CONTENT-BASED METADATA IN MULTIMEDIA AND ITS APPLICATION TO SONG LYRICS

A 2006 Napier University Honours Project in Multimedia Systems

By Gilles Dubuc 04009059@napier.ac.uk

ABSTRACT

Metadata has changed the way many organizations manage their multimedia assets. However, from the simple description of general information concerning a document, some metadata applications have gone further into representing the contents of the documents itself. Standards such as MPEG-7 have opened the way to new applications that are still mostly unexplored.

This report looks at the current state of content-based metadata description of multimedia assets. It outlines what the current trends are, what possible future developments research tries to explore, existing standards and their respective accomplishments. It takes a close look at the MPEG-7 standard and investigates the reasons behind its success in some areas and almost complete lack of application in others.

After discovering that song lyrics description is a feature largely not covered by audio metadata formats, such a file format is designed. Named LML (Lyrics Markup Language), this new format is based on XML and XML Schema. An additional piece of software is developed, the LML player, in order to test the format. IT plays audio files while displaying the lyrics information stored in the corresponding LML file in real time. The testing phase of the LML format reveals that the XML validation tools do not to cover some particular cases and that future development of the LML format should focus on a custom validation tool.

Finally through a survey and a series of tests, the current market and technological situation is evaluated in order to determine to what extent the LML format could be commercially deployed. While consumer demand for song lyrics features in their products is clearly confirmed by the conducted survey, the technical difficulties of automatically generating the song lyrics from audio files appear. Developing the automation of such a process could be the subject of further study in order to help LML's adoption.

CONTENTS

FIGURES

1. HISTORY OF THE PROJECT

From its start this project stumbled across many issues that are worth mentioning before going into the assignment itself. The original idea was to develop an application to annotate the content of video stock footage, in order to make a feature-based video search engine. For example, having a search engine that can find videos that contain dogs, not because the word “dog” is present in the video title or summary, but because the visible elements in the video have been tagged in the timeline, automatically or manually. This would provide much more accurate search results, as well as making it easy to isolate sub-elements of interest inside a piece of media. The simple task of describing the single features that can be seen, read, heard, felt in media, leads to tremendous uses and potential new technologies that will be explained in this study.

At first the application seemed reasonable to create, as nowadays the amount of programming libraries available simplifies the task of developing such a high-level application. The MPEG-7 standard, aimed at content description, including video, appeared to be perfect for this task. MPEG-7 has many tools and was designed to be used modularly depending on what is needed out of the standard. Additionally, some simple similar annotation tool prototypes have already been developed using MPEG-7, notably one by IBM research labs. Only improvement over the existing applications’ capabilities was necessary.

The next issue that emerged was the lack of MPEG-7 library. There was only one available, limited to a very specific platform and with such license limitations that it could not be used for this project. At this point the focus of the project shifted from developing the annotation application to creating an MPEG-7 library, as such toolsets being unavailable was a clear factor causing the absence of MPEG-7 compatible software. Creating an open multiplatform MPEG-7 library, even if the implementation would be very incomplete by the end of this project, seemed like a good advancement for the community. Many aspects of the MPEG-7 standard (such as BiM<->TeM conversion), which are crucial ingredients, have been designed on paper but seem to have never been implemented in any tool or library. This also led later in the project to question how effective MPEG-7 could be if it has never been tested as a whole, as no official or unofficial library supporting all MPEG-7 features has been developed to this date.

After getting hold of all the literature available on MPEG-7 (there is only one book published dealing with the subject, [5] B.S. Manjunath, P. Salembier and T. Sikora, “Introduction to MPEG-7: multimedia description interface”), MPEG-7 became much easier to understand. Regrettably the technical details of the specification were missing in order to implement an MPEG-7 library. It became necessary to read the MPEG-7 specification, a very technical document. It is where the main issue lied. Not only is this specification almost impossible to get by regular academic means, unavailable from all major UK libraries, but it would also cost Napier University £1400 to acquire this set of documents describing the MPEG-7 specification. This did not seem reasonable for the scope of an honours project and forced once again to shift the focus of the research. The study now

focuses the wider world of multimedia feature-based tagging and metadata, in a bigger context than just video stock footage.

Undoubtedly the original “MPEG-7 based video stock footage annotation tool” topic is a long stretch from the current “Content-based multimedia metadata and its application to song lyrics”. It shows how some aspects of these technologies are not quite ready for prime time for various reasons. In this study, in addition to studying the currently available tools, predictions about possible future uses of feature-based content description are made. The potential outcomes of such technologies will greatly increase user experience and can revolutionize how media is produced and consumed. Even further than what the current summary-based metadata efforts have already achieved.

2. METHODOLOGY

It has been quickly established that there was only one book about MPEG-7. Which is the reason why, once that book obtained, research of documentation has been focused on journals. A few key journal articles have been found, which led through their reference to further research works. The amount of research concerning MPEG-7 other than the one part of its design was relatively limited. After realizing that most of the literature had already been found, new solutions had to be found in order to gather more information.

This is when interviews seemed to be an interesting prospective. The choice of contacting people via email might not have been the best, as out of three only one person replied to the interview. This insight from a company using MPEG-7 and commercializing tools using this standard has helped getting a better view of its current state in real applications. Phone interviews could have been an alternative, but it is unclear to what extent it would have helped getting more replies.

When the development of most of the prototype was finished, it became necessary to evaluate it. The LML player prototype as such had been created in order to evaluate the LML standard, which is the reason why the evaluation did not focus on the LML player. As for the standard itself, the decision was made to rely on technical means to test it, like the W3C validator. Getting it peer-reviewed would have been a better option, but it did not seem a good idea for an honours project, as the collaborative aspect of such a task could have questioned the authorship of the standard corrected by involving peers.

The other issue that emerged during the prototype development was that the idea of the LML format was based on the assumption that there would be a need for it. The solution adopted to confirm this was to perform a survey among digital music consumers. The way the survey was designed could have been more researched, but the final results show clear tendencies and the data makes it easy to draw conclusions. Further studying of consumers could help determine what features within the LML format were most expected. But given the core functionalities covered by LML this would apply more to further extensions of the format.

3. LITERATURE REVIEW

In order to study what solutions help generate media descriptions based on their features are currently available, most of the literature review focuses on what metadata formats and standards exist for the main types of media. The present state of metadata, which generally only scratches the surface of features description, then leads to studying what has been achieved in terms of feature-based description of contents outside of the boundaries of metadata. The literature review examines video and audio independently, even if it has been kept in mind that those media inevitably mix, as most of times video contains audio. The difference in nature of the two makes them easier to comprehend by separate studies, linking the two being only a matter of time synchronization.

3.1. VIDEO

Metadata in the video world can be considered still being in its infancy, despite the quantity of existing standards and efforts. Among those standards the adoption rate is very variable, but the need to organize large libraries of video content will certainly make efficient solutions relying on feature-based tagging a necessity.

3.1.1. DUBLIN CORE

Standards concerning the summary information of videos, such as Dublin Core ([9] The Dublin Core Media Initiative (DCMI), http://www.dublincore.org/), are very successful. They provide very basic metadata about the video footage. This simplicity makes Dublin Core the most widespread standard used for tagging the main properties of a video. The main advantage of Dublin Core is its straightforwardness of utilization. It is very effortless for developers to use Dublin Core libraries in order to make their applications compatible with the standard. Employed by various institutions and professionals, this standard is a good choice in order to store information such as author, summary, visual ratio etc.

What composes the strength of Dublin Core is additionally its main weakness. By focusing only on the summary information, Dublin Core's official philosophy is not to extend, and in particular not to describe the inner features of the videos, such as what is visible on screen over time. It covers what is already essential for the most common applications that rely on descriptions of videos but will never go further. Video search, for example, will never get more accurate than probing the full text summary stored in the Dublin Core part of a video file. In a nutshell, Dublin Core is a good standard for the basic functionalities required for description of video content. However it will never help explore ways of express video content other than a simple summary.

3.1.2. MPEG-7

When dealing with feature-based description of content, MPEG-7 is inevitably the standard that goes the farthest in this domain and provides the most sophisticated tools. MPEG-7 is developed by the Moving Picture Experts Group

([6] ISO/IEC JTC1/SC29/WG11, "MPEG-7 Overview (version 10)"). It is aimed at providing a solution for the description and cataloging of media contents. From audio, video, still pictures, to 3d models various types of media can be handled by MPEG-7. Both automatic and manual tagging have been taken into consideration in the design of MPEG-7. This proves to be very crucial when dealing with generating descriptions of the inner features of media, as both philosophies require different toolsets in order to be efficient.

MPEG-7 seeks providing multimedia content description for any type of content used for any kind of application. The target being very wide, MPEG-7 is provided in various modules that can be implemented or not, depending on which use is made of the standard. Sorting which tools will be employed and how they will be is a tedious task. However it guarantees future interoperability of the system with MPEG-7 described footage from other sources, and of the developed footage library with other content management platforms.

It offers different levels of abstraction in the description of content, from very low-level information like color, texture, melody, that are likely to be generated automatically, to high level semantic descriptions of what is happening and what can be seen in a video, a picture, or what mood a song has. The scope of this honours project makes both aspects interesting. Low-level characteristics can be especially useful, for instance to determine visual similarity between different video clips. Such a characteristic would greatly help develop functions like "get more video clips like this one". And the high-level semantic descriptions would be the core of the search mechanism. This also applies to the other types of media covered by MPEG-7.

3.1.2.1. STRENGTHS OF MPEG-7

MPEG-7 provides a lot of freedom to developers. It is possible to create graphs and hierarchies of any type of information. The management of controlled vocabularies can be very efficient. For example a term can be given identification based on a URI, and be defined in many different languages. This provides direct translation possibilities to all the keywords, or even all the words present in controlled vocabularies. The implications for a search engine are vast, as by simply translating the terms present in the controlled vocabulary, the entire search engine can seamlessly switch the language it is currently using. Relying on that standard also guarantees the robustness of the tools that are helping to develop this critical aspect of the project.

The direct applications for a MPEG-7 based search engine go beyond what is currently available. For instance, the user could be given the possibility to search video using a great range of features, from the overall color of a shot, to how fast the camera or the objects in the video are moving, and of course semantic elements ([3] G. Gaughan, A. Smeaton, C. Gurrin, H. Lee, K. McDonald, "Design, implementation and testing of an interactive video retrieval system"). The possibilities seem endless. Once the effort of tagging and describing in detail the videos has been completed. Fractions of this process can be automated, especially regarding the low-level features of the videos. Shape recognition could help define using MPEG-7 what in a video is foreground, background, characters or moving objects. This applies to still pictures too. It could lead to tremendous functionalities, such as color correcting specific sub-elements of a video, or chroma-keying any element, isolating semantic parts of a picture or video

automatically.

The large amount of tools provided by the standard itself helps greatly the description process. While letting developers decide on the specifics, it provides all the necessary guidance to design a system that helps describing video content. It also integrates very well with MPEG-4 (becoming MPEG-47 when it does), which is already widely adopted. It can even be streamed. It could be used for live broadcasts, for example providing detailed information about an ongoing sports event while the video is being broadcasted, or similarly providing information regarding a radio broadcast.

3.1.2.2. WEAKNESSES OF MPEG-7

MPEG-7 does not standardize vocabularies. As a result, solutions will have to be found elsewhere in order to develop the most standard controlled vocabulary for a specific type of media. This is a very risky task, as defining a one can be very subjective. MPEG-7 on its own cannot provide a solution for this, only offering the tools to manage vocabularies. Further research will have to be done in order to help solving the issue of how the terms should be selected and associated.

As a standard, MPEG-7 aims at giving all the tools essential to any kind of media description application. This large target makes it difficult for developers to decide which tools are useful or not for a specific task. For example in the specific situation of video stock footage, all the audio tools become unnecessary. And some tools provided really seem to have been designed for very specific tasks (affective response measurement of people exposed to the media, financial data regarding the cost of production and potential revenue of the media). This adds to the big amount of tools in the standard that are useless in the vast majority of projects based on MPEG-7.

MPEG-7 is a very large-scale project for a standard and the industry has not truly followed the initiative. This can be a risk, as the standard itself may not be that good, since the amount of tools available has not quite developed, thus the standard has not been tested in real life situations. Moreover, it could be that the industry has not yet realized the necessity of per-segment and multi-level description of media content. The need to increase the performance of media search engines will probably drive more products into supporting MPEG-7.

The main weakness of MPEG-7 regarding video, but this can be applied to most video content description solutions, is that there is a very small amount of software available to make use of it. Of course this is subject to change if MPEG-7 finds wider success. However, so far there is only one commercial MPEG-7 production tool available, called Frameline 47 ([7] www.frameline.tv). It was only released in February 2006, 5 years after the introduction of the standard. This is the only solution available to annotate video content using MPEG-7, and there is no open source or free alternative to it. James Engwell, the Frameline 47 spokesman, confirmed in an interview conducted for this project (Appendix 1) that in order to make use of the MPEG-7 metadata generated by this software, it is necessary to build one's own tools. Mr Engwell also admitted that during the development of their software the company did not purchase the standard's specification due to its cost, but based their work on the very same book ([5] B.S. Manjunath, P. Salembier and T. Sikora, "Introduction to

MPEG-7: multimedia description interface"). And that same book proved to be insufficient in describing the standard for the purpose of this research project. Making software based on a summary of the specification raises concerns about the true compliance to the MPEG-7 standard.

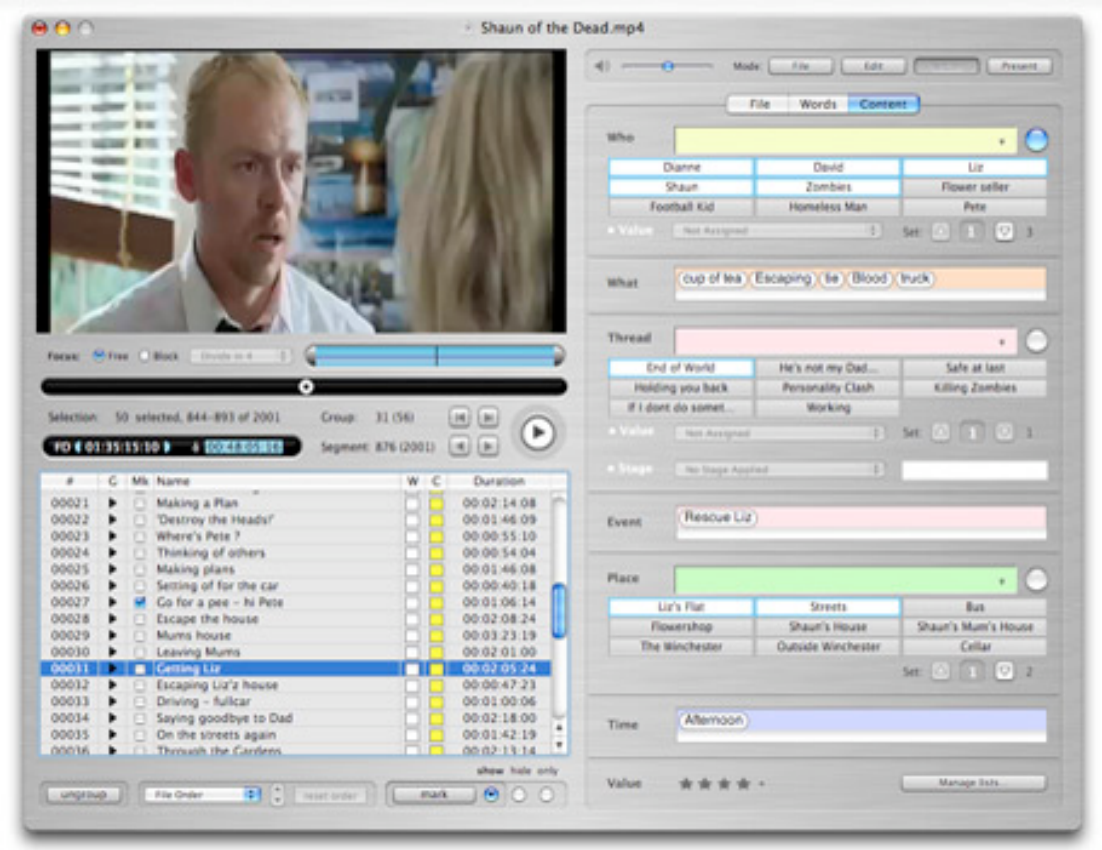

Figure 1. Frameline 47, the first commercial tool using MPEG-7. Source: www.frameline47.tv

The above issue is commonly caused by the great cost of the standard documentation. Napier University declined the purchase of the £1400 documentation for obvious budget reasons, and this is probably the situation for most UK universities, as the documentation was not available in any library in the UK. This evidently impedes academic use of the MPEG-7 standard and could be the main cause of the implementation flop of the standard, particularly regarding video. This budget issue also impacts on small businesses, which would need to allocate that money upfront for a project relying on MPEG-7. Without being able to get an idea about how good the standard and its documentation can be.

The cost of the MPEG-7 specification could be an issue of the past if there were good programming libraries managing the technical side of MPEG-7 and compliance to the standard. Unfortunately there is only one library available ([34] Joanneum Research, "MPEG-7 library") to let applications make use of MPEG-7, and it is limited to a single programming language on a specific platform. In addition to which it only covers a small fraction of the MPEG-7 standard. There is clearly a need for a more open alternative, as the lack of library could be another explanation of the very small amount of software relying on MPEG-7.

The previous MPEG standards took a considerable amount of time to get integrated into production tools, and the amount of tagging and metadata-based

products that appeared recently will soon make the use of MPEG-7 more widespread. Once the most common video editing tools will start supporting MPEG-7 and MPEG-47, the technology will be able to take off. The need for the various video content providers to interact will also be inevitable as people will want to make use of their video content anywhere, on any device, from any source.

3.1.3. VIDEO SEARCH ENGINES

Video search is still generally inefficient for numerous reasons. The main one being the relative inaccuracy of results (even with the most developed video search engines available today). There have been many attempts to create a video search engine that would work well with any kind of video and get the results need by the user, but the current technology still cannot perform as well as what is available for text search. Folksonomies ([33] A. Mathes, "Folksonomies - cooperative classification and communication through shared metadata"), better represented by YouTube and its competitors, is an interesting new experiment. Previously, the tendency was to use controlled vocabularies to describe best the contents of a video. But with folksonomies it became the opposite, as users are free to use any kind of textual description for a video. Rules and specific vocabularies are not needed to perform such a task.

The advantage of folksonomies is that videos being tagged by the same kind of users who would consume the information, it is more likely that they will use the same type of words to describe the content than they would use to search for it. This merging of media consumers and producers has the big advantage to solve the issue of controlled vocabularies created by the media producers that do not match how the users would search for the content. Of course this mass of users could be very heterogeneous, but it is probable that people with the same areas of interests will use the same kind of vocabulary to describe and search for content.

It brings up an interesting question regarding what could be done with video search in a corporate environment. Instead of letting people who produce the videos tag it or even people specialized in tagging do it, it would be more interesting to let the users do it themselves. The biggest difficulty is to motivate this tagging, as consumers are less expected to be interested in doing so. A policy of having the consumer of a search engine forced to produce content or tags every now and then would maybe augment the effectiveness of the tagging, as seeing both sides would help make the need of accurate tagging more obvious to the search engine's users.

Evidently the main drawback of such a free tagging system is that it can effortlessly be abused. For example during the world cup many videos were tagged with the keywords related to the world cup on YouTube, even when the concerned videos were completely unrelated to that event. This misuse, aiming at increased popularity of the videos produced in the online community by the user, is very difficult to stop. Particularly when the quantity of videos produced makes it nearly unfeasible to watch everything in order to check that keywords are not abusive. In a corporate environment this would be less of a concern, as it is likely employees using a professional video search engine would not try to make their videos more popular than the ones tagged by fellow workers as much as users do on community websites.

But the “search engine users tag the content themselves and eventually produce content” philosophy cannot be appropriate to all corporate uses of video search engines. For example, a company that sells video stock footage cannot afford to make its clients tag the videos. The clients pay for a service and expect the search engine to be efficient. And it is the company’s best interest to have accurate search results in order to increase sales. If the video cannot be found it cannot be bought either.

This previous example shows how some video search engines truly need to rely on controlled vocabularies while others get all their strength from the free-for-all user tagging of content. In free tagging people will generally not create as many tags as possible for a video, just put what they consider is enough. With more controlled environments, search engines could use all the power of feature-based tagging. By not only describing the video as a whole, but describing what is inside it, what its structure is and all the visual elements that are in it, users can get access to a whole new control of searching.

Feature-based tagging can tremendously increase the precision and especially the quantity of search results. For example a user, who operates a stock footage repository to make his/her own film, needs videos of dogs because he/she is making a short news piece about them for a TV channel. Traditional free tagging and controlled vocabularies would return only results of video that deal with dogs. The summary of those videos would mention that one of the main topics of those videos is dogs. But this means a big amount of videos that do contain dogs will not be returned, because as a whole they do not concern dogs. This is a big issue that feature-based tagging can solve.

By tagging the semantic elements present in a video sequence by sequence, the user could find many video sequences with dogs in them, taken from video footage that does not specifically deal with dogs. In this example, dogs being a fairly general feature found in many videos, it would increase the amount of results, and of choices for the user to pick from. Evidently, too many results do not help either, as the choice becomes hard to make, but again feature-based search can help.

The user could specify that results would be organized by sequence length, by added length of sequences in the same footage, by size of the feature on screen. He/she could prefer to have the biggest dog possible visible on screen, which would make results organized by average size of the element on screen relevant. Similarly the user might want to have the same dogs appearing in different sequences. Such advanced search features are completely lacking in currently available video search engine solutions and would easily give greater power to the search engine user in order to find exactly what he/she is looking for.

The main issue of such feature-based search is the cost of tagging the video data. For a company selling stock footage, tagging the content of the footage available in their catalogue precisely would increase sales as users would find more easily what they need. But on the other hand this detailed tagging would cost a lot to implement if it cannot be done automatically. Even with automatic tagging, the current computing power needed to detect such complicated elements as dogs that can come in many sizes and shapes, using shape and

motion recognition, is still very high. Whoever would make that feature tagging process automatic would need to invest in heavy computer power.

The solution to this could be adaptive tagging: tag in-depth what needs to be by adapting according to the topics searched by users. Still seeing this from a stock footage company's perspective, it would be easy to see what kind of topics users would search for. This way issues could become apparent when for example some keywords get very few results meaning low sales on those topics. In that case it would seem a good idea to search for this specific feature in other videos available in the catalogue by using feature-based description. Similarly, extremely popular topics could be given a special treatment by particularly tagging all videos that have those features and give more search options like described before. Searching for dogs? What size? What breed? How long does the sequence need to be?

Such adaptive level of tagging would help reduce the costs that could involve systematic in-depth tagging of video features. Even if the users do not take part in the tagging process directly, their input helps improve the accuracy of the search results by tagging in more detail what needs to be. And keywords that are never searched for over a long period of time could also be put in an "unnecessary" list of keywords, which would indicate to the video taggers or tagging system what does not need to be mentioned when describing the features present in a video.

It is difficult to balance in the same search engine free tagging and controlled vocabularies. Any proposed solution would need to be modular enough to be applied to the use intended. Dealing with standards, MPEG-7 provides that freedom, by giving the possibility to create various levels of content description. Two different search engines, one very "static" relying on controlled vocabularies set up by the content producer and another one completely based on free user content tagging, could both heavily use MPEG-7 in order to work. Which would in theory make the MPEG-7 described content compatible between the two search engines.

But that is where compatibility does not meet efficiency. Video footage tagged by users with free tagging, the description information being stored in MPEG-7 format, would be inadequate in a controlled vocabulary-based search engine, because the tagging does not meet the requirements of the controlled vocabulary. In that way MPEG-7 documents are technically compatible but in practice can be very incompatible because of how differently the data was described. Not to mention that they could use different modules, all part of MPEG-7, having almost no common module. In that way, MPEG-7 should be considered a toolbox rather than a compatibility standard. It provides a framework that facilitates the creation of content description tools, but does not solve the issue of describing footage once in order to be used in many different ways. Sadly this points to the fact that to a certain level every solution will have to be very proprietary, especially in the way the content is described and how the vocabularies are chosen. As such, the necessity to use a standard highly decreases as the description will need to be translated somehow in order to be used in a different context than the one it was originally done for.

There are examples of initiatives in normalizing MPEG-7 data coming from different sources ([27] G. Tummarello, C. Morbidoni, P. Puliti, A. F. Dragoni and F. Piazza,

"From multimedia to the semantic web using MPEG-7 and computational intelligence"), which clearly shows the big drawback of providing too much flexibility, the data being in the same format, but still incompatible in terms of practical use unless it is homogenized manually.

But let us not forget that feature-based content description can improve greatly the functionalities available in a search engine. Imagining that there would be an affordable solution in the near future to automatically describe in detail a video from its colors, textures, semantic elements to more difficult features such as mood and emotions of characters, let us imagine what could be achieved in future video footage search engines. Being able to instantly find all close-ups of eyes, all aerial day shots of a specific city from a wide library of varied footage would tremendously change the way video is made for content producers.

It will give an immense power of recycling footage, saving production money by finding for a low cost the exact sequence that a filmmaker would need to shoot would change the way this business currently works. The video stock footage production would greatly increase and filmmakers would rely more and more on using it in order to save time and money. This would become some kind of film outsourcing by letting others do the filming, the filmmaking becoming more centered on editing and shooting only what is necessary.

In another field, journalism, face recognition would greatly facilitate archive footage search. For example any filmed public appearance of a politician could be found, even in events where this person was not the main focus.

But on the other hand the very same feature could help increase the performance of more contradictory services such as searching for every appearance of a specific person on CCTV archives. In both last cases search would be most likely using the picture of the person searched instead of the name, relying on automated face recognition tagging and search. Such a very targeted use is likely to be efficient in terms of automatism compared to more general shape recognition tagging aimed to be working for any kind of use.

In a nutshell, describing video content according to the footage features is inevitable in the future of search engines. Cross-search engines compatibility still seems very hard to reach given the extremely varied ways content can be described using the same tools depending on the final use. It probably means standards initiatives such as MPEG-7 are not likely to become widespread, due to the difficulty to exchange metadata between platforms even if the data is technically in the same format. It appears that every video search engine will have to find its own solution to be more efficient, depending on what use is needed, and standardization could happen by harmonizing the toolsets, such as what MPEG-7 aims at providing. A better solution would probably be having different separate toolsets aimed at different uses. Earlier it has been shown that many search engines will fall either into the category of free tagging or controlled vocabularies. Both are similarly stored but the inherent difference is probably a hint that solving the issue of video content description will probably be to provide different toolsets and identify what different needs can be grouped in the same toolsets, rather than trying to provide tools for everything and ending up with too many tools to choose from, which is one of the big drawbacks of MPEG-7.

3.1.4. FUTURE DEVELOPMENTS IN VIDEO METADATA

Not taking into account the issues of real life implementation of the standard mentioned previously, the potential uses of the MPEG-7 standard could totally change the way video media is produced and consumed. Features such as shape recognition could revolutionize video editing. If an MPEG-7 compatible camera had onboard shape recognition, the video straight out of the camera could let the editor select any sub-element of the video. This would make chroma-keying obsolete, as the shape masks generated by the camera and stored in MPEG-7 format would give the video editor the possibility to isolate seamlessly any character, any object present in the video. Face recognition, also part of the MPEG-7 standard could help a filmmaker instantly filter the raw footage of the film by selecting what actor or actors should be seen on screen. It is currently one of the most developed and used features in the MPEG-7 format, with research and application mostly focused on surveillance solutions. Those systems relying on MPEG-7 have proved to produce very good results, with more than 90% of successful matches on the test sets provided by the MPEG consortium in some cases ([31] L. Torres, L. Lorente and J. Vila, "Automatic face recognition of video sequences using self-eigenfaces"). The current facial recognition systems generally need a few pictures of the same person from different angles to be able to train and find other occurrences of the same person in video clips. Solutions based on automated generation of the 3d model of a face based on a single picture, in order to generate the needed other angles have been proposed ([32] WS. Lee and KA. Sohn, "Face recognition using computer-generated database"), but are still far from the success found with traditional approaches based on a series of photographs.

Those are just simple examples of what could be done if MPEG-7 or a similar feature-based standard was implemented in future digital film cameras as well as linear editing systems, but there are probably more tremendous possibilities that are hard to imagine before the standard is integrated in those systems. This shifts the paradigm from digital video being the representation of moving pixels over time, to be the semantic, visual, shape, face recognition and many more fields of data of something being filmed. This is how the future of digital video is likely to be, this increased amount of dimensions available in video footage would greatly increase the freedom of what can be achieved. Of course the easiest way to make this work would be if the creation of all this additional descriptive information would be automatic, whether done at the camera or the editing system level.

Despite the various issues that make the use of MPEG-7 very marginal, it has very interesting tools that could revolutionize the way videos are described. Even if it fails as a whole, MPEG-7 is likely to inspire future ways of describing video content and how that can improve performance of video applications and possibilities for both content producers and consumers.

3.2. AUDIO

After text, audio is probably the most widespread media across the globe. It becomes more and more uncomplicated to explore audio libraries online or offline and consume audio data. The technology has mostly focused on quality of the sound and compression over the last few years, in order to provide to users the best quality possible with the smallest amount of storage needed for the data. Metadata in the world of audio has generally been constrained to describing general information about the audio data, like the author, copyrights, etc. More in-depth feature-related metadata is still not widespread.

3.2.1. DOLBY DIGITAL

An example of metadata used for more than describing generalities about a file is Dolby Digital metadata. Dolby digital is still commonly focused on film and home cinema setups, and it does not give the impression yet that the music industry is heading towards using it. But it is likely a similar standard will appear in the next few years when surround-sound music becomes more successful. Dolby's technology ([12] Dolby Laboratories Inc., "All about audio metadata", 2001) focuses on the mixing of the audio.

The metadata present in Dolby Digital describes the mixing of the various audio channels over time. It is the portrayal of a feature (mixing levels) for multi channel audio, making it the first true feature-describing solution for audio. Of course the purpose being only focused on mixing, it can be considered very limiting compared to the amount of features that can be described in audio data. But the way it has been developed by Dolby makes it very powerful for the specific purpose of mixing. To simplify how it works, the operator doing the mixing of Dolby Digital audio data can create different mix sets depending on the target playback devices.

This way, a digital TV channel can make different audio mixing sets in order to get the best playback experience for the users, whether they use a full surround-sound setup or an old mono or stereo TV set. In addition to this freedom given to the audio content producers, the standard leaves open the possibility for the content creator to let the consumers adjust the mix themselves. With Dolby Digital it is achievable to increase or decrease the volume of separate audio tracks in order to have for example louder dialogues in an action movie over the rest of the sound. And this option could be given even to users who do not have a surround-sound system.

Other attributes include dynamic range control, which helps facilitate the compression of dynamic range in order to get the full audio at reduced volume, in situations for example where the user does not want the sound to be too loud for neighbors. In a nutshell, Dolby Digital metadata is very effective for mixing control, guaranteeing that the mixing information of the various channels remains the same all along the distribution path of the audio data, from producer to consumer.

It is probable that the advancements provided by Dolby Digital will become more widespread in the music industry, by letting consumers change the mixing setting of the multiple channels of the music. Artists and music producers are

perhaps not pushing into that direction, but when consumers will become used to such mixing control in films the demand for it in music might grow. The advantages of such a feature with music are obvious when different listeners will prefer the sound of a musical instrument over another, or simply want to listen to it without hearing the singers, thus making the release of karaoke and instrumental versions of songs something obsolete.

3.2.2. TAGGING STANDARDS

The most known format of audio metadata is probably id3 tags that are present in MP3 files. This is a de facto standard originally created by an individual ([21] M. Nilsson and J. Sundström, "The short history of tagging", http://www.id3.org/history.html). It now has five different versions, which cause compatibility issues and still none of these versions has been developed by a proper standard body. As a result, the quality of the specifications greatly varies from a version to another, bringing other troubles related to implementation and interoperability of software relying on id3 tags. Each application letting users tag with id3 arbitrarily decided the version of id3 it relied on. The nature of the id3 tags is simply to provide general description of the audio file, such as author, date, genre, etc. This is what most audio metadata formats tend to do, along with digital rights management.

In the open source world, the Vorbis format has its own metadata format, known as Vorbis comments ([22] Xiph.org foundation, "Vorbis I specification", http://xiph.org/vorbis/doc/Vorbis_I_spec.pdf, 2004). In this situation it has been designed by a community of developers rather than an individual, but still has not been developed through the same standards process as more widespread standards such as the ones created by the MPEG consortium. Vorbis is a good initiative but still has not taken off and is still mostly used by tech-savvy Linux users rather than the general consumer.

It is surprising to see that the audio and music industry still rely on very simplistic metadata to describe data. Many initiatives aim at providing standard digital rights management in order to make music from various stores compatible, but very few focus on describing the content better. Of course in a context such as online music stores, customers are likely to search for a specific artist. In which case describing a song file by its author and performer is enough to let the users of the store find the music that they like. But feature-based audio tagging can bring much more than simply classifying songs and albums by genre and artists.

3.2.3. EXISTING AND FUTURE APPLICATIONS IN AUDIO

Some advanced research is being done in the area of automatically generating the feature description of music, such as mood, intent of the artist, instruments being played etc. Most of the time this research is not directly linked to storing that data as metadata. But ultimately automatic determination of the linguistic features of a song ([13] B. Whitman, D. Roy and B. Vercoe, "Learning Word Meanings and Descriptive Parameter Spaces from Music"), such as adjectives (eg. sexy, loud, fast) describing a song, will lead to the possibility of storing this information along the audio data itself, and even making this analysis vary along the timeline.

So far most projects attempt at analyzing the mood of a song by analyzing the

audio data of the song as a whole, but adapting the same process to apply to sub-sequences of a song could lead to much more interesting and accurate description of music audio data.

The main issue that remains is standardizing the vocabulary used to describe music or audio. In the example above the generation of the adjectives set relied on music reviews gathered on the web by a crawler that searched for reviews concerning songs part of the test set. The main drawback of such a choice is that the demographics of consumers who write reviews on the web is probably not the same as of users who read the reviews, or even music buyers in general. In addition to that, reviewers who try write them as seriously as possible are likely to search for a more elaborate vocabulary than simple adjectives or words that people who listen to music would use in general.

The dilemma concerning which kind of vocabulary to choose is difficult to make, and it is probably the reason why standards such as MPEG-7 which try to provide the tools needed to store such feature description of music tend not to offer sample vocabularies, but only the tools needed to create them. The matter of knowing whether there is or not one vocabulary that would suit music description in general is irrelevant, since the nature of the tools developed and their target audience seem to make it very likely that each service relying on feature description of audio data will have its own vocabulary to do the description.

Ultimately the issue of interoperability in audio description arises. A song described using the specific vocabulary of an audio service would not benefit from the metadata of this description in another audio service using a different kind of vocabulary.

In the specific case of classical music very advanced tools have been developed. Notably Variations2 ([14] J. Dunn, D. Byrd, M. Notess, J. Riley, and R. Scherle, “Variations2: retrieving and using music in an academic setting”). This search engine introduces features such as describing the inner parts and sequences of a song, with the ability to switch between interpretations in order to compare them. The aim being to let users of the search engine, music students, teachers and researchers very easily compare the various ways a classical song can be interpreted. In such a context it is straightforward to sense the advanced features that can be developed simply by having the metadata added to the simple audio data.

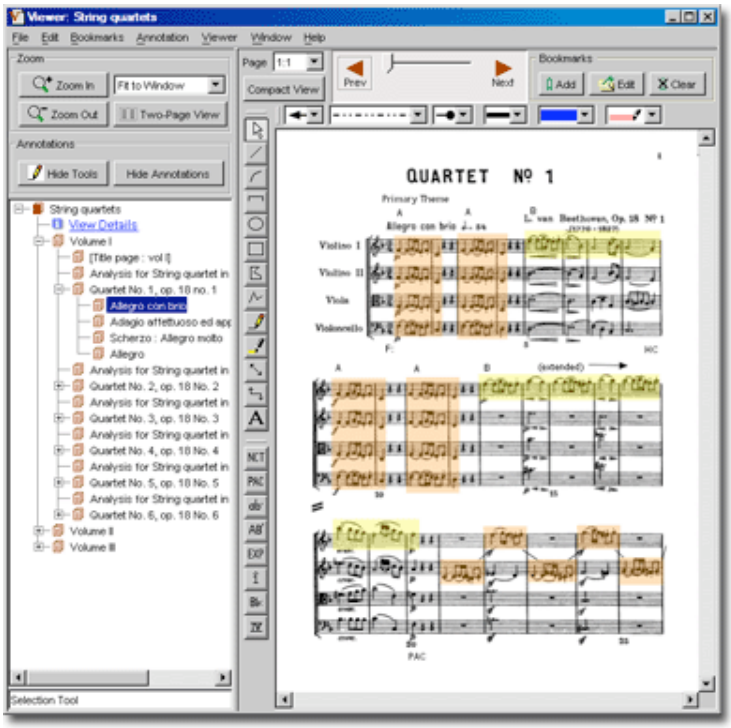

Figure 2. Variations2's annotation tools. Source: Indiana University, variations2.indiana.edu

Metadata does lead to impressive features that would be hard to achieve without it, the main challenge being to develop techniques automating the generation of the metadata at a low cost in order to make the new features economically interesting. Bad uses of metadata in its early years have unfortunately earned a bad reputation regarding its operation among some companies and even libraries.

The current systems used to suggest songs they might like to customers are highly based on studying what the current users buy and suggesting to consumers what others with similar tastes bought. But this is somehow limiting, as new artists are not likely to be suggested if not enough people buy their music to make them relevant to suggestion tools. Describing the features of a song helps find similar ones by comparing those features. Two pieces of music could have many rhythmic and frequency similarities without being classified in the same genre, and those high level audio feature descriptions could help suggest to users songs and artists they might like, based on the structure and audio content themselves. Prototypes of such systems exist already ([15] P. Cano, M. Koppenberger, N. Wack, "Content-based music audio recommendation"), based on audio fingerprinting, which generates a unique signature for every song, calculated from its acoustic and melodic features.

Furthermore, if the structure of the songs could be analyzed thanks to such feature-describing metadata, users "taste profiles" could be generated from common structures found in the various songs a specific person has purchased. It would help being able to transcribe in notions of rhythm, structure, sounds, what makes someone like a song. Having such power could even lead to online stores being able to tell music studios how a song should be structured and what features it should have in order to appeal to a target audience. Such a perspective can be scary regarding the creativity aspect of making music that could be highly reduced compared to such a new kind of "engineered"

composition made by analyzing what people actually like in a song, but after all the audience will decide and that is still what makes music successful or not, independently of the way it is made.

Additionally, such tools, helping music studios check if their songs will be hits in sales or not, are already available ([23] Polyphonic HMI, "Hit Song Science technology"). Whether they rely on feature-based metadata is impossible to know, due to the secrecy existing in that branch of business, music companies not wanting to be publicly known that their hit songs are getting more and more engineered rather than made imaginatively by artists.

MPEG-7 has a part dedicated to audio description that involves describing the features of the audio data, in ways similar to what has been described above. MPEG-7 description of songs can contain lyrics, musical instruments description, mood, melody, etc. All the description can be streamed and is very precisely synchronized to the playback of the audio. So far that branch of MPEG-7 seems to be the more successful in terms of real-world applications and adoption. For example services relying on this standard exist that use the signature calculation of a song. This powerful feature of MPEG-7 makes it possible for a person to search music in a library by simply singing out loud a part of the song ([24] Fraunhofer Institute, "Query by humming", [25] B. Pardo, J. Shifrin and W. Birmingham, "Name that tune: a pilot study in finding a melody from a sung query"). The singing can be imprecise and still generate accurate results. The current state of such technologies is still at the research level. Commercial services letting users call a number on their mobile phones to identify a song being played on the radio or in a public space or by humming it could work well thanks to these existing technologies, and prove how strong the tools for audio provided by MPEG-7 can be.

With melody detection and creation of metadata containing the properties of melodies, it will be made much easier to check if a portion of music is copyrighted or not. Technologies designed for this use have already been patented ([26] Audible Magic Corporation, "Content Alert"). Be it for a composer willing to check if his/her creation has not been made and copyrighted before, or for copyright owners willing to enforce their rights when a copyrighted melody is used or performed (for example as a mobile phone ring tone or sung in public). It would be interesting to see melody metadata integrated or linked to DRM in order to show what author has rights for which melody present in this song, as more and more genres rely on sampling previously created melodies into new songs.

This way if only excerpts of a song are used in a movie for example, and some melody made by a specific author present in another part of it is not present in the sequence played in the movie, only the artists concerned by the melodies heard in the sequence should receive royalties. Such a system would make the repartition of royalty fee more accurate and in some cases more automated, for instance helping radio broadcasts who need to know what they need to pay to which composers and performers.

Evidently, a humming search engine like the ones described above could not play the audio of all the songs in the database in order to compare it to the query sample, computing power required for this would be far too high. That is why most of those systems rely on computing a signature based either on the

whole song or the extracted melody. Such information can be stored in MPEG-7 format, the standard providing tools helping compute the signature of a song in order to facilitate fast query and comparison in a database. Some of the applications mentioned above (particularly the one developed by the Fraunhofer institute) rely on MPEG-7 for storage.

Another information area that could be a wanted feature in popular music is lyrics. As of now they are copyrighted just like the audio of a song itself, and more and more copyright owners fight against online repositories of song lyrics. Same applies to music scores. Both could easily be integrated as metadata to digital music. This way, future portable music players could have functionalities letting users read the lyrics of a song on-screen, the display being synchronized to the song as it plays, in a similar fashion to karaoke systems. Such an application would be relatively easy to create with the currently available technologies and could also be automated via voice recognition in order to synchronize the lyrics to the music without the need for someone to do it manually. What looks like a simple feature could give the edge to a digital music store that would offer such a service along with compatible portable players.

The issue would be for such a functionality to be defined as a standard before proprietary solutions appear and become de facto standards. This matter will be studied in the prototype, which will aim at defining a file format for song lyrics storage and playback based on XML. The idea is to focus on this core functionality, unlike MPEG-7, which tries to cover all aspects of audio metadata at once.

4. A SONG LYRICS OPEN FORMAT

Among the many potential outcomes and applications that could happen thanks to feature-based content description described in the above study, one appeared to be missing while it can reasonably be realized with current available tools. Lyrics in songs could effortlessly be associated as metadata to the audio data. There has been a previous attempt to do so, called 4ML, unfortunately to this date the project's website has shut down and the last signs of activity happened in 2004. 4ML's scope was broader than lyrics, including song scores and other additions.

Given the size of the current digital music market and its growth speed, any new feature for a music/portable player platform will be what could potentially make the difference with the competition and attract customers. All the digital music offers, from stores to digital music player offer a wide range of configurations but always rely on the same functionalities. The simple feature of having the lyrics of the songs displayed and synchronized with the music while it is being played on a computer or a portable player is something that will give an edge to the digital music platform that offers it first.

The aim of this project will be to define an XML-based format in order to store song lyrics along with audio data. Of course the various audio compression and encapsulation formats would need to integrate this format into the music files. For this prototype the XML data will be stored separately, but ultimately its design should make it possible to easily incorporate the metadata within the same file as the audio, and even to make the lyrics streamable.

The decision to make this format XML-based is due to the need for this format to be human readable. Evidently, when comes the need to incorporate the lyrics data into an audio file the XML could be compressed. This matter will not be developed in this prototype, as XML compression and streaming is a well-explored field of research and many solutions already exist in order to achieve this. Additionally, thanks to XML schemas, XML provides great tools in order to create XML-based formats. And MPEG-7, which is also based on those technologies, introduces good insights regarding how timing is managed in describing the contents of media, which will be adapted to the purpose of this project.

The prototype will be highly focused on defining the format, named LML, which stands for Lyrics Markup Language. Getting it outlined well is crucial to its success, which is the reason why the authoring and playback tool prototype will be secondary in the development of this project.

4.1. FORMAT DESIGN

The choice of using XML as a format for LML seemed evident since all the tools necessary to define a timeline of lyrics for a song can be text-based. XML schemas provide all that is required in order to constrain LML files so that they are all compatible and use the same syntax. XML schema also offers simple types that could be exploited directly in LML files.

LML files mainly consist of a text, which includes all the lyrics of the songs, with

attributes for every word and if needed every syllabus describing when does the word start and ends to be sung in the timeline.

In order to test the format a set of four songs will be used. The feature of being able to mix various languages will be made possible by using the UTF-8 charset, which could let for example mix English and Japanese lyrics in the same song (something quite common in Japanese pop music). This is a crucial feature to develop, as a lyrics display feature on a portable music player is likely to be more successful in countries that already have an existing culture of karaoke.

In addition to having words in various languages linked by attributes to specific sequences in the timeline, it would be possible to define the name of the interpreter singing a chunk of text or a specific word. This information could be displayed as well, an interesting feature in bands that have multiple singers that music fans would probably appreciate, as this information is often lacking in song lyrics, even in album leaflets.

The great advantage of using XML Schema is that it leaves the definition of the format completely open to future additions and revisions. For example mood properties could be added to sequences of text so that the display of the text on the playback device adapts depending on which mood is being evoked by a specific sequence of the song. It could be as simple as darker colors and slower animations for the text when the lyrics are tagged as “sad” in the LML file. This way the motion of the text would be less monotonous and convey more meaning. Also specific color or movement attributes could be added to the text. Those are just examples of what extensions the format could benefit from, but will not necessarily be part of this prototype design.

Before creating the single or multiple XML schemas that define the LML format, it will be necessary to design example instances of LML files. By creating the files manually, ignoring for now the format constraints defined in the future XML schemas, it will be easier to see which tags and which format work best in most situations. The methodology will then be to deduct the underlying syntax rules from the examples created manually.

4.1.1. CREATING THE REFERENCE LML FILES

The methodology used to create the first LML files is very simple. By playing the songs in GarageBand and inserting markers at the beginning and end of every word, precise times have been extracted for the song lyrics. In order to start with a first draft of what a LML file should be like, only the simple lyrics information will be taken into account. The format will be revised after this first draft in order to integrate more advanced features, such as song intensity.

```
<?xml version="1.0" encoding="utf-8"?>

<Song xmlns="http://www.kouiskas.com"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.kouiskas.com lml-0.1.xsd">
```

Let us have a look at the header of a sample LML file:

The encoding of the file, utf-8, is important, even if not used at this stage. Because it is exactly what will help use different languages based on various character sets in the same file. Something that would not be possible with an ISO character encoding.

Then there is the root tag of every LML file, the Song tag. Inside which are parameters pointing to the XML schema file defining the LML format. It is very useful for the draft schema to be online so that the W3C validator can check both the schema and XML files using it. This is a very powerful tool that will make it much easier to test whether the newly created format is valid or not.

Inside the Song root tag of the LML file resides a sequence of Sentences tag, each of which holds a series of Word tags. This very straightforward description of the song also does not need to necessarily be in order, it will be the responsibility of the application interpreting the file to find where the relevant word timing information is described within the file.

```
<Sentence>

<Word startTime="23.031" endTime="23.500">for</Word>

<Word startTime="15.594" endTime="16">What</Word>
<Word startTime="16" endTime="18.156">if</Word>
<Word startTime="18.469" endTime="19.031">there</Word>
<Word startTime="19.031" endTime="21.156">is</Word>
<Word startTime="21.594" endTime="22.375">nothing</Word>
<Word startTime="22.375" endTime="23.031">else</Word>

<Word startTime="23.500" endTime="24">us</Word>
<Word startTime="24" endTime="24.969">after</Word>
<Word startTime="24.969" endTime="25.719">all</Word>
<Word startTime="25.719" endTime="26.916">this</Word>

</Sentence>
```

The description of start and end times is straightforward, as attributes of each word tag. It appears that each word should have a start and end time, something that will be implemented in the XML schema defining the LML format. The first drawback of defining start and end time as attributes is that if the same sentence repeats itself through the song, each word will have to be defined again. This can be relatively space-consuming within the LML file.

While making each word able to represent different occurrences would save space within the LML file, it would greatly impact on the human-readability. This is the reason why the possibility of a single word representing different occurrences will not be implemented. Furthermore, it is very likely that if the LML data comes to be included in sound files or streamed, it will be compressed somehow. And having textual redundancy inside the LML file by defining the same word twice will result in great compression ratios. Even if the LML file appears more verbose, the amount of actual file storage space lost will not be significant once compressed.

It is also interesting to notice in the example above that within a sentence the words do not have to be defined in chronological order. Similarly, the sentences within the song do not have to be defined in order. It will be effortless for an application using the LML information to sort the extracted data.

4.1.2. CREATING A FIRST DRAFT OF THE XML SCHEMA

Now that creating the sample LML files made an internal structure emerge, this structure can be turned into a set of rules that will define the LML format and how a LML file should be organized.

Let us have a look at the XML schema designed from the sample LML data and study how it works:

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.kouiskas.com"
xmlns="http://www.kouiskas.com" elementFormDefault="qualified">

<xs:element name="Song" minOccurs="1" maxOccurs="1">
 <xs:complexType>
  <xs:sequence>

   <xs:element name="Sentence" minOccurs="1" maxOccurs="unbounded">
    <xs:complexType>
     <xs:sequence>

      <xs:element name="Word" minOccurs="1" maxOccurs="unbounded">
       <xs:complexType>
        <xs:simpleContent>
         <xs:extension base="xs:string">
          <xs:attribute name="startTime" type="xs:float" use="required"/>
          <xs:attribute name="endTime" type="xs:float" use="required"/>
         </xs:extension>
        </xs:simpleContent>
       </xs:complexType>
      </xs:element>

     </xs:sequence>
    </xs:complexType>
   </xs:element>
```

The parameters of the schema itself define which namespace the schema will be part of. As it has been established before, the schema will be available online so that the W3C XML schema validation tool ([16] http://www.w3.org/2001/03/webdata/xsv) can be used.

The structure of the schema reflects directly the LML file. A root node, Song, is defined as a complexType, made of a single sequence of nodes. These are Sentence nodes, and they have been defined with constraints stating that there should be at least one of them. To clarify, each LML file must have one Song node, and only one. This main Song node must have at least one Sentence node.

As for the Sentence nodes, they are also sequences of Word nodes, and each sentence must have at least one word. The hierarchy song->sentences->words is clear and must be respected to conform to the LML format. The choice of dividing sequences of words into sentences has been made to help divide the display of the lyrics. Having a constant stream of words would probably impact on readability, which is why the Sentence nodes help separate logical series of words. That does not mean that Sentence nodes have to be real sentences in the grammatical sense of it, it could be verses, or any grouping of words. One might decide to ignore the possibility to use sentences in a particular song and define a sigle large sentence containing all the words within the song. Users of the LML format are still free in how they use it, even if the structure is there to give a guideline on how it should be used.

Finally, after defining the Song and Sentence types of nodes, which give its structure to the file, the real data is defined within the Word nodes.

```
<xs:element name="Word" minOccurs="1" maxOccurs="unbounded">
  <xs:complexType>
    <xs:simpleContent>
      <xs:extension base="xs:string">
        <xs:attribute name="startTime" type="xs:float" use="required"/>
        <xs:attribute name="endTime" type="xs:float" use="required"/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>
```

The Word nodes are an extension of the basic type “xs:string” provided by the W3C. Word nodes are an extended type, because in addition to holding a string of characters, they also have to contain two attributes, startTime and endTime.

Given that most song files have a length counted in minutes and not hours, the precision of the float type, representing the amount of seconds since the beginning of the song, seemed more than enough to store the time information of the markers indicating the beginning and the end of a single word in the song’s timeline. startTime simply indicates the exact moment in the song when a word starts being sung, and endTime contains similar information about when the word ends being sung.

In further discussion, the possibility of extending the way time is defined in the LML format to let users have access to other formats of data to define time will be explored. The first draft of the LML format's XML schema aims at covering the core functionalities, thus having only one syntax for the time based on floating point numbers will be enough in order to develop a LML player prototype.

4.2. LML PLAYER PROTOTYPE

Now that a working draft has been established for the XML schema and that the LML files can thus be validated online, it is necessary to test the format in its main application. By developing the prototype of a music player that will be able to display the lyrics as they are being sung, thanks to the data present in an LML file associated with the song file, it will make it much easier to assess if as a format LML has been well designed. And see if it works for the kind of application it was designed for.

This project being developed on a Mac platform, the tools provided by Apple to create software, mainly XCode, made Cocoa and Objective-C a good programming platform for the rapid development of a prototype.

The first thing needed for the prototype was the ability to play audio files. Being a very common need and with core functionalities covered by Quicktime, it was straightforward to find sample source code letting a Cocoa program play MP3 files. The source code found, [17] Borkware, QTMP3, http://www.borkware.com/rants/sound/, provided all the playback functionality required, under BSD licensing which made it a perfect choice for this project. Of course this only is far from covering the objective of the prototype, but saved a lot of time in not reinventing the wheel concerning MP3 playback.

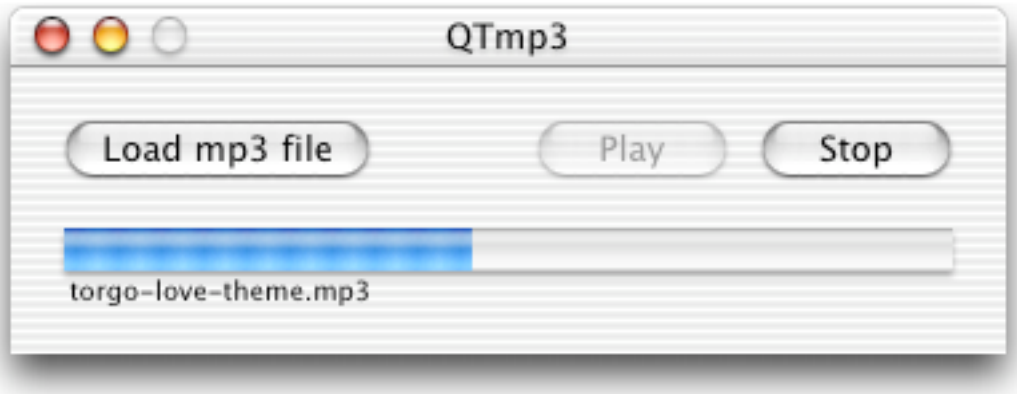

Figure 3. The original QTmp3 sample program. Source: www.borkware.com

Starting from there, the application needed the added support of reading LML files to display the lyrics synchronized to the playback of the song. The very object-oriented Cocoa framework provided different options for XML data manipulation.

Out of the various official or third-party Cocoa toolkits, all of them fall into two categories. Event-based and tree-based toolkits. As the data in the LML file

most likely needs to be read once when the file is opened and then stored into the program's memory, the event-based approach made more sense for this prototype. The same result could have been achieved with a tree-based toolkit but would probably have required more programming. The default choice for event-driven XML parsing in Cocoa is the NSXMLParser ([18] Apple, "Event-Driven XML programming guide for Cocoa", http://developer.apple.com/documentation/Cocoa/Conceptual/XMLParsing/index.html), designed by Apple and used by most applications made by Apple relying on XML, including OS X itself.

The only drawback of that toolkit is that it supports validation only with DTDs and not XSD XML Schemas. For the scope of this study this can be overlooked as the files can be validated using the online W3C tool. In the context of an online music shop this could also be achieved at the shop level, saving the MP3 portable player on the user's end to spend processing time on validating the LML files. Mostly because compared to the parsing, the validation process would use too much processing power, thus battery, to perform the task of playing back LML data on a portable player.

4.2.1. PARSING THE LML DATA

The way the event-based XML parser works is that it calls functions when specific events happened when reading through the XML file. By instantiating those functions or not, it is made possible to catch the information of the event as it is triggered.

In the case of this prototype Sentence tags are ignored, as the display is basic and displays the sung words one by one. Which makes the main focus of the XML event functions the Word tags.

```
- (void)parser:(NSXMLParser *)parser didStartElement:(NSString *)elementName
namespaceURI:(NSString *)namespaceURI qualifiedName:(NSString *)qName
attributes:(NSDictionary *)attributeDict {

     if ( [elementName isEqualToString:@"Word"]) {

          currentStartTime = [[attributeDict objectForKey:@"startTime"] floatValue];
          currentEndTime = [[attributeDict objectForKey:@"endTime"] floatValue];

          [currentStringValue setString: @""];

          return;
     }

 }
```

The didStartElement event is triggered when the opening tag is detected. The function checks that the tag encountered is of the Word kind and looks for the startTime and endTime attributes. It should be pointed out that at this point the LML file is considered valid, which means that by definition complying with the LML XML schema, every Word tag must contain the startTime and endTime attributes. The current string is also set to zero, so that characters detected from now on are added to the currentStringValue, which should contain the text data in between opening and closing Word tags by the time the end tag event is triggered.

```
- (void)parser:(NSXMLParser *)parser foundCharacters:(NSString *)string {

    if (!currentStringValue) {

        currentStringValue = [[NSMutableString alloc] initWithCapacity:50];

    }

    [currentStringValue appendString:string];


}
```

As stated above, the foundCharacters event happens when characters that are not tags are detected. By appending found characters to the currentStringValue, the code makes sure that everything appearing between opening and closing Word tags is captured.

The first part of the didEndElement function below deals with the event triggered when the end Word tag is reached. At this point the information gathered about the <Word></Word> tag pair is gathered and stored into a dictionary. This contains the startTime, endTime attributes as well as the string found between the start and end tag.

```
- (void)parser:(NSXMLParser *)parser didEndElement:(NSString *)elementName
namespaceURI:(NSString *)namespaceURI qualifiedName:(NSString *)qName {

     if ( [elementName isEqualToString:@"Word"]) {

          if (!songLMLData) {

               [songLMLData release];
               songLMLData = [[NSMutableArray alloc] init];

          }

          [songLMLData addObject:[NSDictionary dictionaryWithObjectsAndKeys:
[NSNumber numberWithFloat:currentStartTime], @"startTime", [NSNumber
numberWithFloat:currentEndTime], @"endTime", [NSMutableString stringWithCString:
[currentStringValue cString]], @"word", nil]];


          return;
     }

     if ( [elementName isEqualToString:@"Song"]) {

          // End of the file


          NSSortDescriptor *startDescriptor = [[[NSSortDescriptor alloc]
initWithKey:@"startTime" ascending:YES] autorelease];

          NSArray *descriptors = [NSArray arrayWithObjects:startDescriptor, nil];

          [sortedLMLData release];
          sortedLMLData = [[NSArray alloc] init];

          sortedLMLData = [[songLMLData sortedArrayUsingDescriptors:descriptors]
retain];

     }

}
```

The second part of the function happens when the Song end tag is reached, which should effectively be the very last tag of every LML file. At this stage all the Word tags should have been treated and are now stored in a dictionary variable. As mentioned before, a LML file cannot be forced by the XML schema to define the Word information in chronological order. Which is the reason why in the function above the dictionary generated during the parsing is sorted and the sorted copy stored in a different variable for later use. This way the new array containing the Word data of the LML file is in ascending order of startTime values, so that when the timeline is at a particular point, the next word chronologically is simply the next one in the array.

Figure 4. The prototype, displaying lyrics as they are sung in real time.

4.2.2. DISPLAYING THE LYRICS

The lyrics display in the prototype simply consists in a text field displaying the current sung word, automatically synchronized to the playback of the song. In order to achieve this, an array containing the words and their timing information has been generated in the XML parsing described above.

From that point, the algorithm of the words display is relatively straightforward. Next page is the timer function that handles the playback. The current playback time information is provided by the quicktime object representing the song. According to this value the appropriate word is displayed. The crucial point is how to handle words that have a common time point (the end point of the first one being the start point of the second one).

```
- (void) pollMovie: (NSTimer *) timer
{
    if (IsMovieDone([qtmovie QTMovie])) {
     [stopButton setEnabled: NO];
     [playButton setEnabled: YES];
     [timer invalidate];
     timer = nil;
    }

     currentPlayTime = GetMovieTime ([qtmovie QTMovie], NULL) / 600.0 + 0.2;

    [progressIndicator
     setDoubleValue: GetMovieTime ([qtmovie QTMovie], NULL)];

     if (sortedLMLData != nil) {

          NSDictionary *dict;

         if (currentIndex < [sortedLMLData count])
          {
               dict = [sortedLMLData objectAtIndex:currentIndex];

               if (currentWordDisplay && [[dict valueForKey: @"endTime"] floatValue]
< currentPlayTime)
               {
                    currentWordDisplay = FALSE;
                    currentIndex++;

                    if (currentIndex < [sortedLMLData count])
                         dict = [sortedLMLData objectAtIndex:currentIndex];
               }

               if (currentIndex < [sortedLMLData count] && [[dict valueForKey:
@"startTime"] floatValue] <= currentPlayTime)
               {
                    currentWordDisplay = TRUE;
               }

               if (currentWordDisplay)
               {
                    [lyricsField setStringValue:[dict valueForKey: @"word"]];
                    [lyricsDisplay setLyric:[dict valueForKey: @"word"]];
               }
               else
               {
                    [lyricsField setStringValue:@""];
                    [lyricsDisplay setLyric:@""];
               }
          }
     }

} // pollMovie
```

4.3. IMPROVING THE LML FORMAT

As the first draft of the LML format has proved to work successfully with the LML player prototype, the necessity to add more features to it appeared, since there are many ways through which the functionality provided by the LML player could be made more attractive.

The first idea that emerged earlier in the design was to describe moods present in the song. The main issue that appeared was that the classifications and taxonomies of song mood themselves are already widely considered subjective ([19] D. Liu, L. Lu and H.J. Zhang, "Automatic mood detection from acoustic music data"). All the proposed taxonomies were dissatisfying because they did not look objective enough and usually tried to have too many variations on the same moods, giving so much choice that determining which mood was best to describe a specific part of a song would be made almost impossible.

Even if the perfect taxonomy were available for song mood description, it would be difficult to guarantee the objectivity of the manual or automatic tagging, as the same song can be interpreted differently by various listeners. Sometimes it is even perceived differently by the same listener when the song is heard on different occasions.

From those findings, an alternative take on music classification based on intensity ([20] V. Sandvold and P. Herrera, "Towards a semantic descriptor of subjective intensity in music") seemed to provide better objectivity in the classification and was also much closer to the feature searched for in the case of the prototype, being to adjust the display of the lyrics according to the energy within the song. Below is the music intensity classification taken from Sandvold and Herrera's research on which the revision of the LML format will be based.

Wild: Marked by extreme lack of restraint or control; intensely vivid. Synonyms: *intense, manic, fiery*.

Energetic: Possessing or exerting or displaying energy. Synonyms: *lively, sparkling, raucous/rowdy, exciting*.

Moderate: Being within reasonable or average limits; not excessive or extreme. Synonyms: *laid-back/mellow*.

Soft: Having or showing a kindly or tender nature. Synonyms: *gentle, soothing, calm/peaceful*.

Ethereal: Characterized by lightness and insubstantiality; as impalpable or intangible as air. Synonyms: *detached, hypnotic, unreal*.

From this, the application to the LML format should be straightforward, by creating an Intensity tag that can take any of the 5 above values. Groups of sentences, single sentences, groups of words and single words can be within Intensity tags in order to offer better flexibility on how the new tag can be used.

The implementation of the Intensity tag actually demanded a much deeper change in the XML schema defining the LML format. Mainly because the Intensity tag introduces an optional tag hierarchy change, as it is considered that a LML file can be defined without the use of the Intensity tags. Let's examine the second revision of the LML XML schema to study what changed with the introduction of the Intensity tag.

```
<xs:simpleType name="IntensityValue">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Wild"/>
    <xs:enumeration value="Energetic"/>
    <xs:enumeration value="Moderate"/>
    <xs:enumeration value="Soft"/>
    <xs:enumeration value="Ethereal"/>
  </xs:restriction>
</xs:simpleType>
```

The first addition, above, is the definition of the type IntensityValue, that defines which values the “value” attribute of the Intensity tag can take. Only the five options can be used, which does not let users define their own intensity value descriptions. This choice has been made in order to maintain a standard, as the words used for intensity description can quickly become very subjective, which would defeat the purpose of defining the standard.

```
<xs:complexType name="WordType">
  <xs:simpleContent>
    <xs:extension base="xs:string">
      <xs:attribute name="startTime" type="xs:float" use="required"/>
      <xs:attribute name="endTime" type="xs:float" use="required"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
```

In order to prevent redundant information, the Word tag had to be defined as an xml type.

```
<xs:complexType name="IntensityType">
  <xs:sequence>
    <xs:element name="Word" type="WordType" minOccurs="1"
maxOccurs="unbounded" />
  </xs:sequence>
  <xs:attribute name="value" type="IntensityValue" use="required"/>
</xs:complexType>
```

Similarly, the intensity had to be defined as a type since it is used twice in the rest of the XML schema definition. It should be noted that the IntensityType is forced to include a sequence of Word tags. The IntensityType has a single attribute, whose type has already been defined above.

Now that the Intensity tag type has been defined, Sentence tags should be able to contain only a sequence of Word tags, or a mixture of Word and Intensity tags (each Intensity tag compelled to having at least one Word tag within itself). The main constraint that should not be left is that the Sentence tag must contain at least one element, be it a Word or an Intensity tag containing a series of words.

```
<xs:complexType name="SentenceType"  mixed="false">
  <xs:choice>
  <xs:sequence minOccurs="1" maxOccurs="unbounded">
    <xs:element name="Word" type="WordType" minOccurs="1" />
    <xs:element name="Intensity" type="IntensityType" minOccurs="0" />
  </xs:sequence>
  <xs:sequence minOccurs="1" maxOccurs="unbounded">
    <xs:element name="Intensity" type="IntensityType" minOccurs="1" />
    <xs:element name="Word" type="WordType" minOccurs="0" />
</xs:sequence>
  </xs:choice>
</xs:complexType>
```

The way this has been achieved is that an element of type SentenceType has to either be a sequence containing at least one Word tag or a sequence containing at least one Intensity tag.

```
<xs:complexType name="SentenceIntensityType">
  <xs:sequence>
    <xs:element name="Sentence" type="SentenceType" minOccurs="1"
maxOccurs="unbounded" />
  </xs:sequence>
  <xs:attribute name="value" type="IntensityValue" use="required"/>
</xs:complexType>
```

In order to make possible the fact to use a couple of Intensity tags around a group of sentences, the SentenceIntensityType had to be defined. It supports the fact that such an Intensity tag should be compelled to contain at least one Sentence. Effectively, Intensity tags that contain Word tags are a different XML type than the ones containing Sentence tags.

Finally, now that all the sub-types have been defined, the main structure of the LML file is simply a sequence of optional Intensity and Sequence tags.

```
<xs:element name="Song">
  <xs:complexType>
    <xs:sequence minOccurs="1" maxOccurs="unbounded">
      <xs:element name="Sentence" type="SentenceType" minOccurs="0"
maxOccurs="unbounded" />
      <xs:element name="Intensity" type="SentenceIntensityType" minOccurs="0"
maxOccurs="unbounded" />
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

5. EVALUATION

The LML format being more clearly defined after a few drafts, and a working prototype demonstrating the potential outcomes of the LML format being finished, it is now necessary to determine to what extent this project has been successful.

One of the possible stumbling blocks of deploying such a technology would be the automation of the LML files generation process. By simply testing existing voice recognition software packages, it will be made clear to what extent existing technologies can be used in order to achieve this.

The strong belief that came out of the initial research was that lyrics metadata directly integrated into audio file formats was a relatively simple technology that did not exist despite the fact that it could make a difference for commercial solutions by attracting customers. This assumption needed to be verified, and to achieve this a simple survey was designed, aimed at digital music consumers.

5.1. DIGITAL MUSIC SURVEY

The aim of the survey (Appendix 2) was to determine to what extent users would be interested in lyrics-related features on their computers or portable digital music players. It was also to see if such features could influence their choice when deciding which online digital music retailer to use.

The demographics are difficult to determine as the survey was advertised on anonymous Internet forums, but the largest amount of respondents were fellow Napier students. The nature of the survey being Internet-based it also changes the nature of the respondents, but in this case it is useful as the main focus is consumers of online digital music, who need to be internet users in the first place for obvious reasons.

The first series of questions was to determine what kind of online music consumers the respondents were. That is, to know if they mostly download music free of charge (illegally or not), and if they pay for downloadable music. A wide range of consuming patterns could be found, from people who almost do not consume online music (overall average of 2 songs downloaded per month), to very big music consumer (over 500 songs downloaded per month).

This initial profiling of users was followed by the questions drawing outcomes regarding this study. The feature developed in the software prototype of the project is a real-time display of song lyrics, which seemed to be the main potential development that could be achieved with the LML file format. In order to see if users would be interested in this feature, they have been asked to rate to what extent the feature would influence their choice for an online music retailer solution. To make that rating more meaningful, respondents also had to rate other features for such a service.

112 people have responded to this survey and interesting patterns have emerged, which will now be analyzed.

5.1.1. SURVEY RESULTS

The consumers most relevant to this study, users who buy downloadable music from online shops such as iTunes, represented a fair amount of the final pool of respondents. The results of the most meaningful questions for this study will be looked at by seeing the differences between users who have never bought music online, users who have and finally, more specifically, users who have and do it on a regular basis.

Question 9 of the survey explored how relevant lyrics-based features would be in the choice of a commercial digital music platform. Below are the overall results among the 112 respondents.

9. If you were to compare the available solutions for online music purchase, combined with portable audio players (such as iTunes+iPod), please assess how important the following features would be in making your choice. 1 representing a feature irrelevant to your choice and 5 a feature directly influencing your decision.

Overall sound quality (4.1)
Real-time display of song lyrics (2.3)
Wireless transfer of songs between computer and audio player (3.0)
Customizable audio player interface (skins, wallpapers, animations, etc) (2.3)

It is interesting to see that regardless of the consumer profile, real-time display of song lyrics, as developed in this project's prototype, scored as much as customizable interface. Both features are largely unexplored with digital music portable players, but customization has proved to be a big selling point and source of many new businesses in the mobile phone market. Given that more and more brands attempt to merge mobile phones and portable music players, the above score for real-time lyrics display seems to be truly relevant to the selling potential of such a feature.

Looking more closely at the results of this question depending on the consumer profile and how the respondents consume digital music or not draws further interesting results.

Results of question 9 among respondents who have bought downloadable music from an online shop similar to iTunes at least once:

Overall sound quality (4.04)
Real-time display of song lyrics (2.19)
Wireless transfer of songs between computer and audio player (2.89)
Customizable audio player interface (skins, wallpapers, animations, etc) (1.96)

At this point the tendency towards the lyrics function compared to customization is more accentuated. This could lead to think that even a single experience of existing commercial digital music platforms made the need for an access to lyrics more important.

Results of question 9 among respondents who have bought downloadable music from an online shop similar to iTunes in the past month:

Overall sound quality	*(3.56)*
Real-time display of song lyrics	*(2.36)*
Wireless transfer of songs between computer and audio player	*(2.64)*
Customizable audio player interface (skins, wallpapers, animations, etc)	*(1.8)*

The most relevant users to the study seem to have a very different opinion on that specific topic than the general pool of respondents. Not only does lyrics display become surely more wanted than customization, but it is almost closing the gap with wireless transfer.

Finally examining how respondents who have never used an online downloadable music store answered these questions:

Results of question 9 among respondents who have never bought downloadable music:

Overall sound quality	*(4.18)*
Real-time display of song lyrics	*(2.37)*
Wireless transfer of songs between computer and audio player	*(3.12)*
Customizable audio player interface (skins, wallpapers, animations, etc)	*(2.54)*

Interestingly, lyrics display ranks the lowest in this case, very much to the contrary of users who have experienced online music shops. It's still close to customizable interface, which indicates that overall the two features are comparable, the gap being only clear for users who buy downloadable music regularly.

The next question asked in this survey from which results seem to emerge, 12, asked users how often they would use an automatic access to song lyrics if they had access to it. This did not specify the nature of the display of the lyrics, contrary to further questions. Below are the overall results among the 112 respondents.

12. If you had the possibility to access the song lyrics automatically from your computer or portable audio player, how often would you use it?

I would put it on all the time	*6.3%*
I would use it often	*25.9%*
I would use it from time to time for specific songs	*60.7%*
I would never use it	*7.1%*

The most dominant use is clearly casual, and the most interesting result here is that overall only a small amount of respondents declared that they would not use the lyrics access at all, as well as the fact that almost a third of the respondents would make a heavy use of such a feature. This indicates that regardless of the fact that lyrics-related feature would be relevant to the choice of a digital music commercial platform, consumers believe that they would use this feature which confirms its popularity. The underlying consumer groups studied before do not show any relevant variations in the results of question 12.

Finally the last question, 13, asked about where a more specific feature, real-time display of the lyrics, would be used. Below are the overall results among the 112 respondents:

13. If you were given the option to display song lyrics in real time automatically (displayed as they are sung), where would you rather use this feature?

On my computer	*27.7%*
On my portable audio player	*6.3%*
On both	*34.8%*
I wouldn't use it	*31.3%*

The results show that real-time display of the lyrics is a less popular feature than just accessing the lyrics. Nevertheless a big majority of users would still use the feature, with a tendency to use it more on the computer than on the portable audio player.

Results of question 13 among respondents who have bought downloadable music from an online shop similar to iTunes at least once:

On my computer	*17%*
On my portable audio player	*10.6%*
On both	*32%*
I wouldn't use it	*40.4%*

Surprisingly the real-time lyrics display is even less popular among users who have had some experience of online downloadable music shops. The tendency towards computer use is also greatly reduced.

Results of question 13 among respondents who have bought downloadable music from an online shop similar to iTunes in the past month:

On my computer	*16%*
On my portable audio player	*20%*
On both	*20%*
I wouldn't use it	*44%*

Again, the core of regular online shops users is the least interest in that feature, with only slightly more than half of them declaring that they would use the feature. Half of the users is probably still highly relevant. It should also be noted that the shift of use seems to be towards portable players now.

5.1.2. SURVEY CONCLUSIONS

This survey seems to have confirmed the assumption on which the prototype of this project was based. Even among users who have never bought music online or do not own a portable music player, lyrics-related features attract more than half of the consumer groups, something up to 70%. It also appears that among regular online music shops users, these features would make more of a difference when compared to customization of the audio player. In a nutshell, in the emerging competition of downloadable music retail industry, very similar offers looking to make a difference could very well use song lyrics features as a selling point. This justifies the relevance of this study from a user needs and commercial point of view.

5.2. EVALUATING THE LML FILE FORMAT

The main prototype of this project was the design of the LML format itself. As the LML player prototype demonstrates, the file format works. The main unknown regarding the format is how well it would work with thousands of different songs, something that was impossible to achieve in the scope of this project, given that the LML files needed to be created manually.

The validation process provided by the W3C is a real help in determining whether LML files are correct or not, but having a closer look at what could be put into LML files, this validation would be insufficient. The reason behind this is that even if the syntax of an LML file can be valid XML, its information can still be wrong, as in the following examples:

```
<Sentence>

<Word startTime="0.0" endTime="0.0">for</Word>

<Word startTime="0.0" endTime="0.0">What</Word>
<Word startTime="0.0" endTime="0.0">if</Word>
<Word startTime="0.0" endTime="0.0">there</Word>

</Sentence>

<Sentence>
<Word startTime="4.0" endTime="7.0">is</Word>
<Word startTime="3.0" endTime="5.0">nothing</Word>
<Word startTime="1.0" endTime="8.0">else</Word>

</Sentence>
```

The only way to circumvent this problem would be to create a logical validation tool in addition to the syntax validation offered by the W3C. It would be an LML-specific validation tool that would search for overlapping word time information and zero-length words in order to determine if an LML-file is valid. This issue also shows the limitation of the XML schema language, which does not provide enough tools in order to achieve this.

Making such a validation tool should be relatively straightforward, as it mostly consists in a simple algorithm calculating if word timings overlap in the file or not.

5.3. FEASABILITY OF LYRICS RECOGNITION

The crucial aspect in the deployment of LML is that content providers would need to generate the LML files. While right holders of the songs probably have access to digital copies of the song lyrics, those do not contain the timing information needed to generate a LML file. This is the reason why speech recognition technologies could be used for that purpose.

The main problem is that most commercial speech recognition solutions are not designed to recognize singing, but talking. And the music itself could be too loud and cover the voice to the speech recognition software. In order to see how well or bad existing speech recognition software would behave with song lyrics recognition a test has been conducted with a popular mac tool ([28] MacSpeech Incorporated, "iListen speech recognition software"). This commercial product is mostly aimed at transcription of audio files, contrary to other speech recognition packages which focus on speech recorded live from a microphone. This is the main reason why iListen was selected, as working on an audio file is closer to what a song lyrics recognition tool should do.

The same selection of four songs used for the LML player prototype was used with iListen in order to determine its accuracy.

The quality of the results was extremely low, given that not a single word was recognized correctly, in addition to which the words found by the speech recognition did not sound anything like the real ones. This was probably due to the inability of the software to make the difference between the voice and the music and/or find where words start and end. While the reasons behind this poor performance are difficult to determine accurately, this clearly demonstrates that existing speech recognition software packages will probably be of no help to transcribing the lyrics of a song. Most of those systems rely on mechanics such as voice calibration and learning, which would be impossible to apply to songs. A part of the solution would be automated signal separation to isolate the voice from the signal before feeding it to the speech recognition software. There have been efforts towards automatic separation of signals ([29] M. A. Casey and A. Westner, "Separation of mixed audio sources by independent subspace analysis", [30] GJ. Jang and TW. Lee, "A probabilistic approach to single channel blind signal separation"), but the practical results are unclear, as those mathematics-oriented solutions to the problem do not show practical uses of the developed techniques, and especially do not mention the case of music.

What probably needs to be found is an efficient way to isolate the sound of sung voice in most types of music, which in itself is a big challenge. The voice recognition itself could probably do with existing technologies, given that it is more a matter of mapping existing lyrics to the timeline rather than transcribing the lyrics. It is assumed that the right holders will have access to the lyrics transcriptions already, but it will be required to determine the timing of each word thanks to this new kind of speech recognition.

6. CONCLUSION AND SUGGESTIONS FOR FUTURE WORK

The issue of expensive access to a standard's specification came as a surprise, and unfortunately quite late in the development of the original project. This has proved the assumption that the access to computing standards is usually easy very wrong. Nevertheless, studying MPEG-7 in-depth helped get a better idea of what the future of multimedia metadata will be, whether the standard itself succeeds or not.

After a study of the metadata associated with video and audio, lyrics had been identified as a potential area where a new metadata format could be defined. The survey conducted during this study confirmed that there is commercial potential for lyrics-associated features. This is something that should be considered by media companies and manufacturers or audio playback equipment.

The LML format itself has proved to work well, but it would need further testing and a development more open to external help in order to succeed as a standard. As it is now, it could help attract potential key actors to participate, thanks to the prototype and the validation mechanisms developed with the XML schema. Other functionalities and tags could effortlessly be added to the format, but would most likely require studying users more closely and what they would expect or prefer from the display of lyrics.

Developing the player prototype further in order to make it a proper application or a plugin for popular audio playback software could also help raise awareness of the format.

Possible future work in order to help the LML format succeed would focus on combining existing research regarding audio signal separation and voice recognition in order to automatically generate the lyrics. Given that generally the lyrics for a specific song are already available in digitized format, it would be more a matter of matching every single word to the timeline of the song. Similarly, low-level audio analysis could determine the intensity data for the LML files.

APPENDIX 1: INTERVIEW WITH FRAMELINE 47'S SPOSKESMAN

First, given that this is the only MPEG-7-supporting commercial product, if I was to use it, would that mean that I, as a student, or as a potential company buying your product, would need to develop my own tools to make use of the MPEG-7 annotated file generated by frameline47?

Essentially yes - you would need to build your own tools that would be able to make use of the MPEG-7 XML.

Do you think that this issue of very little amount of commercial products that can be interacted with is something that prevents your product from selling more?

Not really a problem, as our XML can be parsed easily enough to other forms of XML - for example our 1.1 release enable export to Final Cut Pro XML, and the code only took a few days to write. We are also working with a digital archive company in the US (http://www.ptfs.com) on integrating our MPEG-7 XML with their search engine, and all we need to do is translate from one XML to another.

Essentially we chose MPEG-7 to represent our data, as it saved us lots of work as it already had a way of describing segments and groups of segments over time. Also to be honest MPEG has been very succesfull, and associating ourselves with their standards is good PR for us.

My original honours project was to develop an MPEG-7 library, but given that the cost of the standard's specification is over £1400 my university refused to acquire it. What made your company bet on MPEG-7 and purchase this costly specification given that MPEG-7 has not proved yet to be a well designed (since you have developed the first commercial tools, years after the specification was made)?

We didn't by the entire spec, just a book from Amazon.
MPEG-7 has some VERY esoteric stuff - audio & pattern recognition, but we don't (yet) use any of this.

Similarly, have you developed an MPEG-7 library internally and do you intend to sell it to potential developers?

We developed our ontology first and then looked for an easy way to code it. Our ontology is open for anyone who asks to use.

This last question comes from the fact that there is currently no MPEG-7 library for developers that I know of, and I believe that it's one of the reasons why so little MPEG-7 based application are available.

The trouble is that the MPEG-7 spec is SO big.

Is there a current process to validate your application as officially generating MPEG-7-compliant data?

Yes there is a web site that offers MPEG-7 validation - but can't remember the URL. Will ask our Technical Director & get back to you if you like ?

Do you know what your customers use your software for, and especially…

Mainly production & archiving, but also sports notation, dance notation - speech notation - pretty much anything over time really.

…what proprietary or other commercial tools they use the annotated videos with?

Amazingly, to our knowledge there are no other video tagging tools - love to hear otherwise!

How do you envision the future of digital video and the impact that detailed annotation will have on it?

Big. We call the paradigm ' describe to distribute' i.e., if media isn't described you cannot distribute it. All media will be delivered by IP in the next ten years, so this will become an increasingly important set of technologies.

APPENDIX 2: SURVEY

Below are the questions asked in order for the survey conducted in November 2006. The full results are available in CSV format on the honours project CD.

1. Do you own a portable digital audio player (iPod, MP3 player, etc)?
Yes
No

2. Have you ever downloaded music on the internet (websites, P2P, FTP, etc)?
Yes
No

3. Have you downloaded music on the internet in the past month?
Yes
No

4. Please give a rough estimate of how many songs you download per month on average:
Numeric value had to be entered

5. Have you ever purchased downloadable music on the internet (on a shop like iTunes or similar)?
Yes
No

6. Have you purchased downloadable music on the internet in the past month?
Yes
No

7. Please give a rough estimate of how many downloadable songs you purchase per month on average:
Numeric value had to be entered

8. From your downloaded or purchased music library, how much is already transfered to your audio player?
All of it
More than half of it
Half of it or less
None

9. If you were to compare the available solutions for online music purchase, combined with portable audio players (such as iTunes+iPod), please assess how important the following features would be in making your choice. 1 representing a feature irrelevant to your choice and 5 a feature directly influencing your decision.

Overall sound quality
Real-time display of song lyrics (lyrics appear on screen while they can be heard)
Wireless transfer of songs between computer and audio player
Customizable audio player interface (skins, wallpapers, animations, etc)

10. Have you ever searched for song lyrics on the web?
Yes
No

11. Have you ever searched for song lyrics on the web and not found what you were looking for?
Yes
No

12. If you had the possibility to access the song lyrics automatically from your computer or portable audio player, how often would you use it?
I would put it on all the time
I would use it often
I would use it from time to time for specific songs
I would never use it

13. If you were given the option to display song lyrics in real time automatically (displayed as they are sung), where would you rather use this feature?
On my computer
On my portable audio player
On both
I wouldn't use it

REFERENCES

[1] J. Yuan, L. Duan, Q. Tian and C. Xu, “Fast and robust short video clip search using an index structure”, Proc. of the 6th ACM SIGMM international workshop on Multimedia information retrieval, New York, NY, USA, pp. 61-68, 2004.

[2] A. Amir, M. Berg and H. Permuter, “Mutual relevance feedback for multimodal query formulation in video retrieval”, Proc. of the 7th ACM SIGMM international workshop on Multimedia information retrieval, Hilton, Singapore, pp. 17-24, 2005.

[3] G. Gaughan, A. Smeaton, C. Gurrin, H. Lee, K. McDonald, “Design, implementation and testing of an interactive video retrieval system”, Proc. of the 5th ACM SIGMM international workshop on Multimedia information retrieval, Berkeley, CA, USA, pp. 23-30, 2003.

[4] A. Sheth, C. Bertram and K. Shah, “VideoAnywhere: a system for searching and managing distributed heterogeneous video assets”.

[5] B.S. Manjunath, P. Salembier and T. Sikora, “Introduction to MPEG-7: multimedia description interface”.

[6] ISO/IEC JTC1/SC29/WG11, “MPEG-7 Overview (version 10)”.

[7] Frameline 47, first commercial MPEG-7 tool, www.frameline.tv.

[8] IBM MPEG-7 Annotation Tool, http://www.alphaworks.ibm.com/tech/videoannex.

[9] The Dublin Core Media Initiative (DCMI), http://www.dublincore.org/.

[10] C. Binstock, D. Peterson, M. Smith, M. Wooding, C. Dix, C. Galtenberg, “The XML Schema complete reference”, 2002.

[11] M. Pilgrim, “Dive into Python”.

[12] Dolby Laboratories Inc., “All about audio metadata”, 2001.

[13] B. Whitman, D. Roy and B. Vercoe, “Learning Word Meanings and Descriptive Parameter Spaces from Music”.

[14] J. Dunn, D. Byrd, M. Notess, J. Riley, and R. Scherle, “Variations2: retrieving and using music in an academic setting”.

[15] P. Cano, M. Koppenberger, N. Wack, “Content-based music audio recommendation”.

[16] W3C, XML Schema Validation Tool, http://www.w3.org/2001/03/webdata/xsv.

[17] Borkware, QTMP3, http://www.borkware.com/rants/sound/.

[18] Apple, “Event-Driven XML programming guide for Cocoa”, http://developer.apple.com/documentation/Cocoa/Conceptual/XMLParsing/index.html.

[19] D. Liu, L. Lu and H.J. Zhang, “Automatic mood detection from acoustic music data”, Proc. of the 4th International Conference on Music Information Retrieval, Baltimore, Maryland, USA, 2003.

[20] V. Sandvold and P. Herrera, “Towards a semantic descriptor of subjective intensity in music”, Proc. of the International Computer Music Conference, Barcelona, Spain, 2005.

[21] M. Nilsson and J. Sundström, “The short history of tagging”, http://www.id3.org/history.html.

[22] Xiph.org foundation, “Vorbis I specification”, http://xiph.org/vorbis/doc/Vorbis_I_spec.pdf, 2004.

[23] Polyphonic HMI, “Hit Song Science technology”, http://www.polyphonichmi.com/technology.html.

[24] Fraunhofer Institute, “Query by humming”, www.idmt.fraunhofer.de/eng/press_media/download/product_information/qbh_eng_web.pdf, 2004.

[25] B. Pardo, J. Shifrin and W. Birmingham, “Name that tune: a pilot study in finding a melody from a sung query”, Journal of the American society for information science and technology, vol. 55, 4, pp. 283-300, 2004.

[26] Audible Magic Corporation, “Content Alert”, http://www.educause.edu/elements/attachments/rfi/rfi_1/AudibleMagic_summary.pdf, 2003.

[27] G. Tummarello, C. Morbidoni, P. Puliti, A. F. Dragoni and F. Piazza, “From multimedia to the semantic web using MPEG-7 and computational intelligence”, Proc. of the Web Delivering of Music, Fourth International Conference on (WEDELMUSIC'04), vol. 00, pp. 52-59, 2004.

[28] MacSpeech Incorporated, “iListen speech recognition software”, http://www.macspeech.com/.

[29] M. A. Casey and A. Westner, “Separation of mixed audio sources by independent subspace analysis”, International Computer Music Conference, pp.154–161, 2000.

[30] GJ. Jang and TW. Lee, “A probabilistic approach to single channel blind signal separation”, Proc. of the Neural Information Processing Systems 2002, Vancouver, Canada, pp. 1173-1180, 2002.

[31] L. Torres, L. Lorente and J. Vila, “Automatic face recognition of video sequences using self-eigenfaces”, Proc. of the International Symposium on Image/video Communication over Fixed and Mobile Networks, Rabat, Morocco, 2000.

[32] WS. Lee and KA. Sohn, “Face recognition using computer-generated database”, Proc. of the Computer Graphics International, pp. 561-568, 2004.

[33] A. Mathes, “Folksonomies - cooperative classification and communication through shared metadata”, University of Illinois Urbana-Champaign, 2004.

[34] Joanneum Research, “MPEG-7 library”, http://iiss039.joanneum.at/cms/index.php?id=80