AscoGraph: A User Interface for Sequencing and Score Following for Interactive Music

Thomas Coffy, Jean-Louis Giavitto, Arshia Cont

AscoGraph: A User Interface for Sequencing and Score Following for Interactive Music

Thomas Coffy
IRCAM - INRIA, Paris, France
coffy@ircam.fr

Arshia Cont
IRCAM
cont@ircam.fr

Jean-Louis Giavitto
IRCAM / CNRS
giavitto@ircam.fr

ABSTRACT

Composing interactive music using score following, requires tight coordination and several round trips between many tools to write the score to follow, and to author the electronic actions, and assess their synchronisation. In addition, at performance time, the score-following must be monitored to ensure proper execution. Unifying composition and performance phases provides composers and electronic music designers a global approach with the best of both worlds. The *AscoGraph* interface is an incarnation of the need for unifying authorship and performance for *Antescofo*'s score following and reactive engines. *AscoGraph* provides high precision tools for textual and graphical authorship of complex dynamic interactive music pieces with intuitive and sustainable design. This article presents the design, challenges and integration of *AscoGraph* for edition and representation of mixed music scores using *Antescofo* by affording complex hierarchical constructions such as nested blocks and polyphony in electronic actions while maintaining readability and ease of authorship.

1. INTRODUCTION

Automatic Score Following has been an active line of research and development among composers and performers of Interactive Music since decades. In such setups, an automatic score following system comprises realtime listening machine that in reaction to recognition of events in a score from a human performer launch necessary computer music actions. The process of composing and performing such pieces require dedicated interfaces for both the *authorship* and *performance* of mixed music pieces considering expressivity for two important primary elements that are *Time* and *Interaction*. The study of such interfaces is as vital as the design of the real-time system itself since it underlies expressivity and reliability of the final result.

Dedicated interfaces for editing and visualising augmented scores for Score Following systems has been long studied in the computer music literature. Though not directly related to score following, the *SCRIVA* system [1] implemented a score editing tool for electronic sounds destined for real-time performance and generation of pre-notated events. On similar lines, the *Animal* system in [2] is another example where the notion of *authoring of time* through complex hierarchical graphical objects is explicitly addressed. Despite addressing authorship of time in both systems, they lacked explicit considerations for *interaction control* which was shifted entirely towards the performer rather than the designer/composer. The problem of sequencing mixed with score following was first approached by Puckette in *EXPLODE* [3] where actions are primarily triggered by events and much of temporal expressivity is lost to attain real-time performance. *NoteAbilityPro* [4] is another recent example of integrated score environment capable of communicating with the *Antescofo* score-following engine [5] amongst others. Despite all these efforts, our community still lacks integrated environments that address needs for a smooth workflow between *composition* and *performance* phases of interactive music pieces.

This paper introduces *AscoGraph* (Figure 1), a new interface for editing, visualising and simulating augmented scores specifically designed for *Antescofo* [6] musical scores. *AscoGraph* provides visual interfaces for authoring of multiple time semantics for electronic actions, polyphonic authorship for concurrent electronic voices, and facility tools for users composing and performing with complex scores. Electronic and instrumental scores in *AscoGraph* are tightly coupled. *AscoGraph* emphasises editing for electronic actions but provides necessary tools for importing and visualisation of instrumental scores from common platforms. Unifying heterogeneous timed events in *AscoGraph* is possible by constant translation of event/action time-stamps to a *relative musical* timeline. This assumption makes visualisation possible for performance situations where tempo is dynamically detected and constantly changing. Furthermore, Simulation Tools are available to visualise an *ideal* and *flattened* performance especially for dynamically programmed electronic actions and complex scores.

We begin this paper by describing the integration of *AscoGraph* within common existing workflows for composition and performance of interactive music. Section 3 describes *AscoGraph* environment in terms of its software ecosystem for interactive computer music. Section 4 details specific concepts for score representation and edition in *AscoGraph* as well as facility tools for composers such as simulation capabilities, followed by future perspectives and concluding remarks.

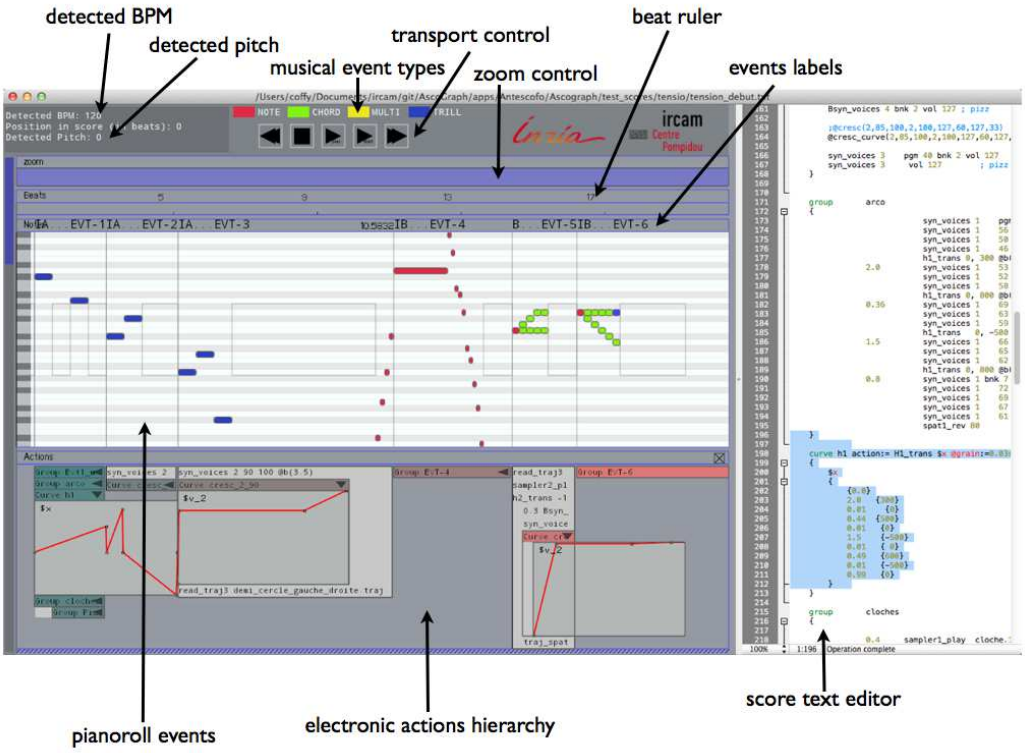

Figure 1. AscoGraph general UI.

2. COMPOSITION WORKFLOW

2.1 Composition phase

Composers often prepare instrumental sections of their pieces using *Finale*, *Sibelius*, or other score engraving system. An important feature for *AscoGraph* is the score import capabilities from MusicXML or MIDI formats, which make the first steps of composing an interactive piece much smoother. MusicXML is a score format which can be exported from main classical musical notation editors. Writing electronic parts is made less difficult with *AscoGraph* and *Antescofo* new reactive engine since version 0.5, which provides new ways of writing interaction, based on clear musical hierarchies such as groups, loops, curves and more.

Visualisation capabilities of *AscoGraph* enters the composition workflow as a huge gain since *Antescofo*'s score syntax is text based, where a list of events, such as notes, chords, trills or multi (glissandi), with associated pitch and duration describe the score as it should be followed by the score follower next to their electronic actions. *AscoGraph* distinguishes itself strongly from other softwares such as *NoteAbility* in its sole focus on Electronic Action authorship rather than music engraving. The integrated import facility of *AscoGraph* is solely there to support visual feedback on the instrumental score, for which we believe there are numerous stable software. . Importing existing instrumental score is done by *drag'n'drop* where it is directly converted to *Antescofo* notes notation. For automatic MIDI accompaniment, it is also possible to convert MIDI file to electronic actions (MIDI notes on/off) by pressing CMD during import.

AscoGraph engine shares the same score parsing routines with *Antescofo* core, so the validity of the score is checked on saving while editing in *AscoGraph*, with detailed parsing errors handling.

2.2 Performance phase

AscoGraph is strongly connected with *Antescofo* core object (using OSC over UDP) : when a score is edited and modified it is automatically reloaded in *Antescofo*, and on the other hand, when *Antescofo* follows a score (during a concert or rehearsal) both graphical and textual view of the score will scroll and show the current position of *Antescofo*.

3. ENVIRONMENT

AscoGraph emphasises editing the reactive score. One key design choice while creating *AscoGraph* editor was to allow both textual and graphical editors, representing different views of the same score. "Time of composition" and "time of performance" need proper dedicated environments [7]: *AscoGraph* and *Antescofo* combines both representations.
The general mechanisms which articulates each component are discussed in this section.

3.1 Link to Antescofo

The *Antescofo* engine is embedded in a Max/MSP or PureData external (dynamically linked) object. The *AscoGraph* editor can be opened by double-clicking on the *Antescofo*

object. It is then launched as a standalone application, and it loads the score if one was loaded in *Antescofo*. Communication with *Antescofo* uses OpenSoundControl protocol (OSC) [8] in both ways as shown in Figure 2.

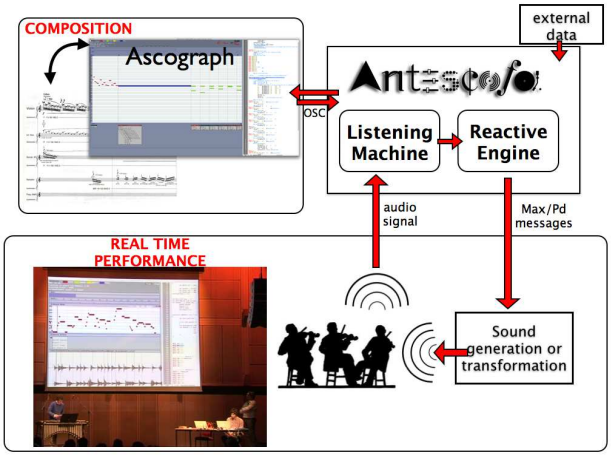

Figure 2. Antescofo and AscoGraph general diagram.

- From *Antescofo* : detected tempo, detected pitch, score position in beats relative to tempo, score loading commands are displayed in *AscoGraph*.
- To *Antescofo* : User interface buttons in *AscoGraph* generally send OSC command to *antescofo* object, in order to control transport: play sequence, start score following, stop, next event, previous event, load a score, etc.

Thus, *AscoGraph* can display accurately, at low CPU cost, visual feedback to help composers during rehearsals or performances.

4. SCORE REPRESENTATION AND EDITION

4.1 Dual representation

Antescofo score is a text file containing events (chord, notes, trills, glissandi, ...) to follow, synchronisation strategies on how to trigger actions, and electronic actions (the reactive language). Visually, the application is divided in two parts, on the left a graphical representation of the score, using a timeline with tracks view. On the right, a text editor (with syntax coloring and auto-completion) of the score is displayed. Both views are editable and synchronised on saving. Special objects such as "curves", are graphically editable : they are used to provide high-level variable automation facilities like breakpoints functions (BPF) with more than 30 interpolations possible types between points, graphically editable.

Due to its programmatical nature, score-following requires a dual representation of interactive scores as text and graphics. Allowing edition in both modes and providing ways to pass from one to the other was one of the challenges around *AscoGraph* realization. When an object is modified (such as a curve) graphically, it displays an "Apply" button, when pressed it replaces the corresponding text in the score with new values. On the opposite, when the text score is modified manually, it is graphically redisplayed when the file is saved.

4.1.1 Graphical score representation

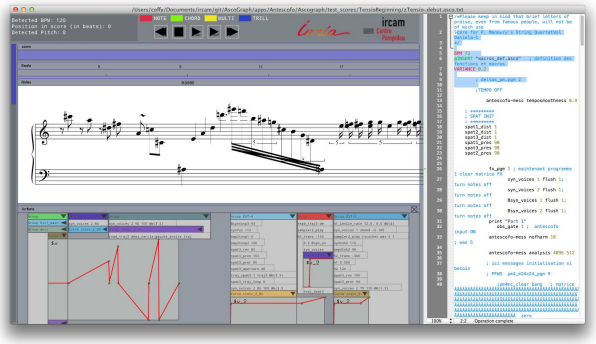

Figure 3. Manoury - Tensio, first measures

Currently, two visualisation modes are available for instrumental section:

- **Pianoroll:** A MIDI-style graphical notation display has been developped to represent easily musical events to be followed by *Antescofo*. Events in *Antescofo* are distinguised in this view by their color. Four types of events are separately recognized by *Antescofo* : individual notes, chords (polyphonic notes), trills (for tremolo, vibrato, etc) and multis (continuous variation of pitch, such as glissandi). Last special kind of musical event is rest note (silence), which are displayed as en empty black frame separating other events.
- **Guido:** A more traditionnal way of displaying western classical music is the staff view based on solfège. For displaying complex scores, the GUIDO Music Notation format [9] has been integrated in *AscoGraph*. When an *Antescofo* score is parsed (when loading or saving), a GUIDO string is dynamically translated from *Antescofo* events notation (cf Figure 3).

For fast navigation in the interactive score, a 2D zoom bar has been introduced intuitive to most users.

4.1.2 Textual score representation

Embedded in each *AscoGraph* window, is a textual editor, based on the Open-Source multi-platform project Scintilla [1] with the following features:

Auto-completion : when dealing with large number of electronic action receivers names, simplifying text input has quickly become a useful feature. An advanced mechanism was imagined with IRCAM's Computer Music Designers' precious help : easing the input of Max receivers names. Many pieces are based on really big Max patches, which can count several thousands receivers, each controlling a distinct parameter of a sound effect, synthesis, etc. To answer this demand, a "namespace getter" has been implemented in the *Antescofo* object, loaded by Max. When

[1] Scintilla - Open-Source editor : `http://scintilla.org/`

asked, this object lists recursively every receivers name in the loaded patch, then sends it by OSC to *Ascograph*. This way, these names are added to a list containing by default every keywords of *Antescofo* langage.
By pressing **Ctrl-Tab** in *AscoGraph* (after a receiver name or a langage keyword prefix is typed), a small window listing completion possibilities is displayed. An other demand was to allow user variables to be added to this completion list: this can be done by sending *add_completion_string* to *Antescofo* object. **Syntax coloring** is supported to help reading the textual score.

Embedded Antescofo parser : The *Antescofo* langage parser is embedded in *AscoGraph*, and provides detailed parsing errors when saving or loading a score.

Actions to Event Bi-directional Sync enables a strong link between both graphical and textual editor. This feature allows smooth transition in-between graphical and textual representations in the score. This feature is of utmost importance since certain hierarchical electronic patterns demand constant go-between the two representations.

Furthermore, it is possible to evaluate a partial score by simply selecting appropriate text in the score and a special **playstring** mechanism which sends the selected score excerpt to *Antescofo*, allowing fast execution of small chunk of code as well as auditioning of parts during composition.

4.2 Writing electronic actions using Antescofo's reactive langage

4.2.1 Action templates

Figure 4. Electronic action templates

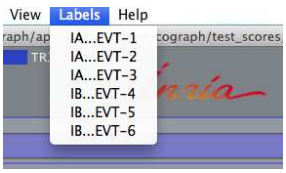

Figure 5. Accessing event markers

The "Create" menu in *AscoGraph* (Figure 4) provides a generic electronic action template insertion mechanism to ease adding electronic actions in an *Antescofo* score. Another very useful menu is "Label": to access directly a specific event marker (Figure 5).

4.2.2 Hierarchical group display

Musical objects in the *Antescofo* electronic score can be defined in nested block hierarchies, and run concurrently (polyphony), with various behaviour (loop, group, curve) and attributes (delays, group delays, variables) etc. *AscoGraph* arranges such complex schemes into ordered blocks with correct vertical alignment with regards to themselves (polyphony among electronic phrases) and with regards to the instrumental score as shown in Figure 3 and 8.

4.2.3 Curves graphical editing

Continuous breakpoint functions can be controled by the *Curve* construct: curves can be seen as control automation (Figure 6). *AscoGraph* provide efficient ways to add,

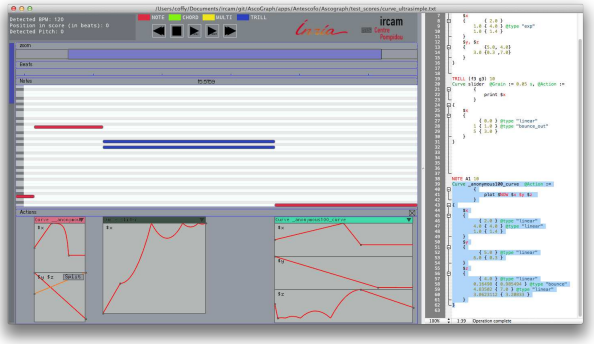

Figure 6. Editing curve constructs

delete, move breakpoints and regenerate dynamically corresponding groups in the text score. By pressing the "Apply" buttons after modifying graphically a curve, *AscoGraph* dumps in the score the new breakpoints values.
One important feature of *AscoGraph* is its graphical breakpoint editing. *Antescofo* provide more than 30 standard interpolation types, from linear to exponential, polynomial (with several orders), etc.

4.3 Simulating a performance

Like any interactive system, *Antescofo* scores can become rapidly complex and the need of simulating a performance has been expressed by several composers and electronic music designers in order to test their compositions and directly see the output.

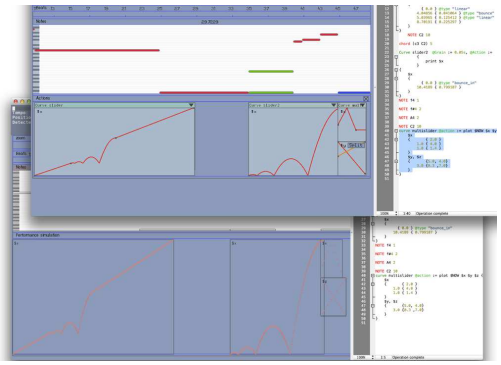

Figure 7. Viewing simulation results: top window is original score, bottom window displays simulated output.

A standalone version of *Antescofo* has been embedded in *AscoGraph* to provide a virtual execution of a score, simulating clocks as fast as possible. This mechanism can be a great help when dealing with *processes* or *whenever* constructs.

5. FUTURE WORKS AND CONCLUSION

The work on *AscoGraph*, since late 2012, has been highly incremental and based on important user feedback deploying the software in various compositional and performative setups. The joint ambition of *AscoGraph* in becoming both a compositional and performance interface for Interactive

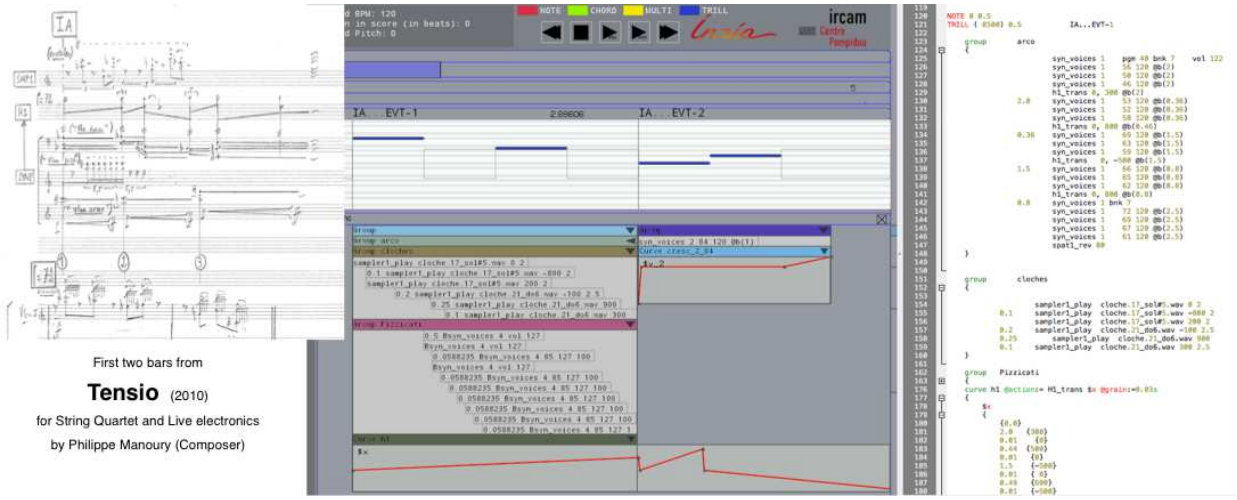

Figure 8. First bars of `Tensio` for String Quartet and live electronics by Philippe Manoury (2010). Left: Hand-written manuscript by composer showing Violin part (below) and electronic actions (top) staves. Right: Antescofo equivalent visualised in AscoGraph

Pieces, has allowed composers and computer musicians to rapidly integrate the software in their process and to start composing at the very onset using the interface. Figure 8 shows an example of such integration.

AscoGraph is released under Open-Source MIT license and has been released publically along with new *Antescofo* architecture since IRCAM Forum in 2013. Due to its modular architecture, *AscoGraph* can be deployed on multiple platforms such as commonly available mobile and tablet architectures.

Future work will focus on the following points amongst others:

- Tracks display for easy message filtering,
- Visualising dynamic processes: control structures which trigger actions not in response to a recognised musical event but to the occurrence of an arbitrary logical condition,
- Specialised editing mode: e.g. dragging audio files with analysis.
- Alternative representations and views for editing.

Acknowledgments

AscoGraph has been created with the help of INRIA, and through work within the INEDIT Project funded by the French National Research Agency (ANR) (ANR-12-CORD-009): http:// inedit.ircam.fr.
AscoGraph is based on ofxTimeline by James George https://github.com/YCAMInterlab/ofxTimeline

6. REFERENCES

[1] W. Buxton, S. Patel, W. Reeves, and R. Baecker, "The evolution of the sssp score-editing tools," *Computer Music Journal*, vol. 3, no. 4, pp. 14–25, 1979.

[2] E. Lindemann, "Animal : A rapid prototyping environment for computer music systems," in *ICMC: International Computer Music Conference*, S. A. et Graham HAIR, Ed., Glasgow, Ecosse, Septembre 1990. [Online]. Available: http://mediatheque.ircam.fr/articles/textes/Lindemann90b

[3] M. Puckette, "Explode: a user interface for sequencing and score following," in *Proceedings, International Computer Music Conference*, 1990, pp. 259–261.

[4] D. Litke and K. Hamel, "A score-based interface for interactive computer music," in *Proceedings, International Computer Music Conference*, 2007.

[5] A. Cont, "Antescofo: Anticipatory Synchronization and Control of Interactive Parameters in Computer Music." in *International Computer Music Conference (ICMC)*, Belfast, Ireland, Aug. 2008, pp. 33–40. [Online]. Available: http://hal.inria.fr/hal-00694803

[6] J. Echeveste, A. Cont, J.-L. Giavitto, and F. Jacquemard, "Operational semantics of a domain specific language for real time musician-computer interaction," *Discrete Event Dynamic Systems*, vol. 23, no. 4, pp. 343–383, Aug. 2013. [Online]. Available: http://hal.inria.fr/hal-00854719

[7] A. Cont, "Modeling Musical Anticipation: From the Time of Music to the Music of Time," Ph.D. dissertation, University of California in San Diego and Université Pierre et Marie Curie - Paris VI, Oct. 2008. [Online]. Available: http://hal.inria.fr/tel-00417565

[8] M. Wright, "Open sound control: an enabling technology for musical networking," *Organised Sound*, vol. 10, no. 3, pp. 193–200, 2005.

[9] D. Fober, S. Letz, and Y. Orlarey, "Open source tools for music representation and notation," in *Proceedings of the first Sound and Music Computing conference-SMC*, vol. 4, 2004, pp. 91–95.