GUIDO Music Notation – Towards an Adequate Representation of Score Level Music

Holger H. Hoos[*][†] Keith A. Hamel[‡] Kai Flade[†] Jürgen Kilian[†]

Abstract

GUIDO Music Notation is a novel approach for adequately representing score-level music. Using a plain-text human-readable and platform independent format, the GUIDO design is based on a conceptually simple, yet powerful and easily extensible formalism. GUIDO Notation is focused on musical and logical concepts, but also supports notational and performance-related aspects within a coherent formal framework. The key feature of our design is *adequacy*, meaning that using GUIDO, simple musical concepts can be expressed in a simple way, while more complex musical notions require more complex representations. GUIDO Notation can be used for a broad range of applications, including notation software, compositional and analytical systems and tools, musical databases, and music on the World Wide Web. In this paper we introduce the basic concepts and features of GUIDO Music Notation and illustrate these giving some simple examples. We also compare GUIDO with other music representation formats like DARMS, MIDI, and NIFF. Based on our own experience, we belief that GUIDO Music Notation will prove to be very useful, both as a music representation format and as a data exchange format for musical information.

1 Introduction

The GUIDO Music Notation Format[1] is a novel, 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 only 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.

The GUIDO design is organised in three layers: Basic, Advanced, and Extended GUIDO Music Notation. *Basic GUIDO* introduces the basic GUIDO syntactical structures: events and tags. Using these concepts, the basic musical notions, such as notes, rests, accidentals, voices, chords, time and key signatures, clefs, dynamic markings, etc. are realised. The next layer, *Advanced GUIDO*, extends Basic GUIDO to allow the representation of exact score formatting information. This way, GUIDO Notation can be used in the context of professional notation software. Furthermore, Advanced GUIDO comprises more sophisticated musical concepts, such as advanced lyrics or figured bass. The third layer, *Extended GUIDO*, introduces features which are beyond conventional music notation including mechanisms for representing musical structure, abstract scores, micro-tonal music, exact timing information, and generalised timed events (for multi-media support). GUIDO Music Notation is designed as a flexible and easily extensible open standard. Thus, it can be easily adapted and customized to cover specialized musical concepts as might be required in the context of research projects in computational musicology or when integrating GUIDO and other computer music systems.

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

[*]Corresponding author.

[†]Fachbereich Informatik, Technische Universität Darmstadt, Alexanderstr. 10, D-64283 Darmstadt, Germany, Email: {hoos,flade,kilian}@informatik.tu-darmstadt.de

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

[1]GUIDO Music Notation 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).

application areas include notation software, compositional and analytical systems and tools, musical databases, performance systems, and music on the World Wide Web.

This paper is structured as follows. After this introduction, we describe the three layers of GUIDO Notation, discussing the main features and giving examples for typical usage. Next, we compare GUIDO to some other approaches for music representation, including MIDI [HS97], DARMS [Sel97], NIFF [GB96], SMDL [Slo97], and Common Music Notation (cmn) [Scho97]. Finally, we sum up the main features of this work and indicate some ongoing and further research and development regarding GUIDO Music Notation.

2 Basic GUIDO Notation

Basic GUIDO Notation covers many of the relevant aspects of specifying "simple" music. It is designed for the adequate representation of musical material, which can then be interpreted by computer music software and also be used for conventional music notation. All syntactical elements important to GUIDO are contained within Basic GUIDO. Within GUIDO notation, two basic syntatic elements are distinguished: events and tags. An event is a musical entity which has a duration (e.g. notes and rests) whereas tags are used to define musical attributes (such as a meter, a clef, or a key).

Notes and Rests: In GUIDO, notes are represented by their names (eg. `c d e`). As GUIDO has been developed for a broad range of musical applications, different systems of diatonic notenames (like, for example `do re mi`) are supported; in addition to the well-known pitch classes, GUIDO also provides chromatic pitch-classes: There is, for example, a pitch-class called `cis`, that is different to `c#` and `d&` (where the `#` denotes a sharp, the `&` denotes a flat), eventhough when played on a regular MIDI device sounds the same. This concept is important to cover aspects of 12-tone-music. Important to GUIDO is the fact, that only those parameters need to be specified, which are important for the application context. In the case of notes, the following parameters can (optionally) be applied[2]

notename accidentals octave duration

An example for a complete note description in GUIDO looks like `c#1*1/4` which represents a `c'`-sharp with duration of a quarter note. GUIDO allows arbitrary duration of notes, also including single or double dots. Accidentals can be applied multiple times, for example `c&&#-3*7/8`, which is an interesting feature for the description of extended tonality in music, that can be used to control self-tuning systems like MUTABOR [ARW92]. Other aspects of microtonal music are covered by Extended GUIDO (see Sec. 4). Rests are represented like notes; instead of a notename, an underscore is used: "`_*1/4`". Octave and accidentals are commonly ignored when applied to rests while durations are specified as for notes.

Sequences and segments: Basic GUIDO offers two orthogonal concepts for the description of complete pieces of music: *sequences* and *segments*. A *sequence* describes a series of temporally consecutive musical objects, whereas a *segment* describes a number of simultaneous musical objects. These two concepts adhere to the fact that there are mainly two normalforms for describing music: A chord-normalform and a poly-normalform. The first describes music as a sequence of chords (which is a sequence of segments), the other uses a set of sequences (a segment of sequences). The usage of either one of these normalforms for the description of a piece depends heavily on the style of the piece: homophonic choral music should be written in chord-normalform, whereas a fugue is generally more adequately described using poly-normalform. Basic GUIDO facilitates both normalforms by supporting an extended poly-normalform which allows chords within sequences of a segment (i.e. chords within polyphonic voices). This restriction is overcome in Extended GUIDO (see Sec. 4).

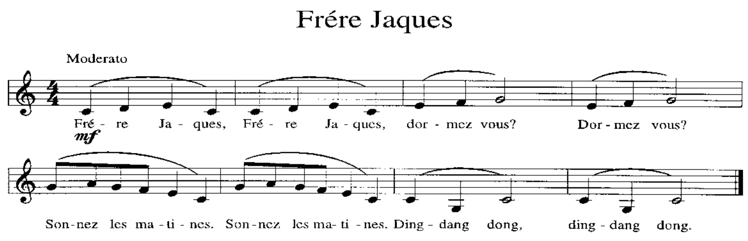
[2]Note that in the case of sequences, partial descriptions are supported.

Sequences and segments are realized in GUIDO using the following syntax: A sequence begins with a square bracket '[' and ends with the corresponding closing bracket ']'. Segments are enclosed in curly braces '{ ... }'. Musical objects within segments are seperated by commas. Within sequences and chords, octaves and durations are implied from the previous note, if they are not explicitly specified. A short expample of a commonly known piece should demonstrate this[3]:

```
{ [ c1*1/4 d e c | c d e c | e f g/2 | e/4 f g/2 ], [ { c0/4,e,g } { g-1,h,d0 } { c,e,g } ] }
```

Tags and basic musical concepts: Written and played music does not only consist of notes and rests but also contains other musical information. GUIDO can represent all the commonly known musical attributes using *tags*: A *tag* has a name, optional parameters, and an optional range of application. A multitude of tags are defined in Basic GUIDO: for a complete description see [HH97]. Just to name a view, there are tags to describe clefs, meter, tempo, intensity, crescendo, accelerando, and more. The semantical meaning of the tags defined in GUIDO can normally be seen easily from their names. The following example, which is an extended version of the one given above, illustrates the usage of tags:

```
{ [ \title<"Frère Jaques">
    \tempo<"Moderato"> \clef<"treble"> \key<"C"> \meter<"4/4"> \intens<"mf">
    \slur(c1*1/4 d e c) \slur(c d e c) \slur(e f g/2) \slur(e/4 f g/2)
    \slur(g/8 a g f e/4 c) \slur(g/8 a g f e/4 c) \slur(c g0 c1/2) \slur(c/4 g0 c1/2) ] }
```


Frére Jaques
Moderato
Fré - re Ja - ques, Fré - re Ja - ques, dor - mez vous? Dor - mez vous?
mf
Son - nez les ma - ti - nes. Son - nez les ma - ti - nes. Ding - dang dong, ding - dang dong.

3 Advanced GUIDO Notation

Advanced GUIDO Notation comprises some of the more advanced concepts not covered in Basic GUIDO Notation, such as glissandos, arpeggios, clusters, different types of noteheads and staves, and some established features from contemporary notation. It also addresses issues of advanced score formatting such as exact spacing and positioning of notational and graphical elements. Using Advanced GUIDO, it is possible to specify exact score formatting information that can be used by any kind of professional notation software. This feature should make Advanced GUIDO an ideal format for transmitting and exchanging notational information in a platform- and application-independent way.

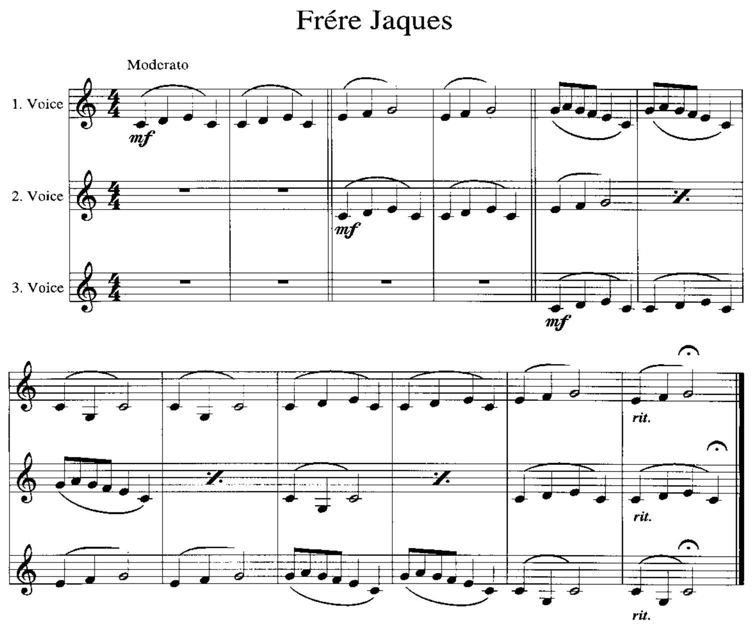
Exact formatting and score layout: To understand the exact formatting features included in Advanced GUIDO, it is necessary to understand the underlying graphical frame: All vertical spacing is defined using halfsteps, where one halfstep is defined as the difference in vertical position between two adjacent notes (e.g. between `c` and `d`). The mapping of halfsteps to length units is defined using the `\staffFormat` tag: in this way, arbitrarily sized staves can be displayed within the same system. Another layout tag is the `\pageFormat` tag, which determines the page geometry. The relation of rhythmical units (x dimension) to halfspaces (y dimension) can be defined with the `\spacing` tag. The most important tag to exactly specify locations is the `\space` tag, which overrides automatic spacing and forces the given amount of space (specified in halfspaces, mm, cm, in, or pt) to be inserted at the current position.

[3]For the aequivalent conventional music notation see the following example.

All tags defined in Basic GUIDO can be equally used within Advanced GUIDO; many of these tags have additional optional parameters for specifying exact positioning and scaling information. This approach ensures that adequacy is maintained: only when exact score formatting is really needed, additional parameters have to be supplied, otherwise these will be inferred automatically by the application when needed. This is similar to HTML, where not all tags are supported by all browsers.

Advanced notation and advanced musical concepts: Advanced GUIDO also supports features like glissandos, arpeggios, or clusters, which can be found in some scores. Not all of the advanced musical concepts have a commonly defined way of notation in conventional music notation: any notation software using GUIDO should deal with this in a controlled way. Another feature of Advanced GUIDO lies in the possibility to include and use arbitrary graphical elements. Using this feature it becomes possible to describe part of a score using newly constructed graphical elements. This feature seems to be important when notating modern music that expands the historically available musical expressiveness.

To exemplify some of the features of Advanced GUIDO, we further elaborate the example from the previous section:



```
{[\units<"mm"> \pageFormat<"A4",10,10,10,10>
\title<"Frere Jaques">  \tempo<"Moderato">
\instr<"1. Voice">
  \clef<"treble"> \key<"C">  \meter<"4/4">  \intens<"mf">
  \slur(c1*1/4 d e c) \slur(c d e c) \doubleBar
  \slur(e f g/2) \slur(e/4 f g/2) \doubleBar
  \slur(g/8 a g f e/4 c) \slur(g/8 a g f e/4 c) \newLine
  \slur(c g0 c1/2) \slur(c/4 g0 c1/2) \slur(c1*1/4 d e c) \slur(c d e c)
  \slur(e f g/2) \slur(\rit(e/4 f \fermata(g/2)))],
[\instr<"2. Voice">
  \clef<"treble"> \key<"C">  \meter<"4/4">  \intens<"mf">
  _*2/1 \doubleBar
  \intens<"mf">  \slur(c1*1/4 d e c) \slur(c d e c)  \doubleBar
  \slur(e f g/2) \slur(e/4 f g/2) \newLine
  \slur(g/8 a g f e/4 c) \slur(g/8 a g f e/4 c)
  \slur(c g0 c1/2) \slur(c/4 g0 c1/2)
  \slur(c1*1/4 d e c) \slur(\rit(c d e \fermata(c)))],
[\instr<"3. Voice">
  \clef<"treble"> \key<"C">  \meter<"4/4">
  _*2/1 \doubleBar _*2/1 \doubleBar
  \intens<"mf"> \slur(c1*1/4 d e c) \slur(c d e c)  \newLine
  \slur(e f g/2) \slur(e/4 f g/2) \slur(g/8 a g f e/4 c)
  \slur(g/8 a g f e/4 c) \slur(c g0 c1/2)
  \slur(\rit(c/4 g0 \fermata(c1/2)))]}
```

4 Extended GUIDO Notation

Unlike Basic and Advanced GUIDO, the third level of the GUIDO design, Extended GUIDO Notation, is still in the specification phase. It includes various features which extend Basic and Advanced GUIDO beyond conventional music notation and its final specification will reflect the experiences and needs of the groups (such as our cooperation partners at UBC/Vancouver and GRAME/Lyon) involved in using and testing GUIDO to date. Due to space restrictions, we present only a few concepts of Extended GUIDO here.

Exact Timing: While in Basic and Advanced GUIDO, durations are always specified as relative durations represented by fractions and tempo indications, Extended GUIDO allows the representation of exact timing by specifying absolute durations. This is realised by introducing the syntactic forms *ev*`*`*x*`ms` and *ev*`*`*x*`s` for durations of x milliseconds or seconds, where *ev* is an arbitrary event. Thus, `c2*100ms` is the note `c''` with a duration of 200 milliseconds. Relative and absolute durations can be arbitrarily mixed. Note that using this concept, it is possible to represent unquantized MIDI information in GUIDO. Thus, MIDI quantization can be seen as a transformation between different normalforms of GUIDO Notation.

Microtonal Tuning: While certain aspects of microtonality such as just tuning can already be realised in Basic GUIDO (using multiple accidentals such as in `d##`, `e`, or `f&`), Extended GUIDO introduces representations for other types of microtonal information. One of these is the use of generic pitch-classes such as `pc<`x`>` or `hz<`f`>`, which represent relative and absolute pitches in the following way: `pc<`x`>` is the pitch-class obtained by transposing `c` by x semitones (e.g., `pc<0>` = `c`, `pc<`-2.5`>` is a pitch-class 250 cents below `c`).[4] `hz<`f`>` is the pitch given by an absolute base frequency of f Hertz, thus `hz<440>` would be a 440hz `a'`. Based on these generic pitch-classes, arbitrary scales, such as an equally divided quarter-tone scale, can be represented. This, however, is not always the most adequate representation. If, for instance, standard notes are microtonally alterated, such as when using quarter-tone accidentals, a new tag `\alter<`x`>` is used, which alterates all pitches in its range by x semitones. Thus, `\alter<`-0.5`>(a1/8)` represents an eigth note `a1` alterated downwards by a quarter tone. For realising different tuning systems, another new tag `\tuningMap<`p, x`>` should be used. In its range, this tag alterates all notes of pitch-class p (specified as a string) by x semitones, such that `\tuningMap<"b",`-0.1`>` would adjust all notes `b`, regardless of their octave, by -10 cents. Note, how GUIDO Notation provides different concepts for adequately representing the different notions of microtonality, such as just tuning, arbitrary scales, microtonal alterations, and non-standard tunings.

Other Concepts: Further concepts covered by Extended GUIDO can be only briefly mentioned here. One of these are *hierarchical scores*, which allow arbitrary nesting of sequence- and segment-constructors. This way, the hierarchical structure of music can be adequately represented within GUIDO — this information can be useful, e.g., in the context of musical analysis applications or for musical databases. Extended GUIDO also allows you to represent *abstract scores*, which are basically scores containing variables. In some sense, abstract scores can be used to represent incomplete structural information such as musical schemata. Another important feature of Extendend GUIDO are *user-definable tags and events*. These allow the individual definition and use of convenient

[4]To be consistent with general usage of pitch-classes, we have chosen `c` as the reference point.

abbreviations for specific tag combinations, generalised events, more synonymes for notenames, tags, events, and musical fragments.

5 Related Approaches

Unlike GUIDO, the binary MIDI File Format [HS97] introduced in 1988, was not designed for analysis and notation purposes. It basically covers the representation of performance related information such as exact timing and intensity of notes. Because structural information like chords, slurs or ties cannot be stored in a MIDI file, a high- or multilevel description is almost impossible. A further, severe restriction of MIDI is the fact that accidentals cannot be represented. The big advantage of MIDI, however, lies in the fact that there is a huge amount of music available as MIDI files, and most music software supports MIDI. Although hypothetically almost any kind of musical information can somehow be encoded in MIDI by using specialized MIDI event types in a non-standard way, the representations obtained this way are far from adequate.

SMDL (Standard Music Description Language, ISO Standard 10743) [Slo97] is an architecture for representing musical information mainly for publishing and commercial purposes. SMDL is based on both SGML (Standard Generalized Markup Language) and HyTime (Hypermedia/Time-Based Structuring Language). Like GUIDO, SMDL is a text-based format which can be easily ported to a wide variety of platforms. Being designed as an interchange format, SMDL offers means to translate into and out of existing codes. SMDL does not impose any normalform, which makes it easy to adapt it to different needs. However, due to the lack of normalforms and both a simple and coherent formal framework, SMDL representations are hardly adequate, especially when it comes to simple music. Furthermore, as SMDL support is rather difficult to implement, it does not seem to be very widely used today. Similar arguments apply to NIFF (Notation Interchange File Format) [GB96], which is, like MIDI, a binary format. Like SMDL, NIFF is designed primarily for the representation and exchange of musical notation data. Again, we find a similar lack of adequacy as in SDML. An additional drawback of NIFF is that, even though some major companies are involved with its specification, apparently it is yet not widely used within the computer music community.

DARMS (Digital Alternate Representation of Musical Scores) [Eri74, Sel97], a text-based music representation developed in 1963, was one of the most prevalently used code for both music printing and music analysis until the early eighties. In many respects, GUIDO Notation is related to principles behind DARMS, like using text-based encoding, the existence of canonical normalforms, and extensibility. Nevertheless, there are some important differences. Although DARMS was intended (and actually used) for a broad range of compositional, notational, and analytical applications, conceptionally it is primarily a print-oriented code. Furthermore, in some respects, DARMS code reflects clearly the severe limitations of both space and parsing technology from the time of its development. Nevertheless, it can be considered as a partially adequate representation. To us, comparing DARMS and GUIDO is somewhat like comparing assembler language to a high-level programming language.

The most adequate music representation formalism we are currently aware of is Common Music Notation (*cmn*) [Scho97], a textual music description language which is part of the LISP-based Common Music system by Heinrich Taube. Like DARMS, *cmn* has been developed mainly for notational purposes. Nevertheless it is our impression that, unlike SMDL or NIFF, *cmn* is an adequate representation formalism in the sense defined before. There is a close correspondance between Advanced GUIDO Notation and *cmn*, and actually part of Advanced GUIDO has been inspired by the corresponding *cmn* concepts. There are two main differences between GUIDO and *cmn*. The first of these concerns the representation of notes and rest; while in *cmn* the same representation as in Score [Smi97] is used, GUIDO is based on a different model, which we think, is a bit more intuitive. Secondly, while *cmn* focuses on graphical aspects (represented, however, – like in GUIDO – as musical attributes) GUIDO also covers performance-related information. Finally, the various extensions realised in Extended GUIDO, such as microtonal tuning, exact timing, or hierarchical scores, do not have equivalents in *cmn*.

6 Conclusions and Future Work

In this paper, we introduced GUIDO Music Notation, a novel approach for adequately representing score-level music. While focussing on purely musical and logical concepts, the GUIDO design covers also graphical and performance-related aspects of music representation. Based on a conceptually simple yet powerful syntax, GUIDO is a human-readable and portable text format. Its most important feature is *adequacy*, meaning that simple musical concepts can be expressed in a syntactically simple way, while musically more complex material requires more complex representations in GUIDO.

GUIDO Notation is realised as a three level hierarchy. While Basic GUIDO covers basic musical concepts, Advanced GUIDO comprises exact score formatting and advanced musical concepts. Extended GUIDO introduces concepts which are not covered by conventional music notation, such as exact timing, micro-tonal tuning, generic pitch-classes, hierarchical and abstract scores, and user-defined GUIDO tags and events. GUIDO is designed as a flexibly extensible, open standard. Specific requirements which are not covered by the standard GUIDO design can be easily met by introducing new, application- or user-specific tags and events. This way, GUIDO Notation can be easily adapted to specific needs arising in the context of various computer music applications. Like HTML browsers, standard GUIDO applications, such as MIDI-converters or noteviewers, ignore tags and events they do not support while still being able to make use of the remaining information. This way, even when using customized GUIDO extensions, one can still interface with other standard GUIDO applications and use standard GUIDO tools.

GUIDO differs in many important aspects from existing approaches for music representation like DARMS, *cmn*, NIFF, SMDL, or MIDI. At the same time, it shares important concepts with some of these approaches. To our best knowlegde, the combination of features supported by GUIDO (particularly notions covered by Extended GUIDO) in connection with the easily extensible, syntactically simple and coherent formal framework which facilitates an adequate representation of musical material, is unique to GUIDO Music Notation.

The range of applications which can benefit from supporting GUIDO Notation comprises notation software, computer music systems, compositional and analytical software tools, musical data bases, and many more. Realising GUIDO support for music software and computer music applications is facilitated by a Parser Kit and some simple standard GUIDO tools which are available from the authors. A GUIDO-to-MIDI converter has already been realised. The reverse conversion is considerably more diffcult, since musical information which can be expressed in GUIDO has to be inferred from the low-level MIDI data; nevertheless a prototype for a MIDI-to-GUIDO converter has been implemented by the authors. Converters between GUIDO and other music representation formats are planned. To further promote GUIDO and its wide-spread use, a GUIDO WWW Server has been developed. This public WWW service converts GUIDO music descriptions into graphical images of conventional music notation to be included into web-pages; this way, it is for the first time possible to *dynamically* create and display music on WEB pages.

Other projects we are currently working on include the realization of a GUIDO interface for Keith Hamel's notation program NoteAbility [Ham97]. As part of a cooperation with GRAME at Lyon/France, we also work on GUIDO support and a customized GUIDO extension for Elody [OFL97], a compositional system based on lambda calculus which is currently being developed at GRAME. Further projects involves the realisation of GUIDO support for HARMONET [HFM92], a neural network based harmonizer developed at the University of Karlsruhe/Germany, and Mutabor [ARW92], a microtonal instrument developed at the Universities of Darmstadt and Dresden/Germany.

From our experience so far, we believe that GUIDO will prove to be extremely useful, both as a music representation format and as a data exchange format for musical information. However, at this point we do not consider GUIDO as a statically defined standard, but more as “work in progress”. It is our hope that, based on more experience gathered by propagating the ideas and the use of GUIDO within the community, GUIDO Music Notation will finally become what we want it to be: an adequate and widely used formalism for representing score-level music.

Acknowledgements

The work on GUIDO Music Notation is part of the SALIERI Project on Computational Musicology and Computer Music. We wish to thank Michael Fuhlbrügge, Matthias Huber and Thomas Helbich for their contributions in the early stages of the GUIDO development. We also thank Prof. Herrman K.-G. Walter and the Center for Interdisciplinary Studies in Technology, TU Darmstadt for their support of the SALIERI Project. The first author also wishes to thank the German National Merit Foundation (Studienstifung des deutschen Volkes) and Prof. 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.

References

[ARW92] Volker Abel, Peter Reiss, Rudolf Wille; MUTABOR II Ein Computergesteuertes Musikinstrument zum Experimentieren mit Stimmungslogiken und Mikrotönen; Technische Hochschule Darmstadt, Fachbereich Mathematik, Preprint Nr. 1513; Darmstadt 1992

[Eri74] R. Erikson; "The DARMS Project: A Status Report"; Computers and Humanities, 8, p. 161-172; 1974

[GB96] Cindy Grande, Alan Belkin; The Development of the Notation Interchange Format; Computer Music Journal 20(4):33-43; MIT 1996

[Ham97] Keith Hamel; NoteAbility Reference Manual; Opus 1 Music Inc.; Vancouver, B.C. 1997

[Ham93] Keith Hamel; "Music Printing"; in: Encyclopaedia of Computer Science and Technology: Kent A. and Williams G. (eds.); Vol. 27, Supp. 12, pp.93-106; Marcel Dekker; New York, 1993

[HFM92] Hermann Hild, Johannes Feulner, Wolfram Menzel; HARMONET: A Neural Net for Harmonizing Chorales in the Style of J.S.Bach; in: Advances in Neural Information Processing Systems 4 (NIPS*4); San Mateo, CA; p. 267-274; Morgan Kaufmann Publishers; 1992

[HH97] Holger H. Hoos, Keith Hamel; The GUIDO Music Notation Format – Specification Part 1; Technical Report TI 20/97; Darmstadt University of Technology; Darmstadt 1997; available from `http://www.informatik.tu-darmstadt.de/AFS/CM/GUIDO/docu/`

[Hoos94] Holger H. Hoos; Strukturorientierte Beschreibung und Erzeugung von Musik; Studienarbeit am Institut für Theoretische Informatik; TH-Darmstadt 1994

[HS97] Walter B. Hewlett, Eleanor Selfridge-Field; MIDI; in: Beyond MIDI, The Handbook of Musical Codes; Eleanor Selfridge-Field (ed.); pp. 41-72; MIT Press; Cambridge, Massachusetts 1997

[Lay85] Gareth Loy; Musicians Make a Standard: The MIDI Phenomenon; in: The Music Machine; Curtis Roads (ed.); p. 181-198; MIT Press; Cambridge, Massachusetts 1989

[OFL97] Yann Orlarey, Dominique Fober, Stéphane Letz; L'environnement de composition musicale Elody; in: Proceedings of JIM'97; p. 122-136; 1997

[Scho97] Bill Schottstaedt; Common Music Notation; in: Beyond MIDI, The Handbook of Musical Codes; Eleanor Selfridge-Field (ed.); pp. 217-221; MIT Press; Cambridge, Massachusetts 1997

[Sel97] Eleanor Selfridge-Field; DARMS, Its Dialects, and Its Uses; in: Beyond MIDI, The Handbook of Musical Codes; Eleanor Selfridge-Field (ed.); pp. 163-174; MIT Press; Cambridge, Massachusetts 1997

[Slo97] Donald Sloan; HyTime and Standard Music Description Language: A Document-Description Approach; in: Beyond MIDI, The Handbook of Musical Codes; Eleanor Selfridge-Field (ed.); pp. 469-490; MIT Press; Cambridge, Massachusetts 1997

[Smi97] Leland Smith; SCORE; in: Beyond MIDI, The Handbook of Musical Codes; Eleanor Selfridge-Field (ed.); pp. 252-280; MIT Press; Cambridge, Massachusetts 1997