INTERNATIONAL STANDARD

ISO/IEC 23000-12

First edition
2010-07-15

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

Part 12: Interactive music application format

Technologies de l'information — Format pour application multimédia (MPEG-A) —

Partie 12: Format d'application musicale interactive

Reference number
ISO/IEC 23000-12:2010(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

Contents

Page

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 23000-12 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 23000 consists of the following parts, under the general title *Information technology — Multimedia application format (MPEG-A)*:

— *Part 1: Purpose for multimedia application formats* [Technical Report]

— *Part 2: MPEG music player application format*

— *Part 3: MPEG photo player application format*

— *Part 4: Musical slide show application format*

— *Part 5: Media streaming application format*

— *Part 6: Professional archival application format*

— *Part 7: Open access application format*

— *Part 8: Portable video application format*

— *Part 9: Digital Multimedia Broadcasting application format*

— *Part 10: Video surveillance application format*

— *Part 11: Stereoscopic video application format*

— *Part 12: Interactive music application format*

Introduction

The market for interactive music service which provides users with an experience of recomposing the music to their taste is getting formed and matured. Hence, a standardized form needs to be specified to guarantee the interoperability of the interactive music contents in order to broaden and to boost the interactive music service market.

Interactive music application format (IM AF) defines a file format designed for interactive music service. It specifies how to combine the multiple audio tracks before conventional mixing process, with associated information for a presentation in a well-defined format that facilitates storage, interchange, management, editing, and presentation of interactive music contents in interoperable ways.

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

Part 12: Interactive music application format

1 Scope

This part of ISO/IEC 23000 specifies a file format designed for interactive music services. It integrates the multiple audio tracks with appropriate additional information for enabling users to experience various preset mixes and to make their own mixes complying with interactivity rules imposed by the producer.

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 10918-1:1994, *Information technology — Digital compression and coding of continuous-tone still images: Requirements and guidelines*

ISO/IEC 11172-3:1993, *Information technology — Coding of moving pictures and associated audio for digital storage media at up to about 1,5 Mbit/s — Part 3: Audio*

ISO/IEC 14496-3:2009, *Information technology — Coding of audio-visual objects — Part 3: Audio*

ISO/IEC 14496-12:2008, *Information technology — Coding of audio-visual objects — Part 12: ISO base media file format*

ISO/IEC 15938-5:2003, *Information technology — Multimedia content description interface — Part 5: Multimedia description schemes*

ISO/IEC 23003-2, *Information technology — MPEG audio technologies — Part 2: Spatial Audio Object Coding (SAOC)*

3GPP TS 26.245, Version 6.1.0, *Transparent end-to-end Packet switched Streaming Service (PSS) — Timed text format*

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

3.1
track
collection of related samples in an IM AF file

NOTE For media data, a track corresponds to a sequence of images or sampled audio and an audio track is either mono or stereo. For hint tracks, a track corresponds to a streaming channel.

4 Overview

4.1 General

This clause provides necessary information on the content, the organization and the playback of an IM AF file.

4.2 File contents and organization

Creating an IM AF file involves formatting different types of media data, especially multiple audio tracks with interactivity data and storing them into an ISO-Based Media File Format (ISO-BMFF).

An IM AF file is composed of:

- multiple audio tracks which represent the several parts (e.g. instruments and/or voices) of a song (i.e. a musical piece);
- groups of audio tracks which define a hierarchical structure of audio tracks (e.g. all guitars of a song can be gathered in a same group);
- preset data which is pre-defined mixing information on multiple audio tracks (e.g. karaoke and rhythmic version);
- rules which introduce specific data related to user's interaction (e.g. definition of action allowed concerning audio tracks/groups selection and audio volume control);
- additional media can be used to enrich the user's interaction space (e.g. timed text synchronized with audio tracks which can represent the lyrics of a song, images related to the song, to the music album and/or to the artist);
- metadata can be used in the same goal (e.g. data used to describe the song, the music album and/or the artist).

4.3 Usage model

By means of an interactive music player, users can listen and handle audio tracks that compose an IM AF file (and eventually the other media and metadata). Two possible mix modes are defined for interaction and playback:

- the Preset-mix mode;
- the User-mix mode.

In Preset-mix mode, user selects one preset among the presets stored in the IM AF file then audio tracks are mixed according to preset parameters associated with selected preset.

In User-mix mode, user selects/unselects the audio tracks/groups and/or controls the volume of each of them. Each user's interaction is analyzed in order to determine compatibility with rules defined by the producer and/or the artist. This compatibility is verified by means of a rule analyzer (see Figure 1). A detailed description of this checking is provided in Annex A.

Figure 1 shows the block diagram for both the Preset-mix mode and the User-mix mode. The preset selection, the audio track/group selection and the volume change are the possible user's interaction.

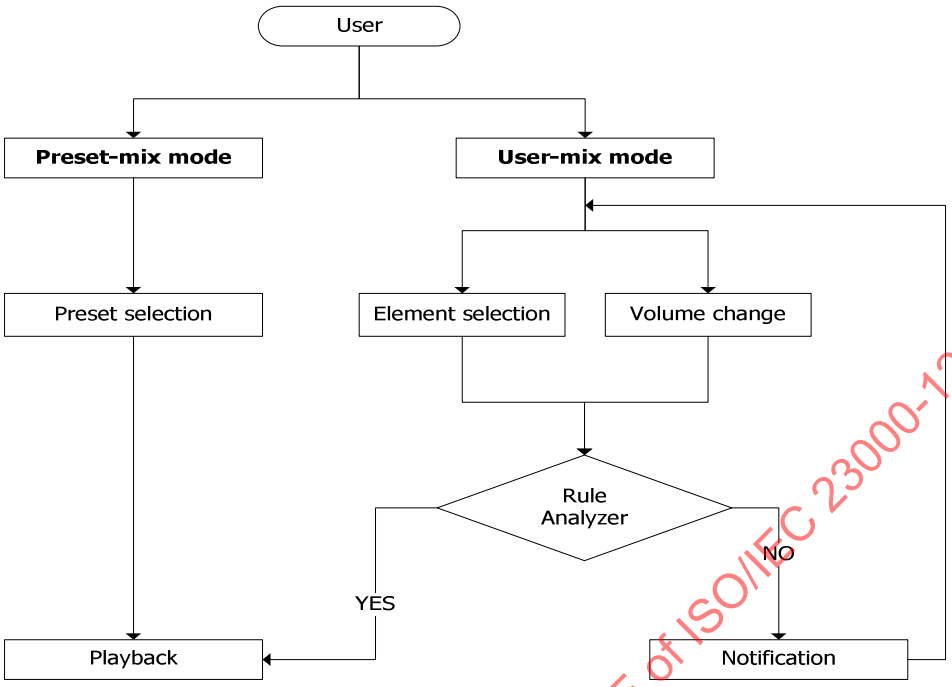

Figure 1 — The illustration of IM AF usage mode

5 Supported components of Interactive Music AF

In Table 1, all the supported components of IM AF are listed with the specification reference of the respective component. Note that IM AF does not mandate all the syntax and semantics of the 'Specification' column of Table 1. Instead, it does mandate the component specification with some restrictions on usages, which are defined in the specification of 'Restriction' column of the table.

IM AF file includes various kinds of audio stream such as MP3, AAC, SAOC and PCM. Also it includes JPEG for image, 3GPP Timed Text for text and MPEG-7 MDS for metadata. In this specification, ISO-BMFF is used as the base file format structure.

Table 1 — Supported components of IM AF

Type	Component Name	Abbreviation	Specification	Restriction
File Format	ISO Base Media File Format	ISO-BMFF	ISO/IEC 14496-12:2008	Defined in sub-clause 6.1
Audio	MPEG-1 Audio Layer III	MP3	ISO/IEC 11172-3:1993	None
	MPEG-4 Audio AAC profile	AAC	ISO/IEC 14496-3:2005	None
	MPEG-D SAOC Baseline profile	SAOC	ISO/IEC 23003-2:200x	None
	PCM	PCM	-	None
Image	JPEG Image	JPEG	ISO/IEC 10918-1	Baseline coding
Text	3GPP Timed Text	3GPP TT	3GPP TS 26.245	None
Metadata	MPEG-7 Multimedia Description Scheme	MDS	ISO/IEC 15938-5:2003	Defined in sub-clause 6.9

Note that audio shall be done in accordance with MP3 (ISO/IEC 11172-3), AAC (ISO/IEC 14496-3) and SAOC (ISO/IEC 23003-2). And requirements of JPEG (ISO/IEC 10918-1) apply for image in IM AF.

6 File Structure

6.1 Table for boxes

The file format of IM AF can be represented in two kinds of structures for single IM AF content and multiple IM AF contents. The file format structure of this specification is derived from the ISO-BMFF standard. Table 2 shows the structure of the boxes and their description. The mandatory boxes are marked with an asterisk (*).

Unless specifically mentioned in this specification, the requirements of ISO-BMFF (ISO/IEC 14496-12) apply for the listed boxes.

Table 2 — Table for boxes of IM AF

*	ftyp						*file type and compatibility*
*	moov						*container for all the metadata*
		mvhd					*movie header, overall declarations*
		trak					*container for an individual track or stream*
*			tkhd				*track header, overall information about the track*
			tref				*track reference container*
			edts				*edit list container*
				elst			*an edit list*
*			mdia				*container for the media information in a track*
*				mdhd			*media header, overall information about the media*
*				hdlr			*handler, declares the media (handler) type* *“soun” for audio data* *“text” for timed text data* *“hint” for protocol hint track*
*				minf			*media information container*
					smhd		*sound media header, overall information (sound track only)*
					hmhd		*hint media header, overall information (hint track only)*
					nmhd		*Null media header, overall information (some tracks only)*
*					dinf		*data information box, container*
*						dref	*data reference box, declares source(s) of media data in track*
*					stbl		*sample table box, container for the time/space map*
*						stsd	*sample descriptions (codec types, initialization etc.)*
*						stts	*(decoding) time-to-sample*
*						stsc	*sample-to-chunk, partial data-offset information*
						stsz	*sample sizes (framing)*
						stz2	*compact sample sizes (framing)*
*						stco	*chunk offset, partial data-offset information*
						co64	*64-bit chunk offset*
		grco					*container for the groups*
			grup				*group box, describes the structure (hierarchy)*
*		prco					*container for the presets*
*			prst				*preset box, container for the preset information*
		ruco					*container for rules*
			rusc				*selection rule box, container for a selection rule*
			rumx				*mixing rule box, container for a mixing rule*
	mdat						*media data container*
	free						*free space*
	skip						*free space*
	meta						*Metadata*
*		hdlr					*handler, declares the metadata (handler) type*
		dinf					*data information box, container*
			dref				*data reference box, declares source(s) of metadata items*
		iloc					*item location*
		iinf					*item information*
		xml					*XML container*
		bxml					*binary XML container*
		pitm					*primary item reference*

6.2 File structure for a single type file

Figure 2 illustrates a single type file structure of IM AF which is containing single movie presentation with associated data. This type of file structure mainly consists of `ftyp`, `moov` and `mdat` boxes. The `moov` box describes the presentation of the scene in which more than one `trak` box are contained. The `trak` box contains the presentation description for one media. A media in each `trak` box can be individual audio, image and text as shown in Figure 2. The `trak` box supports time information for the synchronization with other media of other `trak` box. The `mdat` box contains the real contents which are described in the `trak` box or the `trak` box may import the content by URL without deposing the content in the `mdat` box.

Besides this type of file structure includes `grco`, `prco` and `ruco` boxes. The `grco` box contains zero or more `grup` boxes and each `grup` box describes the hierarchy structure of a single group which is composed of one or more audio tracks and/or groups. The `prco` box contains one or more `prst` boxes which describe the predefined mixing information. The `ruco` box contains zero or more `rusc` boxes and/or `rumx` boxes which describe the interactivity rules related to selection and/or mixing of audio tracks contained in the file.

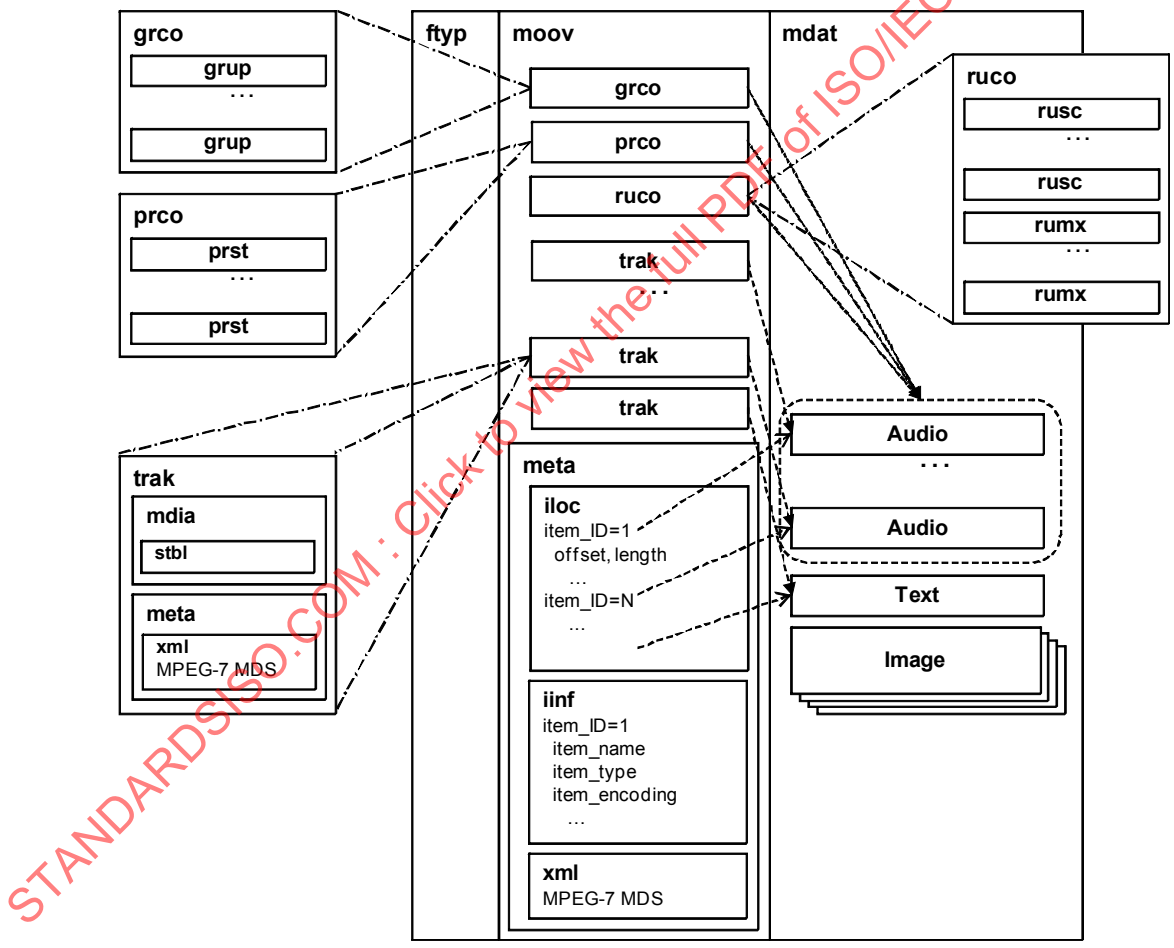

Figure 2 — A single type file structure with meta box

6.3 File structure for a multiple type file

A multiple type file structure of IM AF supports the album functionality i.e. collection of several single type files in one single file. In Figure 3, the multiple type file structure of IM AF contains multiple movie presentations with associated data. There are two layers: outer file format and inner file format. Outer file format consists of the `ftyp`, `meta` and `mdat` boxes and inner one which consists of `moov`, `meta` and `mdat` boxes. The information of the items in the outer and inner file format can be described by the `iloc` and `iinf` boxes. In the multiple type of IM AF, `item_ID`s in the outer `iloc` box shall indicate the hidden `moov` boxes and `item_ID`s in the inner `iloc` box shall indicate the items in each hidden `moov` box.

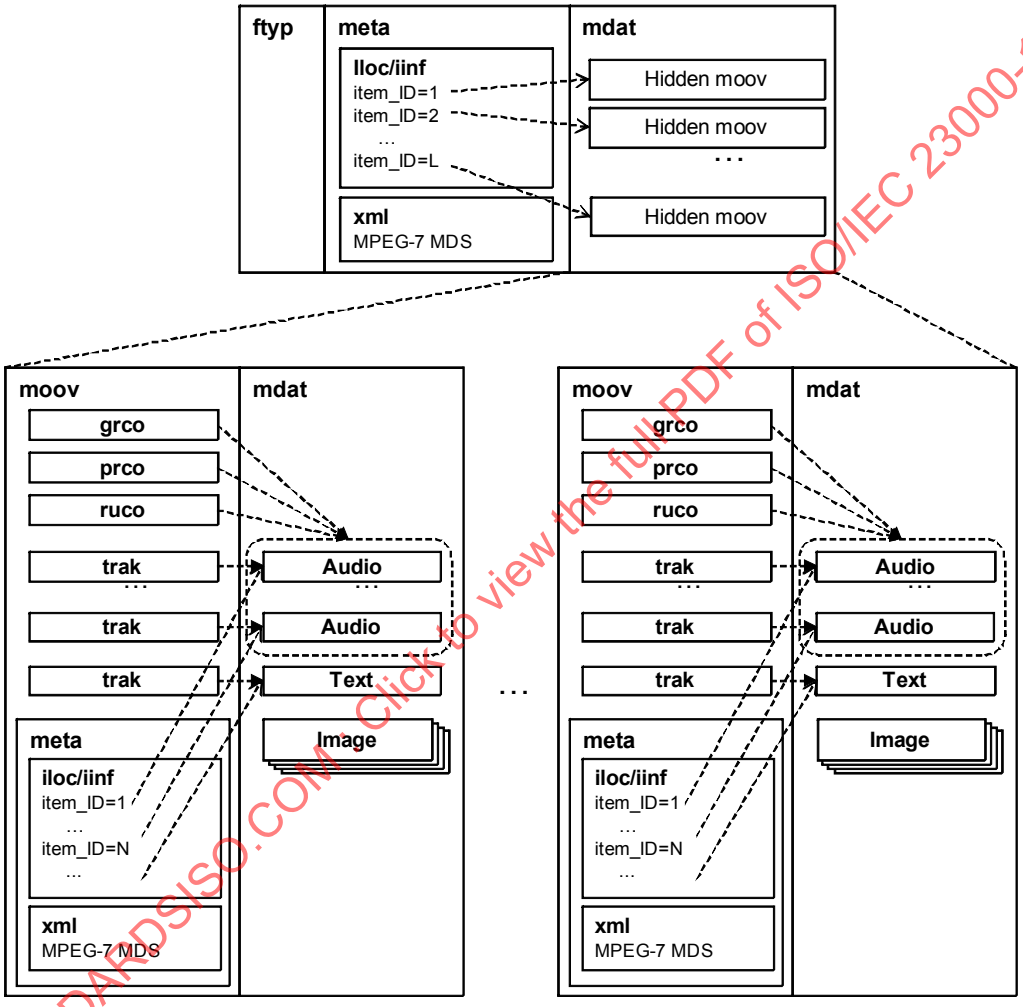

Figure 3 — A multiple type file structure with `meta` box

6.4 Backward compatibility to legacy music player

For legacy music players or devices that are not capable of simultaneous decoding the multiple audio tracks, the AR (All Recorded) audio track stored in IM AF file can still be played. In order to distinguish the AR audio track from the audio tracks for interactive music service, `flag` field of `tkhd` box is used. For the AR audio track, the `flag` is set as 'Track_enabled' whose value is 0x000001 whereas the `flag` of the audio track for interactive music service is set as 'Track_disabled' whose value is zero. Therefore, only the audio track with 'Track_enabled' is decoded by the legacy player whereas the IMAF player can enable (in the process of playing) any disabled tracks.

6.5 Hierarchical structure of audio tracks

6.5.1 Introduction

At the structural level, two kinds of elements are introduced: audio tracks and groups. Hence, a song stored in an IM AF file is composed of a set of audio tracks and groups.

The audio track is the atomic element of an IM AF file. It represents a sound signal identified as a part of a song. For instance, an audio track might represent an instrument version, a voice, several instruments or voices mixed together.

The group is a composition object. It might be composed of a set of audio tracks and/or groups. Thus, the groups allow defining hierarchical structures. Their name and their number are determined by the producer. For instance, a group can gather several audio versions of the bass guitar in a song.

In groups, each element contained is ordered. This order is defined by the producer by choosing the places of the elements in the hierarchy defined by the group structure.

Each element has a state represented by a boolean value. Hence, an element can take two values: active or inactive. When an audio track is played, its state is active otherwise its state is inactive.

The definition of the state for a group is slightly different. A group can have several hierarchical levels. Each of them represents the children, the sub-children, the sub-sub-children, etc. of the root node represented by the initial group. Regardless, each hierarchical level can contain audio tracks and groups. Thus, a group is in active state when at least one of the sub-elements of the first hierarchical level (i.e. the directs children of the group) is in this state. A group is in inactive state when all the sub-elements of the first hierarchical level are in this state.

The order combined with the `group_activation_mode` (see group box in 6.5.4) allow defining which elements of a group shall be put in active state when the group is switched to this state.

Each element also has a reference volume, which is the volume gain that has to be used when the element is switched on. This value is also used to compute reference ratios for equivalence, upper and lower rules (see 6.7.3).

Figure 4 shows the audio tracks of a same song gathered in four groups: Bass_Guitar, Drums, Keyboard and Lead_Guitar. In this example each of these groups contains several versions of the corresponding audio tracks: jazz, reggae and rocksteady.

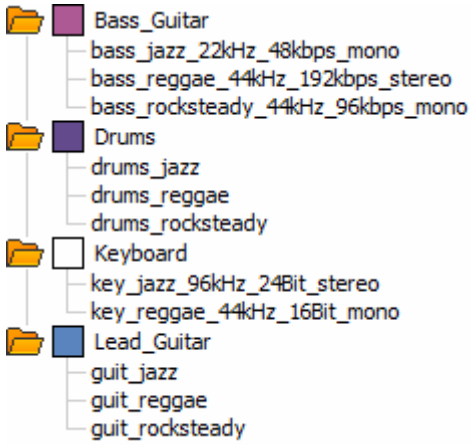

Figure 4 — Song composed of several groups

6.5.2 Tracks and groups identifiers

A group is composed of audio track(s) and/or group(s). In IM AF file, the groups contain the IDs (see `element_ID` in 5.4.4) of the elements contained. Since a group contain audio tracks and groups, `element_ID` shall be ID of tracks (`track_ID`) and/or ID of groups (`group_ID`).

In ISO-FF specification, a `track_ID` shall take 32-bit unsigned integer. In IM AF, a `track_ID` shall be represented from 0x00000001 to 0x7FFFFFFF and a `group_ID` shall be represented from 0x80000000 to 0xFFFFFFFF. Hence, elements contained in `element_ID` are recognized as a `track_ID` or as a `group_ID` according to the Most Significant Bit (MSB) of the element ID.

6.5.3 Specific usage of `trak` box

Since the IM AF file format is based on the ISO-BMFF standard, the Track Header Box is used (see `trak` box in Table 2). In the case of IM AF, usage of some fields such as `track_ID` and `volume` are limited. The `track_ID` is used in accordance with the details in 6.5.2. The `volume` is mandatory and it used as a reference volume for the tracks and the represented volume is included in the interval from 0.00 to 4.00.

6.5.4 Group Container Box

The following is the definition of the '`grco`' box which defines the number of groups contained in an IM AF file.

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

Syntax

```
aligned(8) class GroupContainerBox extends Box('grco') {
   unsigned int(16) num_groups;
}
```

Semantics

`num_groups` – is an integer that specifies the number of groups contained in the Group Container Box.

6.5.5 Group Box

The following is the definition of '`grup`' box which contains specific information on a single group. This box details the data that compose a particular group.

Box Type: 'grup'
Container: Groups Container Box ('grco')
Mandatory: No
Quantity: Zero or more

Syntax

```
aligned(8) class GroupBox extends FullBox('grup', version = 0, flags){
    unsigned int(32) group_ID;
    unsigned int(16) num_elements;
    unsigned int(32) element_ID[num_elements];
    unsigned int(8) group_activation_mode;
    if(group_activation_mode==2){
       unsigned int(16) group_activation_elements_number;
    }
    int(16) group_reference_volume;
    string   group_name;
    string   group_description;
}
```

Semantics

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

`flags` – is an 8 bits integer with flags; the following values are defined:

display_disable & edit_disable indicates that the group information contained in the Group Box is disabled to display on the player and to be edited by user. In this case, only group_name is displayed. Flag value is 0x01.

display_enable & edit_disable indicates that the group information contained in the Group Box is enabled to display on the player, but it is disabled to be edited by user. Flag value is 0x02.

Flags	Display	Edit
0x01	disable	disable
0x02	enable	disable

`group_ID` – is an integer that uniquely identifies the group.

`num_elements` – is an integer that indicates the number of elements involved in the group.

`element_ID[num_elements]` – is an array where elements are IDs of elements contained in this group. In this array, the order of elements is the same that the order of elements in the hierarchical structure defined by the producer. The channel type of all audio tracks involved in a group shall be identical (see `output_channel_type` in 6.6.3).

`group_activation_mode` – is a flag that defines the way the elements of a group shall be activated when a group is switched on.

group_activation_mode	Meaning
0	Try to switch on the minimum number of elements that can be played. If there is min/max rule in the IM AF file, minimum number of played elements shall be the same as the `min_num_elements` of min/max rule. If there is no min/max rule in the IM AF file, default min value shall be 0 .
1	Try to switch on the maximum number of elements that can be played. If there is min/max rule in the IM AF file, maximum number of played elements shall be the same as the `max_num_elements` of min/max rule. If there is no min/max rule in the IM AF file, default max value shall be the number of elements in the group.
2	Try to switch on the producer defined number of elements If there is min/max rule in the IM AF file, `the group_activation_elements_number` is in the range of min/max number of that rule.

`group_activation_elements_number` – is an integer that indicates the number of elements which shall be switched on when the group is activated. This number, according to the order of the elements in `element_ID`, allows determining which elements will be switched on (used when `group_activation_mode`=2). Its default value takes 0.

`group_reference_volume` – is a fixed 8.8 value specifying the reference group's audio volume (this is the volume gain that shall be applied on this group when it is switched on). A group reference volume will be included in the interval 0.00 to 4.00.

`group_name` – is a null-terminated string in UTF-8 characters which gives a human-readable name of the Group Box.

`group_description` – is a null-terminated string in UTF-8 characters which gives a human-readable brief description of the Group Box.

6.6 Preset Information

6.6.1 Introduction

One or more presets included in IM AF support a user to recompose the music with easy and convenient way. Some examples of preset are given below:

- general preset: a preset composed of multiple audio tracks by music producer;

- karaoke preset: a preset composed of multiple audio tracks except vocal tracks;
- acappella preset: a preset composed of vocal and chorus tracks.

Especially for limited interface device such as car-audio, mp3 player and mobile phone, presets function appears more helpful. Users simply select the preset then they listen and enjoy the music to their taste.

6.6.2 Specific usage of `dinf` box

In the case of IM AF, `dinf` box inside `prco` box shall contain objects that declare the location of the IM AF file where audio tracks involved in the preset are stored.

6.6.3 Specific usage of `dref` box

In the case of `dinf`/`dref` box stored in `prco` box in IM AF, it is recommended that the data reference that declare the location of the IM AF file where audio tracks involved in the preset are stored is described as URN, especially ISRC (International Standard Recording Code) which contained `moov`/`meta` box of IM AF file where audio tracks involved in the preset are stored.

If the flag is set indicating that audio tracks involved in the preset are stored in the same file, then no string (not even an empty one) shall be supplied in the entry field.

6.6.4 Preset Container Box

The following is the definition of '`prco`' box, which contains general information on presets.

Box Type: 'prco'
Container: Movie Box ('moov')
Mandatory: Yes
Quantity: Exactly one

Syntax

```
aligned(8) class PresetContainerBox extends Box('prco') {
   unsigned int(8) num_preset;
   unsigned int(8) default_preset_ID;
}
```

Semantics

`num_preset` – is the number of presets in Preset Container Box

`default_preset_ID` – is an integer which indicates the `preset_ID` activated at initial condition without user interaction. Its default value takes the smallest `preset_ID` in the file. If this field set to be 0 and the SAOC bitstream is contained inside file, the first preset stored in SAOC is activated initial condition without user interaction.

6.6.5 Preset Box

The following is the definition of '`prst`' box, which contains specific information on a single preset.

Box Type: 'prst'
Container: Preset Container Box ('prco')
Mandatory: Yes
Quantity: One or more

Syntax

```
aligned(8) class PresetBox extends FullBox('prst', version = 0, flags){
   unsigned int(8)   preset_ID;
   unsigned int(8)   num_preset_elements;
   unsigned int(32)  preset_element_ID[num_preset_elements];
   unsigned int(8)   preset_type;
   unsigned int(8)   preset_global_volume;
   if(preset_type == 0){
       for(i=0; i<num_preset_elements; i++){
           unsigned int(8) preset_volume_element;
       }
   }
   if(preset_type == 1){
       unsigned int(8)   num_input_channel[num_preset_elements];
       unsigned int(8)   output_channel_type;
       for (i=0; i<num_preset_elements; i++){
           for (j=0; j<num_input_channel[i]; j++){
               for (k=0; k<num_output_channel; k++){
                   unsigened int(8)   preset_volume_element;
               }
           }
       }
   }
   if(preset_type == 2)){ // dynamic track volume preset
       unsigned int(16)   num_updates;
       for(i=0; i<num_updates; i++){
           unsigned int(16)   updated_sample_number;
           for(j=0; j<num_preset_elements; j++){
                unsigned int(8) preset_volume_element;
           }
       }
   }
   if(preset_type == 3){ // dynamic object volume preset
       unsigned int(16)   num_updates;
       unsigned int(8)   num_input_channel[num_preset_elements];
       unsigned int(8)   output_channel_type;

       for(i=0; i<num_updates; i++){
           unsigned int(16)   updated_sample_number;
           for(j=0; j<num_preset_elements; j++){
               for(k=0; k<num_input_channel[j]; k++){
                   for(m=0; m<num_output_channel; m++){
                       unsigned int(8)   preset_volume_element;
                   }
               }
           }
       }
   }
   string   preset_name;
}
```

Semantics

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

`flags` – is an 8 bit integer with flags; the following values are defined:

display_disable & edit_disable indicates that the preset information contained in the Preset Box is disabled to display on the player and to be edited by user. In this case, only `preset_name` is displayed. Flag value is 0x01.

display_enable & edit_disable indicates that the preset information contained in the Preset Box is enabled to display on the player, but it is disabled to be edited by user. Flag value is 0x02.

display_enable & edit_enable indicates that the preset information contained in the Preset Box is enabled to display on the player and to be edited by user. Flag value is 0x03.

flags	Display	Edit
0x01	disable	disable
0x02	enable	disable
0x03	enable	enable

`preset_ID` – is an integer that uniquely identifies the preset. The `preset_ID`s are never re-used and cannot be zero.

`num_preset_elements` – is an integer that gives the number of elements involved in the preset.

`preset_element_ID[num_preset_elements]` – is an array whose element is the `element_ID` involved in the preset.

`preset_type` – is an integer that indicates the preset type.

Static track volume preset has the *time invariant* volume information related to each *track* involved in the preset. In this case, the output channel type is the same as channel type of the track which has the largest number of channels among tracks involved in the preset. Type value is 0.

Static object volume preset has the *time invariant* volume information related to each *object* which is individual channel (i.e. mono) of the track involved in the preset. Type value is 1.

Dynamic track volume preset has the *time variant* volume information related to each *track* involved in the preset. In this case, the output channel type is the same as channel type of the track which has the largest number of channels among tracks involved in the preset. Type value is 2.

Dynamic object volume preset has the *time variant* volume information related to each *object* which is individual channel (i.e. mono) of the track involved in the preset. Type value is 3.

preset_type	Meaning
0	static track volume preset
1	static object volume preset
2	dynamic track volume preset
3	dynamic object volume preset

`preset_global_volume` – is an integer which indicates the playback volume gain of preset output signal. The quantization table for playback volume gain is below:

index	0	1	2	3	…	149	150
value(ratio)	0	0.01	0.02	0.03	…	1.49	1.50

`preset_volume_element` – is an integer which indicates the playback volume gain of each audio track or object. The quantization table for playback volume gain is below:

Index	0	1	2	3	…	199	200
value(ratio)	0	0.02	0.04	0.06	…	3.98	4.00

`num_input_channel[i]` – is an integer that is the number of channels of the audio track related to `i`-th element of `preset_element_ID[•]`, which is `ChannelCount` in `trak/media/minf/stbl/stsd`.

`output_channel_type` – is an integer that defines the rendering configuration.

output_channel_type	Meaning	num_output_channel
0	mono	1
1	stereo	2
2	5 channels	5

`num_output_channel` – is an integer that indicates the number of output channels decided by `output_channel_type` in the above Table.

`num_updates` – is an integer that gives the number of updates on `preset_volume`.

`updated_sample_number` – is an integer that indicates the sample number when the `preset_volume` is updated.

`preset_name` – is a null-terminated string in UTF-8 characters which gives a human readable name for the preset.

6.7 Interactivity Rules

6.7.1 Introduction

When such an IM AF file is rendered on an IM AF player, users can select the audio tracks that they want to listen and also adjust the volumes of the different audio tracks. These choices are fitted by rules which are defined by the producer at the creation time of an IM AF file.

These rules allow determining user's interaction where goal is to fit the artistic creation. Their definition is optional and so not imposed by the format.

IM AF defines two kinds of rules which are applied onto both selection and mixing of audio tracks.

6.7.2 The selection rules

The first kind of rules affects the selection of the audio tracks and groups at rendering time. Four kinds of selection rules are introduced: the min/max rule, the exclusion rule, the not mute rule and the implication rule.

6.7.2.1 The min/max rule

The min/max rule is an unary rule applied to a group element. It allows specifying both minimum and maximum number of elements of the group that might be in active state. Two parameters are used:

- the minimum number of elements – called min – of the group that might be in active state at the same time;
- the maximum number of elements – called max – of the group that might be in active state at the same time.

The default values of min and max are: min=0 and max=n where n is the number of elements contained in the group (i.e. number of direct children) where rule is applied.

The min/max rule focuses on the states of the elements of the first hierarchical level (i.e. the direct children of the group).

This rule can result by the respect of the following inequality at any time: $\max(G_i) \geq \sum$ direct children of the group G_i with active state $\geq \min(G_i)$.

6.7.2.2 The exclusion rule

The exclusion rule is a binary rule applied on elements that compose an IM AF file. This rule allows specifying that several elements of a song will never be in the active state at the same time. This rule can be defined by a NAND logic operator between two elements. Two parameters are used. Each parameter defines an element of an IM AF file.

Thus, A Θ B (where Θ stands for exclusion and A and B are elements) means that the element A in active state implies the exclusion of the element B (i.e. the element B will be set in inactive state). Reciprocally, when the element B is in active state, then the element A will be set in inactive state.

At rendering time, this rule means that several elements of a same file will never be played at the same time.

The exclusion can be applied on audio tracks and groups. It can concern elements of a same group and elements of different groups. Concerning groups, exclusion means that elements of a same group will never be played at the same time that the elements of another group. Concerning audio tracks, exclusion means that two audio tracks will never be played at the same time.

6.7.2.3 The not mute rule

The not mute rule can be applied onto elements which compose the IM AF. This rule is an unary rule that defines an element always in the active state.

At rendering time, this rule means that an element will always be played.

The not mute rule can be applied on audio tracks and groups. Concerning the groups, this rule means that at least one element contained in the group (i.e. at least one element of the direct children) will always be played according to other selection rules.

6.7.2.4 The implication rule

The implication rule is a binary rule applied on elements of an IM AF file. This rule allows specifying that the activation of an element implies the activation of another element. Two parameters are used:

- the first one defines the element that imply the activation ;
- and the second one defines the implied element.

At rendering time, this rule means that several elements will be played simultaneously.

The implication can be applied on audio tracks and groups. It can concern audio tracks of the same group or audio tracks of different groups. Concerning groups, implication means that elements of a same group will be activated at the same time or implies the activation of elements of another group. Concerning audio tracks, implication means that the activation of one audio track implies the activation of another audio track.

The reverse implication is not implicit. If reverse is needed, it can be defined by the opposite imply rule, i.e. A ➔ B and B ➔ A (where ➔ stands for implication and A and B are elements).

A ➔ B means that if the element A is in the active state then the element B will be in the same state. If the element B is in the inactive state then the element A will be in this state. Finally, if the element A is in the inactive state then the element B might be in the active or inactive state.

6.7.3 The mixing rules

The second rules category is related to the audio mixing. These rules are also defined by the creator of the IM AF file, and will allow driving the way the listener will interact with groups and audio tracks volumes at rendering time. Two kinds of volumes are associated to each element: the relative one and the absolute one.

The Relative Volume is the current real value Vr (0.0 to N) of each element of the structure that can be modified by the listener of the IM AF file at any time during rendering time. In User-mix mode, the default relative volume of each element is equal to the reference volume defined in the appropriate boxes (see `group_reference_volume` and `volume` in group and audio track header boxes). In Preset-mix mode, the default relative volume of each element is equal to the volumes defined in the preset box (see `preset_global_volume` and `preset_volume_element`).

The Absolute Volume is a real value coefficient Va (0.0 to N) applied on the sound signal of each element of a song. This coefficient is computed by the mean of the relative volume given by the listener. The relation between these two volumes is given by the following equation: Vaei=Vrei . Va(father(ei)) where ei is the current element, Va the absolute volume and Vr the relative one. If the current element do not have any father then the relation is reduced to the following one: Vaei=Vrei.

During IM AF rendering time, the absolute volumes of each element are computed by the mean of each relative volume. When the listener changes the relative volume of an element, the absolute volume of this element is computed and updated by the system. If this element has children in the structure, all of their absolute volumes are computed and updated by the system. The final mixing takes only care of track audio signals combined to their absolute volumes.

Four kinds of mixing rules are introduced: the limits rule, the equivalence rule, the upper rule and the lower rule applied on volumes elements. Mixing rules shall be considered only for the elements which are in active state at rendering time.

6.7.3.1 The limits rule

The limits rule can be applied onto relative volumes of each element of a song. This rule allows the producer to fix the minimum and maximum limits of the relative volume of each element.

This rule forbids the listener to move the volumes out of these limits. And the relative volume default limit values are min: 0.00 and max: 4.00.

6.7.3.2 The equivalence rule

The equivalence rule is a binary rule that can be applied between relative volumes of two elements that compose a song. Volumes which are under an equivalence rule shall respect a strict equivalence relationship defined as follows: the equivalence between elements A and B is represented by the relationship VrA/VrB=VdA/VdB where VdA and VdB are the reference volumes of the elements A and B. Note that if one of these elements is linked to other elements by other mixing rules, all linked volumes elements shall be updated according to the related rules.

The equivalence rule can be applied onto volumes of elements belonging to the same group or onto volumes of elements belonging to different groups.

6.7.3.3 The upper/lower rule

The upper/lower rules are binary rules applied between relative volumes of two elements which compose a song. Volumes which are subject to upper/lower rules shall respect a non strict superiority/inferiority relationship defined as follows: the upper/lower of an element A regarding an element B is represented by the relationship VrA/VrB ≥(≤) VdA/VdB where VdA and VdB are the reference volumes of the elements A and B. Note that if one of these elements is linked to other elements by other mixing rules, all linked volumes elements shall be updated according to the related rules.

The upper/lower rules can be applied onto volumes of elements belonging to the same group or onto volumes of elements belonging to different groups.

6.7.4 Rule Container Box

The following is the definition of 'ruco' box, which contains general information on the rules applied between the elements of an IM AF file.

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

Syntax

```
aligned(8) class RuleContainerBox extends Box('ruco'){
    unsigned int(16)    num_selection_rules;
    unsigned int(16)    num_mixing_rules;
}
```

Semantics

num_selection_rules – is an integer that specifies the number of selection rules contained in the rule container box.
num_mixing_rules – is an integer that specifies the number of mixing rules contained in the rule container box.

6.7.5 Selection Rule Box

The following is the definition of 'rusc' box, which contains specific information on a particular selection rule applied onto elements that compose an IM AF file.

Box Type: 'rusc'
Container: Rule Container Box ('ruco')
Mandatory: No
Quantity: Zero or more

Syntax

```
aligned(8) class SelectionRuleBox extends FullBox('rusc', version = 0){
    unsigned int(16) selection_rule_ID;
    unsigned int(8) selection_rule_type;
    unsigned int(32) element_ID;
    if(selection_rule_type==0){ // only MINMAX
        unsigned int(16) min_num_elements;
        unsigned int(16) max_num_elements;
    }
    else if(selection_rule_type==1 || selection_rule_type==3){
    // EXCLU | IMPLY
        unsigned int(32) key_element_ID;
    }
    string selection_rule_description;
}
```

Semantics

version – is an integer that specifies the version of this box.
selection_rule_ID – is an integer that uniquely identifies the selection rule.
selection_rule_type – is an integer that identifies the type of the selection rule.

selection_rule_type	Mnemonic
0	Min/Max rule (MINMAX)
1	Exclusion rule (EXCLU)
2	Not mute rule (NOTMUTE)
3	Implication rule (IMPLY)

`element_ID` – is an integer which represents the ID of the element involved in this selection rule.

`min_num_elements` – is an integer used uniquely by the min/max rule (i.e. when `selection_rule_type=0`). It specifies the minimum number of elements that shall be played simultaneously. If this value is not defined by the producer, then the default value is equal to 0.

`max_num_elements` – is an integer used uniquely by the min/max rule (i.e. when `selection_rule_type=0`). It specifies the maximum number of elements that shall be played simultaneously. If this value is not defined by the producer, then the default value is equal to the number of elements contained in the group involved (see `num_elements` in the group box).

`key_element_ID` – is an integer which represents the ID of the element on which the rule is applied. It is defined only for both exclusion and implication rules (i.e. when `selection_rule_type=1 and selection_rule_type=3`).

`selection_rule_description` – is a null-terminated string in UTF-8 characters which gives a human-readable description of the selection rule.

Rule compatibility checking

When an IM AF file is read on an IM AF player, in the User-mix mode only, the compatibility between the selection rules shall be assumed. Hence, the selection rules compatibility checking method used is described as a pseudo-code algorithm.

In order to consider the elements status, two methods are defined as follows:

- `isActive(element_ID)`: returns true if the element referenced by `element_ID` is in active state, otherwise returns false;
- `getActiveDirectChildrenNumber(group_ID)`: returns the number of direct children of the group referenced by `group_ID` which are in active state.

The compatibility is checked as follows:

```
switch(selection_rule_type){

    case 0: // MINMAX
        compatibility=
            (min_num_elements>=getActiveDirectChildrenNumber(element_ID))&&
        (max_num_elements<=getActiveDirectChildrenNumber(element_ID));

    case 1: // EXCLU
        compatibility=!(isActive(key_element_ID) && isActive(element_ID));

    case 2: // NOTMUTE
        compatibility=isActive(element_ID);

    case 3: // IMPLY
        compatibility=
            (isActive(key_element_ID)&&isActive(element_ID))||
        (!isActive(key_element_ID));
}
```

6.7.6 Mixing Rule Box

The following is the definition of '`rumx`' box which contains specific information on a particular mixing rule applied onto elements that compose an IM AF file.

Box Type: 'rumx'
Container: Rule Container Box ('ruco')
Mandatory: No
Quantity: Zero or more

Syntax

```
aligned(8) class MixingRuleBox extends FullBox('rumx', version = 0){
    unsigned int(16) mixing_rule_ID;
    unsigned int(8) mixing_rule_type;
    unsigned int(32) element_ID;
    if(mixing_rule_type==3){ // only LIMITS
        int(16) min_volume;
        int(16) max_volume;
        }
    else{
        // EQUIV or UPPER or LOWER
        unsigned int(32) key_element_ID;
    }
    string mixing_rule_description;
}
```

Semantics

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

`mixing_rule_ID` – is an integer that uniquely identifies the mixing rule.

`mixing_rule_type` – is an integer that identifies the type of the mixing rule.

mixing_rule_Type	Mnemonic
0	Equivalence rule (EQUIV)
1	Upper rule (UPPER)
2	Lower rule (LOWER)
3	Limit rule (LIMITS)

`element_ID` – is an integer that represents the element ID involved in the mixing rule.

`min_volume` – is a fixed 8.8 value specifying the minimum audio volume of the element (element_ID). It is used uniquely by the limit rule (i.e. when mixing_rule_type=3). A minimum volume will be included in the interval 0.00 to 4.00.

`max_volume` – is a fixed 8.8 value specifying the maximum audio volume of the element (element_ID). It is used uniquely by the limit rule (i.e. when mixing_rule_type=3). Note that min_volume ≤ max_volume. A maximum volume will be included in the interval 0.00 to 4.00.

`key_element_ID` – is an integer that describes an element ID which represents the element on which the rule is applied. It is defined for equivalence, upper and lower rules.

`mixing_rule_description` – is a null-terminated string in UTF-8 characters which gives a human-readable description of the mixing rule.

Rules compatibility checking

When an IM AF file is read on an IM AF player, in the User-mix mode only, the compatibility between the mixing rules shall be assumed. Hence, the mixing rules compatibility checking method used is described as a pseudo-code algorithm.

In order to consider the relative volumes and the reference volumes of the elements, two methods are defined as follows:

- `getRelativeVolume(element_ID)`: returns the relative audio volume of the element identified by `element_ID`;

- `getReferenceVolume(element_ID)`: returns the reference audio volume of the element identified by `element_ID`. This value corresponds to the audio volumes that are attributes of both audio tracks and groups (see `volume` and `group_reference_volume` in the appropriate boxes).

The compatibility is checked as follows:

```
switch(mixing_rule_type){

    case 0: // EQUIV
        compatibility=(
                abs(getRelativeVolume(key_element_ID) –
            (getRelativeVolume(element_ID) * getReferenceVolume(key_element_ID)/
            getReferenceVolume(element_ID))) <= 0.01);
    case 1: // UPPER
        compatibility=(
            getRelativeVolume(key_element_ID)/getRelativeVolume(element_ID)>=
        getReferenceVolume(key_element_ID)/ getReferenceVolume(element_ID));
    case 2: // LOWER
        compatibility=(
            getRelativeVolume(key_element_ID)/getRelativeVolume(element_ID)<=
        getReferenceVolume(key_element_ID)/ getReferenceVolume(element_ID));
    case 3: // LIMITS
        compatibility=(
            min_volume >= getRelativeVolume(element_ID))&&(max_volume <=
        getRelativeVolume(element_ID));
}
```

6.8 Timed text data

6.8.1 Introduction

For using the timed text with music playback such as lyrics for karaoke application, IM AF adopts 3GPP Timed Text format (3GPP TS 26.245) so that the requirements of the 3GPP TT (3GPP TS 26.245) apply for timed text in IM AF.

3GPP Time Text data consists of text samples and sample descriptions.

6.8.2 Sample format

Each text sample is composed of a string of text and optionally followed by sample modifier boxes. Text string consists of a list of characters of the text. Text modifiers describe how the text string should be rendered such as highlighted text, karaoke and hypertext link.

The text samples are stored in the `mdat` box as a text track. By using the boxes in the sample table box, such as `stts`, `stsc`, and `stco`, the text samples can be synchronized with other timed media.

6.8.3 Sample description

The sample descriptions for the text track are stored in the `stsd` box. TextSampleEntry extends SampleEntry with format name of ‘tx3g’. The sample description specifies the manner of text rendering, the horizontal and vertical justification of the sample, background color of the sample in RGB color space with the alpha (transparency) value. It also specifies the location and style of the text which is the inset of the text within the region of the text, such as the font color, font style, and font size.

6.9 Metadata

For Interactive Music AF, metadata provides simple background information as following:

Table 3 — Album/song/track level metadata of IM AF

Description	Level		
	Album	Song	Track
Title	o	o	o
Singer	o	o	-
Composer	-	o	-
Lyricist	-	o	-
performing musician	-	-	o
Genre	o	o	-
file date	o	o	o
CD track number of the song	-	o	-
Production	o	o	-
Publisher	o	o	-
copyright information	o	o	-
ISRC (International Standard Recording Code)	-	o	-
Image	o	o	-
URL site address related to the music and the artist (e.g. album homepage, fan café, music video)	o	o	-

The two types of metadata (textual XML) included in the IM AF file format are:

- metadata for the song and track *(item);*
- metadata for the album *(collection):*
  - album level metadata allow grouping the songs and storing metadata relating to those group, of the ordering in the album.

IM AF allows the metadata for each level to be stored inside the Meta Box in Table 4.

Table 4 — Metabox location according to each level metadata of IM AF

Metadata	Location
track level	trak/meta box
song level	moov/meta box
album level	meta box of file

Figure 2 and Figure 3 show the location of the metadata for a single type file structure and a multiple type file structure of IM AF respectively.

The metadata handler type is ‘mp7t’. For track and song level metadata, MPEG-7 CreationInformation, MediaInformation and Semantics DS are used. The album level metadata have the structure information because songs are structurally arranged in an album. Hence for album level metadata, MPEG-7 ContentCollection DS is used as well as MPEG-7 CreationInformation DS. In order to combine the metadata, the CreationInformation DS is contained under the Content element in the ContentCollection DS for the album level metadata. Note that the requirements of the MDS (ISO/IEC 15938-5) apply for metadata in IM AF.

Table 5, Table 6 and Table 7 summarize the semantics of available tools: track, song, and album level metadata, respectively.

Table 5 — Semantic for the track level metadata

Tag Name	Semantics
CreationInformation/Creation/Creator[@type="Instrument"]	The title of the track
- CreationInformation/Creation/Creator[Role/@href="urn:mpeg:mpeg7:RoleCS:2001:PERFORMER"]/Agent[@xsi:type="PersonType"]/Name/{FamilyName, GivenName} (Artist Name) - CreationInformation/Creation/Creator[Role/@href="urn:mpeg:mpeg7:RoleCS:2001:PERFORMER"]/Agent[@xsi:type="PersonGroupType"]/Name (Group Name)	The name of a musician who is performing instruments, such as vocal, guitar, keyboard and so on.
CreationInformation/CreationCoordinates/Date/TimePoint	Time point of the recording

Table 6 — Semantic for the song level metadata

Tag Name	Semantics
CreationInformation/Creation/Title[@type="songTitle"]	The title for the song
- CreationInformation/Creation/Creator[Role/@href="urn:mpeg:mpeg7:RoleCS:2001:PERFORMER"]/Agent[@xsi:type="PersonType"]/Name/{FamilyName, GivenName} (Artist Name) - CreationInformation/Creation/Creator[Role/@href="urn:mpeg:mpeg7:RoleCS:2001:PERFORMER"]/Agent[@xsi:type="PersonGroupType"]/Name (Group Name)	The name of a musician such as singer, composer and lyricist
CreationInformation/Classification/Genre[@href="urn:id3:v1:genre ID"]	Genre
CreationInformation/CreationCoordinates/Date/TimePoint	Time point when the song is released
Semantics/SemanticBase[@xsi:type="SemanticStateType"] /AttributeValuePair	CD track number of the song
CreationInformation/Creation/Abstract/FreeTextAnnotation	Information on production, publisher and site address related to the music and the artist (e.g. album homepage, fan café and music video)
CreationInformation/Creation/CopyrightString	Textual label indicating information that may be displayed or otherwise made known to the end user
MediaInformation/MediaIdentification/EntityIdentifier	ISRC
CreationInformation/Creation/TitleMedia[@type="TitleImage"]	The title of the multimedia content in image form

Table 7 — Semantic for the album level metadata

Tag Name	Semantics
ContentCollection/CreationInformation/Creation/Title[@type="albumTitle"]	The title of the album
- ContentCollection/CreationInformation/Creation/Creator[Role/@href="urn:mpeg:mpeg7:RoleCS:2001:PERFORMER"]/Agent[@xsi:type="PersonType"]/Name/{FamilyName, GivenName} (Artist Name) - CreationInformation/Creation/Creator[Role/@href="urn:mpeg:mpeg7:RoleCS:2001:PERFORMER"]/Agent[@xsi:type="PersonGroupType"]/Name (Group Name)	The name of representative musician of the album
ContentCollection/CreationInformation/Classification/Genre[@href="urn:id3:v1: genre ID"]	Genre
ContentCollection/CreationInformation/CreationCoordinates/Date/TimePoint	Time point when the album is released
ContentCollection/CreationInformation/Creation/Abstract/FreeTextAnnotation	Information on production, publisher and site address related to the music and the artist (e.g. album homepage, fan café and music video)
ContentCollection/CreationInformation/Creation/CopyrightString	Textual label indicating information that may be displayed or otherwise made known to the end user
ContentCollection/CreationInformation/Creation/TitleMedia[@type="TitleImage"]	The title of the multimedia content in image form