3 NIFF Transcription and Generation of Braille Musical Scores

Didier Langolff[1], Nadine Baptiste-Jessel[2], Danny Levy[2]

[1]IUT Informatique
133 Avenue de Rangueil
31077 Toulouse cedex 4, France
langolff@info.iut-tlse3.fr

[2]UPS-IRIT
Centre TOBIA
118 Route de Narbonne
31062 Toulouse cedex, France

Abstract

The aims of MfB (Music for the Blind) software are to directly create Braille musical scores and to automatically transcribe into Braille notation musical scores previously represented in NIFF (Notation Interchange File Format). This article describes briefly some NIFF specifications and the precise MfB software features which facilitate these tasks.

COMPUTING IN MUSICOLOGY 12 (1999-2000), 34–44.

Our main objective is the creation of a software system which will allow automatic, accurate transcription of musical scores into Braille. The need we are trying to meet is to find a standard approach to representing musical notation so that it can be used by a variety of software and yet retain a pertinent structure.

From a technical perspective, visually impaired users have several sets of needs: those who compose need a method for converting sound to notation readable by sighted musicians. Thus they need an alternative way of using standard notation software, which generally depends on a graphical user interface. Those who wish to perform need scores in Braille musical notation, which is produced by impact printers as raised holes on light cardboard.[1] Sources which may be desirable to convert to Braille musical notation include ink-printed scores and the encoded data which produce them. Audio files (.WAV et al.) are immediately usable and require no special handling.

[1] Braille "fonts" which can be used with standard word-processing programs merely represent the configuration of the six-dot cells from which Braille characters are composed—for the benefit of sighted readers.

Encoded data can be obtained either from musical-score editors, where, however, problems of the representation code (i.e. the lack of an interchange standard for musical data) and/or the graphical interface (software unusable by the blind) may be encountered, or from an optical character recognition system for musical scores.

Our challenge was to decide which notation code we would use to represent the scores in printed form. To this end, we undertook a conceptual and structural study of different musical-notation encoding systems, such as DARMS, and proposed interchange standards such as SMDL (Sloan 1997, Langolff et al. 1997). NIFF [Notation Interchange File Format] was chosen.

In the first part of this article we will explain the reasons for the choice of NIFF, and in the second we will describe relevant features of MfB.

3.1 NIFF Notation

The lack of an accepted standard format for music notation has been for several years a source of great frustration for computer-based musicians, engravers, and publishers. Numerous attempts have been made in the past to create a standard format.

NIFF is a file format designed to allow the interchange of music-notation data between music-notation editing and publishing programs, including music-recognition programs. Its design is a result of combined input from many notation-software developers, music publishers, and experienced music-software users.

The effort resulting in NIFF is interesting to us for different reasons:

- NIFF files can be extracted from printed scores by different processes.

  We opted to use the scanning program *Midiscan*,[2] which automatically produces a NIFF translation., the other way being the result (the NIFF output) of editing programs. One of the main features of NIFF is that the description of a score in NIFF format is purely graphic. For example, a note is described not by its pitch name (A, B, C, ...) but by its position on the staff.

- The basic NIFF data structure is a tree.

  A NIFF file is divided into two sections: the Setup Section, containing general information related to the whole score, and the Data Section, containing the music symbols and layout information. The tree structure is inherent in the NIFF file structure and also in the use of the anchor concept. We will describe in the following paragraphs these key features of the NIFF format.

[2] Now superseded by *SmartScore*.

3.1.1 Chunks and Lists

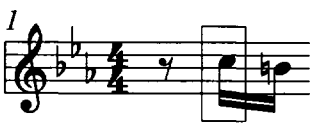
Lists and chunks, familiar from the Standard MIDI File Format, are the basic components of a NIFF file. All the lists and chunks available to create a NIFF file are presented in the NIFF specifications (NIFF 6.A, 1995).

THE DATA CHUNK

A chunk is the basic structure of a NIFF file (which is an extension of RIFF, the Microsoft Resource Interchange File Format developed for MS *Windows*). A chunk is composed of a header and a body. The header is written over eight bytes. Four bytes give the name of the chunk and four give the size of the body.

The chunk may be separated into two classes—the *setup chunk* and the *symbol chunk*. Symbol chunks describe the musical score. Setup chunks define the context of the description.

In the following table we present a NIFF file sample concerning the notehead chunk.

Notehead chunk specification		Printed score sample	Interpreted NIFF code
Notehead	**CHUNK**		
shape	**BYTE** 1 = breve (double whole) 2 = whole 3 = half (open) 4 = filled (solid) ... 14 = open triangle		**note: [6] (4, 5*, 0, 1, 0, 16**)**
staff step	**SIGNEDBYTE**		
duration	**RATIONAL**		
Default anchor: Default reference pt:	Time-slice The center of the bounding box of the notehead.		"[6]" is the size of the body. * position on the staff; first line has the value 0. ** duration of the note: 1/16th.
Tags: Part ID, Voice ID, placement tags, MIDI Performance, Grace Note, Cue Note, ...			

LISTS

A *list* is a type of chunk which can contain nested chunks. Its code of four characters is "LIST." The type of the list consists of a "four characters code" following the size field, which is followed by a series of chunks or other lists. Pages, systems, and staves are all represented by lists.

Staff list specification		NIFF notation
Staff	**LIST**	`LIST:[2382]syst` `  syhd:[0]( )`
Location:	System list →	
Required:	Staff Header chunk →	`  LIST`[1]`:[1252]`[2]` staf`[3]
Optional:	Any number of the following:	`    sthd:[18](...)`
	Time-Slice chunk	`    ...`[4]
	Any music symbol chunk	...
	NIFF Font Symbol chunk	
	Custom Shape Graphic chunk	[1] Four characters code (always "LIST")
	Text chunk	[2] Size of field
	Line chunk	[3] Type of list
		[4] Series of chunks or other lists

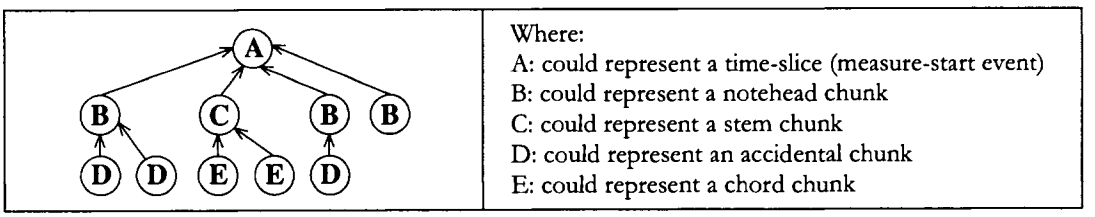
The chunks given in the body of a *list chunk* are the children of this chunk. List chunks generate a tree, called a *reading tree*, where nodes are the list chunks and leaves are the data chunks. The last level of the reading tree is the staff. Thus all symbols describing a staff are at the same level as the reading tree, but informally, the tree structure continues deeper than the staff level. Measure, stem, and notehead could be considered as nodes of a tree, taking into account the concept of anchor/dependent relationships.

3.1.2 The Anchor/Dependent Relationship

In NIFF, a musical symbol whose placement depends on one or more other symbols is called a *dependent symbol*, and the symbol or other chunk-type on which its placement depends is called its *anchor*. For each symbol chunk-type, a default anchor chunk-type is defined (an exception is made for the root: it is a *page-header chunk*).

An example of this is given in the first table above in the representation of the notehead chunk: note that the default anchor of the notehead is the *time-slice*. This means that every symbol chunk is a child of the immediately preceding chunk corresponding to the anchor type. For example, let (A←B, A←C, B←D, C←E) be a set of (Anchor←Dependent) relationships. The tree structure shown in Figure 1 would correspond then to the resulting sequence *ABDDCEEBDB*.

Figure 1. Example of tree structure concerning the anchor/dependent relationship.



Thus there are two tree structures in a NIFF file:

- An explicit tree—the reading tree, in which nodes are lists and data chunks are leaves. This tree is the NIFF file.
- An implicit tree—the final tree, in which data chunks may be nodes. The basic notion associated with the final tree is the anchor concept and the memory table. To build this tree, external information common to all NIFF files (the anchor/dependent relationship) is needed.

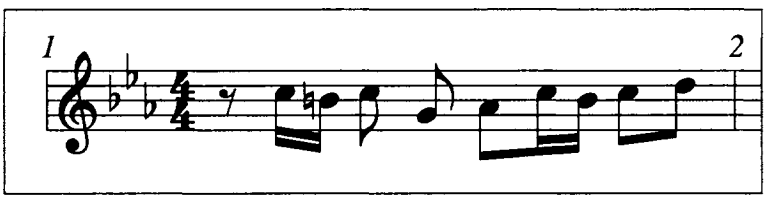
In the MfB software, we name the notes according to their pitch (A, B, C rather than the "staff step"); their octave number is given only if needed. We adopt this notation (different from NIFF notation) only to correspond to the Braille musical definition.

3.2 The MfB Software

After the study of each code specification (including Braille musical-notation code [Krolick 1996]) we began the development of the MfB software, which allows a sighted person to transcribe or create a Braille score. This software is composed of different modules—(1) the NIFF Interpreter, (2) the Punch, (3) the Dictionary, (4) the Command tool, and (5) the Scanner.

To present the features of these modules, we will take the example of the Bach extract shown in Figure 2.

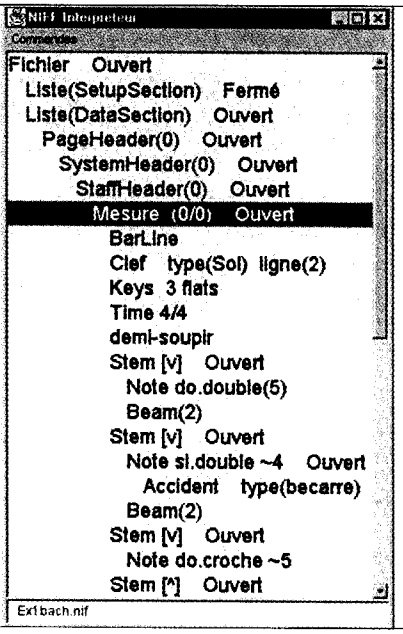
Figure 2. First measure of a Bach score.



3.2.1 The NIFF Interpreter: Textual Description, Navigation, and Transcription

The NIFF Interpreter enables the user to convert a NIFF-encoded score to a text file (NIFF code is a binary code). The navigation strategy is based on the tree structure. A NIFF-encoded score could be compared to a tree where a score contains pages, a page contains systems, a system contains staffs, and a staff contains musical items.

Figure 3. The NIFF Interpreter tool.



If you launch the NIFF interpreter, you have to open a NIFF file, and then you can see the root of the tree structure which appears (called "Fichier"). You could develop the tree by successive double-clicks on the nodes. For example, by a double-click on the root, two new nodes appear called "Liste(SetupSection)" and "Liste(DataSection)". These two nodes correspond to the two main sections of a NIFF file.

For a sighted person, it's easy to see the different levels of the tree structure with the indentation used.

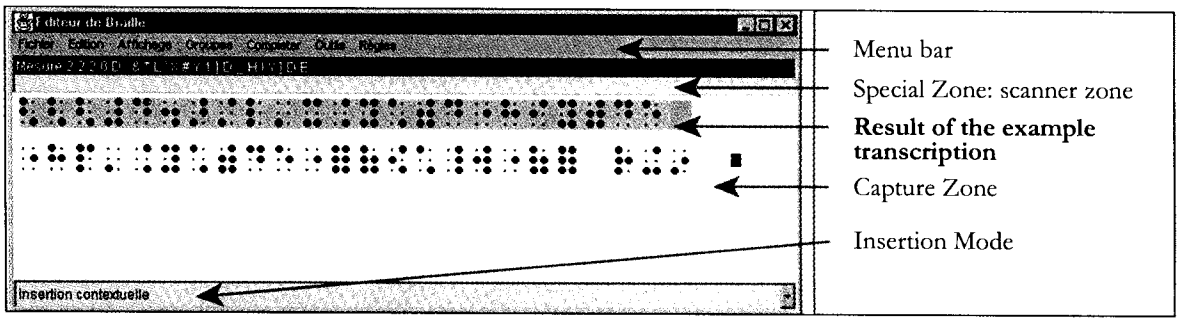
During the navigation, the user can at any time transcribe any selected part (a musical item, a measure, a staff, a system, or the whole score) by using the "commandes" menu and choosing the transcription command.

The result of the transcription appears in the capture zone of the Braille Editor window.

3.2.2 The Braille Editor: The Heart of the Software

The Braille Editor is the main tool of the software. The editing window is shown in Figure 4 (the screen menu is labelled in French).

Figure 4. The Braille Editor.



From the menu bar you can activate all the different tools from the *Outils* menu. The other menus are either more general—as, for example, the *Fichier* (file) and the *Edition* (editing) menus—or more specific, like the *Groupes* (grouping), *Completer* (finishing), and *Règles* (rules) menus.

These three menus allow respectively:

- the grouping of musical items by category such as stem, beam, or measure;
- the completion of a stem, a beam, or a measure (it is impossible to insert a new musical item in a completed group); and
- the application of some specific rules of Braille notation (use of Dot 3, rhythmic-group notation, and so forth) to an already drafted score.

The result of the transcription is highlighted. To obtain it, the user has to select the corresponding node (in this case the first measure has been selected) and activate the transcription in the NIFF interpreter.

The insertion mode has the default value we call a "contextual insertion." This mode takes into account all the Braille notation rules described in the *New International Manual of Braille Music Notation* (Krolick 1996). The two other available modes are "insertion before" and "insertion after," which allow a "manual" insertion before or after the current item without the use of Braille notation rules.

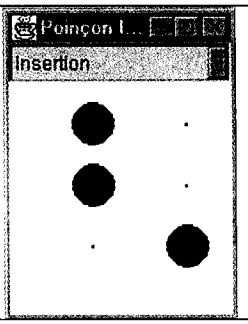
3.2.3 The Punch and the Dictionary Tools

The MfB program allows users to create Braile scores directly in the Braille Editor window, whether they know Braille musical notation or not. A user who has a command of Braille musical notation can use the standard ASCII keyboard to enter the score in the capture zone, but he can also use the *punch tool.*

THE PUNCH TOOL

The Punch Tool was created for blind users who normally employ a "Perkins" keyboard to write musical scores. On this keyboard the S, D, and F keys are mapped to Dots 3, 2, and 1; the J, K, and L keys are mapped to Dots 4, 5, and 6 [that is, the six-dot cell is traversed column by column, starting at the left].

Figure 5. The Punch Tool.



In conformance to the Perkins keyboard, the "S-D-F, J-K-L" keys respectively correspond to the "3-2-1, 4-5-6" Braille dots. The user has to press simultaneously the desired keys to write a Braille character. When he releases the keys, the Braille character is inserted and the Editor waits for the input of a new character.

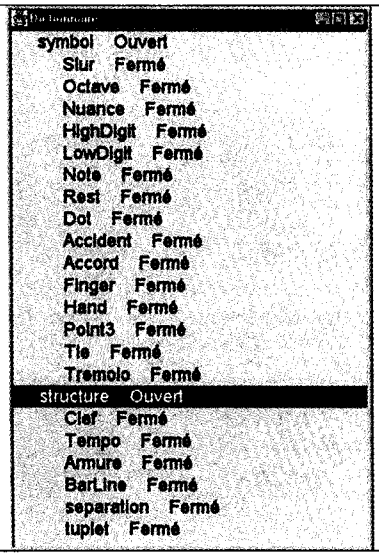
THE DICTIONARY TOOL

This Dictionary Tool was developed for creating a Braille score without any knowledge of Braille musical notation. The dictionary uses the same navigation system as the NIFF Interpreter (double-click to develop nodes) and is composed of two main parts:

- *structure node*, in which the user can find musical structures such as clef, key signature, and tempo;
- *symbol node*, in which notes, rests, and chords can be found.

This is a menu-driven tool and is illustrated in Figure 6.

Figure 6. The Dictionary Tool.



To insert a musical item, the user has to find it in the tree structure of the dictionary and double-click on it. The insertion mode is the contextual insertion by default. Contextual insertion means that the rules of the *New International Manual of Braille Music Notation* are taken into account automatically.

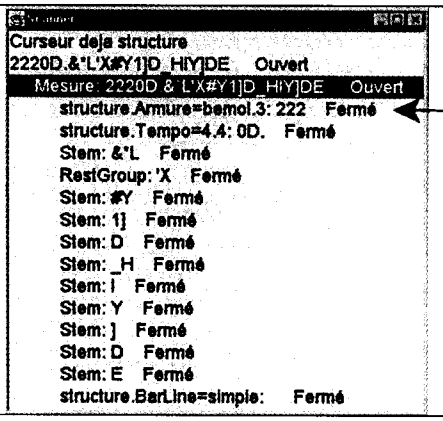
If we take, for example, the first measure of the Bach score part, in order to write this score with the "Dictionary," a sighted person will begin by inserting the clef, then the key signature and the tempo.

In Braille musical notation, the order of these musical items at the beginning of a score is different. With the contextual insertion, this problem is solved—no matter how the sighted user enters these three items, they will always be ordered correctly (by the Braille musical rules).

THE SCANNER: A CONTROL TOOL

This tool was created to allow a textual reading of events in a Braille musical score. In Figure 7 object names are in French; the categories to which they belong are in English.

Figure 7. The Scanner Tool.



Here the key signature (3 flats) is described in French in this expression: "structure.Armure=bemol.3: 222".

This description means that the expression is a musical structure called *armure* (key signature), composed of three *bemols* (flats), and the corresponding ASCII character for this key signature is "222".

3.3 Conclusion

MfB is very easy to use. After testing it with sighted as well as blind persons, we decided to work on the accessibility for blind users.

We chose the Screen Reader *JAWS* (Henter Joyce) and the vocal synthesis program *Eloquence*. We have solved the problem of the multi-windowing (one element is active at a given time *t*) which caused confusion for the screen reader.

Concerning navigation through the tree structure, a blind user can use the *directional* (navigation) and *enter* (development of a node) keys. We decided to add the text *ouvert* (opened) and *fermé* (closed) after each node to describe its status. The score in the capture zone of the Braille Editor is in a graphic form (Braille dots) which cannot be read by the Screen Reader, so we created a new zone called the scanner zone (See Figure 4: the Braille Editor), which is a textual echo of the graphic form.

This software is a prototype. It needs some improvement, especially in the transcription domain (we have to add to the specific rules). In the future, we would like to create a specific tool for the learning of Braille

musical notation and integrate a MIDI output of the created or transcribed score.

References

European Commission Green Paper: Living and Working in the Information Society. People First. Com(96)389.

Krolick, Bettye, ed. (1996). *New International Manual of Braille Musical Notation.* World Blind Union. ISBN 90-9009269-2.

NIFF 6.A, Notation Interchange File Format, 1995. *mistral.ere.umontreal.ca/~belkina/NIFF.doc.html*

PLAY project, TIDE project no. 3212, deliverable D2.2.4, "Musical Standards," Didier Langolff (Paul Sabatier University), Steve Mounce (Bradford University), Anna Liisa Salminen (STAKES), Monique Truquet (Paul Sabatier University), European Commission DG XIII, Brussels, 1997.

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