Using XML for Score Representation

Margaret Cahill

Centre for Computational Musicology and Computer Music
Department of Computer Science and Information Systems
University of Limerick
margaret.cahill@ul.ie

ABSTRACT

XML is a subset of SGML (ISO 8879), the international standard metalanguage for text markup systems. It has been widely used in data representation and interchange formats. This research proposes to investigate the use of XML for musical score representation.

1. INTRODUCTION

XML (Extensible Markup Language) is a markup language defined by the World Wide Web Consortium. It allows for the creation of user defined markup languages for storing structured data in text files and is therefore often referred to as a meta-language [1][2]. It has been used to define languages for topics as wide-ranging as chemicals and electronic banking services [3]. It is also widely used in web and database applications. A number of related standards have also emerged for formatting and displaying XML documents: XSL (Extensible Stylesheet Language) and XSLT (XSL Transformations) and for referring to and linking to specific parts of documents: XPath, XPointer and Xlink.

This paper focuses on investigating the use of XML to represent music. The language has been successful in representing textual data and we should ".. examine the extent to which techniques developed for text can be applied to music."[4] It is first important to define what exactly is to be represented. Music means many different things to different people. Do we mean a live musical performance, a score, a MIDI file, or an audio file downloaded from or streamed over the web? "Within our musical culture, what is meant by the term "music" is, as often as not, the score rather than the performance"[5]. While this may not be strictly true in the present era of downloadable and streaming web files, "common music notation is the cornerstone of all efforts to preserve a sense of the musical present for other and later performers and listeners"[6].

Some methods of representation, including SMDL [7], adopt a broad interpretation of music e.g. including frequency, note-name, and MIDI note number to represent pitch. When such an approach is taken it is usually difficult to represent a piece of music for a specific application. The opposite approach is taken here. A markup language defined in XML is proposed for the representation of the musical score. Once the score is described in an XML document it can be used as an interchange format to and from other music representations. If the score is represented, the core of the musical piece has been captured and can then be built upon.

2. OBJECTIVES

There are a number of considerations to be taken into account when designing a score representation. Among the objectives are completeness, objectivity [7], extensibility and readability. There are a number of levels of detail one could include when recording a score [8]. The goal of completeness involves recording all the important musical details of the score. Certain spatial elements and visual elements that do not have musical implications can be omitted (e.g. the location of the title on the page, the thickness of the stave lines). Objectivity is very important when defining the description mechanism for the score. Personal interpretations of musical symbols at this stage may lead to a misleading representation and unreliable data. The aim is to record the details as they are seen on the printed page, not to make judgments on the meaning of the symbols.

3. USING XML

XML involves creating a language to describe data. These descriptions come in the form of tags, much like HTML. XML elements are created by the use of an opening tag, such as <Name>, followed by the element content which can be text or other elements, and ending with a matching closing tag such as </Name>. Elements can be empty. Figure 1. shows an example of using XML to store a person's name.

```
<Name>
<FirstName>Mark</FirstName>
<LastName>Jones</LastName>
</Name>
```

Figure 1. *XML used to store a name*

The user defines the structure of the language in the DTD (Document Type Definition). This DTD indicates the names of the elements to be used and the rules for combining these elements and alphanumeric data.

There are many reasons why XML suits the task of score representation. One of the benefits of using XML for any task is that is license-free, platform-independent and well–supported. This support includes free parsers for the most commonly used programming languages and platforms. XML has been used in the area of text representation and manipulation for a number of years now and the growing community of XML users supplies a valuable support network across numerous disciplines and applications.

XML documents can be easily read and understood. (This of course depends on the choice of markup). Much of the complexity and ambiguity involved with previously used representations can be avoided, for example

```
<clef>treble</clef>
```

There is no need to remember symbols or consult a manual to interpret the representation. [9] Ideally the XML document would be read only by computer programs and this would not be so important. Until reliable tools are available for encoding and reading these documents it is convenient to have easily recognisable elements. It also makes the job of programming such tools easier if it is clear what each element represents in the score.

XML separates description from implementation. This is particularly suited to representing the score, where the implementation could be a performance of the piece or a graphical display of the score.

XML is extendable. New elements can be added in the future. Software designed now can simply ignore elements it does not recognise and deal with only the elements it expects to find.

4. IMPLEMENTATION

Currently a language has been defined in XML for the representation of a monophonic score. The DTD (Documentation Type Definition) describes the hierarchy and structure of the language, see Figure 2. At present the DTD breaks the score into one or more parts. Movements or sections in music have not been taken into account at this stage of development. An encoding of the first bar of the well-known tune Twinkle, Twinkle is shown in Figure 3.

```
<!ELEMENT SCORE (PART+)>
<!ELEMENT PART ( INSTRUMENT?, CLEF, KEY, TIME,
                ((NOTE|REST)*, BARLINE?)* ) >
<!ATTLIST PART id ID #REQUIRED>
<!ELEMENT INSTRUMENT (#PCDATA)>
<!ELEMENT CLEF (#PCDATA)>
<!ELEMENT KEY (NO_SHARPS | NO_FLATS |NONE)>
<!ELEMENT NO_SHARPS (#PCDATA)>
<!ELEMENT NO_FLATS (#PCDATA)>
<!ELEMENT NONE EMPTY>
<!ELEMENT TIME (TEXT| (NUM,DENOM))>
<!ELEMENT NUM (#PCDATA)>
<!ELEMENT DENOM (#PCDATA)>
<!ELEMENT TEXT (#PCDATA)>
<!ELEMENT NOTE (DURATION,PITCH)>
<!ELEMENT DURATION (VALUE,NO_DOTS*)>
<!ELEMENT VALUE (#PCDATA)>
<!ELEMENT NO_DOTS (#PCDATA)>
<!ELEMENT PITCH (NAME,ACCID*,OCTAVE)>
<!ELEMENT NAME (#PCDATA)>
<!ELEMENT ACCID (#PCDATA)>
<!ELEMENT OCTAVE (#PCDATA)>
<!ELEMENT REST (REST_VALUE,REST_NO_DOTS*)>
<!ELEMENT REST_VALUE (#PCDATA)>
<!ELEMENT REST_NO_DOTS (#PCDATA)>
<!ELEMENT BARLINE (#PCDATA)>
```

Figure 2. *The DTD for representing musical scores.*

```
<SCORE>
  <PART id = "p1">
      <INSTRUMENT>recorder</INSTRUMENT>
      <CLEF>treble</CLEF>
      <KEY>
            <NO_SHARPS>1</NO_SHARPS>
      </KEY>
      <TIME>
            <NUM>4</NUM>
            <DENOM>4</DENOM>
      </TIME>
      <NOTE>
            <DURATION>
                  <VALUE>crotchet</VALUE>
                  </DURATION>
            <PITCH>
                  <NAME>G</NAME>
                  <OCTAVE>5</OCTAVE>
            </PITCH>
      </NOTE>
      <NOTE>
            <DURATION>
                  <VALUE>crotchet</VALUE>
            </DURATION>
            <PITCH>
                  <NAME>G</NAME>
                  <OCTAVE>5</OCTAVE>
            </PITCH>
      </NOTE>
      <NOTE>
            <DURATION>
                  <VALUE>crotchet</VALUE>
            </DURATION>
            <PITCH>
                  <NAME>D</NAME>
                  <OCTAVE>6</OCTAVE>
            </PITCH>
      </NOTE>
      <NOTE>
            <DURATION>
                  <VALUE>crotchet</VALUE>
            </DURATION>
            <PITCH>
                  <NAME>D</NAME>
                  <OCTAVE>6</OCTAVE>
            </PITCH>
      </NOTE>
      <BARLINE>light</BARLINE>
  </PART>
</SCORE>
```

Figure 3. *The XML document to represent bar 1 of Twinkle, Twinkle*

As already mentioned, there are a number of XML parsers available. The Apache Xerces [10] parser is a free commercial-standard parser available in Java and C++ that supports the DOM and SAX. DOM (Document Object Model) is a W3C standardised programming interface for XML documents. It takes a tree-based approach to handling documents. SAX (Simple API for XML) is an event-based widely used API for XML documents. The SAX interface of the Xerces C++ parser is currently used to parse the XML score documents.

Currently the parsed documents are used to create a C.P.N.View score object. C.P.N.View [11] is a score representation written in C++ that was developed in the mid-1990s. It implements a representation of scores as containers and provides iterators for use with algorithms. It is used to display textual and graphical versions of the score to assess the success of the XML document representation and parsing. Current work features the use of XSLT (Extensible Stylesheet Language Transformations) to convert the XML representation into other score representations. These would include ALMA, EsAC etc. Some research has been carried out into the representation of polyphonic scores but more work is needed in this direction.

5. PROBLEMS

Some concerns have arisen in the course of this research regarding the use of XML for this purpose. Although XML is easy to read, its tagged structure also makes it verbose. While the size of files may not be a significant factor as hard drives gain in capacity and lower in price, consideration must be given to the speed with which one can access and manipulate large databases of XML documents.

All XML element content eventually breaks down to nonmarkup text (PCDATA). There is no way in an XML DTD of specifying that one of the following options is appropriate and valid text and one is not:

<CLEF>treble</CLEF>
<CLEF>Hello</CLEF>

The only way to keep check of this is to write large amounts of error-checking code in the parsing and browsing software. ".. but, one major goal .. is to reduce the need for error-checking code"[11]. The use of attributes can improve this situation but there is a need maintain the structure and hierarchy obtained by the use of elements. XML Schemas are a relatively new addition to the growing list of XML standards and features more strong data typing, which may solve this problem.

6. IMPLICATIONS

This research has implications for many other areas of computer music research. The use of XML in Music Information Retrieval (MIR) is one obvious area. The descriptive nature of XML means that software can be created to search for individual items in the score representation or to deal with more complex queries involving melodic and rhythmic structure. Web distribution of music could also benefit. Consider the case where one user wishes to print a score from the web, another wants to listen to an audio version, and another wants to download a particular part in EsAC for use in their own computer music research. The relevant XML document could be stored on the web server in the form of a score representation. The user could then choose the format in which they would like to download the music and a software application translates between the stored XML document and the required format.

7. CONCLUSION

Further research is needed into the ability to represent polyphonic scores using XML. Polyphonic piano music in which melodic lines move between the piano staves seems a particular challenge. Temporally based features in a score also need attention, specifically items in a score whose meaning relies heavily on the time based structure of a performance of the score rather than the hierarchical structure of parts, notes, pitches etc. Such items would include musical symbols such as a crescendo that starts after the beginning of a note. It is hoped to resolve some of these issues through the continuance of research in this area.

8. REFERENCES

[1] Holzner, S., *Inside XML*, New Riders Publishing, Indiana, 2001, p.1.

[2] Roland, P., "XML4MIR: Extensible Markup Language for Music Information Retrieval", ISMIR 2000 Proc., 2000.

[3] St.Laurent, S. and Cerami, E., *Building XML Applications*, McGraw-Hill, New York, 1999, pp.46-49.

[4] Pickens, J., "A comparison of Language Modeling and Probabilistic Text Information Retrieval: Approaches to Monophonic Music Retrieval", ISMIR 2000 Proc., 2000

[5] Arnold, S., Boehm, C., Hall., C., "MuTaTeD! – Music Tagging Type Definition", Report, University of Glasgow, 2000, p3.

[6] Selfridge-Field, E., *Beyond MIDI*, MIT Press, London, 1997, p.4

[7] Ó Maidín, D., *A Programmer's Environment for Music Analysis*, Dissertation for The National University of Ireland 1995, pp.4-5.

[8] Dowland Page, S., *Computer Tools for Music Information Retrieval*, Dissertation for Oxford (Bodelian) 1988, and p.53.

[9] Huron, D., "Design Principles in Computer-Based Music Representation", *Computer Representations and Models in Music*, Academic Press, London, 1992, p.17

[10] http://xml.apache.org/

[11] Ó Maidín, D. "Common Practice Notation View User' Manual" Technical Report UL-CSIS-98-02, University of Limerick, 1998.

[12] Goldfarb, C., *The XML Handbook*. Prentice Hall, New Jersey, 2001, p.904.