PUBLISHING MUSIC SIMILARITY FEATURES ON THE SEMANTIC WEB

Dan Tidhar, György Fazekas, Sefki Kolozali, Mark Sandler
Centre for Digital Music
Queen Mary, University of London
Mile End Road, London E1, UK
{dan.tidhar, gyorgy.fazekas, sefki.kolozali, mark.sandler}@elec.qmul.ac.uk

ABSTRACT

We describe the process of collecting, organising and publishing a large set of music similarity features produced by the SoundBite [10] playlist generator tool. These data can be a valuable asset in the development and evaluation of new Music Information Retrieval algorithms. They can also be used in Web-based music search and retrieval applications. For this reason, we make a database of features available on the *Semantic Web* via a SPARQL end-point, which can be used in *Linked Data* services. We provide examples of using the data in a research tool, as well as in a simple web application which responds to audio queries and finds a set of similar tracks in our database.

1. INTRODUCTION

Similarity-based retrieval is an important subject area in music information research. Yet, researchers working in this field are often limited by the unavailability of large audio collections, copyright restrictions, and even more often, unreliable metadata associated with songs in a particular music database or personal library. This paper describes a system for collecting and publishing music similarity features from a large user base coupled with valuable editorial metadata. Metadata are verified against MusicBrainz, [1] a large public database of editorial information on the Web, and published together with the matching similarity features on the *Semantic Web* [1]. We explore some research opportunities opened by the system, and describe SAWA [2] recommender, a sample web application which demonstrates how the published data can be used. Rather than describing a music recommender in detail, our primary motivation is in making high quality data available for similarity and recommendation research in a standardised way.

The heart of the data collection system is SoundBite [10] [15], a tool for similarity-based automatic playlist generation. Soundbite is available as an iTunes plugin, and is currently being implemented as a plugin for other audio players as well. Once installed, it extracts features from the user's entire audio collection and stores them for future similarity calculations. It can then generate playlists consisting of n most similar tracks to any given seed track specified by the user. The similarity data currently consists of 40 values per track, based on the distribution of Mel-Fequency Cepstral Coefficients (MFCC) as described in [10]. The extracted features are also reported to a central server, where they become part of the so called *Isophone* database. This database is used for aggregating information from SoundBite clients, consisting of editorial metadata and similarity features for each audio track. The entire system may therefore be regarded as a distributed framework for similarity feature extraction. The accumulated data can be valuable to the research community, and may also be used by other audio similarity and recommendation systems. In order to facilitate such usage, we publish a cleaned-up portion of the data on the Semantic Web.

The rest of the paper is organised as follows: In section two, we provide brief explanations of some of the key terms relevant to the technologies we use. In section three, we describe the published data set, the collection system architecture, the data clean-up process, and the way researchers as well as Semantic Web applications can access the data using a *SPARQL end-point* [3]. Finally, in section four, we describe our prototype recommender, a publicly accessible web application based on this data set.

2. LINKED DATA AND THE SEMANTIC WEB

Building the Semantic Web involves creating a machine-interpretable web of data in parallel to the existing web of documents [1]. By uniformly integrating diverse data and services, it aims to enable applications which would be difficult, if not impossible, to build using prevailing incompatible interfaces and representation formats. An example application from the world of music would interlink content providers (music labels, music sellers, online radio stations), meta-databases holding musical and artists infor-

[1] http://www.MusicBrainz.org/

[2] SAWA stands for Sonic Annotator Web Application. A search and recommendation system built on SAWA and the SoundBite data set is available at: http://www.isophonics.net/sawa/rec

[3] A web resource that responds to queries using the SPARQL Protocol and RDF Query Language, an SQL-like language for accessing RDF [9] data bases.

mation, semantic audio tools and music identification services, and perhaps even music collections held on personal computers. This could revolutionise the way we access or discover new music. However, creating such a distributed network requires that all data sources speak the same language, i.e., are governed by a common schema.

Because of the diverse and unbounded nature of information on the general Web (and we believe that musical information is just as diverse), a major challenge was set forth to Semantic Web developers: How to design a standard, extensible schema for representing information encompassing a wide range of human knowledge? The Semantic Web's answer to this apparently complex and circular problem is in specifying how information is published, rather than trying to arrange everything into rigid data structures.

2.1 Semantic Web Technologies

The key concepts and technologies enabling the development of the Semantic Web are the Resource Description Framework (RDF) [9], Semantic Web ontologies, and RDF query languages.

RDF is a conceptual data model. It provides the flexibility and modularity required for publishing diverse semi-structured data — that is, just about anything on the Semantic Web. It is based on the simple idea of expressing statements in the form of *subject — predicate — object*. Elements of these statements are *literals*, and *resources* named by Uniform Resource Identifiers (URI). This provides the model with an unambiguous way of referring to things, and – through the HTTP dereferencing mechanism – access to additional information a resource may hold. Simple RDF statements, however, are not sufficient for expressing things unambiguously. In order to be precise in our statements, we need to be able to define, and later refer to concepts and relationships pertinent to a domain or application. Ontologies are the tools for establishing these necessary elements.

Semantic Web ontologies are built on the same conceptual model that is used for expressing data. However, additional vocabularies were created for expressing formal ontologies. RDF is the basis for a hierarchy of languages recommended by the W3C [4] . This includes the *RDF Schema Language* (RDFS) for defining classes and properties of RDF resources and the *OWL Web Ontology Language* for making RDF semantics more explicit. [5]

Besides a standard way of representing information, access to data also needs to be standardised. The *SPARQL Protocol and RDF Query Language [6]* is a recent recommendation by the W3C for accessing RDF data stores. A Web interface which accepts and executes these queries is commonly referred to as a *SPARQL end-point*.

SPARQL allows access to information in a multitude of ways. In the simplest case, it is used in a similar manner to querying a relational database using SQL [6] . A query – consisting of a set of triple patterns – is matched against the database. Results are then composed of variable bindings of matching statements, based on a *select* clause specified by the user. This can be used to retrieve information about a particular resource. More complex SPARQL queries are frequently used to aggregate information in a particular way. For example, a user agent may interpret a query and aggregate data from various sources on the fly. The standardisation and increasing support of the SPARQL query language strongly promotes the adoption of RDF as a prevailing metadata model and language.

2.2 Linked vs. Structured Data

There are already a large number of services exposing structured data on the Web. Examples include Google, Yahoo, OpenSearch, Amazon, Geonames, and the MediaWiki APIs. Music-related data providers include the Magnatune and Jamendo labels, and the MusicBrainz database. Most of these services use proprietary XML-based data formats. This is sufficient for structuring data for a given application, yet, because of the fairly ad-hoc definition of concepts in XML schema, these formats do not provide the means for transparent access to a variety of services. The *Linked Data* community [7] offers standardised access to some information exposed by the previously listed services, as well as other related data sets. In Linked Data services, the reliance on diverse interfaces and result formats is reduced by using RDF as a common representation. This also provides the means for making data available on the Semantic Web.

Most existing metadata formats for expressing audio features are also based on XML. MPEG-7 [7] and ACE-XML [11] are perhaps the most prominent examples. The structural and syntactical requirements for expressing elements and schemes in MPEG-7 are fulfilled by using an extended XML schema language. Although this allows the production of machine-parsable data, it does not provide a machine-interpretable representation of the semantics associated with MPEG-7 metadata elements. The same problem arises with the ACE-XML format developed for the jMIR package, linking components such as jAudio for feature extraction, and the ACE classification engine. A common problem can be recognised in using XML for standardised syntax, while the data model remains disjoint and often arbitrary, with ad-hoc definition of terms, and without the ability to define meta-level relationships such as the equivalence of certain concepts. This hinders the ability to integrate services expressing metadata in these formats, or the reuse of any of the defined terms in other domains. Our data, on the other hand, is expressed using a flexible RDF and Web Ontology based data model. It is compatible with the Music Ontology [12], which is already widely used in Linked Data applications.

[4] The World Wide Web Consortium: `http://www.w3.org/`

[5] For example, OWL-DL (description logic) can impose restrictions on the range and domain types of properties, or constraints on cardinality.

[6] Structured Query Language

[7] "Linking open data on the semantic web", `http://linkeddata.org/`

2.3 Ontologies

As mentioned in section 2.1, only a conceptual model is provided by RDF. Ontologies are used for the actual definition of pertinent terms and relationships. Recent efforts [13] toward integrating music-related web services and data sources have led to the creation of the Music Ontology [12]. It serves as a standard base ontology which can be readily used for describing a wide range of concepts. These include high-level editorial data about songs or artists, production data about musical recordings, and detailed structural information about music using events and timelines. The ontology provides the basis for numerous extensions, including the Audio Features Ontology [14]. The music similarity features published and used by the services described in this paper are expressed using these ontologies.

3. THE SOUNDBITE DATASET

The SoundBite dataset consists of MFCC features and MusicBrainz identifiers for a cleaned-up subset of the data reported back to the central server by the different instances of the SoundBite client application. Currently, the database includes metadata for 152,410 tracks produced by 6,938 unique artists. These numbers are expected to grow as the number of SoundBite users grows, and the data clean-up procedure is refined. We believe that this dataset can be especially valuable because of its scope and diversity. Furthermore, it originates from real-world users, and therefore reflects at least a part of the users' community interests and relevant needs. It is not susceptible to any biases which might be implicit in datasets which are artificially-created for research purposes. We currently do not collect personal data about SoundBite users, although this information might be of interest for other studies. However, at the time of writing this paper, the growing user community already seems sufficiently large and varied for the dataset to cover the most popular genres. The dataset coverage is expected to further improve as a direct result of user base growth and further clean-up.

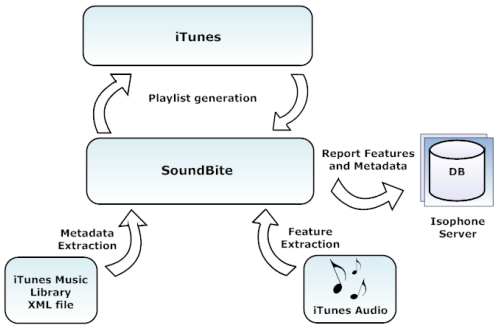

Figure 1. Simplified SoundBite Architecture.

As mentioned in section 1, the features extracted by each instance of the SoundBite client application are reported back to a central server, where they are stored in a database alongside the relevant textual metadata. Figure 1 illustrates the interaction between the iTunes application, the SoundBite plugin, and the Isophone server. The relevant resources on the client side are iTunes music library and the corresponding XML file which describes the collection. Since textual metadata contained in this XML file, such as title and artist, are often inserted or altered by the users themselves, we cannot rely on their accuracy. They certainly cannot be used as unique identifiers which are necessary for facilitating public usage of the dataset. Prior to publishing, the data need to undergo a clean-up process, as described in following sections. Using the MFCC data for automatic playlist creation, as done by the Soundbite plugin, requires similarity metrics to be defined on the data. These are not provided as part of the dataset, but are rather considered part of an algorithm which utilizes the data for a particular application, namely, playlist creation. The published data facilitate the exploration of further similarity algorithms and applications.

3.1 Data filtering and publishing

Since the audio tracks to which the features relate reside in end-users' audio collections, they are inaccessible to us and we obviously cannot provide them as part of the dataset. It is therefore of crucial importance that we do provide unique identifiers to the audio material, without which the provided features can be of very little use. As a source for such unique identifiers, and as an aid in metadata-based filtering, we use the MusicBrainz database.

MusicBrainz is a comprehensive public community music meta-database. It can be used to identify songs or CDs, and provides valuable data about tracks, albums, artists and other related information. MusicBrainz can be accessed either through their web site or by using client applications via an application programming interface (API). We use the MusicBrainz service as metadata reference in the filtering process, and use MusicBrianz ID's as unique identifiers which are published together with the MFCC's.

The editorial metadata reported back to the server by SoundBite (as depicted in figure 1) include the entire content of the iTunes Music Library XML file. The data clean-up procedure currently uses the following metadata items:

- Track Title
- Main Artist
- Album Title
- Track Duration
- File Format
- Bit Rate

In the first stage of the clean-up process, title, artist, and album are matched against the MusicBrainz database. The track's duration is used for resolving ambiguities, as well as for sanity check (a large difference between the reported duration value and the duration retrieved from MusicBrainz may indicate that the other fields are erroneously or maliciously wrong). Each matching track is assigned an

ID provided by the MusicBrainz database, which serves as unique identifier. We found that about 28% of the entries in our database had exact matches (artist, title, album, and approximate duration) in the MusicBrainz database. The remaining 72% are stored for possible future use, but do not currently qualify for publishing. The relatively small proportion of tracks that do qualify can be regarded as an indication of the poor reliability of textual metadata in end users' audio collection.

As indicated in [16], MFCC features are more robust at higher bit rates. Therefore, in the second stage the data is further filtered according to maximum bit rate and best quality audio file type (e.g. keeping AACs as opposed to MP3s), in order to preserve the highest quality features for each track. Since these parameters are included in the metadata reported to the server, this doesn't require access to the audio files themselves.

Once cleaned-up and filtered as described above, the MFCC features and the obtained MusicBrainz ID's are exported from the database as RDF's using the D2R Mapping [2], with the appropriate linking to the Audio Features [14] and SoundBite ontologies (see figure 2). They are then made available via a SPARQL end-point on our server [8] .

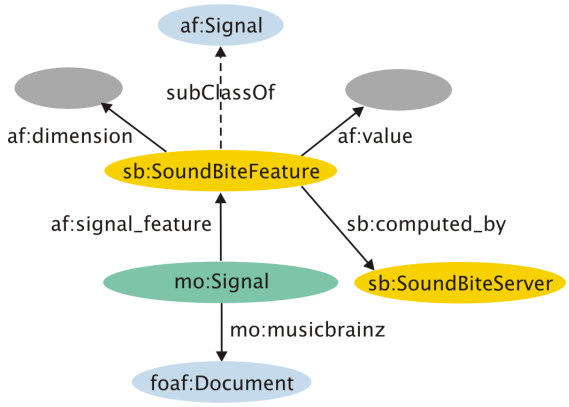

Figure 2. Accessing the SPARQL endpoint using the SoundBite ontology.

4. APPLICATIONS

In this section we describe how our data set can be used as basis for the development of new music similarity and music recommendation algorithms. Additionally, we provide an example of a prototype audio search engine. The service uses our database to find tracks similar to an audio query and returns editorial metadata about the found set obtained from external web-services.

4.1 Research Platform

There has recently been a significant amount of research on music similarity and audio-based genre classification. Both fields use content-based descriptors extracted from audio signals. Apart form being computationally expensive, audio-similarity features coupled with matching textual metadata are not easily obtainable in large quantities. The published Isophone data provide an excellent opportunity for further research based on a reliable music collection with readily-available MFCC features. Obviously, since the available features are calculated prior to being published, the dataset does not accommodate changes to the algorithms which produced them in the first place. There is, however, plenty of room for experimentation with the way the different features are combined to form similarity metrics, and the way they are used on the application level. We use the dataset in a research platform, which facilitates such experiments. We are currently exploring different similarity metrics based on the published features, as well as different ways to combine the features with other relevant data, e.g. in the context of hybrid recommender systems (see, for exmple, [5]). As a proof of concept, and to demonstrate how the research community could use the published data, we have implemented a tool which queries the SPARQL endpoint to obtain MFCC's for given tracks, to facilitate the above mentioned research activities.

4.2 SAWA-recommender

SAWA-recommender [9] is a simple *query by example* search service made available on the Web. Its main goal is to demonstrate an application of the published music similarity features. In this section, we outline the use and construction of this service.

A query to SAWA-recommender is formed by one or more audio files uploaded by the user. It is typically based on single file, however, uploading multiple audio files is also allowed. In the latter case, a small set of songs forms the basis of the query, either by considering similarity to any of the uploaded songs (and ranking the results appropriately), or formulating a single common query by jointly calculating the features of the query songs. The calculated query is matched against the Isophone database holding similarity features and MusicBrainz identifiers associated with each song in this database. Finally, the MusicBrainz web API is used to obtain metadata about songs in the result set. These are displayed to the user. The metadata consist of basic information such as song title, album title and the main artist's name associated with each song. We also provide direct links to MusicBrainz, as well as Linked Data services such as BBC Music [10] artist pages.

For each uploaded file, the system also attempts to identify the audio by calculating a MusicDNS [11] fingerprint and associated identifier. This identifier is matched against the MusicBrainz database to obtain editorial data, hence one can also use the service to find more information about an audio file (see figure 3).

The architecture of the web application is depicted in figure 4. The system is built on software components de-

[8] http://dbtune.org/iso/

[9] http://isophonics.net/sawa/rec

[10] http://www.bbc.co.uk/music/

[11] http://www.musicdns.com/

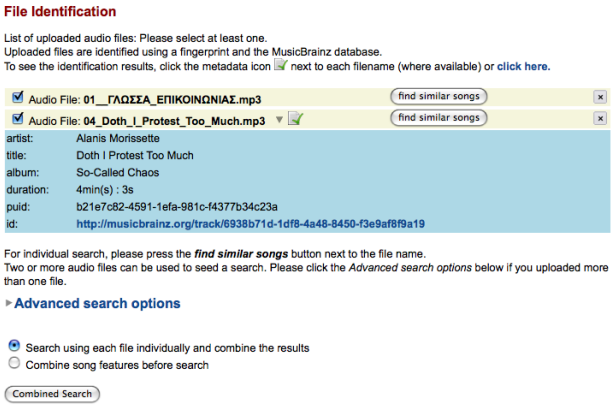

Figure 3. File Identification and Selection Interface.

veloped in the OMRAS2 project [12] and a small set of common open-source libraries.

The signal processing back-end of the service is provided by Sonic Annotator [13] together with Vamp audio analysis plugins [3]. These plugins use an application programming interface (API) designed for audio feature extraction. They take audio input and return structured numerical results. While Vamp plugins perform the feature extraction step (implemented in efficient C++ code), Sonic Annotator is the host application that reads audio data and applies plugins to one or more files in batch. This program accepts configuration data and returns audio features in RDF according to specific ontologies [14] [4]. For the purpose of this present search system, we configure Sonic Annotator and a suitable Vamp plugin to extract audio similarity features based on MFCCs [10].

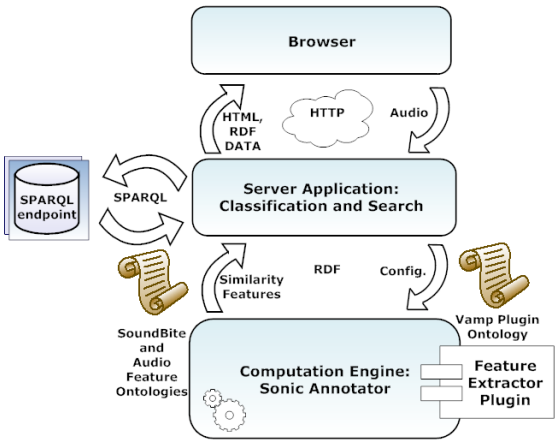

Figure 4. Search Engine System Architecture.

The core of the search system is a Python application which provides a Web interface and a basic search and classification engine. It also manages user sessions and uploaded files. Since users may upload copyrighted material, user sessions are fully isolated, and all audio files are automatically deleted as the user leaves the service.

[12] http://www.omras2.org/
[13] Available at: http://omras2.org/SonicAnnotator

The Web interface is built using the Cherrypy [14] Python library. This allows the implementation of HTTP request handlers as ordinary methods defined within a web application class. Using this library, it is straightforward to accept audio files as well as publishing data received from other system components using dynamically generated web pages.

Query processing and database search is performed in three steps. First, we extract features from the uploaded audio files. For optimised search, the query features are matched against a model trained on the whole database. Finally, a selected group of songs are ranked based on their similarity to the query and the results are displayed to the user.

Although simple linear search was suggested for personal collections, [10] the size of our current database is over 150.000 tracks and it is expected to grow. For this reason, we partition the data space by similarity to form self-similar groups of songs. These groups or clusters can then be used to index the database. We can limit the search space by choosing the best matching cluster based on its proximity to the query song. Hence, the number of direct similarity calculations is greatly reduced. Since our goal is search optimisation rather than classification, we choose an unsupervised learning algorithm using a self-organising model, similar to a Self Organising Map [8]. The details of this exceed the scope of our current discussion. However, it is important to note that using the symmetrised Kullback-Leibler (KL) divergence as basis for training and classification, we could verify the scarcity of hubs reported in [10] using a 100-times larger database of features. The songs are roughly equally distributed among the nodes. Only 4% of the nodes became hubs (containing a large set of songs) and 3% of them contain fewer songs. We also found that this phenomenon is largely independent of the size of the model (the number of nodes). The fact that the collection can be partitioned automatically by grouping similar songs - without obtaining too many over-populated clusters (hubs) - shows that the database is well balanced and justifies the choice of metrics and learning algorithm. This is also favourable for the search application, since we can limit the number of songs where the similarity has to be explicitly calculated and compute the divergence only within a single class without significantly modifying the results set. In our current implementation, a local copy of the partitioned database is used for searching, however, the model is trained on the data available at the SPARQL end-point. This is achieved by an appropriate SPARQL query, generated and issued in each training iteration. This way, the model can easily be adjusted if the database is expanded in the future. For producing the final results, a limited set of similar songs is collected and ranked by similarity to the query song(s) using the KL divergence described in [10]. Finally, the metadata are obtained from MusicBrainz and displayed to the user.

Since our similarity assessment follows the same principles applied in SoundBite, these results can be seen as

[14] Available at: http://www.cherrypy.org/

content-based recommendations. However, given the size of the database they might be useful for identifying unknown songs or song segments. In a commercial situation, our service might be useful in finding an alternative for a song, where a copyright agreement for its use can not be obtained.

5. CONCLUSION

We described the SoundBite dataset and its publication on the Semantic Web. We believe that due to its scope and diversity (which are expected to grow even further), it is a valuable resource for researchers as well as application developers. We provided some examples of applying the data in research and prototyping web applications. These initial examples strengthen our beliefs regarding the value and potential of this dataset, and we therefore intend to continue to follow our policy of publishing accumulated data on the Semantic Web. We intend to further develop this particular dataset by collecting more raw data and refining the filtering process, and to continue developing applications which utilize the data for research purposes and public use.

6. ACKNOWLEDGEMENTS

The authors acknowledge the support of the School of Electronic Engineering and Computer Science, Queen Mary, University of London. This work has been carried out at the Centre for Digital Music and supported by the EPSRC-funded ICT project OMRAS- 2 (EP/E017614/1).

7. REFERENCES

[1] T. Berners-Lee, J. Handler, and O. Lassila, “The semantic web”, *Scientific American*, pp. 34–43, May 2001.

[2] C. Bizer and R. Cyganiak, “D2R Server-Publishing Relational Databases on the Semantic Web", *5th International Semantic Web Conference,* Athens, GA, USA, 2006.

[3] C. Cannam, “The Vamp Audio Analysis Plugin API: A Programmer’s Guide”, `http://vamp-plugins.org/guide.pdf` Last accessed: July, 2009.

[4] C. Cannam, “The Vamp Plugin Ontology”, `http://omras2.org/VampOntology`, Last accessed: July, 2009.

[5] J. Donaldson,, “A hybrid social-acoustic recommendation system for popular music", *In proceedings of the 2007 ACM conference on Recommender systems*, Minneapolis, MN, USA, 2007.

[6] K. Grant Clark, L. Feigenbaum, E. Torres, (eds.) “SPARQL Protocol for RDF” *W3C Recommendation*, 15. January 2008 `http://www.w3.org/TR/2008/REC-rdf-sparql-protocol-20080115/` Last accessed: July, 2009.

[7] H. Kim, N. Moreau, T. Sikora, “MPEG-7 Audio and Beyond: Audio Content Indexing and Retrieval." *Wiley and Sons*, October 2005.

[8] T. Kohonen, “Self-Organizing Maps", *Springer*, Berlin, 1995.

[9] O. Lassila, R. Swick, “Resource description framework model and syntax specification”, 1998. `http://citeseer.ist.psu.edu/article/lassila98resource.html` Last accessed: March, 2009.

[10] M. Levy and M. Sandler, “Lightweight measures for timbral similarity of musical audio”, *In Proceedings of the 1st ACM Workshop on Audio and Music Computing Multimedia*, (Santa Barbara, California, USA, October 27, 2006). AMCMM ’06. ACM, New York, NY, 27-36.

[11] D. McEnnis, C. McKay, I. Fujinaga, P. Depalle, "jAudio: A Feature Extraction Library", *in Proc. of the International Conference on Music Information Retrieval,*London, UK, 2005.

[12] Y. Raimond., S. Abdallah, and M. Sandler. “The Music Ontology”, *Proceeedings of the 8th International Conference on Music Information Retrieval*, Vienna, Austria, 2007.

[13] Y. Raimond. and M. Sandler, “A Web of Musical Information”, *Proceeedings of the 9th International Conference on Music Information Retrieval*, Philadelphia, USA September 14-18, 2008.

[14] Y. Raimond, “Audio Features Ontology Specification”, `http://motools.sourceforge.net/doc/audio_features.html` Last accessed: March, 2009.

[15] M. Sandler, M. Levy, “Signal-based Music Searching and Browsing”, *ICCE 2007. International Conference on Consumer Electronics*, 10-14 Jan. 2007.

[16] S. Sigurdsson, K. Brandt Petersen, T. Lehn-Schiler, “Mel Frequency Cepstral Coefficients: An Evaluation of Robustness of MP3 Encoded Music,” *ISMIR 2006 7th International Conference on Music Information Retrieval*.