Music Notation as a MEI Feasibility Test

Baron Schwartz
University of Virginia
268 Colonnades Drive, #2
Charlottesville, Virginia 22903 USA
bps7j@cs.virginia.edu

Abstract

This project demonstrated that enough information can be retrieved from MEI, an XML format for musical information representation, to transform it into music notation with good fidelity. The process involved writing an XSLT script to transform files into Mup, an intermediate format, then processing the Mup into PostScript, the de facto page description language for high-quality printing. The results show that the MEI format represents musical information such that it may be retrieved simply, with good recall and precision.

1 Introduction

Most uses of musical information require storage and retrieval, usually in files. Unfortunately, files created for one purpose may not be usable for others, and the files may be incompatible even when the uses are similar. This is true of the many file formats that exist. Because each fails to address some need adequately, and because the formats are not extensible, there is a plurality of formats, leading to vast bodies of computer-encoded knowledge that cannot be shared effectively. It is often possible to translate between formats, but unless the data is trivial, some information is usually lost.

The need for a universal format is self-evident; one would like to encode data once and use it many times for many purposes. Perry Roland, a researcher at the University of Virginia's Digital Library project, is developing MEI (Music Encoding Initiative) into such a format (Roland, 2002). MEI uses XML (Extensible Markup Langauge), a universal data-interchange syntax, to define a musical-data encoding.

MEI is different from existing formats in that it supports all of the widely identified "domains" in which music exists: visual, gestural, performance, and analytical. Most existing languages only address the visual domain. The other major XML format, MusicXML, is designed as a notational interchange format (Good, 2001). MEI defines music as an abstract concept, something more general than either notation or performance, but does not leave important domains, such as notation, up to the implementer. The MEI format's design also allows a processing application to ignore information it does not need and extract the information of interest. For example, the format includes information about page layout, but a program for musical analysis can simply ignore it.

Generating printed music notation is a revealing test of MEI's capabilities because notation requires more information about the music than do other domains. Successfully creating notation also confirms that the XML represents the information that one expects it to represent. Because there is a relationship between the information encoded in both the MEI and Mup files and the printed notation, the notation serves as a good indication that the XML really does represent the same music as the notation.

2 Methods

Music notation is so complicated to create that it was infeasible to write a program to view MEI as notation directly for this project. However, it was a fairly simple matter to transform MEI into Mup, a plain-text format that represents commands to the Mup interpreter. This has its disadvantages: it means learning another notation file format, it involves several lossy steps, and it subjects the resulting notation to any constraints of the notation software. However, it is a reasonable way to check MEI's basic capabilities.

Because MEI files are written in XML, the easiest way to transform them to Mup notation is with XSLT , a functional programming language written in XML syntax and designed for XML transformations. This involves writing XSLT templates, which are mappings that specify the desired output format for a given input, for each type of element in the MEI file. The templates define the transformation to the equivalent Mup notation; thus the relationship between MEI and Mup syntax is defined formally by the XSLT script. Unfortunately, the Mup syntax is not defined formally, so it is not possible to verify formally that MEI is equivalent to the resulting PostScript notation file.

3 Test Pieces

To test the transformations, it was necessary to transform some complicated pieces. The following pieces are used to test the capabilities of various encoding formats (Selfridge-Field, 1997). The examples transformed are rendered very similarly to the original. In most cases, the XML files from which these pieces

were produced are not specially "tweaked," other than scaling the notation to fit correctly onto the page. Exceptions are noted in the text.

Perry Roland encoded these test pieces and furnished them for testing. They may be found in several formats at the MEI website,
`http://dl.lib.virginia.edu/bin/dtd/mei/`.

3.1 The Mozart Trio

This example is taken from the second trio section of Mozart's Clarinet Quintet. The challenge is transposing the first staff, which is notated in C Major but, since it is played on an A clarinet, is actually in A Major. It is very well rendered on the whole. There is a phrase or tie that begins on the last note of the first staff, but since there is no note for it to extend to, Mup ignores this. Mup also places some phrase marks oddly, such as the phrase mark on the tuplet, and has trouble with phrases that cross system breaks.

3.2 The Mozart Piano Sonata

This example has mixed durations within chords, grace notes preceding chords, and arpeggiated grace notes. The arpeggio on staff 1 in the first measure also crosses voices. In this case, Mup's default placement of the slurs on the left-hand grace notes is not optimal, so the slurs are encoded with explicit endpoints and curve values. Mup cannot slur to an entire chord, as in the right-hand part in measure 4, so those are also placed explicitly.

3.3 The Saltarello

This piece demonstrates multiple endings. The sections were originally numbered with a small number above the staff in the first measure of each section. This information is not encoded in the XML, but could have been.

3.4 The Telemann Aria

This example demonstrates lyrics and multiple voices. Many of the notes have small ("cue" size) heads, and the voices are extremely complex. Mup's default note placement does not match the original in some places, but little can be done about this without hand-editing the Mup code after it is transformed. It is necessary to remove the fourth voice from the piece before Mup will process it. Mup is limited to three voices, but additional voices can be notated by placing the notes manually.

3.5 Unmeasured Chant

This example is difficult to render with Mup because there is no meter signature. Mup requires each bar to have exactly the right number of notes, so it is necessary to change the time signature, instruct Mup not to print the time signature, and print invisible bars.

3.6 The Binchois *Magnificat*

This example demonstrates several tricky layout problems Mup does not handle gracefully. In particular, there is no way in Mup to get a stem to point down on the right-hand side of a note. It was necessary to use a small macro to get Mup to draw a line in the appropriate place. This macro is embedded in the XSLT. Some of the other interesting features of this piece are the absence of note stems in the first measure, the absence of a second staff in the first measure (Mup displays the staff, but there is none in the original), and a number of editorial elements, such as editorial accidentals. The first measure is also in $\frac{5}{4}$ time, but the time signature is hidden until it changes to $\frac{3}{4}$ in the second measure. The key signature and clefs should be placed in the second measure as well, but Mup's default behavior does not re-display the clef and key signature in bar 2. This could be encoded explicitly in the XML to force Mup to display it as required.

4 Conclusion

As the examples demonstrate, MEI represents musical data well enough to generate acceptable printed music notation. Because MEI is written in XML format, the information in a MEI file is accessible and easy to manipulate. This may mean that MEI can be used for many other, far more general, application domains. For example, using XSLT, it is relatively simple to extract those notes having a certain pitch followed by another note at a certain interval. It would also be easy to extract bibliographical information from a MEI file (and highly efficient as well, since this information is at the beginning of the file). Since XML is supported by a wide variety of software, writing tools to work with MEI files should also be greatly eased.

The transformation itself is simple, efficient, and demonstrates good recall and precision in the retrieval process. The transformation itself is straightforward, and merely involves writing XSLT templates for each element in MEI; with few exceptions, the MEI elements map easily to Mup notation. The retrieval demonstrates good recall, since the music notation is clearly the correct result of the transformation, and good precision, which can be inferred intuitively from the fact that the music notation is indeed that information for which the transformation queried.

Future research should test the MEI format for other purposes, such as the analytical domain, to assess its feasibility as a universal musical data encoding format. Transformations to and from other formats would also enhance its usefulness as an interchange format. An obvious transformation would be the reverse of the one described in this paper, from Mup to MEI. This may be difficult because of the informal nature of the Mup format, but should be possible with a two-step process to first "canonicalize" the Mup notation, then transform it to MEI.

References

Extensible Markup Language (XML) (n.d.). Retrieved August 11, 2003 from `http://www.w3.org/XML/`

Good, Michael (2001). MusicXML for Notation and Analysis. In Hewlett, Walter B. and Eleanor Selfridge-Field (Eds.), *The Virtual Score: Representation, Retrieval, Restoration*: Vol. 9. (pp. 113–124). Cambridge, MA: MIT Press.

Roland, Perry (2002). The Music Encoding Initiative (MEI). In Haus, Goffredo and Maurizio Longari (Eds.), *Proceedings of the First International Conference on Musical Applications Using XML* (pp 55–59). Milan: State University of Milan.

Selfridge-Field, Eleanor (1997). Introduction: Describing Musical Information. In Selfridge-Field, Eleanor (Ed.), *Beyond MIDI: The Handbook of Musical Codes* (pp. 3–37). Cambridge, MA: MIT Press.

XSL Transformations (XSLT) (n.d.). Retrieved August 11, 2003 from `http://www.w3.org/TR/xslt`