Music Tagging Type Definitions,
Systems for Music Representation and Retrieval

Carola Boehm
University of Glasgow
C.Boehm@music.gla.ac.uk

Donald MacLellan
University of Glasgow
D.MacLellan@music.gla.ac.uk

Abstract

The paper will discuss the general issues surrounding structured music information representation and music information retrieval, present the result of the UK-JISC funded "proof-of-concept" project: MuTaTeD! [1] (Music Tagging Type Definition) and inform about further development within the UK-Library Information Commission funded MuTaTeD'II [2] project. Work within the two projects MuTaTeD! and MuTaTeD'II, striving towards the design and implementation of an expandable, flexible music information retrieval system with delivery/access services for encoded music, has resulted in involvement in standards development, such as MPEG7, in order to support the needs of developers within music information retrieval research.

1. Introduction

A serious problem for IT today concerns Information Overload. When, as now, the internet is becoming the platform, the browsers become the operating system, and applications become services [3], services to access music content have to define new methods of storing and distributing time-based data if they are to serve large quantities of high-quality information across wide area networks (WANs). Future information services will need to carry the burden of extensive metadata management, - content searching and manipulation of time-based data - if they are to facilitate intelligent access and efficient delivery to their user communities.

It is now, therefore, quite widely accepted that there needs to be development in three areas (i) *Information Structure* (ii) *Information Representation and* (iii) *Information Access.*

In a number of IT sectors, there has been considerable progress in user interfaces, metadata management systems for access, information retrieval, workflow management, and similar areas. But in the area of Music the most widely encountered music information structure is midi and visual information representation over WANs is still largely restricted to static images. Music information retrieval is still in its infancy.

While this leaves plenty of scope for researchers, there is a sense that users have still to benefit as fully within the area of Music as they clearly have in other domains.

2. The Music-Specific Context

Accessing and manipulating time-based data over wide-area networks is a research area ready to deliver its first results. For a variety of tasks, such as automatic page-turning and slide-shows, and within a variety of fields, such as multi-media applications and professional video or media companies, there is a requirement for a language that describes time-based media in its structure.

Two main options seem to be possible here:

- handle the media files as they are and synchronize their interaction with each other, or
- use a structured language to represent time-based media files, such as those associated with music, as a time-based medium within its own time-dependent structure.

The first option has yielded solutions that have already been commercially exploited, e.g. interaction of sound and movies, parallel depiction of scrolling text and movies, web slide-sound shows, etc. The second option has so far been neglected, so that the Music community is left without the musical equivalents of, for example, SGML (Standard Generalized Markup Language) and XML (eXtensible Markup Language) and packages to handle them.

"Music Representation Languages" are essentially a means to describe certain musical elements and their time-relation to each other. At the macro level, such a language might represent a sequence of four sound files

played one after the other, corresponding to the four movements in a symphony. At the micro level, it might represent single notes on several staffs in their relative "time-positions". Thus, within this model of a "time-based media structure", by heightening the detail of granularity, we conjoin the two options outlined above.

An application with the combined implementation of two standards, SMDL (Standard Music Description Language, SMDL ISO/IEC 10744) and NIFF (Notation Interchange File Format, created 1995 in collaboration with several music companies.), would support the representation of music as a structured, time-based entity in its own gestalt, - while simultaneously supporting high-quality display via the NIFF format. The publication of the SMDL standard draft in 1995 was recognized in the NIFF specification of the same year when it proposed this very combination for applications. Among applications which might be founded on this concept are:

- Platform independent file interchange (music software, web)
- Intelligent and dynamic access to structured music (libraries, education)
- Using music-recognition functionality, easy and instant access to a critical mass of music (sheet music sales and loans, music publishers, performers, music and audio distributors)
- Collaborative creation and working processes and the synchronization of time-based media of the same or different type (media production industry)

As mentioned above, in order to achieve this, the existence and acceptance of a language that describes time-based media in its time-based structure is required and with it the possibility to represent it in a musical way. The development and adoption of such a language as we envisage will make possible the realization of these objectives.

The basic idea of the MuTaTeD! Project (Music Tagging Type Definition) grew out of these considerations and suggested the following primary project goals:

- the proof of concept of integrating two existing standards, one for music content (SMDL) and one for music score representation (NIFF)
- the use of a standard Meta-DTD for music-tagging languages

3. Music Representation Standards – The Historical Context

Research into Music Representation Standards is not new. There have been many and varied approaches to standardised music representation. The following, historically oriented listing helps show the context for our current efforts. [4]

Early Number Notation (19th century and earlier): Davantes, Kircher, Avella, Mersenne, Bontempi, Stierlein, Mine, Schulz, Rousseau, Galin-Paris-Cheve, Geisler, Teule, Gatting
19th Century Shorthand: Sauveur, la Salle, Romanos, World Music Language
Formats for Typewriter and Early Computer Age: DARMS, Plain and Easy and ALMA, WMN, MECOS, ANTOC, SPECO, NUTOC, Essen EsAC
Computer Age: Formats for Notation Software - Binary File Formats: Encore File Format, Syqyest, Lime's Tilia, Nightingale, Score's Score,
Computer Age: Formats for Composition/Performance Systems: Midi, Mode's Small-Music, Kyma's Kyma, Common (Lisp) Music Notation, Max Score, Csound's Score, NeXT's Score format, Radio Baton Score File, Hush/Hymne/DejaVu, MML
Computer Age: Formats for Notation - Text based File Formats: Finale's Enigma, MusicTex, Lilyponds Mudela, abc, GUIDO,
Computer Age: Interchange File Formats: NIFF
Computer Age: Analysis and Content Serching: Humdrum's Kern, MuseData, CPN,
WebAge: Formats for Storage, Compression, Delivery: MPEG4, SMIL
Web Age: Tagged Music Description Formats: SMDL/SGML, MuXML, MusicML, MNML, MPEG7

Independently of (and in parallel with) "conventional" music notation, numerically based systems for encoding musical information have been devised, perhaps governed by an idealistic Pythagorean sense [5] that numerical abstraction and elegance was especially appropriate. But they were also to fulfil a practical need, at a time predating audio recording, to capture musical data as it was being played – a task for which common music notation is not at all well suited.

Using typewritten text and early computers, substantial archives of incipits and melodies were developed, giving rise to an ASCII-based notation. These holdings may still be found in many catalogues of music. With the development of professionally oriented music applications for the computer, new data structures, file formats, and music description standards emerged with almost with every application. The Music Notation software applications, in particular, were forced to create their own underlying music data structures to contain all the information needed to print music. In the absence of established and widely adopted standards, this was inevitable if the applications were to be of any practical use. Surprisingly, perhaps, there was not much impetus to agree a joint standard.

NIFF (Notation Interchange File Format) was the first (and only) commercially-based effort by several companies to agree on an interchange file format for Notation. Although there are now a few NIFF-compliant programs, all the most widely used notation and music programs are still not capable of reading NIFF. Perceptions of their commercial interest appear to lie behind various music companies' failure to follow through with previously announced plans for NIFF implementation. This is difficult for the Music user community to bear, forced as it is to struggle on with MIDI, which cannot and indeed was never intended to support the full extent of the needs of the field.

Recent activity to revive NIFF-related development is largely to the credit of the user community rather than commercial concerns. A positive development can be seen with music tagging languages. The number of text-based music languages, cognate with SGML/XML/ SMDL, is rising. A number of groups are pushing for standardisation of a single language to be used over the web. There are developments spinning off from XML under the W3C, and developments with SMDL (Standard Music Description Language), backed by MuTaTeD! and their work within MPEG7 [6]. As ever, agreement is required so that application developers may safely integrate the standard in their application. Even though SMDL unlike all the other tagging music standards is an ISO standard, it seems so far to have failed the final test of acceptance by application developers.

It is hard to understand why it is, that until now no music representation standard has been accepted. A list of problems of developers of music representation standards, which could explain this absence of accepted standards would encompass difficulties of collaborative development efforts as well as difficulties in the intellectual effort itself. The creation of a music standards will always involve the problematic issue of design and implementation of time and granularity, representation, and language functionality.

3.1. Time

The most relevant content based problem which is faced by any developer of a music representation language is the definition of time.

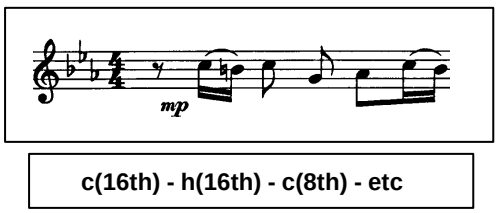

Fig. 1 Time as an internal attribute

Decisions have to be made in the design of a language to either define time as an internal attribute of the musical entities (Fig.1) or define time as an external entity with links between musical objects? (Fig.2)

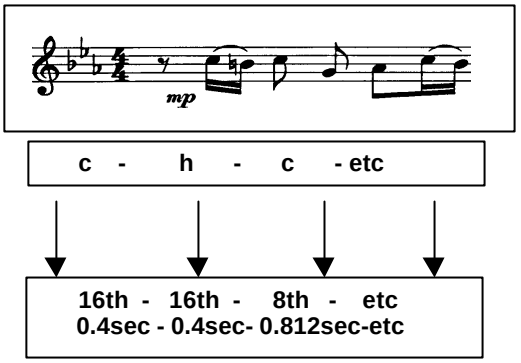

Fig. 2 Time as an external attribute with links to the musical objects

Other possible differences in design can be the choice of a recursive (Fig. 3)

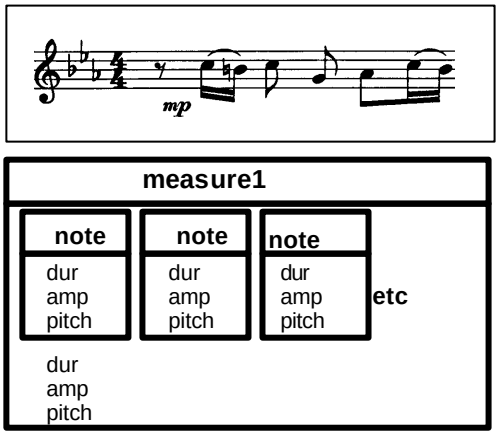

Fig. 3 Time as a recursive entity?

or a sequential representation of time. (Fig. 4)

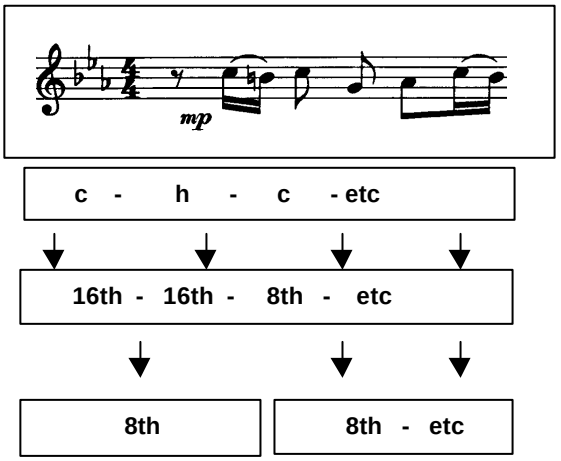

Fig. 4 Time as a sequential entity

Deciding on one on or the other model has consequences on the performance of the data structure in certain use contexts. Recursive time structures, for instance, might be easier for implementing searches, but might be slow in processing changes in speed. Having an externally defined timeline means a high performance in handling temporal changes but does imply a more complex implementation of the associated note values.

3.2. Representation paradigms and language functionality

With text, most applications already have an established design for separation of content from representation,- or "model" from "view". Examples are html, with stylesheets and SGML, storing the basic text in its "meaningful" structure with levels of headers and logic text elements as for instance "quotes", "headers level 1-10", "emphasis", etc. In music, a similar model-view separation is needed in order to create a flexible basic structure to hook up different notations on to, be it guitar riffs, common music notation, frequencies, midi, or other representations. In order to switch from one to the other, a central logic music data structure is needed to which different views can be hooked.

This general Model-View-Controller paradigm, originally deriving out of the object-oriented Smalltalk environment, could and should be one of the design goals of any music information description language.

Comparing these requirements with our very successful common western music notation as a possible language for describing musical information, one can immediately see that it has at least three major characteristics defined, which actually fit very well into a model-controller-view paradigm. It is foremost a graphic oriented language. It represents music in a very graphical manner, so much so, that other languages have raised its graphical representation to the definition of music itself by describing in order to achieve a common music notation of the music, ignoring the existence of any other means of transcribing musical information.

Additionally our common music notation is action oriented, as it often contains information of actions to be taken by the performers: how fast, how slow, how loud and with what character. In this characteristic it does not tell the absolute value of an entity but rather how it should be played, the value being derived out of the following of these action commands. There is actually long history of pure action-oriented languages, such as tabulators and guitar riffs, which do not represent the music itself, but rather the actions to be taken to achieve a sound. Midi is a further machined based example of this type, as it does not contain information about the music in itself, but rather represents a sequence of actions the instrument has to execute. The sounding music is derived out of the instruments following these directions.

Finally, our common western music notation is highly structure oriented, the user can immediately grasps the smaller and the bigger structures and other relational complexities, such as which instruments play together in a group, which notes belong together, which is the accompaniment and which the melody. In this way it is structure or logic oriented.

To achieve the same level of usability, a music description language for music information retrieval would in the best case either need to encompass all three levels of language functionality or it would need to be able to be combined with languages to fulfill them.

Having experimented with own developed model-view-controller based music data structures in the past, we realized that the barrier of acceptance was to high to overcome if a newly developed data structure, thus "yet another standard" would be used. We looked towards existing standards, which on the one hand would fulfill our own requirements of a) having a model-view separation, b) being a structure- or logic-oriented language and c) using a standard in order to have a stable base and d) preferably being a text based language for platform independence and web use.

Thus within the MuTaTeD projects it was decided to use two existing standards, SMDL and NIFF. SMDL would function as our structure oriented model of our central music data structure with NIFF providing one possible view on this structure. SMDL had the advantage of being text-based, structure-oriented and being a standard. NIFF provided us with a standardised music notation format representing the graphical information of music.

4. A Short Introduction to SMDL (Standard Music Description Language)

SMDL (Standard Music Description Language) is a HyTime application that conforms to international Standard ISO/IEC 10744. It builds upon SGML (ISO 8879) and HyTime (ISO/IEC 10744).

There are four domains in SMDL:

- logical domain - contains the abstract musical content , described as "the composer's intentions with respect to pitches, rhythms, harmonies, dynamics, tempi, articulations, accents, etc.". It contains any number of `cantus' elements.
- gestural domain - contains any number of performances, each of which specifies how and where components of the logical domains are rendered in a specific performance, as in "the information added by performers".

- visual domain - contains any number of scores, each of which specifies exactly how components of the logical domain is rendered visually, as in "the information added by human editors, engravers, and typesetters".
- analytical domain - comprised of any number of theoretical analyses.

The process of creating an SMDL document instance involves generating a logical domain from a score or a performance, and (optionally) of generating a visual or gestural domain which represents all the correspondences between that score or performance and the logical domain. The relationships between the different domains are created using hyperlinks which may connect chunks of multimedia materials as well.

SMDL is expected to be published in its revised form with compatibility with XML and HyTime2 by 2001. [7]

5. A short introduction to NIFF (Notation Interchange File Format) [8]

As Stephen Mounce writes in his introduction to NIFF:

> *"The NIFF project began in February 1994 with a meeting between technical people representing three major music notation programs and three music scanning programs. The group's goal was to define a new standard format for exchange of music notation data, which everyone agreed was long overdue in the industry.*
>
> *The original companies involved were: Passport Designs (publisher of Encore), San Andreas Press (Score), Coda Music Technology (Finale), Musitek (MidiScan) and TAP Music Systems/MusicWare (NoteScan). The list of advisors has continued to grow over time.*
>
> *In January of 1995 Coda decided to withdraw from the process (with the intention to publish their own Enigma format). Shortly thereafter, Mark of the Unicorn, Twelve Tone Systems, Opcode Systems, and TAP Music Systems/MusicWare agreed to replace Coda as financial sponsors." [9]*

Thus NIFF is the result of more than two years collaboration between major music software publishers and experts in the field of music notation/representation. NIFF files include graphical object and page layout as well as MIDI performance information.

Up to now, MIDI files have been the de facto standard for exchange of music data between programs. Although this is sufficient for playback, it is inadequate for displaying and printing of music notation. NIFF tried to answer this need with a dedicated notation score representation with its major functionality of being an interchange file format.

To sum up the positive characteristics of NIFF:

- platform-independent interchange
- intended to preserve a significant amount of visual detail and allows representation of the most common situations occurring in conventional music
- makes provision for software developers to define their own extensions to handle the more unusual situations and allows inclusion of Encapsulated PostScript (EPS) files and fonts to allow interchange of features not otherwise defined in the format
- the standard is open and non-proprietary and there are Software Developers Kits (SDKs) available

Both SMDL and NIFF seemed stable enough to start developing SMDL based and NIFF compatible services for music information retrieval. Although one may be aware of the fact that other standards are much more accepted and hyped within the music industry, and that both NIFF and SMDL are not accepted by it, they still seem to be the only standards powerful enough and close enough to being officially standardized, for developers not to be prone to the danger of working towards moving goalposts. Within this context, plans were made to set up a proof of concept project to validate the concept of integrating these two existing music representation standards.

6. MuTaTeD! – a proof of concept

Objectives of the project were: to integrate SMDL as the Model with NIFF as one possible View, and establish a standard Meta-DTD for music tagging languages, which could be used by the wide user community. The project MuTaTeD! validated the concept of integrating two existing music representation standards. Additionally, it was to research into the development and integration of a SMDL DTD for the wider music user community and develop a web-application as a demonstrator. The work heavily influenced the "Structured Music MPEG7 proposals" [10] which were proposed in order to ensure an SMDL-compliant standard.

Due to the time and financial constraints of this project an early decision was taken to use freely available software development environments. For the parser and compiler the Lex/Yacc Parser Technology (lexer, parser, code generator) was chosen and a multipass compiler has been developed with this technology. To create the NIFF output files the NIFF Software Development Kit, also freely available, was used.

In addition to this, we could utilize already existing multipass compilers, which convert SMDL to NIFF, developed by the CANTATE [11] project (developer: Steven Mounce). Including our multipass compilers, which

also make use of the NIFF SDK we had the following compiler passes1:

CANTATE's 2 Pass Compiler: NIFF --> SGML(SMDL):

- NIFF -> SGML(NIFF) and
- SGML(NIFF) -> SGML(SMDL)

MuTaTeD!'s Multi-Pass compiler: SMDL --> NIFF:

- SMDL --> SGML(SMDL),
- SGML(SMDL) --> SGML(NIFF),
- SGML(NIFF) --> C(NiffSDK) --> C-Code,
- C-Code --> NIFF binary .

The feedback from several large libraries and library projects, which have shown interest in this work, emphasized the value of having the full circle of converters from SMDL to NIFF and NIFF to SMDL available. Although the SMDL to NIFF converter as developed still has some restrictions in its functionality, it should be noted that MuTaTeD!, besides having proved the concept of integrating SMDL and NIFF, it was the very first project to build a SMDL to NIFF converter.

6.1. The technical Set-Up

The ideal situation, would have been to distribute the multipass converter across server and client whilst using a platform independent client (see Fig.5).

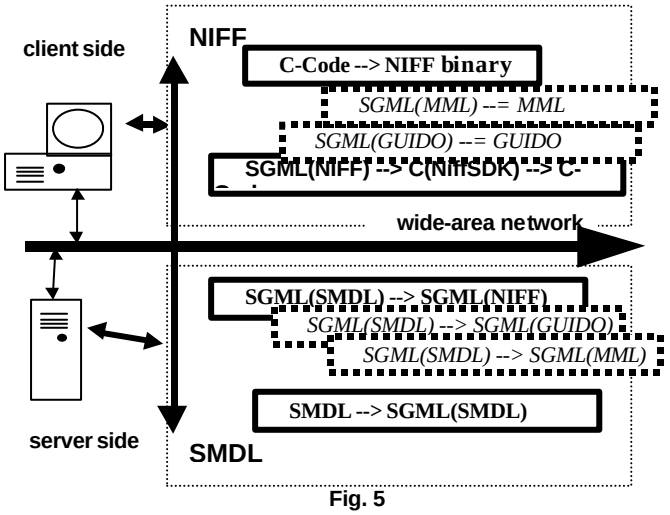

Fig. 5

This design would have had the advantage of client-server distribution in order to maximise the performance for the user. The conversion processes to the final binary NIFF files would be executed on the client machines, thus minimising the downloading time and platform dependencies. Only text-based information would be send to the client, the binary being created within the client-side application. Additionally this would have the advantage of involving separate passes, allowing the modularity to add different client side converters, such as for instance GUIDO [12]. In this way it could be easily expanded. But facing the problem that the NIFF SDK was available only for Unix, and not having the time for tedious recompilation procedures of the software tools for WinNT, the decision was made to put all of the passes onto the server side.

6.2. The MuTaTeD! Multipass Compilers

As with any compiler, our programs contain a *lexer*, a *parser* and a *code generator*. The lexer splits up the program text into what are called 'tokens', pieces of information. These are then examined by the parser, which builds a tree representing the data structure. If the data can be parsed successfully, then the tree is well-defined and is passed on to the code generator, which produces target code as it examines each node in the tree [13].

The compiler developed in MuTaTeD! worked with two passes. The first translated SMDL into an intermediate form of NIFF, called SGML(NIFF). The second translated into NIFF binary code. Each of the two passes was written using LEX and YACC. The second pass translated an SGML(NIFF) file to a C program, using the NIFF SDK. This was compiled and executed, resulting in a NIFF binary file that can then be loaded into a NIFF compatible music editor, such as LIME. We have made these compilers for the MuTaTeD!1 project freely available for the developers' community (and hope that interest exists to use this technology to expand upon).

Besides proving the concept of integrating these two standards, the use of the LEX/YACC technology enabled us to implement the converters, but to certain extent also restricted our flexibility. With LEX/YACC we had to hardcode our chosen DTD's into the converters. This restriction was the major reason for implementing the follow-up system in the project MuTaTeD'II in a different way. The MuTaTeD'II team decided to use the Groveminder Technology, which, besides other advantages, caters for the reading-in of different DTDs. NIFF, being binary, also posed some problems of accessing and searching the content. The fact that any compilation/search procedure has to always be preceded by a reading in or out of the whole binary data in the case of binary formats, has consequences in the design of intuitive content music retrieval, specifically for displaying dynamically parts of music. So although NIFF has proved to be a powerful format, we are contemplating using other text based representation languages for future implementations. As depicted in Fig. 5, the effort involved in implementing compliance with other tag-based description languages is not immense.

1 For future notation within this document, a file written in SGML and using an X DTD is described as being in SGML(X). A file written in C and using functions from the NIFF SDK is referred to as being in C(NIFF SDK).

7. From MuTaTeD!I to MuTaTeD'II

Although proving the initial concept by having accomplished a prototype which was able to convert SMDL to NIFF for a restricted set of music examples (music up to 7 sharps and flats in time signatures of 4/4), we were aware of the restrictions imposed on this prototype due to the choice of technologies to be used and the time restraints on the whole project. NIFF, being binary, posed some problems of accessing and searching the content. The fact that a compilation procedure has to always precede a reading in, has consequences in the design of intuitive content music retrieval out of NIFF, specifically for displaying music. So although NIFF has proven to be the powerful format, another text based representation language for future implementations is being considered. Using LEX/YACC the DTDs had to be hardcoded into the parsers, which meant that future use of these parsers with slightly different types of music or DTDs was a matter of adding or changing the code. The second pass of the compiler restricts handling to a single line using treble clef, where as the first pass is able to process other clefs, and up to five staves. Although it would have been easy to build a second project on top of the same technology, expanding the parsers and converters to cover the full requirements of traditional music notation, we wanted to base the follow-up project on a technology which would be able to tackle these issues without the restrictions of the earlier project. To utilise the full portability and the scalability of the original SMDL standard, we chose to use the Groveminder system from TechnoTeacher Inc. as the basis for the MuTaTeD'II System. Using Grove technology [14] and its underlying object-oriented database management to store the groves we started to build a new system which we believe holds considerable promise for the music information retrieval community of the future. More details of the two projects can be found at their relevant websites [1 and 2].

8. MuTaTeD'II, the technical set-up

MuTaTeD'II used the same basic system architecture, this time with Groveminder's support for tagged based languages and its parsers and compilers. The Groveminder technology allows the construction of groves from any valid SGML document. A document is opened, parsed and validated by Groveminder. A SGML grove is then constructed ready for the second stage of processing.

Two main API's are used, both written in C++. The first is the Groveminder system itself which is used to lex, parse and search SMDL files. The second is GNU Cgicc, providing functions for talking to a HTTP server and creating HTML on the fly. Similar to MuTaTeD! the steps involved in the successful indexing for the searching process include the parsing and validation with simultaneously building a tree called the "grove", stepping and navigating through the resulting tree in order to locate specific information to be indexed for the search, creating containers and filling them with the values and finally applying standard search functions to these containers.

Essentially, the method of creating new in-memory data models containing only search-specific information promises a very efficient and very expandable basic software design. It allows the non-proprietary ethos of SGML to be extended to the way data is constructed and referenced. Although this conformity to a data model does not dispense with the need for specific code to be written for SGML files with different DTD's, it does provide a number of advantages.

1. Using Groveminder automatically provides support and the existence of a standard data model, which is closely linked to our source data.
2. By using a standard object-model like groves and property sets, the interoperation of a wide variety of different applications is easily realized. (For more information on Groves see (The XML Cover Pages))
3. Portability across platforms.

This is of immense benefit when expanding the system in light of future developments.

9. Conclusion

In the past the availability of music information on the net has been hampered for a variety of reasons, one of the main ones being the propriety nature of the file formats used in most score notation packages. Throughout the Mutated projects, we have tried to overcome some of these restrictions. In MuTaTeD! the task to make SMDL files viewable was undertaken by creating SMDL to NIFF converters. This opened the door for the next obvious step which was carried out in MuTaTeD'II. SMDL being a tag-based language was one of the obvious choices for transporting music information over the internet.

While the process of creating a fully interactive online music information search and analysis tool is still in its embryonic stage it is immediately obvious that a solution of this nature could have a variety of benefits to the music community.

- high-level to low-level music search facilities
- adaptable, expandable analysis tools
- use of an underlying powerful, standardized description language, which does not necessarily restrict the handling of purely encoded music
- the expansion possibilities into areas of audio/video tagging

Finally, the expansion of the web and the general increase in tag-based languages like XML and SGML mean that, in future, there will be a greater provision for developing applications that deal in these languages. The next generation of browsers (i.e. Netscape 6) will be fully XML compliant and, given time, SGML (or something of a similar level of complexity) will be more commonly used. This drive towards easier interchange of information is inevitable and provides an opportunity to put data under the control of users rather than leaving them prone to the fickle trends of commercial application developers who more often than not have been responsible for the bottleneck in information interchange.

References

[1] Cordy Hall, Carola Boehm Boehm, *Website of the Project "Music Tagging Type Definition", MuTaTeD!1*. http://www.pads.ahds.ac.uk/mutated 2000-06-05.

[2] Carola Boehm, and Donald MacLellan, *Website of the Project "MuTaTeD'II, A system for Music Information Retrieval of Encoded Music"* http://www.pads.ahds.ac.uk/mutated2.html 2000-06-05.

[3] Kostas Glinos (EU, DG3), *Information Access and Interfaces*, presentation for the last call of the 4th framework, 29 September 1997, Brussels.

[4] Following references are useful starting points for music representation standards of the past and present:
Wolfram Steinbeck, "Struktur und Aehnlichkeit, Methoden automatisierter Melodieanalyse", in: *Kieler Schriften zur Musikwissenschaft*, Band XXV, Baerenreiter Kassel Basel London 1982, p.29 and p.392.
Eleanor Selfridge-Field, *Beyond Midi*, Cambridge Massachusetts, CCRMAH 1997.
Murray J. Gould and George W. Longemann, "ALMA, Alphameric Language for Music Analysis." Barry S. Brook *Musicology and the Computer, Musicology 1966-200: A practical Program* (New York: The City University of New York Press 1970), p.57.
Helmut Schaffrath, "The ESAC Databases and MAPPET Software," in: *Computing in Musicology 8* (1992), p.66 and Schaffrath, "The Essen Associative Code", in: Selfridge-Field, p.343-359.

[5] Joahnnes Wolf, *Handbuch der Notationskunde II*, Hildesheim 1963, p.387-389.

[6] The Moving Picture Experts Group (MPEG) is a working group of ISO/IEC in charge of the development of international standards for compression, decompression, processing, and coded representation of moving pictures, audio and their combination. So far MPEG has produced: MPEG-1 (approved Nov. 1992), MPEG-2 (approved Nov. 1994), MPEG-4 version 1, the standard for multimedia applications (approved Oct. 1998). and is now developing MPEG-7 the content representation standard for multimedia information search, filtering, management and processing (to be approved July 2001). (http://drogo.cselt.stet.it/mpeg/ 11/01/00)

[7] Steven R. Newcomb, "Standards. Standard Music Description Language Complies with Hypermedia Standard." *IEEE Computer 24/7* (July 1991) p.76-79. ISSN: 0018-9162 and
Steven R. Newcomb, V. T. Newcomb, "Some Background Information about HyTime." *Journal of the Institute of Image Electronics Engineers of Japan 21/5* (October 1992) p.459-467.

[8] Cindy Grande and members of the NIFF project team, *Specification for the Notation Interchange File Format*,1995. http://esi24.ESI.UMontreal.CA:80/~belkina/N/NIFF6a3.txt 2000-06-05.

[9] Stephen Mounce (ed), *Niff Homepage*, http://www.student.brad.ac.uk/srmounce/niff.html, 1/11/00

[10] Carola Boehm, Cordy Hall, *ISO/IEC JTC1/SC29/WG11/ MPEG 98/P620/W2463, MPEG proposal: Description Scheme for description of music content*, (Vancouver: MPEG 1999)

[11] Cantate, *Computer Access to Notation and Text in Music Libraries. Download Deliverable 5-3 : Development Model with Summary and Recommendation*, Amsterdam 1997. http://www.svb.nl/project/cantate/cant_deliv.htm 2000-06-02.

[12] Holger Hoos, K. A. Hamel, K. Renz, J. Kilian. 1998. "The GUIDO Music Notation Format - A Novel Approach for Adequately Representing Score-level Music" (H. H. published in: *ICMC'98 Proceedings*, p.451-454) Specifications at http://www.informatik.tu-darmstadt.de/AFS/GUIDO/ 21/02/00

[13] Carola Boehm, Donald MacLellan, Cordy Hall, "MuTaTeD!1and MuTaTeD'II" *International Computer Music Conference Proceedings*, ICMC, Berlin 2000.

[14] Steve Newcomb, *Groveminder, Technical Description*, http://www.tecno.com/ 2000-06-05