Music Encoding Initiative (MEI) DTD

Perry Roland
Digital Library Research & Development Group
Alderman Library, P.O. Box 400149
University of Virginia
Charlottesville, VA 22903-4149 USA
+1 434 982 2702
pdr4h@virginia.edu

ABSTRACT
This paper provides a technical introduction to the Music Encoding Initiative (MEI) DTD currently under development by the author. It is consciously modeled on the highly successful Text Encoding Initiative (TEI) DTD. The primary purpose of the MEI DTD is the creation of a comprehensive yet extensible standard for the encoding and transmission of music documents in electronic form.

KEYWORDS
Document Type Definition (DTD) design, eXtensible Markup Language (XML), Music Encoding Initiative (MEI), music notation, music representation, Text Encoding Initiative (TEI)

1 INTRODUCTION

This paper provides a technical introduction to the Music Encoding Initiative (MEI) DTD currently under development by the author. The DTD is consciously modeled on the highly successful one created by the Text Encoding Initiative (TEI). The primary goal of TEI was the creation of a comprehensive yet extensible standard for the encoding and transmission of textual documents in electronic form. The MEI shares the same goal for music documents.

The MEI DTD relies upon design principles found in SGML and XML literature [Jelliffe][Maler][Megginson] and in music representation literature [Huron][O Maidin]. For a more detailed discussion of these principles, the reader is encouraged to read the author's MAX2002 paper [Roland].

2 BASIC DESIGN

The MEI DTD adopts the following basic structure. The author proposes this structure as the foundation for the DTD to be designed by the MUSICNETWORK. The salient features of this structure are discussed below. The complete MEI DTD is available from http://www.people.virginia.edu/~pdr4h/.

```
<!-- bar-level events -->
<!ENTITY % m.events.log
      "beam|beatrpt|btrem|chord|ftrem|note|pad|rest|space|tuplet">
<!-- events which potentially cut across the principal hierarchy -->
<!ENTITY % m.events.control
      "arpeg|beam2|bend|dir|dynam|gliss|hairpin|harm|lyrics|midi|
      mordent|octave|pedal|phrase|reh|slur|tempo|tie|trill|turn">
<!-- graphic primitives -->
<!ENTITY % m.primitives
      "curve|line|symbol|text">
<!ELEMENT mei (meihead, music)>
<!ELEMENT meicorpus (meihead, mei+)> <!-- *Not yet implemented!* -->
<!ELEMENT meihead (meiid, filedesc, projectdesc?, editorialdecl?, profiledesc?,
      revisiondesc?, sourcedesc*)>
<!ELEMENT music (front?, (body|group), back?)>
<!ELEMENT group ((music|group), (music|group)*)>
<!ELEMENT body (mdiv, mdiv*)>
<!ELEMENT mdiv (score*, parts*)>
<!ELEMENT score ((section, ending*)+ | (bar, (pb|sb)?)+)>
```

```
<!ELEMENT pb EMPTY>
<!ELEMENT sb EMPTY>
<!ELEMENT parts (part)+>
<!ELEMENT part ((section, ending*)+ | (bar, (pb|sb)?)+)>
<!ELEMENT section ((section, ending*)+ | (bar, (pb|sb)?)+ | (strophe, strophe+))>
<!ELEMENT strophe ((section, ending*)+ | (bar, (pb|sb)?)+ | (strophe, strophe+))>
<!ELEMENT ending  ((section+ | (bar, (pb|sb)?)+) | (strophe, strophe+))>
<!ELEMENT bar ((staff+ | (%m.events.log;)+  | msrest | msrpt | multirest | multirpt),
       (%m.events.control)*, (%m.primitives)*, annot*)>
<!ELEMENT staff (layer+ | (%m.events.log;)+ | msrest | msrpt)>
<!ELEMENT layer ((%m.events.log;)+ | msrest | msrpt)>
```

The `mei` document element divides the markup into two major parts – meta-data and data. It is generally agreed that a clear separation between data and meta-data is highly desirable. Separating these is the primary goal of DTD design since the separation itself reflects the worldview of the DTD creator. Practically speaking, separating these allows the meta-data to be shared with other entities, both internal and external, to which it also applies. There is a separate DTD for the `meiheader` entity, which allows the header to stand alone, for example, in a meta-data catalog. All file identification and cataloging information is contained in the header which is modeled after the ISBD(G) standard [IFLA]. In addition, the header may contain a declaration of editorial practice and a file revision history. The other possible document element, `meicorpus`, makes it possible to group several `mei` instances each with its own header.

Since MEI takes a broad view of a music document, which includes textual matter often found in a modern critical or historical edition of music, the DTD has `front` and `back` matter elements that provide basic logical and presentational text markup functionality. This approach gives control of the text and notation to the encoder of the MEI instance, which embedding notation in another text format will not do.

The `group` element facilitates the creation of a collection of `music` elements, each of which represents a distinct work not requiring a complete meta-data header. A good example of this phenomenon is a collection of songs by different composers issued under a single title. Basic meta-data for each work in the collection may be encoded in its front matter or in the `sourcedesc` element in the file header. As mentioned above, when a complete header is required for each member of a collection, the `meicorpus` element should be employed.

When a musical work can be broken into high-level, discrete, linear segments, such as movements in the case of a symphony, the `body` element may contain multiple `mdiv`, or musical division, elements. An `mdiv` is the highest-level indication of the structure of the composition. A single `mdiv` indicates a single-movement work.

The `mdiv` element may contain one or both of two possible views of the music. The `score` element contains the traditional full open score while the `parts` element contains each performer's view of the score. These two views are necessary because it is not always possible or desirable to generate one from the other. The `score` and `parts` elements are placed here and not directly within the body element because score and part characteristics may change from `mdiv` to `mdiv`. For example, the second movement of a symphony may require different performing forces, and therefore different layout, than the other movements.

A `part` element contains an individual performer's view of the score. It is, in fact, a mini-score, requiring all the encoding features of a full score. The encoding of individual parts is practical when they don't share visual characteristics, such as typeface or layout, with the full score. Part-by-part encoding is also useful for music with non-aligning barlines or for accommodating software that requires staff-by-staff encoding. When assembly of a group of parts into a score is desired and there are non-aligning barlines, barlines that indicate points of alignment across all the parts should be marked as 'controlling'. `Part` elements in MEI have little to do with voice leading; that is, indicating how individual parts may be constructed from a score when parts lie on more than one staff. Voice-leading information can be encoded using the `next` attribute available on all event-type elements.

A part or score may be divided into linear segments or sections. `Section` elements usually function as a scoping mechanism for clef signs, key and meter signatures, as well as metronome, tempo, and expression markings. Said the opposite way, changes in these entities usually indicate structurally and analytically significant changes in the

music and, therefore, the need for a new containing element. The use of `section` elements also minimizes the need for backward scanning to establish context when the starting point for access is not at the beginning of the score. `Section` elements may also be used for other user-defined, i.e., analytical or editorial, purposes. `Section` elements may be arbitrarily nested to any desired level. When a `section` contains `section` elements the `expan` attribute may be used on the outer `section` element to encode the performance order of the inner `sections`.

A `strophe` element contains an alternative encoding. When used, there should always be at least two `strophe` elements, of course. Like `sections`, `strophe` elements may also be nested to any depth.

The `ending` element is a specialized instance of the `section` element that may not be recursively nested.

Within the `bar` element, events are modeled, not symbols. While events may have visual properties, modeling symbols places too much emphasis on presentational qualities and makes the markup less generally useful as a "music" (as opposed to a "notation") markup standard. `Staff` and `layer` elements are provided in order to indicate the division of the `bar`'s events into multiple data streams. Even though the terminology of music notation is used, these elements do not necessarily convey any layout information. As an alternative to the `staff` and `layer` elements, each event may be related to a particular stream of data via its own `staff` and `layer` attributes.

In printed music, page and system breaks create an artificial, that is, non-musical, hierarchy. Therefore, they are represented by empty elements.

3 OTHER FEATURES

While space does not permit a complete discussion of all of the features of the MEI DTD, it is important to note that the DTD is designed to be comprehensive. Therefore, it addresses many of the requirements stated in the call for XML based music notation solutions, including relationships between elements, cooperative creation and editing of music markup, navigation within the music structure as well as to external multimedia entities, the inclusion of custom symbols, etc. The author does not intend to suggest that the MEI DTD be adopted without modification. On the contrary, discussion of the merits and faults of the DTD is welcome.

REFERENCES

Huron, David. “Design Principles in Computer-based Music Representation” in Computer Representations and Models of Music. Alan Marsden and Anthony Pople, eds. New York: Academic Press, 1992.

International Federation of Library Associations and Institutions. ISBD(G): General International Standard Bibliographic Description. Annotated Text. rev. ed. UBCIM Publications New Series Vol. 6. 1992. http://www.ifla.org/VII/s13/pubs/isbdg1.htm. Accessed June 24, 2003.

Jelliffe, Rick. The XML and SGML Cookbook: Recipes for Structured Information. Charles F. Goldfarb Series on Open Information Management. Upper Saddle River, NJ: Prentice Hall PTR, 1998.

O Maidin, Donncha. “Representation of Music Scores for Analysis” in Computer Representations and Models of Music. Alan Marsden and Anthony Pople, eds. New York: Academic Press, 1992.

Maler, Eve and Jeanne El Andaloussi. Developing SGML DTDs: From Text to Model to Markup. Upper Saddle River, NJ: Prentice Hall PTR, 1996.

Megginson, David. Structuring XML Documents. Upper Saddle River, NJ: Prentice Hall PTR, 1998.

Roland, Perry. “The Music Encoding Initiative (MEI) ” in Proceedings: First International Conference on Musical Application using XML. September 19-20, 2002. Goffredo Haus & Maurizio Longari, eds.