

Development of an IM AF encoder

Eugenio Onate, Panos Kudumakis
School of Electrical Engineering and Computer Science
Queen Mary University of London
Email: eo301@eeecs.qmul.ac.uk

Abstract—In this work an encoder able to create a new interactive file following the so-called Interactive Music Application Format (IM AF), has been developed. A file created with the IM AF allows users more control over the song. More specifically, it permits to vary the volume of each instrument or choose a predefined preset. The encoder has been programmed in C following the standard defined by MPEG. The paper describes first the previous file formats which have been the basis for the development of the IM AF file. Then the main features and the structure of the new encoder are detailed. The performance of the encoder has been validated by creating three IM AF files with special music features. These conformance files have been successfully tested in the IM AF player provided by the MPEG group.

I. INTRODUCTION

The music market is strangled with decreasing sales. Customers need new products that attract their attention. The time has come to innovate and create a new file format that will change the way people listen the music. Formats like mp3 were a good revulsive for industry, but have been here since the 90s; today it is all about interactivity between user and technology. Nowadays technologies have led people to live in a society where everyone is connected among themselves and sharing all kind of information.

The music industry has been reluctant to let people interact with the songs, basically because the technology was not ready. Now it is the perfect time to change the concept of the listener, making him/her participate of the musical experience. The Moving Picture Experts Group (MPEG) defined a new file format called Interactive Music Application Format (IM AF) [1] as part of the Multimedia application format (MPEG-A) [2]. IM AF is a versatile file format standard for mixing different types of multimedia data (music, images and text). It allows users to modify the volume of each instrument separately or change the mixing style according to some presets predefined by the producer.

The process for creating standardized files like MPEG-1 Layer III (.mp3) or IM AF (.ima) is split in two sections: encoder and decoder. In the encoder section, the file is created following the appropriate standard; parts like headers, track information and samples data are put together inside a binary file. The decoder is responsible to read and understand the file so that it will be able to reproduce it.

When a new file is defined, the file decoder is made publicly available through ISO/IEC MPEG to let companies design their own encoder. In this manner, they ensure compatibility of the file no matter how it was created. Since MPEG defined

the standard for IM AF in 2010 no one has publicly released an implementation of the encoder yet, although commercial services exist.

In this work an encoder for IM AF files has been designed, implemented and tested following the ISO 14496 Part 12: ISO base media file format and ISO 23000 Part 12: Interactive music application format. The new IM AF file encoder can support a maximum of 16 simultaneously audio tracks with a sampling frequency of 44.1kHz at 16 bits per sample. In this version, individual music-tracks must be encoded in MP3. Also, it is able to add different mixing presets and rules. Presets are predefined values of the volume of each instrument that allow the producer to create different versions of the track. Also users can exchange and share their own mixtures. Rules are limitations imposed by the creator of the song, usually producers, to avoid users destroy the essence of the song. The encoder has been programmed in C and has a simple command line user interface for introducing the information required by the program.

II. BACKGROUND RESEARCH

In this section we analyze some previous standards that have contributed to create the IM AF file format. We also present similar applications based on interactivity files.

A. MPEG-1 Layer III

MPEG-1 Layer III, most commonly known as MP3 [3], was released in 1991 and soon become the most used tool for Internet and audio delivery. It became very popular due to its big impact on the music industry, offering good sound quality with low bit-rate. MPEG-1 works on different sampling frequencies and supports variable compression ratio. For Layer-III the standard determines a variety of bit-rates from 8 Kbit/s to 320 Kbit/s. The most common ones are 192 Kbit/s and 320 Kbit/s as they provide transparent quality. Bit-rate can change from frame to frame allowing higher bit-rates for more complex parts of the song, while less space is needed for less complex ones.

Audio encoders may have different architectures depending on the model. MPEG standard encoders have no restrictions for their design. However, all files produced have to be compatible with the corresponding decoder. A MP3 file is split into small blocks or frames where data is stored. Each frame has 1152 samples per frame [4]. The standard sampling frequency is 44.1kHz, which means that the duration of one frame is $1152/44100 = 0.026$ sec. Knowing the bit-rate it is possible

to calculate the size in Bytes of one frame [4]; if the bit-rate is 128kbps the size is 417 Bytes, while if it is 192kbps the size is 626 Bytes. At beginning and end, MP3 files may have ID3 TAGs. Those are metadata containers with information related to the song.

[ID3 TAG] Frame1 Frame2 Frame3 ... FrameN [ID3 TAG]

Each frame is structured in the same way. It has a 32-bit header followed by the samples. The header contains information associated to technical specifications such as MPEG version, Layer, Bit-rate and sampling frequency. The decoder uses this information to decode the file properly.

B. MPEG-4

MPEG-4 was an evolution of MPEG-2. Unlike its predecessor, improving the compression method was not the main task. The main difference with previous audiovisual coding standards is the object-based representation model that helped consolidate MPEG-4. Objects can be audio objects like multi-channel audio content or speech, or video and movies. Transmitting several audio objects with multiple tools creates an audio composition system or soundtrack. This ability furnishes MPEG-4 with important superiority in quality and flexibility versus its predecessors.

The file format defined by MPEG-4 is called MP4; it is described in ISO 14496 Part 12 Base Media File Format. It is an important part of the IM AF file as explained in a next section. It is intended to accommodate multi-media information in a versatile arrangement that allows ease manipulation, correction and display of the media.

C. Previous interactive music systems

IM AF was not the first file format to include interactive control over the media. In this section we briefly describe three examples of earlier formats: IEEE 1599 [5], iXMF [6], and iKlax [7].

IEEE 1599 combines in a single file music and XML symbols, as graphical representation of the music or performance indications. This information is integrated and synchronized within the same framework and can be accessible individually or as a whole. An exhaustive description of music must integrate different types of information. IEEE 1599 has developed a new XML encoding system to allocate this heterogeneous information in a single file. XML organizes the information in six different layers [5]

- General - It stores information about the piece.
- Logic - Coherent representation of score symbols.
- Structural - Classifying musical objects and the connection.
- Notational - Visual representation of the score.
- Performance - Computer-based description of musical representation.
- Audio - Digital audio recording.

iXMF or interactive eXtensible Music Format is a file format created by the video games industry to provide a standard

structured audio file format that supports cross-platform interchange of advanced interactive audio tracks. An iXMF file includes in a single file all the information needed to perform the audio track as the artist intended. The same information defines the structure of the file. iXMF uses a structure such that an event can be triggered at a particular instant. The selected event can activate a wide range of activities, such as the playing of an audio file or the execution of a specific code.

iKlax technology was developed by iKlax Media company and LaBRI in France. iKlax proposes a file format with separated tracks and interactivity to manipulate the music piece. The project includes a music player and a music editor. iKlax format groups all the tracks of a song and related metadata in a single file. It has two levels of interactivity; the first level allows the selection of the track, where listeners can choose the track that they want to listen. The second level is the mixing; here listeners can modify the level of each track.



Fig 1. iKlax Player

III. DEVELOPMENT OF THE IM AF ENCODER

In this section we present the IM AF encoder developed in this work. Also, we will analyze the different sections of the file that are used in the new encoder. IM AF uses parts of the MPEG-4 standard as well as new parts designed specifically for this new interactive file.

A. Interactive music service

To ensure interoperability between different multimedia content, the International Standard Organization (ISO) presented in 2010 IM AF. IM AF specifies how to connect multiple tracks with related information under a well-defined structure that facilitates the manipulation of interactive music content. This content includes audio tracks, preset data and rules. Audio tracks represent the instruments of the song, they can be a single instrument or a group. Preset data is a predefined information related to the volume of each track and allows users to create different versions of the song. Those values cannot be changed after the file is created. Hence, it can be useful for producers to present the same track from different points of view. Finally, IM AF files introduce rules to avoid users destroy the initial intention of the author. The created files are reproduced by an interactive music player

as shown in Figures 2 and 3. Users have two different options to listen the song: preset-mix mode and user-mix mode. In the preset mode the user chooses one of the pre-defined presets in the IM AF file, and then the tracks change the volume according to the preset values. In the user mode, the user selects/de-selects the tracks and controls their volume. All the actions performed by the user need to be compatible with the rules; otherwise the actions will not be carried out.



Fig 2. IM AF player. User-mix mode (left) and preset mode (right).

B. Structure of the IM AF file

The encoder is responsible for creating the IM AF file. For doing that it follows the standard defined in ISO 23000-12. The framework of this file is based on the MPEG-4 ISO based Media File Format standard; IM AF has introduced some improvements to enable interactive control. IM AF files consist of a series of boxes that include all data. There are two different types of boxes; those that may contain other boxes inside them and others that just contain data (called *FullBoxes*). All boxes start with the header which defines the *size* and *type*. *FullBoxes* incorporates in the header the *version* and *flag* information. The *size* defines the total size of the box, including data and, if necessary, other boxes. It has a size of 32 bits (unsigned integer), but if the data stored in the box is bigger than that it can use 64 bits. *Type* is the identification of the box, and each box has its own type, i.e. *ftyp*, *moov*, *mdia*. The *version* is an integer (8 bits) and specifies the version of the box, and the *flag* is a 24-bit integer and its use depends on each box. A complete list of the boxes that form the structure is presented below:

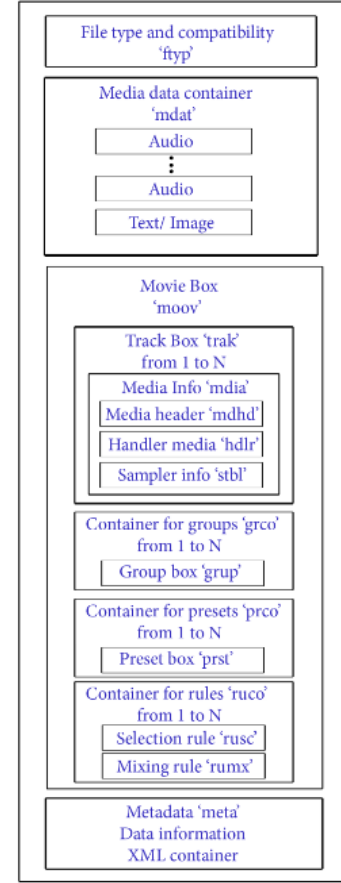


Fig 3. Boxes inside the IM AF file

We will not explain each box individually, further information and description of each box can be found in [1], [8], and [9].

IV. IMPLEMENTATION

The encoder that creates the IM AF file has been implemented in C. It does not use any external library, ensuring that it will work easily in any computer. The program has two parts: the main program and the IM AF header. The main program includes the functions to create the file and the header defines the structure of the boxes. There is also a basic command line tool to let users introduce values such as the number of tracks and their names. The program has a static structure; meaning that the maximum number of tracks, presets and rules are defined by a global variable.

Right at the beginning, the encoder asks the user to introduce the number of tracks that the IM AF file will incorporate. Then, the user writes the names of all the tracks to be stored in the file. The encoder detects if the track exists or not. If it does not, the program will ask again for the name. When the user has finished to introduce all the tracks, the encoder creates the binary file that will store all the information. The first function called in by the program is the *filetypebox*. It specifies the different types supported by the encoder. Once the type is clear, the encoder converts the audio files (MP3) into MP4 files. The first step is to extract the samples

of the audio file and store them into the *media data container*. The idea is to find the beginning of the first frame and read the data until it reaches the end of the file. The intention is to avoid the ID3 header, as it does not contain useful information. The second step is to extract the sample information from the frames of the MP3 and write it in the corresponding sample box inside the *Track Box*. In the *sample size box* stores the duration and size of each frame, as well as the total number of frames (Figure 4). In this case the search for information is more precise than in the *media data container*. Rather than reading the whole file, it localizes the beginning and end of each frame, extracting the information of one frame at a time. This information is reused by the *time to sample*, *sample to chunk* and *chunk offset boxes*.

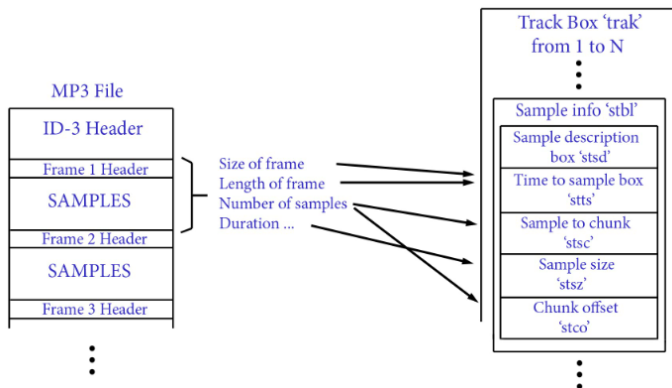


Fig 4. Sample data information

The *preset container* is next implemented after the track box. It creates a static preset with a fixed value. First, it defines the number of presets that there will be in the file. Then, it assigns the volume of each track; initially takes the value of 100. Changing this value will modify the volume of each track in the specific preset. The next step is to create the rules. The rule container creates two rules: one selection rule and one mixing rule. Finally, the movie box is created together with the movie header box. Here the encoder puts together the sizes of all boxes within the movie box. Doing that at the end of the code ensures that the total size of the movie box will match the sum of each one.

A. Programs that analyze the IM AF file

In the process of creating the IM AF file we have used various programs that analyze the structure of the file. Specifically programs that show the boxes of the MPEG-4 file and programs that work for IM AF too. The creation of a file like the one created here is full of little parameters that can lead to ill-function of the file. The order of the boxes is very important, so it is the size of each box. These programs have been an important part of the success of the encoder. The *MP4 Browser* [10] is a free software for MS Windows. It gives a very clear idea of how the IM AF file is structured. It details the content, *type* and *size* of all *boxes*, except the ones used only in IM AF, like the *preset* and *rules boxes*. This software

was created to work with ISO 14496 files (MPEG-4), so it does not support the new features of IM AF. Without this type of programs the task of building an IM AF encoder would be more difficult.

V. RESULTS

In this section we analyze the characteristics of the files created by the encoder. When the user wants to create a new file, he/she introduces the information through a command line interface. Before presenting the conformance files, we will explain the steps for doing that task. First, the encoder welcomes the user and asks for the number of tracks to be included in the IM AF file.

```
Welcome to the IM_AF encoder
This program will allow you to create an IM_AF file.
How many tracks there will be in your IMAF file?
3
```

Fig 5. Creation of file - Step 1

In this case the user has selected 3 tracks. The maximum number of tracks that the IM AF file supports is limited by the larger brand in the *filetypebox*. After specifying the number of tracks, the next step for the user is to enter the name of each track. The audio files must be in the same folder as is declared in the code and the names have to be written together with their extension, as showed below.

```
How many tracks there will be in your IMAF file?
3
Name of the track number: 1
bass.mp3
Name of the track number: 2
drums.mp3
Name of the track number: 3
synth.mp3
```

Fig 6. Creation of file - Step 2

When the user has finished entering all the names, the encoder creates the preset defined previously in the code. To provide clarity on what is being written in the file, the encoder prints on the screen the characteristics of the preset.

```
Presets:
Static track volume preset: invariant volume related to each track
-----
Preset number 1: static_track
Enter volume for bass.mp3 = 120
Enter volume for drums.mp3 = 100
Enter volume for synth.mp3 = 80
```

Fig 7. Creation of file - Step 3

The last step is to write the rules into the file. They are also written directly in the code rather than using the command line.

Rules:**Rule 1: Not mute for channel 3****Rule 2: Upper rule between channel 1 and 2****File is created successfully, and ready to use!***Fig 8. Creation of file - Step 4*

After that, the encoder writes all the information into the file. Hence, the only thing that remains to be done is to inform to the user that the file is created. A file with extension IM AF ready to play in the IM AF player has been created.

A. Conformance files

Three different files have been implemented in order to validate the performance of the new encoder. Each time a new IM AF file is presented we supply various examples to prove the efficiency of the encoder. For that purpose, we have selected three different multi-track songs from [11] with different styles. Each song has different configurations of the encoder parameters. Below it presents one conformance file, the rest of the file can be found in [8].

Example 1	
Name:	Example_Acoustic.ima
Format:	MP3
#Instruments	6
Presets	Yes
Rules	Yes (2)

Type/N° of instrument	Value of Static Preset	Selection Rule: Not mute	Mixing Rule: Equivalence
ID 1 - Drums	40	-	Element involved
ID 2 - Bass	80	-	Key element
ID 3 - Voice	120	Element involved	-
ID 4 - Guitar	180	-	-
ID 5 - Clarinet	200	-	-
ID 6 - Accordion	240	-	-

Fig 9. Conformance file 1

This example has six instruments, two rules and one static preset. The selection rule is Not mute and the element that will be always in active state is number 3: the voice. The rest of the instruments can be mute. The mixing rule is Equivalence, meaning that two elements will have the same volume during the whole piece of music. In this case the elements involved are the drums and the bass. At no time one element will sound louder than the other. The preset defines a static volume for the instruments; the secondary elements like the clarinet or accordion get more presence and the important ones like the drums or voice reduce the level.

VI. CONCLUDING REMARKS

In this work an IM AF encoder has been implemented that creates a new interactive file. It is intended for all users, from a beginner that does not know much about programming to a professional engineer who wants to personalize the encoder to suit his/her needs. The beginner will find the encoder useful and intuitive as he/she does not need to modify the code for generating an IM AF file. There is a command line interface that guides newcomers through the creation of the file. This

interface allows users to enter the instruments in the file. The customization of the encoder by a user is also a relatively simple task.

On the other hand, an experimented user will find the encoder sufficiently robust tool for his/her professional work. Moreover, in case a user would want to introduce new improvements in the file, the program created will represent the framework of his/her work and will be flexible enough to support the changes.

This encoder has been created in a relatively short period of time. Thus, there are some aspects that could have been done differently. The command line interface is limited; the user can introduce information related to the number and name of the tracks. Also the presets and rules information must be written directly in the code, hence making it troublesome for someone who does not know C. Life would be more comfortable for a user if the interface was an application by itself, rather than being part of the C compiler. The encoder supports only MP3 files. Nowadays this is not a problem as there are lots of free software that converts audio files into MP3.

A weakness for the immediate application of the IM AF file format developed is the difficulty to find music in multi-track format. For the time being, this encoder is intended for musicians or professionals who have access to the recordings of the song. This standard file hopefully would soon be embraced by major music labels such as EMI or SONY via their huge sound library. In this manner, the new encoder will help in the popularization of the IM AF interactive file format to the general public. The intention is that this file format achieves enough popularity to be present in the recording studios as the standard format of the future.

Future extensions of the decoder will include the ability to store images, like the cover of the album of pictures of the artist, and the possibility to insert synchronized text for karaoke. Also, it would be interesting to store the tracks online. This will help to create a social network where users could share instruments without the need to have physically the file. We note that IM AF is a new standard that creates an interactive file that allows the manipulation of the track volume. This file can be improved by adding more tools such as an equalizer for each channel. In this manner, users would have the chance of manipulating the frequencies in addition to controlling the volume, thus converting the file into a more complete and professional tool. Another improvement would be to introduce a panorama for each channel, where users could move the instrument in the 3D space, placing each track in a different position. We finally point out that the programming language chosen did not allow to incorporate the encoder inside a mobile application. Thus, translating the code to C++ or Java will be useful for the future development and implementation of the encoder in Apps.

REFERENCES

- [1] ISO/IEC Std. 2010, *Information Technology Multimedia application format (MPEG-A) MPEG music player application format Part 12: Interactive music application format*, ISO/IEC FDIS 23000-12.

- [2] ISO/IEC Std. 2006, *Information Technology Multimedia application format (MPEG-A) Part 2: MPEG music application format*, ISO/IEC 23000-2.
- [3] ISO/IEC Std. 2006, *Information Technology Coding of moving pictures and associated audio for digital media at up to about 1,5 Mbit/s Part 3: Audio*, ISO/IEC 11172-3.
- [4] *MP3 File Structure* Available at:
<http://www.multiweb.cz/twoinches/mp3inside.htm>
- [5] L. A. Ludovico, *Key Concepts of the IEEE 1599 Standard*, Laboratorio di Informatica Musicale (LIM), Milano, 2008.
- [6] Draft 0.9.1a, *Interactive XMF: File Format Specification*. La Habra CA, February 18, 2008.
- [7] F. Gallot, O. Lagadec, M. Desainte-Catherine, S. Marchand, *iKlax: A New Musical Audio Format for Interactive Music*. Proc. ICMC (International Computer Music Conference) 2008, August 2008.
- [8] E. Onate, *Development an IM AF encoder*. MSc Digital Music Processing Final Report, Queen Mary, 2012.
- [9] ISO/IEC Std. 2002, *Information Technology Coding of Audio-Visual Objects Part 12: ISO base media file format*, ISO/IEC 14496-12.
- [10] *MP4 Browser* by MiraVid. Available at:
<http://download.cnet.com/MiraVid-MP4-Browser>
- [11] *The Mixing Secrets*, Free multitrack download library. Available at:
<http://www.cambridge-mt.com/ms-mtk.htm#Acoustic>