

8.1.1.1 LibIsoBaseMedia

The library provides items and containers conforming to the ISO Base Media FF specification. Additionally it provides for:

— Basic item check

— Visualization of the container-box DOM structure

— error report mechanism and error concealment (can be switched on and off)

NOTE The library files and items use the prefix ISOBMF for name convention. The individual names are generally identical to those from the related standard.

8.1.1.2 LibIsoSurveillanceAF

The library provides items and containers conforming to this specification. Features provided by LibIsoBaseMedia are preserved.

8.1.1.3 Utilities

Utilities are used to create more specific boxes or customized boxes externally derived from ISO Media Base file format, and can simply use the framework to insert and remove boxes from the container. AAC utility and M-JPEG utility illustrate extensions to handle other media formats.

8.1.1.3.1 AVC Utility

Parses a given AVC media stream and:

— creates parameter sets for avcC in MediaDescriptionBox of stbl

— extracts slice NAL units for sample generation

— extracts picture types for creating random access point for SAF fragment

— creates timing for CompositionOffsetBox

8.2 SAF Creator

The creator generates Surveillance AF conformant files, it is designed as console command application. As media data input the creator accepts AVC elementary video streams of Baseline, Main and High profile to generate SAF fragment files.

A timestamp meta data track for each media track is created and referenced automatically as well as fragment UUIDs.

Configuration information for the creation is placed in a separate file which should be customized according to the media data provided.

The duration setting indicates if multiple files or a single file is generated. If the duration is set shorter than the media data to be packaged, multiple files will be the outcome; each with a duration as specified by the user. If the duration is set longer than the media data to be packaged, a single file will be the outcome and the duration element will provide for the correct duration information. The actual length of a fragment will vary slightly as every SAF fragment has to start at a random access point. If the media stream does not provide for multiple random access points only a single SAF fragment will be created.

If not provided by the user, explicitly default configuration settings will be applied by the creator module, i.e. system time is used to generate startTime and the timed metadata track.

```
## default configuration of SAF creator
## Case sensitive

[SAF_CONFIG]

## file level

# Length in minutes. This value, however, will just be used approximately. The exact length depends
# on what time the next available IDR picture.

AFFragmentDuration   = 5          # in minutes

OutputBaseName       = saf        # files generated will be saf000.mp4 saf001.mp4 ...

StartTime            = now        # in UTC time format e.g. 2013-02-28T18:25:30.000Z or now



[TRACK_0]

ESFileName           = foreman_cif.264

FrameRate            = 25.0           # float point number of frames per second

Timescale            = 90000          # Hz

SamplesPerChunk      = 30             # specifies how many samples (frames) should be contained in
each chunk



[TRACE]

TraceFile        = null           # null = console window, otherwise use file name with valid path

TraceLevel       = 3              # 0 = On ERROR, 1 =On WARNING, 2 = On VERBOSE, 3 = Output ALL
```

Figure 2 — SAF creator configuration file example

SAF_CONFIG

AFFragmentDuration – length of the SAF fragment in minutes

StartTime – UTC or now

TRACK_0

ESFileName – input file, path and name of an elementary stream file

FrameRate/Timescale – the values will be used to emulate the timestamp value for both SampleTable and timestamp meta data track

TRACE

TraceFile – trace information might be collected in the specified file

A published interface for SAFCreator is: ISOSAF_mp4creator.cpp and the struct ISOSAF_FileContext , use of the configuration file is just a way to demonstrate how to fill out the ISOSAF_FileContext structure, and initialize the application.

```
#include "ISOSAF_mp4creator.h"

#include "tool_inifile.h"

#include "avc_handler.h"

... code sample

{

        ISOSAF_FileContext fileContext;

        ZERO( fileContext );

        CIniFile * cfg = CIniFile::GetInstance( argv[1] );

        if( !cfg )

         urn -2;



        CAVCHandler avcHandler;

        CISOSAF_MP4Creator * mp4Creator = initEngine( *cfg, fileContext );

        if( mp4Creator && avcHandler.open(fileContext, mp4Creator) ) {

         cHandler.run();



        else {

         rintf( stdout, "Failed to create with: %s\n", argv[1] );

        }

}
```

Figure 3 — Initialization of SAFCreator code sample

8.3 SAF Parser

The parser module is able to parse input files for conformance to this specification and is designed as console command application.

Parsing of concatenated fragments and files using the UUID mechanism is supported if URIs are provided. Provides interface for application to receive audio and video frame packets in an interleaved manner. All AV-Packets contain timestamp information. Replay functionality of the media content of the file can be achieved by usage of available player.

The parser accepts any mp4 container file as input. The output produced might be displayed in a console window or redirected to a file. In case of a parsing error or a conformance test failure reporting information is generated as specified below

a) The major_brand identifier is not ‘sff1’.

b) No AF fragment contained.

c) Condition of at least one metadata box at file level and each at track level is not fulfilled.

d) “AF Identification Box” validity check failure.

e) Not all fragments use the same number of tracks.

f) “Camera/microphone Identification Box” for each track contains an invalid UUID.

g) Condition not fulfilled for each media track a timed metadata track exists linked using a track reference of type ‘vsmd’ for video tracks and ‘asmd’ for audio tracks.

h) Condition not fulfilled that media tracks using identical media coding parameters exist in all fragments.

i) Condition not fulfilled that all media tracks for each fragment start with a random access point.

j) “AF Identification Box” coincides with the duration parameter in the “Movie Header Box” for different duration.

k) “Track Header Box”, “Media Header Box” and SurveillanceMetadataSampleConfigBox does not have box version 1.

l) Any of following boxes appears: shadow sync sample, SampleGroupBox (‘sbgp’, ‘sgpd’), SampleGroupDescriptionBox, ‘subs’, ‘stsl’, ‘imif’, ‘ipmc’, ‘avcp’, ‘iods’, ‘mp4v’, ‘mp4s’, MPEG4BitRateBox, and MPEG4ExtensionDescriptorsBox.

m) Not each fragment start with an instantaneous decoding refresh access unit if the sample entry type avc1 is present.

n) Sample entry type avc2 is used.

```
#include "ISOSAF_filehandle.h"

#include "Test_fragment.h"

#include "stdlib.h"

#include "ISOBMF_avpacket.h"

string message;

CISOSAF_FileHandle * handle = CISOSAF_FileHandle::open( “file_location”, messsage );

if( handle ) {

    handle->close();

        delete handle;

}

else {

    fprintf( stdout, "load failed: %s\n", message.c_str() );

}
```

Figure 4 — Initialization of SAFParser code sample