

SUITABILITY OF MUSICXML AS A FORMAT FOR COMPUTER MUSIC NOTATION & INTERCHANGE

Stuart Cunningham

*Centre for Applied Internet Research (CAIR), University of Wales / NEWI
Plas Coch Campus, Mold Road, Wrexham, LL11 2AW, North Wales, UK
s.cunningham@newi.ac.uk*

ABSTRACT

Developments in the world of musical notation have produced a new computer file format, to be used for storage and interchange of musical information. This format is called MusicXML. MusicXML is a text-based, robust, and user friendly format, designed for interchange, analysis, and performance of musical information. By using XML as the framework for the development of the MusicXML language, the user is permitted great levels of data structuring and organisation. MusicXML is a strong format, which shows potential for being developed as a standard for musical notation interchange. Its current, and soon-to-be-available, capabilities provide a very useful format for musical notation, and with the correct market exposure and support in the industry, could be well on its way to becoming a standard format for musical notation and, particularly, interchange.

KEYWORDS

MusicXML, XML, music, notation, interchange

1. INTRODUCTION

There have been many methods used to represent musical notes and information on computers, since early experimentation with electronic musical sequences. These various methods have ranged from being highly prosperous formats, with widespread implementations, to the unsuccessful, little known, non-user friendly forms of strong musical semantics. One method of representation to have emerged over recent years is the MusicXML file format, which is based on the conventions of the eXtensible Mark-up Language (XML) format.

Very little published information is available about MusicXML, aside from the official documentation made available through the format's creators; Recordare LLC, in comparison to many of the other computer musical notation formats such as MIDI and NIFF. This paper discusses the significant features of MusicXML.

There has been no officially accepted standard file format to be used for the electronic representation of music notation, though the MIDI file format has become the ad-hoc standard over the years. However, MIDI does not provide the necessary musical semantic information to be used as a comprehensive format for music representation. The Standard MIDI File format was never intended to be used as a format for musical notation purposes, and so it is unsurprising that the format is unsuitable for the precise notation of musical information. Standard MIDI Files can produce pseudo notation by facilities provided by notation software that is acceptable for some purposes, but for precise and professional musical notation, MIDI is entirely unsuitable. The only notable attempt to create a standard file format was NIFF, which was never adopted as an official standard. This document represents initial research into MusicXML and its suitability to become a standard format for the interchange of musical notation data.

2. BACKGROUND OF MUSICXML

2.1 MusicXML Foundations

Michael Good, the CEO and founder of the company Recordare LLC, first started to develop MusicXML in February 2000. MusicXML is in continual development and is currently in version 0.9 as of the 1st December 2003. MusicXML is used for the representation of common Western music notation from methods of the 17th century onwards (Good, 2001). The main development goals of MusicXML were to create a file format for musical representation and interchange that would be Internet-friendly and in a non-binary format. These two specifications pointed towards the eventual implementation of XML as the template to create this new music interchange format. XML was already a popular format in use on the Internet, with plenty of authoring and application support tools available, and had the advantage of being text-based. This produced a format with a high level of human readability and existing support tools. MusicXML attempts to provide a method for music interchange that would be supported and implemented by software companies in their music products.

MusicXML is based around the standards and formats defined in the XML meta-language. XML is a subset of SGML (Standard Generalised Mark-up Language). XML documents are containers for information (St Laurent, 2000), and MusicXML has been designed for the representation of scored or notated music, which provides for digital storage and archiving, as well as the sharing and interchange of musical data. Creating a document using XML allows separation of data content, as well as a file structure definition that is controlled by creating and referencing to, a Document Type Definition (DTD). This defining of information about other information is the reason why XML is often referred to as a meta-language. MusicXML currently employs DTD files as a control mechanism for documents, as opposed to the emerging replacement for the DTD files; the XML schema.

2.2 Aims of MusicXML Development

The need for one standard interchange format for computer musical notation has long been recognised by music notation experts (Castan et al, 2001).

The benefits of creating one standard for all developers and users to adhere to has long been something that industries have strived towards, and can be seen by the large amount of standards making bodies that are in existence worldwide, such as the IEEE, ISO, and BSI. By employing one standard for the interchange of musical information, the transportation and sharing of musical data would become regulated, and improve communication and sharing among composers and musicologists. Commercial, and non-commercial, software applications would be able to read and write compositions to a standardised format. This would provide a method of representing work that could be transported, read, and edited by others with a music notation software tool. It would be expected that major software manufacturers and small developers alike would provide support for the standard format in their packages as the norm.

Another key area that will benefit by the implementation of MusicXML is the market for downloadable sheet music. MusicXML would solve problems of compatibility and ease the limitations currently imposed by the use of proprietary formats and the other alternative of employing music representations that contain no revisable musical data, other than the typescript. MusicXML hopes to do for online sheet music and music software what MIDI did for electronic musical instruments (Good, 2001). Currently, the main formats used for the distribution of sheet music via the Internet, are proprietary file formats for notation applications, and the Adobe PDF format. While these formats are useful under certain conditions, a standard format which provided the ability to be transported and used regardless of the notation software tool would be exceptionally useful, and as well as being printed, could be updated and revised with these software tools, as well as being used for musical performance and playback.

3. EMPLOYING MUSICXML

3.1 MusicXML Notation Methods

XML is a text-based language, meaning that organised and structured documents can be created and understood relatively easily. Although there are software tools that can aid the creation of XML documents, a simple text editing application such as Notepad in Windows, SimpleText on Macintosh OS or any one of many UNIX / Linux text editors can be used to produce well formed XML documents, provided the user is familiar with the syntax.

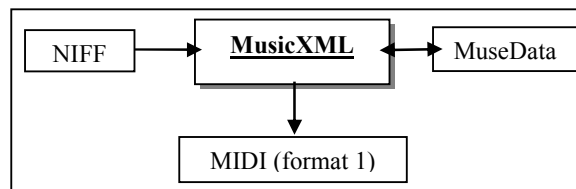
Although there have been previous attempts to create XML based music notation formats, none ever became successful and this is generally because these other formats had been too simplistic, and did not provide enough scalability and a diverse range of notational elements. Previous formats using XML also suffered a lack of commercial support from the music software industry because they did not represent enough aspects of musical notation to be of any great use. At the other end of the complexity spectrum, there had been the creation of SMDL (Standard Music Description Language), which was a language based on SGML. SMDL showed some promise but was found to be far too complicated to use and understand, and was never implemented in any commercial music product to be commonly found in the marketplace.

MusicXML aims to strike a balance between these two extremes to become a powerful and user-friendly music format. Support for MusicXML in music products will also be critical to its success. The design of MusicXML would best be accomplished by basing the foundations for the language on existing, successful formats, and also by developing software extensions in parallel with the language. This would ensure that the new format would be based on solid foundations of technical knowledge and experience, and also that the new format would perform as expected when required to provide interchange. By developing the software alongside the language, further developments, and maintenance, of the format could be achieved with a lesser degree of upheaval, than in previous, failed, formats.

To this extent the design of MusicXML followed two main strategies:

- ◆ MusicXML design was based on the fundamentals of two existing computer notation formats; MuseData and Humdrum. MuseData is intended to be generic, software independent, and used for multiple purposes (Hewlett, 1997). Humdrum was designed primarily for use in research, analysis, and music studies; it allows an unlimited amount of representation syntax to be defined (Huron, 1997).
- ◆ MusicXML was grown and developed alongside associated software applications and plug-ins. The notation capabilities of MusicXML were tested, especially in terms of interchange, by providing the initial ability of being able to convert MusicXML to and from MuseData, output to Standard MIDI File type 1, and convert from NIFF to MusicXML. See Fig.1 for diagram.

Figure 1. Initial conversion capabilities of MusicXML, based on design methods.



MusicXML is based on XML conventions, the most important being the use of tags to represent information, in this case musical information. XML allows the creation of a common file format using tags, which can be defined by the developer for their own specific purposes. By employing XML in the creation of a new

format, tags can be defined that have meaningful names. XML provides the developer with the ability to define what the syntax is to be, whilst the XML semantics are being obeyed. MusicXML is no exception, and users with little or even no knowledge of musical notation can read and, though not always, understand the content of the MusicXML file. Tags have names such as “note”, “clef”, “key”, and “measure”, providing indications of the musical data that is being represented. Combined with the content and data associated with these named tags, the meaning, and ultimately the music representation, associated with these tags can be derived. MusicXML documents use tags to provide document structure, which are complemented by the use of tag elements, attributes, attribute values, and content. These are the key components of any XML document and allow for depth and control of data, to a degree of detail only limited by the design constraints of the language creator.

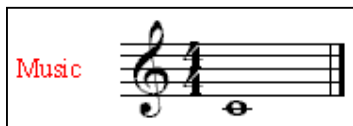
3.2 Simple MusicXML Example

The following code is a small amount of MusicXML which can be used to generate the simplest notation using MusicXML:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE score-partwise PUBLIC "-//Recordare//DTD MusicXML 0.6 Partwise//EN"
"http://www.musicxml.org/dtds/partwise.dtd">
<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>
```

The following image in Fig.2 shows the graphical representation of this piece of MusicXML code, when it is viewed using a computer software notation package, in this case Coda Music’s Finale.

Figure 2. Graphical representation of simple MusicXML code.

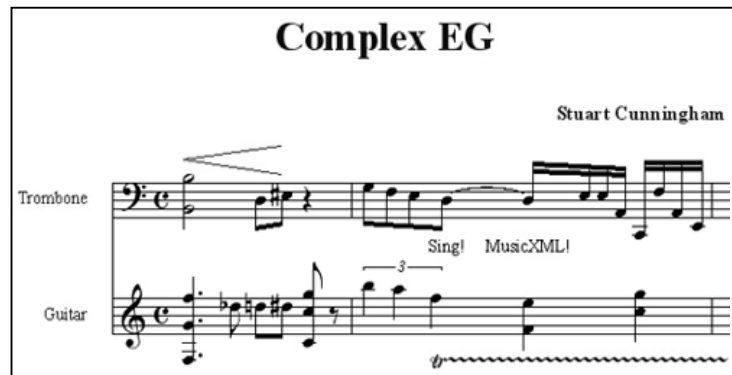


3.3 Complex MusicXML Example

The following piece of code demonstrates the representation of multiple aspects of music notation. The full code was not included because of its length, so attention has been drawn to sections of significance. The original source code for this example can be found in (Cunningham, 2003_b)

The example is as follows:

Figure 3. Graphical representation of more complex MusicXML code.



The image shows a musical score titled "Complex EG" by Stuart Cunningham. It features two staves: Trombone (bass clef) and Guitar (treble clef). The Trombone part starts with a crescendo hairpin above the first bar. The Guitar part has a triplet of eighth notes in the second bar, with the lyrics "Sing! MusicXML!" written below it. The score includes various musical notations such as notes, rests, and dynamic markings.

This particular example draws attention to the use of MusicXML for representing data such as text for lyrics, dynamics, sharp / flat / accidental symbols, and ornaments, as well as basic musical notes. The following code snippets indicate some of these points of interest.

Authoring software, composition title, and composer information:

```
<score-partwise>
  <movement-title>Complex EG</movement-title>
  <identification>
    <creator type="composer">Stuart Cunningham</creator>
    <rights>2003 Stuart C / NEWI</rights>
    <encoding>
      <software>Finale for Windows</software>
    </encoding>
  </identification>
```

Crescendo above the first bar in the trombone part:

```
<direction placement="above">
  <direction-type>
    <wedge type="crescendo" spread="0" relative-y="11" />
  </direction-type>
</direction>
```

Lyrics "Sing!" being attached to the appropriate note, below bar two of the trombone part:

```
<lyric number="1">
  <syllabic>single</syllabic>
  <text>Sing!</text>
</lyric>
</note>
```

The second note, first bar, in the guitar part which is flat:

```
<note>
  <pitch>
    <step>D</step>
    <alter>-1</alter>
    <octave>5</octave>
  </pitch>
  <duration>6</duration>
  <voice>1</voice>
  <type>eighth</type>
  <accidental>flat</accidental>
  <stem>down</stem>
</note>
```

The starting point of the trill ornament:

```
<notations>
  <ornaments>
    <trill-mark />
    <wavy-line type="start" />
  </ornaments>
</notations>
```

One other item of note in this representation is the inclusion of MIDI performance data in the MusicXML document. This was automatically generated by Finale when the file was encoded, and is not indicating by looking at the image in Figure 3. For example, the following is associated with the guitar part:

```
<midi-instrument id="P2-I19">
  <midi-channel>2</midi-channel>
  <midi-program>26</midi-program>
</midi-instrument>
```

3.4 Future of MusicXML

MusicXML was created specifically to produce a musical notation file format that was designed for the purposes of interchange. MusicXML aims to be able to represent, more than amicably, the majority of notational elements that are commonly encountered when writing music. The primary goal of MusicXML is to provide users with a format that is Internet friendly (easy to transport and use across Internet transfer mediums) and is therefore suitable for being used as a format for interchange of data.

MusicXML continues to grow and be updated by Recordare, and support for the file format in commercial music notation programs has grown significantly over recent years, with many software developers currently in the process of implementing MusicXML support (Recordare, 2003). MusicXML has already been adopted in a range of music applications, aside from pure notation packages, such as music scanning and recognition programs, guitar and tablature based applications, and a recent Java based software development entitled 'Project XEMO' (Cunningham, 2003_a). Support in software applications from dominant forces in the music notation marketplace, such as Sibelius and Finale, has been a crucial landmark for MusicXML, and demonstrates that major industry developers are interested in the format, and believe that it is a worthwhile cause. It is only by gaining the respect and implementation in these products that MusicXML will stand a chance of being adopted as a standard format for music notation. Adoption in a range of software notation products may see MusicXML being accepted as an ad-hoc standard at the very least, which would place the format ahead of NIFF, and already a promising replacement for the Standard MIDI File format. Recordare is a member of OASIS (Organization for the Advancement of Structured Information Standards), a non-profit organisation devoted to assisting the development and adoption of worldwide standards, and coordinating with organisations such as ISO, ITU, and W3C.

MusicXML has very recently reached version 0.9, with version 1 being released within the next 4 months. The path for the development has been mostly determined by Recordare, however, beyond version 1 the development of the language should have reached a stable and adequately complete level to be used for the

majority of music notation tasks. Beyond version 1 the development of further features will centre on the needs and demands of MusicXML users. Some aspects of MusicXML which may benefit from further development would be the representation of non-musical data which could be used to assist music analysis or research tasks. The ability to respond to the demand of users by Recordare combined with the flexibility provided by using XML means that MusicXML development could continue years into the future, and reaches a level where it would be an extremely concise method of representing many different styles of musical notation. This means that the future for MusicXML looks promising, and this impinges on users of the format, meaning that a standard, if only ad-hoc at best, is available for use which is supported in a variety of applications, large and small.

Although there are software applications that already provide extensive MusicXML support, one feature which would be profitable would be the creation of a standalone viewer for MusicXML files. This would allow musical notation to be quickly accessed and displayed on screen and also allow the printing of the notated musical information. Such an application could also be extended to allow very simple editing of the musical data. With the growth of the World Wide Web, such features could be accommodated by a web browser plug-in, that could not only view MusicXML documents on the local file system, but also allow MusicXML files to be viewed remotely, rather than simply displaying the XML source code for the file, which is the current result of viewing a MusicXML document in a web browser. Given that browsers are based on HTML technology, which is based on SGML; as is XML, implementation of such a browser plug-in if only to view MusicXML files should not be a particularly laborious task.

Other possible areas that would benefit from MusicXML implementation are outlined below:

- ◆ Context based searching of notated musical information via the World Wide Web or the Internet.
- ◆ Mobile computing devices, used as a medium for editing, and presentation of musical notation data.
- ◆ Music archiving. MusicXML files have relatively small file sizes which can be easily compressed.

4. CONCLUSION

The success of MusicXML will hinge significantly on the support and adoption of software developers including functionality in their products. It is important to assess reasons given by developers who do not currently provide MusicXML support:

- ◆ Unfamiliarity with the MusicXML format
- ◆ No demand from clients
- ◆ Might implement MusicXML if others did first

These three reasons alone can easily form the basis of a vicious circle in the uptake of MusicXML as a recognised and widely supported format. One of the major contributing factors to the downfall of NIFF as a standard format for musical notation was the lack of support and implementation of the product by software manufacturers, due mainly to lack of awareness amongst developers and no demand from consumers, essentially leaving NIFF redundant and non-existent in the marketplace (Cunningham, 2003_a) (XEMO, 2003).

It would not be unreasonable to suggest that if software developers and consumers of music notation products were to be made more aware of MusicXML and the features it can provide, this would increase the use of MusicXML. To make the people that matter more conscious of MusicXML would not only remove an unfamiliarity of MusicXML among developers, but also increase the demand of the consumer for MusicXML compatible software. After all, in essence this would simply be a tried and tested sales technique: basic marketing. MusicXML is a format that can be freely used by individuals and companies in the notation software industry, providing that the free licence agreement is observed.

Using XML as the language for defining musical information brings with it a number of very useful advantages and features. One of the main advantages of using XML is its ability to be used and understood on a wide variety of computing platforms. XML is a universally accessible format, and is not restricted to being used on one particular platform. This is mainly due to it being a text / ASCII based format. This also brings benefits of XML being easily edited and can also aid in the storage and transmission of XML documents. Text based files means that storage requirements can be kept to a minimum, and common lossless compression techniques, such as run length encoding, can be easily applied to text based documents, and a chosen compression method is independent of the file format itself, which operates in clear text. By basing MusicXML on XML, the ability to extend and update a flexible format is allowed (Castan, 2001), and in its most simplistic form is merely a case of updating the DTD file(s) and ensuring that MusicXML users reference an up to date DTD in their documents.

MusicXML has the capability to become a dominant, widespread, format in the coming years.

ACKNOWLEDGEMENT

Thanks to: Dr. Vic Grout (NEWI), Dr. Rich Picking (NEWI), and Dr. Derek Turner (University of Paisley).

REFERENCES

Castan, G., ed. Selfridge-Field, E., Hewlett, W.B., 2001, *NIFFML: An XML implementation of NIFF, The Virtual Score: Representation, Retrieval, Restoration – Computing in Musicology 12*, MIT Press, Massachusetts, USA.

Castan, G., Good, M., Roland, P., ed. Selfridge-Field, E., Hewlett, W.B., 2001, *Extensible Markup Language (XML) for Music Applications: An Introduction, The Virtual Score: Representation, Retrieval, Restoration – Computing in Musicology 12*, MIT Press, Massachusetts, USA.

Cunningham, S., 2003, *Music File Formats and Project XEMO*, MSc Dissertation, University of Paisley, Scotland, UK.
Available from: http://www.newi.ac.uk/cunninghams/research/SCunningham_MSc.pdf

Cunningham, S., 2003, *MusicXML Complex Example*, University of Wales / NEWI, UK.
Available from: <http://www.newi.ac.uk/cunninghams/research/complex.xml>

Good, M., 2001, MusicXML: An Internet-Friendly Format for Sheet Music. *IdeAlliance: XML 2001*. Florida, USA.
Available from: <http://www.idealliance.org/papers/xml2001/papers/pdf/03-04-05.pdf>

Hewlett, W. B., ed. Selfridge-Field, E., Hewlett, W.B., 1997, *MuseData: Multipurpose Representation, Beyond MIDI: The Handbook of Musical Codes*, MIT Press, Massachusetts, USA.

Huron, D., ed. Selfridge-Field, E., Hewlett, W.B., 1997, *Humdrum and Kern, Beyond MIDI: The Handbook of Musical Codes*, MIT Press, Massachusetts, USA.

Recordare, 2003, *MusicXML Definition*, Recordare LLC, California, USA.
Available from: <http://www.musicxml.com/about.html>

St Laurent, S., 2000, *XML Elements of Style*, McGraw – Hill Publishers, Berkshire, UK.

XEMO, 2003, *Project XEMO: Musical Codes*, Xemus Software LLC, California, USA.
Available from: <http://www.xemo.org/fileFormats.html>