laboratories dak
scientific and technical undertakings

3102 Thunder Road
Alamogordo, New Mexico 88310
Douglas A. Kerr, P.E. (Ret.)
Principal
575-921-6795
doug@dougkerr.net

TECHNICAL REPORT

Subject: MusicXML—the duration element

Issue: 4 Reference: MusicXML_duration*.doc

Date: May 12, 2021

Author: Douglas A. Kerr, P.E. (Ret.)

To: File

ABSTRACT AND INTRODUCTION

MusicXML is a language for transporting musical scores between musical notation programs. It describes the score of interest in terms of its graphical notational symbols and their arrangement on the page. It also conveys further details of how the score is to be "played".

An important (albeit optional) encoding of the MusicXML element for a note is the element <duration>. The MusicXML documentation does not clearly explain the intended meaning of <duration>. As a consequence, different notation program developers have taken different and incompatible views as to how the value of that element should be interpreted when reconstructing the score. As a result the universal interchange of scores via MusicXML is severely compromised.

This report describes this situation, and suggests an "understanding" of the meaning of the element <duration>.

1 ADMINISTRATIVE

1.1 Report not sponsored

This report, and the research underlying it, was not sponsored by, nor done at the behest of, any external organization.

1.2 Distribution

This report is not "published", but its distribution is not limited.

1.3 Disclaimer

The author has used his best professional skill and available information in the preparation of this report, but makes no representation that it is accurate or useful for any purpose. The reader who relies upon this report does so at his own risk, and the author cannot be responsible for any result not deemed satisfactory.

2 READER BACKGROUND

It is hoped that the reader is generally familiar with musical notation, with the concept of music notation programs, and with the basics of the MusicXML language.

But, considerable background pertinent to this topic is given in the companion technical report, "MusicXML—background", by the same author. It is probably available where you got this. I commend it to the reader of this report.

3 THE MusicXML LANGUAGE

3.1 Introduction

The MusicXML language was developed to provide a standard language for the transport of Mu8iscal scores between, most commonly, music notation programs, given that programs from different publishers typically use, natively, parochial formats for the representation of a score.

3.2 The "MusicXML documentation

What I will refer to here as the "MusicXML documentation" comprises in the main three components:

- The syntactic specification. This is done in two form, each using a specialized language. It describes the various syntactic ingredients of the MusicXML language and how they may and must appear in the MusicXML file.
- The syntactic description. This is in textual form, and illuminates in a more accessible form the implications of the syntactic specification.
- The tutorial. This describes, with the use of sample score excerpts and the corresponding MusicXML code excerpts, the actual use of the MusicXML syntax for many common score situations.

3.3 Critique

The MusicXML documentation describes in exquisite detail and precision the lower-level structure of the MusicXML syntax.

But when we get to the higher syntactic level—perhaps what we could think of as the "grammar" of the language—the document often leaves a number of uncertainties. The result is that implementers of MusicXML in, for example,

notation programs, must come to their own conclusions as to the fuller meaning of various parts of the language. The result, of course, is that the transport of a score from one notation program to another, via MusicXML is often significantly imperfect in one matter or another. One of those matters is the use of the MusicXML element <duration>.

4 TECHNICAL DEFINITIONS

4.1 Time

We will be concerned with two kinds of "time":

1. *Musical time* is abstract, and quantifies the "flow" of the musical structure of the notation itself. It is typically denominated in measures, beats, and in some cases, fractions of a beat.
2. *Clock time* is the familiar "time", and may be denominated in hours, minutes, seconds, and in some cases, fractions of a second.

4.2 About "duration"

The word "duration" can in general have two meanings. One is the length of time of an event (which implies noting about when it begins). ("The duration of the concert was two hours.") The other is the span of time of an event beginning at a certain instant. ("The duration of the concert was from 4:00 pm to 6:00 pm.") In technical writing, the latter is often more precisely spoken of as an *era*. But here, where there is little chance of confusion between the two, I will call both "duration".

4.3 Three durations of a note

4.3.1 *The notational duration of a note*

In conventional musical notation, we have different symbols representing, for example, half notes, quarter notes, eighth notes, and such. These differ in the length of *musical time* they normally occupy. In many musical texts, the property that differs between these different kinds of note is called their *time value*. But it is quite common, and reasonable, for the property to be spoken of as their "duration". However, in this report we will encounter other meanings of the term *duration*. To avoid confusion, and for consistency with other terms we will encounter, here I will generally speak of the *time value* of a note as its *notational duration*.

4.3.2 *The musical duration of a note*

Each note (or rest) occupies a certain amount of musical time, the *musical duration* of the note. It is essentially the duration of the "realm" of the note. Perhaps its most important role is as the amount of musical time before the "realm" of the next

note begins. These realms of consecutive notes (or rests) are by definition contiguous.

In the "normal" situation, the *musical duration* of a note (or a rest, actually) is the same as its *notational duration*. But later we will encounter some situations where that it not so; thus I have separate names for them.

Note that this term is not found in the MusicXML documentation. But we will encounter there a term that we will consider equivalent to *musical duration*.

4.3.3 *The performance duration of the note*

When we play the score, each note sounds for a clock time that corresponds, given the tempo of play, to a certain amount of musical time, commonly called the *play duration* of the note. But for consistency with certain terminology in the MusicXML documentation, here I will use the term *performance duration*.

In the simplest situation, the *performance duration* corresponds to the *musical duration* of the note. But the scorist might want the *performance duration* to be less than the *musical duration*. The usual reason is to avoid the play being "fully legato" when that is not the musical style that is desired.

The *performance duration* of notes is often spoken of in terms of "% of face value", where "face value" alludes to the *notational duration* of the note.

5 ELEMENTS AND ATTRIBUTES

5.1 Introduction

The MusicXML language is a form of the XML language, and its details conform to the basic concepts and structures of XML.

5.2 Elements and attributes

The basic ingredient in an XML file (and thus of a MusicXML file) is the *element*. Elements are identified by XML "tags", which bracket the element content. The "opening" form of such a tag, in the classical formation, is: <box>, the "ending" form of such a tag, again in the classical formation, is: </box>.

The properties of an object described by an element may be given in two different ways. One is by subordinate ("child") elements: The other is by attributes of the element. The rationale for when a property should be defined one way and when the other is mysterious.

5.3 Typographical conventions

In the text of the reports I will give the names of elements as if we were seeing their opening tag, thus: "The element <box>"; I will give the names of attributes underlined: "The attribute color."

6 CONTROL OF THE PERFORMANCE DURATION

Most modern notation programs allow the scorist various modes of control over the *performance duration* of the notes. In some programs, the scorist can set a default duration (usually as a fraction of the *notational time*) which deposited notes will initially be given.

Often there is a graphical display on which the *performance durations* of the individual notes can be seen as bars on a musical time scale. Typically the play durations of one or more notes can be changed here, either by numerical input or by "dragging" the end of the bar. In many cases, the data describing the *performance durations* of the notes is spoken of as "MIDI data", since it is in effect the precursor of the MIDI data that will be sent to a sound module during actual play.

In many cases, the scorist can not only control the *performance duration* of the notes but as well their *performance start time*. For example, for certain notes, the scorist may arrange for the "sounding" in play to commence not at the beginning of the "realm" of the note but perhaps what amounts to the time of a 32nd note "late", this being done to attain a certain musical style effect.

7 TWO PARTS OF A MusicXML FILE

The MusicXML documentation states that there are two parts of a MusicXML file:

1. The Notation part (which des not have that as an actual name, but I will call it by that "name"). This describes the actual visible notation.
2. The "MIDI-compatible part" (its actual formal name). This describes how the score should sound if played. We assume that its name comes from the fact that what it conveys is, in a sense, a script for the generation of a MIDI stream, by way of which a sound module will be caused to perform the notes.

Does the MIDI-compatible part include provisions for representing, at the pleasure of the composer's notation program, the *performance duration* (and offset of the *performance start time*) of a note? Yes. But is not fully clear just how that is supposed to work.

8 The unit

The musical time unit which applies to the element <duration> and other creatures we will meet shortly, is the *division*, which is a certain fraction of the ideal *musical duration* of a quarter note. That fraction is declared in a MusicXML file by the element <divisions>, which is usually placed at the beginning of a part (it could vary from part-to-part)[1]. If we have:

```
<divisions>24</divisions>
```

[1] In fact it can change in the middle of a part.

then one *division* is 1/24 of the ideal *musical duration* of a quarter note.

What determines the value of <divisions> used in a certain MusicXML file? For now we can think of it as essentially arbitrary. We will hear more about that in a little while.

9 THE MusicXML ELEMENT <duration>

9.1 Introduction

The author has made an extensive analysis of the relevant passages of the MusicXML Documentation. This was assisted by various statements made in various forums by Michael Good, the original developer of the MusicXML language and its current *de facto* "keeper".

What follows to some extent reconstructs that analysis and its conclusions.

9.2 In the syntax

In a properly-formed MusicXML file, each <note> element contains (among many other things) one and only one <duration> element.

9.3 Definition

The MusicXML documentation defines the <duration> element thus (in part):

> The duration element is an integer that represents a note's duration. This is the intended duration vs. notated duration.

9.4 Meaning

It at first seems that "intended duration" means "the duration for which we expect the note to sound" (our *play duration*), recognizing that this may in fact not be the same as the *notational duration* of the note.

The fact that, in the MusicXML tutorial, <duration> is presented as a creature of the MIDI-compatible part of MusicXML, rather than the Notation part, lends further credibility to the interpretation above.

In fact, a more accurate definition is that <duration> defines the *musical duration* of the notes. The main significance of that is that it tells where the next note starts, in musical time terms. Insofar as the graphic score is concerned, that is an abstract construct. It only takes on concrete significance when the score is played (by a human or by the program through a sound module).

But we will see shortly why the *musical duration* may in fact not be the same as the *performance duratio*n.

10 THE ELEMENT <type>

10.1 In the syntax

The element <type> is optional, and may appear only once in a <note> element.

10.2 Definition

The MusicXML documentation defines the element <type> thus (in part):

> Type indicates the graphic note type. . .

It does this as an enumerative text variable, whose value may be 'quarter', 'eighth', '16th', etc.

10.3 Significance

The value of <type> defines the notation symbol to be placed for the note.

10.4 Absent a <type> element

If for a note there is no <type> element (perfectly legitimate), the receiving program is expected to deduce the "time value" of the note, and thus the proper symbol for its notation, from the *musical duration* of the note, which came from the <duration> element.

10.5 Raison d'etre

The MusicXML documentation introduces the <type> element by observing that the receiving program should be able to deduce the intended note symbols (and thus, notational duration) from the value of <duration>.

The discussion continues to say that, however, the (optional) <note> element gives the receiving program an explicit clue in that regard (perhaps thus easing the work of the receiving program).

In fact, if there is no <type> element for a note, the receiving program has no choice but to deduce the intended *note symbol* (and thus, the *notational duration*) from the value of <duration>.

10.6 Two ways of stating the same thing?

So if there is a <type> element, what is then the significance of the <duration> element (which must also appear)? Is that just a different way to state the same thing that is stated by the <duration> element, and thus both elements should describe (in their own ways) the same duration? So it would seem. (But this is not clearly confirmed in the MusicXML documentation.)

10.7 What about performance duration?

Accepting that, when there is a <type> element, the <duration> element should describe a *musical duration* consistent with the symbol defined by <type>, and assuming that the *performance duration* should be the same as that, we certainly can't use <duration> to describe a *performance duration* that is different from the *notational duration*.

But, as mentioned earlier, we often arrange for that difference in our notation program, and probably want that conveyed to the receiving program when we transport the score via MusicXML. Have we talked ourselves out of being able to do that?

11 ENTER attack AND release

11.1 Introduction

That need is neatly accommodated by two optional *attributes* of the <note> element, attack and release. These are defined thus in the MusicXML documentation:

> The attack and release attributes[2] are used to alter the starting and stopping time of the note from when it would otherwise occur based on the flow of durations - information that is specific to a performance[3]. They are expressed in terms of divisions, either positive or negative.

Again here there is an ambiguity, the meaning of "duration". But by triangulating among many passages in the MusicXML documentation, I conclude that "durations" here must mean *musical durations*.

11.2 Application

As an example, assume that the value of <divisions> is 120, a half note is being encoded, and we wish the note in the reconstructed score to sound for "90% of face" (and, to be complete, we wish the sounding of the note in play to commence at the nominal start of the realm of the note).

We then make <type> 'half' and <duration> 240, both defining the same *notational duration*, and <duration> as well defining the *musical duration*. And we give the <note> element the attribute release with value -24. This causes the sounding of the note to cease 24/120 of the nominal musical duration of a quarter

---

[2] Do not confuse these *attributes* of an element with the MusicXML *element* <attributes>, which gives a number of properties of a part (spoken of as "attributes") in the form of child *elements*. The *element* <divisions> is in fact one such "attribute". Aargh!

[3] I suspect that what is meant here would better have been said, "information that is specific to performance" (not "**a** performance").

note (10% of the nominal musical duration of the half note) before the end of the *musical duration* of the half note.

Suppose, in a more subtle definition of the sounding of the note, we want the sounding of the note to commence "5% of face" late and extend until "10% of face" from the end of the *musical duration* of the note.. We again make <type> 'half' and <duration> 240 (both defining the same *notational duration*). and <duration> as well defining the *musical duration*. We give the <note> element the attribute attack with value 12, and the attribute release with value -24.

12 CHOICE OF A VALUE OF <divisions>

What determines the value of <divisions> in MusicXML file? It is essentially arbitrary. But, if in fact we recognize <divisions> as giving one of two definitions of the *notational duration* of the note, the value of <divisions> must be large enough that the notes to appear in the score can be properly represented. Thus, if the score involves 16th notes (either on their own or as a "dot" on an eighth note), then the value of <divisions> cannot be less than 4 (a 16th note has 1/4 the notational duration of a quarter note).

In fact, a passage of the MusicXML documentation suggests that the value of <divisions> be made no larger than is required as discussed just above.

This seems to rule out the thought that <duration> could be used to represent the *musical duration* of a note (and thus, basically, its *performance duration*) when the *performance duration* is different from the *notational duration*. This is in fact consistent with the conclusion I have come to above.

Some notation programs uniformly use a value of "divisions" that matches the units used internally for musical time matters. In Overture, for example, the value of <divisions< is uniformly 480. That is well suited to using <duration> to represent *play duration* to a fine resolution. Except that I don't think it should be used for that.

13 AN ALTERNATE VIEW

In the discussion above, I adopt the conclusion that the likely intent of the MusicXML specification (adjusted for some modern pragmatic considerations) is that <duration> states the *musical duration* of the note (most often duplicating what is given, in a different form, by <type>).

But it is understandable that some developers have concluded that the purpose of <duration> is to give the *performance duration* of a note, which might well be different from its *musical duration*.

This of course leads to the prospect of error when transporting a score between notation programs whose logic is predicated on these two disparate views.

14 THE CURIOUS MATTER OF *NOTES INÉGALES*

14.1 Introduction

An interesting complication in seeking to interpret the <MusicXML documentation to discern the intent of the <duration> element is the matter of the *notes inégales* construction. The term is French, and means "unequal notes". It is explicitly mentioned in the MusicXML tutorial.

The most common usage today of this principle is in the "swing eighths[4]" construction, widely used in jazz and other genres. There, in a series of eighth notes, the notes are considered in consecutive pairs. For each pair, in performance, the first note is sounded for greater than its nominal duration, and the second note for less than its nominal duration, the sum of the two *performance durations* nominally matching the sum of the notes' *notational durations*.

14.2 Encoding in MusicXML

14.2.1 *An obvious way*

How might we encode into MusicXML such a structure? There is a very straightforward way, using tools we heard of above. For the first note of the pair, we apply the attribute release to delay the ending of play, increasing the *performance duration* of that note. For the second note of the pair, we apply the attribute attack to delay the *performance start time* of that note to match, decreasing its *performance duration*.

In addition, we may actually also adjust the value of release for the first note, and use a release for the second note, to avoid the "full legato" effect.)

14.2.2 *Another way*

But the MusicXML tutorial suggests a different approach, this involving adjusting the values of <duration> for the two notes. It presumes that in this construct, the value of <duration>, defines the *musical duration* of the note, which here may differ from its *notational duration*.

Then, in the MusicXML encoding, we would give the first note a value of <duration> that is greater than equivalent to the *notational duration* of the note, and the second note a value of <duration> less than equivalent to the *notational duration* of that note, the two values of <duration> adding to the sum of the *notational durations* of the two notes.

Michael Good, the original developer of the MusicXML language and today its *de facto* "keeper", in a forum devoted to MusicXML development, has suggested that this approach to the encoding of *notes inégales*, intimated by the MusicXML

[4] Sometimes called "swung eighths"

tutorial, is essentially obsolete, implying that the technique described in section 14.2.1 is used instead.

14.2.3 *Probably the best way*

Harking to that, we can use the technique described in section 14.2.1 to deal with the swing eighth structure (or even other more esoteric *notes inégales* structures).

And then we are free to adopt the outlook, suggested earlier, that <duration> and <type>, when both are present, should represent equivalent amounts of musical time.

15 WHAT DO CURRENT NOTATION PROGRAMS DO?

15.1 Introduction

It is instructive to see what current notation programs do in this area. My most detailed investigations pertain to the programs Overture, MuseScore, and Finale. My findings are synopsized here.

In all cases in this section I will discuss the situation in which the <note> element contains both <type> and <duration> elements (almost universally the case for MusicXML code generated by modern notation programs).

15.2 Overture

15.2.1 *Version*

This was observed with Overture version 5.6.3-3.

15.2.2 *Receiving a MusicXML file*

a. Overture relies on the <type> element to select the note symbol and to establish its *notational duration*.
b. Overture treats <duration> as defining the *performance duration* of the note.
c. The attributes attack and release are not recognized.

15.2.3 *Generating a MusicXML file*

a. Overture makes the element <type> reflect the *notational duration* of the note.
b. It makes the element <duration> reflect the *performance duration* of the note. Departure of the play start time from the nominal is not reflected.
c. The attributes attack and release are not used.

15.3 MuseScore

15.3.1 *Version*

This was observed with MuseScore version 3.5.2.

15.3.2 *Receiving a MusicXML file*

a. MuseScore relies on the <type> element to select the note symbol and to establish its *notational duration*.

b. It uses the value of <duration> to establish the *musical duration* of the note. (This among other things implies the musical time at which the following note commences its realm.)

c. On play, the sounding of the note commences at the start of the musical time realm of the note. The *performance duration* is equal to the notational time of the note ("100% of face").

d. The attributes attack and release are not recognized.

e. The total of the <duration> values is taken my MuseScore to be the intended musical time length of the measure. If that does not match the length given by the time signature, MuseScore will give an error message upon loading of the MusicXML file. If we proceed anyway, the notation structure for the measure may be "strange", perhaps involving a gratuitous rest to make up for the seeming discrepancy.

Here, a sensible result will only be attained if the values of <duration> match the notational durations implied by the values of <type>.

15.3.3 *Generating a MusicXML file*

The <type> element comes from the note symbol. The element <duration> has a value corresponding to the notational time implied by the <type> elements. Departures of the established *play duration* or *play start time* from the nominal are not reflected in the encoding. The attributes attack and release are not used.

15.4 Transport of scores from Overture to MuseScore via MusicXML

Transport of score from Overture to MuseScore via MusicXML is at this time of special interest to the author and a colleague. As to the matter of note durations, a sensible result in MuseScore will only result if, before generating the MusicXML file from the score in Overture, all note performance durations must be set to "100% of face" and any offsets of play start times expunged.

Of course in some cases these timing subtleties were achieved either by initially recording the work as performed by a human player on a MIDI-connected keyboard, or were tediously crafted by the scorist. In either case, when this score is carried across the River Styx in MusicXML form, all those subtleties must be jettisoned.

15.5 Finale

15.5.1 *Version*

This was observed with Finale version 26.3.1

15.5.2 *Special role of Finale?*

We note that, especially given that Michael Good, the original developer of the MusicXML language, and still its *de facto* principal "keeper", is heavily involved in the development of Finale, and that in fact Finale first got its MusicXML capability by way of the plug-in "Dolet", provided by Recordare, the firm within which the MusicXML language was originally developed and promoted, we trend to think of Finale as the arbiter of taste in matters MusicXML.

15.5.3 *Receiving a MusicXML file*

a. Finale creates the note symbol based on the value of <type>, and makes the *musical duration* to match.

b. If there are no attack and/or release attributes for a note, the *play duration* of the note will be its notational duration.

c. The values of the <duration> elements for the note of a measure are added up, and Finale treats the sum as the length (in musical time terms) of the measure. After what is deemed to be the length of the measure, play of the first note of the next measure commences forthwith. If the play of the notes in the first measure has not been completed, it continues apace as well.

   This behavior of itself is truly bizarre.[5] But it would be moot for a MusicXML file in which, for each note, the value of <duration> matches the *notational duration* implied by the value of <type> (in accordance with the interpretation of the MusicXML specification I recommend).

d. Finale responds appropriately to the attack and release attributes. It is most useful to speak of this behavior in a case where the value of <duration> corresponds to the notational time implied by <type>. Then, if there is an attack attribute, the start of note sounding in play is shifted the time it would otherwise have by the value of attack. If there is a release attribute, the end of the note sounding in play is shifted from when it would otherwise be by the value of release.

15.5.4 *Generating a MusicXML file*

a. The element <type> states the symbol to be used for the note.

b. The element <duration> has a value corresponding to the notational time implied by the <type> attribute.

---

[5] It is somewhat parallel to the behavior of MuseScore version 3.5.2.

c. Departures of the established *play duration* or *play start time* from the nominal are not reflected in the encoding.

d. The attributes attack and release are not used.

16 WITH NO <type> ELEMENT

16.1 Introduction

I suspect it is today rare for a notation program to generate a MusicXML file ion which the <note> elements were not provided with a <type> element. But that possibility, perfectly legitimate according to the MusicXML documentation, plays a role in our effort to divine the overall intended scheme regarding note durations. Accordingly, I have made tests of the response to such a file of several notation programs.

16.2 Overture

Tests were also made with Overture (version 5.6.3-3) of a MusicXML file in which the note elements did not contain the optional <type> element.

a. Overture sets the note symbol (and thus the *notational duration*) based on the value of <duration>, quantized to an increment of a 64th note.

b. If the value of <duration> exactly corresponds to the *notational duration* of the chosen note type (no "rounding" was needed), Overture makes the *musical duration* equal to the notational duration. It makes the *play duration* of the note equal to 90% of the *notational duration* of that note. (This is presumably Overture's default default value to avoid a "fully legato" rendering.)

c. If the value of <duration> does not exactly correspond to the *notational duration* of the chosen note type ("rounding" was involved), Overture makes the *musical duration* equal to the *notational duration*. It makes the *play duration* of the note equal to the *notational duration* of that note.

Perhaps as another part of that "plan", when creating a MusicXML file, Overture makes the value of <duration> one less than the value that would correspond to the notational time of the note as given by the <type> element, which is always included.

16.3 MuseScore

Tests were made with MuseScore (version 3.5.2) of a MusicXML file in which the note elements did not contain the optional <type> element. The <duration> element values corresponded exactly to the notational durations of notes quarter, eighth, 16th, etc., down to 128th.

a. MuseScore makes the note symbol 'quarter' (that also dictates the *notational duration*) regardless of the value of <duration>.

b. The *musical duration* is made according to the value of <duration>

c. The *play duration* is made the same as the value of *notational duration*.

Suffice it to say that this can create a rather bizarre result on both notation and play fronts. The *play duration* of the last note(s) in a measure can well extend into the realm of the next measure.

16.4 Finale

Tests were made with Finale (version 26.3.1) of a MusicXML file in which the note elements did not contain the optional <type> element. The <duration> element values corresponded exactly to the notational durations of notes quarter, eighth, 16th, etc., down to 128th.

a. MuseScore makes the note symbol in accordance with the value of <duration>, seemingly to a resolution of a 2048th note.

b. The *musical durations* are made to essentially match the *notational duration*.

c. The *play durations* are made slightly less than the *musical durations*.

17 INTEROPERATION

Suffice it to say that this "diversity" (to be polite) in dealing with the element <duration> in MusicXML among various notation programs does not bode well for the successful interchange between different notation programs of scores that include *play durations* or *play start times* departing from the nominal.

18 MY RECOMMENDATION

I recommend the following doctrine be adopted in MusicXML-enabled application programs and the like.

18.1 Encoding in the MusicXML file

a. The *notational duration* ("time value") of the note should be encoded in the <type> element.

b. The <duration> element should be given a value consistent with the *musical duration* implications of the <type> element.

c. If the established play duration or start time differ from the nominal, that should be encoded using the attack and/or release attributes, as needed.

18.2 Response to the MusicXML encoding for a note

a. If there is a <note> element:

1. Make the note symbol to match
2. Give the note a *musical duration* with the time value of the note symbol.
3. Ignore the <duration> element.

b. If there is no <note> element:

1. Make the note symbol (and thus the *notational duration*) that implied by the value of the <duration> element.

2. Give the note a *musical duration* (allotment of musical time) consistent with the *notational duration* of the note symbol.

c. in either case:

1. Make the base *play duration* the same as the note's *musical duration*.

2. If there is an attack attribute and/or a release attribute, shift the *play start time* and/or *play end time* (from those of the base *play duration*) according to the values of those attributes, respectively.

19 ISSUE RECORD

Issue 4, May 12, 2021 (this issue). Editorial changes.

Issue 3, January 30, 2021. Editorial changes.

Issue 2, January 4, 2021. Editorial changes.

Issue 1, December 22, 2020. Initial issue.

-#-