The MusicXML Meeting

MICHAEL GOOD
VICE PRESIDENT OF RESEARCH AND DEVELOPMENT

APRIL 17, 2015
Agenda

• Introduction to MusicXML

• MusicXML community progress in the past year

• MusicXML and the Standard Music Font Layout (SMuFL)
  – With Daniel Spreadbury, Steinberg

• Future directions for MusicXML: content and governance
  – With Joe Berkovitz, Noteflight

• Focused discussion on future directions at the end

• Reception at 4:00 pm sponsored by Hal Leonard / Noteflight
What is MusicXML?

- The standard open format for exchanging digital sheet music between applications
- Invented by Michael Good at Recordare in 2000
- Developed collaboratively by a community of hundreds of musicians and software developers over the past 15 years
- Available under an open, royalty-free license that is friendly for both open-source and proprietary software
- Supported by almost 200 applications worldwide
MusicXML Is a Notation Format

- Music is represented using the semantic concepts behind common Western music notation
- Includes both how a score looks and how it plays back
- Includes low-level details of the appearance of a particular engraving, or the nuances of a particular performance
  - Allows transfer of music between applications with high visual fidelity
  - Also allows the visual details to be ignored when appropriate
  - The best display for paper is often not the best for an interactive application
MusicXML as an Archival Format

• MusicXML is an XML format, with all its advantages:
  – Files can be opened in any computer text editor
  – Fully internationalized via Unicode
  – Files are human-readable as well as machine-readable
  – Can use all the standard XML tools developed by larger industries than the music industry

• Backward compatibility: all valid MusicXML 1.0 files are also valid MusicXML 3.0 files

• MusicXML license allows continued development of the format by anyone, not just MakeMusic

• Already implemented by nearly 200 programs
# Who Uses MusicXML

Usage map as of April 2015

<table>
<thead>
<tr>
<th>Products Shipping Now</th>
<th>Beta/Prototype Software</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMML</td>
<td>NoteAbility Pro</td>
</tr>
<tr>
<td>Braille Music Editor</td>
<td>Noteflight</td>
</tr>
<tr>
<td>Calligr Suite</td>
<td>Noteworthy Composer</td>
</tr>
<tr>
<td>capella</td>
<td>Notion</td>
</tr>
<tr>
<td>Cubase</td>
<td>Obiv Octava</td>
</tr>
<tr>
<td>Electric Pipes</td>
<td>OpenMusic</td>
</tr>
<tr>
<td>Encore</td>
<td>Open Score Format</td>
</tr>
<tr>
<td>Ensemble Composer</td>
<td>Pizzicato</td>
</tr>
<tr>
<td>Finale (inc. NotePad,</td>
<td>Primus</td>
</tr>
<tr>
<td>PrintMusic, SongWriter)</td>
<td>Progression</td>
</tr>
<tr>
<td>forte</td>
<td>ProxyMusic</td>
</tr>
<tr>
<td>FreeClef</td>
<td>QuickScore Elite Level</td>
</tr>
<tr>
<td>Guitar Pro</td>
<td>SCORE</td>
</tr>
<tr>
<td>Harmony Assistant</td>
<td>Score Perfect Professional</td>
</tr>
<tr>
<td>iComposer</td>
<td>Scorio</td>
</tr>
<tr>
<td>JFugue</td>
<td>Sibelius / Sibelius First</td>
</tr>
<tr>
<td>ibMusicXML</td>
<td>SmartScore</td>
</tr>
<tr>
<td>Lime</td>
<td>Songs2See Editor</td>
</tr>
<tr>
<td>MagicScore</td>
<td>Speech Analyzer</td>
</tr>
<tr>
<td>Mozart</td>
<td>StaffPad</td>
</tr>
<tr>
<td>MuseEdit</td>
<td>Symphony Pro</td>
</tr>
<tr>
<td>MuseScore</td>
<td>TaBazar</td>
</tr>
<tr>
<td>MusicNotation SDK</td>
<td>TabEdit</td>
</tr>
<tr>
<td>music21</td>
<td>VocalEasel</td>
</tr>
<tr>
<td>NotateMe</td>
<td>Notification</td>
</tr>
<tr>
<td>AudioScore Ultimate</td>
<td>Notation Switchblade</td>
</tr>
<tr>
<td>Audivis</td>
<td>Opusmodus</td>
</tr>
<tr>
<td>Cadencii</td>
<td>PDFtoMusic Pro</td>
</tr>
<tr>
<td>CameraScore</td>
<td>PhotoScore Ultimate</td>
</tr>
<tr>
<td>capella-scan</td>
<td>Plaine and Easy</td>
</tr>
<tr>
<td>capella wave kit</td>
<td>Rosegarden</td>
</tr>
<tr>
<td>Cavatina</td>
<td>Rousseau</td>
</tr>
<tr>
<td>Crescendo</td>
<td>Samplitude / Sequoia</td>
</tr>
<tr>
<td>Django</td>
<td>ScoreCloud</td>
</tr>
<tr>
<td>HarmonyWiz</td>
<td>Score Creator</td>
</tr>
<tr>
<td>Impro-Visor</td>
<td>ScoreMaker</td>
</tr>
<tr>
<td>iReal Pro</td>
<td>SharpEye</td>
</tr>
<tr>
<td>JMSL</td>
<td>SmartScore NoteReader</td>
</tr>
<tr>
<td>Logic Pro</td>
<td>SONAR</td>
</tr>
<tr>
<td>Ludwig</td>
<td>Symphonix Evolution</td>
</tr>
<tr>
<td>Melomics</td>
<td>Synfire</td>
</tr>
<tr>
<td>muscript</td>
<td>tonica fugata</td>
</tr>
<tr>
<td>MyScript Music SDK</td>
<td>TuxGuitar</td>
</tr>
<tr>
<td>Archivarius 3000</td>
<td>MusicEase</td>
</tr>
<tr>
<td>Auto-Tune EFX 2</td>
<td>Music Prodigy</td>
</tr>
<tr>
<td>Avid Scorch</td>
<td>Myriad Music Plug-in</td>
</tr>
<tr>
<td>Blackbinder</td>
<td>Myriad QuickLook Plug-in</td>
</tr>
<tr>
<td>capella playAlong</td>
<td>neoScores</td>
</tr>
<tr>
<td>capella start</td>
<td>NtEd</td>
</tr>
<tr>
<td>CsoundAC</td>
<td>OrganMuse</td>
</tr>
<tr>
<td>Denemo</td>
<td>Overture</td>
</tr>
<tr>
<td>Don's MusicXML Viewer</td>
<td>OveScore</td>
</tr>
<tr>
<td>EasyABC</td>
<td>Personal Composer</td>
</tr>
<tr>
<td>Extempore</td>
<td>Phonicscore</td>
</tr>
<tr>
<td>Frescobaldi</td>
<td>Pianoludic</td>
</tr>
<tr>
<td>GUIDO</td>
<td>Pocket Jainz</td>
</tr>
<tr>
<td>Igor Engraver</td>
<td>Practice Player Live Midi</td>
</tr>
<tr>
<td>JNScore</td>
<td>Purely Musical</td>
</tr>
<tr>
<td>Jellynote</td>
<td>Score Writer</td>
</tr>
<tr>
<td>Klavarscript</td>
<td>SeeScore / SeeScore SDK</td>
</tr>
<tr>
<td>Kooplet</td>
<td>Singer's Mate</td>
</tr>
<tr>
<td>Legato</td>
<td>Songs2See Game</td>
</tr>
<tr>
<td>LilyPond</td>
<td>Soundslice</td>
</tr>
<tr>
<td>MelodicMatch</td>
<td>TEpAd / TEFView</td>
</tr>
<tr>
<td>Melody Assistant / Player</td>
<td>teoria</td>
</tr>
<tr>
<td>Mobile Music Trainer</td>
<td>TNOTh</td>
</tr>
<tr>
<td></td>
<td>xml2abc</td>
</tr>
</tbody>
</table>

**MUSICXML**

- Audimus Notes
- Bach
- Flat
- Haskell Library
- Humdum
- KGuitar
- MySQL
- Nightingale Notelist
- Ossia Viewer
- Pomeleic
- pyScore
- Scored
- Zongl Editor

**Usage Maps**

- abc2xml
- abcd4j
- Amadeus
- Braille Music Compiler
- BUZZle
- Digital Performer
- FOMUS
- jChing
- MaxScore
- mccursion
- Middle C
- mingus
- Overture
- Power Tab
- PWGL
- Song Builder

**Tools**

- Antescofo
- Arduino
- BrailleMUSE
- Canorus
- CrestMuse Toolkit
- FreeDots
- GLozart
- GVstreamer
- Harmonia
- HTML5 Guitar Tab Player
- MATLAB
- MuseBook Score
- Musicista
- musicxml2midi
- musicxml2words
- MusicXML to MP3
- MXML2Midi
- Opus
- Philomelos
- PSAM Control Library
- Quantified Artist
- Sinsy
- Zongl Player
Publishing Scores in MusicXML

- MusicXML is the way that scores get from composition/publishing applications like Finale and Sibelius to the new wave of musician applications.

- No DRM controls built-in, though these have been added in the MusicXML-based Open Score Format.

- For copyrighted music, MusicXML has usually been a Business-to-Business format, not Business-to-Consumer.

- Many sites available with public domain MusicXML scores: see www.musicxml.com/music-in-musicxml
What Is New With MusicXML?

• New and improved application support
  – 25 new applications since last Musikmesse
  – 4 more applications out of beta

• New MusicXML Forum replacing the MusicXML mailing list
  – Atom feed, forum and topic subscriptions in place

• New possibilities for MusicXML 4.0
  – SMuFL
  – Evolution for more use cases
  – Change of governance
New MusicXML Reader/Writers

- BMML
- NotateMe
- StaffPad
- Mozart (formerly read-only)
- bach (beta)
- Flat (beta)
- Scored (beta)
New MusicXML Writers

- CamraScore
- Cavatina
- HarmonyWiz
- Melomics
- MyScript Music SDK
- Opusmodus
- Score Creator
- SmartScore NoteReader
- Braille Music Compiler (beta)
- Digital Performer 9 (beta)
- MaxScore (beta)
New MusicXML Readers

- Don’s MusicXML Viewer
- Music Prodigy
- PhonicScore
- Practice Player Live Midi
- Purely Musical
- Soundslice
- Antescofo (beta)
- Musicista (beta)
MusicXML Out of Beta

- Denemo
- INScore
- Jellynote
- neoScores
MusicXML and Music Fonts

• MusicXML provides a standard interchange format for music notation semantics, layout, and performance

• But when translating between MusicXML and a music notation application, fonts complicate things

• What code point do I use for a particular MusicXML element in a particular music notation font?

• What do MusicXML’s positioning attributes mean specifically with regards to any particular music font?

• Enter the Standard Music Font Layout (SMuFL)
Standard Music Font Layout

MusicXML community meeting
19 April 2015
Daniel Spreadbury
What is SMuFL?

- A standard way of mapping musical symbols to the Private Use Area of the Basic Multilingual Plane in Unicode
- A set of technical guidelines for how music fonts should be designed and built
- Simple JSON metadata formats to help applications use SMuFL fonts easily
- Released under MIT license, free to use/modify
What’s included

• 2407 glyphs in 108 ranges
• Includes all 220 glyphs from the Unicode Musical Symbols range
• Also includes recommendations for ligatures, stylistic alternates/sets, etc.
• Reference fonts for scoring and text-based application
Bravura
Bravura

- Reference SMuFL font (OpenType/SVG/WOFF)
- Includes all SMuFL recommended glyphs, and hundreds of optional glyphs
- Released under the SIL Open Font License
  - Free to use, bundle, embed, create derivative versions, etc.
  - Only licensing restrictions are that the font cannot be sold on its own; derivative versions cannot use the same name; and derivative versions must be released under the same licensing terms
Current status

• Version 1.12, released January 2015
• Short backlog of pending suggestions and issues
• Considered stable at this time
Implementations

• Bravura now supported by or shipping with:
  – MuseScore 2.0 (open-source scoring software)
  – Logic Pro X 10.1 (Apple’s DAW)
  – Soundslice (web-based interactive sheet music)
  – Verovio (web-based MEI viewer)
  – Groove Freedom (iOS drum tuition app)

• All above SMuFL-compliant to varying degrees
Implementations

• Third-party SMuFL-compliant fonts include:
  – November 2.0 (commercial license)
  – Gootville (based on Gonville, MuseScore 2.0)
  – Leipzig (Laurent Pugin, ships with Verovio)
  – Maestro (MakeMusic Inc., for future Finale)
  – Others in development by independent font developers
Future directions

• Integrate with MusicXML 4.0
• Continue advocacy of standard to font designers and application developers
• Further development to be guided by requirements of community, and Steinberg’s own needs
More information

www.smufl.org
Thank you!

d.spreadbury@steinberg.de
MusicXML 4.0 and SMuFL 1.12

• SMuFL addresses many standardization issues that have troubled MusicXML developers for years

• What could better MusicXML support for SMuFL mean?
  – Does MusicXML add support for all of SMuFL’s thousands of glyphs?
  – If not, what guidelines to determine which ones?
  – Could we add escape methods to access SMuFL glyphs by their canonical name?
  – How about standardization on areas such as font metadata?
  – Should MusicXML documentation reference SMuFL canonical names to clarify the graphical appearance of different MusicXML elements?
SMuFL Support in MusicXML 3.0

• A count of “glyphs supported” is tricky because there is not always a 1-1 mapping between MusicXML concepts and SMuFL glyphs

• Glyphs intended for music text font use in particular may not line up exactly with MusicXML concepts

• Some SMuFL sets of glyphs make semantic or graphical distinctions not captured in MusicXML 3.0, though the common set of base semantics are supported

• Nevertheless, here are some preliminary counts…
SMuFL Glyphs in MusicXML 3.0

- MusicXML 3.0 fully supports 839 out of the 2407 glyphs in SMuFL 1.12, or 35%
- MusicXML 3.0 partially supports another 254 glyphs, for a total of 1093 glyphs or 46%
- Large areas of non-support:
  - Extended accidentals: 11 ranges not supported at all, covering 348 glyphs or 14% of SMuFL
  - Pre-CMN notation: 11 ranges not supported at all, covering 217 glyphs or 9% of SMuFL
- Missing less common symbols in other ranges, or most symbols in some ranges like multi-segment lines
MusicXML 4.0

• SMuFL provides one motivation for a major new MusicXML release

• What else besides SMuFL support?
  – Features and fixes as discussed at last year’s meetings and on the MusicXML forum
  – Improved online documentation

• But most interesting is the concept of evolving MusicXML for better support of more use cases
  – Joe Berkovitz will be discussing this in more detail
MusicXML as a Document Format

• MusicXML has very much focused on a printed musical score as a reference for its data model

• Remember the context in 2000: there had been repeated failure to build a useful music notation interchange format

• So make it easier to standardize among competing programs by primarily modeling physical, real-world object

• Make modeling compatible with leading commercial and academic applications to ensure ease-of-use for developers

• MuseData was primary starting point, plus Humdrum
Times Have Changed

- All but 2 of the major applications related to notation now support MusicXML.
- Document interchange gets better as software matures and – even more important – publisher processes change to emphasize digital-ready scores.
- Starting with the iPad, digital sheet music has gotten much more popular.
- MusicXML can improve its support for interactive applications that reflow and go beyond a substitute for paper.
- The need for a better specification has grown with success.
Change Can Be Hard

- MIDI and HTML are two standards whose success led to limited change and lack of innovation over time.
- Transitions from MIDI 1.0 to HD Protocol and HTML4 to HTML5 have been difficult.
- MusicXML 3.0 works very well as an exchange and archival format for common Western music notation, and we need to keep that compatibility.
- But if MusicXML does not evolve, the odds increase for fragmentation and losing the interchange that all here have worked so hard to achieve.
MusicXML as an Interactive Platform

• One great advantage of MusicXML is that it serves as a model for what one needs to cover in representing music notation on paper and on screen.

• Can we expand that to being a model for what one needs to cover to interact with music notation on screen?
  – While maintaining capability with MusicXML 3.0
  – And maintaining agreement between different vendors who implement things in different ways
  – The paper score is no longer the external authority outside of software implementations.
Could Change of Governance Help?

- Standards organization resources could help create a tighter specification with better validation
- A standards organization could provide greater stability than single-company ownership
- More opportunity for integration with web standards
- Need a lightweight home that still keeps widespread participation from individuals and small companies
- New standards venues might avoid past pitfalls of standards organization efforts
- Explored interest at W3C, MMA, and IEC this past year
Staying in Touch

- Shows: Musikmesse, NAMM, SF MusicTech
- Twitter: @MusicXML
- Facebook: [www.facebook.com/MusicXML](http://www.facebook.com/MusicXML)
- Email: mgood@makemusic.com
MusicXML: Framing the Future
MusicXML Community Meeting
Musik Messe 2015

Joe Berkovitz (joe@noteflight.com)
President, Noteflight LLC
Co-chair, W3C Web Audio Working Group
W3C Advisory Committee Rep., Hal Leonard Corporation
Who am I?

• I compose and play music
• I build notation software
• My company is owned by a music publisher
• I work on Web standards
Where can MusicXML go from here?

- The Case for Evolution
- How to Evolve?
- Choices in Governance
Facets of Evolution

- History
- Process
- Specification
- Features
- Governance
History

• De facto use cases: exchange and archive
• De facto steering: by notation software vendors
• Non-PDF digital music publishing initially a sideline, now growing rapidly
• Needed to create conditions for success of a viable standard
Present-day Publishing with MusicXML (wishful version)

• Get hold of some MusicXML files from any source
• Feed them into some compatible application
• Everything looks great
• Drink a beer, glass of wine, shot of amaro (perhaps several) in celebration
Reality Check!

• Get hold of some MusicXML files from any source
• Discover that they use different subsets of MusicXML in different ways
• Discover that engravers used different features to mean the same thing, or the same features to mean different things
• Discover that your renderer requires certain features to be present that are not in your files, or can’t use the features that are there
• Discover that you have no way to specify how your scores should look in diverse end-user environments (paper, desktop, mobile, …)
• Drink something stronger (perhaps several) in despair
For Developers, It’s No Easier

- See previous page :-)
Looking Forward

• Many historical goals have been met
• Other goals remain to be clarified, addressed
• Successful digital publishing with MusicXML is possible, yet challenging
• MusicXML remains the best way forward
• What is the best way to chart and pursue its future path?
Evolving the Process

• Identify major stakeholder roles
• Form group of active stakeholders
• Develop use case document
• Identify underserved use cases
• Identify key features unlocking these cases
Some Use Cases and Needs

- Notation editor import/export
- Music Publishing
- Reading/performance systems, both desktop & mobile
- Non-editor notation apps (e.g. theory, ear training)
- Scholarly and specialist publications
- In-house publishing
- Libraries and archival services
- Convergence with Web and Epub technologies
Developing a Specification

- MusicXML needs one - XSD distillation is not equivalent or sufficient
- Large number of optional features creates confusion, makes results unpredictable
- Spec must identify distinct feature profiles addressing common use cases.
- Spec must make testable statements about conformance.
Likely areas for evolution

- Flexible Styling and Layout
- Syntactic validation of semantics
- Metadata vocabularies
- Manipulation, interactivity and selection
- Playback
- Graphics and hypertext inclusion
- Anchors and Pointers
- Annotations
- Accessibility
One likely area: Cascading Style Sheets (CSS)

- Stylesheets allow definition of “how it looks” to be cleanly separated from “what it is”.
- Many documents can share the same set of stylesheets.
- To customize the look of a document, change the stylesheet you are using.
- Style “attributes” can reflect high-level concerns of engravers (e.g. density or placement conventions) not low-level details (X/Y positions of many individual objects)
- Stylesheet queries support responsive design
Cascading Style Sheets (CSS)

**print.css: (print-oriented stylesheet)**

credit.title { /* manner in which title-type credit should be shown */
    position: absolute;
    top: 120px;
    horizontal-align: center;
}

part { /* inherited attributes applying to all <part> children */
    staff-line-spacing: 6px; /* 6 pixels between staff lines */
}

part#P1 { /* override for violin part shown at smaller size */
    staff-line-spacing: 4px;
}

measure {
    duration-spacings: 5L 3L 2L 1L 0.71L; /* standard spacings for durations */
}

direction.tempo { /* How should a tempo direction look? */
    font-weight: bold;
    font-size: 15px;
    default-offset: +2L; /* 2 lines above staff */
}

note.alternateReading { /* special style class for alternate readings */
    note-size: 0.5; /* relative size of note heads */
}
Example: Styling and CSS

Excerpt of score.xml: (note independence from CSS stylesheet)

```xml
...<credit class="title">Sinfonia XLVII</credit>
...<part id="P1">
  <measure number="1">
    <direction class="tempo">Grandly</direction>
    <note>
      ...
    </note>
    <note class="alternateReading">
      ...
    </note>
  </measure>
...</part>...```
Example: Styling and CSS

**mobile.css:**

```css
credit.title {
    display: none; /* in mobile app, title is not part of score rendering */
}

part { /* inherited attributes applying to all <part> children */
    staff-line-spacing: 8px;
}

note.alternateReading { /* special style class for alternate readings */
    color: rgb(127,127,127); /* on mobile, gray out rather than make smaller */
}
```
Interactivity plus styling with CSS, DOM, jQuery

// Highlight the most recently clicked note as green and play it
var highlightedNote = null; // track last-clicked note

// Highlight a given note
function highlightNote(note) {
    if (highlightedNote) {
        highlightedNote.css("color", ""); // remove previous highlight
    }
    highlightedNote = note;
    highlightedNote.css("color", "rgb(0,255,0)"); // apply new highlight
}

// Process click events dispatched from note elements in MusicXML DOM
document.addEventListener("click", function(event) {
    var target = $(event.target);
    if (target.is("note")) {
        highlightNote(target);
    }
})
The Best Way to Evolve

- Consortium-based governance is the best way forward
- Standards-track process will force clear specification
- Consortium ownership assures openness and fairness
- Membership supplies diverse, fresh viewpoints
- Leadership supplies continuity, domain expertise
- Consortium supplies adjacent expertise, technical/legal/process support
Compatibility

• Goal: preserve as much as possible
• Easier to migrate MusicXML than begin over again
• Always have a well-defined mapping in both directions across any syntactic change
• Public domain tools for transforming old <-> new
Some Adjacent Standards

- CSS
- SVG
- SMuFL (should ideally be open, too!)
- HTML
- MIDI
- EPUB
- others?
Consortium Choices

- W3C owns many adjacent specifications and provides access to their experts
- W3C has excellent technical support for developing specs and seeing them through
- W3C has proven its ability to adapt
- MMA is custodian of an important but singular and domain-specific spec
- EPUB still primarily targets text-oriented publications, moving towards Web, Arts, STM
Proposal

• Form W3C Community Group (CG) with Michael Good in a leadership role. CGs are the initial step on a track to W3C standard. No membership fees are required.

• CG immediately publishes current rev. of MusicXML

• Begin to identify use cases, needs, features

• Begin to codify complete, verifiable specification.

• Recruit best musical experts and experts in adjacent technologies (e.g. CSS, EPUB, MIDI)

• Eventual W3C Working Group and Recommendation
Questions for Attendees

• Do you feel this proposal is worth further consideration?

• Are there ways you would like to see MusicXML evolve?

• Would you like to be a stakeholder directly involved in this evolution?

• Or… do you want to be represented by a stakeholder whose interests align with yours?

• Do you feel consortium ownership would bring benefits for MusicXML?

• Do you feel that W3C could deliver these benefits? Would another consortium be better?