MusAssist: A Domain Specific Language for Music Notation

Ilana Shapiro
Pomona College
issa2018@mymail.pomona.edu

ABSTRACT

MusAssist is an external, declarative domain specific language for music notation. Users can change key signatures, start a new measure, and describe musical structures such as notes, rests, and custom chords in MusAssist's straightforward syntax much in the same way they would when composing. MusAssist is unique in that users can also describe complex musical templates for triads and seventh chords, cadences, and the four primary harmonic sequences with desired length. The level of abstraction of a MusAssist template MusAssist matches that of the theoretical musical structure it describes (e.g. users can describe a harmonic sequence without lowering the abstraction level to chords and notes). This allows users to write out specifications precisely at the conceptual levels of the musical structures they would organically conceive when composing by hand. The musical expressions described by the specifications are expanded out (i.e. the level of abstraction is fully lowered) by the Haskell-based MusAssist compiler and are translated to MusicXML, a language accepted by most major notation software, allowing for further manual editing.

1. INTRODUCTION

When writing music, composers must manually transition from musical theoretical concepts to notes on a page. This process can be tedious and slow, requiring the composer to expand complex structures, such as cadences and sequences, by hand to the notes that they constitute. The level of abstraction of the musical theoretical structure is higher than what the composer actually writes.

Domain specific languages, or DSLs, are programming languages highly specialized for a specific application and thus characterized by limited expressiveness. An $external$ DSL has custom syntax that is separated from the primary language of its application. MusAssist is an external, declarative domain specific language (DSL) for music notation that attempts bridge the divide between music theory and notation. Users describe a composition in MusAssist's straightforward syntax, and the MusAssist compiler writes out the music via these instructions. MusAssist's declarative programming paradigm was chosen to correspond with the declarative nature of handwritten music.

Fundamentally, MusAssist supports notes (including rests) and custom chords (i.e. any desired collection of notes) in the octave and key of choice, as well as change the key signature or start a new measure at any point. MusAssist is unique in that users can also write specifications for complex musical templates *at the same level of abstraction as the musical theoretical structures they describe*. MusAssist supports **chords** (all triads and seventh chords in any inversion), **cadences** (perfect authentic, imperfect authentic, plagal, half, deceptive), and **harmonic sequences** (ascending fifths, descending fifths, ascending 5-6, descending 5-6) of a desired length. The musical expression described by this specification is completely expanded out (i.e. the level of abstraction is fully lowered) by the Haskell-based MusAssist compiler.

The target language of the MusAssist compiler is MusicXML, itself a DSL that is an extension of XML (Extensible Markup Language). MusicXML is accepted by most major notation software programs (such as MuseScore). Thus users can open can open the resulting MusicXML file of a compiled MusAssist composition in MuseScore or another program for further customization and editing, thus bypassing the need to write out complex musical templates by hand at a note- and chord-level of abstraction. Beyond a professional music compositional aid, MusAssist may be particularly helpful to music theory students as an educational tool, enabling them to visualize the relationship between a theortical musical structure and its expanded form, such as a cadence and the chords resulting from its expansion.

2. RELATED WORK

The era of music DSLs began in 2008 with Ge Wang's ChucK audio processing language, which spans the application domains of "methods for sound synthesis, physical modeling of real-time world artifacts and spaces (e.g., musical instruments, environmental sounds), analysis and information retrieval of sound and music, to mapping and crafting of new controllers and interfaces (both software and physical) for music, algorithmic/generative processes for automated or semi-automatic composition and accompaniment, [and] real-time music performance." Since then, researchers have taken advantage of the increased flexibility afforded to DSLs via their limited expressiveness to create music DSLs tailored towards notation, algorithmic composition, signal processing, live coding with music performance, and more. In the notation domain, MusicXML, LilyPond, and PyTabs stand out.

Michael Good's MusicXML is an Internet-friendly, XML-based, declarative DSL capable of representing Western music notation and sheet music since c. 1600. It acts as an "interchange format for applications in music notation, music analysis, music information retrieval, and musical performance," thus supporting sharing between specialized applications.

MusicXML attempts to emulate for online sheet music and music software what the popular MIDI format did for electronic instruments. It is derived from XML in order to help solve the music interchange problem: to create a standardized method to represent complex, structured data in order to support smooth interchange between "musical notation, performance, analysis, and retrieval applications." XML has the desired qualities of "straightforward usability over the Internet, ease of creating documents, and human readability" that translate directly into the musical domain, and it is more powerful and expressive than MIDI.

MusicXML is more expressive than MusAssist, but the level of abstraction of all musical elements is extremely low (i.e. chords must be written out as individual notes) and its syntax is very difficult and tedious to write by hand. However, its flexibility and expressiveness make MusicXML an excellent target compilation language for MusAssist's user-friendly syntax and high-level musical theoretical templates.

LilyPond, an external declarative DSL created by Han-Wen Nienhuys and Jan Nieuwenhuizen, is similar to MusAssist. It features a "modular, extensible and programmable compiler" written in Scheme to generate Western music notation of excellent quality and supports the mixing of text and music elements. Text-based *musical expressions*, or fragments of music with set durations, are compiled to an aesthetically formatted score.

LilyPond and MusAssist are both music notation DSLs tailored to non-programming audiences. However, they differ in two fundamental areas: (1) MusAssist is more expressive than LilyPond as it supports complex music templates at the levels of abstraction of the musical structures they represent and (2) the output of the MusAssist compiler is intentionally editable via notation software (unlike LilyPond's compiler, which produces a static, printable PostScript or PDF file by taking in a file with a formal representation of the desired music).

Simic et al.'s external, declarative DSL PyTabs similarly is geared toward music notation, but in a different domain than MusAssist. Specifically, the authors attempt to solve visual problem of tablature notation, and the lack of standardization of how to specify note duration in this format, by consolidating these issues into a formal language. Tablature notation is outside the scope of MusAssist's focus on Western musical theoretical structures.

3. LANGUAGE FEATURES

3.1 Low-Level Fundamentals

On the most basic level, MusAssist supports individual notes and rests. Rests are given a duration from sixteenth to whole note, and notes are further defined by note name (A to G), accidental (from double flat to double flat), and octave (1 to 8, after the range of a piano). Just as in normal notation, the absence of an accidental indicates natural quality. Finally, users can also define "custom chords," or collections of user-defined notes. These are not considered templates as the high-level description of the chord is not given.

3.2 High-Level Templates

MusAssist's supports templates for chords, cadences, harmonic sequences, specified at the abstraction level of the musical theoretical structures they represent.

Just as in music theory, chords are specified by root note (defined as the fundamental note is), quality (major, minor, augmented, diminished, or half diminished), inversion (root, first, second, or third), and chord type (triad or seventh). Half diminished and third inversion options apply to seventh chords only. The root note cannot have a double accidental, as this can introduce triple accidentals in the chord, which MusAssist does not support.

Cadences are specified by cadence type (perfect or imperfect authentic, half, plagal, or deceptive) and key (defined by a fundamental note and a quality, either major or minor). Currently, MusAssist only supports a single treble clef line. Thus, cadences are written out in the upper voices only, in keyboard voice leading style and incorporating principles of smooth voice leading.

Based on the principles of functional harmony, there are several ways to represent each cadence. In MusAssist, the following representations were chosen. The major version is presented first, and the minor version after that, in parentheses.

Perfect Authentic	Imperfect Authentic	Plagal	Dece
IV-V-I (iv-V-i)	IV-vii$^{o}{}^{6}_{4}$-I$^{6}_{4}$ (iv-vii$^{o}{}^{6}_{4}$-i$^{6}_{4}$)	IV$^{6}_{4}$-I (iv$^{6}_{4}$-I)	IV-V

All cadences except perfect authentic are built exclusively with triads, which also doubles the root in the final chord to simulate the 4-5-1 bass line as well as to preserve the requisite 2-1 downward step in the uppermost voice. The MusAssist syntax `(PerfAuthCadence Eb5 min sixteenth)` produces such a cadence when compiled and loaded into MuseScore notation software in

Figure 1. Perfect Authentic Cadence in E♭ minor

Finally, harmonic sequences are specified by harmonic sequence type (ascending fifths, descending fifths, ascending 5-6, descending 5-6), key (just as in cadences), duration of each chord, and length of the sequence.

Harmonic sequences can be implemented in several ways depending on desired inversion scheme. Since MusAssist does not yet support multi-line composition, harmonic sequences are writte like cadences in keyboard-style voice leading. Though the upper-voice harmonization of a harmonic sequence need not follow the direction in the sequence's name, MusAssist chooses a chord inversion and voice leading pattern such that each sequence does align with the direction of its name (i.e. ascending sequences will ascend directionally).

The chord progressions chosen for each sequence are summarized in (all in major, for the sake of example, but they can certainly be minor as well). Each sequence consists of fourteen chords before it repeats in the subsequent octave.

3.3 Additional Features

Beyond compositional elements, users can set the key signature at the beginning of any measure up to seven sharps

Ascending Fifths	I$^{6}_{4}$	V	ii$^{6}_{4}$	vi	iii$^{6}_{4}$	vii^{o}	IV$^{6}_{4}$	I	V$^{6}_{4}$	ii	vi$^{6}_{4}$	iii	vii$^{6}_{4}$	IV
Descending Fifths	I	IV$^{6}_{4}$	vii^{o}	iii$^{6}_{4}$	vi	ii$^{6}_{4}$	V	I$^{6}_{4}$	IV	vii$^{o6}_{4}$	iii	vi$^{6}_{4}$	ii	V$^{6}_{4}$
Ascending 5-6	I	vi^{6}	ii	vii^{o6}	iii	I^{6}	IV	ii^{6}	V	iii^{6}	vi	IV^{6}	vii^{o}	V^{6}
Descending 5-6	I$^{6}_{4}$	V	vi$^{6}_{4}$	iii	IV$^{6}_{4}$	I	ii$^{6}_{4}$	vi	vii$^{o6}_{4}$	IV	V$^{6}_{4}$	ii	iii$^{6}_{4}$	vii^{o}

or flats by specificing note name, accidental, and quality (sharp or flat). Users can also start a new measure or create a blank measure. Finally, users can label MusAssist expressions and reuse them later in the program. MusAssist comments are designated with \\.

The tempo for all MusAssist programs is set at ♩ = 80bpm and cannot currently be customized or changed.

3.4 Title and Authors

The title is 16 pt Times, bold, upper case, centered. Authors' names are centered. The lead author's name is to be listed first (left-most), and the co-authors' names after. If the addresses for all authors are the same, include the address only once, centered. If the authors have different addresses, put the addresses, evenly spaced, under each authors' name.

3.5 First Page Copyright Notice

Please include the copyright notice exactly as it appears here in the lower left-hand corner of the page. It is set in 8 pt Times, one column in width, and should not descend into the page margins (i.e. it should keep clear of the 1" margin at the bottom of the page).

3.6 Page Numbering, Headers and Footers

Do not include headers, footers or page numbers in your submission. These will be added when the publications are assembled.

4. HEADINGS

First level headings are in Times 12pt bold, centered with 1 line of space above the section head, and 1/2 space below it. For a section header immediately followed by a subsection header, the space should be merged.

4.1 Second Level Headings

Second level headings are in Times 10 pt bold, flush left, with 1 line of space above the section head, and 1/2 space below it. The first letter of each significant word is capitalized.

4.1.1 Third and further Level Headings

Third level headings are in Times 10 pt italic, flush left, with 1/2 line of space above the section head, and 1/2 space below it. The first letter of each significant word is capitalized.

Using more than three levels of headings is strongly discouraged.

5. EQUATIONS, FIGURES, FOOTNOTES

5.1 Equations

Equations should be placed on separated lines and numbered. The number should be on the right side, in parentheses.

$$E = mc^2. \tag{1}$$

5.2 Figures, Tables and Captions

All artwork must be centered, neat, clean, and legible. All lines should be very dark for purposes of reproduction and artwork should not be hand-drawn. The proceedings will be distributed in electronic form only, therefore color figures are allowed. However, you may want to check that your figures are understandable even if they are printed in black-and-white.

String Value	Numeric value
Hello ICMC	2021

Table 1. Table captions should be placed below the table.

Numbers and captions of figures and tables always appear below the figure/table. Leave 1 line space between the figure or table and the caption. Figure and tables are numbered consecutively. Captions should be Times 10 pt. Place tables/figures in text as close to the reference as possible, and preferably at the top of the page.

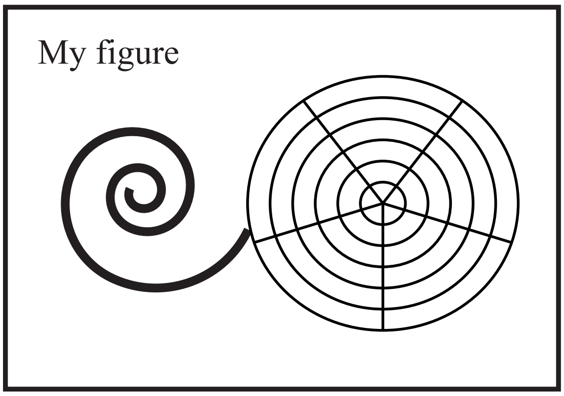

Figure 2. Figure captions are placed below the figure, exactly like this.

Always refer to tables and figures in the main text, for example: see Figure **??** and Table **??**. Place Tables/Figures in text as close to the reference as possible. Figures and tables may extend across both columns to a maximum width of 17.2cm.

Vectorial figures are preferred. When using `Matlab`, export using either Postscript or PDF format. Also, in order to optimize readability, the font size of text within a figure should be at list identical to footnote font size. If bitmap figures are used, please make sure that the resolution is enough for print quality.

5.3 Footnotes

Indicate footnotes with a number in the text.[1] Use 8 pt type for footnotes. Place the footnotes at the bottom of the

[1] This is a footnote.

page on which they appear. Precede the footnote with a 0.5 pt horizontal rule.

6. CITATIONS

All bibliographical references should be listed at the end, inside a section named "REFERENCES".

References must be numbered in order of appearance, *not* alphabetically. Please avoid listing references that do not appear in the text.

Reference numbers in the text should appear within square brackets, such as in [**?**] or [**?**, **?**].

The reference format is the standard IEEE one. We recommend using BibTeX to create the reference list.

7. CONCLUSIONS

Please, submit full-length papers. Submission is fully electronic and automated through the Conference Management System. Do not send papers directly by e-mail.

Acknowledgments

At the end of the Conclusions, acknowledgements to people, projects, funding agencies, etc. can be included after the second-level heading "Acknowledgments" (with no numbering).