EXPRESSIVE NOTATION PACKAGE - AN OVERVIEW

Mika Kuuskankare
DocMus
Sibelius Academy
mkuuskan@siba.fi

Mikael Laurson
CMT
Sibelius Academy
laurson@siba.fi

ABSTRACT

The purpose of this paper is to give the reader a concise overview of Expressive Notation Package 2.0 (henceforward ENP). ENP is music notation program that belongs to a family of music and sound related software packages developed at Sibelius Academy in Finland. ENP has been used in various research projects during the past several years.

1. OVERVIEW

ENP [3] is a music notation program that has been developed in order to meet the requirements of computer aided composition, music analysis and virtual instrument control. ENP is intended to represent Western musical notation from 17th century onward including 20th century notation. ENP is not a full featured music typesetting program. It is, however, designed to produce automatic, reasonable musical typesetting according to the common practices [12]. ENP output should not generally require adjustments made by the user.

There are also some other non-commercial LISP-based programs that are aimed at representing complex musical data such as Common Music Notation [13], PatchWork's Rhythm-Editor [6] and the musical editors in OpenMusic [1].

ENP is programmed with LispWorks ANSI Common Lisp by Xanalys. LispWorks, in turn, is a Lisp implementation that is source code compatible across Windows, Linux, Mac OS X and UNIX platforms. ENP uses the OpenGL API for graphical output. OpenGL is a widely used 3D graphics library that is fast and portable with implementations in all of the aforementioned operating systems. It is equally suitable for developing 2D or 3D interactive applications.

At the time of writing the development version of ENP runs in Mac OS X. However, some preliminary attempts have been made to port it to Windows.

Some of the key concepts behind ENP are:

1) ENP can be used to represent a wide range of notational styles.

2) ENP has a mouse driven user interface that relies on direct editing, i.e., almost every notational object can be edited in the score with synchronized visual feedback.

3) ENP provides access to its notational data structures, thus it can be controlled algorithmically. The user can inspect and modify the properties of the notational objects (e.g., time, pitch, duration).

4) ENP provides a rich library of standard and user-definable expressions. They range from standard articulation markings to fully interactive multi-purpose graphical expressions.

Also, ENP is used as a notational front end in a visual programming language called PWGL [8]. PWGL is a combination of several complex software packages build on top of Common Lisp. The components include a rule-based programming language (PWConstraints [6]), and a sound synthesis program (PWSynth [7, 10]). These closely integrated software packages can be used to further analyze, construct, and modify the musical data contained by ENP scores.

This rest of the paper is divided into four main sections. Section 2 contains an outline of the notational structures and expression scheme of ENP. Section 3 explains the principles of ENP user interface. Section 4, in turn, covers two separate issues, ENP-score-notation and scripting. Finally, in the last Section, we discuss the future development of ENP.

2. MUSIC REPRESENTATION

ENP supports a number of notational styles (e.g., mensural or non-mensural notation, frame notation, tape notation). In the next two subsections we discuss the two fundamental notational styles supported by ENP: mensural and non-mensural notation. The third subsection, in turn, discusses frame notation in brief.

2.1. Mensural Notation

An ENP score is built out of hierarchical object structures. Typically, a mensurally notated ENP score consists of a list of parts, a part consists of a list of voices, a voice consists of a list of measures, and a measure consists of a list of beats. Beats, in turn, can contain either other beats (to create complex nested rhythms) or chords. Finally, a chord contains a list of notes. Notes and chords can also contain information about any additional attributes, such as expressions (see Section 2.4).

Next we give an example of the score structure of ENP in mensural context. Figure 1 contains a simple musical excerpt consisting of two eight-notes and a quarter-note. Figure 2, in turn, displays the corresponding ENP score as a tree structure revealing the score hierarchy.

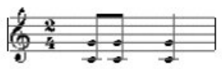


Figure 1. A simple ENP score.

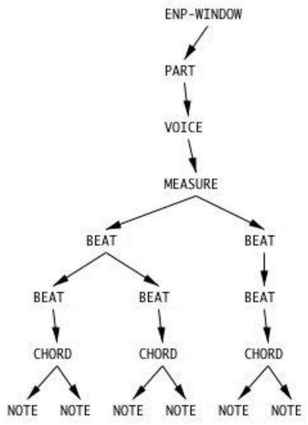


Figure 2. The hierarchical structure of the ENP score (shown in Figure 1) displayed as a tree structure.

2.2. Non-mensural Notation

The difference in non-mensural notation, when compared to mensural notation, is that internally a voice contains a list of chords with absolute start-times and durations. The non-mensural notation can be used, for example, when writing contemporary proportional or 'time-notation' (à la Berio). Figure 3 gives an example of an ENP score that is written using non-mensural notation. The example includes also some expressions and special note-heads, etc.

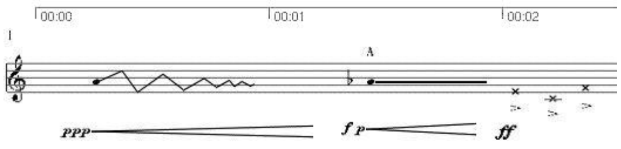


Figure 3. An ENP score written using non-mensural notation. The absolute time (minutes:seconds) is shown above the staff.

2.3. Frame Notation

Apart from the two fundamental notational styles described above ENP can also display musical material using frame notation. Frame notation can be used in both mensural and non-mensural context and it is indicated in the score by enclosing a group of pitches or gestures inside a rectangle. These are then to be played either in a random order or according to a given rhythm. Two examples of the frame notation can be found in the Appendix.

2.4. Expression Scheme

ENP provides a predefined set of both standard and non-standard notational attributes, called ENP-expressions [4]. Standard ENP-expressions include, for example, staccatos, slurs, playing styles, etc. Non-standard expressions, in turn, include groups and score-BPF (see Figure 4 for examples of both). Score-BPF, for instance, is a multipurpose graphical object that can represent breakpoint functions as a part of a musical texture. The user can also create new expressions through inheritance.

All ENP-expressions are dynamic, i.e, they adjust their visual appearance according to their notational context. Expressions are also editable directly in the score thus editing a score-BPF or adjusting the slope of a slur is both straightforward and interactive.

Figure 4 gives an example of a score containing various ENP-expressions.

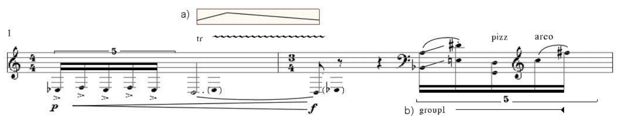


Figure 4. A score containing a collection of different ENP-expressions, such as a score-BPF (a) and a group (b).

2.4.1. Compound Expressions

One example of an ENP-expression that inherits properties from another ENP-expression is the crescendo expression. As it inherits form the score-BPF the user can draw an arbitrary break-point function to express the increase and decrease of loudness as a function of time. The break-point function can be edited directly in the score:

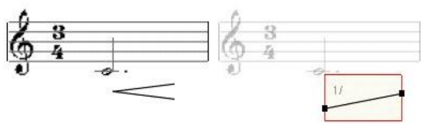


Figure 5. The crescendo expression (left) inherits form the score-BPF and thus contains an editable break-point function (right). At the right the crescendo expression is displayed in an editable state (i.e., after the user has double-clicked it). Notice also how the musical material remains in the background while editing.

2.4.2. Instrument Sensitive Expressions

An expression may have a different graphical representation depending on the instrument it is written for. In ENP these kind of expressions are called instrument sensitive expressions [8].

Let us look at an example where we use an expression that designates that a passage of music is to be played on a specific string. In a guitar part it is usually written differently than, for example, in a violin part (Figure 6). In case the user would change the instrument of a part, all the instrument sensitive expressions in it would adjust their appearance accordingly.

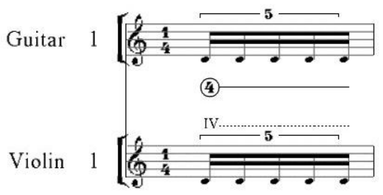


Figure 6. An instrument sensitive expression attached both to a guitar part and a violin part. The expression has a different outlook in both parts depending on the instrument in question.

3. GUI

Now we discuss the graphical user interface of ENP. First, we introduce the general ideas behind editing in ENP. After that we explain the selection mechanism and also take a look at how rhythms are edited.

ENP has a graphical user interface that allows any musical object to be edited directly in the score [3]. Generally all editing operations provide a synchronized visual feedback for the user.

3.1. Editing

There are two edit modes in ENP:

1) General edit mode in which the user can, for example, enter and edit pitch information, add and edit expressions, etc.

2) Rhythm edit mode. This mode is intended for inputting and editing the rhythmic structures and timing information.

The underlying idea behind the ENP user interface is to allow the user to access the information contained by a digital musical score as straightforward as possible. As a general guideline every musical object of any complexity can be edited directly in the score. This reduces the need for any external dialogs or editors. Every musical object in the score reacts to a set of operations:

1) Dragging is the primary way to edit notational objects including transposing, repositioning, shape adjusting and even editing the rhythm.

2) Zooming is mainly used to scale the score view or to adjust the spacing of the score.

3) Panning can be used, for example, to displace objects or to reposition the score view.

The effect of the editing operation depends on the type of the object in question and usually on the direction of the mouse movement. For example, notes and chords can be transposed (vertical drag) or displaced in time (horizontal drag). The expressions can be repositioned or reshaped, etc.

Most operations that can not be performed by dragging are handled with the help of context sensitive menus. They can be used to control various notational attributes like note-head shapes, enharmonic spelling or beaming information. Context sensitive menus are also used to add expressions, make changes to page layout, and so on.

3.2. Selecting

ENP supports all the widely used ways of selecting objects. It also supports discontinuous selection in any complexity (i.e., notational objects of any kind across voices and parts) and any form (e.g., including sweep selection). The novelty in ENP is not the way the actual selection is made but rather the way the selection and especially the multiple selection is managed [3].

There are two ways of making a selection in ENP:

1) Single selection (e.g., clicking to an object).

2) Multiple selection (e.g., sweep selection, extended selection).

In (1) it is unambiguous to determine which object is affected by the consequent editing operations. In (2), however, there may be different kinds of objects in the selection and hence there must be a mechanism to determine which subset of the selection is active at the time. For this purpose we introduce a new concept, selection filtering.

3.2.1. Selection Filtering

There are two ways that selection filtering can take place in ENP:

1) Explicitly, when the user applies a suitable filter. Currently supported filters of this kind are: note-, chord-, beat-, and measure-filter. This approach is usually used when applying some keyboard shortcuts to a selection.

2) Automatically, in consequence of some editing operations. This occurs when editing objects with the mouse or using context sensitive menus. In this case the eventual editing operations affect only to those selected objects that are of the same kind as the one the user is manipulating with the mouse.

It is to be noted, however, that selection filtering does not collapse the original selection; all the objects remain selected after editing. This is especially useful when the user has to perform a number of operations to different subsets of the selected notational objects.

3.3. Rhythm Edit Mode

There are two basic operations to modify the beat structures:

1) The user can divide an existing beat into arbitrary number of sub-beats. This can be done by using keyboard shortcuts or context sensitive menus.

2) The user can change the proportional duration of any beat. This can be done by using keyboard shortcuts, context sensitive menus or mouse.

Typically, the numeric keys are used to indicate the corresponding numeric values. When the user types a number, the proportional durations of the selected beats are changed. When the user types a number while holding down the shift key, the selected beats are divided into corresponding number of sub-beats. The user can also drag a beat at any level to change its proportional duration (Figure 7). This can also be used to change a note to a rest. As can bee seen in Figure 7 there is also some additional information drawn along with the standard notation. Each beat level in the hierarchy is brought out by drawing a thick line along with a number referring to its proportional duration. The line serves both as an editable handle to the beat and also as a visual indication of the extent of the beat.

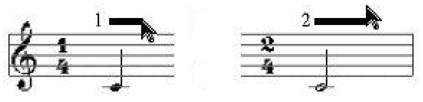


Figure 7. Editing rhythm in ENP. The user has dragged the beat handle of a quarter-note (left) to change it to a half-note (right).

4. ENP-SCORE-NOTATION AND SCRIPTING

In this section we discuss ENP-score-notation and also describe how ENP can be scripted.

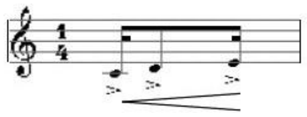
4.1. ENP-score-notation

ENP allows to describe scores in a special text format called ENP-score-notation [9]. This approach is akin to the XML based MusicXML format [2] or the LaTeX flavored LilyPond [11] where the user can describe the elements of the score rather than the layout itself. The layout is then handled either by the software itself (as it is the case with LilyPond) or with another software capable of importing the format in question. [1]

Although ENP scores are saved in a text format it is not, however, very easily readable, especially for the untrained. The ENP-score-notation is offered as an intermediate step between the score and the elaborate file format. By using the ENP-score-notation a score can be converted into a more human readable form. ENP-score-notation can also easily be converted back to an ENP score. This way practically all the user definable properties of an ENP score are accessible and definable.

The structure of ENP-score-notation reflects the score structure described in Section 2. The syntax is similar to the LISP list syntax. Every level in the hierarchy is collected into a list.

Next we give a relatively simple example of an ENP score and its counterpart written in ENP-score-notation (Figure 8). The ENP-score-notation example shown here is generated automatically by exporting the score in ENP-score-notation format.



```
(((((1
     ((1
       :NOTES
       (60)
       :EXPRESSIONS
       (:ACCENT
        (:CRESCENDO/290019720
         :USER-POSITION-Y-CORRECTION
         -0.4)))
      (2
       :NOTES
       (62)
       :EXPRESSIONS
       (:ACCENT
        :CRESCENDO/290019720))
      (1
       :NOTES
       (64)
       :EXPRESSIONS
       (:ACCENT
        :CRESCENDO/290019720))))))
  :STAFF :TREBLE-STAFF))
```

Figure 8. A simple ENP score (above) and its equivalent written in ENP-score-notation (below).

4.2. Intelligent Scripting

Most professional music notation programs provide a scripting language and/or plug-in interface that allows the user to modify the notational information contained by the score. In Sibelius, for example, the user can write scripts by using the built-in scripting language called ManuScript. In Finale, on the other hand, there is an interface that allows the user to write plug-ins using C++. LilyPond provides a plug-in functionality by its built-in Scheme interpreter.

In a music notation program a script could typically be used to apply a certain articulation pattern to a passage of music or to recalculate the enharmonic identity of selected notes, etc.

ENP Script [5] is a scripting language that is derived from the pattern-matching language of PWConstraints [6]. PWConstraints, in turn, is a general-purpose rule-based programming language. The use of the PWConstraints pattern-matching language as the basis of ENP scripting offers several advantages:

[1] In case of MusicXML there are several commercial and non-commercial music notation packages that can import MusicXML, including Igor Engraver, Sibelius, Finale, etc.

1) Complex musical patterns can easily be defined with the help of a pattern-matching syntax.

2) The syntax of the pattern-matching language is compact and powerful and easy to learn.

3) PWConstraints contains rich knowledge about the melodic, rhythmic and harmonic properties of the score that can be accessed by the script.

4) There is no need to write any control structures (e.g., loops) because PWConstraints provides the basic ability to map through the notational objects in the score.

Without going into details of the concepts behind rule based languages, and PWConstraints in particular, we give an example of an ENP script. In the example we use a script to insert a repeating articulation pattern to a passage of sixteenth notes (the resulting score is shown in Figure 9). The benefits of this kind of a script are naturally more obvious when the musical passage in question is very long.

The script is defined as follows:

```
(* ?1 ?2 ?3 ?4
   (?if (when (downbeat? ?1)
          (add-expression 'slur ?1 ?2)
          (add-expression 'staccato ?3)
          (add-expression 'staccato ?4))))
```

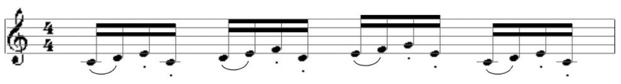


Figure 9. An articulation pattern is applied to passage of music with the help of a script.

5. FUTURE DEVELOPMENTS

There are still many improvements and features planned into the future. ENP is far from finished and continues to be an active field of improvement. For example, there is currently no visual synchronization between the mensural and non-mensural notation. This is a visualization problem that should be addressed.

Also, one of the main tasks will be to add appropriate import/export filters in ENP. Ability to export at least MusicXML or LilyPond formats is planned and the Enigma Transportable Format (ETF) support is under consideration.

There have also been some preliminary attempts to incorporate a Find feature in ENP that would allow to find and display notational objects in the score according to multiple search criteria.

6. ACKNOWLEDGMENTS

The work of Mikael Laurson has been financed by the Academy of Finland.

7. REFERENCES

[1] Gerard Assayag, Camillo Rueda, Mikael Laurson, Carlos Agon, and Olivier Delerue. Computer Assisted Composition at IRCAM: From PatchWork to OpenMusic. *Computer Music Journal*, 23(3):59–72, Fall 1999.

[2] M. Good and G. Actor. Using MusicXML for File Interchange. In *Third International Conference on WEB Delivering of Music*, page 153, Los Alamitos, CA, September 2003. IEEE Press.

[3] Mika Kuuskankare and Mikael Laurson. ENP2.0 A Music Notation Program Implemented in Common Lisp and OpenGL. In *Proceedings of International Computer Music Conference*, pages 463–466, Gothenburg, Sweden, September 2002.

[4] Mika Kuuskankare and Mikael Laurson. ENP-Expressions, Score-BPF as a Case Study. In *Proceedings of International Computer Music Conference*, pages 103–106, Singapore, 2003.

[5] Mika Kuuskankare and Mikael Laurson. Intelligent Scripting in ENP using PWConstraints. In *Proceedings of International Computer Music Conference*, 2004. Accepted for publication.

[6] Mikael Laurson. *PATCHWORK: A Visual Programming Language and some Musical Applications*. Studia musica no.6, Sibelius Academy, Helsinki, 1996.

[7] Mikael Laurson and Mika Kuuskankare. PWSynth: A Lisp-based Bridge between Computer Assisted Composition and Sound Synthesis. In *Proceedings of the International Computer Music Conference*, pages 127–130, Havana, Cuba, September 2001.

[8] Mikael Laurson and Mika Kuuskankare. PWGL: A Novel Visual Language based on Common Lisp, CLOS and OpenGL. In *Proceedings of International Computer Music Conference*, pages 142–145, Gothenburg, Sweden, September 2002.

[9] Mikael Laurson and Mika Kuuskankare. From RTM-notation to ENP-score-notation. In *Journées d'Informatique Musicale*, Montbéliard, France, 2003.

[10] Mikael Laurson and Vesa Norilo. RECENT DEVELOPMENTS IN PWSYNTH. In *Proceedings of DAFx 2003*, pages 69–72, London, England, September 2003.

[11] Han-Wen Nienhuys and Jan Nieuwenhuizen. LilyPond, a system for automated music engraving. In *XIV Colloquium on Musical Informatics (XIV CIM 2003)*, Firenze, Italy, May 2003.

[12] Gradner Read. *Music Notation*. Victor Gollancz Ltd., 1982.

[13] Bill Schottstaedt. Comon Music Notation. In *Beyond MIDI, The Handbook of Musical Codes*. MIT Press, Cambridge, Massachusetts, 1997.

A. APPENDIX

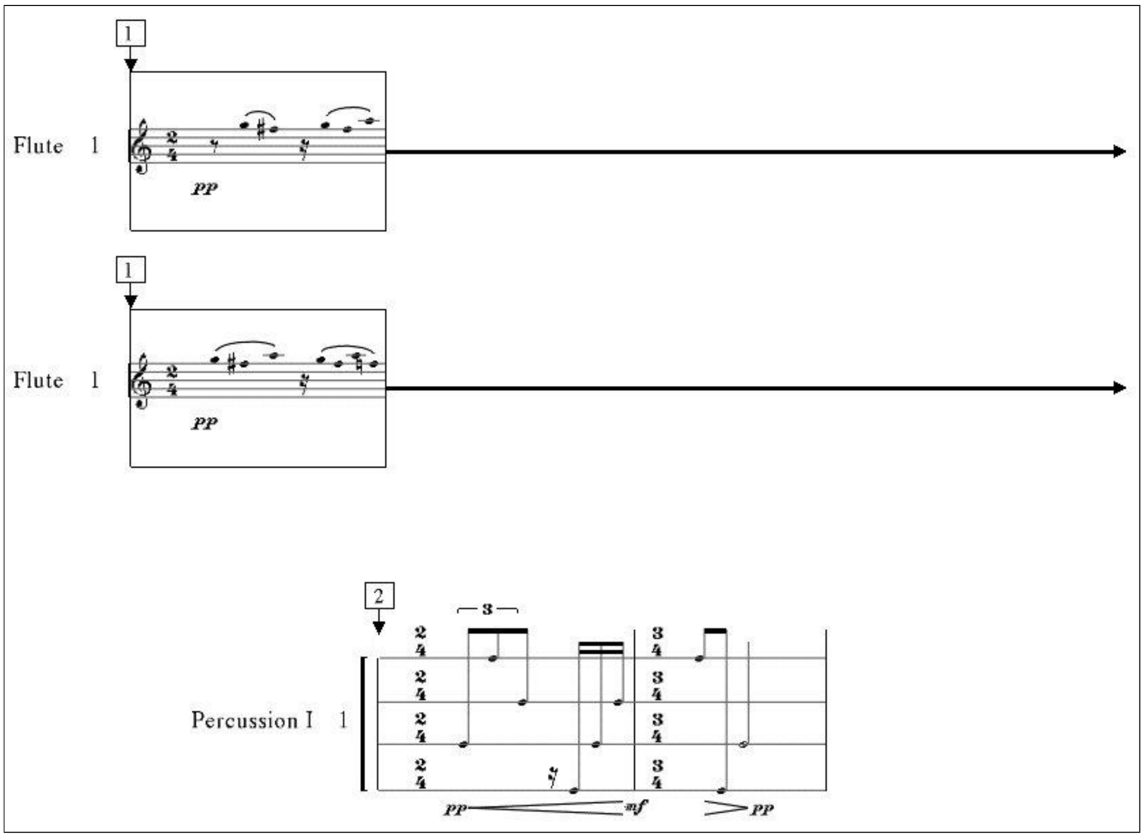


Figure 10. An example of the frame notation in ENP. All the frames (the two flute frames and the percussion frame) can be freely dragged both vertically and horizontally (in time).

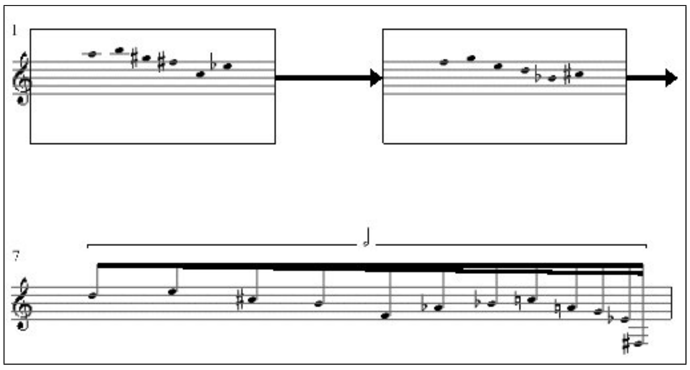


Figure 11. An example of the frame notation in mensural context. There is also a special accelerando-beat in the last measure.