Beyond Images: Encoding Music for Access and Retrieval

Nitin Arora
LS598, University of Alabama SLIS, Dr. MacCall, Spring 2010

Abstract

Libraries have embraced the digital encoding of textual documents for improved access and search-based retrieval but have largely ignored similar possibilities regarding the digital encoding of Symbolic Music Representation (SMR) for traditionally notated Western music. This is likely due in part to not only general unawareness by librarians of the prevailing types of Digital SMR formats, but also perhaps even to misguided beliefs that SMR is necessarily a graphical medium - hence the apparent satisfaction with the presentation of sheet music collections as strictly image-based entities.

This paper seeks to provide an brief overview of ASCII and XML-based Digital SMR and the possibilities they present libraries in terms of access and retrieval.

The primary focus of the paper will be to discuss such possibilities using MusicXML within the context of a prototypical Internet-based MusicXML access and retrieval platform written by the author for the purposes of initial research. The platform, entitled "MXMLiszt", seeks to demonstrate that open-source software can be leveraged by digital libraries for automated descriptive metadata generation as well as automated generation of manifestations of the source MusicXML document. These manifestations include sheet music images, audio, and musical transpositions. Metadata searching and preliminary Music Information Retrieval (MIR) functionality is also coded into the platform.

A. Introduction

Libraries have embraced the digital encoding of textual documents for improved access and search-based retrieval but have largely ignored similar possibilities regarding the digital encoding of Symbolic Music Representation (SMR) for traditionally notated Western music. This is likely due in part to not only general unawareness by librarians of the prevailing types of Digital SMR formats, but also perhaps even to misguided beliefs that SMR is necessarily a graphical medium - hence the apparent satisfaction with the presentation of sheet music collections as strictly image-based entities.

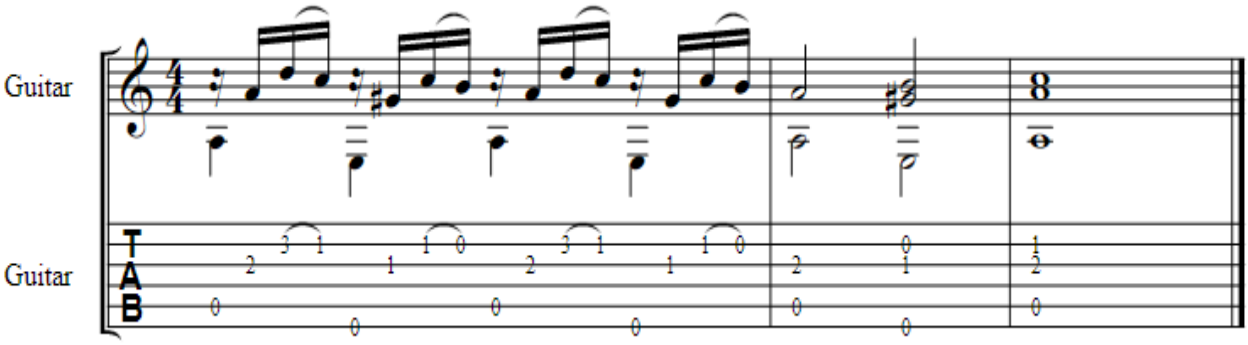
Research in regard to high-level Music Information Retrieval activities is typically carried out by computer scientists with musical knowledge or musicians with computer science knowledge. Conversely, libraries and librarians seem to be contributing little even though advances in the field have the potential to serve patrons and research efforts of all types. It is incumbent upon the library community to recognize the symbolic nature of their sheet music collections and to move beyond images so as to provide users with better access and retrieval options in regard to musical content.

Symbolic Music Representation

For the purposes of this paper, Symbolic Music Representation refers to musical notation, the "formal indication of how sounds and silences intended as music should be reproduced."[1] In contrast to the documentation of actual sounds (i.e. sound recordings), SMR is, as its name implies, symbolic. In terms of its ability to help a performer create music - the structured interaction of pitches and their duration across a canvas of time at various levels of amplitude - it is a loose documentation of an abstraction, though as a document it can be thought of as an entity onto itself.

In the Western tradition, myriad systems of musical notation have been devised to address a host of concerns. Though a survey of these systems and their histories lies well beyond the scope of this paper, it should be noted that the tradition of developing alternative musical notation systems continues[2] despite the current large scale adoption of Western "traditional music notation".

For demonstrative purposes, the example below depicts a musical excerpt for classical guitar utilizing traditional musical notation in the top portion, while the bottom portion presents the same basic pitch information, symbolically represented using modern guitar tablature, a system of depicting music for guitar from within the context of the physical layout of the instrument itself, with lines indicating strings and numerals serving to indicate frets.



The example above is not meant to stir debate regarding the superiority of one method of SMR over another, but rather to explicate the fact that SMR is not the music itself. Rather, it is both referential and deferential to underlying musical concepts; thus, diverse approaches to SMR can achieve the same essential function. This is no less true in the digital arena.

Digital Symbolic Music Representation

While analog SMR (as described above) can be thought of as both a lens, resting atop underlying musical attributes that work in tandem to formulate a musical composition, and as a human-readable document used as the basis for musical performance, Digital SMR occupies, arguable, only the first role. That is to say, the digital encoding of musical instruction is generally not the desired end result in terms of a human-readable document. In short, musicians will still want a static document that looks akin to the example above, even though that static document is to be outputted from dynamic digital encoding.

---

[1] Anthony Pryer. (2010) Notation in Oxford Music Online. Retrieved from http://www.oxfordmusiconline.com/subscriber/article/opr/t114/e4761

[2] See the Music Notation Project at "http://musicnotation.org" for examples of alternate systems of Western musical notation.

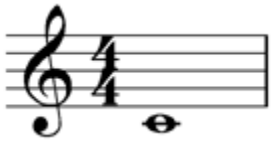
Mimicking the diverse approaches to analog SMR, digital formats also abound. Furthermore, even the relatively brief age of personal computing has seen the decline and fall of many efforts. A rather thorough list of many notation formats, active and inactive, can be viewed on Gerd Castan's music notation website[3]. Of note for their absence are the Institute of Electrical and Electronics Engineers' (IEEE) P1599, "Project MX"[4], and MSCX, an XML format used internally by MuseScore[5], a relatively new and open-source GUI music notation application. In fairness, it should be noted that Castan does list MuseScore amongst his list of score editors; it is simply the corresponding file format that is not listed in his compilation.

Castan's classification of the formats themselves is important: distinctions are made between proprietary and closed formats, such as those used by some of the dominant GUI score editors in use today (e.g. MakeMusic's Finale[6] and Avid's Sibelius[7]), and between ASCII-based and XML-based systems[8] which are both plain-text systems than can be opened, read, and edited with the simplest of text editing software. From the perspective of the librarian, this distinction amongst open text formats is arguably the more important simply because proprietary and closed systems of Digital SMR offer far less possibilities to the library community in terms of cost restrictions, yes, but also because they offer less opportunities to apply open systems of automation, delivery, and retrieval. These opportunities will be demonstrated later in this paper within the context of a widely deployed Digital SMR format, MusicXML.

B. Digital SMR examples: Lilypond and MusicXML

As a way to compare ASCII and XML-based approaches to Digital SMR, it is worthwhile to examine the encoding of a simple musical example using both the Lilypond format and MusicXML. Based on both general open-Web activity and software support, both appear, respectively, to clearly be the dominant examples of ASCII and XML-based Digital SMR in use today.

For the purposes of comparison - and sake of simplicity, consider the minimalist musical example below:



---

[3] Gerd Castan. (2010) Music Notation - Musical notation codes. Retrieved from http://www.music-notation.info/en/compmus/notationformats.html

[4] Denis Baggi and Goffredo Haus. (2010) The Concept of Interactive Music: The New Standard IEEE P1599 / MX. Retrieved from http://www.springerlink.com/content/hn847p3713220407/fulltext.pdf

[5] Musescore|Free music composition & notation software. (2010) Retrieved from http://musescore.org/

[6] Finale Music Composing & Notation Software. (2010) Retrieved from http://www.finalemusic.com/

[7] Sibelius. (2010) Retrieved from http://www.sibelius.com/home/index_flash.html

[8] Margaret Cahill's 2001 paper "Using XML for Score Representation" provides and excellent overview of XML and its implications for Digital SMR; available at http://www.csis.ul.ie/dafx01/proceedings/papers/cahill.pdf

A possible version of the music above in Lilypond's ASCII format might be the following:

```
\score {
        <<
        \time 4/4
        \relative c'
        c1 |
        >>
}
```

while corresponding components of a MusicXML version could look like this[9]:

```
<score-partwise>
  <part-list>
    <score-part id="P1">
      <part-name>Music</part-name>
    </score-part>
  </part-list>
  <part id="P1">
    <measure number="1">
      <attributes>
        <divisions>1</divisions>
        <key>
          <fifths>0</fifths>
        </key>
        <time>
          <beats>4</beats>
          <beat-type>4</beat-type>
        </time>
        <clef>
          <sign>G</sign>
          <line>2</line>
        </clef>
      </attributes>
      <note>
        <pitch>
          <step>C</step>
          <octave>4</octave>
        </pitch>
        <duration>4</duration>
        <type>whole</type>
      </note>
    </measure>
  </part>
</score-partwise>
```

Obviously, from the examples above one might assume Lilypond to be a naturally less verbose format, but one should factor in the generally inherent verbosity of XML itself before drawing such conclusions. Furthermore, the particular Lilypond snippet contains less explicit information about the music than does the MusicXML code. For instance, the Lilypond snippet does not inform the reader that there is indeed only one instrumental part to the music nor does it explicitly declare the key signature. Lilypond code most definitely becomes

[9] Hello World" in MusicXML. (2010) Retrieved from http://www.recordare.com/xml/helloworld.html

rather verbose when “real world" compositions are encoded into its format to include such intricacies as fingering and performance marks.

It should also be noted that Lilypond is both a format and a command-line executable open-source application that renders graphical images from ASCII Lilypond files. MusicXML is, on the other hand, a human-readable XML musical encoding though unlike the Lilypond format it is not, pragmatically speaking, human-editable. Therefore, when one refers to "Lilypond" they may be referring to either the Lilypond ASCII syntax, the Lilypond software component, or both.

C. Digital SMR and the World Wide Web

Broadly speaking, the library community at-large seems to be ignoring the possibilities of Digital SMR in terms of dynamic digital sheet music access and interactive contexts while the open World Wide Web has flourished with such activity. Following is a cursory survey of some of the repositories, browser-based applications, and Internet platforms available for web deployment and interactive functionality regarding Digital SMR.

Online Digital SMR Repositories

Occupying a more traditional role, somewhat akin to institutional libraries that generally offer only images of music collections, are online repositories of Digital SMR files. Most notable of these are perhaps the Mutopia Project[10], which offers scores in the aforementioned Lilypond format, and Project Gutenberg[11], the well-known provider of e-texts, which houses dotted offerings of sheet music in various digital formats, including Lilypond, MusicXML, Finale, and Sibelius files. Generally, images files and MIDI[12] files are presented alongside the files in the formats mentioned.

While such repositories offer only "flat files" in the same manner institutional digital libraries offer images of sheet music collections, the primary difference is the ability for users to download the various Digital SMR formats from the open-web repositories and, with the correct software applications, manipulate and interact with the musical information locally (i.e. on their personal computer).

Interactive, Browser-based Score Viewing and Playback

Based on simple empirical evidence it would seem quite safe to claim that MakeMusic's Finale and Avid's Sibelius are the leading GUI notation applications in use today in institutional, publishing, and personal computing environments. Therefore, not surprisingly, both offer free browser-based solutions to view scores written in their native Digital SMR formats. The Finale Viewer integrates with Microsoft's Internet Explorer and Apple's Safari web browsers, allowing users to "view, play, transpose[13], and print digital sheet music created in Finale"[14]. Similarly, Sibelius Scorch, which integrates with Internet Explorer,

[10] The Mutopia Project. (2010) Retrieved from http://www.mutopiaproject.org/
[11] Project Gutenberg. (2010) Retrieved from http://www.gutenberg.org/
[12] It should be noted that MIDI files instruct a computer sound card to produce aural playback but are not themselves sound files nor true examples of Digital SMR as MIDI does not differentiate some of the vital features of SMR, such as spelling and note beaming concerns. MIDI does, however, have a tenuous history of being co-opted as a Digital SMR interchange format given its ability to store pitch and duration information.
[13] Transposition refers to the modulation of pitch level at which a musical composition exists.
[14] Finale Viewer. (2010) Retrieved from http://www.finalemusic.com/viewer/

Safari, Netscape, and the Firefox browser mirrors the Finale Viewer functionality yet for scores written in Sibelius. Additionally it adds options to save the score and change the instrumentation[15], allowing the user to hear the music utilizing different sounds from one's computer sound card. Both companies' websites offer scores written by their customer base that can be viewed via their respective plug-ins.

Three interesting examples of applications that offer similar functionality to Finale Viewer and Sibelius Scorch are Myriad Software's Myriad Music Plug-In[16], the Adobe Flash-based Sheet Music Viewer[17] by Legato Media, and Xenoage Software's Java applets, the Zong! Player[18] and Zong! Viewer[19]. All three use MusicXML documents as the source material rather than proprietary formats. Of the three, only the offering by Xenoage Software is open-source. Still in early development, it likely possess the greatest potential in terms of adoption by institutional libraries. Though it does not yet offer transposition options or the ability to alter the playback tempo of a composition, these features could be introduced by third parties if necessary given the open nature of the source code.
Also of note is KernScores[20], which 'hosts musical data encoded in the Humdrum's predominant encoding method, the "kern" format"[21], an ASCII-based Digital SMR format developed at Ohio State primarily for the purposes of musical analysis and research[22]. The scores in the "kern" format can be pasted into the site's Online Humdrum Editor and be transposed or exported to various image formats, MIDI files, or MEI[23], the XML-based Music Encoding Initiative format developed by Perry Roland of the University of Virginia.

Web 2.0 Platforms

It should be noted that the Web 2.0 site Wikipedia.org, in considering Digital SMR standards for inclusion of sheet music snippets within its online articles, has struggled with selecting a default Digital SMR standard for Wikipedia given the inherent difficulties with a general lack of true, real-time read/write functionality in regard to online Digital SMR.[24] However, efforts are underway to cope with this issue elsewhere on the Internet. That is to say websites like Wikifonia.org and Noteflight.com offer users the ability to not only view musical documents online but also to manipulate musical content and share documents via social networking sites, such as Facebook.com.
Wikifonia is a publishing platform for lead sheets, or depictions of one-voice melodies on a traditional music staff with added textual elements to show the harmonic accompaniment (chords) along with any lyrics[25].

---

[15] Scorch. (2010) Retrieved from http://www.sibelius.com/products/scorch/index.html
[16] Myriad Music Plug-In. (2010) Retrieved from http://www.myriad-online.com/en/docs/plugdoc.htm
[17] Legato Media. (2010) Retrieved from http://musicrain.com/products-sheetmusicviewer.php
[18] Andreas Wenger. (2010) Zong! Player. Retrieved from http://www.xenoage.com/zongplayer.html
[19] Andreas Wenger. (2010) Zong! Viewer. Retrieved from http://www.xenoage.com/zongviewer.html
[20] KernScores. Retrieved from http://kern.humdrum.net/
[21] KernScores Guided Tour. Retrieved from http://kern.humdrum.org/help/tour/
[22] David Huron. (2010) The Humdrum Toolkit: Software for Music Research. Retrieved from http://www.musiccog.ohio-state.edu/Humdrum/
[23] Music Encoding Initiative (MEI), U.Va. Library. (2010) Retrieved from http://www2.lib.virginia.edu/innovation/mei/
[24] Music markup - Meta. (2010) Retrieved from http://meta.wikimedia.org/wiki/Music_markup
[25] Robert Witmer. (2010) Lead sheet in Oxford Music Online. Retrieved from http://www.oxfordmusiconline.com.libdata.lib.ua.edu/subscriber/article_citations/grove/music/J262200

Wikifonia users can attain free membership and upload lead sheets with the requirement that the file be in MusicXML format[26]. Registered users can also alter a composition's descriptive metadata via an online interface or download the source MusicXML file, edit it locally, and then upload a newer version.

Apart from browser-based viewing of compositions, Wikifonia's lead sheets can be downloaded in Adobe's PDF format; this includes on-the-fly user-initiated transpositions. Interestingly, Wikifonia's PDF and transposition features are handled by Lilypond, which does not natively support MusicXML documents. That is to say once the user initiates printing and/or transposition operations, the MusicXML source is converted to Lilypond format after which the end document is transposed - if necessary - and delivered.

To achieve this conversion the developers wrote their own Java-based MusicXML to Lilypond converter though a Python script, musicxml2ly[27], written by Reinhold Kainhofer comes bundled with the version of Lilypond employed by Wikifonia. According to the developers, Kainhofer's script was not fully mature at the time Wikifonia went live, hence the domain-specific Java-based solution.[28] Furthermore, Lilypond was chosen as the final output software given it is, according to one of the site administrators, "the best renderer for making sure no elements collide"[29].

Noteflight takes the read/write notation notion to a more extreme level than Wikifonia. Noteflight is, in essence, a browser-based GUI notation application. Users can notate music online, share their compositions with others, enable comments from other users, and give other users the option to edit one's original score online. Compositions may also be embedded into other websites via provided embed code. The only Digital SMR export option Noteflight currently offers is export to MusicXML, though one may also export MIDI files and WAV, a digital audio format[30].

D. Digital SMR and Music Information Retrieval (MIR)

"*We are not impressed any more by functionality that allows us to search for a text pattern, highlighting all occurrences in the text itself. But in score editors, searching for a music pattern and highlighting the results in score, is a functionality that is only noticeable by its absence*"[31].

The statement above is as troubling as it is true. That is not to say that MIR activities are being conducted without vigour, but it is to suggest that for mainstream users of musical documents few easy-to-use MIR options exist. These users may be college-level vocal students seeking to find pieces within their given vocal range, elementary music teachers seeking songs on a given textual theme, or serious music scholars trying to extrapolate linear and intervallic “rules of motion" from compositions.

While searching of musical information may seem like a leap from that of typical textual documents (articles, poems, etc.), it should not be viewed as such. Simply put, musical

[26] Frequently Asked Questions. (2010) Retrieved from http://www.wikifonia.org/faq#n1125
[27] Reinhold Kainhofer. (2010) MusicXML to Lilypond (musicxml2ly) development. Retrieved from http://wiki.kainhofer.com/musicxml2ly/start
[28] MusicXML to LilyPond formatting. (2010) Retrieved from http://www.wikifonia.org/node/1115
[29] Backend software for Wikifonia. (2010) Retrieved from http://www.wikifonia.org/node/5084
[30] Noteflight FAQ. (2010) Retrieved from http://www.noteflight.com/info/faq
[31] Ganseman, Scheunders, and D'haes. (2008) Using XQuery on Musicxml Databases for Musicological Analysis. Retrieved from http://ismir2008.ismir.net/papers/ISMIR2008_217.pdf

information that is encoded into a rigid syntax (Digital SMR) is by its very nature able to be queried programatically. That is to say, just as much music is comprised of patterns discernable by trained musicians, the syntaxes of various forms of Digital SMR also exhibit patternistic structures that can be mined for information.

This mining can be achieved in alternate ways, such as

1. employing software written specifically to work in conjunction with a given Digital SMR format or

2. by leveraging the common syntactical nature of certain types of Digital SMR along with pre-existing search/retrieval technologies.

An example of the former is the Humdrum tookit, developed specifically to work with musical information encoded in the the native Humdrum syntax. The syntax can encode a multitude of musical information from various genres and cultures; the toolkit's "set of operations ... are probably the most comprehensive set of symbolic music operations in the MIR world"[32]. The Humdrum approach to MIR is, however, highly complex and requires the user to have facility with Unix-style commands[33], likely making it out-of-reach for many.

GoldSmith Digital Studios has also developed an alternative toolkit for Humdrum scores and a method for outputting search results via Lilypond[34]. These tools, PerlHumdrum and PerlLilypond are also for advanced users.

More user-friendly examples for MIR can be found on abcnotation.com's[35] list of search interfaces to query collections generally composed of documents encoded in ABC, a simple ASCII-based Digital SMR format. Some of these interfaces are extremely well-geared toward end-users, allowing one to input search criteria via “playing” melodic snippets into a "piano roll" (a physical depiction of a piano keyboard) or via humming into a microphone.

Another example of a fairly user-friendly search interface is Riff Roll[36], which appears to be a search front-end for the MusicXML-based Wikifonia collection. Due to the website's limited documentation, it could not be determined prior to the conclusion of this paper exactly what technologies drive Riff Roll's search capabilities. It appears at one point Riff Roll also offered MIR capabilities for Noteflight scores with Noteflight's API, though that Application Programming Interface appears now to be closed[37].

In terms of using pre-existing search and retrieval technology to take advantage of Digital SMR with a common syntactical nature, two possibilities using MusicXML serve as an example. As MusicXML documents are, as the name implies, XML encoded data, they are immediately subjectable to search and retrieval via XQuery, a native technology to query XML documents, and SQL (Structured Query Language).

---

[32] Ian Knopke. (2008) The PerlHumdrum and PerlLilypond Toolkits for Symbolic Music Information Retrieval. Retrieved from http://ismir2008.ismir.net/papers/ISMIR2008_253.pdf
[33] Humdrum Toolkit FAQ. (2010) Retrieved from http://www.musiccog.ohio-state.edu/Humdrum/FAQ.html
[34] Ian Knopke. (2008) The PerlHumdrum and PerlLilypond Toolkits for Symbolic Music Information Retrieval. Retrieved from http://ismir2008.ismir.net/papers/ISMIR2008_253.pdf
[35] Chris Walshaw. (2010) com | tune search. Retrieved from http://abcnotation.com/search
[36] Riff Roll. Retrieved from http://riffroll.com/
[37] Music Hack Day Boston / Melody Search with Noteflight API hack. (2010) Retrieved from http://musichackdayboston.pbworks.com/Melody-Search-with-Noteflight-API-hack

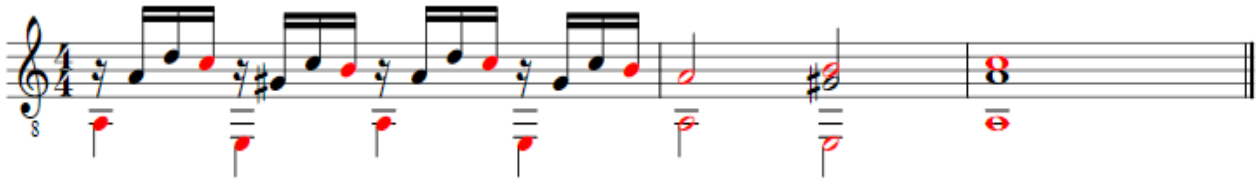
In a paper submitted for the 2008 ISMIR Conference (The International Society for Music Information Retrieval), European researchers Joachim Ganseman, Paul Scheunders, and Wim D'haes demonstrated the appropriateness of implementing XQuery for MIR activities in regard to MusicXML[38], following up on prior research conducted by others circa 2002. The researchers used a set of pieces from the Wikifonia collection and used the open-source native XML database eXist-db[39] to implement queries regarding the number of notes per piece, pieces in a given key, and key signature statistics for the entire collection, etc.

A slightly different approach has been taken by Bret Aarden, author of MusicSQL. MusicSQL ingests MusicXML documents into a more traditional database . Via command-line Python scripts one can query the database (currently driven my MySQL)[40]. Snippets of query results can be exported back to MusicXML and can be printed via Lilypond with result highlighting. Given its simple interface and use of MySQL, it would be no great leap to implement MusicSQL via a web interface on a server supporting the Python scripting language.

Lilypond is clearly a popular output choice for many of the efforts mentioned due, in no small part, to its seemingly unmatched output capabilities. Moreover, displaying query results in traditional notation is essential given that "many traditional musicologists and theorists without a back-round in MIR have difficulties with non-traditional notation formats."[41]

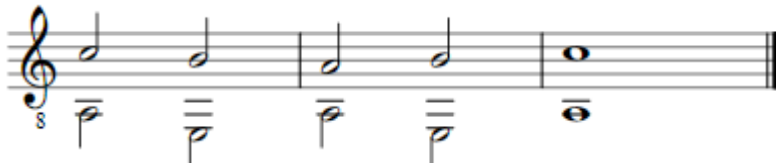
It should also be noted that MusicXML has also served as the basis for MIR in recent projects researching its potential as the basis for Schenkarian Analysis, a system of notating the core structure of traditional tonal music[42] (very broadly speaking, compositions that have a central "key"). Developed by the Austrian musical theorist Heinrich Schenker (1868-1935), Schenkarian Analysis, while beyond the grasp of casual musicians, does lend itself to algorithmic computer operations, potentially allowing end-users to retrieve compositions from a collection based upon similar underlying musical structures[43].

Though not an example of true Schenkarian Analysis, the example below aims to demonstrate the structural components of a simple musical excerpt. Notes in red are interpreted to be the essential components of the excerpt.



[38] Ganseman, Scheunders, and D'haes. (2008) Using XQuery on Musicxml Databases for Musicological Analysis. Retrieved from http://ismir2008.ismir.net/papers/ISMIR2008_217.pdf
[39] eXist-db Open Source Native XML Database. (2010) Retrieved from http://exist.sourceforge.net/
[40] MusicSql Documentation. (2010) Retrieved from http://musicsql.googlecode.com/files/ReadMe-0.1.2.pdf
[41] Ian Knopke. (2008) The PerlHumdrum and PerlLilypond Toolkits for Symbolic Music Information Retrieval. Retrieved from http://ismir2008.ismir.net/papers/ISMIR2008_253.pdf
[42] Tom Pankhurst's Guide to Schenkerian Analysis - What is Schenkerian analysis. (2010) Retrieved from http://www.schenkerguide.com/whatisschenkerianananalysis.php
[43] Philip Kirlin. (2009) Using Harmonic and Melodic Analyses to Automate the Initial Stages of Schenkerian Analysis. Retrieved from http://ismir2009.ismir.net/proceedings/PS3-6.pdf

The figure below shows the prior musical progression consisting of only the core musical elements:



E. Library adoption of Digital SMR

Search and Retrieval

Since MIR activities outside of libraries has been surveyed, it may be helpful to consider why MIR has not been deployed in the digital library setting - an environment for which the deployment of MIR is an obvious candidate for both scholarly and non-scholarly research.

Given the abundance of library search and retrieval technologies in regard to fiction and non-fiction textual documents, there seems to be little defense for the virtual non-existence of a corallary in terms of widely-deployed, Internet-friendly search and retrieval mechanisms for Digital SMR. In fact, even the more ardous endeavors of information retrieval for image, audio, and video content receives more mainstream attention than does MIR. One could argue that aside from simple ignorance of the existence of Digital SMR there are likely two reasons for this, one primary and the other secondary.

First and foremost is perhaps that over the years in which traditional music notation has been dominant in terms of Western tonal music, the shadow has - in the Platonic sense - become mistaken for the object from which it eminates. That is to say that SMR is likely widely regarded by many as music itself, rather than a symbolic representation of something far more ethereal.

Second is the likely unawareness of the existence of Digital SMR as an entity onto itself. That is to say that domain specialists within the library and information hemispheres, i.e. music librarians, are more then likely attuned to the use of score editors (Finale, Sibelius, et al.) to create digital scores but perhaps not actively conscious of the fact that the traditional images these editors produce on the screen or print to paper are themselves manifestations of encoded data - data that can be harnessed for purposes other than display.

If the suggested causes hold any kernels of truth, it is not the responsibility of musicians to change their ways of thinking about SMR, analog or digital, but that of librarians and information specialists, particularly those in the music and digital library fields. This needs to happen before any large-scale academic consumption of MIR can occur.

Digital SMR and Preservation Activities

Akin to the widespread use of word processing applications, much of what is composed today is likely to be inputted into GUI notation software at some point. That is to say that the composition will be notated directy into notation software or, if originally written by hand,

will likely later be inputted into notation software for the purposes of reading ease and professional presentation/publication concerns given both the inherent difficulities in rendering aesthetically pleasing hand-written scores as well as the ease of part preparation and audio generation that notation software affords. That is to say if one were to notate a song for guitar and voice, notation software would easily allow for the extraction of parts, or sheet music containing only the respective music for each instrument. Computer generated audio files can also be rendered from GUI notation software, allowing for prospective performers to get a cursory, aural "glance" of the music.

As an aside, it should be mentioned that while the quality of computer-generated audio files are improving, they still have a long-standing bad reputation in terms of aesthetic appeal. It can be argued that such opinions are largely born of ignorance from a larger perspective. That is to say that the lack of interpretive and aesthetic appeal found in most computer generated audio - i.e. that which makes it sound "dull" - is not necessarily a negative trait, but rather a potentially positive one even from a pedagogical approach. Non-human playback can offer aural insight into the piece without unduly influencing a prospective performer's interpretation of the various nuances of the music, interpretation being a performer's primary musical responsibility. Computer-generated audio of digital music scores are not a replacement for human performance, they are simply an additive tool in regard to the musical collective of a given work.

Digital libraries charged with preservation of born-digital scores that think only in terms of music documents as an image might seek only to house image formats of the music and, at best, perhaps also any corresponding audio files for enhanced web content delivery. Others may seek to also store the notation software's project files, proprietary in the case of programs such as Finale and Sibelius. Yet, these formats may or may not be sustainable in the long term. Therefore it is the library that also stores the composition in a non-proprietary, open textual format that allows itself the greatest number of options in terms of enhanced user experience and long-term survival potential of the fundamental musical information contained within each composition.

In an increasing world of born-digital content, Digital SMR has massive potential in terms of digital library preservation activites. Yet, this potential also extends beyond born-digital content.

In terms of content that is not born-digital - i.e. the likely bulk of many digital library sheet music collections - Optical Character Recognition for music, or Optical Music Recognition (OMR), can be used to derive, in an essentially retrograde fashion, Digital SMR formats from existing image-based manifestations of musical works. This entails, however, philosophical issues not necessarily found in digital library OCR processes concerning textual documents, namely prose documents.

More specifically, text files derived from OCR of prose documents still posess a high degree of utility even for recognition rates substantially less than one hundred percent accuracy. That is to say that such writings arguably have semantic meaning beneath the surface of their linguistic rendering. As such, less-than-perfect OCR of such documents can still lead to effective search and retrieval capabilities and could also, more or less, be used by a reader to at least glean the primary concepts from many such documents. This argument, of course, may be less and less true for articles of highly complex scientific or mathematical natures.

For poetic scripts and particularly musical documents the surface level rendering of the creator's ideas - i.e. lines of poetry and music - is, in essence, the idea itself. As such, OMR of musical information would from the perspective of artistic allegiance require manual intervention so that the resultant OMR output is virtually one hundred percent accurate.

Given the overhead involved in human remediation of OMR results, digital libraries with large image-based sheet music collections would, in reality, only be able to apply this methodology to small sub-collections when using OMR applications which are still far less-than-perfect though impressive nevertheless.

Given the current predilection for XML documents in the digital library arena and MusicXML's rapid and widespread adoption by myriad closed and open-source technologies, MusicXML arguably emerges to the fore as the current leading candidate for a preferred Digital SMR format from a digital library perspective.

Why MusicXML?

Recordare LLC's MusicXML is the most widely supported XML-based encoding for Digital SMR[44] of traditional tonal music[45], with wide ranging support by closed and open-source applications. It is considered by many to be an "emerging standard"[46].

From the perspective of digital libraries, two open-source, natively MusicXML-compatible applications would seem to bare particular mention:

1. MuseScore, an open-source, cross platform GUI notation application capable of MusicXML import/export and subsequent rendering of image and audio files from MusicXML documents. MuseScore's current native project file format is also XML-based.
2. MusicXML Library[47], an open-source, cross-platform application that is, among other things, capable of transposing MusicXML documents.

With the obvious cost and sustainabiltiy advantages of being open-source, both applications also support command-line interfaces. That is to say, both can perform tasks without GUI interfaces. The MusicXML Library is, in fact, command-line only. As both can be implemented via command-line, both are ideally suited to batch automation processes for large collections of MusicXML documents as well as for backend use in websites.

Additionally, the inherent nature of MusicXML - that is to say that it is XML-based - makes it subjectable to two important standard XML technologies, namely:

1. XSLT (EXtensible Stylesheet Language Transformation), a World Wide Web Consortium (W3C) standard that can be used to crosswalk data between XML schemas. For example, XSLT can be used to create preliminary XML-based descriptive metadata (Dublin Core, etc.) from MusicXML documents.
2. XQuery: another W3C standard which can be used to query XML documents.

In short MusicXML is supported by technologies that can be used for not only search and retrieval activities but also to derive manifestations (images, audio, transpositions) from a single source, i.e. MusicXML which is itself a manifestation of the larger work, the composition.

---

[44] Ganseman, Scheunders, and D'haes. (2009) Using XML-formatted Scores in Real-time Applications. Retrieved from http://ismir2009.ismir.net/proceedings/PS4-15.pdf
[45] For information on deliberate limitations of MusicXML, see the section “Limit Scope Carefully" at “http://www.recordare.com/good/xml2006.html".
[46] MusicSql Documentation. (2010) Retrieved from http://musicsql.googlecode.com/files/ReadMe-0.1.2.pdf
[47] MusicXML Library. (2010) Retrieved from http://libmusicxml.sourceforge.net/

F. MXMLiszt: a protoypical MusicXML digital library platform

Introduction

For the purposes of investigating the possibilities of exploiting the MusicXML format in the digital library context, a preliminary web-based MusicXML digital library platform was developed by the author; it is meant only to serve as proof-of-concept to the at-large library community of said possibilites. That is to say, from an argumentative perspective it made much more sense to "show" rather than to "tell", an emphasis on demonstrable practice over mere theoretical conjecture.

The platform is coded exclusively in PHP and employs HTML[48], CSS[49], and XSL[50] for controlling content display. Though the platform functions in the latest iterations of the Internet Explorer, Safari, and Google Chrome browsers, it is only fully functional in Firefox version 3 (the most current version is 3.6.3). The Opera browser is not supported.

The platform, entitled MXMLiszt, was approached from the basis that in order to satisfy curiosities of the at-large library community the software would need to support:

1. the following basic user needs:

    a. access to surrogate records in the form of widely-used library metadata schema such as OCLC's Dublin Core or the Library of Congresses' Metadata Object Description Schema (MODS), etc.
    b. browse functionality to support serendipitous discovery
    c. search and retrieval mechanisms built around the surrogate metadata
    d. and access to media manifestations of the scores (PDF, audio).

2. the following features to be generally of interest to musicians and music researchers:

    a. the ability to transpose the score - i.e. change "key"
    b. support for MIR queries
    c. and access to the MusicXML files themselves for local manipulation via desktop notation software - for example, changing lyrics or extracting specific passages for inclusion in research, etc.

3. the following administrative features:

    a. automated creation of preliminary surrogate metadata (to be later enhanced or "remediated" by metadata librarians) as well as the creation of reports as to what surrogate metadata has and has not been remediated
    b. automated creation of thumbnail images and PDF documents for each score
    c. and automated preparedness of surrogate metadata and MusicXML data across the entire online collection for the purposes of end-user search/retrieval and MIR.

[48] See: HTML Tutorial. (2010) Retrieved from http://www.w3schools.com/html/default.asp
[49] See: CSS Tutorial. (2010) Retrieved from http://www.w3schools.com/css/default.asp
[50] See: XSLT Tutorial. (2010) Retrieved from http://www.w3schools.com/xsl/

"MXMLiszt" stands for "MusicXML List" and also plays on the name of Franz Liszt, the renowned 19th century composer/pianist. The platform is, as of this writing, accessible at http://opensourcelibrarian.org/MXMLiszt.

Software Dependencies

To meet the aforementioned goals, MXMLiszt utilizes the following open-source software:

- Musescore (version 0.9.5)
- MusicXML Library (version 2.0)
- ImageMagick (version 6.5.9-5)
- Xenoage Player (version 0.4.1)
- Saxon-HE (version 9.2)
- BaseX (version 6)
- XAMPP (version 1.7.3)
  - Apache (version 2.2.14)
  - PHP (version 5.3.1)

While the basic capabilities of MuseScore and the MusicXML Library have already been mentioned, a brief explanation of the features of the other software is needed. ImageMagick is a powerful "software suite to create, edit, and compose bitmap images" and is capable of handling over 100 different types of image file formats[51] though its role in MXMLiszt is limited to resizing image previews of the musical scores to "thumbnail" sizes.

The Xenoage Player is a Java-based application capable of rendering audio from MusicXML files either as a standalone desktop application or as a web applet. End-users of must have the Java Runtime Environment installed in order to use the Xenoage applet. The sound samples used to generate the audio are taken from Oracle/Sun Microsystems' Java Soundbank[52].

The Xenoage Player is actually no longer actively developed and has been replaced by a newer player, Zong!, which was at the time of this writing still in early beta stage, hence the deference to the older Xenoage player. The Zong! player holds much promise as its capabilites extend beyond the Xenoage Player to include viewing and printing of MusicXML scores via a web applet. As such, the Zong! software holds a particular set of interest for digital libraries interested in deploying MusicXML in an Internet environment and its development should be eagerly watched.

Saxon-HE is the non-commercial version of this well-known XSLT and XQuery processor. That is to say, the software is capable of implementing XSL transformations (i.e. automated crosswalks across metadata schema) as well as XQuery searches of XML documents. The XQuery functions are not used in MXMLiszt due to this non-commercial version's lack of support for advanced XQuery features such as wildcard and "fuzzy logic" searching. Available in both Java and .NET flavors, it is the Java version used for MXMLiszt.
Unlike the Saxon-HE software, wildcard and fuzzy logic searching is supported in the BaseX XML database software. Its XQuery processor supports, as of this writing, "the latest W3C Full Text and Update Recommendations"[53]. The Java version is used for MXMLiszt given its support of a command-line interface.

---

[51] Imagemagick Studio Llc. (2010) Retrieved from http://www.imagemagick.org/
[52] Java Sound API: Soundbanks. (2010) Retrieved from http://java.sun.com/products/java-media/sound/soundbanks.html
[53] Basex Team. (2010) BaseX – XML Database and XPath/XQuery Full Text Processor. Retrieved from http://www.inf.uni-konstanz.de/dbis/basex/index

XAMPP stands for x-Apache-MySQL-PHP-Perl, where "x" equates to one's operating system. It is a software bundle that can instantly turn one's home computer into an active web server, similar to the ubiquitous LAMP (Linux-Apache-MySQL-PHP) software bundle which is the basis for many home-based and commercial web servers running some distribution of Linux. For MXMLiszt, only the Apache HTTP Server and PHP scripting language components of the XAMPP bundle are required.

It should be noted that while only open-source and cross-platform MusicXML and XML software is used for this project, the operating system used by the server for the beta version of the MXMLiszt platform is ironically a proprietary one - specifically, the 32-bit Home Edition of Windows XP (Service Pack 3). Given that all the other dependencies are open-source, it is theoretically possible to run MXMLiszt on Linux-based servers as well, though the efficacy of this has not been verified by empirical fact. This will be tested at a later date using the Xubuntu[54] Linux distribution.

Naturally, the Java Runtime Environment[55] must be installed on the server-side to run BaseX and Saxon-HE.

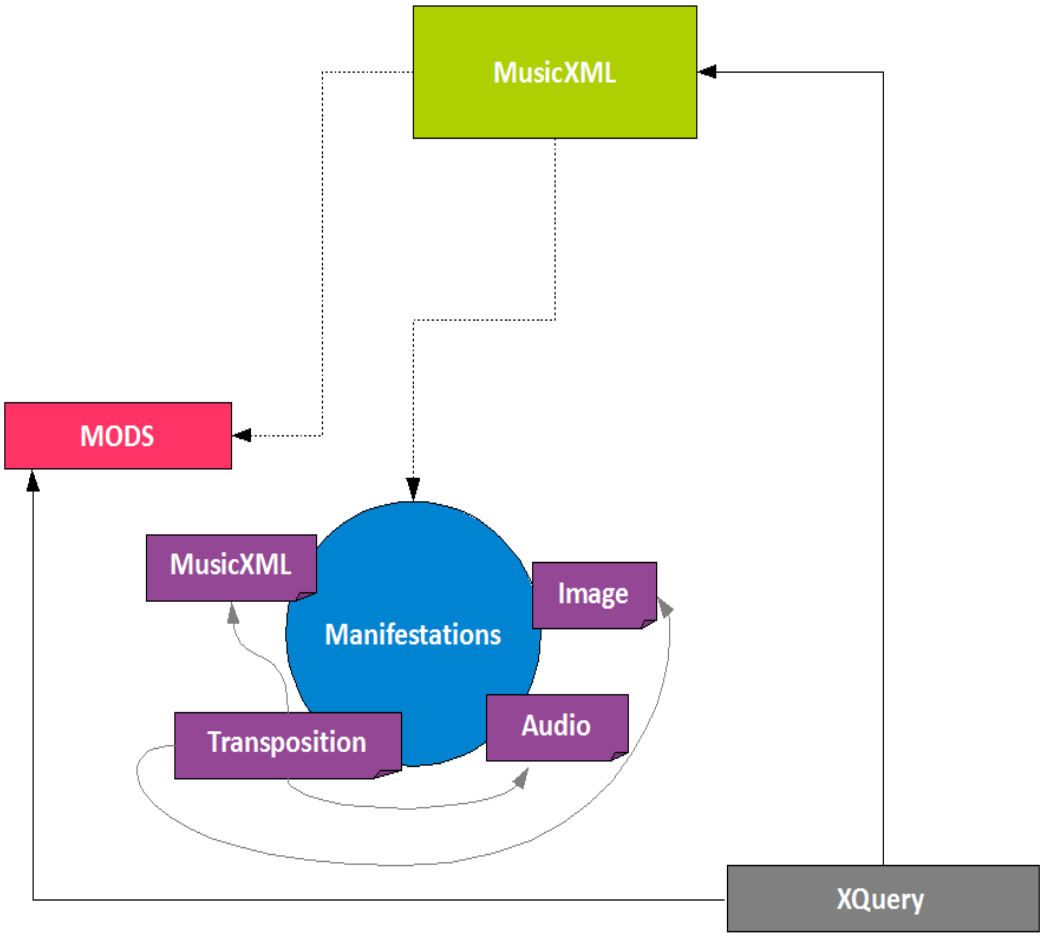
How it Works: a Basic Overview

Those interested in experimenting with the platform from the server side perspective can learn more about MXMLiszt's overall folder structure and the role of each and every script, file, etc. by reading the software manual[56].Therefore, only an overview of the essential operations of the platform seems needed.

MXMLiszt works on the principle that while the MusicXML encoding for a given composition is itself a manifestation of a work (i.e. the “composition”), the MusicXML file itself can function as a sub-work capable of birthing derivative image and audio files; manifestations also include musical transpositions in MusicXML format, thus allowing the the transpositions to serve as sub-sub-works also capable of spawning new image and audio content.

[54] Xubuntu. (2010). Retrieved from http://www.xubuntu.org/
[55] Java. (2010) Retrieved from http://www.java.com/en/download/manual.jsp
[56] Nitin Arora. (2010) MXMLiszt. Retrieved from http://blog.humaneguitarist.org/projects/mxmliszt/

The following graphic attempts to depict the essential overall framework of the platform.



Metadata

Given that MusicXML contains descriptive metadata or what can be thought of as descriptive metadata, it can also serve as the source of preliminary XML-based descriptive metadata in library-friendly schema.

MODS was chosen as the metadata format for the beta version of the platform as it is just slightly less simple than Simple Dublin Core but allows, unlike Simple Dublin Core, for the distinction between MusicXML's <creator> element as composer, arranger, and/or lyricist. As these distinctions are disambiguated within the MusicXML files themselves, thus it only made sense to opt for a descriptive metadata schema such as MODS that easily preserved such disambiguation.

For example, consider the following hypothetical MusicXML excert for a song by the famed songwriting duo of Richard Rodgers (1902-1979) and Oscar Hammerstein (1895-1960):

```
<creator type="composer">Rodgers, Richard</creator>
```

```
<creator type="lyricist">Hammerstein, Oscar</creator>
```

MODS easily supports losless retention of this descriptive metadata:

```
<namePart>Rodgers, Richard</namePart>
<role>
   <roleTerm type="text" authority="marcrelator"composer</roleTerm>
</role>
...
<namePart>Hammerstein, Oscar</namePart>
<role>
   <roleTerm type="text" authority="marcrelator"lyricist</roleTerm>
</role>
```

By contrast, Simple Dublin Core would relegate both Rodgers and Hammerstein to the generic <creator> element, resulting in lossy descriptive metadata:

```
<creator>Rodgers, Richard</creator>
<creator>Hammerstein, Oscar</creator>
```

By using the Saxon XSLT processor and a home-grown XLST transformation sheet that houses descriptive metadata crosswalking rules from MusicXML to MODS, the MXMLiszt platform is capable of automatically generating library-ready descriptive metadata from the source MusicXML files.

Finally, both the MODS metadata and the MusicXML files themselves create opportunities for search/retrieval via XQuery given their XML-based composition. Simply put, MusicXML assumes the progenitive role within the beta platform and XQuery the inspective.

It should be noted that the Open Score Format (OSF), an open and non-proprietary distribution, interchange and archive file format for digital scores[57], is specifically centered around MusicXML. It, too, holds potential as a preferred metadata format, though its capabilities reach far beyond simple descriptive metadata. Further investigation of OSF would seem to be in order for those interested in deploying MusicXML in the digital library context.

Initial Setup

It is by first running administrative PHP scripts that the MXMLiszt collection and search environment is created for one or many pre-existing MusicXML files on the server.

By describing the completion of an administrative "cycle" for a MusicXML file called "foo.xml", it is simpler to demonstrate via example the process depicted in the diagram above.

Administrative scripts available are as follows:

a. Setup Scripts

1. Normalize MusicXML
2. Generate MODS
3. Generate PDF
4. Generate PNG

[57] MusicXML Definition. (2010) Retrieved from http://www.recordare.com/xml.html

b. Search Related Scripts

1. Concatenate MODS
2. Concatenate MusicXML

c. Reporting Script

1. MODS remediation list

a. Setup Scripts

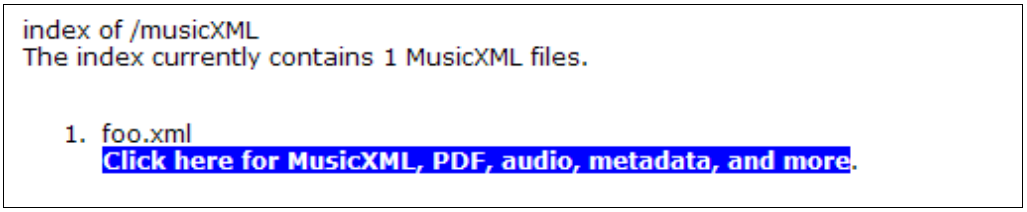
Once MusicXML files are placed on the server, setup scripts prepare the platform so that a user may browse the collection and have access to images and audio of the compositions.

1. Normalize MusicXML

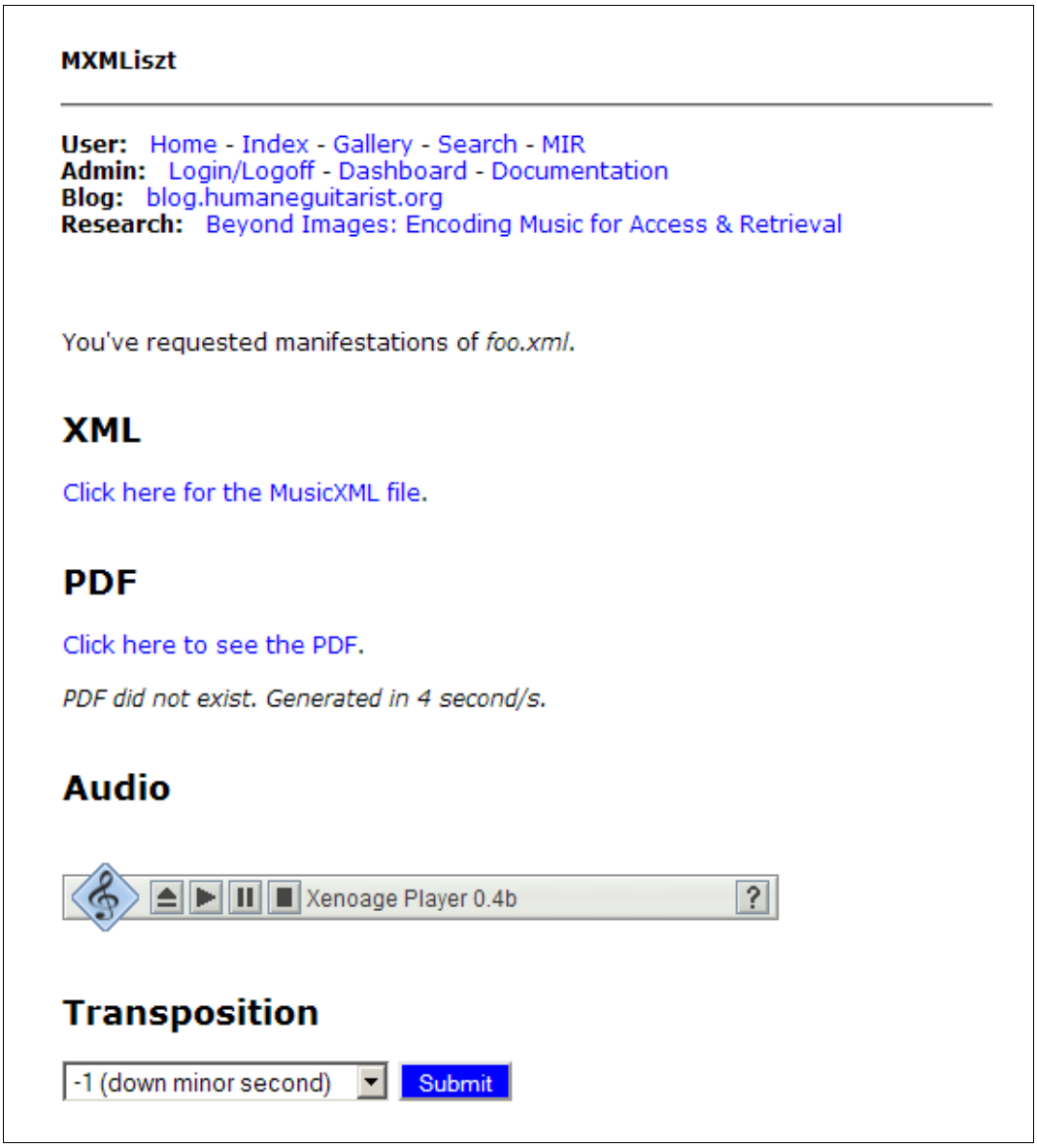
Variations occurs between MusicXML files exported from different notation software packages. Due to this variation, it seems necessary to "normalize" each MusicXML. That is to say, it seems necessary to import each MusicXML file - irrespective of which notation software it was generated from - and export it from the same application, namely MuseScore. This helps to standardize the MusicXML files, allowing for more consistency in regard to subsequent image creation by MuseScore and metadata creation via XSLT.

In a real working environment our test file "foo.xml" would now have been opened and re-saved by MuseScore.

MXMLiszt's "index view" option would now result in displaying the following:



Prematurely clicking on the link above would be less than ideal in that no descriptive metadata is yet available. Also, since no PDF is readily available one would have be be created "on the fly", slowing down the user experience:



Neverthless, preliminary functionality exists at this point allowing the user to see and hear the original composition as well as access transpositions of the piece.

2. Generate PDF

Though scripts a-2 through a-4 may be run in any sequence, it is best to generate PDFs first otherwise PDFs will be generated by user activity, adding to the amount of time a user has to wait before being able to view the PDF file.

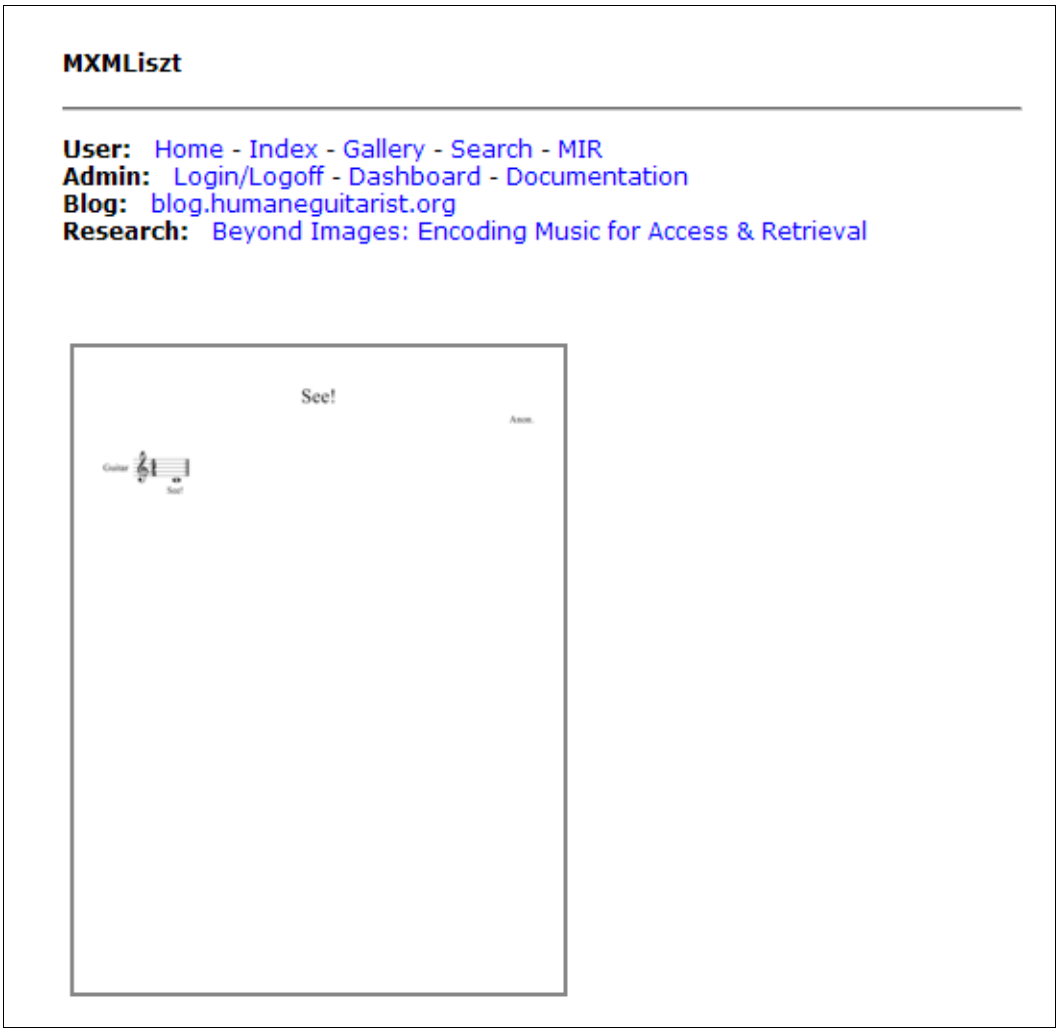
This script simply would instruct MuseScore to open "foo.xml" and render a file called "foo.pdf". If this PDF already exists, no new file will get rendered.

It should be notated that this script can be fairly time intensive for large amounts of MusicXML files in the collection.

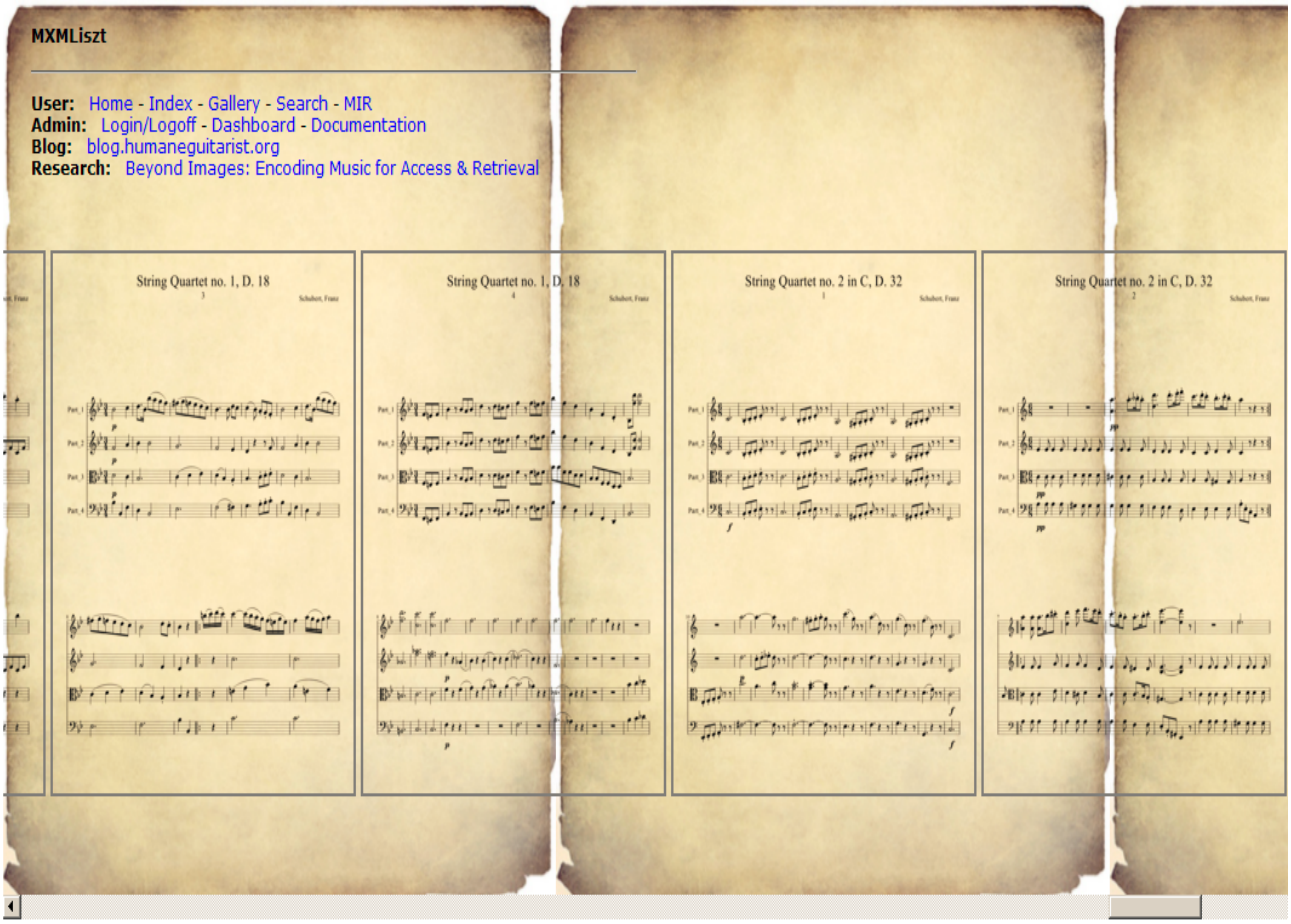
3. Generate PNG

Though not necessary and strictly an added "feature" of the platform, this script would call MuseScore to output a PNG image file of each page of each MusicXML file. It would then delete all but the first page. The ImageMagick software is then called to resize all remaining PNG files to a near thumbnail image, leaving a smaller preview image, "foo.pre.png". If this PNG file already exists, no new PNG files will get rendered for “foo.xml”.

This preview image can be viewed by hovering over a selection in the "index view" of the platfom or by using the "gallery view" of the platform (i.e. visual browsing):



The efficacy of this feature is more evident in cases where the online collection contains more than one MusicXML file:



The rational behind this feature was to emulate some print editions of sets of musical compositions in which the user is provided a "visual table of contents", a page containing the first measure or so of each composition within the edition, the "incipient", so that traits such as primary key signature and music texture can be gleaned for each composition within the edition without having to flip through the entire manuscript.

Given the additional processing, this script is naturally even more time consuming than the previous PDF script.

4. Generate MODS

As stated prior, it is via XSLT that one is able to automatically crosswalk descriptive metadata elements in MusicXML to MODS while remaining in the domain of XML technologies, as opposed to using a scripting language such as PHP, Python, etc. to achieve the same results.

The XSL transformation sheet used by the platform is called "mxml2mods.xsl" and seeks to create MODS XML files for the following primary descriptive properties about the composition:

titles (primary, alternate)
creators (composer, lyricist, arranger)

instrumentation

Additional elements include the unique, such as the URI identifier, and the fairly generic, such as the Library of Congress Subject Heading "sheet music". The encoding date of the normalized MusicXML file is also captured.

While the "mxml2mods.xsl" transformation stylesheet is too lenghty to include in its entirety, the excerpt below demonstrates that which would create the portion of the MODS records above for Rodgers and Hammerstein as "composer" and "lyricist", respectively. Furthermore, it also shows that the stylesheet will force the outputted MODS to end with a comment line expressing the following phrase: *Preliminary MusicXML to MODS metadata created via XSLT*. The inclusion of this phrase at the end of the MODS file allows subsequent tracking of MODS records that are yet to be remediated by skilled metadata librarians.

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
version="2.0">

...

    <!-- CREATOR -->
        <xsl:for-each select="score-partwise/identification/creator">
     <!-- composer -->
            <xsl:if test="@type = 'composer'">
               <name type="personal">
                  <namePart>
                     <xsl:value-of select="." />
                  </namePart>
                  <role>
                     <roleTerm type="text"
authority="marcrelator">composer</roleTerm>
                  </role>
               </name>
            </xsl:if>
         </xsl:for-each>
      <!-- lyricist -->
         <xsl:for-each select="score-partwise/identification/creator">
            <xsl:if test="@type = 'lyricist'">
               <name type="personal">
                  <namePart>
                     <xsl:value-of select="." />
                  </namePart>
                  <role>
                     <roleTerm type="text"
authority="marcrelator">lyricist</roleTerm>
                  </role>
               </name>
            </xsl:if>
         </xsl:for-each>

   ...

      <xsl:text disable-output-escaping="yes">&#xA;&lt;!--Preliminary
MusicXML to MODS metadata created via XSLT--&gt;</xsl:text>
```

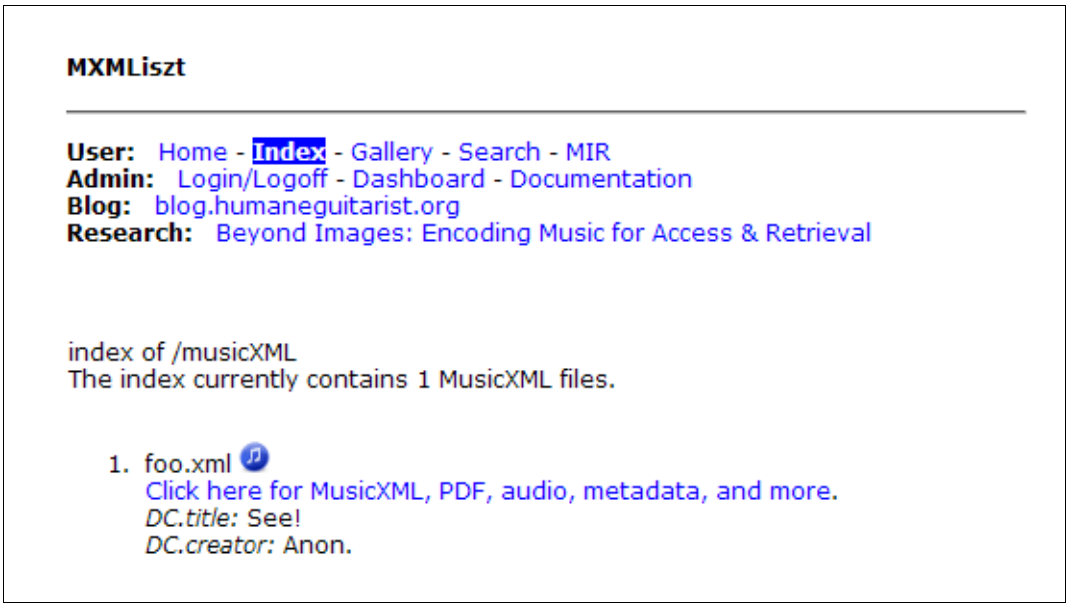
...

</xsl:stylesheet>

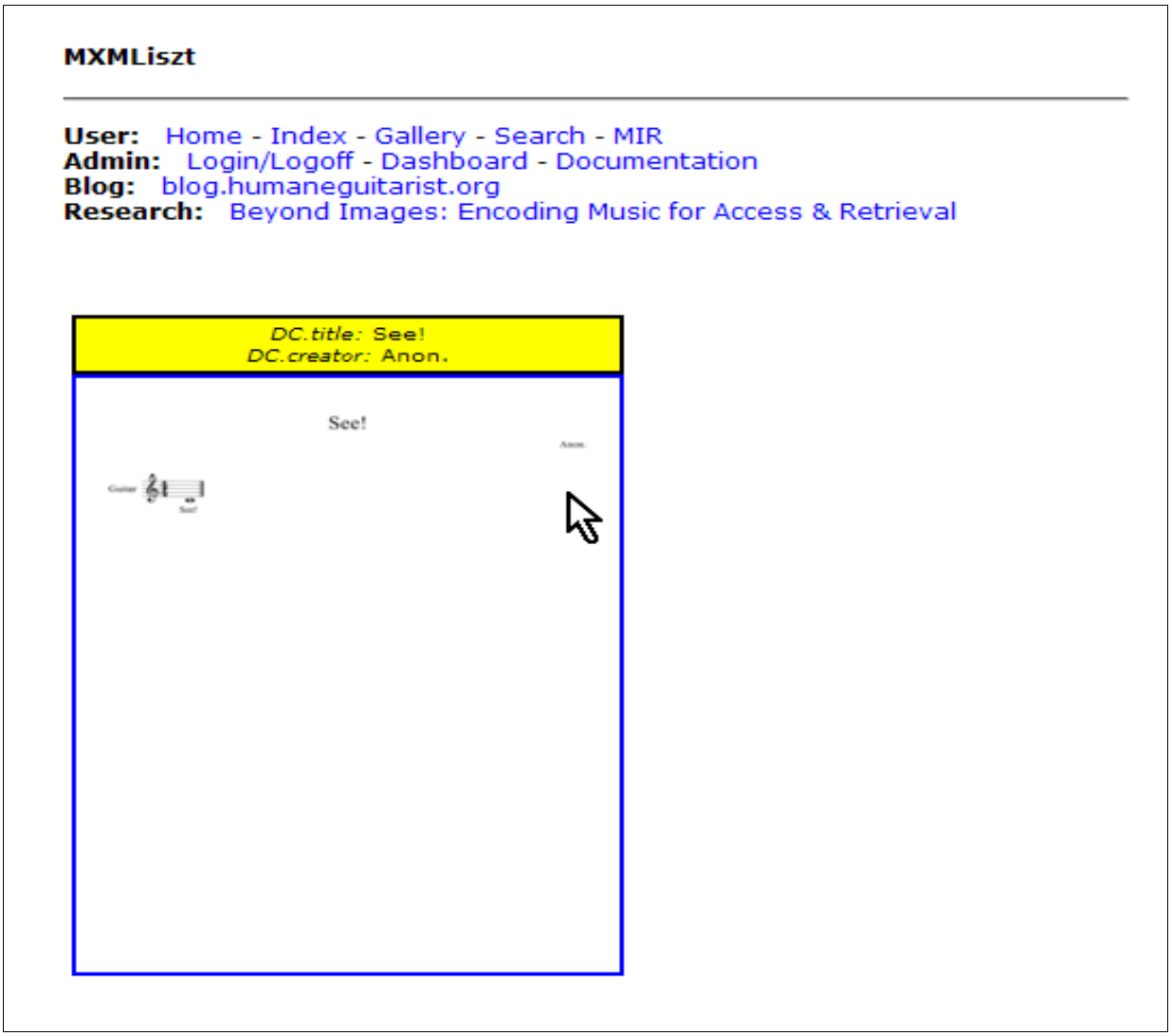
Getting back to the example of "foo.xml", running the Generate MODS script would create a file entitled "foo.mods.xml", pasted below:

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="../mods.xsl"?>
<mods xmlns:mods="http://www.loc.gov/mods/v3/">
  <titleInfo>
    <subTitle>See!</subTitle>
  </titleInfo>
  <name type="personal">
    <namePart>Anon.</namePart>
    <role>
      <roleTerm type="text" authority="marcrelator">composer</roleTerm>
    </role>
  </name>
  <typeOfResource>notated music</typeOfResource>
  <subject authority="lcsh">
    <topic>sheet music</topic>
  </subject>
  <abstract type="instrumentation">Scored for Guitar.</abstract>
  <originInfo>
    <dateModified encoding="w3cdtf">2010-04-14</dateModified>
  </originInfo>
  <identifier type="uri">/MXMLiszt/musicXML/foo.xml</identifier>
</mods>
<!--Metadata last modified on 2010-04-14T20:21:37.953-05:00-->
<!--Preliminary MusicXML to MODS metadata created via XSLT-->
```

The "index view" of the platform would now show descriptive metadata as such:



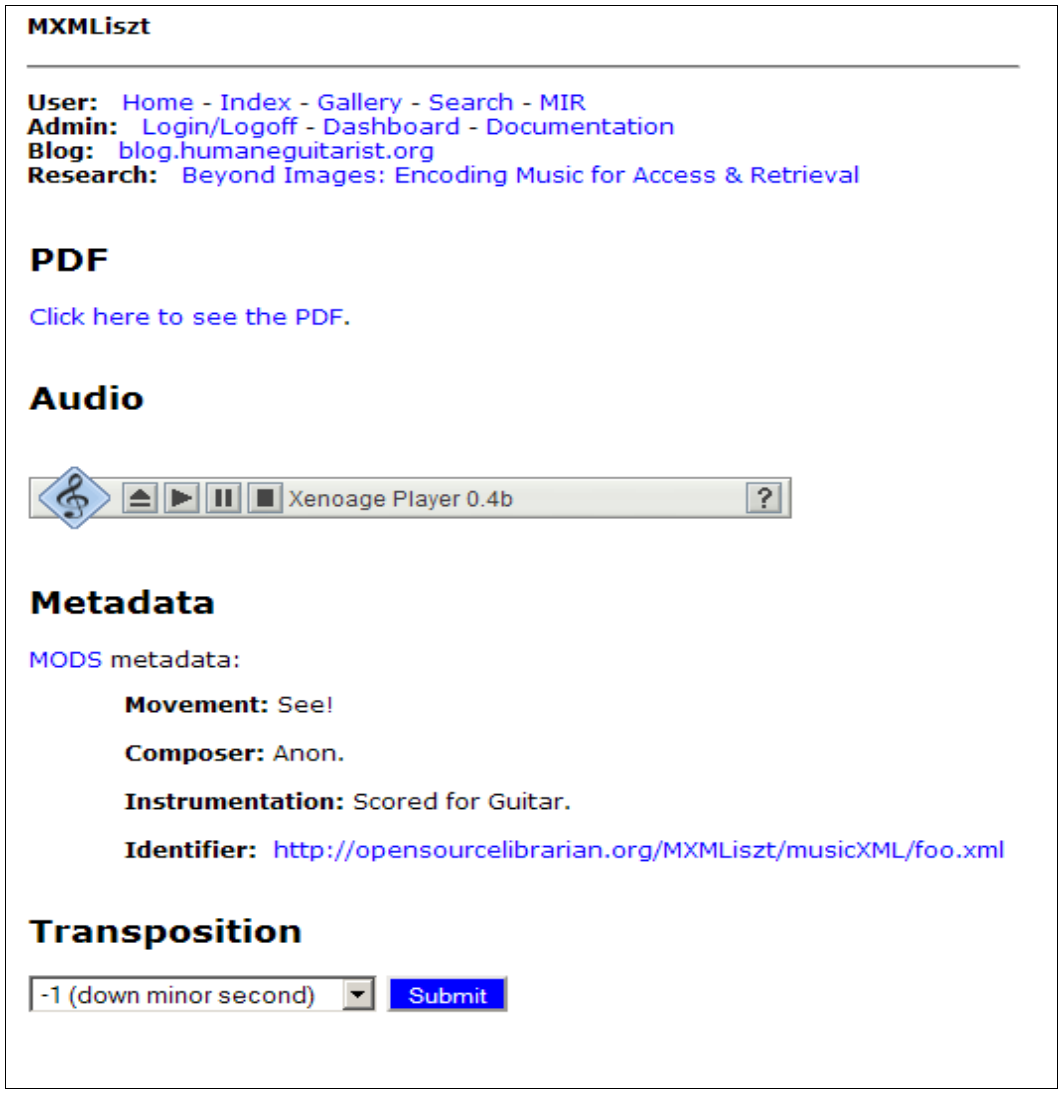
While the Gallery view of the of the platform would show the metadata encapsulated in a top-centered yellow box only when the user hovers over the preview image of the composition:



Note that the metadata, though MODS, is showing up on these pages as Simple Dublin Core, with the "DC" prefix. This was done for two reasons:

1. Ease of programming, given that the platform exists solely for the purpose of demonstration to the library community and
2. because it seemed simpler from a mere browse perspective to have the metadata be fairly "filtered down" to Dublin Core.

Clicking on the selection from either the Index or Gallery views will, however, now reveal the MODS record as shown below:



MXMLiszt uses an XSL stylesheet to display the MODS record in a user-friendly way as above, though librarians and information specialists can open the MODS record in a separate browser tab and view the page's source code to view the full MODS XML record.

At this point, everything is ready from a user browse perspective as they will have access to MusicXML, descriptive metadata, and image files. The Xenoage player will produce sound in real-time and thus no sound files need to be made.

b. Search Related Scripts

The two scripts related to preparing the platform for search/retrieval and MIR can be run in any order.

1. Concatenate MODS

In order to create an adequate search environment for the descriptive metadata (MODS) on the platform, a "virtual" database file for all descriptive metadata is

created via this script. In other words, the data from all MODS files are concatenated into one "super" file as such:

```
<hyperMODS>
        <hypoMODS file="foo.xml">
                1st MODS document
        </hypoMODS>
        <hypoMODS file="foo2.xml">
                2nd MODS document
        </hypoMODS>
      ...
<hyperMODS>
```

It is this hyperMODS file that is queried via XQuery searching and the BaseX XQuery processor. This allows for the processor to access only one XML file as opposed to iterating through all individual MODS files on the platform, as the latter option would yield unbearable long retrieval times.

2. Concatenate MXML

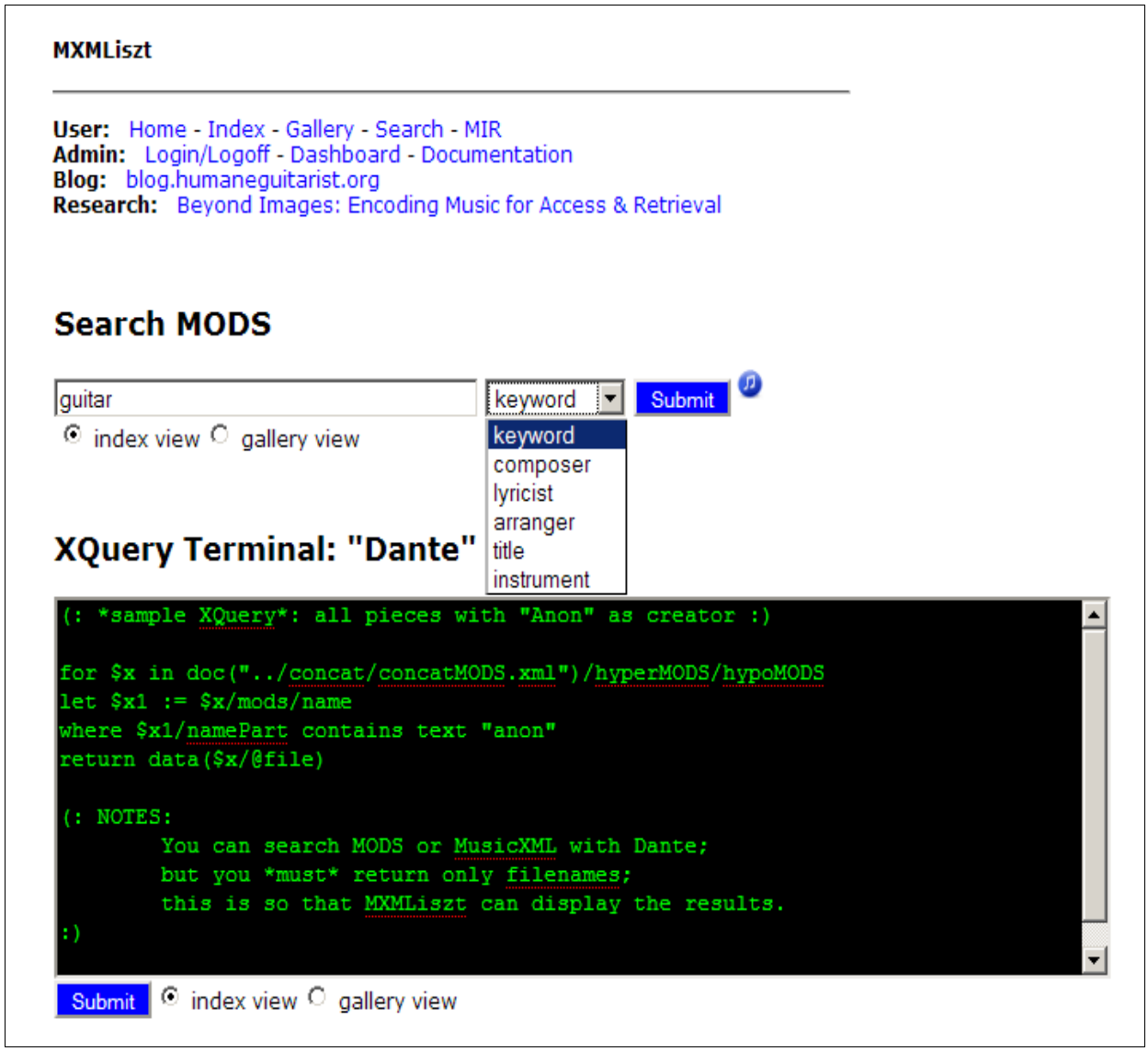
Similarly, this script concatenates the data for all MusicXML files on the server for quicker search and retrieval in regard to MIR functionality. Given how much larger MusicXML files are in contrast to MODS files, this concatenated "super" MusicXML file can be extremely large, depending on the amount of MusicXML files in the online collection.

Note the similar structure to the hyperMODS file above:

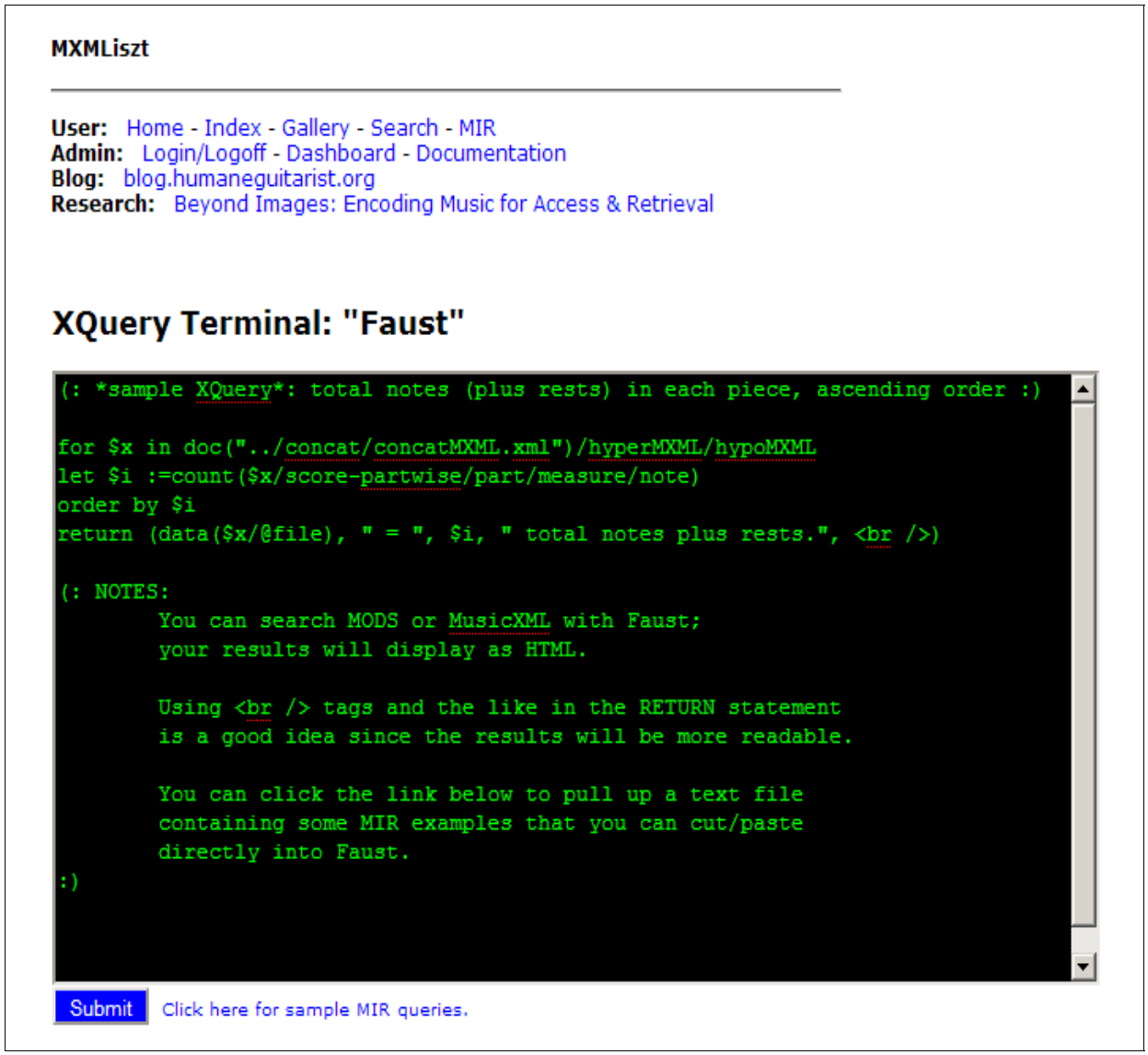
```
<hyperMXML>
        <hypoMXML file="foo.xml">
                1st MusicXML document
        </hypoMXML>
        <hypoMXML file="foo2.xml">
                2nd MusicXML document
        </hypoMXML>
...
</hyperMXML>
```

By using administrative scripts to prepare these large concatenated XML files the platform is initialized for search/retrieval and MIR across the entire online collection.

The descriptive metadata can be searched by a user-friendly drop-down box or by manual entry of valid XQuery syntax into a terminal style input box - nicknamed "Dante" for one of Liszt's symphonic works. In fact, MIR searches can also be initiated via terminal input, however this terminal is fairly restrictive in that query results must return only MusicXML filenames so that the platform can display the results in either the Index or Gallery view styles.



A less restrictive terminal ("Faust", also for one of Liszt's symphonies) allows advanced users to perform MIR (and even descriptive metadata) searches and return results in the output style of their choosing. Currently, intermediate users can access a link to a plain text file with pre-written MIR queries that can be cut and paste into the terminal.



Obviously, both terminal input boxes are primarily intended for advanced users who understand both XQuery fundamentals and the file and folder structure of the MXMLiszt platform. Ideally future versions of the platform would support graphics-based input of musical information, such as pitches and durations.

c. Reporting Script

As mentioned, the XLS transformation that produces initial MODS records from the the MusicXML files ends the outputted MODS file with a commented-out line that reads:

<!--Preliminary MusicXML to MODS metadata created via XSLT-->

The "MODS remediation list" script simply reports to interested administrators (i.e. metadata librarians) which MODS files within the platform do and do not contain this line, the principle being that remediation of MODS records by librarians would include

the removal of this line, hence allowing subsequent remediated MODS records to be reported as remediated.

G. Conclusion

The symbolic representation of music (SMR) is a complex affair, often appearing to match the complexities of that which it aims to represent. This has not, however, stopped active volunteer and professional developers from continuing efforts in regard to encoding SMR digitally and developing mechanisms to search Digital SMR formats through a variety of textual, graphical, and interactive methods.

Research in regard to high-level Music Information Retrieval activities is typically carried out by computer scientists with musical knowledge or musicians with computer science knowledge. Conversely, libraries and librarians seem to be contributing little even though advances in the field have the potential to serve patrons and research efforts of all types. It is incumbent upon the library community to recognize the symbolic nature of their sheet music collections and to move beyond images so as to provide users with better access and retrieval options in regard to musical content.