Journal of New Music Research
2005, Vol. 34, No. 2, pp. 173–184

Routledge
Taylor & Francis Group

Editorial Metadata in Electronic Music Distribution Systems: Between Universalism and Isolationism

François Pachet, Amaury La Burthe, Jean-Julien Aucouturier and Anthony Beurivé

Sony CSL – Paris

Abstract

We address the problem of metadata management in the context of future Electronic Music Distribution (EMD) systems, and propose a classification of existing musical editorial systems in two categories: the isolationists and the universalists. Universalists propose shared information at the expense of consensuality, while isolationist approaches allow individual parameterization at the expense of the lack of reusability. We propose an architecture and a system for managing editorial metadata that lies in the middle of these two extremes: we organize musical editorial information in such a way that users can benefit from shared metadata when they wish, while also allowing them to create and manage a private version of editorial information. A mechanism allows the synchronizing of both views: the shared and the private.

1. Introduction

We propose in this article a short scenario describing how users could benefit from a new metadata management system. Then, we expose the relationship between editorial data management (EMD) systems and editorial metadata.

1.1 Vision

Picture yourself on a trip to Iceland. Before leaving, you have obtained a Bjork mp3 from a friend. You have loaded the file on your mp3 walkman, which already contains about 5,000 of your favorite tunes. Each song on this walkman is associated to metadata: about the artist, the title, and so on. Although you enjoy the new tune, you do not know much about Bjork: you have only been able to classify the artist in your walkman's local genre taxonomy as ''Electronica / Icelandic''.

As you walk by a café in Reykjavik, you decide that you would like to see what information the local community has to offer about Bjork (she's Icelandic, isn't she?) You turn on your walkman, which immediately connects to the nearby metadata server at the café and download locally available information. Unfortunately, access to the music files is copyrighted and restricted. However, you have access to the corresponding metadata. Your walkman automatically browses the local genre taxonomy and is able to trace down Bjork, although she has been entered with her full Icelandic name, Gudmundsdottir, and classified in a different genre: Pop/Trip Hop. The local server has a lot of available metadata about Bjork, but its access is restricted to Iceland so you choose to gather metadata while you are there. Your walkman's metadata manager is able to fill in the fields corresponding to album, track listing and recording date, and also downloads the fact that Bjork was a member of another Icelandic band: the Sugarcubes. Your friend back home will certainly be interested in knowing this! However, you decide to preserve your carefully built genre taxonomy, and do not update your local genre metadata or Bjork's full name: it is only useful on the local server because it also holds some mp3s of classical music by a female pianist named Anna Gudmundsdottir.

You later notice that the local server has a metadata field that does not exist in your metadata manager: the mood of the song. You realize how useful this would be for browsing your own collection. Your walkman

Correspondence: François Pachet, Sony CSL Paris, 6, rue Amyot, 75005 Paris, France. E-mail: Pachet@csl.sony.fr

DOI: 10.1080/09298210500148504

matches the many songs that you have in common with the server, creates a new field in your metadatabase, and uploads the mood information. While you are at it, you also download information about the many other songs by Bjork that sit on the server. The next thing you do is to connect to your favorite online music shop, and buy all Bjork's songs that have the same mood as the one that sits on your walkman. As the shop's server does not support browsing by mood, you decide to contribute to their metadatabase by submitting the mood metadata about your songs. In return, your point card is credited, which allows you to download one extra song for free: Why not try this "Sugarcube" thing?

1.2 EMD and editorial metadata

The notion of musical metadata is now well established as a key ingredient of systems dedicated to the electronic distribution of music audio files. To manage collections of audio music titles, either personal or online, an application must have access to many kinds of information to identify, categorize, index, classify and generally organize music titles. Because there are so many types of information that can be made explicit about music titles, musical metadata comes in many flavors. However, classifying musical metadata based on its ontological nature is a difficult task because there is virtually no limit to what can be said about a music title. In this context, we are interested in metadata that has the following properties:

- It is *useful* – that is, it corresponds to actual features of the applications targeted.
- It is *consensual* – that is, the information makes sense to a large part of the targeted audience, and these people would usually agree on them.

The distinction between the various forms of musical metadata is usually made based on the way this metadata is extracted, adopting an engineering viewpoint. Not only is this approach easier, but it is probably the only one today that is reasonable. Musical metadata can be divided in the following categories in this scheme:

- *Identification information*: this allows for the unique characterization of a music title.
- *Editorial information*: this is related to prescriptive knowledge about the music (consensual information, facts, content description, etc.).
- *Acoustic features*: this corresponds to objective, acoustic features of the music titles. It is normally extracted automatically from the signal.
- *Cultural information*: this captures the similarity between music that emerges from socially shared sources of textual information such as web search engines.

Sony CSL's Music Browser project (Pachet et al., 2004), conducted in the context of the Cuidado and SemanticHifi IST projects (http://shf.ircam.fr/; see section 8), consists of designing and implementing a music browser that gathers all these kinds of metadata. One exemplar feature of the Music Browser is that it implements the complete chain linking music titles seen as objective items (signals or texts) to users, considered as complex subjects. This article focuses on editorial metadata as it is designed and used in the Music Browser. Although some of the architecture and techniques presented here also may apply to the management of acoustic and cultural metadata, they will not be discussed here. Notably, connections between our architecture and the MPEG7 standard, which provides a format for representing acoustic metadata information, will not be discussed (see, e.g., Herrara et al., 1999).

Moreover, we place ourselves in the emerging context of local and mobile *ad hoc* networks (MANET) (Basagni et al., 2004). *Ad hoc* networks are wireless, self-organizing systems formed by cooperating nodes within communication range of each other that form temporary networks. In such environments, different users, with different goals, share the resources of their devices and form an open community. A few projects envision music browsing in the context of MANETs. SoundPryer (Axelsson & Östergen, 2002) describes a system aimed at car travellers that enables the p2p wireless exchange of music files. TunA (Bassoli et al., 2003) is designed for urban pedestrians, and allows the sharing of music as well as information about the user (name, age, etc.) in order to foster social interaction. In both systems, the issue of sharing metadata about the actual content being shared is left unaddressed.

2. Existing editorial metadata information systems

2.1 Isolationist versus universalist

Editorial metadata is no longer a fantasy: they crop up in virtually every musical application. There are, however, two radically opposed approaches to how this metadata is organized. The first, the "Isolationist" approach, consists of letting individual users handle their metadata in isolation, with very limited sharing. This is the approach of most peer-to-peer systems such as Kazaa (www.kazaa.com). The second, the "Universalist" approach, consists of creating a central, shared database server from which all clients feed. Examples of this approach are CDDB (www.cddb.com), FreeDB (www.freedb.org), AllMusicGuide (AMG) (www.allmusic.com), MusicBrainz (www.musicbrainz.org) and MoodLogic (www.moodlogic.com).

The isolationist approach consists of letting users manage editorial information themselves. A system such as Kazaa (see Figure 1) proposes different fields based on ID3 tags for describing music titles. The fields are: title, artist, album, category (corresponding to genre) and year. Additionally, one can add the language, some keywords and a short description of the track. This approach has an obvious drawback in terms of usability: users must painstakingly fill in the fields for all the new titles they enter into their collection. There is almost no sharing of this metadata, other than through the actual transfer of files. When user A downloads a file from user (B), he or she also gets the associated metadata. This metadata can itself be incompatible with existing metadata. For instance, user A may have decided to enter an artist's name as ''McLaughlin, John'', while B prefers ''John McLaughlin''. Thus, A has to change manually all the artist metadata of the downloaded files.

The universalist approach is aimed at suppressing this drawback by imposing fixed metadata. A central server contains the metadata for a certain number of songs. When a user decides to annotate his or her local files, a query is made to the server, together with signatures of the files. The server identifies the files from the signatures and provides the required metadata. While this approach does avoid manual annotation, it also comes at a price: the metadata is fixed, and imposed, and the user cannot change it; and it works only to the extent that the signature database of the server actually includes all the music files of the user's collection. CDDB and later the FreeDB and MusicBrainz (see Figure 2) systems are typical examples of this approach and the community vision of the two last (i.e., everyone contributes for the benefit of everybody) achieve excellent results. However, while they are good at expressing consensus, there is no room for personalization.

The editorial information system we propose lies in the middle of these two extremes. We allow both sharing of information and local personalization. In the ensuing sections, we will detail the nature of the metadata managed and the client-server architecture of the system.

2.2 Two kinds of editorial metadata

Editorial information servers such as Musicbrainz, Moodlogic or AMG provide two sorts of metadata: consensual information or facts about music titles and artists; and content description of titles, albums or artists. The first category does not raise any particular problem as this information is universal by nature. It includes, for instance: artists and song name (AMG, Moodlogic & Musicbrainz); albums and track listings (AMG & Musicbrainz); group members (AMG); date of recording for a given title (Moodlogic); short biography for artists with date of birth and years of activity (AMG); albums plus, sometimes, album credits (AMG); labels (AMG); and charts and awards (AMG with Bilboard.com). However, these kinds of information are not particularly useful for content-based search systems such as the music browser, which aim at matching music titles with tastes. Tastes, whatever they are, are rarely well expressed using administrative information on music.

The second category is both more interesting and problematic. Content description includes such widely needed information as: artist style (AMG); artist instru-

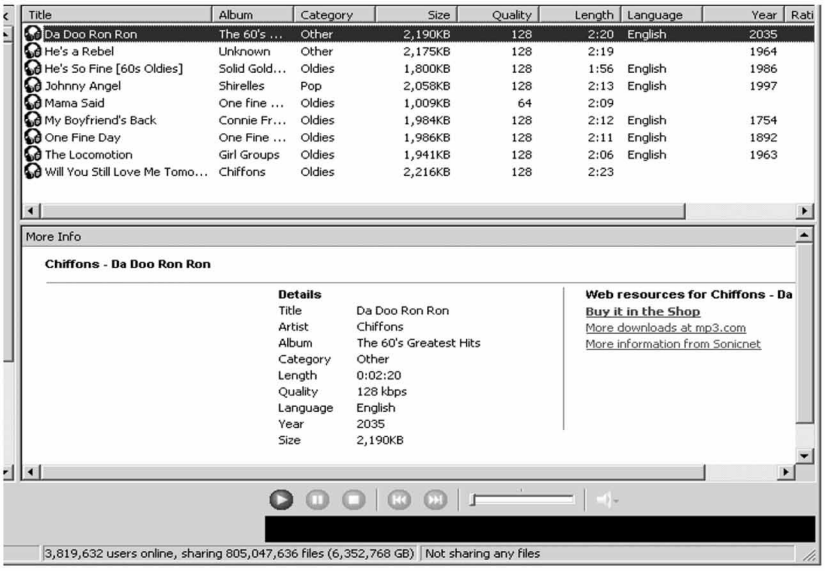

Fig. 1. The management of ID3 tags in Kazaa.

Fig. 2. Track listing of the album *A Hard Day's Night* by The Beatles, as found in the MusicBrainz database.

ments (AMG); song mood (Moodlogic); song review (AMG); song or artist genre (AMG, Moodlogic); and more generally attributes aimed at describing the intrinsic nature of the musical item in question (artist or song). These descriptions, again, are useful to the extent that they can be used for musical queries in large catalogues. The user tests performed in the Cuidado project showed that there is virtually no limit to such information. As explained in the next section, we have added several more such descriptors in the Music Browser. Moreover, we propose an open approach where the user can adapt/add any descriptor to suit his or her needs or tastes.

In conclusion, the existing approaches cover only the two extreme cases: editorial metadata that is universal and shared by the whole world (AMG, Musicbrainz) or metadata that is unique to their author (peer-to-peer systems) and transmitted on a file-by-file basis. We propose an intermediate approach that covers the case where users want both to share consensual editorial metadata and yet be able to express their own vision of the world by adapting it locally. In the following sections, we describe the nature of editorial metadata managed by the Editorial Information Manager, the choice made for music title identification and describe the architectural issues raised by the management of private and shared information.

3. Editorial metadata in the music browser

The Music Browser aims to exploit all possible metadata that can be extracted or accumulated for music titles. The architecture we present here is focused on the management of editorial data described in sections 3.2 and 3.3. Therefore we will focus on the editorial metadata manager integrated in the music browser.

Fig. 3. The Music Browser query panel.

3.1 The music browser

The kind of editorial metadata we are interested in the Music Browser is that which can be used readily for searching music. More precisely, our editorial metadata appears directly under the form of search fields that can be used in the browser. Figure 3 shows the query panel of the Music Browser in which several kinds of editorial metadata information are displayed and can be used to issue musical queries.

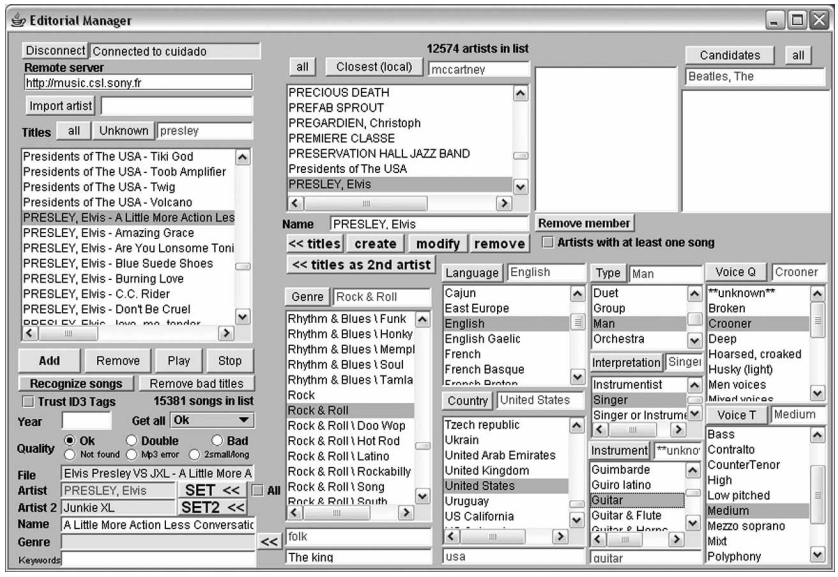

Fig. 4. The editorial data management panel: ''A Little Less Conversation'' has artist 1 ''Elvis Presley'' and artist 2 ''JunkieXL''.

3.2 Editorial information about titles

As can be seen in Figure 4, editorial information is managed with a specific tool proposing choice lists for each property. Concerning music titles, our tool enables basic editions as title name or keywords, as well as less obvious features such as title genre and primary and secondary artist. The notion of primary and secondary artist has been introduced to represent the various degrees of association between artists and music titles in a generic way: what is important for a musical query system is not necessarily to make the distinction between all possible roles of artists (composer, performer, conductor, remixer, etc.), but to propose a simple indexing scheme. In all cases, the Editorial Information Manager proposes a unified view of artists links to songs as ''primary'' and ''secondary''.

These notions of primary and secondary have different significations according to the context. In popular music, performers are usually put forward for identifying music (e.g., ''With a little help from my friends'' by ''Joe Cocker'') and composers come last (e.g., ''With a little prayer'' sung by ''Aretha Franklin'' (primary artist) is in fact composed by ''Burt Bacharach'' (secondary artist)). In classical music, the distinction is inverse (e.g., the Opera Rinaldo is primarily identified with Haendel (composer); a user may want to access a particular recording of this opera by conductor René Jacobs (secondary artist)). In another context, some remixes of songs can be identified primarily by the remixer (e.g., the recent remix of the song ''A little less conversation'' is primarily identified with JunkieXL (also known as the Amsterdam-based remixer Tom Holkenborg)). In second approximation, this song is an Elvis Presley song (i.e., usually performed by him).

3.3 Editorial information about artists

On top of the artist metadata already described in section 2.2, the Editorial Information Manager adds some content features deemed useful for browsing and not present in any existing editorial server:

- Type: Michael Jackson is a *singer*, while the Beatles are a *band*. Elvis Presley is a *singer* and a *musician*, while JunkieXL is a *DJ* and a *remixer*.
- Interpretation: The Beatles have mainly recorded *music with vocals*, while John Coltrane has mainly played *instrumental music*.
- Voice quality: Frank Sinatra is a *crooner*, while Janis Joplin has a *broken voice*.
- Voice range: Barry White has a *bass* range, while Britney Spears has a *soprano* range.
- Language: The Beatles sing in *English*.
- Keywords: Any other relevant information such as ''*1975 live version*'', ''*remix*'' or ''*birthday song*''.

Moreover the Editorial Manager proposes some semantic information about artists. For instance, many artists belong to bands: Paul McCartney belongs to The Beatles, Phil Collins to Genesis, and so on. This information is not only useful for administration purposes, but can also readily be used for browsing. We introduced the ''memberOf'' predicate in the Editorial Database. Figure 5 shows an example of the use of this information.

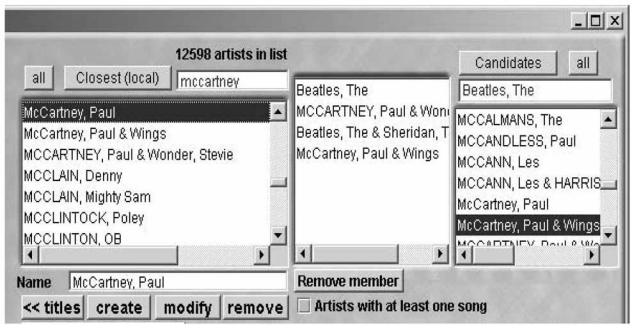

Fig. 5. The "member_of" predicate.

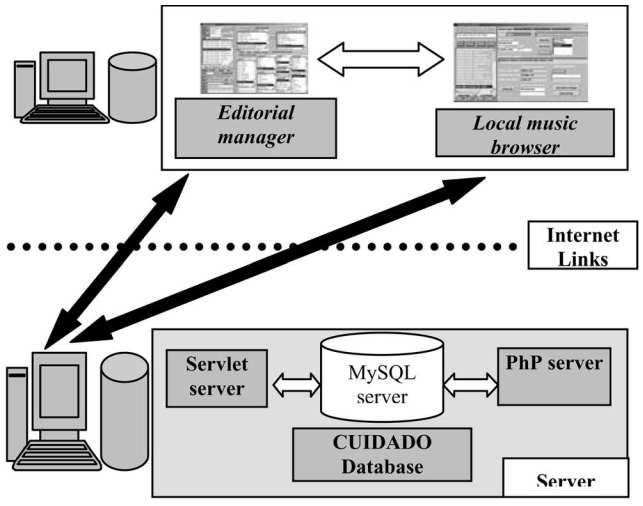

Fig. 6. Editorial data management architecture.

4. Architecture

This section describes the software and client-server architecture underlying the management of editorial information in the Music Browser.

4.1 Architecture of the editorial information manager

As shown in Figure 6, the CUIDADO metadatabase is a MySQL database hosted on a SQL server. The server acts as a server both for Php scripts and servlets. The MusicBrowser is implemented in Java and communicates with the MySQL database using JDBC drivers. The editorial metadata server runs a Php server accessible over the Internet. Specific Php scripts allow client applications to fetch and submit editorial metadata using this server.

However, Php scripts are not efficient enough to handle a variety of operations – in particular, operations requiring large amounts of information to be loaded into memory. To address this issue, the server includes a servlet server that is able to load precompiled information in memory (typically the list of artist and title names) to speed up operations like approximate string matching algorithms. Note that such an architecture uses only free and standard middleware components. The music browser, as well as our architecture, runs on Windows, Mac and Linux machines in a transparent way.

4.2 The MCM application interface

Following our experiments with content-based interactive music systems, we have started developing a more general application interface (API) – the so-called "MCM" (Multimedia Content Management) – on which the implementation of the client-side Music Browser relies. MCM is a set of java classes that offer the following data structures and functionalities:

- Multimedia *itemTypes,* which describe the entities with which we are dealing (e.g., songs or artists).
- *Items* of a given itemType (e.g., "Yesterday" is an item of the itemType "song"; "The Beatles" is an item of the itemType "artist") that exist synchronously both in the database and in memory.
- *Fields* or metadata for each of these ItemTypes (e.g., a song's tempo or an artist's name). Fields have a value for each item. The values are read/written in the database, and can be cached in memory for applications which require more processor power.
- Items may link to one another (e.g., song items can be associated with artist items). These associations are treated like fields: the "song" itemType has a *Reference* field to the "artist" itemType. A reference field's value is the corresponding item (e.g., the value of the "artist" field of the song item "Yesterday" is the artist item "The Beatles").
- Fields may be *computable* – that is, their value is the output of an extractor, either computed online or offline, in batch mode. Fields corresponding to editorial metadata are per definition non-computable. Computable fields typically represent acoustic metadata that can be computed from the audio signal (e.g., length, tempo, pitch, etc.). However, there is no restriction to the kind of information computable fields can operate on: this could as well be text, web data, and so on.
- Items can link to other items with *relations* (e.g., cultural similarity, or memberOf predicates).
- Items, fields, relations can be added (e.g., add a new directory of mp3s into the Browser or add a third-party extractor), updated, retrieved or deleted from the database.

Using MCM, all the architectural difficulties of creating databases, synchronizing data, calling extractors are hidden out. Applications like the Music Browser can be

developed very quickly by concentrating only on meaningful, higher-level concepts.

In MCM, editorial metadata can take the form of a TextField (a subclass of MCMField) of a given ItemType. For instance, the "titleName" is a TextField of the ItemType "song", and can have value (e.g., "Yesterday" or "Piano Sonata No.4 E-flat Major"). To prevent duplication problems, metadata that take their value in a taxonomy (e.g., genre, country) are implemented as ReferenceFields to an ItemType. For instance, the "genre" field of the ItemType "song" is a reference to an item of the itemType "genre" (e.g., "pop"). In turn, items of the itemType "genre" can receive further metadata/fields like most prototypical artist (e.g., "The Beatles", for genre = pop).

Using this representation, the structure of a metadatabase can be abstracted to the list of its available MCM ItemTypes and Fields while the actual values of the metadata are encapsulated in the Items. We will use the MCM representation in the following sections to describe the problems raised by editorial metadata management, as well as the experiments we made with the MusicBrowser system.

5. Synchronization techniques

In the context of *ad hoc* and local-network-based communities, users want both to share metadata and manage metadata of their own. This situation raises a key issue: the synchronization of the data.

5.1 Music identification

One common element of every EMD system is a front-end able to link musical files, either available on the user's devices or online, to the metadata describing the corresponding music object (i.e., to a fully recognized "song" item, with its associated field values). Although this identification stage is independent of the management strategy of the whole system and may be found either in the peer-to-peer, universalist or *ad hoc* approach, it is nevertheless an essential component of the metadata management chain. In this section, we describe the choice made in the Music Browser.

5.1.1 Content-based identification

Identification can be done in a blind way simply by analyzing the music signal. Over the past few years, there has been a great deal of academic and industrial efforts concerning this technique, usually referred to as "Audio/Music Fingerprinting" or "Hashing". The general idea is to extract a very compact representation of the music signal, its *signature* or *fingerprint*, and compare it to a database of already extracted and identified signatures. The signatures should be very robust to noise so that many distorted/compressed/broadcast/variously encoded instances of the same music title can be matched to one unique entry in the database. Signatures should also be compact so that matching one test signature against a huge database can be done in feasible time (as an example, it is claimed that the AudibleMagic database can identify over 3.5 million recorded songs with new content added daily; source : http://www.audiblemagic.com). Different indexing schemes and search algorithms are then used to match the extracted signature against the database.

The reported performances of the various identification algorithms are all very good, usually in the top 1% using realistic levels of noise and distortion. This makes these technologies well-suited for many commercial applications. The business model used by Moodlogics (www.moodlogic.com), ID3Man (www.id3man.com; with its fingerprinting technology Auditude: www.auditude.com), MusicBrainz (www.musicbrainz.org; with Relatable: www.relatable.com), Tuneprint (www.tuneprint.com), GraceNote (www.gracenote.com; integrated into Apple's iTune and mp3 walkman iPod) allows users to link their personal music files to metadata that has been gathered on a server by the provider. One common extension of this is to automatically fix the ID3 tags of the user's mp3, or even to rename the files themselves with their correct title and artist name as identified from the database. Other commercial, much advertized applications of fingerprinting technologies are Broadcast Monitoring (Nielsen: www.bdsonline.com; Audible Magic: www.audiblemagic.com; Yacast: www.yacast.com), filtering technology for file sharing (e.g., preventing copyrighted files being exchanged in Napster) or "Name that tune" applications on mobile phones (Shazam: www.shazamentertainment.com) or digital radio on personal computers (Clango: www.clango.com).

5.1.2 Using external information

Another way to identify music files and link them to metadata consists of using external information on the titles when available. For instance, Sony's Emarker system (discontinued in September 2001; see Bricklin, 2001) was used to exploit the geographical and temporal location of a radio listener requesting a song by querying a large database containing all radio station programs by time and location. The approach is much lighter than the signal-based one since no signal processing is required and it can scale-up to recognize virtually any number of titles. It works, of course, only for titles played on official radio stations.

External information can be as simple as file names, with the difficulty that names are even less standardized. For instance, consider a file name like:

D:\mp3\CSL2-9\Various – RockFM \Original Rock – 5 - Crack The World Ltd - Fine Young Cannibals - She Drives Me Crazy.mp3.

To start with, which section is the artist name? ''Rock Fm'', ''Original Rock'', ''Fine Young Cannibals'' or ''She drives me crazy''?

In Pachet and Laigre (2001), we proposed a heuristic-based parsing system to exploit the information possibly contained in the filename itself. We have studied large corpora of files, whose names are decided by humans without particular constraints other than readability, and have drawn various hypotheses concerning the natural syntaxes that emerge from these corpora. A central hypothesis is the local syntactic consistency, which claims that filename syntaxes, whatever they are, are locally consistent within clusters of related music files. These heuristics allow the parsing of filenames successfully without knowing their syntax *a priori* using statistical measures on clusters of files rather than on parsing files on a strict individual basis.

It is impossible for an automatic system to parse a filename like the one given above. However, we can observe that in the same directory, there are many filenames having the same syntax

$$a - 0 - b - c - d,$$

where a, b, c, d are strings and 0 is a number (''|'' indicates separation between recognized segments):

Original Rock | 1 | Crack The World Ltd | Animals | The House of the Rising Sun |
Original Rock | 2 | Crack The World Ltd | The Moody Blues | Nights in White Satin |
Original Rock | 3 | Crack The World Ltd | The Beatles | Mister Moonlite |
Original Rock | 4 | Crack The World Ltd | The Beatles | Ain't She Sweet |
Original Rock | 5 | Crack The World Ltd | Fine Young Cannibals | She Drives Me Crazy |
Original Rock | 6 | Crack The World Ltd | Fine Young Cannibals | Good Thing |

We then look at the statistics on the different sections: a and b are always the same (''Original Rock'' and ''Crack the World Ltd''); 0 is incrementing; there are several different d's for each c (''Fine Young Cannibals – She Drives Me Crazy'', ''Fine Young Cannibals – Good Thing''); there are usually more words in d than in c; and so on. From all these statistics and with a few appropriate heuristics, the algorithm is able to infer that, for example, c is the artist field and d is the song title field. Experiments in Pachet and Laigre (2001) showed that the parsing error with this algorithm is below 5%, which compares with the recognition rates achieved by fingerprinting techniques. This second approach was used in the Music Browser and is now integrated into the MCM API.

Audio fingerprinting is well suited to the universalist approach, in which it is considered implicitly that the collection of titles is finite and shared by all. In our context, we target communities of users who do not necessarily access files that are sufficiently well known to be included in the signature databases. Furthermore, communities may wish to specialize in specific musical areas, including sharing metadata on music titles that are not produced by majors. Finally, we deemed that maintaining very large databases of fingerprints was not suitable on small devices aimed at local or *ad hoc* networks.

5.2 Synchronization between recognized items

In a given metadatabase, items are uniquely identified with an id or hashcode. For instance, ''The Beatles'' may be the artist item #34 in a database (A) and the item #1546 in another (B). In the next section, we will present an architecture that enforces common indexes via a central id server while maintaining full local flexibility. However, there are situations where, when sharing data between 2 arbitrary databases, we cannot presume that the ids of equivalent items are the same. The MCM API has a provision (through the Comparator interface) for a number of matching algorithms able to uniquely identify an item from a database (A) as a corresponding item of the same ItemType in another database (B).

The available implementations so far are:

- A regular expression comparator that identifies as an exact match for string *s* any string *t* that entirely contains *s*. This is currently used to synchronize genre taxons (e.g., ''Rock'' is matched to ''Rock and Roll'').
- A heuristic-based artist matcher that copes with the various artist syntaxes we encountered in Pachet and Laigre (2001): ''The Beatles'' is uniquely matched with ''Beatles'' and ''Beatles, The''; ''Paul McCartney'' is uniquely matched with ''McCartney, Paul'' and ''McCartney (Paul)''; and so on.
- An acronym-based comparator used for countries: ''United States of America'' is matched with ''USA'' and ''the USA''.

In the cases where an exact match cannot be found, an edit-distance comparator (see, e.g., Crochemore & Rytter, 1994) is able to rank all approximate matches in similarity order so as to assist the decision process. For instance, if the system cannot find a match for the artist item ''beetles'' in database A, it will suggest a list of close matches, among which will appear ''The Beatles''.

5.3 Synchronization between different structures

Adding another degree of complexity, one might want to synchronize two meta-databases that have different structures (i.e., different ItemTypes and Fields). It may be that the same objects have received different names; for instance, the ''song'' itemType of database A may be referred to as ''*chanson*'' (the French word for ''song'') in database B, or the Field ''bpm'' in database A may be called ''tempo'' elsewhere. In certain cases, the whole structure may even be different; for instance, the songs' ''artist'' field in database A is a ReferenceField to an ''artist'' item, while it is a simple TextField in database B. A fully automatic translation between such different schemes is not realistic. However, MCM has a built-in tool to assist such conversions. Figure 7 shows the process of translating the ''filename'' field of an itemType ''song'' into the French ''*nom de fichier*'' of the corresponding ItemType ''*chanson*''. Once a symbolic link is made between the two metadata, values can be transferred automatically.

The process can be further assisted by providing a list of alternative possible names for each itemType through the use of Comparators similar to those discussed in section 5.2. For instance, the system may know that ''artist'' is likely to be equivalent to any of the following identifiers: ''musician'', ''composer'', ''performer'', ''band'', ''singer'', ''*musicien*''. In fact, Figure 7 illustrates a further special case where one of the two databases is not a MCM one, but an arbitrary SQL database (as can be found, e.g., in a third-party EMD system), where one first has to examine non-described tables and columns and look for clues of the underlying concepts and semantic relations between them.

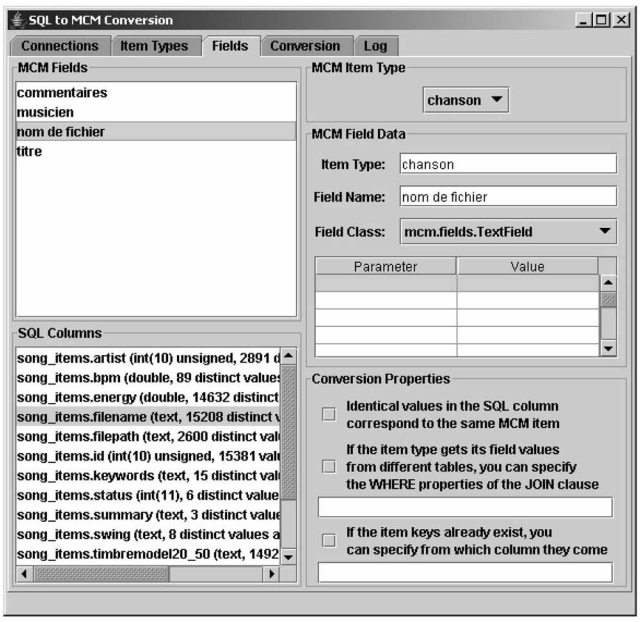

Fig. 7. The MCM conversion tool to synchronize two meta-databases with different structures.

6. Sharing scenarios and topologies

With the apparition of *ad hoc* networks, single or multiple users can share their data easily and in a transparent way. This situation raises a key issue: the management and synchronization of the data. How can users keep their database up-to-date while benefiting from new entries without degrading their customized databases?

6.1 Open topology

For some simple sharing scenarios, it is possible to envision that each client (or rather: peer) runs his or her own metadatabase server, each with its own naming conventions, database structures, ontologies, and so on. When two such nodes are within communication range of each other, they form a temporary, *ad hoc* network and may share some metadata.

Scenario: Import a genre taxonomy. It is a well known fact that music genre taxonomies are largely syntactically inconsistent and non-consensual. However, they do provide a lot of useful information for browsing, and users have the ability to very quickly navigate between different, even orthogonal taxonomies (Aucouturier & Pachet, 2003). As envisioned in the introduction to this article, user A may want to import in his or her local metadatabase the genre taxonomy used by user B. We suppose here that, like in the MusicBrowser, there exists a ''genre'' metadata that is a ReferenceField of the itemType ''artist'' linking to a ''Genre'' ItemType, which has only one TextField: its ''name''. As mentioned before, this is only one example of a metadata scheme from among the very many enabled by an API like MCM. For instance, one could design a system where artists can have several genres (as it is likely to change over a career), or where genre is a per-song or per-album metadata. Efficiency arguments in the context of massive commercial music databases may call for a (over)simplified metadata scheme, while expert, niche communities (e.g., jazz lovers) may need specialized schemes such as different types of artists for each song (composer, performer, soloist, rhythm section, sound engineer, etc.). The very purpose of an API like MCM is to enable the user to design his or her own scheme, while ensuring that the various metadata management mechanisms presented in this article still work.

The process of importing the genre metadata into user A's local database is summarized here:

- Synchronize the structures: match the corresponding ''artist'' and ''genre'' itemtypes, as well as the Fields, as described in section 5.3.
- Create a new genre itemType (e.g., ''Reykjavik-Genre'') in A.
- Download all the ''genre'' items (i.e., instances of the genre itemType: Rock, Pop, Jazz, etc.), and keep a correspondence table of the ids in A and B (which may be different).
- Synchronize the ''artist'' items (i.e., instances of the artist ItemType): uniquely match each artist in A into B's list of artists using a Comparator as described in section 5.2.
- For each artist item in A that also appears in B, download the corresponding genre value (a genre index in B's taxonomy) and convert it to an index in A's newly downloaded taxonomy.

While the whole chain is made possible by the various MCM services introduced in this article, we would still remark that this process may become unrealistic in practice. As shown earlier, the various synchronization processes in use here are still prone to error and are not yet fully automatic, which makes the processing of large databases tedious. Furthermore, the fact that we cannot rely on unique indexes for the same objects in both databases makes even more difficult the sharing of more complex metadata structures like the memberOf predicate or similarity relations that link the ids of several artists.

6.2. Hybrid topology

We propose an architecture for the Music Browser that addresses this problem by relying on a central server architecture (Figure 8), while still offering the flexibility of local customization. In our architecture, community users all work on a community server, itself synchronized with the central server. The music browser is installed on each user's computer and used as a front-end to create/modify metadata. The architecture is based on two main operations: *update* and *infer*, which are described in the next section.

6.2.1 Updating metadata (client side)

Scenario: Adding new songs or artists. Users can add new songs and/or artists to their local database using the data management tool (see Figure 4). Users choose song files (e.g., mp3, wav) or enter artist name manually and the corresponding file/artist names are automatically analyzed. As described in section 5, we use a parsing mechanism to automatically recognize artists and songs names. Using Comparators, the client metadata manager looks for artists and songs in the local database as well as in the central server database. Three results are possible:

- If the song and/or artist exists on the central server, the user is offered a list of closest matching artists and/or songs and can link his or her new entry with the chosen one while all metadata are imported. This process ensures that every database shares the same artist and song indexes to avoid compatibility problems. If Michael Jackson is referenced as artist #98 and as a memberOf Jackson5 referenced as artist #10, then every local database must use these same indexes. See reference 1 in Figure 9.
- If the song and/or artist already exists in the user's local database, then the new entry can be removed (to avoid duplication), marked as double, or as a new song and/or artist (e.g., for a live version or homonym artists). See reference 2 in Figure 9.
- If the song and/or artist does not exist at all, users create all metadata using the management tool. These data are stored on the local server and broadcast to

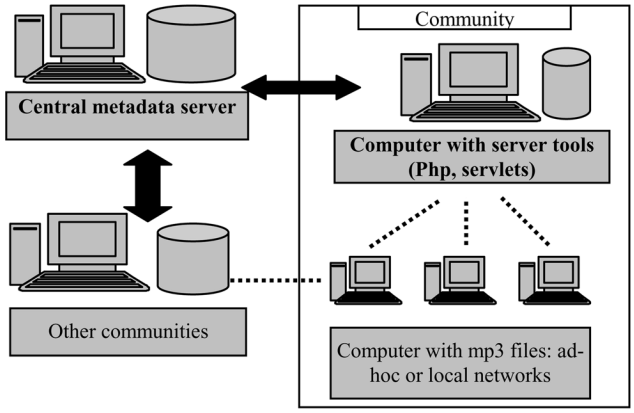

Fig. 8. Interaction between systems.

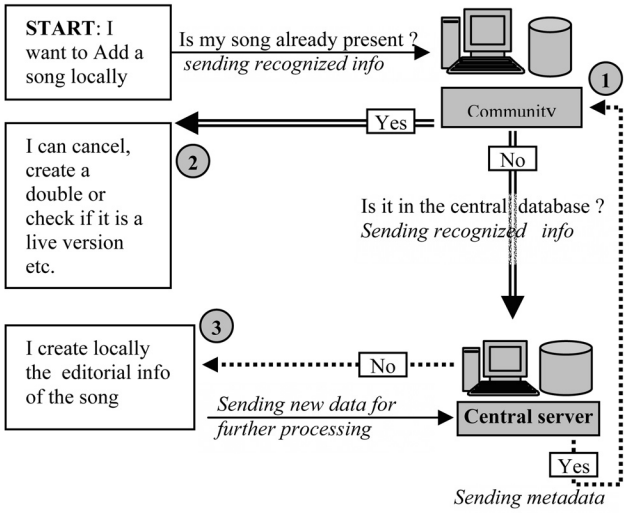

Fig. 9. Adding and importing new data.

the central server for further processing. See reference 3 in Figure 9.

Scenario: Adding new metadata. Music is constantly evolving and no system could reasonably forecast everything (Datta, 2002). Furthermore, as communities can run their own metadata servers, they will most probably want to tune them to create new fields or simply add a new musical genre. The Editorial Manager includes such a feature (i.e., the ability to update the database structure itself): users can update their database structure to evolve it. As for songs and artists, such modifications are broadcast to the central server for further processing.

Scenario: Synchronizing the local metadatabase. It can happen that a song or an artist in the central server is created or modified. The metadata manager has the ability to synchronize local metadata with the shared metadata of the server. When required, users can choose to update part or all of the local metadata. The same mechanism is available for the database structure.

6.2.2 The infer process

When new data are submitted to the central server, they need to be integrated. We call this the "infer" process. The integration can be unconstrained and accumulative. Each contribution is accepted in a non-destructive way: conflicting entries (e.g., genre = "Rock" and genre = "Pop" for the same artist item: "The Beatles") are duplicated with information about the contributing user, date and so on. This scheme very much resembles existing democratized web publishing models like weblogs (Blogs) or the Portland Pattern Repository (Wiki: http://c2.com/cgi/wiki?) (see Figure 10).

In the current architecture, the infer process is rather envisaged in a collaborative filtering way (Sarwar et al., 2001). Data are stored and regularly analyzed by the central server. The emergence of consensus enables the consolidation of new entries. This process is performed automatically to avoid manual moderation, which is a time-consuming process. Once a week, metadata are updated on the central server.

When a community user performs a complete synchronization, all local data are updated. For each community server using the central one, at least for the basic indexes, compatibility is ensured. However, there is always the opportunity to refuse updates, new entries may be considered non-relevant for the community, and so on. As in Musicbrainz, the central server will benefit from users entries (although the MusicBrowser already performs pretty well as stand-alone software to manage large collection of music files).

Gathering data being a key issue for most metadata systems, we believe that community vision can represent an interesting new approach. Shared among specific music genre specialists (people involved in "East Coast Rap" or in "Intelligent Techno" using an *ad hoc* network), a database can quickly become highly specialized with a limited number of users. Members will probably be keener to add and consolidate entries if they

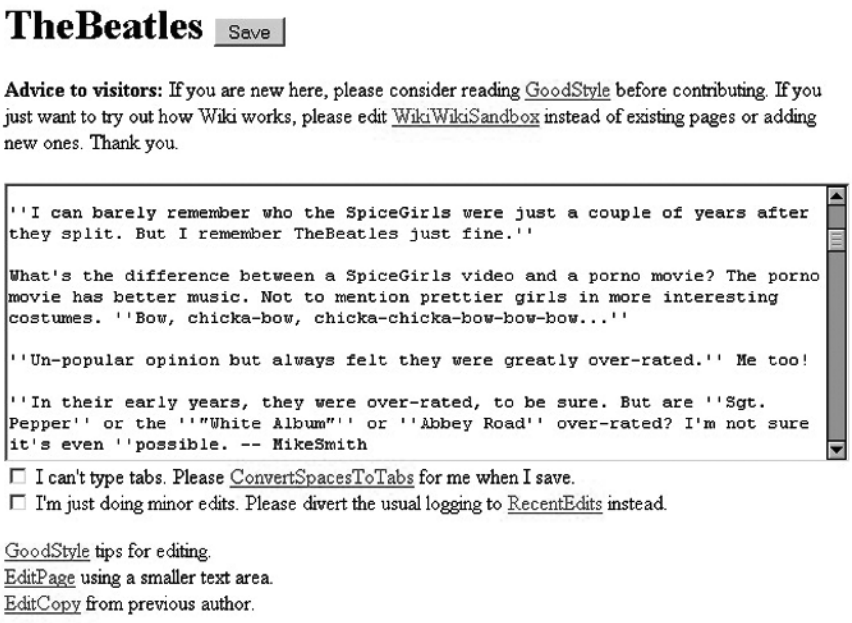

Fig. 10. Web interface to contribute to the "Beatles" entry in the Portland Pattern Repository.

see an immediate benefit for their community. This community then shares their data with the central database server without degrading their data and without necessarily opening their database to everyone.

7. Conclusion and future work

In the context of *ad hoc* and local-network-based communities, users want to share metadata and manage metadata of their own. We have presented an architecture for managing musical editorial metadata that allows client applications to exploit shared metadata when available, as well as creating and managing local, private information. This architecture is based on two basic principles: an update mechanism, which warns the central database of any local modifications, and an infer mechanism, which computes emerging, consensual values from user inputs. The resulting architecture provides greater flexibility in editorial metadata management for electronic music distribution systems. However, the synchronization of the database structure is a difficult task and is still only half automatic.

This system is a first step in the direction of hybrid metadata systems in the sense that it lies between the two extremes of the universalist and isolationist approaches. Current work focuses on extending this paradigm to include other forms of metadata – in particular, acoustic metadata computed from the audio signal, as well as musical similarity relations computed from data mining techniques and enhancement of the synchronization processes. Finally, user experiments are in progress to assess the robustness of our approach in the context of real-world network environments.

Acknowledgements

The Music Browser project originated and has been conducted inside Sony Computer Science Laboratories in the context of the Cuidado and SemanticHifi IST projects. The Cuidado project (Sarwar et al., 2001), entitled "Content-based Unified Interfaces and Descriptions for Audio/Music Databases available Online", ran from January 2001 to December 2003. The project is currently being continued in the context of the SemanticHifi project (Vinet et al., 2002) started in December 2003.

References

Aucouturier, J.-J. & Pachet, F. (2003). Musical genre: A survey. *Journal of New Music Research*, *32*(1), 83–93.

Axelsson, F. & Östergren, M. (2002). *SoundPryer: Joint music listening on the road*. Paper presented at UBICOMP'02.

Basagni, S., Conti, M., Giordano, S. & Stojmenovic, I. (Eds) (2004). *Mobile ad hoc networking*. Wiley-IEEE Press.

Bassoli, A., Moore, J. & Agamamnolis, S. (2003). *TunA: Local music sharing with handheld Wi-Fi devices*. Paper presented at the 5th Wireless World Conference, Surrey, UK, July.

Bricklin, D. (2001). *Sony eMarker: How a clever system works*. Available online at: www.bricklin.com/emarker.htm.

Crochemore, M. & Rytter, W. (1994). *Text algorithms*. Oxford: Oxford University Press.

Datta, D. (2002). *Managing metadata*. Paper presented at the 3rd International Conference on Music Information Retrieval (ISMIR 2002), Paris, France, October.

Herrera, P, Serra, X. & Peeters, G. (1999). *Audio descriptors and descriptors schemes in the context of MPEG-7*. Paper presented at the ICMC, Beijing, October.

Pachet, F. & Laigre, D. (2001). *A naturalistic approach to music file name analysis*. Paper presented at the 2nd International Symposium on Music Information Retrieval, Bloomington, IN, October.

Pachet, F., La Burthe, A., Aucouturier, J.-J. & Zils, A. (2004). The Sony Popular Music Browser. *Journal of the American Society for Information (JASIS)* (special issue on Music Information Retrieval), *35*(12), 1037–1044.

Sarwar, B., Karypis, G., Konstan, J. & Riedl, J. (2001). *Item-based collaborative filtering recommendation algorithms*. Paper presented at the 10th World Wide Web International Conference, Hong Kong, 1–5 May.

Vinet, H., Herrera P. & Pachet, F. (2002). *The CUIDADO Project: New applications based on audio and music content description*. Paper presented at the International Computer Music Conference, Göteborg, Sweden, September.