# A generic method to design pen-based systems for structured document composition: Development of a musical score editor

Sébastien Macé, Éric Anquetil, Bertrand Coüasnon
IRISA – INSA de Rennes
Campus Universitaire de Beaulieu
35042, Rennes Cedex, France
*{sebastien.mace@irisa.fr, eric.anquetil@irisa.fr, bertrand.couasnon@irisa.fr}*

**We present in this paper a new generic method to recognize highly structured documents in the context of pen-based edition. The interpretation of the user's strokes is realised progressively, directly while the user composes his document. This approach is based on a formalism making it possible to model, externalised from the system, the composition rules of the corresponding document. This knowledge is exploited by a handwritten stroke parser, which, according to the structural context of the analysed element, can drive the recognizers, interpret it and display its corresponding symbol on the screen. As our system interprets the strokes progressively, it takes into account the human-computer interaction: the user can validate or reject an answer of the parser. Finally, a framework for pen-based applications proposes the basic editing possibilities. The originality of this approach is its genericity, which facilitates and speeds up the development of pen-based applications. We present a particular context to which the method has been applied, the composition of musical scores. In order to realise a system as usable and ergonomic as possible, this application has been designed in collaboration with professional musicians.**

*human-computer interaction, pen-based interaction, structured documents, on-line recognition, generic method, formalisation of knowledge.*

## 1. INTRODUCTION

Today, computer systems based on an interaction with a pen, such as smartphones, pocket PC or tablet PC, are subject to a strong expansion. The use of a stylus is indeed very intuitive, because it reproduces the metaphor of the "paper-pen", media everybody knows. It makes it possible to use mobile computer systems in the way we do with a sheet of paper, drawing on a touch screen; such an interaction can increase the application user-friendliness and offer an access to these new technologies to a larger public. Figure 1 presents an example of a tablet PC with such an interface: the user can edit music on a traditional way thanks to the use of a stylus. This musical score editing application has been developed using the generic method we present in this paper.

**FIGURE 1:** Example of tablet PC with a pen-based interface: the user interacts with the system with a pen.

The interpretation of handwritten drawings realised with the pen on the touch screen, which we call on-line recognition, is a complex problem in the context of highly structured documents. Indeed, such documents are constituted of many elements of various natures. Moreover a same drawing can have different meanings according to the context in which it has been realised. As a consequence it is necessary to take into account knowledge of the document that is composed in order to interpret it. Concerned documents are of various natures: electronic figures, plans, sketches, mathematical formulas, tables, diagrams, musical scores, *etc*.

The interpretation of an on-line document can be either lazy, *i.e.* occurring only when explicitly requested by the user, either eager (or "on-the-fly"), *i.e.* occurring directly while the user draws[1][2]. The main advantage of the lazy

interpretation is that all the information for the analysis of the document is available, but the interaction with the user can not be exploited.
On the contrary, when the analysis is done directly while the user is drawing, which is the context of the approach we present, it is possible to exploit all the possibilities of the pen-based interaction. Nevertheless the recognition system has to deal with partial documents and as a consequence with more ambiguities. Many systems have been developed in order to recognize a given nature of on-line structured documents, such as musical scores[3][4][5], mathematical formulas[6][7], *etc.* Such systems with eager recognition are often very constraining for the user; for example, in the context of musical score editing, a widely use principle is to constrain the user to another way to compose his document[3][4]: he has in particular to learn a new alphabet to write musical symbols, and doesn't compose his document the way he does on paper. The advantage of such an approach is to increase the efficiency of the system, because the new alphabet is chosen so as to reduce the possible ambiguities. In our approach, we want to keep an editing as natural as possible and avoid the introduction of specific gestures. It is then necessary to exploit, as much as possible, structural information and the human-computer interaction.
Moreover, the existing systems are mostly *ad hoc* and can not be generalised to any kind of structured document. Yet, it turns out that the needs are the same, such as driving the recognition process and taking into account the relative positions of the document elements and the interaction with the user. The approach we present in this article is based on a generic method that can, given this specific knowledge concerning the composition rules of a document, *i.e.* the way and the order the user realises its elements, generate the corresponding pen-based system. This approach has various advantages: it facilitates and speeds up the development of the application, allowing the programmer to focus, in collaboration with specialists of the domain of the application, on other aspects of pen-based interaction, such as usability and user-friendliness. Moreover, the maintenance of the application is simplified.
As far as we know, there is no generic approach for an eager recognition of structured on-line documents. Some systems have been developed in optical recognition to model the structure of off-line documents (scanned documents). We can mention the DMOS (Description and Modification of Segmentation) approach[8] for off-line document analysis that makes it possible to describe the parsed structure thanks to a bi-dimensional grammar.

In this paper, the generic approach is exploited to design a pen-based musical score editor. This paper focuses on this application to expose the principles of the presented method. Other example domain will be considered in later publications to demonstrate the real generic aspect of the methodology.

The second section describes the approach we propose in order to recognize on-line documents directly while they are being composed. In the third section, we focus on two particular aspects of this method, which are a formalism to describe the composition of documents and the specific parser to which it is associated. In the forth section, we expose how the system deals with ambiguities, and in particular how it takes into account the interaction with the user to realise its analysis. The fifth section gives a particular example of application, a pen-based musical score editor, on which we are working with professional musicians. Finally, we summarize our future works.

## 2. PRESENTATION OF THE APPROACH

Before presenting the architecture of the system in section 2.2, we expose the basic concepts associated to the modelling of on-line composition of structured document, which are:

- the management of the chronological information;
- the representation of the document spatial structure;
- the recognition process driven by the structure analysis;
- the pen-based human-computer interaction.

These generic concepts will be illustrated on the example of music composition.

### 2.1 Basic concepts associated to the modelling of on-line composition of structured document

*2.1.1 Management of the chronological information*

As we want to recognize the document directly while it is composed, it is necessary for our system to be able to deal with incomplete documents. A way to deal with them is to model chronological information.
We want to model which element can or must exist before another one. Figure 2 presents the composition of a musical note, composed of a head, a stem and a flat and the corresponding “composition trees”, which represent the symbols that can be realised in consequence of the existence of another one. The figure shows two different sequences, either beginning by the head (figure 2(1)), either beginning by the stem (figure 2(2)); on the contrary, in these sequences the flat can only be drawn once the head to which it is associated exists.

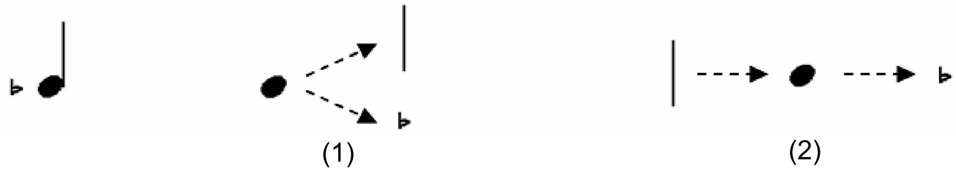

**FIGURE 2:** Composition trees associated to two chronologically different ways to compose a note.

It is necessary to take into account these composition rules in order not to constrain the user to a unique way to compose his document.
It must be also possible to begin the drawing of an element, draw others, and come back to finish the first one anytime later. Figure 3 represents this: the user draws a head of note, and then draws another one, to which he adds a stem; he can then come back to the first head to add it a stem.

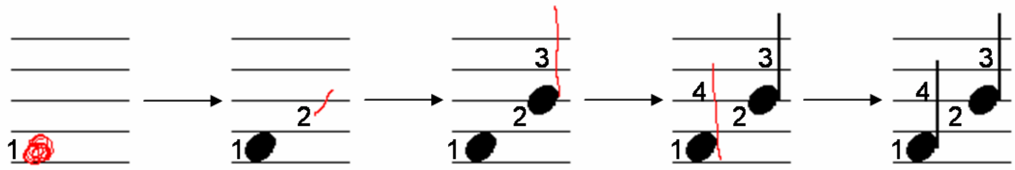

**FIGURE 3:** Drawing of two heads with their associated stem; the numbers represent the order of composition of the symbols.

*2.1.2 Representation of the document spatial structure*
In order to interpret the drawings of the user, it is necessary to exploit structural knowledge about the edited document. Indeed, we know, as it is represented in figure 4, that a musical note is constituted of a head and a stem that can be for example upward on its right; it is also possible to associate an alteration to a head, located on its left (in particular a flat as it is represented in the figure). A composition tree is represented with the associated areas, corresponding to the case when the composition of the note starts with the head.

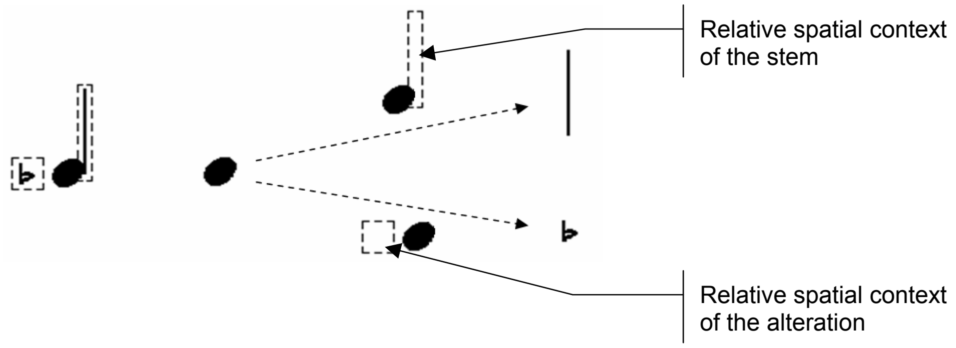


**FIGURE 4:** Example of spatial structure of musical symbols and corresponding composition tree.

This structural knowledge must be exploited in the analysis of the document.

*2.1.3 Recognition process driven by the structure analysis*
We want to enable rich pen-based input. As the edited documents contain elements of different natures, it is necessary to use several recognizers, each able to recognize a family of symbols. Indeed, it is not possible to have a unique recognizer for all the musical symbols, because the more symbols a recognizer must interpret the less efficient and the less robust it is.
Moreover, as a symbol can have several interpretations according to the structural context in which it has been drawn, it must be possible to choose the ones it is pertinent to call. For example, when the user realises a drawing on the left of a note, as it is presented in figure 5, it is pertinent to try to recognize an alteration. We represent this knowledge in the corresponding composition tree: to each area, we associate a recognizer dedicated to a specific family of symbols.

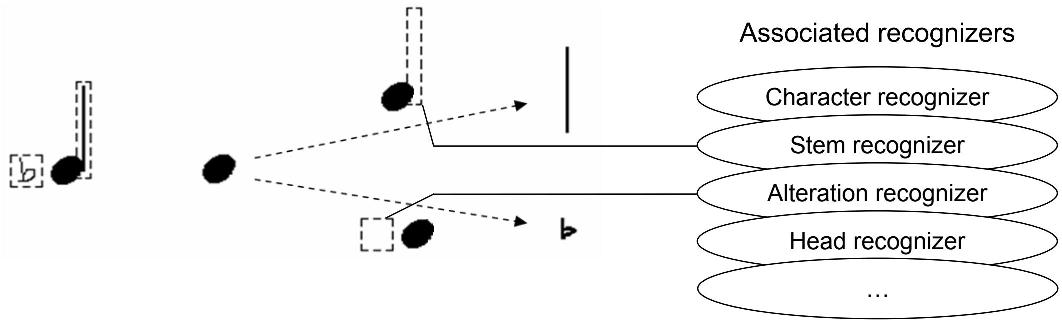


**FIGURE 5:** Drawing of a flat, on the left of a note: activation of the alteration recognizer.

As a consequence, it is necessary that the analysis of the structure of the document drives the recognition of the drawings, in order to enable rich pen-based input.

*2.1.4 Pen-based human-computer interaction*
The system must propose some classical editing gestures, in order the user to move the elements of his document, delete them, zoom in or out of the document, save it and reload it, *etc.* These gestures are independent of the nature of the document and as a consequence are generic.

As the system works with an eager recognition, it transforms the strokes of the user into their corresponding symbolic representation directly while the document is being composed. This means that the user can see the answers of the system. It is necessary to exploit the possible pen-based interaction for user-based error repair to validate implicitly the interpretations or correct explicitly the errors.

In the following sections, we present how the proposed approach deals with all these concepts.

### 2.2 Presentation of the main components of the system

The proposed system, illustrated by the figure 6, is constituted of three main components: a framework for pen-based interaction, a formalism to describe the composition of documents, and a parser of handwritten strokes.

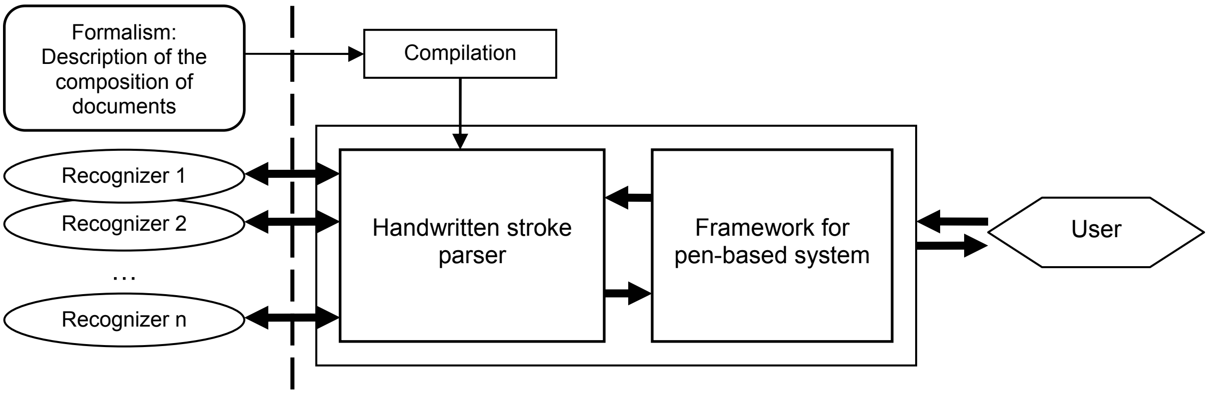


**FIGURE 6:** Architecture of the system.

#### *2.2.1 A framework for pen-based interaction*

The first component of the system is a framework that already contains all the graphical functions needed in a pen-based system: it is independent of the application.

The framework deals with the user interface, the display of the document and basic interaction gestures. It is possible to select graphical elements only using the pen by a gesture that surrounds them. Once selected, the elements can be translated to another part of the document, by pointing to one of them and moving the pen. A translation out of the editing window is interpreted as a deletion of the selected elements.

The framework also proposes some of the usual editing possibilities, like undoing the last element, zooming in or out, saving and loading the document; all of these can be done using the pen.

#### *2.2.2 A Formalism to describe the composition of documents*

A second component of the system is a formalism that describes the composition of documents. It models chronological information and composition rules, for example which element can or must be realised before another one. It also represents physical information, such as spatial structure of the document. It describes the recognition technique for a rich pen-based input, based on a structural analysis.

This formalisation of the knowledge is externalised from the system, which means it can be easily modified, without having to look at the rest of the code. Once compiled, the formalism can be directly exploited in the system we have developed.

#### *2.2.3 A parser of handwritten strokes*

The last component of the system is a handwritten stroke parser which is able to deal with an incomplete document. Thanks to the use of the knowledge modelled with the formalism, it analyses the strokes progressively, directly while the user composes his documents. It is able to call the pertinent handwritten drawing recognizers (for example handwritten character recognizer, geometrical shape recognizer, alteration recognizer, *etc.*) thanks to the editing context, and then create the new contexts that will help recognising the following strokes.

This parser is generic which means it doesn't need to be adapted when the knowledge is changed. As a consequence, in order to realise a pen-based system thanks to our approach, it is necessary to describe the composition of the document with the formalism, realise the specific handwritten drawing recognizers and indicate the visual aspect of the corresponding interpreted symbols.

In this paper, we focus on the formalism and on the parser to which it is associated: we present them more in details in next section.

## 3. PRESENTATION OF THE FORMALISM AND ITS EXPLOITATION BY THE PARSER

This section presents in parallel the formalism we propose in order to model the composition of documents and the way this knowledge is exploited by the associated parser. The formalism is an extension of the context-free grammatical one, making it possible to represent the composition and the interpretation of bi-dimensional documents. As the system analyses a partial document, the rules are exploited on a very different way than the usual one. To illustrate the possibilities of our formalism, we present examples of music composition knowledge modelling, but the approach is generic and can by applied to another kind of structured documents.

### 3.1 Formalism based on context-free grammars

*3.1.1 Basic description of the principles of the formalism*

The formalism we propose is an extension of the classical context-free grammatical one (type-2 grammar according to the Chomsky's hierarchy), which defines a language (in our case a document) thanks to the use of rules of the form

```
A → γ.
```

with A a non-terminal and γ a string of terminals and non-terminals.

We use the same notation and vocabulary, such as axiom, terminal, non-terminal and production rule. We add the concatenation symbol, represented by a comma, which makes the chronological information explicit in a rule of the form

```
A → B , C.
```

which means that, to apply the non-terminal A, it is necessary to be able to apply B. Once B is applied, the element A is created, and it will be possible, later, in the continuation of the composition of the document, to recognize C, *i.e.* to continue the element A. As a consequence, this rule implies that in order C to exist thanks to this rule, B must already be present.

For example, the modelling of the composition of musical note, constituted of a stem, a head and a flat, drawn in that order (like it is presented on figure 2(2)) is defined by the rule

```
Note → Stem , Head , Flat.
```

*3.1.2 Principle of the exploitation of the formalism in the parser*

It is not possible to exploit our formalism in the same way as a traditional grammar one; indeed, it would imply to compose the document exactly in a sequence described by the formalism. For example, it wouldn't be possible to begin a symbol (*i.e.* a non-terminal) without finishing it before beginning another one. We have seen in section 2.1.1 that this constraint is not acceptable.

As a consequence, it is necessary to redefine the way the information is exploited in the associated parser. The principle is that several rules can be begun, waiting to be completed in the continuation of the composition of the document. Thus, when a stroke is realised, we can try:

- on the one hand the new production rules than can be applied, *i.e.* which can begin with the last stroke;
- on the other hand the production rules that have been begun with the previous elements, but that are not finished and that can be applied, *i.e.* which can continue with the last stroke.

### 3.2 Extension of grammars

In this section, we present the extensions to the classical context-free grammar formalism in order to model the specific aspects of on-line composition of documents.

*3.2.1 Using drawing recognizers*

*a) Description of recognition process driven by the structure analysis*

We have seen that it is necessary to specify the calls of the recognizers. In the formalism, a terminal is assimilated to a call to a recognizer. Thus, a terminal which corresponds to the call to the recognizer X is represented as follows:

```
Recognizer(X).
```

For example, if we want to recognize a musical alteration, we can write the terminal:

```
Recognizer(Alterations).
```

This way, it is possible to choose which symbol family recognizers to call given the structural context.

Sometimes, we do not want to interpret a drawing as any of the symbols a recognizer can interpret, but as a subset of them. As it is impossible to have a dedicated recognizer for each of the possible subsets of symbols that the user can draw in a given context, we want to use the more general recognizers which are already able to translate them. As a consequence, we propose to enable to add a second parameter to the call of recognizers, which corresponds to the awaited interpretation Y:

```
Recognizer(X, Y).
```

For example, if we want to recognize a flat, which is a particular alteration, we can write:

```
Recognizer(Alterations, Flat).
```

*b) Principle of the recognition process driven by the structure analysis in the parser*

This knowledge is exploited in the same way than in the classical grammar formalism: as a call to a recognizer is assimilated to a terminal, in order it to match with the drawing, the recognizer X must be able to translate it into its corresponding meaning. If a second parameter, corresponding to the awaited translation, is present, the answer of the recognizer must match with it. On the example we presented in previous section, a flat will only be created if the alteration recognizer interprets the drawing as a flat.

If none of the applicable recognizers is able to interpret the drawing, it is lost and disappears for the screen: indeed we believe that, rather than displaying an interpretation that has a high probability of being wrong and letting the user delete it, it is better not to display it at all.

Note that on the contrary of classical grammatical formalism, a drawing can match with several different terminals.

*3.2.2 Modelling of spatial structure of the document*
We have seen how it is possible to specify which recognizer to call on a drawing. The interest of such a possibility is to choose, given the context, the recognizers it is pertinent to call. We now present the modelling of such bi-dimensional structural contexts, impossible with the classical grammatical one, and their exploitation in the associated parser.

*a) Representation of the spatial structure of the document*
The principle is to determine, relatively to a particular element, an area in which a given element can be drawn. As a consequence, the creation of the first element implies the creation of a structural context that didn't exist previously. For this purpose, we propose a position operator, which is an area of the document, represented in our formalism with the operator `[]` and the following syntax, both inspired from the DMOS method[8]:

`A , [Pos]B.`

This rule means that once the element `A` exists, it will be possible to find, in the continuation of the composition of the document, an element B which will be located at the position `Pos` of the element `A`.
In order to be able to locate more than one position operator relatively to one element, we introduce a factorisation operator, represented by a colon. Thus, the rule

`A , [Pos1]B : [Pos2]C.`

is equivalent to

`A , [Pos1]B` and `A , [Pos2]C.`

For example, the composition of a note beginning by the head with a stem uprising on its right and an alteration on its left (like it is presented on figure 2(1)) is modelled as follows:

`Note → Head , [right, uprising]Stem : [left]Alteration.`

One of the advantages of our formalism is that it is possible to create as many position operators as necessary, making it possible to model complex spatial structures.

*b) Exploitation of this spatial structure in the parser*
Once an element is interpreted, it can create new structural contexts represented by position operators, waiting for the elements located in them to exist. In fact, the production rules wait on these empty position operators. As a consequence, the parser knows where the rules have stopped and the corresponding operators. The mechanism of the analysis consists in finding, among the position operators waiting for their content to be realised, which one corresponds to the stroke. Then, the parser tries to apply or continue the corresponding rule, by creating the new structural context, checking if the corresponding element already exists and otherwise blocking on it.
Thanks to the framework to which the parser is associated, it is possible to display the location of the position operators to the user, in order to have reference points and to know where to draw the elements. Figure 7 presents the areas corresponding to the position operators of the rule we presented in the previous section: on the left of the head, an alteration can be drawn, and on the right and uprising, a stem can be drawn.

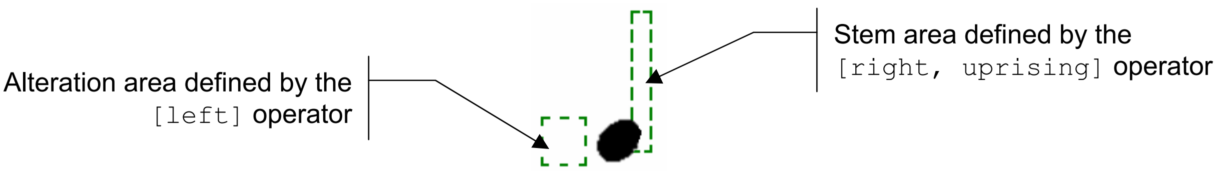


**FIGURE 7:** Areas corresponding to the position operators.

A user can even decide to make these areas disappear, because they are not essential to the edition. Thus, once he has got accustomed to the application, he can change from a "novice mode", in which all the free zones are visible, to an "expert mode", in which none of them appear.

## 4. DEALING WITH AMBIGUITIES

We present in this section how the system deals with ambiguities, which correspond to the situations in which several interpretations of the stroke are possible.

### 4.1 User Validation

The system interprets the strokes progressively, directly while the user is composing his document, and the results of the analysis are displayed on the screen. We can then exploit the existing human-computer interaction and integrate the user to the recognition process: he can validate or reject the interpretations:

- if, after the display of the interpretation of his previous stroke, the user continues the composition of his document, he implicitly indicates that it is correct;
- if, on the contrary the user doesn't agree with the interpretation, he can delete his last stroke with a suppression gesture.

As a consequence of this validation, it is not necessary for the parser to backtrack and so to modify a decision made previously, because the user agreed with it; this is not possible with a lazy recognition. Thus, the recognition system is more robust, because the user limits the ambiguities, and also faster, because it just has to interpret the last stroke without having to backtrack.
In spite of the exploitation of this implicit validation, ambiguities can still exist for a stroke. We now present how the parser deals with them.

### 4.2 Depth of position operators

Of course, it is possible, for a given stroke, to correspond to various position operators, and then maybe have various possible interpretations. In order to deal with this ambiguity, we use a priority value for the position operators containing the stroke, indicating its “depth” and defined as follows:

- a position operator specified in the axiom of the formalism has a priority of value 0;
- when a stroke, realised in a position operator of priority n, is interpreted, it implies the creation of new position operators; their priority is n+1.

This way, we can define an order on the position operators, from the highest priority to the lowest priority. As a consequence, we analyse the position operators from the more probable, *i.e.* the one with the highest priority, to the less probable; once a position operator is tested and makes it possible to recognize the stroke, it is not necessary to try the less probable ones.

### 4.3 Waiting for more information

We have seen a way to discriminate position operators according to their priority value, but two operators of the same priority can be able to recognize the stroke. Then, it is impossible to make a decision immediately. As a consequence, the parser waits for the continuation of the composition of the document, in order to have enough information. Concretely, there is a particular display to the user, who is therefore aware that the system waits for complementary strokes before coming to a decision.

## 5. RESULTS

The methodology has been applied on the development of a pen-based musical score editor. This system makes it possible, for the musician, to write music the same way as if it was on a sheet of paper. In order to develop an application as user-friendly and as adapted to musician needs as possible, the system is developed in collaboration with professional musicians from the MIAC (Music and Image: Analysis and Creation) of the Rennes 2 University[9].
The experimental study of usability and acceptability of the musical score editor is not yet finished. Nevertheless several demonstrations have been realized among musicians. The first feedbacks are really satisfactory and a major part of the testers was enthusiastic. All agree about the innovation of this application and its facility of use, even if the prototype does not yet integrate all the musical symbols.

### 5.1 Description of the pen-based musical score editor

Figure 8 presents eight screenshots of the pen-based musical score editor prototype, corresponding to the composition of a document: the strokes are replaced with their corresponding symbols progressively. The fifth screenshot corresponds to a “novice mode”: it shows the areas associated to the position operators.

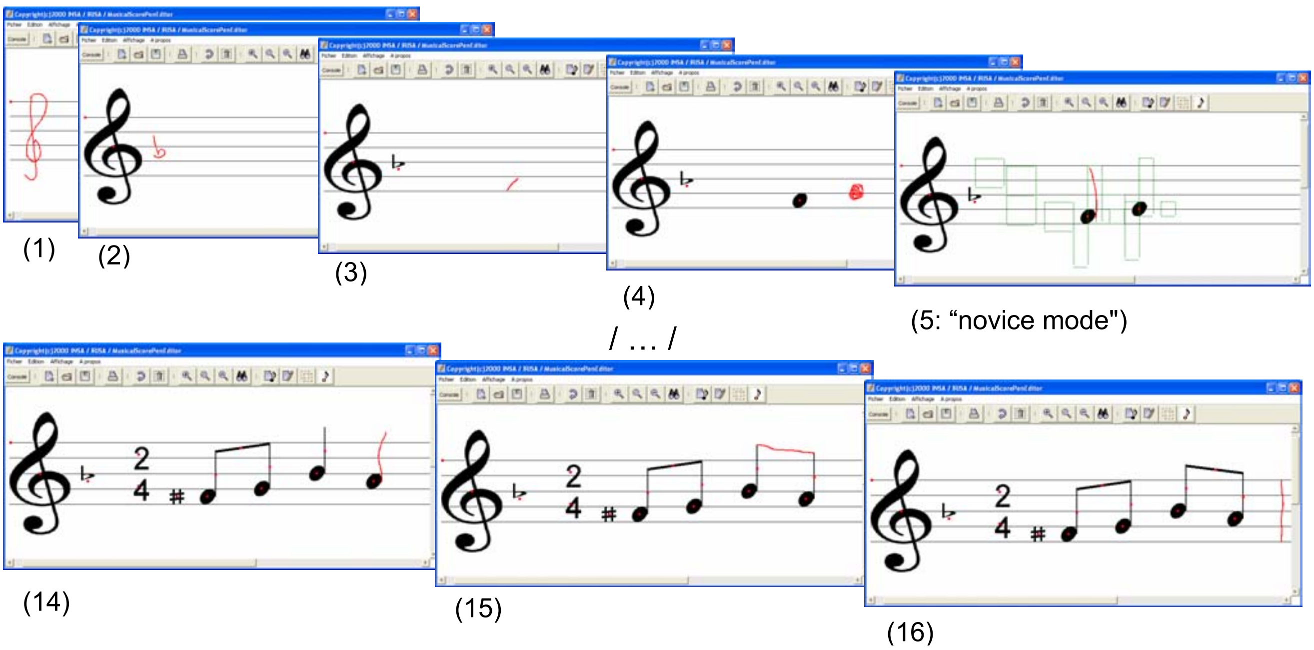


**FIGURE 8:** Screenshots of the pen-based musical score editor prototype.

Even though it is still just a prototype, the editor already includes a lot of musical symbols. It is possible to draw bass or treble keys, whole, half and quarter notes. Notes can have a durational dot. All the accidentals are available for the notes (flat, double flat, sharp, double sharp or natural) and for the clefs (flats or sharps). Line bars can be realised on the staff. All silences are already available.

### 5.2 Development of the pen-based musical score editor using the generic method

In collaboration with professional musicians, who can tell us the way a musical score is being drawn, we write the composition rules with the formalism. This paper does not present the whole composition rules used to describe

the composition of a musical score, but a few rules it contains. For instance, below is reported the description that makes it possible to compose the head and the stem in any order, like presented in figure 2, and to begin a note and another one and to come back to the first one, like presented in figure 3. Respectively on the left and on the right of a head, an alteration and a durational dot can be drawn.
The corresponding description is as follows:

```
Staff → [inside]Note*.
Note → Head , [right, uprising]Stem :
                [left]Alteration :
                [right]DurationalDot.
Note → Stem , [left, bottom]Head , [left]Alteration :
                                   [right]DurationalDot.
Head → Recognizer(Heads, FilledNote).
Stem → Recognizer(Shapes, VerticalSegment).
Alteration → Recognizer(Alterations).
DurationalDot → Recognizer(Shapes, Dot).
```

After the definition of the composition rules, it is necessary to realise the specific handwritten drawing family recognizers (such as musical head recognizer, shape recognizer or alteration recognizer). For handwritten characters and digit letters, we exploit the RESIFCar recognizer[10]. For other symbols, we develop basic recognizers based on empirical heuristics or based on Neural Networks[11]. Finally, we have to specify the visual aspect of the interpreted symbols. Then, the corresponding application can be generated: the generic parser can analyse the document during its composition using the modelled knowledge and the framework deals with editing gestures and the display.

## 6. CONCLUSION

This paper presents a new generic method to develop pen-based systems that recognize the user's strokes progressively, while they are realised. This approach is based on a formalism, extension of the context-free grammatical one, making it possible to describe knowledge about the composition of the document, such as chronology, spatial structure and call of the handwritten drawing recognizers. This information is exploited by a specific parser, which analyses the strokes progressively. This interpretation, occurring directly during the composition process, makes it possible to exploit the human-computer interaction: the user can validate or correct the interpretation made by the system.
This method has been used in particular to develop a prototype of pen-based musical score editor. In order to make the system as intuitive and as user-friendly as possible, we work in collaboration with professional musicians. Future works will aim at enriching the formalism in order to describe more complex knowledge, such as the composition of "composite" symbols, constituted of more than one stroke. To validate the genericity of the presented approach, we are going to work on the development of other pen-based application systems, such as mathematical formulas or electronic figures.


## ACKNOWLEDGEMENTS

We thank Bruno Bossis, from the MIAC (Music and Image: Analysis and Creation) of Rennes 2 University, who collaborates to the pen-based musical score editor design.


REFERENCES


[1] D. Blostein, E. Lank, A. Rose and R. Zanibbi (2002) User Interfaces for On-Line Diagram Recognition. *Proceedings of the 4th International Workshop on Graphics Recognition Algorithms and Applications (GREC 2002)*, pp. 92-103.
[2] M. Nakagawa, K. Machii, N. Kato and T. Souya (1993) Lazy recognition as a principle of pen interfaces. Proceedings of the conference companion on Human factors in computing systems (INTERACT '93 and CHI '93), pp. 89 - 90 .
[3] A. Forsberg, M. Dieterich and R. Zeleznik. (1998) The Music Notepad. *ACM Symposium on User Interface Software & Technologie*, pp. 203-210.
[4] J. Anstice, T. Bell, A. Cockburn and M. Setchell. (1996) The Design of a Pen-Based Musical Input System. *Proceedings of the 6th Australian Conference on Computer-Human Interaction (OZCHI 1996)*, pp. 260-267.
[5] H. Miyao and M. Maruyama. (2004) An Online Handwritten Music Score Recognition System. *Proceedings of the 17th International Conference on Pattern Recognition (ICPR 2004)*, pp. 461-464.
[6] K. Toyozumi, K. Mori, Y. Suenaga and T. Suzuki. (2001) A System for Real-time Recognition of Handwritten Mathematical Formulas. *Proceedings of the 6th International Conference on Document Analysis and Recognition (ICDAR 2001)*, pp. 1059-1064.
[7] W. Thimbleby. (2004) A novel pen-based calculator and its evaluation. *Proceedings of the third Nordic conference on Human-computer interaction (NordiCHI 2004)*, pp. 445-448.
[8] B. Couäsnon. (2001) DMOS: A generic document recognition method, application to an automatic generator of musical scores, mathematical formulae and table recognition systems. *Proceedings of the 6th International Conference on Document Analysis and Recognition (ICDAR 2001)*, pp. 215-220.
[9] S. Macé, E. Anquetil, E. Garrivier and B. Bossis. (2005) A Pen-Based Musical Score Editor. *To be published in the International Computer Music Conference (ICMC 2005).*
[10] E. Anquetil and H. Bouchereau. (2002) Integration of an On-Line Handwriting Recognition System in a Smart Phone Device. *Proceedings of the 16th IAPR International Conference on Pattern Recognition (ICPR 2002)*, pp. 192-195.
[11] C. M. Bishop. (1995) *Neural Networks for Pattern Recognition*. Oxford University Press, Oxford.