Cue Point Processing: An Introduction

Eoin Brazil

Interaction Design Centre
Department of Computer Science and Information Systems,
University of Limerick, Ireland
eoin.brazil@ul.ie

ABSTRACT

Modern digital sound formats such as *aiff*, *mpeg1/2/4/7*, *wav* and *ra* support the use of cue points. A cue point may also be referred to as a seek point or a key frame. These mechanisms store meta data about the sound files. In this paper, we identity and describe how these formats are encoded, and process meta data information with a focus on cue points.

Finally, we conclude with the direction of our future research for improving multimedia browsing mechanisms and additional applications by leveraging the use of cue points within those applications.

Keywords: sound file formats, cue points, sound file, audio files, seek point, key frame, audio indexing

1. INTRODUCTION

Since the beginning of sound file formats some decades ago, a plethora of sound formats have been defined. Cataloging the numerous formats would result in an encyclopedic text. Adding older and more archaic formats to this text can result in a foray into archaeological pursuits. We have narrowed our research into a few well-known, modern formats to prevent these unnecessary forays. As stated previously, these formats are *aiff, mpeg1/2/4/7, wav* and *ra*.

Narrowing our research to these particular formats, we will point out what Cue Points are and why they are necessary, then give an overview of the different file formats, how they each support Cue Points in specific, discuss an application we have developed to exploit this mechanism and finally outlining why Cue Points are a necessary extension to improving browsing and multimedia applications.

2. Cue Points

2.1. Definition

In this paper, we assume the term *file format* to refer to sound file formats and *Cue Point* as the general term for the marker points which can also be referred to as seek points, index points or key frames. Cue Points are the meta data in a file format which refer to points of interest within the sound. Cue Points are the particular meta data we are concerned with in this paper.

Meta data is critical information that allows further editing, indexing and browsing of sound files. Metadata has been defined by [1] as "any information required to make other data useful". He states "meta data provides an architecture or framework describing the user's data within a data environment". In our research, the corpus that we are dealing contains sound files and the related information or metadata on the files within the corpus. Prothman breaks metadata into five categories. Our research is concerned with one of these: access metadata, which is used to help browsing, retrieval and indexing of the data. This type of Meta information is normally stored in the file headers, which are standardised and are available and readily usable by the inclusion of extra code to many common applications.

2.2. Scope

Cue Points are akin to marker points for important events within a sound. These points can mark sections of a sound. Music files for example can have the chorus and/or an instrumental solo event marked as cue points. For recognition of sounds, while browsing, it is most often sufficient to hear only 500 ms to 2 seconds of the characteristic or significant part of a sound file [2]. Using Cue Points, we can play only these significant parts of sound files when the user is browsing. Exploiting this fact can lead to faster browsing of digital multimedia resources by only playing a sort relevant section of an audio file. Cue Points can then enable users to explore large numbers of digital audio resources quickly by concentrating on the significant portion of the audio, allowing for faster recognition of a sound.

3. Sound File Formats

The majority of conventional file formats store meta data about the sound varying from author, date recorded and sound quality to genre [3]. Cue Points are implemented in a variety of manners in the different file formats. The areas of applications for Cue Points are in digital audio browsing, searching, management and creation. Specialised sound applications such as sound tools (e.g. SoundForge, Cubase, ProTools) have historically focused on authoring audio resources, but to the knowledge of the authors no

applications exist that exploit the benefits of Cue Points for searching and browsing. Since most of these applications can read and process the file formats, extensions to the packages to support Cue Points should be easily handled. In the following paragraphs, we discuss the technical aspects of different formats with regard to how they implement Cue Points.

3.1. AIFF

Full Name: Audio Interchange File Format
Originator: Apple Computer, Inc.

AIFF is a file format for storing digital audio (waveform) data. It supports a variety of bit resolutions, sample rates, and channels of audio. This format is used in professional programs that process digital audio waveforms. An Audio IFF file is a collection of a number of different types of chunks [4]. There is a required Common Chunk, which contains important parameters describing the waveform, such as its length and sample rate. The Sound Data chunk, which contains the actual waveform data, is also required if the waveform data has a length greater than 0 (ie, there actually is waveform data). All other chunks are optional as shown in figure 1. Among the other optional chunks are ones, which define markers, list instrument parameters, store application-specific information, etc. Our research is interested in the optional chunk containing marker information.

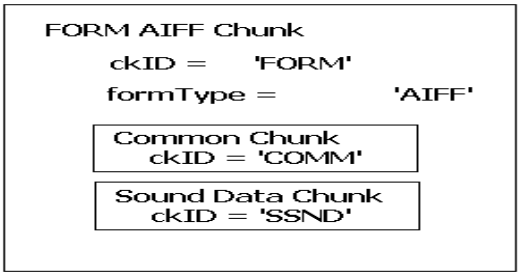

Figure 1. *Graphical overview of an example, minimal AIFF file.*

The marker chunk contains markers that point to positions in the waveform data. The instrument chunk uses the markers from this chunk to mark loop beginning and end points. A marker chunk is defined using a C-like language that is used to describe all the data structures and chunks in the file. The layout of a marker structure can be seen below in figure 2. The id is a number that uniquely identifies that marker within an AIFF. The id can be any positive non-zero integer, as long as no other marker within the same file has the same id. The marker's position in the WaveformData is determined by the position field. Markers conceptually fall between two sample frames. A marker that falls before the first sample frame in the waveform data is at position 0, while a marker that falls between the first and second sample frame in the waveform data is at position 1. Therefore, the units for position are sample frames, not bytes or sample points [4].

```
typedef short   MarkerId;

typedef struct {
  MarkerID        id;
  unsigned long   position;
  pstring         markerName;
} Marker;
```

Figure 2. *Layout of a Marker Structure*

An important point to note with this file format is that there may only be one marker chunk in the file. Marker structures are stored within the marker chunk. The format of a marker chunk can be seen in figure 3. Once the data field *numMarkers* is not zero, it is followed by that many marker structures, one after the other. The marker structures are packed together with no unused bytes between them. The marker structures need not be placed in any particular order within the marker chunk.

```
#define MarkerID 'MARK'  /* chunkID for Marker Chunk */

typedef  struct {
  ID              chunkID;
  long            chunkSize;

  unsigned short  numMarkers;
  Marker          Markers[];
} MarkerChunk;
```

Figure 3. *Layout of a Marker Chunk.*

Another point to remember is the precedence order of the chunks in the file format. The marker chunk comes third in order of preference is illustrated in figure 4. This is the highest precedence of the optional chunks as both the common and sound data chunks must always be included in the file. The application is responsible for managing and updating these chunks to prevent conflicts from arising [4].

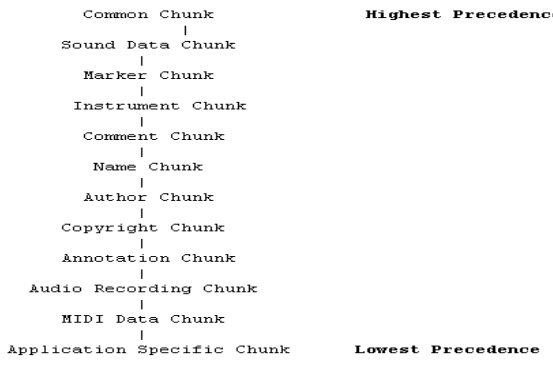

Figure 4. *Precedence of Chunks*

3.2. MPEG – 1 / 2 / 4 / 7

Full Name: Moving Picture Experts Group
Originator: ISO/IEC (International Organization for Standardization / International Electrotechnical Commission)

The MPEG series of formats are a set of international standards for compression, decompression, processing, and coded representation of moving pictures, audio, and their combination [5]. MPEG defines the syntax of low bitrate video and audio bitstreams of synthetic and natural sources, descriptions of their structure and content, and the operation of conformant decoders of these bitstreams. A point to note about the MPEG series is that they are designed as coexisting standards which each focus on a different area of multimedia. In essence, this amounts to the fact that the later formats of the series merely complements the earlier formats. Layers are another term used with these standards; these represent a group of coding algorithms.

The first in this sequence is the MPEG-1 standard. Originally the file extension *mp3* was created with the emergence of MPEG-1 Layer III encoder and decoder software for Windows, the structure of which may be seen in figure 5. After standardisation of MPEG-2, sound files encoded with the MPEG-2 lower sampling rate extension of Layer III are also called "MP3"-files. Sometimes *mp3* is incorrectly called MPEG-3[6, 7].

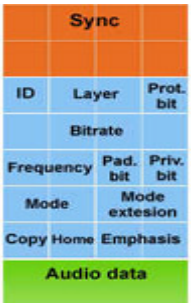

Figure 5. *MP3 structure*

The first two of the MPEG series were aimed at the technical processing of the digital sounds; they do not adequately deal with the presentation and control of the audio. MPEG-4 represents a scene as objects including the different audio and video frames. It also has sections dealing with composition and structured audio. The time stamping mechanism was also improved to cover fields within a scene [8-10].

MPEG recognized the need for an interface dealing with the swift identification and retrieval of digital multimedia content, this led to the MPEG-7 standard. It is used to specify a standard way of describing various types of audiovisual information [11]. The use of content descriptors ensures that regardless of media format or storage format a standard way exists. The MPEG-7 standard uses and extends the object model in use by MPEG-4. The extensions are in the areas of object based querying, identification, access, manipulation and granularity [12].

Several different mechanisms within the various MPEG standards accommodate Cue Points. MPEG-1 and MPEG-2 provide support for Cue Points by the use of ID3 tags with audio seek points [3, 13]. A point to note is that the audio format MPEG layer I, layer II and layer III (*mp3*) has no native way of saving information about the contents, except for some simple yes/no parameters like "private" and "copyrighted". By adding a small chunk of extra data in the end of the file, the *mp3* file carried information about the audio and not just the audio itself. The structure of an ID3 tag is shown in figure 6. Audio files with variable bit rates are intrinsically difficult to deal with in the case of seeking within the file. The ASPI frame provides a list of seek points within the audio file. The seek points are a fractional offset within the audio data, providing a starting point from which to find an appropriate point to start decoding.

Figure 6. *ID3 Tag Structure*

MPEG-4 uses fields within the scene description that carry a time value to enable Cue Points. MPEG-7 holds the necessary information in descriptors to facilitate Cue Point usage. The descriptors are designed for describing the following types of information: low-level audio-visual features such as color, texture, motion, audio energy, and so forth; high-level features of semantic objects, events and abstract concepts; content management processes; information about the storage media, and so forth. Descriptors (DSs) correspond to low-level features that are extracted automatically, whereas human intervention is required for producing the high-level Descriptors [14]. MPEG-7 Multimedia DSs are organized into the following areas: Basic Elements, Schema Tools, Content Description, Content Management, Content Organization, Navigation and Access, and User Interaction. Our research is focused on the content description. The core element of the content description is the *Segment* DS. It addresses the description of the physical and logical aspects of audio-visual content. A segment may be described by creation information, usage information, media information and textual annotation. However, specific features depending on the segment type are also allowed. These specific features are reported in Table 1. MPEG-7 features then allows audio segments to be marked by time and by region. As currently there are few actual implementations of the standard in use, this paper could only focus on the theoretical details as presented [14].

Feature	*Audio Segment*	*Video Segment*	*Still Region*	*Moving Region*
Time	✓	✓	✘	✓
Shape	✘	✘	✓	✓
Color	✘	✓	✓	✓
Texture	✘	✘	✓	✘
Motion	✘	✓	✘	✓
Camera motion	✘	✓	✘	✘
Mosaic	✘	✓	✘	✘
Audio features	✓	✘	✘	✓

Table 1: *Specific features for segment description*

3.3. WAVE

Full Name: Waveform Audio File Format
Originator: IBM & Microsoft Corporation

WAVE is a variant of the AIFF file format. Microsoft in conjunction with IBM created a version for storing chunks in a file. Unfortunately, these efforts failed and whereas AIFF was designed and implemented by a small core grouping, WAVE files suffered from too many cooks that spoilt the standard. Despite the flaws and inconsistencies in this standard, due to its originators it has been one of the *de facto* sound file formats. As it was based upon the AIFF file format, many of the same or similar features exist within the format. For instance, there are again two required chunks for a minimal WAVE file. These are the format and sound data chunks. The Format chunk contains important parameters describing the waveform, such as its sample rate. The Data chunk then contains the actual waveform data. As with AIFF files the other chunks are optional as shown in figure 7. The particular chunk that we are interested in is the Cue chunk, which contains the cue point information for the file.

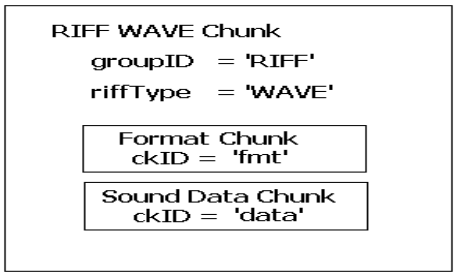

Figure 7. *Graphical overview of an example, minimal WAVE file.*

Each cue point references a specific offset within the waveformData array, and has its own CuePoint structure within this chunk. The cue chunk is used in association with the playlist chunk to hold looping information. As with AIFF a similar C-like language is used to describe the data structures and chunks in the file [15]. The layout of the cue structure is shown below in figure 8. This structure can be contrasted with of the relevant structure in AIFF files as shown in figure 2. The dwIdentifier field contains a unique number, used to associate a CuePoint structure with other structures used in other. The dwPosition field specifies the position of the cue point within the "play order" of the file. The fccChunk field specifies the chunk ID of the Data chunk that contains the waveform data to which the CuePoint refers. The dwChunkStart and dwBlockStart fields are used only for files that contain a Wave List or for a compressed file containing a 'data' chunk. The dwChunkStart field specifies the byte offset of the start of the 'data' chunk. The dwBlockStart field specifies the byte offset of the start of the block containing the position. The dwSampleOffset field specifies the sample offset of the cue point relative to the start of the block. In an uncompressed file, this equates to simply being the offset within the waveformData array. The documentation on this field is very ambiguous, and does not define what it meant by the term "sample offset". There are several plausible possibilities varying from a byte offset, to counting the sample points (for example, in a 16-bit wave, every 2 b7ytes would be 1 sample point), or finally that it is the sample frames (similar to how it is done with the position field in the Marker structure in AIFF files) [15].

```
typedef struct {
  long      dwIdentifier;
  long      dwPosition;
  ID        fccChunk;
  long      dwChunkStart;
  long      dwBlockStart;
  long      dwSampleOffset;
} CuePoint;
```

Figure 8. *Layout of a CuePoint Structure*

As with AIFF files there may be one Cue chunk in the WAVE file and this is optional. The format of the Cue chunk is shown below in figure 9. The ID in the field *chunkID* is always **cue**. The field *chunkSize* is the number of bytes in the chunk excluding the size of the *chunkID* and the *chunkSize* fields. Once the data field *dwCuePoints* is not zero, it is followed by that many cue point structures, one after the other. The cue point structures are packed together with no unused bytes between them. The cue point structures need not be placed in any particular order within the Cue chunk [15]. There are no restrictions upon the order of the chunks within a WAVE file, with the exception that the Format chunk must precede the Data chunk. A related structure is the Playlist chunk which specifies a play order for a series of cue points. The Cue chunk contains all of the cue points, but the Playlist chunk determines how those cue points are used when playing back the waveform [15]. Applications must handle WAVE files similarly to AIFF files in that the application holds the responsibility for managing and updating chunks to ensure conflicts do no arise.

```
#define CueID 'cue '  /* chunk ID for Cue Chunk */

typedef struct {
  ID        chunkID;
  long      chunkSize;

  long      dwCuePoints;
  CuePoint  points[];
} CueChunk;
```

Figure 9. *Layout of a Cue Chunk*

3.4. RA

Full Name: Real Audio
Originator: RealNetworks, Inc.

Real Audio files are rendered from several of the common music file formats such as *aiff*, *au*, *mpg*, *mov*, *snd* and *wav*. The Synchronized Multimedia Integration Language (*SMIL)* is used with this format to create textual descriptions of multimedia component [16]. Another of *SMIL*'s uses includes the creation of multimedia content using a timeline approach. *SMIL* is aimed at a broad audience and to ensure ease of use it is similar to HTML. *SMIL* can be exploited to produce cue points only if the position of the point of interest is known. Another approach using this format is to use the keyframes of the format. A keyframe has all the data about a frame encoded in it. A point to note is there is

always a keyframe at least every ten seconds as this is the maximum interval between keyframes. Reducing the interval between keyframes can be used to improve the ability of the format to seek specific points in the timeline [17].

As this format is design for streaming media and the format is proprietary, there are very few-detailed medium to low-level design documents or format information available. Therefore, we have only provided a brief overview of this format in our research.

4. CONCLUSIONS

With an increasing size of digital multimedia collections, the need for sophisticated search and browsing mechanisms has become more important. Exploiting Cue Points can lead to faster browsing by only playing the noteworthy or distinguishing part of the sound. Without the use of Cue Points, additional time is required to find these characteristic or significant parts for the sound.

After a brief review of the context of Cue Points this paper has introduced the mechanisms used in several common sound formats and has discussed our implementation of an application used to index and create Cue Points. While Cue Points are rarely used in multimedia applications, they offer a mechanism that would allow for faster browsing of digital multimedia resources. This mechanism is available in audio authoring tools, but it has not to the authors knowledge been used in applications for the searching and browsing of audio resources. Before we can exploit these methods, we must firstly need to investigate sound browsing applications. The Sonic Browser is a retrieval application used for browsing audio resources and for managing general sound resources on computers [18, 19]. When browsing for sound files, the Sonic Browser affords the user the possibility of listening to several sound files simultaneously and to navigate through a stereo-spatialised soundscape. Future research aims are to incorporate the use of Cue Points as an additional browsing mechanism in the Sonic Browser.

5. REFERENCES

[1] B. Prothman, "Meta data," *IEEE Potentials*, vol. 19, pp. 20 -23, 2000.

[2] R. M. Warren, *Auditory Perception - A new Analysis and Synthesis*. Cambridge: Cambridge University Press, 1999.

[3] M. Nilsson, "ID3v2 informal standard - Native Frames," Open Source Project, Link ping, Web Page ID3 tag version 2.4.0 - Native Frames, 1 November 2000.

[4] I. Apple Computer, "Audio Interchange File Format: "AIFF"," Apple Computer, Inc., http://www.ai.univ-paris8.fr/~bam/musinfo/AIFF.html, Web Page A Standard for Sampled Sound Files - Version 1.3, 1989.

[5] P. Noll, "High quality audio for multimedia: key technologies and MPEG standards," presented at *Global Telecommunications Conference*, Rio de Janeiro, Brazil, 1999.

[6] H. Purnhagen, "MPEG-1: coded storage of sampled sound waves," Institut für Theoretische Nachrichtentechnik und Informationsverarbeitung, Hannover, Web Page MPEG Audio FAQ, 10 March 2000.

[7] H. Purnhagen, "MPEG-2: coded transmission/storage of sampled sound waves," Institut für Theoretische Nachrichtentechnik und Informationsverarbeitung, Hannover, Web Page MPEG Audio FAQ, 10 March 2000.

[8] S. C. Battista, F.; Lande, C., "MPEG-4: a multimedia standard for the third millennium Part 1," *IEEE Multimedia*, vol. 6 4, pp. 74 -83, 1999.

[9] S. C. Battista, F.; Lande, C., "MPEG-4: a multimedia standard for the third millennium Part 2," *IEEE Multimedia*, vol. 7 1, pp. 76 -84, 2000.

[10] H. Purnhagen, "MPEG-4 Audio: coding of natural and synthetic sound," Institut für Theoretische Nachrichtentechnik und Informationsverarbeitung, Hannover, Web Page MPEG Audio FAQ, 10 March 2000.

[11] F. Pereira, "MPEG-7: a standard for describing audiovisual information," presented at *IEE Colloquium on Multimedia Databases and MPEG-7*, 1999.

[12] H. Purnhagen, "MPEG-7: description of meta-information on sound," Institut für Theoretische Nachrichtentechnik und Informationsverarbeitung, Hannover, Web Page MPEG Audio FAQ, 10 March 2000.

[13] M. Nilsson, "ID3v2 informal standard - Main Stucture," Open Source Project, Linköping, Web Page ID3 tag version 2.4.0 - Main Structure, 1 November 2000.

[14] ISO, "Overview of the MPEG-7 Standard," Moving Picture Experts Group (MPEG), ISO / LEC Standard International Organisation For Standardisation - Organisation Internationale De Normalisation ISO/LEC JTC1/SC29/WG11 - Coding Of Moving Pictures And Audio, October 2000 2000.

[15] I. a. Microsoft, "Waveform Audio File Format, Multimedia Programming Interface and Data Specification v1.0," IBM & Microsoft, Multimedia Programming Interface and Data Specification v1.0, 1991 1991.

[16] G. Flammia, "SMIL makes Web applications multimodal," *IEEE Intelligent Systems*, vol. 13 4, pp. 12 -13, 1998.

[17] I. RealNetworks, "RealSystem iQ Version 8 Production Guide," RealNetworks, Inc., User Documentation Version 8, December 12 2000.

[18] J. M. Fernström and C. McNamara, "After Direct Manipulation - Direct Sonification," presented at *ICAD '98*, Glasgow, Scotland, 1998.

[19] D. Ó Maidín and M. Fernström, "The Best of Two Worlds: Retrieving and Browing," presented at *COST-G6 Conference on Digital Audio Effects DAFx-00*, Verona, 2000.