The GUIDO Music Notation Format
Version 1.0
Specification Part 1: Basic GUIDO

Technical Report TI 20/97
Fachbereich Informatik
Technische Universität Darmstadt
Darmstadt, Germany

Holger H. Hoos
Fachbereich Informatik
Technische Universität Darmstadt
Alexanderstr. 10
D-64283 Darmstadt, Germany
Email: hoos@informatik.th-darmstadt.de

Keith A. Hamel
School of Music
University of British Columbia
6361 Memorial Road
Vancouver, BC, V6T 1Z2, Canada
Email: hamel@unixg.ubc.ca

Contents

1 Introduction

The GUIDO Notation Format is a general purpose formal language for representing score level music in a platform independent, plain-text and human-readable way. It is based on a conceptually simple but powerful formalism; its design concentrates on general musical concepts (as opposed to notational, i.e, graphical features). A key feature of the GUIDO design is adequacy, which means that simple musical concepts should be represented in a simple way and only complex notions should require complex representations.

In the present version, GUIDO supports most of the commonly used concepts of conventional music notation; however, it is designed to be a flexible and easily extendable open standard. Various extensions of the Basic GUIDO Notation Format include mechanisms for representing musical structure, abstract scores, micro-tonal music, and generalized timed events (for multi-media support). These extensions will be specified in subsequent parts of this report.

GUIDO has not been developed with a particular application in mind but to provide an adequate representation formalism for score-level music over a broad range of applications. The intended application areas include notation software, compositional and analytical systems and tools, large musical databases, performance systems, and music on the World Wide Web.

The GUIDO Notation Format has been developed by Holger H. Hoos, Technical University of Darmstadt, Germany and Keith Hamel, University of British Columbia in Vancouver, Canada; some features are based on discussions with Michael Fuhlbrügge, Kai Flade, Thomas Helbich and Jürgen Kilian. It is named after Guido d'Arezzo (990-1050), a renowned music theorist of his time and important contributor to today's conventional musical notation (CMN). His achievements include the perfection of the staff system for music notation and the invention of solmisation (solfege).

The GUIDO Notation Format is publicly and freely available (see also the GUIDO Notation Format Homepage at `http://www.informatik.th-darmstadt.de/AFS/CM/GUIDO` on the World Wide Web); it can and should be freely used for arbitrary scientific, educational or commercial purposes. However, the GUIDO Notation Format specification remains the intellectual property of the authors of this document. It is our wish that every publicly available software supporting GUIDO and every extension of GUIDO as specified in this document should be brought to our attention.

2 Basic GUIDO Notation

Basic GUIDO Notation covers the representation of notes, rests, accidentals, single- and multi-voiced music and the most common concepts from conventional music notation such as clefs, meter, key, slurs, ties, beaming, stem directions, etc.

2.1 Notes and Rests

The general syntax for representing notes is

notename accidentals octave duration

the corresponding represention for rests is

’_’ *duration*

Both notes and rests are instances of GUIDO events; in particular, rests are treated as special notes, which have a duration but neither accidentals nor an octave value.

Octave and duration can be omitted; if so, they are inferred from preceeding notes (or rests) within the same sequence or chord (for details, see below) or assumed to have standard values (see below) when required (such as for musical playback or notation).

2.1.1 Notenames

There are different systems of notenames:

1. c d e f g a h/b *(diatonic)*
2. c cis d dis e f fis g gis a ais h/b *(chromatic)*
3. do re me fa sol la si/ti *(solfege)*

Remarks:

- Arbitrarily mixing these systems is possible, but discouraged.
- *h* and *b* denote the same pitch-class (*b* being used in the international system, *h* in the german system).
- *si* and *ti* denote the same pitch-class (*si* being used in the french and italian, *ti* in the english solfege system).

2.1.2 Accidentals

accidentals is an arbitrary sequence of the symbols '#' and '&' (for sharp and flat); in particular, '##' is a double sharp, '&&' a double flat.

Remarks:

- Accidentals cannot be used with rests.
- Accidentals are valid only for the note in the description of which the appear.
- The usage of arbitrary sequences of accidentals without a reasonable interpretation in CMN (such as #&#) is discouraged.
- Chromatic pitch-classes are assumed to be different from alterated diatonic ones, i.e., *c# != cis*.

2.1.3 Octave

octave is an integer number indicating the octave of the note, where *a1* is the 440Hz *a*. All octaves start with the pitch-class *c*, so *c1* is the *c* just below *a1.*

Remarks:

- The usage of octaves beyond the range of -3..+5 is discouraged.
- If _octave_ is omitted from a note description, it is assumed to be identical to the last octave specified before in the current sequence.
- _octave_ should always be specified at the beginning of each sequence; if this is not done, but an octave is required in the context of an application, a standard value of +1 is to be assumed.

2.1.4 Duration

duration is specified in one of the forms

1. _’*’enum’/’denom dotting_
2. _’*’enum dotting_
3. _’/’denom dotting_

where _enum_ and _denom_ are positive integers and _dotting_is either empty, ’.’, or ’..’; if no dots are present, the relative duration of the notes is assumed to be

- _enum/denom_ beats in case (1);
- _enum_/1 beats in case (2);
- 1/_denom_ in case (3).

Succeeding the duration by

- one dot ’.’ is equivalent to multiplying it by 3/2
- two dots ’..’ is equivalent to multiplying it by 7/4

Remarks:

- Triplets are represented as *1/3, *1/6, ...; quintuplets, septuplets and so on are handled analogously.
- Durations which are specified by fractions that are equal, such as *2/4 and *1/2, are considered to be different.
- Although dotted durations have equivalent dot-less durations, these are considered to be different (*1/4. is not identical to *3/8).
- If _duration_ is omitted from a note description, it is assumed to be identical to the last duration specified before in the current sequence.
- _duration_ should always be specified at the beginning of each sequence; if this is not done, but a duration is required in the context of an application, a standard value of 1/4 is to be assumed.

General remarks:

- No white space (blank, newline or tab) should be used within note descriptions.

Examples:

```
d1*3/4
c#-1/8
h&/6
_*2
c2&&/2.
cis/4..
fa1##
sol&0
```

2.2 Tags and Tagged Ranges

Tags are used to represent additional musical information, such as slurs, clefs, keys, etc. A tag has one of the forms

1. '\'*id*
2. '\'*id*'<'*param-list*'>'
3. '\'*id*'(' *note-series* ')'
4. '\'*id*'<'*param-list*'>' '(' *note-series* ')'

where

- *id* is one of a number of predefined tag-names (see Section 2.7 below),
- *arg-list* is a list of string or numerical arguments, separated by commas (','),
- *note-series* is a series of note descriptions (as defined above) or chord descriptions (see Section 2.4) and tags separated by white space (spaces, newlines, and/or tabs).

Remarks:

- In general, tag-structures can be arbitrarily nested, since the *note-series* in (3) and (4) may contain other tags.
- The tags which are defined in Basic GUIDO are described in detail in Section 2.7 (see below).
- Tags of the from (1) and (2) are used for elements of CMN like clefs, meter, key, etc.
- Tags of the from (3) and (4) are called tagged ranges, they define properties of series of notes such as slurs, beams, etc.
- Generally, tags have no duration; however, tagged ranges can be thought of extending over the duration of the note series they are associated with.

Examples:

```
\bar  % barline
\clef<"g2"> % treble clef
\slur(c1/4 d e) % slurred group
\label<"motive A">(g1/2 f#/4 g g#/2)  % named sequence
```

2.3 Note Sequences

A note sequence is of the form

> *'[' tagged-notes ']'*

where *tagged-notes* is a series of notes, tags, and tagged ranges separated by white space. Note sequences represent single-voiced scores.

Remarks:

- If duration or octave values are omitted for notes within a sequence, they are assumed to be equal to the last value specified in the current sequence.
- Duration and octave values should be specified at the beginning of each sequence; if this is not done, they are assigned standard default values (see above) when this is required in the context of an application (such as playback or notation software).

Examples:

```
[ a1*1 ]      % single note
[ c1/4 d e f g a h c2/2 ] % C major scale

% short motive with meter, barlines and a slur:
[\meter<"4/4"> f0/4 \bar c1/2 \slur(e&/4 c) \bar g/2]
```

2.4 Chords and Chord Sequences

A chord is denoted by

> *'{' note-list '}'*

where *note-list* is a list of (possibly tagged) notes separated by commas.

Remarks:

- All notes of a chord should have the same duration.
- If this is not the case, the duration of the chord is the duration of the longest note within the chord; all other notes are interpreted as being succeeded by rests which compensate for the difference between their duration and that of the chord.

- The notes within a chords do not have to be ordered with respect to their pitch; however, it is suggested to consistently order chords in either an ascending or descending fashion.
- If for some note of a chord, duration or octave values are omitted, these are assumed to be equal to the last respective value specified in the current sequence; this applies also to the case where this value was specified for a preceeding note of the same chord.

Examples:

```
{c1/4., e&1/4., g1/4.}  % c minor triad , ascending order
{c1/4.,e&,g}      % same as above, abbreviated notation
{c#1/4,e#,g#}     % C-sharp major triad, ascending order
{c2, dis1, ais0, fis}   % four-voice chord, descending order
```

Chord Sequences

A chord sequence is a generalized note sequence, where instead of individual notes, chords may also appear.

Remarks:

- Chord sequences may contain chords as well as notes and rests.
- The chords within a sequence need not all have the same number of notes.

Examples:

```
[ {c1/4,e,g} {c,f,a} {d,f,h} {c1/2,g1,c2} ] % progression of triads
[ _/8 \slur(c1 d e) {fis/2, ais} e/2 ]    % mixed notes and chords
```

2.5 Note Segments

Note segments represent multi-voiced scores; they are denoted by

'{ ' *seq-list* '} '

where *seq-list* is a list of note sequences separated by commas. It is assumed that all voices start simultaneously. Normally, the sequences should have equal durations; if this is not the case, the duration of the segment is the duration of the longest voice within the segment; all other voices are interpreted as if they were succeeded by additional rests which compensate for the difference between their duration and that of the segment.

Remarks:

- The voices within a segment don't have to be ordered according to their pitch-range; however, it is recommended that voices be specified in either an ascending or a descending order with respect to their compass.

- Octave and duration values should be specified at the beginning of each voice; if this is not done, default values will be assumed when required in the context of a specific application. In particular, octave and duration values don't carry over from one voice to the next.

Examples:

```
 { [a0/4 h \slur(c1/8 h0) a/4],    % first voice
   [c1/4 d e/2],         % second voice
   [e1/8. \slur(f/16 e/8) d c h0 a/4] }  % third voice
```

2.6 Comments and Spacing

White space, i.e, any combination of blank-, newline-, or tab-characters, can be inserted everywhere except within note- or tag-names. The use of white space is discouraged in the following situations:

- within note specifications
- between tag-names and tag-argument-lists

White space is required in the following instances:

- between notes
- between tags
- between notes and tags and vice versa

Comments

Comments can be inserted everywhere where white space is allowed. A comment is preceeded by '%' and extends to the end of the line it started in. Thus, lines beginning with '%' are comment-lines.

Alternately, '(*' *comment* '*)' can be used for specifying comments. This form of comments can include multiple lines and may be nested.

Examples:

```
% a comment-line
a0/4 h \slur(c1/8 h0) % a/4 <-- the last note is commented out
[ c1/4   (* comment starts here ...
  d e f  <-- part of the comment!
  ... ends here *) d ]   % the sequence has only two notes!
```

2.7 Standard Tags

The following tags have standard interpretations in GUIDO Basic Notation:

`\intens, \i, \cresc, \dim, \crescBegin, \crescEnd, \dimBegin, \dimEnd, \tempo, \accel, \rit, \accelBegin, \accelEnd, \ritBegin, \ritEnd,\instr, \tie, \slur, \sl, \stacc, \accent, \ten, \marcato,\trill, \mord, \turn,\trem, \fermata, \grace, \cue, \repeatBegin, \repeatEnd, \clef, \meter, \key, \oct, \staff, \beam, \bm, \beamsAuto, \beamsOff, \stemsAuto, \stemsUp, \stemsDown,\bar, \doubleBar, \tactus,\text, \t, \title, \composer, \mark, \label.`

These standard tags and their usage are explained in the following sections.

2.7.1 Dynamic markings

\intens<*s*> indicates a dynamic marking *s*

\i<*s*> shorthand for \intens<*s*>

\intens<*s*,*r*> indicates a dynamic marking *s* and associates it with an absolute intensity value of *r*; *r* is a floating point number between 0.0 (minimal intensity) and 1.0 (maximal intensity) in standard (not scientific!) notation.

\i<*s*,*r*> shorthand for \intens<*s*,*r*>

\cresc(*notes***)** indicates an unspecific crescendo on *notes*.

\cresc<*s*>**(***notes***)** indicates a crescendo starting from the current dynamic marking to dynamic marking *s* on *notes*.

\cresc<*s*,*r*>**(***notes***)** indicates a crescendo starting from the current dynamic marking to dynamic marking *s* with associated absolute intensity value of *r* (see above) on *notes*.

\crescBegin indicates the beginning of a crescendo section which ends at the next \crescEnd tag; the crescendo starts from the current dynamic marking.

\crescEnd indicates the end of an unspecific crescendo section which began at the last \crescBegin tag.

\crescEnd<*s*> indicates the end of an crescendo section which began at the last \crescBegin tag; the crescendo ends with dynamic marking *s*.

\crescEnd<*s*,*r*> indicates the end of an crescendo section which began at the last \crescBegin tag; the crescendo ends with dynamic marking *s* and associated absolute intensity value of *r* (see above).

\dim(*notes***)** indicates an unspecific diminuendo on *notes*.

\dim<*s*>**(***notes***)** indicates a diminuendo starting from the current dynamic marking to dynamic marking *s* on *notes*.

\dim<*s*,*r*>**(***notes***)** indicates a diminuendo starting from the current dynamic marking to dynamic marking *s* with associated absolute intensity value of *r* (see above) on *notes*.

\dimBegin indicates the beginning of a crescendo section which ends at the next \dimEnd tag; the crescendo starts from the current dynamic marking.

\dimEnd indicates the end of an unspecific diminuendo section which began at the last \dimBegin tag.

\dimEnd<*s*> indicates the end of a diminuendo section which began at the last \dimBegin tag; the diminuendo ends with dynamic marking *s*.

\dimEnd<*s*,*r*> indicates the end of an diminuendo section which began at the last \dimBegin tag; the diminuendo ends with dynamic marking *s* and associated absolute intensity value of *r* (see above).

Remarks:

- The \intens tag is associated with the metrical position of the note immediately following it and carries to the next dynamic marking.
- The *s* parameter is a string containing a dynamic marking like "f", "p", "mf", "fp", etc; it is recommended to use only markings from conventional music notation.
- For crescendos and diminuendos the appropriate tags and parameters should be used, using \cresc for representing diminuendos or \dim for crescendos should be avoided.
- For longer crescendo or diminuendo sections the \crescBegin and \crescEnd or the \dimBegin and \dimEnd tag combinations should be used; this applies also for situations, where the crescendo or diminuendo section overlaps with other tagged ranges such as slurs.

Examples:

```
[ \intens<"mf"> c2/2 \intens<"pp"> d1/2] % dynamic markings
[ \intens<"mf",0.5> c2/2 \intens<"pp",0.2> d1/2] % same with MIDI velocity
[ \cresc( a#0/4 g# c#1)] % unspecific crescendo
[ \intens<"mf"> \dim<"mp">( a#0/4 g# c#1)] % diminuendo with dynamic markings
[ \intens<"mf",0.6> \dim<"mp",0.3>( a#0/4 g# c#1)] % same with MIDI velocity

% same using \dimBegin, \dimEnd:
[ \intens<"mf",0.6> \dimBegin a#0/4 g# c#1 \dimEnd<"mp",0.3>]
```

2.7.2 Tempo indications

\tempo<*s*> indicates a tempo marking *s*

\tempo<*s*,*s2*> indicates a tempo marking *s* and associates it with an absolute tempo as specified by string *s2*; s2 is of the general form *"x/y=n"* where *x/y* is a duration value like 1/4 or 3/8 and *n* an integer number which indicates the number of these duration values per minute.

\accel(*notes*) indicates an unspecific accelerando on *notes*.

\accel<*s*>(*notes*) indicates an accelerando from the current tempo indication to *s* on *notes*.

\accel<*s*,*s2***>(***notes***)** indicates an accelerando from tempo current tempo to tempo indication *s* with associated absolute tempo as specified by string *s2* (see above).

\accelBegin indicates the beginning of a accelerando section which ends at the next \accelEnd tag; the accelerando starts from the current tempo.

\accelEnd indicates the end of an unspecific accelerando section which began at the last \accelBegin tag.

\accelEnd<*s***>** indicates the end of an accelerando section which began at the last \accelBegin tag; the accelerando ends with tempo indication *s*.

\accelEnd<*s*,*s2***>** indicates the end of an accelerando section which began at the last \accelBegin tag; the accelerando ends with tempo indication *s* and associated absolute tempo as specified by string *s2* (see above).

\rit(*notes***)** indicates an unspecific ritardando on *notes*.

\rit<*s***>(***notes***)** indicates a ritardando from the current tempo to tempo indication *s* on *notes*.

\rit<*s*,*s2***>(***notes***)** indicates a ritardando from the current tempo to tempo indication *s* with tempo indication *s* and associated absolute tempo as specified by string *s2* (see above).

\ritBegin indicates the beginning of a ritardando section which ends at the next \ritEnd tag; the ritardando starts from the current tempo.

\ritEnd indicates the end of an unspecific ritardando section which began at the last \ritBegin tag.

\ritEnd<*s***>** indicates the end of a ritardando section which began at the last \ritBegin tag; the ritardando ends with tempo indication *s*.

\ritEnd<*s*,*s2***>** indicates the end of a ritardando section which began at the last \ritBegin tag; the ritardando ends with tempo indication *s* and associated absolute tempo as specified by string *s2* (see above).

Remarks:

- The \tempo tag is associated with the metrical position of the note immediately following it and carries to the next tempo indication.
- The *s* parameter is a string containing a tempo indication like "Andante", "Allegro", "Fast", etc; it is recommended to use only markings from conventional music notation.
- For accelerandos and ritardandos the appropriate tags and parameters should be used, using \accel for representing ritardandos or \rit for accelerandos should be avoided.
- For longer accelerando or ritardando sections the \accelBegin and \accelEnd or the \ritBegin and \ritEnd tag combinations should be used; this applies also for situations, where the accelerando or ritardando section overlaps with other tagged ranges such as slurs.

Examples:

```
[ \tempo<"Moderato"> c2/2 d1/2] % tempo indication
[ \tempo<"Moderato","1/4=60"> c2/2 d1/2] % same with absolute tempo
[ \rit( a#0/4 g# c#1)] % unspecific ritardando
[ \accel<"Presto">( a#0/4 g# c#1)] % accelerando
[ \accel<"Presto","1/2=160">( a#0/4 g# c#1)] % same with absolute tempo

% same using \accelBegin, \accelEnd:
[ \accelBegin a#0/4 g# c#1 \accelEnd<"Presto","1/2=160">]
```

2.7.3 Instrument

instr<*s*> sets instrument for current voice to *s*.

instr<*s*,*s2*> sets instrument for current voice to *s* and associates it with the instrument specified by *s2*; MIDI instrument codes can be accessed by values like *"MIDI 3"* (= MIDI instrument 3).

Remarks:

- The \instr tag is usually used at the beginning of voices, but can also be used within voices to indicate instrument changes.
- The *s* parameter indicates the name of the instrument; it is recommended that the standard names or abbreviations commonly used in scores be used.
- The *s2* parameter specifies the instrument used for playback; string values "MIDI *x*", where *x* is a number between 0 and 127 represent the standard MIDI instruments.

Examples:

```
[ \instr<"Flauto 1"> c1/2 e&/4 e& | _ f a&/2 ]
[ \instr<"Piano", "MIDI 0"> 0/8 h | c1/2 c/4 | c/2. ]   % with MIDI code
[ \instr<"Cl 1 (in A)">  d&/2 g&/4 _ \instr<"BCl"> e&-1/1 ] % instrument change
```

2.7.4 Ties

tie(*notes***)** ties *notes* together, only to be used for two or more notes with identical pitch.

Remarks:

- Semantically, a group of tied notes is equivalent to a single note with the respective combined relative duration. However, ties are considered to be significant for applications in notation.
- Ties are neither identical nor equivalent to slurs between notes with identical pitches. Whereas a slur indicates a grouping, a tie actually merges the notes it encompasses.

Examples:

```
[ \tie(d1/4 d/16) e/8 ]  % ties the first two notes together
[ \tie(c#1/1 | c# | c#) ]  % tied group with barlines
```

2.7.5 Slurs

\slur(_notes_**)** slur encompassing the group _notes_

\sl(_notes_**)** shorthand for \slur(_notes_)

\slur<_dy1,dy2_>**(**_notes_**)** same as \slur(_notes_), but the slur begins _dy1_ halfspaces above the head of the first note and ends _dy2_ halfspaces above the head of the last note (upward slur for positive values, downward slur otherwise)

\slur<_dy_>**(**_notes_**)** shorthand for \slur<_dy,dy_>(_notes_)

\sl<_dy_>**(**_notes_**)** shorthand for \slur<_dy,dy_>(_notes_)

\slur<_dx1,dy1, dx2,dy2, dx3,dy3_>**(**_notes_**)** three-control-point bezier slur encompassing _notes_; with control-points _(x1+dx1,y1+dy1)_, _(x1+dx2,y1+dy2)_, and _(x2+dx3,y2+dy3)_, where _(x1,y1)_ is the position of the head of the first note and _(x2,y2)_ the position of the head of the last note in _notes_; the _dx_ are measured in metric units (one metric unit always corresponds to a whole note), the _dy_ in half-spaces.

\sl<_dx1,dy1, dx2,dy2, dx3,dy3_>**(**_notes_**)** shorthand for \slur<_dx1,dy1, dx2,dy2, dx3,dy3_>(_notes_).

Remarks:

- Slurs between notes with identical pitches are neither identical nor equivalent to ties. Whereas a slur indicates a grouping, a tie actually merges the notes it encompasses.
- Slurs can be nested.
- The position parameters _dx1,dy1,etc._ are floating point numbers in standard (not scientific!) notation or integer numbers; the reference positions are at the center of the noteheads.

Examples:

```
[ \sl(d1/4 e/16) f0/8 ]      % slur between first two notes
[ \slur<+10,+12>(d1/4 _ e/16) f0/8 ] % non-standard slur
[ \slur<0,+10, 0.25,+16, 0,+12>(c#1/4 g f#) ]  % 3-control-point bezier slur
[ \sl(c#1/4 e) \sl(g f#)) ]         % nested slurs
```

2.7.6 Accents and Expressive Markings

\stacc(_notes_**)** indicates that _notes_ are marked staccato

\accent(_notes_**)** indicates that _notes_ are accented

\ten(_notes_**)** indicates that _notes_ are marked tenuto

\marcato(_notes_**)** indicates that _notes_ are marked marcato (strongly accented)

Remarks:

- (none)

Examples:

```
[ e1/4 \stacc(e&/8 e) e/2] % a staccato group
[ e1/4 \ten(e&/8 d&) e/2] %  a tenuto group
[ \marcato(e1/4 e) f#/2] %  a marcato group
[ \accent(e1/4 e) f#/2] %  an accented group
```

2.7.7 Trills and Ornaments

\trill(*chords***)** indicates trills for each note in the first voice of *chords*; the second voice notes are the auxiliary notes for the trills

\trill<*i*>**(***chords***)** same as above but specifies the trills to be executed as a sequence of $1/i$ notes

\mord(*chords***)** indicates mordents for each note in the first voice of *chords*; the second voice notes are the auxiliary notes

\mord<*i*>**(***chords***)** same as above but specifies the mordents to be executed using $1/i$ notes

\turn(*chords***)** indicates turns for each note in the first voice of *chords*; the second and third voice notes are the auxiliary notes

\turn<*i*>**(***chords***)** same as above but specifies the turn to be executed as a sequence of $1/i$ notes

Remarks:

- For trills and mordents, each chord should have exactly two voices which specifiy the main and auxiliary note, respectively. Upward and downward trills as well as standard and inverse mordents can be realised by specifying these notes accordingly.
- For turns, each chord should have exactly three voices which specifiy the main, the first and the second auxiliary note, respectively. Standard and inverse turns can be realised by specifying these notes accordingly.

Examples:

```
[ e1/4 \trill({e&/4,f}) e/2] % an unspecific trill
[ e1/4 \trill<32>({e&/4,f}) e/2] % trill to be executed as 1/32 notes
[ e1/4 \trill<32>({e&/4,f} {c/2,c#}) e/2] % trill sequence
[ e1/4 \mord({e&/4,f}) e/2] % a mordent
[ e1/4 \mord<32>({e&/4,d}) e/2] % inverse mordent to be executed as 1/32 notes
[ e1/4 \turn({e&/4,f,d}) e/2] % a turn
[ e1/4 \turn<64>({e&/4,d,f}) e/2] % inverse turn to be executed as 1/64 notes
```

2.7.8 Tremolo

\trem(*notes***)** indicates (unmeasured) tremolo for all *notes*

\trem<*i*>**(***notes***)** same as above but specifies the tremolo to be executed as a sequence of $1/i$ notes

Remarks:

- Fingered tremolo can be realised using \trill

Examples:

```
[ e1/4 \trem(e&/4) e/2] % unmeasured tremolo
[ e1/4 \trem<32>(e\&/4) e/2] % measured tremolo
[ e1/4 \trem<32>(e\&/4 c/2) e/2] % tremolo sequence
```

2.7.9 Fermatas

\fermata indicates a fermata at the current metric position (for use between notes)

\fermata(_notes_**)** indicates fermatas for all *notes*

Remarks:

- (none)

Examples:

```
[ e1/4 \fermata e/2] % fermata between notes
[ c0/4 e d \bar \fermata e/2] % fermata on barline
[ c0/4 e \fermata(d/2)] % fermata on single note
[ c0/4 e \fermata(d/2 d#/2 e#/1)] %  fermata sequence
```

2.7.10 Grace Notes and Cue Notes

\grace(_notes_**)** marks *notes* as grace notes preceeding the first note after *notes*.

\grace<_i_**>(**_notes_**)** marks *notes* as grace notes with value $1/i$ preceeding the first note after *notes*.

\cue(_notes_**)** marks *notes* as cue notes for the current voice.

\cue<_s_**>(**_notes_**)** marks *notes* as cue notes played by instrument *s*.

Remarks:

- For grace notes, no durations have to be specified.
- For the \cue tag, the *s* parameter indicates the name of the instrument; it is recommended to use the standard names or abbreviations as commonly used in scores.

Examples:

```
[ \grace(g1 f) e/4 \grace(f) e/2] % grace notes
[ \grace<32>(g1 f) e/4 \grace(f) e/2] % 1/32 grace notes
[ \cue(d#1/4 d# d#) g#/2] % unspecific cue notes
[ \cue<"Ob 1">(d#1/4 d# d#) g#/2] % cue notes played by Oboe 1
```

2.7.11 Repetitions

\repeatBegin indicates the beginning of a repeated section

\repeatBegin<*i*> indicates the beginning of section which is played *i* times (the standard \repeatBegin corresponds to \repeatBegin<2>)

\repeatEnd indicates the end of a repeated section

\repeatEnd<*i*>(*notes*) indicates the end of a repeated section where *notes* are only to be played in the *i*th repetition

Remarks:

- For simple repetitions with variant endings, \repeatEnd<1>(...) is used to indicate the ending for the first round (prima volta), while the notes immediately succeeding this tag represent the ending of the second round (2nda volta).
- For multiple repetitions with variant endings, one \repeatEnd<*i*>(...) tag is used to indicate the endings for all but the last round, which again is represented by the notes immediately succeeding the last \repeatEnd tag. Note that in this case, one \repeatBegin tag is followed by multiple \repeatEnd tags. The \repeatEnd tags are required to appear in the correct order, i.e., \repeatEnd<1>(...) before \repeatEnd<2>(...) and so on.

Examples:

```
[ \repeatBegin c1/2 g/4 f ... e/4 g/2 \repeatEnd ] % simple repetition

% simple repetition with variant ending:
[ \repeatBegin c1/2 g/4 f ... \repeatEnd<1>(e/4 g/2) e&/4 f e& ]

[ \repeatBegin<3> c1/2 g/4 f ... e/4 g/2 \repeatEnd ] % multiple repetition
[ \repeatBegin<3> c1/2 g/4 f ... \repeatEnd<1>( e/4 g/2 )
  \repeatEnd<2>( e/4 g f ) e&/4 f e& ] % same with variant endings
```

2.7.12 Clefs

\clef<*s*> sets clef *s*, where *s* is a string indicating the form of the clef ("g","f","c","perc" = percussion, "gg" = double g-clef) and the position within the staff ("1" for lowest line, ..., "5" for highest line). Optionally, this can be succeeded by "+8","-8", "+15","-15" for transposed clefs (octave or double octave up or down).

Remarks:

- If the staff-position is omitted, default values of "g2", "f4", "c3" are assumed;
- for common clefs, the following synonymes can be used:
  - "treble" = "violino" = "g2"

  - "bass" = "basso" = "f4"
  - "tenor" = "c4"
  - "alto" = "c3"

- Clefs can appear at the beginnig or in the middle of a voice, the latter indicating a clef change.

Examples:

```
\clef<"f">  or  \clef<"f3">  or  \clef<"basso"> % bass clef
\clef<"g2-8">         % transposed treble clef
\clef<"gg">  or  \clef<"gg2">   % double g2 clef
```

2.7.13 Meter / Time Signatures

meter<*s*> sets time signature to *s*, where *s* is of the form *"n/m"* for positive integers *n,m*.

Remarks:

- Additionally, the following special time signature can be used:
  - "C" or "c" for 4/4 with "C" symbol
  - "C/" or "c/" for 2/2 with alla breve symbol
- The \meter tag can appear at the beginning or in the middle of a voice, the latter indicating a change in meter.

Examples:

```
\meter<"2/4">   % 2/4 meter
\meter<"3/8">   % 3/8 meter
\meter<"C">   % 4/4 meter with C time signature
```

2.7.14 Key / Key Signatures

key<*i*> sets key signature to *i* sharps, if *i* is positive, to |*i*| flats, if *i* is negative. For *i*=0, the key signature contains no sharps or flats.

key<*s*> indicates a key of *s*, where *s* is of the form "*n*", "*n#*" or "*n&*" for any note name *n*. Uppercase and lowercase letters are used to denote major and minor modes, respectively.

Remarks:

- The \key tag can appear at the beginning or in the middle of a voice, the latter indicating a change in meter;
- within one piece, one of the two formats of the key-tag should be used consistently.

Examples:

```
  \key<0>      % no sharps or flats
  \key<-3>     % 3 flats, E& major or c minor
  \key<+2>  or  \key<2>  % 2 sharps, D major or h minor
  \key<"C">   % C major, no accidentals
  \key<"e&">    % e& minor, 6 flats
```

2.7.15 Octava transpositions

oct<*i*> indicates beginning of a section which is transposed by *i* octaves upwards.

oct<*i*>(*notes*) indicates transposition of *notes* by *i* octaves upwards.

Remarks:

- Octave transposition *does* affect the notated pitches such that "\oct<+1>(c1)" is pitch-equivalent to "c2".
- Negative numbers *i* can be used to indicate downward transpositions.
- \oct<*+1*> corresponds to "8va" in standard notation, \oct<*-1*> to "8va bassa", \oct<*+2*> to "15ma" etc.

Examples:

```
[ h2/4  \oct<+1> c2 e/2 \oct<0> e1/2]  % 8va transposition
[ h2/4  \oct<+1>( c2 e/2 ) e1/2]  % same, using different notation
[ h1/4  \oct<-2>( d0 e/2 ) e1/2]  % 15ma bassa
```

2.7.16 Staves

staff<*i*> assigns the voice it appears in to staff number *i*

staff<*str*> assigns the voice it appears in to staff named *str*.

Remarks:

- Voices having the same \staff tag share the same staff.
- Standard usage for \staff tags is exactly one \staff tag per voice appearing before the first note or rest.
- Mid-voice \staff tags can be used to indicate voices changing staves.

Examples:

```
{ [ \staff<1> e1/8. \slur(f/16 e/8) d c h0 a/4],  % first voice
  [ \staff<1> c1/4 d e/2],         % second voice, same staff
  [ \staff<2> a0/4 h \slur(c1/8 h0) a/4] }   % third voice, separate staff
```

2.7.17 Beams

\beamsAuto sets beaming to automatic mode; automatic beaming usually depends on the meter.

\beamsOff switches beaming off, i.e., only explicitly beamed notes will be beamed.

\beam(*notes***)** beam *notes*; can be nested to enforce non-standard beam groupings.

\bm(*notes***)** shorthand for \beam(*notes*)

\beam<*dy1,dy2***>(***notes***)** same as \beam(*notes*), but specifies the beam to begin *dy1* half-spaces above (or below, depending on the sign) the first note and to end *dy2* half-spaces above (or below) the last note; the stem lengths are automatically adjusted accordingly.

\bm<*dy1,dy2***>(***notes***)** shorthand for \beam<*dy1,dy2*>(*notes*)

\beam<*dy***>(***notes***)** shorthand for \beam<*dy,dy*>(*notes*).

\bm<*dy***>(***notes***)** shorthand for \beam<*dy,dy*>(*notes*).

Remarks:

- \beam tags always override \beamsAuto or \beamsOff.
- If \beam tags are not nested to account for each level of beaming, beams are assumed to be grouped according to standard rules.
- If \beam tags with explicit beam positions are used for multiple beams, all the beams are assumed to be parallel with a standard distance from one another and the beam positions refer to the toplevel beam.
- Rests can be included in the range of \beam tags, in this case they are assumed to be parted of the beamed structure.
- The position parameters *dy1,dy2* are floating point numbers in standard (not scientific!) notation or integer numbers; the reference position is at the center of the notehead.

Examples:

```
% beam the first group, while the last two eigth notes are not beamed:
[ \beam(d1/16 e c/8) \beamsOff d e ]

[ \beamsOff \beam(d1/16 e c/8) d e ]  % equivalent to the above
[ \beam( \beam(d1/16 e) \beam(c d))]  % explicitly group the 1/16th in 2+2
[ \beam<6,8>( d1/16 e c d))]  % unusual position of the beams
```

2.7.18 Stems

\stemsAuto sets the stem direction to auto mode, i.e., notes have downward stems if they are lower than the 3rd line (if any), upward stems otherwise.

\stemsUp sets stem directions to upward mode, i.e, all stems are drawn upwards.

stemsDown sets stem directions to downward mode, i.e, all stems are drawn downwards.

Remarks:

- These tags determine the stem direction from where they occur to the next occurrence of a \stems∗ tag or the end of the note sequence (or voice of a segment).
- The stem directions inferred for \stemAuto depend on the current clef.
- If no \stems. . . tag is specified, \stemAuto is assumed.

Examples:

```
[ \clef<"g"> \stemsAuto c1/4 e g h/2]

% same as above with explicit stem directions:
[ \clef<"g"> \stemsDown c1/4 e g \stemsUp h/2]

[ \clef<"g"> \stemsUp c1/4 e g h/2] % all stems upwards
```

2.7.19 Barlines and Tactus Strokes

bar indicates a barline at the position where it occurs.

'|' shorthand for \bar.

bar<i> indicates barline and start of measure *i*

doubleBar indicates double barline

doubleBar<i> indicates double barline and start of measure *i*

tactus indicates tactus stroke (dotted line or comma mark)

Remarks:

- Barlines can be automatically inferred from the information provided by the \meter tag.
- If \meter information is present, explicit barlines can be used to facilitate metric consistency checking or to realise up-beats.
- If \meter information is present, explicit barlines can be used to enforce incomplete measures.
- If \meter information is not present, barlines can be used to place arbitrary barlines.
- Double barlines should only be explicitly specified if their usage cannot be inferred from the context such as for key and meter changes.
- Tactus strokes are used to indicates metrical subdivisions (such as beats).

Examples:

```
[ c1/2 e&/4 e& | _ f a&/2 ]   % two 4/4-measures
[ a0/8 h | c1/2 c/4 | c/2. ]   % two 3/4 with up-beat
[ \meter<"C"> c1/4 d e/2 \bar<2> e/4 f g \bar<3> _*1 ] % measure 3 starts 1/4 early
[ c1/2 \tactus e&/4 e& |_ f \tactus a&/2 ]   % two 4/4-measures with tactus strokes
[ c1/2 e&/4 e& \doubleBar _ f a&/2 ]   % two 4/4-measures with double barline
[ c1/2 e&/4 e& \doubleBar<15> _ f a&/2 ]   % same with measure number
```

2.7.20 Text

text<*s*> places text *s* at current metric position

t<*s*> shorthand for \\text<*s*>

text<*s*,*dy*> same as above, but specifies vertical reference position as *dy* halfspaces below the lowest staff-line.

t<*s*,*dy*> shorthand for \\text<*s*,*dy*>

Remarks:

- Although \\text can be used to represent lyrics, Advanced GUIDO will provide a more appropriate and elaborate way for this.
- The position parameter *dy* is a floating point number in standard (not scientific!) notation or an integer number; the reference positions are the bottom of the lowest staff line and the lower left corner of the first character in *s*.

Examples:

```
[ a#0/2 \text<"l.h."> g# c#1)] % standard text
[ a#0/2 \text<"l.h.",-4> g# c#1)] % standard text 4 halfspaces below staff
[ a#0/2 \text<"Cara mia"> g# c#1)] % simple lyrics
```

2.7.21 Title and Composer

title<*s*> specifies the title of the piece as *s*

composer<*s*> specifies the composer of the piece as *s*

Remarks:

- Should be only used once at the beginning of the first voice.

Examples:

```
[ \title<"Aubade"> \composer<"Holger H. Hoos"> e&2/8 _ ] % title and composer
```

2.7.22 Markers and Named Sections

\mark<*s*> sets a marker labelled *s* at the current metric position

\label<*s*> sets a label *s* for the whole piece.

\label<*s*>(*notes*) sets a label *s* for *notes*

Remarks:

- Marks can be used for various purposes such as representing rehearsal numbers, optically marking chords or notes, and acoustically marking time positions; their actual usage depends on the specific software supporting GUIDO.
- Labels can be used to mark a whole piece (segment or sequence), in which case they should appear at the beginning of the first voice. Furthermore they can be used to mark arbitrary sections within a piece. This can be used to indicate motives, themes, or other analytical information.

Examples:

```
[ c1/2 g/4 \mark<"A"> f e f ] % simple marker
[ \mark<"ch:c7"> {c1/4,e&,g,b&} ] % marked chord
{ [\label<"A"> c1/2 g/4 f],[c0/1] } % labeled segment
[ c1/2 g/4 \label<"a1">(g1/8 f e/2 f/4 d/2) ] % labeled section (motive)
```

3 Advanced GUIDO Notation

Advanced GUIDO Notation covers some of the more advanced concepts not covered in Basic GUIDO Notation such as glissando, arpeggio, clusters, microtonal alterations, different types of noteheads, different types of staves, and many features from contemporary notation. It also addresses issues of advanced score setting such as exact spacing and positioning of notation and graphical elements.

Advanced GUIDO Notation will be realised by an extended set of standard tags and specified in Part 2 of this report.

4 Extended GUIDO Notation

Extended GUIDO Notation includes various features which extend Basic and Advanced GUIDO Notation beyond conventional music notation (CMN). Some of these features are generic pitch-classes, microtonal tuning systems and exact timing, generic tone descriptions, abstract and hierarchical scores, generalized events and user-defined tags.

Extended GUIDO Notation will be realised by introducing an extended set of event types; the specification will form Part 3 of this report.

Acknowledgements

We wish to thank Michael Fuhlbrügge, Kai Flade, Jürgen Kilian and Thomas Helbich for their contributions to many discussions in the early stages of the GUIDO development and for their comments on earlier versions of this document.

The first author also wishes to thank the German National Merit Foundation (Studienstifung des deutschen Volkes) and Prof. Dr. Wolfgang Bibel (TU Darmstadt) who supported his stay at the University of British Columbia in Vancouver during which the main part of this work was done.

Appendix - Basic GUIDO Notation Examples

Example 1:

2 voices, 2 staves; dynamic markings in first voice

```
{ [\title<"No.3"> \tempo<"Andantino">
  \staff<1> \clef<"g"> \key<+1> \meter<"3/8">
  \i<"p"> d2/8 |
 \sl(\dim(d h1)) h
 \sl(\dim(h g)) g
 \sl(\cresc(f# a c2))
 \sl(c h1) ],
[ \staff<2> \clef<"g">  \key<+1> \meter<"3/8">
  _/8 |
  h1 _ _
  g _ h0
  \sl(c1 f# a)
  \sl(a g)]
}
```

Example 2:

2+1 voices, 2 staves; dynamic markings in extra voice

```
{ [\title<"No.3"> \tempo<"Andantino">
  \staff<1> \clef<"g"> \key<-1> \meter<"6/8">
  \sl(a1/8 h& c2) \sl(f1/4 f2/8)
  \sl(\tie(e/8 e/18) g f e f g
  h&1/8 ],
[ \staff<2> \clef<"g">  \key<-1> \meter<"6/8">
  f1/8 _ _ _/4 a/8
  \sl(h&/4. g/8) ],
[ \i<"dolce"> _/8 \cresc(_*5/8)
  \dim(_*/4/8)]
}
```

Example 3:

one voice; grace notes and meter-change

```
{ [\title<"II"> \tempo<"Adagio">
  \staff<1> \clef<"tenor"> \key<-5>
  \meter<"4/4"> \i<"p"> \i<"espress">
  \sl(h&0/8 f1 d&2/4)
  \grace(c/16 d&) \sl(c/8 h&1 c2 d&)
 | \meter<"5/4">
  \sl(h&0/8 f1 d&2/8. c/32 d& e&/8 d&)
  \grace(c/16 d&) \sl(c/8 h&1 c2 \grace(e&) d&)
  ] }
```