A Proposal for Adding Trill Playback to MuseScore

The Issue

There is currently no built-in support for the playback of trills in MuseScore (i.e. where the *tr.* Symbol is associated with a particular note). Because this would be a useful feature (certainly for me but also, I suspect, for many other users), I propose the following approach. This paper can be considered an informal design specification for some eager programmer, although some preparatory changes would need to be made to MuseScore's core structure.

Known (Suspected) Difficulties with Trill Playback

Several reasons have been offered for the fact that trill interpretation and playback have not been implemented in MuseScore to date. It seems to me that the primary difficulty facing someone diving in to code a solution is that they would have a difficult time knowing where to begin. User "rj45" summarized this more than two years ago in a forum posting on March 6, 2011:

> "The problem is that there are many different interpretations of how trills should be played, so the software developers (thus far, and as far as I understand it) have decided to avoid the issue. Therefore MuseScore doesn't play back trills." [1]

If anything, "many different interpretations" is probably an understatement. Performance practices, and even the actual definition of "trill" [2] have varied over the years. The execution of trills during the Baroque period (the music of Johann Sebastian Bach, Georg Friedrich Handel, Domenico Scarlatti – 1685 was a great year! – Sylvius Leopold Weiss, and others) was quite different from that of the Classical (e.g. Ludwig van Beethoven and Franz Schubert) and Romantic periods (e.g. Franz Liszt, Sigismund Thalberg and the incomparable Charles Valentin Alkan), and far removed from modern practices, including the performance of Jazz and Popular music in general.

Even when acknowledged experts prescribe styles (Carl Phillippe Emmanuel Bach's "Versuch," for example, is still the primary reference to Baroque ornamentation practices after two hundred years in print[3]), they often don't fully agree with each other. And if Vladimir Horowitz or Glenn Gould didn't play Baroque trills exactly as the books prescribe, I'm not sure who exactly has the qualifications to argue with them.

So, the difficulties are real, and I understand them, but I think many of them can be avoided or successfully addressed – only partially perhaps, but enough to enhance MuseScore's utility. But before describing how I believe this issue can be addressed in a useful manner, let's quickly review some of what users do to accomplish playback of trills. This will help further identify and clarify the practical difficulties that would need to be overcome to develop a useful enhancement to MusScore.

[1] Search for "play trills" in the forum to see the full context of this and various other quotes presented here.

[2] … or "shake" as a trill is also known.

[3] Being a well-respected composer and performer in his own right, son of J.S. Bach, Georg Phillipe Telemann's godson, and a good friend of Mozart's, it seems reasonable to assume that Phillipe knew what he was talking about.

Current Workarounds

MuseScore users seem to have come up with a number of methods for playing back trills that satisfy their own needs. In his posting of March 7, 2011, Marc Sabatella, for example, offered the following workaround:

> "You can make it play them yourself by entering the notes you'd like to *hear* in one voice and making them invisible, and the notes you'd like to *see* in anther voice and making them visible. Or using a separate staff for playback only." (again, see footnote 1)

Difficulty with Current Workarounds

Marc's solution (as well as others that have been posted) seems tedious to me for several reasons:

- None of them are very generally useful solutions – that is, they don't (and can't) cover a variety of situations in the same manner, and the trill notes need to be entered individually.
- For Marc's approach, the individual measures need to be hidden, since MuseScore doesn't (at least currently) support hiding and showing entire voices. This also would be a desirable feature but, as there are likely unrelated reasons for not implementing this as well, it seems out of scope for this discussion.
- Even if the notes in "playback" voices are hidden, they adversely affect the printed layout.
- Even if there is only one note in a measure that has a trill, the resulting printed layout appears to be stretched out excessively. And if only some measures have hidden voices, the score simply looks very odd when printed – a shame given the output quality that can be currently achieved otherwise.
- A hidden voice can still affect visible stem directions undesirably – another circumstance that detracts from the score's appearance.
- Each staff in which a trill occurs would require devoting a separate voice to playback and then hiding all the notes of the trill for display or printing, thus reducing the number of voices available for their original purposes.

An Observation: The idea that there is an innate conflict between the requirements for music display and music playback notation contains the seeds of a possible solution to this situation, but we'll let that go until later in this paper.

First Disclaimer: it will be fairly obvious that my primary focus is on stringed keyboard instruments, but this proposal will also accommodate playing trills on other instruments sufficiently to permit the feedback needed to refine its implementation.

Second Disclaimer: The approach described in this proposal might be considered suitable for some other types of ornamentation, but I believe attempting this in the initial effort would be an unnecessary distraction, so I don't recommend considering that until the issues with this proposal are worked out.

The Proposed Approach

It is in the spirit of "something's better than nothing" and "you have to start somewhere" that I offer the proposed development plan described in the remainder of this document. I'll begin with a statement of the objectives that I believe are important:

Defining the Objective

Put in place an experimental solution that permits a single MuseScore file to be utilized for printing, display, screen capture, playback, and whatever else of trills without *requiring* explicit hiding of staffs or voices, generation of parts that would otherwise not be needed, and so forth. To do this:

1. Give the user full control over the definition of the trill to be played, and not force any particular interpretation. Recognize that, as As Marc Sabatella said in another posting[4]:

   "I'd agree that some sort of automatic playback for trills and other ornaments would be nice, but no automatic facility is likely to do exactly what one wants."

2. Permit users to save their favorite trill styles/definitions in a human readable form – however primitive – for easy reuse, permitting them to benefit from some stylistic consistency – at least within their own creations, and perhaps even in each score.
3. Permit users to "share" their desired/suggested trill executions with others by sending or posting the resulting trill styles/definitions.
4. Good user interface design principles suggest that we permit operation with either keyboard or mouse, and furthermore permit the continued use of existing keyboard shortcuts to the extent possible in order to conform as much as possible to existing usage.
5. Any implementation should be as transparent as possible to users who have no interest in the capabilities described here. Those who don't care shouldn't be forced to consider the issue or adopt any new work habits because of any additional functionality.
6. Adhere to, or at least avoid any overt conflict with, potential future implementation of any standards for the definition of trills.[5]
7. Finally, in order to minimize integration headaches, plan the design process in a manner intended to minimize dependencies within the code base. Although this is probably not completely possible, the preferred approach would:
   - Establish an appropriate road map for the eventual design.[6]
   - Based on that design, first make any anticipated changes to MuseScore's core code that will be needed to support the eventual design. Insure through systematic testing that the core capabilities of MuseScore continue to function properly.
   - Only then could coding of the proposed new features begin. This approach will serve to minimize integration headaches, help minimize the need for further ongoing alterations to the core application's code base, and permit independent development without interfering excessively with existing projects.

[4] On March 8, 2011 at 3:36pm

[5] And what standards might those be? More about this later.

[6] This paper is intended as that first step.

Phase I – Adding Fundamental Capabilities

Assume, for example, that we have a measure with a top staff such as the following:

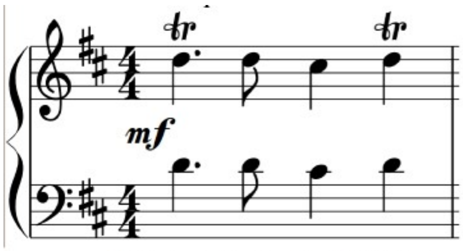


Figure 1: Opening measure of K-490

Figure 1 shows the opening bar of Domenico Scarlatti's keyboard Sonata in D Major, K-490.

Note that widely accepted performance practice, at least for Scarlatti's keyboard works, is to execute the prescribed trill in both parts (i.e. with both hands) in parallel passages such as this[7]. The implications of this will be discussed much later in this proposal. I'll be using this measure (specifically the first dotted quarter note in the top/right hand staff) to illustrate most points in this proposal.

Choosing an Approach

There are, of course, many possible approaches to addressing the playback of trills. Of these, I've selected a group of choices that I believe are practical, work well together and, most importantly, are feasible to implement, albeit perhaps not without some difficulties. Please, therefore, don't judge any of the individual suggestions without considering them in the context of the whole proposal.

There are currently four voices available for each staff or instrument in MuseScore. The underlying premise, and a prerequisite, of this proposal is the creation of a Shadow Voice for each "normal" voice that is in use. Creation of such a "Shadow Voice" will be the basis for implementing the capabilities proposed in this document.

The characteristics of the Normal (existing) Voice(s) would need to change as follows:

- Each Normal Voice would continue to be used for all Screen Display and Print activities, but would be ignored in favor of its respective Shadow Voice during any activities regarding playback / sound generation. Some exceptions to this might include:
  - Fingering indications – see "Optional Phase III-b – Printing or Displaying the Trill Execution" on page 22.
  - Instrument names would remain on the print version; instrument playback information would possibly need to be associated with the Shadow.
- Copying a trilled note in the main window should also copy the hidden playback voice where one has been defined.
- Adjusting the pitch of a Principal note in the main window should adjust all pitches corresponding to that Principal in the Shadow Voice. Assuming that a Principal was moved from "e" to "f" in the key of C Major, a corresponding trill of "d-e-d-e-d-e-d-e" in the Shadow voice would be changed to "g-f-g-f-g-f-g-f."
- A new entry would need to be added to the trill symbol's right-click menu titled "define trill" or "edit trill" that will serve as the "hook" for implementing this proposal.

Each Shadow Voice would, by default, have the following characteristics:

[7] Editors sometimes indicate this, at least for the first instance, with a "courtesy" trill marking (a smaller version of the trill symbol in parentheses) over the left hand note, but I haven't located such a symbol in MuseScore.

- At creation, each Shadow Voice would be automatically set to correspond to the Normal Voice, with the exception that no information regarding elements that relate only to display (e.g. various types of text) need be copied.
- Individual notes in each Shadow Voice may be replaced by the user with a grouping of replacement notes, so long as the timing constraints of the Principal Note are honored. The proposed method of doing this is presented in "A Straw-Man Dialog and Editor" beginning on page 6 below.
- Dynamic notations such as forte markings or crescendos that are supported (and, unlike trills, honored during playback) by MuseScore would remain on the Normal Voice, but their effects would impact and be recorded in the Shadow Voice.

To accomplish this, it might seem that the easiest approach would be to permit a user to switch the current editing screen between the display of Normal Voices and the display of the Shadow Voices but, in my view that would be a misguided approach for the following reasons:

- It could be potentially confusing to a user. It would be too easy to "forget" whether the Normal or Shadow layer is the one being edited, causing a user to be prone to mistakenly enter notes in the "wrong" voice, particularly if the same commands and keyboard shortcuts are to be used for both entry tasks (obviously desirable).
- It would present a disjointed appearance to the user. Consider that, in any score with even a small number of notes with trills, measure 24 could be on page 2 when the Normal Display is active and available for editing, and on page 3 when the Shadow Display was open and active. Switching between the Normal and Shadow Voices could easily become disorienting.

Finally, any changes to the pitch of a note in the normal voice would need to be mimicked in the Shadow Voice as described earlier.

The file format should be altered to accommodate saving the Shadow Voices along with the Normal Voices, while still permitting it to be read by older versions of MuseScore. As a workaround, and to facilitate more rapid development and testing of the core capabilities proposed in this document, there could perhaps be an option to save in the new and old format?

Testing should, of course, be thorough, since display and playback capabilities are at the core of MuseScore's capabilities.

Thus, this proposal would require a substantive change to MuseScore (creation and use of Shadow Voices for each Normal Voice) before any independent development on the remainder of this proposal can begin. But it must be recognized that the requirements for playback are are inherently in conflict with those for presentation (display and printing)[8], so such a change is actually a desirable objective and will, as can be seen, open up other opportunities for future enhancements.

Now, to Phases II and III – adding the utilities necessary to take advantage of the Shadow Voice(s) to define, edit, and play back trills.

[8] This certainly seems to be the core issue behind the complexity of trill playback.

Phase II – Editing the Shadow Voice

In order to continue with implementation of this proposal, the MuseScore core must have been modified to create Shadow Voices and use these for playback. Once that has been done and tested (a whole separate subject beyond the scope of this proposal), construction of a suitable trill editor can begin. This section will describe what I would like to see in such an editor.

The layout of the dialog boxes presented below is not necessarily meant to be taken literally, but should be reasonably similar; they are "visual aids" to the functionality described below – and they will serve as a method of organizing the discussion in a way that follows the anticipated usage of the dialog, so that other readers can respond.

A Straw-Man Dialog and Editor

In Figure 1 on page 4 above, the very first note for the right hand is a dotted "d," which happens to be the tonic note of D Major, the key in which the piece is written. We will refer to the note for which a trill is to be added by the commonly used term "Principal Note."

To define a trill, the user will right-click on any ***tr*** symbol, and choose "define trill" or "edit trill" – whatever new entry was added to the symbol's right-click menu in Phase I of this proposal. Choosing this option for the trill on the first principal note ("d") of Figure 1 on page 4 above would display a modal dialog box similar to that below:

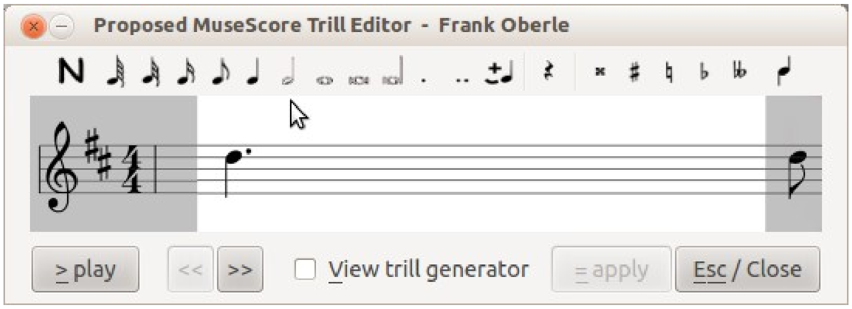


Figure 2. A proposed dialog for editing the new Shadow Voice in MuseScore

Why modal? And why not simply use the existing canvas to edit the shadow voices? And why not some other approach? The reason is that, because good user interface design practices make it quite desirable to support entry with both keyboard and mouse, and because the number of options required to implement semi-automatic (but user-controlled) trill generation (we'll get to that in Phase IIIa on page 10 below) is fairly significant, it is inevitable that conflicts with MuseScore's standard key mappings will arise.

Using a modal dialog box, while possibly seeming to be overkill, seems the most appropriate approach when considering the purpose of this project and, as it turns out, there will be other advantages to doing so.

When the dialog is displayed for the first time, the Principal note from which the trill editor was invoked – in this case Scarlatti's dotted "d" – can be seen in an edit window in the center of the dialog box. This is shown with a white background here, but would presumably use whatever background color the user has chosen for the main MuseScore score.

Above the edit window is a grouping of icons identical to those on the main MuseScore canvas and that function in the same manner. Note, however, that in this example, since the total note length available for replacement (6/4) would not accommodate any note longer than a quarter note, the icons (as well as the equivalent keyboard accelerators) for the half note and longer values are disabled[9]. This use of the familiar icons and keyboard equivalents (e.g. “4” for an eighth note, “5” for a quarter note, “+” to set a tie, etc.) will reduce any learning curve.

In the center below the note icons is the window where the Shadow Voice buffer[10] can be edited by the user. Because the window’s contents are from the Shadow Voice, the trill marking doesn’t appear. This is because, as noted earlier, the Shadow Voice doesn’t have or need this information.

Limits of Editing Capability

When initially opened to edit a trilled note, the edit window contains a single note. This could, of course, be any valid length permitted by MuseScore, which could range from a thirty-second note[11] to a whole note. The size of the edit window should probably be sufficient to handle at least sixteen replacement notes, but likely never more than thirty-two. The edit window as shown can accommodate fourteen, but that is an utterly arbitrary choice.

Continuing

To the left of the edit window, the Clef, Key Signature and Time Signature in effect for the Principal note are displayed – whether or not these are adjacent to that note in the main MuseScore display. The remaining gray space between these elements and the editing window will display the note immediately preceding the Principal note in order to provide some context for entering the trill playback sequence. In this example, of course, no note is displayed since the Principal happens to be the first note of the score. These elements are for display only, and cannot be edited in this dialog.

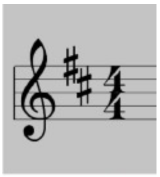


Figure 3.

To provide similar context, the note following the Principal is displayed to the right of the editing window and, similarly, cannot be edited.

The one partial exception to this, however, is that it would be useful if a tie can be extended from the final note in the editing window to this note, particularly if the Principal Note already has such a tie in the Displayed Voice.

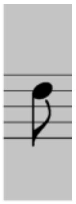


Fig 4.

Below the editing window is a series of buttons and a check box; the purpose of each of these is described below:

[≥ play sequence] :

During editing of the trill, it is likely that the user will wish to play it multiple times in order to refine the definition. In the normal editing screen, this would be probably be done by setting the cursor to a desired start point for each test, initiating playback, and then stopping playback before very long.

With the dialog box being modal, it would be tedious to keep closing it in order to test the effect on playback, so the intent of the [≥ play] button

A note about accelerator keys:
In this case, I’ve specified the Alt+“>” key, but since MuseScore accepts single letter commands without the need for simultaneously using modifiers, it is my expectation that, while Alt+“>” should work, the [space] key (MuseScore’s key command for “Play”) should also work. Likewise, the accelerator keys suggested in the remainder of this paper were chosen to permit use both with and without modifiers.
The advantages of this approach will become more apparent in the description of Phase IIIa below.

[9] This can be seen just above the mouse cursor in the illustration. In similar situations, MuseScore currently pushes the following notes forward if the entered note is longer then the current measure, but this wouldn’t be appropriate in the edit window described here.

[10] Not the Shadow Voice itself – see the description of the [≡ apply] button later for an explanation.

[11] MuseScore permits sixty-fourth notes, but what value would then by used to play a trill?

is to set a playback start point automatically, and then initiate playback until a prescribed stop point. What is important to note is that, until the contents of the editing window have been “committed” to the Shadow Voice (with the [apply] button; see below), playback would need to alternate between the Shadow Voice and whatever buffer is created for the edit window.

For initial testing, it is probably sufficient to simply play the entire sequence in the display buffer, although eventually it may be desirable to permit the user to preset choices such as “start at beginning of previous measure,” and so forth. Use of a prototype by more people will likely result in some consensus on how much should be played or what choices would be useful to establish context.

It seems to me that it would be most helpful if all voices were played for the duration represented by the displayed notes, but this might not be suitable for an orchestral score. This would also complicate playback. For the moment (for testing) it is probably sufficient to only play the affected voice and plan to add more user configuration capability (if desired) at some later date.

[<<] or [Page Up] :

This button is intended to permit the user to have the editor jump back to display any previous note (in the same voice) that is marked with a trill. This would permit editing of multiple trill playback sequences to be handled without having to continually return to the main MuseScore display and invoke the right click menu for each trill marking.

In this example shown in Figure 2, since the Principal note is the first note in the voice that is marked with a trill, the [<<] button is disabled.

[>>] or [Page Down] :

This button is intended to permit the user to have the editor jump forward to display the next note (in the same voice) that is marked with a trill.

Conventional practice suggests that, if changes have been made in the editor, but not yet applied to the Shadow Voice, an appropriate warning would be presented if either [<<] or [>>] is selected, so the user can have a chance to apply those changes prior to displaying a different Principal note.

□ View trill generator :

This unchecked box will be described later, but should be present in any initial design to account for its location on the dialog box layout, since it will be required to access the semi-automatic user-controlled trill generation functionality proposed later in this paper. See “Phase III-a – Semi-automatic Trill Definitions” beginning on page 10 for more details if you have no patience.

[= apply] :

The purpose of this command is to actually update the shadow voice from the display buffer containing the newly defined trill. This will permit experimentation before committing changes to the score.

Because “A” is a valid command in MuseScore (it enters the musical note “A”), and Ctrl + “A” is used to select the entire score, I’ve specified the “=” key to indicate that this is intended to to make the Shadow Voice “equal” to what is in the edit window.

Since we are assuming for this example that no changes have yet been made yet, this button is disabled and, of course, the [=] key will do nothing. Any edits made to the window should, of course, enable the button and the [=] key.

[Esc / Close] :

There is no accelerator key marked for this button; the intent is that this will be equivalent to either using the [Esc] key or clicking on the close/quit button of the window itself.

And, of course, if changes have been made, but not yet applied, the user would be offered the opportunity to either commit or abandon them prior to closing the dialog.

Using the Editor

Use of the edit window described here should be as similar as possible to entering and editing notes on the main score, with the following exceptions:

- no aesthetic spacing corrections would be applied, since this is not intended for normal display and, indeed, spacing isn't relevant to the Shadow Voice we are actually changing.
- cursor movement would be restricted to the time value of the note to which the trill marking is attached. As discussed earlier, the transcriber or composer can directly edit the notes, but will be constrained by the duration of the Principal rather than the duration of the measure as in the main screen.
- Stem direction in this window would be meaningless, except possibly in the case of trilled chords. Although unusual, these are discussed later in this document.
- There should still (eventually) be support for dynamics, slurs, and such things within the trill editing canvas, but I've provided no examples of these, since there are a number of issues that might arise.[12]

As described earlier, none of the changes will be written to the actual Shadow Voice until the user chooses to commit them by using [≡ apply].

So far, all that has been accomplished is to permit a user to add instructions for the playback of trills without having them interfere with the presentation or layout of the score.

In my view, an addition such as this would present a very useful improvement to MuseScore on its own. It seems to me, however, that usability could be greatly enhanced by some additional utility exposed by activating the [View trill generator] check box. The purpose of this will be described with the expanded dialog shown in the next section.

Two independent project phases could be undertaken after everything described above has been coded and confirmed to work successfully. These would:

- Permit a method for users to describe a "short-hand" for automatically building trills to be inserted in the Shadow Voice. This is the ultimate objective of my proposal and will be described in detail under "Phase III-a – Semi-automatic Trill Definitions" beginning on page 10.
- Allow teachers and others to print "hints" for trill play on practice scores. I will refer to this as "Optional Phase III-b – Printing or Displaying the Trill Execution", and it will be described on page 17.

[12] See "Gotchas and Future Refinements" on page 22 for one example.

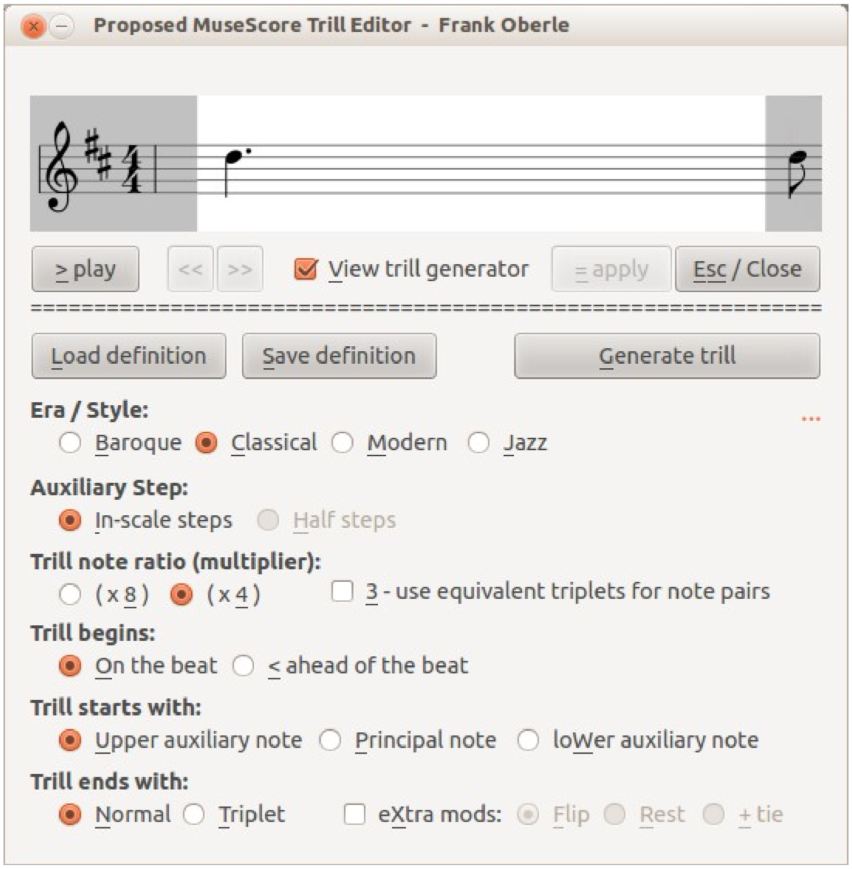
Phase III-a – Semi-automatic Trill Definitions

Any work on implementation of this phase would be very dependent on completion of Phases I and II, and would provide what I believe will be a very useful tool for those composing, arranging or transcribing music, particularly if the user's intention is for MuseScore to play it.

Although the minimal trill editing dialog shown above neatly solves several of the more annoying aspects of the failure to provide trill playback in MuseScore, it is obvious to anyone who has entered detailed trills into a piece of any length that direct entry of these sequences still represents a level of tedium that cries out for computer assistance. This is how I propose that tedium be mitigated.

□ View trill generator

Upon completion of this phase of the proposal as I describe it below, pressing the "v" key or clicking on the [View trill generator] check box would expand the dialog box created in phase II. Assuming that we've started from the beginning – before making any edits – the dialog box is once again displaying the principal note "d" from Scarlatti's piece referenced above (i.e. no manual editing has been done), the extended dialog will initially appear similar to that shown in the following illustration.



☑ View trill generator

Era / Style: ○ Baroque ● Classical ○ Modern ○ Jazz

Auxiliary Step: ● In-scale steps ○ Half steps

Trill note ratio (multiplier): ○ (x 8) ● (x 4) □ 3 - use equivalent triplets for note pairs

Trill begins: ● On the beat ○ < ahead of the beat

Trill starts with: ● Upper auxiliary note ○ Principal note ○ loWer auxiliary note

Trill ends with: ● Normal ○ Triplet □ eXtra mods: ● Flip ○ Rest ○ + tie

Figure 5: The expanded Trill Editor, with the Trill Generator controls

Like the earlier example, this one is intended as a "visual aid" for following the detailed descriptions given below, and isn't necessarily meant to be interpreted literally[13].

The first thing to notice is that the row of MuseScore icons is no longer available while the generator dialog is exposed.

The panel used earlier for editing the trill now serves only to display whatever trill is generated, so neither the icons nor their respective accelerator keys will be recognized.

The ability to play the displayed sequence remains, but the ability to jump to the previous or next principal note in the main score with a trill marking is disabled while the extended dialog is open.

Unchecking the "View trill generator" box will hide the extended dialog and return it to its previous conditions. Additionally, the [Esc/Close] button now closes only this extended dialog (i.e. it is equivalent to using the "v"ac-

[13] Although I hope if anyone eventually codes this, I won't be terribly disappointed with the result. ☺

celerator key to uncheck the “View trill generator” box again), and would therefore need to be pressed twice in order to completely close the utility.

When this dialog is extended as shown in the illustration above, we are switching from what was a “user edit mode” to a semi-automated “generator mode.” The most important change to be aware of is that none of the previously available accelerator keys or commands are available when the extended trill generator dialog is visible except for those elements immediately below the editing window. Until the extended dialog is closed, most of these are replaced by those described below. The list of keys that function differently when the extended trill generator dialog is open include the following:

- The letters “a” through “g” will no longer enter the respective musical notes; while the extended dialog is open, the user should not be able to edit anything at all in the edit window. Of these keys, “b,” “c,” “f,” and “g” have different meanings.
- The letter “h” will no longer set a crescendo, and has a different purpose.
- Since the user cannot enter any notes, the “n” key will not set the note entry mode, but will have a different purpose.
- The “q,” used to indicate that the next note should be half the previous note’s duration, will be inactive, although is not proposed to be used in this extended dialog.
- The “r” key, used to repeat the previous note, is given a different purpose.
- The “x” key, used to flip stem direction and some other elements during normal score editing, is reassigned to a different function in the extended dialog.
- The “+” key, which normally sets a tie between a note and the one that follows, is given a similar, but slightly different function in the extended trill generator dialog.
- The keyboard accelerators “0” through “9,” used to select various note duration values no longer serve that function, although some of them are used for other purposes.
- Finally, as stated earlier, neither the [<<] (Page Up) or [>>] (Page Down) keys are enabled while the extended trill definition panel is open.

Proceeding from left to right and top to bottom, the new functions of each key in the extended dialog are described below:

[Load definition] and [Save definition] :

These first two buttons are probably self-explanatory. The [Save definition] button will permit a user to save the settings displayed in the extended dialog (all described below) to disk with the ability to give them a descriptive name.

Assuming that there are any previously saved definitions, the [Load definition] button will permit one of them to be loaded to reset all of the radio buttons and check boxes automatically.

At the moment, of course, these would seem to be useful to only the truly lazy, since setting several buttons isn’t much more time-consuming than going through a file selection and load process, but this will support whatever expansions might be desirable in the future.

[Generate trill] :

This final button in the top row of the extended dialog will perform the following actions:

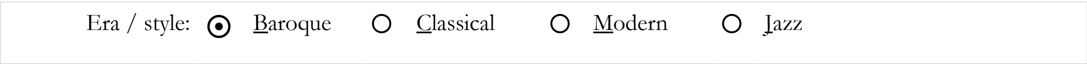
- Clear the entire note entry section in the upper portion of the dialog as well as whatever buffer is associated with it. Note that this does *not* alter the Shadow Voice.
- Replace the contents of the note entry section and its buffer with an equivalent trill based on the settings in the remainder of the dialog.
- Since the contents of the edit box have been altered, the [= apply] button will be enabled if it hasn't already been enabled by prior change activity.
- Write the mnemonic for the active settings as a reminder of what settings will be saved in the section below the button. This is intended to be temporary – solely for demonstration purposes – while the functionality of the trill generator is tested. It can be removed at a later date. In the above example dialog, the contents of this mnemonic are ". . ." which indicates that no trill had yet been generated during this session. Once generation has taken place, the contents of this mnemonic will be set to "ci40oun."

  The mnemonic I'm using actually works well enough for this proposal, but in any implementation subsequent to the "beta" stage, the trill definition should be saved in some form of text based/human-readable format so that it can be more transparent, edited independently, shared with others and so forth. Whether this is an .ini file for Windows, a .config file for Linux or whatever could be dependent on whatever operating system is in use, although an XML-based (i.e. non OS-specific) approach might be more suitable. Objective #6 (see page 3) notwithstanding, the only thing approaching a standardized model to follow that I'm aware of would be that of MusicXML[14], but it doesn't yet appear to be extensive enough to support even the minimal definitions proposed below.

Below the [Generate trill] button is a series of nine parameters arranged in six rows that are used for semi-automatic generation of a trill. Each grouping (usually one per line) represents the domain for its respective variable, however that might eventually be implemented. The purpose of each selection and its impact on playback is likely self-explanatory, but more details about an implementation approach will be provided in "An Explanation – Generating a Trill from the Dialog Settings" which begins on page 18.

For the moment, it is not necessary to care how, for instance, the "Era / Style" parameter came to be set to "Classical" instead of "Jazz" in Figure 5 above, only that there will be some system default used if no previous definitions have been made by the user. The next several paragraphs will cover each of the variable domains, most of which are presented on the screen as radio buttons:

Sets varStyle = "b," "c," "m," or "j" (default = "b")



Era / style: ◉ Baroque ○ Classical ○ Modern ○ Jazz

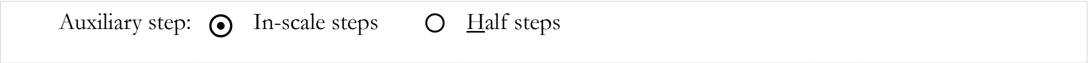
Unlike the remainder of the settings, the "Era/Style" selection will have no effect on trill generation, and is included solely for a user's own classification – a means of identification to aid in building and organizing a collection of customized default settings.

For initial proof-of-concept experimentation, four musical era styles are listed and shown here as a set of radio buttons, although at some later time this list could be expanded. Whatever underlying

[14] I'll refer to MusicXML several times in this document. For specific information for those not familiar with this, browsing the web site http://www.musicxml.com/for-developers/alphabetical-index/ will be helpful.

mechanism is chosen to save any particular trill playback definition should include an era or style as an attribute.

Sets varStep = “i” or “h” (default “i”)



Auxiliary step: ● In-scale steps ○ Half steps

When a trill is performed, it generally involves a relatively rapid alternating playback of a pair or triad of notes. The Principal is, of course, the note over which the trill symbol is placed, and the one played currently by MuseScore. The other note used in the trill is typically referred to as the Auxiliary, and may be either higher or lower than the Principal[15].

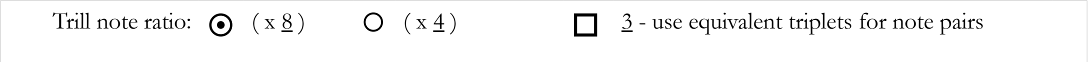
Using in-scale steps simply means that only other notes in the currently defined Key Signature could serve as Auxiliary notes; whether these were whole or half steps would depend on the Key (e.g. in the key of C Major, an “f”-”g” pair (fa-sol) would be whole steps, while a “b”-”c” pair (ti-do) would be a half step.

Even if the Principal itself is an accidental in its respective key, the same rules would apply. If we had a b-flat note on a C Major staff, the auxiliaries would be “b” (a half step) and “c” (a whole step).

The half-steps option would always use notes that were a half step above and below the Principal as auxiliary note choices. The half-steps option would probably be quite inappropriate for Baroque and Classical music[16] but, again, we need to avoid the temptation to judge.

Sets varMultiplier = “4” or “8” (default “8”)

Sets varTriplet to “0” or “3” (default “0”)



Trill note ratio: ● (x 8) ○ (x 4)

□ 3 - use equivalent triplets for note pairs

It is important to note that a multiplier of 8 does not, in and of itself, indicate that a trill would replace the Principal note with eight shorter notes (four pairs of notes). The contribution of the Multiplier to the generated trill will be described in more detail in the section “An Explanation – Generating a Trill from the Dialog Settings” beginning on page 18.

By most musical definitions, there must be a minimum of four notes used to replace the Principal to qualify as a trill. For the purposes of this proof-of-concept, and perhaps under any circumstances, it would likely be unusual for a Principal to be replaced by more than sixteen notes, although a relatively slow tempo might justify as many as thirty-two.

Lang Lang is capable of executing much faster trills than the average piano student. If a teacher wishes to use the MuseScore file to provide playback examples for less gifted students or beginners, particularly if the intention is for the student to “play along,” this choice would easily permit that.

It certainly could be argued that the tempo specified for the score will need to be considered relative to the length of the trill notes, if only to give a warning. Converting a quarter note to a trill made up of eight thirty-second notes at Lento is quite a different matter than doing the same thing at Presto. Again, for purposes of a proof-of-concept, attempting this would probably violate objective #1.

The fact that the smallest note MuseScore supports is a 64th note could also complicate things unnecessarily for some shorter Principal notes at slower tempos if any initial proof-of-concept attempts to be too smart. Such things should be ignored for a working proof-of-concept.

[15] Yes, it’s true that some trill styles may involve both higher and lower auxiliaries, but that will be discussed later.

[16] Mozart died young enough as it was; encountering such a trill likely would have precipitated an earlier demise.

The “3” selection will replace any specified number of trill pairs with trill triads. A little thought will show that each triplet in such a trill will be the opposite of the previous triplet (e.g. “d-c-d” would be followed by “c-d-c” and vice-versa). Choosing this option will also replace any “Trill ends with: Triplet” to “Trill ends with: Normal,” since the two selections would be inconsistent.

Sets varBeginningBeat = “o” or “<” (default “o”)

Trill begin: ● On the beat ○ ≤ ahead of the beat

For a Baroque era style, this would certainly default to beginning “on the beat” or “in time” as some would say, but both choices are justified by long-standing performance practices.

Implementing a trill that begins before the beat would, of course, require “stealing” some time from the previous note or measure. In the case of the Scarlatti example on page 4, there would even be a requirement to add a pick-up measure. Since several difficult decisions would need to be made in that case (e.g. should an apparently empty pick-up measure be displayed? Would that be confusing to many users?), I would suggest simply not permitting this variable to be changed in a prototype. The choice to begin a trill before the beat is ultimately reasonable, however, so the radio button should remain as a place-holder for possible future implementation, and selecting “Trill begins: ≤ ahead of the beat” should have no effect other than to pop up a “Not available – Coming Soon” warning.

Sets varStartWith = “u,” “p,” or “w” (default “u”)

Trill starts with: ◉ Upper auxiliary note ○ Principal note ○ loWer auxiliary note

Once again we have an example of how matters of musical taste should be ignored, at least for this proof-of-concept. Generally recommended performance practice for Baroque keyboard pieces suggests that a trill must always begin on the upper auxiliary note, suggesting that the “loWer auxiliary note” option should only be available if “Trill begins:” is set to “ahead of the beat.” And again, while a later implementation might allow users to define some “rules” for various such situations, no such restraints need to be considered for an initial proof-of-concept.

If preceded by a note with the same pitch as the auxiliary note to be used (if one is used), Baroque performing practice suggests that the initial note be tied to that preceding note.[17] As with other niceties, it would be inappropriate to spend time implementing this condition in a prototype but, like other examples I’ve given, keeping it in mind may help precluding its later implementation.

Sets varEndWith = “n” or “t” (default = “n”) Sets varXtra = “xf,””xr,” or “x+” (default = null)

Trill ends with: ◉ Normal ○ Triplet | | □ eXtra mods: ○ Flip ○ Rest ○ ±tie

This offers the option of replacing the final pair of a trill – assuming there is more than one pair[18] – with a triplet, and will be illustrated in the examples that follow.

The “eXtra mods” that can be selected if the check box is activated include:

- Flip: The second last note of the trill, assuming that it is one of the auxiliary notes, will be “flipped” to use the opposite auxiliary.
- Rest: The final note of the generated trill will be replaced with an equivalent rest. This gives a different feel to the trill and is particularly useful if the final note is identical in pitch to the note that follows, and a tie to that note is not desired.

[17] This is known as a “tied trill” and is sometimes indicated explicitly by the composer with a tie to the trilled note.

[18] A trill, remember, must by definition have at least four notes and, therefore, at least two pairs of notes.

- + tie: A tie will be placed between the final note of the generated trill and the note that follows.

If the "eXtra mods" check box is deselected, the condition of the "f," "r," or "+" choice is retained.

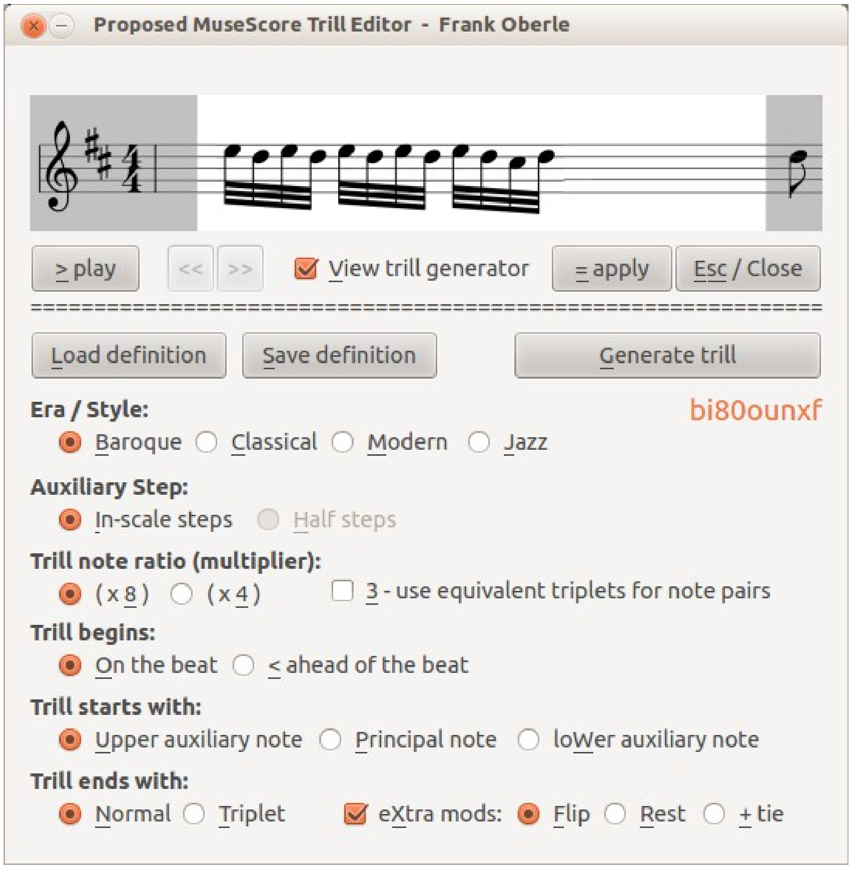
Generating the Trill

Once a trill has been defined, either by loading a previously saved definition, or by changing the settings manually, the [Generate trill] button will replace the contents of the edit window and its buffer with a trill based on the settings.

Because the contents of the editing window have changed, the [≡ apply] has been enabled so that the new trill can be applied to the Shadow Voice once tested.

In the example on the right, for instance, pressing "g" with the settings "bi80ounxf" in effect has replaced the original dotted quarter note with a six pair trill on the upper auxiliary, but with the second-last auxiliary note "flipped" to use the lower auxiliary.

The [≥ play] button (or the space bar) can now be used to test the results of the trill to the extent of whatever design decisions were made for "play" (see the discussion of the options in the description of the [≥ play] button earlier).



☑ View trill generator

bi80ounxf

Era / Style: ● Baroque ○ Classical ○ Modern ○ Jazz

Auxiliary Step: ● In-scale steps ○ Half steps

Trill note ratio (multiplier): ● (x 8) ○ (x 4) □ 3 - use equivalent triplets for note pairs

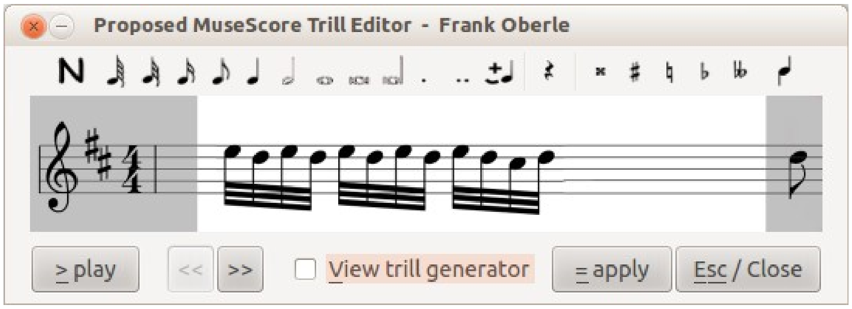
Trill begins: ● On the beat ○ < ahead of the beat

Trill starts with: ● Upper auxiliary note ○ Principal note ○ loWer auxiliary note

Trill ends with: ● Normal ○ Triplet ☑ eXtra mods: ● Flip ○ Rest ○ + tie

Figure 6: Expanded dialog after making selections shown and using the [Generate trill] command

The [≡ apply] button could be used at this point if the generated trill is what the user intended, but it is certainly possible that some manual modifications[19] may be desirable to tweak the trill.



□ View trill generator

Figure 7: Returning to user editing mode after Generating a trill

As indicated earlier, the extended dialog would need to

[19] These would, of course, be limited to whatever capabilities were implemented in the proof-of-concept edit window.

be closed[20] to return from what I earlier referred to as the "automated generator" mode to the "user edit" to permit any tweaks, including the addition of any dynamics, slurs, etc. to the generated trill.

There are, of course, issues that may arise. Suppose, for instance, that the Principal note for which the trill is to be defined is marked as "forte" in the main MuseScore canvas, and further suppose that the user marks the first note of the trill in the edit window as "piano." Or, assume that the Principal note is a "c" and we simply change it to a "b" in the edit window. We could legitimately react in a number of ways:

- Detect such situations and prohibit them.
- Detect such situations and make corresponding changes to the main Voice.
- Ignore the situation; teachers can make good use of these and similar capabilities to confuse their composition students on April Fool's Day.

As with other such quandaries, I would suggest for the purposes of a proof-of-concept implementation that the situation simply be avoided until enough user feedback is obtained to reach a consensus on how such situations should be handled. Indeed, depending on the decisions made based on the discussion in "Limits of Editing Capability" on page 7, many such issues can be safely deferred.

The resulting differences between what is displayed and what is actually played may present an some issues, but they don't need to be solved for a useful proof-of-concept implementation.

Possible Enhancements

In addition to the [Save definition] function, the Extended dialog could also have a "Save as Default" option, either as a separate button (forcing a specific file name) or as a check box on whatever file save dialog is used. This default would then be used for any new score that is created.

In addition to the [= apply] function, there could be an additional option to apply the currently displayed definition to all trill markings in the Voice or even the Score; this itself could eventually be enhanced to adapt the trill to different situations based on some predefined rules the user would be walked through with a wizard (if for instance the user desires that any ending note of a trill that matches the following note might be altered to a rest). An even further refinement would be to develop some smarter adaptation to any notes preceding or following the trill.

Summary

The intent of this dialog is to speed up and reduce the tedium of trill entry, and achieve some consistency in the process. Although it may initially appear to be an attempt to do the impossible (qv) or to impose some arbitrary "rules," it's more a cross between a macro and style setting, but without attempting in any way to force the user into any particular notion of what constitutes a "proper trill," and over which the user has as much control as desired.

[20] Done, you will recall, by clicking the "View trill generator" box or using the [Esc / Close] button or key.

Optional Phase III-b – Printing or Displaying the Trill Execution

I've called this "Phase IIIb" but it could be implemented once even a minimal capability to edit the Shadow Voice has been completed, so it is also dependent on completion of Phase II.

In many printed scores, particularly those for baroque and classical music, a small version of how the trill is supposed to be played (at least in the often not so humble opinion of a particular editor) is presented above or beside the trill marking. In the example shown on the right, for instance, a footnote is used to show the editor's suggestions for how the trill at the beginning of the second measure is to be played.

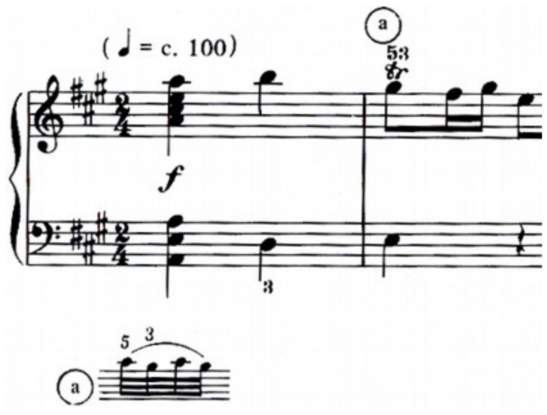


Figure 8: Example of a printed editorial footnote

Separation of print and playback voices opens up the opportunity to implement something like this capability in MuseScore.

This is likely most useful for teachers who wish to prepare customized performing or practice editions for students, but might also be useful for composers wishing to indicate their own feelings about how some portion of their composition is intended to be played.

Interestingly, such musical notation footnotes that are in the form of instructions are often likely to contain fingering. Since the footnote would be drawn from the Shadow Voice in the implementation described in this paper, there is an implication that the Shadow Voice may need to include more than simply the playable notes.

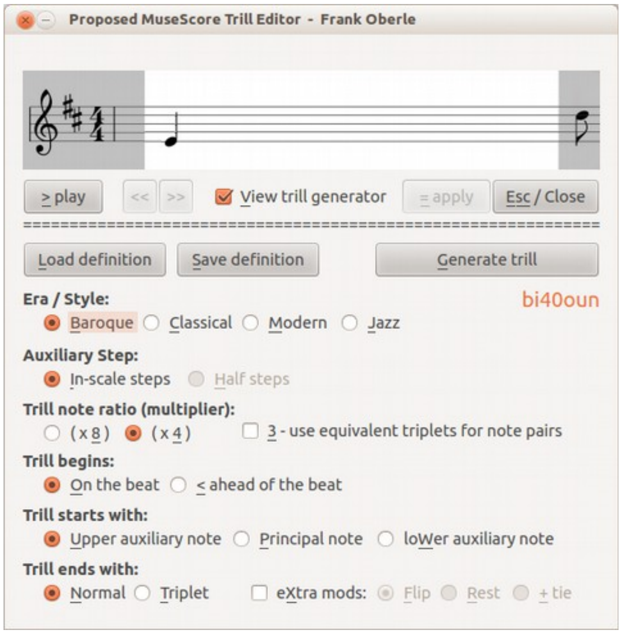
Any effort to develop this capability can likely be done in a relatively independent manner, providing another useful task for some interested developer to work without a lot of pressures based on dependencies on or upon the core code of MuseScore.

An Explanation – Generating a Trill from the Dialog Settings

The following examples each assume a Principal quarter note of "e" in the key of D Major, and in 4/4 time (with one exception), for which we wish to generate a trill based on the conditions established by the example dialog box to the right. The staff is assumed to be the upper staff of a typical piano score with a G Clef. At the top of each example is a mnemonic which reflects the accelerator letters used in the dialog box; the first of these is based on the settings shown in the illustration on the right.

Assume that all of the radio buttons and check boxes have been set as shown, whether this was done by making each of the selections manually or by loading a previously saved definition.

As shown, these settings are represented by the mnemonic "bi40oun." For demonstration purposes, this is shown below the [Generate trill] button. These mnemonics will serve as a quick reference to the relevant settings used for each example below. We'll begin with this one.



☑ View trill generator

bi40oun

Era / Style:
● Baroque ○ Classical ○ Modern ○ Jazz

Auxiliary Step:
● In-scale steps ○ Half steps

Trill note ratio (multiplier):
○ (x 8) ● (x 4) □ 3 - use equivalent triplets for note pairs

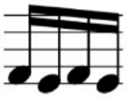
Trill begins:
● On the beat ○ < ahead of the beat

Trill starts with:
● Upper auxiliary note ○ Principal note ○ loWer auxiliary note

Trill ends with:
● Normal ○ Triplet □ eXtra mods: ○ Flip ○ Rest ○ + tie

Figure 9: Principal note of "e" prior to generating a trill

bi40oun (in 4/4 time)

Principal	Multiplier	Trill notes to be generated	Generated Result:	Legend definitions
$\frac{1}{4}$ 4/4 time	x 4 =	$\frac{4}{16}$ – that is four 16th notes, or two pair starting with the upper auxiliary note "f".	 Four (2 pair) sixteenth notes	b = Baroque style i = In-scale steps used 4 = x4 trill note ratio used 0 = no triplet replacements o = trill begins on the beat u = trill uses upper auxiliary n = trill ends normally

In this table describing how the first example is handled when the [Generate trill] button is pressed, the base quarter note "e" is shown in the first column with a marking of ¼ to its right. This seems redundant, but it is important to understand that this ¼ notation represents not the quarter note itself, but the value of the quarter note in relation to the denominator of the current time signature[21]. A quarter note, remember, does not indicate a specific length of time, but rather a duration relative to the tempo and time signature of the music. This distinction will become more clear with the next example, which shows the same quarter note in 6/8 time. Rather than concern ourselves with the stated note value (quarter note), we'll use the value of the relative equivalent note (which also happens to be ¼ in this example) when calculating an appropriate trill for playback.

Because we've specified "In-scale" steps rather than "Half steps" the auxiliary notes for the trill are "d" (lower Auxiliary – a whole step down in the D Major scale) and "f#" (upper auxiliary – a whole

[21] In MusicXML, the time signature denominator is known as "beat-type," although that term seems perversely vague. The time signature numerator seems to be the same as what MusicXML refers to as "beats."

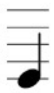
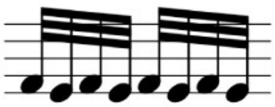
step up in the D Major scale), rather than “d#” (an absolute half step down regardless of key) and “f” (an absolute half step up regardless of key) if we had chosen “Half steps”.

The specified Multiplier variable[22] shown in the second column – in this example 4 – is used to multiply the relative value (one over four), giving four over sixteen. This indicates that the trill will be made up of four[23] sixteenth notes, or two pairs of alternating sixteenth notes.

We’ve chosen “Trill begins: On the beat,” and “Trill starts with: Upper auxiliary” for the paired notes in this trill, so the initial note of each trill pair will be the upper auxiliary, in this case, the “f#”. Therefore, two “f#-e” sixteenth note pairs will be used to replace the quarter note in playback, as shown in the fourth column above.

Since it is important to understand the distinction between the term ”Quarter note” and its value relative to the time signature (i.e. the “1 over 4” to the right of column 1 above), the next example will (hopefully) clarify this by using a different time signature.

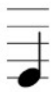
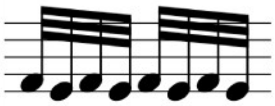
bi40oun (in 6/8 time)

Principal	Multiplier	Trill notes to be generated	Generated Result:	Comments
 $\frac{2}{8}$ 6/8 Time	x 4 =	$\frac{8}{32}$ – that is eight 32nd notes, or four pair starting with the upper auxiliary note “f”.	 Eight (4 pair) thirty-second notes	Note that, although the dialog box settings (bi40oun) are the same as in the example above, the generated trill differs due to the different time signature!

Because the denominator of the time signature in this example is given in 8ths, the relative quarter note value is stated as two-eighths rather than as one quarter; two eighths is, of course, the equivalent of one fourth.

Note that, although the dialog settings (“bi40oun”) are identical to those used in the previous example, the resulting trill is generated as eight thirty-second notes rather than the earlier example’s four sixteenth notes. The point is that no one factor alone, including the multiplier, will dictate the form of the trill.

bi80oun (in 4/4 time)

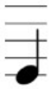
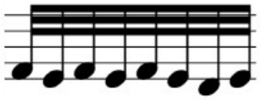
Principal	Multiplier	Trill notes to be generated	Generated Result:	Comments
 $\frac{1}{4}$ 4/4 time	x 8 =	$\frac{8}{32}$ – that is eight 32nd notes, or four pair starting with the upper auxiliary note “f”.	 Eight (4 pair) thirty-second notes	

We’ll now return to 4/4 time for all but one of the remaining examples.

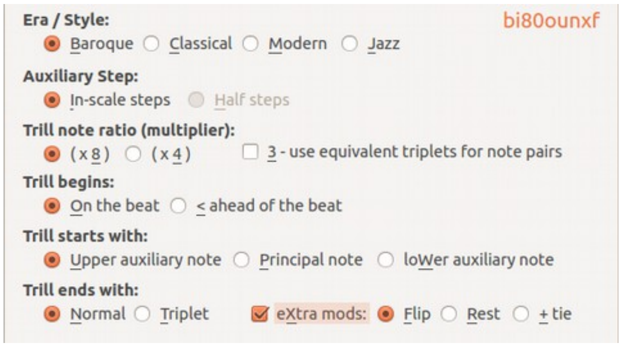
[22] MusicXML doesn’t have a direct analog of this variable, but it is implied by the “trill-step” attribute discussed below.

[23] This numerator (of the product of the relative note value times the multiplier) seems to be combined by MusicXML with what I’m calling the Multiplier in the “trill-step” attribute, and is expressed as a relationship to the length of the principal note in terms such as whole, half, etc. I believe using a separate multiplier is more understandable.

bi80ounxf (in 4/4 time)

Principal	Multiplier	Trill notes to be generated	Generated Result:	Extended Legend
 $\frac{1}{4}$ 4/4 time	x 8 =	$\frac{8}{32}$ – that is eight 32nd notes, or four pair starting with the upper auxiliary note "f".	 Eight (4 pair) thirty-second notes with a "flip" on the second last note of the trill ending	X = eXtra modifications ON f = Flip second last note to the opposite auxiliary r = change last note to an equivalent Rest s = add a Slur from the last note to the following note in the score

This (finally) is a more realistic example of a classic trill. It should be obvious by now that, with the selections shown in the figure on the right, this trill will be comprised of eight thirty-second notes, with the upper auxiliary ("f#") as the first note.


bi80ounxf

Era / Style: ● Baroque ○ Classical ○ Modern ○ Jazz

Auxiliary Step: ● In-scale steps ○ Half steps

Trill note ratio (multiplier): ● (x 8) ○ (x 4) □ 3 - use equivalent triplets for note pairs

Trill begins: ● On the beat ○ < ahead of the beat

Trill starts with: ● Upper auxiliary note ○ Principal note ○ loWer auxiliary note

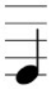
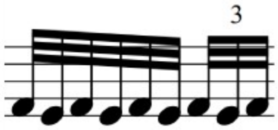
Trill ends with: ● Normal ○ Triplet ☑ eXtra mods: ● Flip ○ Rest ○ + tie

Figure 10: Settings for a "bi80ounxf" style trill

In this example, the xf at the end of the mnemonic indicates that "eXtra" modifications have been activated and the second last note of the trill should be "flipped" from the upper auxiliary ("f#") to the lower auxiliary ("d"). If it happens that the second last note is the same as the Principal – that is if it is not one of the auxiliary notes – the instruction to "flip" should be ignored. An example of one of the ways this might occur is shown in the next example.

Note that if flip is applied to a trill with the bi40oun settings (in 4/4 time – the first example), it begins to approximate a turn. While this is interesting, and does indicate that the approach to playing trills recommended in this proposal can be adapted to almost any ornament, it would seem distracting to attempt generalizing this proposal to cover other ornaments until a decent level of user feedback has been received. The Weight of this is (4 x 2) + (3 x 3) + (1 x 1) = 8+9+1 = 18 / 8 = 2.25

bi80out (in 4/4 time)

Principal	Multiplier	Trill notes to be generated	Generated Result:	Comments
 $\frac{1}{4}$ 4/4 time	x 8 =	$\frac{8}{32}$ – that is eight 32nd notes, or four pair starting with the upper auxiliary note "f".	 Eight (3 pair and 1 triplet) thirty-second notes	The Principal note is an "e" and, for most music, an ornament should "decorate" this but not alter it. The "Weight" of an ornament is a measure of the percentage of time spent on the Principal.

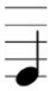
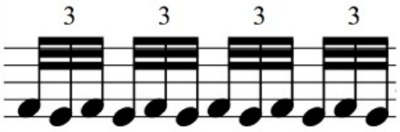
In this example, the switch from "Trill ends with: Normal" to "Trill ends with: Triplet" causes the final pair – in this case, the fourth of four – to be changed to a triplet of the same value. This is often useful if the final note of the trill would otherwise fall on the same pitch as the following note.

Note that, in this case, a setting of "bi80outxf" (requesting that the second last note be flipped to the opposite auxiliary) is meaningless because it isn't an auxiliary note, but the Principal.

should probably be prohibited because a flip is intended to switch between auxiliary notes and the second last note is actually the principal. Furthermore, this particular execution of a trill would probably not be musically desirable because the total time occupied by the auxiliary notes is far longer than the total time devoted to the principal[24].

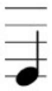
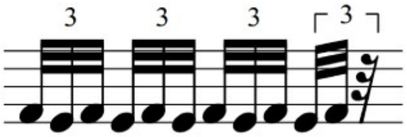
Although it is possible for software to detect either of these conditions, the objective of avoiding any effort to regulate taste suggests that no attempts be made in this regard, at least for the moment.

bi83oun (in 4/4 time)

Principal	Multiplier	Trill notes to be generated	Generated Result:	Comments
 $\frac{1}{4}$ 4/4 time	x 8 =	$\frac{8}{32}$ – that is eight 32nd notes, or four pair starting with the upper auxiliary note "f".	 Four thirty-second note triplets, equivalent to eight thirty-second notes.	If the settings were bi83ounxf, the second last note of the trill would be the lower auxiliary note "d" rather than the upper auxiliary note "f#" that is shown.

In this example, the "Trill note ratio: 3 use equivalent triplets for note pairs" option has been selected. Any implementation of the trill generation approach described in this proposal should therefore ignore (or even prevent) selecting "Trill ends with: Triplet", since the last group of notes is already a triplet. This can be viewed as replacing the four pair of notes called for with four triplets.

bi83ounxr (in 4/4 time)

Principal	Multiplier	Trill notes to be generated	Generated Result:	Comments
 $\frac{1}{4}$ 4/4 time	x 8 =	$\frac{8}{32}$ – that is eight 32nd notes, or four pair starting with the upper auxiliary note "f".	 Four thirty-second note triplets, equivalent to eight thirty-second notes.	

This example is equivalent to the previous example, with the addition of the "Trill ends with: eXtra mods: Rest" setting, which will cause the final note of the generated trill to be replaced by an equivalent rest. This option would be a questionable choice in this instance, since the length of time spent on the auxiliary notes would be greater than that spent on the Principal notes – a subtle alteration of the significance of the Principal (see footnote 24).

Where the note following the trill happens to be the same pitch as the Principal of the trill, however, this execution of a trill can be used to provide a separation (somewhat the opposite of a slur), which alters the apparent start timing of the following note (i.e. making it seem to begin before the beat).

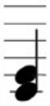
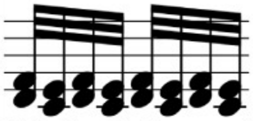
For the parallel opening notes of the Scarlatti D Major Sonata shown on page 4, it is generally accepted that the "d" played with the right hand should also be executed with a trill that is identical to that used in the right hand, and it is often marked with a courtesy trill by publishers.

[24] … which renders the term "principal" rather meaningless. If the intent was for this particular note to be sounded mostly as an "f" rather than an "e," the composer would probably have done that in the first place.

Playing such a trill doesn't present any insurmountable difficulties, but some composers are far more sadistic in their treatment of struggling pianists, calling for single-handed double trills.

For the next example, which is simply a benign form of such digital sadism, assume that there is a chord (in this case as "c#" and "e") marked with a trill. For keyboard music, this sort of trill can only be played by normal humans with certain key combinations and, therefore, only in certain Keys. But there were pianist-composers who thought nothing of writing and playing such things[25].

bi40oun (in 6/8 time) – (trills in thirds?)

Principal	Multiplier	Trill notes to be generated	Generated Result:	Comments
 $\frac{2}{8}$ 6/8 Time	x 4 =	$\frac{8}{32}$ – that is eight 32nd notes, or four pair starting with the upper auxiliary note "f".	 Eight (4 pair) thirty-second notes	

It may not be necessary (or even desirable) to add such a capability to MuseScore for a prototype if this project is undertaken, but such possibilities should be considered when making any implementation decisions. Although my focus is obviously on keyboard scores, many orchestral arrangements could make good use of the ability to handle chord trills.

Broken trills

Obviously, the automated generation of trills described above can be considered incomplete when compared to the variety of trills available in the real world. Should one need to enter a broken trill, as in the lower example of interpreting the trilled "f" to the right[26], it would need to be done manually until more options are added to the extended portion of the dialog box. Nevertheless, my expectation is that the limited capability described here would add a great deal of flexibility and potential for further refinement to what is already a terrific piece of software.

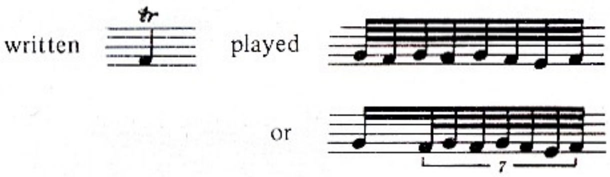


Figure 11: Example of a broken trill

Gotchas and Future Refinements

It may or may not be apparent that, with the generating capabilities described so far, the transcriber or composer would have the capability to create and play some truly appalling and tasteless trills. It seems to me, however, that this is of little concern when one considers that MuseScore itself permits the creation and playback of some truly appalling and tasteless compositions. "De gustibus non disputandum," as the Roman Horace once opined. The aim of this exercise is not, as I've suggested, to arbitrate trill styles, but simply to make the overall usability of MuseScore better.

[25] One thinks of nineteenth century artists such as Liszt, Thalberg, and Busoni, but this is sometimes seen as well in music composed by twentieth century virtuosi like Godowski and Rachmaninoff.

[26] The editor Maurice Henson's notes in Volume 11 of "Scarlatti – Sonatas for the Keyboard"; ISBN 038081-03080-7.

Having said that, however, it may or may not be appropriate for a system of "warnings" to be established at some later stage of development; these could, of course, be turned on (for students) or off (for typical users. Some thoughts for warnings might include: ??????

- terminated trill requires at least six notes
- note values of trills must be shorter than that of following note
- it might be useful to have a reset function that re-initializes some or all of the shadow voices.
- If a score indicates a desired trill completion [example] load that into the editor as a protected note...

Add a "Courtesy Trill" symbol to MuseScore, not only because it is a standard symbol that MuseScore doesn't currently offer, but this could serve as a means of indicating to MuseScore that any trill defined for the main Trill symbol should be duplicated in the Shadow Voice marked with the courtesy trill symbol.

Conclusion

I'm not going to attempt writing any code to implement any of the suggestions in this proposal. I'm retired and my programming days are too far in the past for me to have any expectations that I would be able to match the caliber of what a practicing coder would produce (although the lack of usability experience evident in those who have chosen to use red type on black backgrounds gives me pause). Nor do I make any promises that I've considered every aspect of the trill play issue.

I will be quite happy to participate in any testing of whatever someone might choose to put together, however. My main purpose was to take a first stab at thinking this through, and then documenting my thoughts in such a way that other users can chime in with their own opinions and suggestions. To that end I would encourage that others with an interest provide some feedback as to how useful any implementation of this proposal might be.

Forum user skaufman made the following post back on October 30, 2011 at 8:34am:

> "Yes, trills and grace notes should play. Don't make it too 'smart' by assuming a scale; just let the user decide on the side note including accidentals if needed, which note to start on, and 'how fast,' i.e. the trill note values. As an alternative method, I believe Noteworthy Composer had a 'layer' feature so you could write them out explicitly but put them in a hidden layer. In that case, the hidden layer was played and the displayed layer muted, which is troublesome as you then have to create both layers throughout the piece."

On March 8, 2011 at 3:36pm, Marc Sabatella said:

> "It should just play both voices normally. Most likely, you'd want it not to play the notes that are for display only, so you could set their velocity to 0 (right click, Note Properties)."

Hopefully, this proposal will meet and exceed both their expectations although, to be honest, it's really proposed just so I can enter all those darn Scarlatti trills much more easily. ☺

Thanks for listening. – Frank