CORRECT AUTOMATIC ACCOMPANIMENT DESPITE MACHINE LISTENING OR HUMAN ERRORS IN ANTESCOFO

Arshia Cont, José Echeveste, Florent Jacquemard, Jean-Louis Giavitto

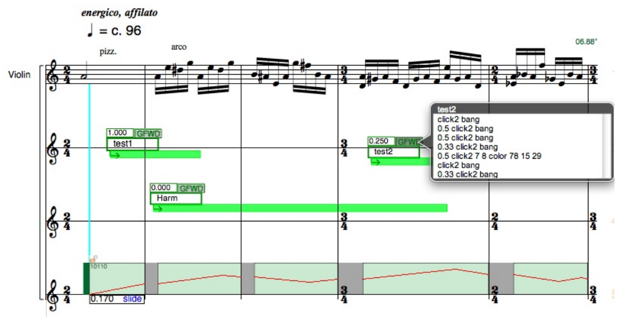
MuSync Team-Project (Ircam, Inria, CNRS)
Ircam-Centre Pompidou, Paris, France.
name.lastname@ircam.fr

ABSTRACT

Automatic accompaniment systems are comprised of realtime score following systems that in reaction to recognition of events in a score from a human performer, launch necessary actions for the accompaniment section. While the realtime detection of score events out of live musicians' performance has been widely studied in the literature, score accompaniment (or the reactive part of the process) has been rarely addressed. In this paper, we expose the respective literature concerning this missing consideration. We show how its explicit design considerations would allow correct accompaniment despite machine listening or human errors introduced during score following, and furthermore how it enables a more elaborate authoring of time and interaction for mixed live electronic pieces.

1. INTRODUCTION

Automatic accompaniment is the act of delegating the interpretation of one or several musical voices to a computer in interaction with a live solo (or ensemble) musician(s). The paradigm was first put through disjointly by Dannenberg and Vercoe in [4, 9] where a computer would provide automatic accompaniment out of incoming symbolic signals (MIDI) from a musician. The most popular form of such systems is the automatic accompaniment of an orchestral recording with that of a soloist in the classical music repertoire (concertos for example) as described in [6]. In a larger context, these systems became popular for interactive computer music repertoire [7], where the association of live musicians with computer generated processes becomes crucial. Figure 1 shows the score of a simple interactive computer music piece where the top staff corresponds to the part for human musician and the lower staves correspond to automatic accompaniment sections that should be run synchronously to the first staff during live performance. Note here that the accompaniment commands depend exclusively on the nature of the computer music process in question. They can be symbolic commands, continuous curves or sequences written relative to the live instrument section of the score. Within this context, the case of automatic accompaniment for classical music repertoire is thus a special case of this larger context where accompaniment commands are replaced by either symbolic MIDI messages or a live rendering of an audio recording of the orchestral part.



```
test2
click2 bang
0.5 click2 bang
0.5 click2 bang
0.33 click2 bang
0.5 click2 7 8 color 78 15 29
click2 bang
0.33 click2 bang
```

Figure 1. Score example for an interactive piece, the top staff contains the *instrumental* section whereas lower staves correspond to specific computer music commands.

The best starting point in the design of an accompaniment system is to observe the human counterpart. Musical accompaniment among musicians is a combined act of *listening* and *coordination* governed by music scores. While human listening plays a crucial role in accompaniment, it is (computationally speaking) fallible. Moreover, actions produced by musicians in an ensemble can contain (perceivable or unperceivable) errors. Despite such inconsistencies, the overall musical output should stay more or less intact and error-free. This is to say that despite the importance of the recognition (or listening) phase in musical accompaniment, the coherence of the overall musical output is to a great extent covered by the ability of human musicians to coordinate and anticipate their actions and adopt the best synchronization strategies in realtime to achieve the best musical output.

Automatic accompaniment systems in general are comprised of two main components as illustrated in figure 2: A score follower that is capable of decoding the live instrument position as well as necessary musical parameters in realtime given the instrumental score (for the top staff of fig 1); and a second component in charge of launching the accompaniment commands synchronously to the live musician (for lower staves in fig 1). In realtime control theory [8], the score follower is an *interactive system* with transactions at its own speed with the environment (usually governed by signal processing and machine learning techniques). Execution speed is formally abstracted to be perceptively close to zero for such processes. The accompaniment block is however a *reactive system*, reacting continuously to the external environment and at the speed imposed by the environment (the human performers).

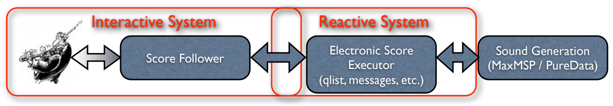

Interactive System
Reactive System
Score Follower
Electronic Score Executor (qlist, messages, etc.)
Sound Generation (MaxMSP / PureData)

Figure 2. Architecture of an automatic accompaniment system comprising of *interactive* and *reactive* systems.

Considering both human and computer accompaniment together, the "healthiness" of the musical output of an accompaniment system requires as much considerations for the coordination/reaction phase as for the recognition phase. The general score following and accompaniment literature has however underestimated the first, focusing on robustness in the listening phase; leaving the action phase to hand-engineered and most often preliminary considerations for the accompaniment actions. This lack of consideration for action coordination would in return reduce the paradigms of interaction between human and computer mediums for interactive pieces, leading to severely simplified programming and authorship for mixed instrumental and live electronic pieces as discussed in [3].

In this paper, we aim to illustrate the importance of the *reactive system* architecture in automatic accompaniment and showing its usefulness within two contexts: (1) the ability to automatically handle machine listening or human errors with no propagation to the accompaniment output; and (2) to enrich the vocabulary of live electronics accompaniment by providing explicit access to the authoring of time and interaction in the interactive computer music repertoire. We therefore assume throughout this paper that an interactive listening system is readily available and focus on handling and authoring accompaniment actions. This paper discusses the integration and employment of the proposed paradigms within the *Antescofo*[1] [1] software coupling both listening and coordination aspects of such systems. In section 2 we discuss previous works that explicitly handle accompaniment actions integrated within a score following systems. We showcase and motivate problems with existing approaches in section 3 and propose an approach to exposed problems by adopting synchronization strategies during authorship and run-time in section 4. We conclude the paper by illustrating several examples of this approach for automatic accompaniment as well as live electronic pieces using *Antescofo*.

2. PREVIOUS WORKS

In this section, we overview two existing and popular systems that handle both machine listening and online accompaniment. For the scope of this paper, we focus on the way automatic accompaniment is transcribed and handled in such systems. Therefore, we do not provide any discussions on the machine listening aspects and do not overview all existing score following systems.

2.1. Music-Plus-One

Music-Plus-One is a musical accompaniment system developed by Christopher Raphael for the classical music repertoire and destined for a soloist in a concerto-like setting. The system is decomposed into three modules: one for a realtime score match using hidden Markov models, a second for coordinating audio playback of the existing accompaniment using a phase-vocoder, and a third for linking the two using a Kalman filter model for tempo adjustments [6]. Whereas the listening and decoding parts of *Music-Plus-One* is well documented, there are no explicit documentation on the authoring and handling of the coordination for the accompaniment part (at least at the time of writing this paper). However, demos of the software are available on the web[2]. Here we aim to study the making of the accompaniment part in order to understand the behavior of the system in an online accompaniment setting. *Music-Plus-One*'s application is destined for the classical music repertoire and concerto-like settings. It has been however rarely used for contemporary interactive music repertoire. Understanding the synchronization and coordination strategies are important for the creation of new pieces for the system. The discussions hereafter are thus by no means a criticism of this approach.

Music-Plus-One uses the following minimal text data to undertake automatic accompaniment: a musical score of the solo part, an audio recording of the orchestral or accompaniment part, a set of trained parameters for the listening model (trained offline and used during realtime detection), and some timing data in charge of associating the solo score to the accompaniment audio for the rendering phase. Among these, the timing data is of outmost importance since it creates the necessary binding between the soloist score and the accompaniment audio that will be employed during realtime rendering. Figure 3 shows an excerpt of *Music-Plus-One* text input for illustrating this point. On the left, is a text description of the soloist music score, in terms of relative position in a measure, MIDI pitch numbers and other necessary information for the listening module. On the right (a separate text file), each line contains the accompaniment audio onset time for the specific event in the solo score. This correspondence is (most probably) obtained by a rigorous segmentation of the accompaniment audio with regards to the solo score, to provide coherent musical phrases to be used during live accompaniment rendering. Note that the coordination points on the right, do not necessarily correspond to the solo score. This is normal since an optimal segmentation of accompaniment onsets should be based on musical phrasing rather than a one-to-one correspondence.

Within the structure described in the above example, it is not clear how *Music-Plus-One* handles machine listening or human errors during live performance. Depending

[1]http://repmus.ircam.fr/antescofo

[2]http://music-plus-one.com/

```
1+0/1 0/1 94 77 68
1+3/8 3/8 94 77 0
1+3/8 3/8 94 79 77
1+13/32 13/32 94 79 0
1+13/32 13/32 94 77 71
1+7/16 7/16 94 77 0
1+7/16 7/16 94 76 69
1+15/32 15/32 94 76 0
1+15/32 15/32 94 77 76
1+1/2 1/2 94 77 0
1+1/2 1/2 94 81 88
1+11/16 11/16 94 81 0
1+11/16 11/16 94 79 72
1+3/4 3/4 94 79 0
1+3/4 3/4 94 82 84
```

```
start_pos = 1+0/1
end_pos = 103+5/8
firstnote = 0
lastnote = 1022
atm_1 + 0/1 197.000 0
atm_1 + 1/4 249.000 0
atm_1 + 1/2 299.000 0
atm_1 + 3/4 345.000 1
atm_2 + 0/1 391.000 0
atm_2 + 1/4 436.000 1
atm_2 + 1/2 476.000 1
atm_2 + 3/4 521.000 1
atm_3 + 0/1 568.000 0
atm_3 + 1/4 611.000 1
atm_3+1/2 653.000 0
```

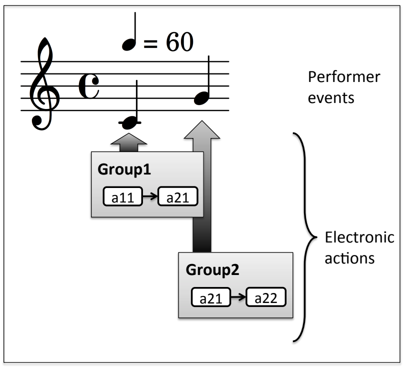
Figure 3. Excerpts of *Music-Plus-One* input. Left is the soloist score, Right is the coordination score. Extract from Beethoven's Romance 2 for violin and orchestra, Op. 50'.

on the musical context, and in presence of machine listening or human errors (missing events for example), one might want to skip an accompaniment phrase or launch it with varying speed. Moreover, preparing such accompaniment/coordination score seems to be a heavy phase in preparing an accompaniment and not very suitable for compositional purposes.

2.2. Antescofo

Antescofo [1] is another polyphonic score following system capable of handling accompaniment actions and integrated within the same system. The listening machine of *Antescofo* is documented in [2] and some aspects of its accompaniment language in [1, 3]. The main musical paradigm addressed by *Antescofo* is that of mixed instrumental and live electronic (or interactive computer music) pieces, where the "accompaniment actions" can range from a simple concerto-like setting to triggering of live electronic processes as common in interactive computer music. *Antescofo* requires no training for its listener and accept a single text as score input.

An *Antescofo* score is composed of both instrumental (soloist) score and the accompaniment actions within one integrated score. Figure 4 shows an example of a simple *Antescofo* score with both instrumental and accompaniment actions. The semantics of the instrumental score allows the construction of complex events such as trills, glissandi, improvisation boxes and also continuous events. The (accompaniment) action semantics is entirely based on *message-passing* and provides constructions for *grouping* of events in order to create polyphonic phrases, as well as loops and continuous trajectories. The timing of actions can be relative (to the score tempo) in floating points or rational numbers, or in absolute time. Graphical representation and authoring of such scores is possible thanks to its integration within the *NoteAbility* score editor. Figure 1 is in fact an illustration of an *Antescofo* score featuring three discrete-sequence groups (green boxes), a continuous trajectory and a symbolic action group (lower staff) as accompaniment actions, all living within a single score framework. In its original text format, composers are also able to create nested hierarchies within electronic phrases (groups inside groups), employ macro expansions and use data flow functionals (see [3] for descriptions).



```
BPM 60

NOTE 60 1.0

 Group1{
   0 action11
   ½ action12
 }

NOTE 67 1.0

 Group2{
   0 action21
   ½ action22
 }
```

Figure 4. Sample score diagram and the corresponding simple *Antescofo* score

The role of *Antescofo* in realtime is to decode the position and tempo of the performer within a synchronous reactive system to best interpret the accompaniment section. Compared to previous approaches in interactive computer music, the accompaniment actions can be described as relative to the performers' tempo and thus dynamically rescheduled in realtime. Each action's starting point is relative to its backward closest event in the instrumental score. The phrasing schemes available in *Antescofo* allow the scope of such electronic phrases to go beyond inter-onsets of the instrumental score.

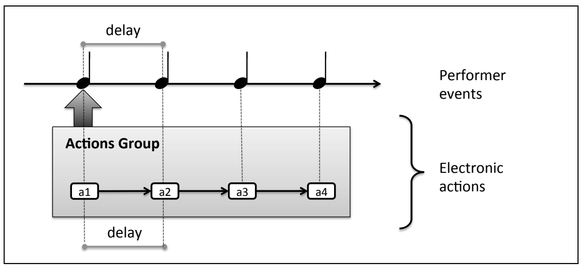
While the ability of authoring parallel phrases (as opposed to segmented and chopped actions) makes the act of authoring more appealing, such timing overlaps create important issues both for realtime coordination of events with the live performer and also their authoring. For example, for a simple concerto-like accompaniment setting, one might want to specify the accompaniment part as one single (and non-chopped and non-segmented) electronic phrase. This appealing tendency creates interesting challenges for synchronization strategies as well as error handling of the virtual accompaniment interpreter which are described in the next section.

3. MOTIVATIONS

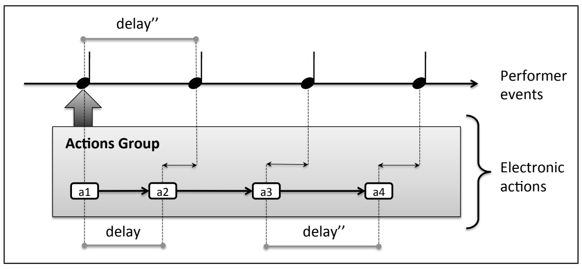
3.1. Synchronizing electronic with live musicians

For compositional purposes, it seems more natural to be able to express accompaniment actions as phrases as opposed to small segments within instrumental note onsets. However, such additions require explicit and dynamic strategies for handling synchrony between accompaniment actions when their scope goes beyond that of its starting instrumental event. The need for such strategies become even more crucial when tempo changes occur after the phrase's launch. Figure 5 attempts to illustrate this within a simple example: Figure 5(a) shows the *ideal perfor-*

mance or how actions and instrumental score is specified to the system. In this example, an accompaniment phrase is launched at the beginning of the first event from the human performer. The accompaniment here is a simple group consisting of four actions that are written parallel to subsequent events of the performer in the original score. In an ideal setting (i.e. correct listening module) the action group is launched synchronous to the onset of the first event. For the rest of the actions however, the synchronization strategy depends on the dynamics of the performance. This is demonstrated in figure 5(b) where the performer hypothetically decelerate the consequent events in her score. In this case, the delays between the actions and their corresponding instrumental event will naturally increase. Such asynchrony is despite correct decoding of tempo from the listening machine. We note however, that this behavior ensures a smooth synchronization with performer tempo changes, but without any guarantee for position synchronicity. Although this behavior is desired in some musical configurations, it seems natural to propose an alternative strategy where the electronic actions would be synchronous to the events detection.



(a) Ideal performance



(b) Slower performance

Figure 5. The effect of tempo-only synchronization for accompaniment phrases. Illustration for different tempi.

3.2. The case of machine listening or human errors

The musical output of an automatic accompaniment system should not solely depend on its listening module or even to the human performer at specific circumstances. This is in analogy to human coordination for ensemble performance: A live music performance output should not be at stake in presence of any error in realtime.

In a live performance situation different errors may be encountered: the listening module could confuse an event with another, miss an event, or produce a false-alarm. Additionally, musicians might introduce performance errors that can affect the accompaniment results. In all cases, we expect not only that the system continues to work, but also that it reacts as musically as possible. Parts but not all of these errors can be handled directly by the listening modules (such as false-alarms and missed events by the performer). The critical safety of the accompaniment part can thus be reduce to handling of missed events (whether missed by the listening module or human performer). The natural question to ask in this case is what the system should do in case of a missed event? Should the associated actions be performed or not? The answer to this question seems more musical than technical: In some automatic accompaniment situations, one might want to dismiss associated actions to a missed event if the scope of those actions does not bypass that of the current event at stake. On the contrary, in many live electronic situations such actions might be initializations for future actions.

This discussion shows that while such considerations can be addressed automatically in special cases, error handling attributes should be first-class citizens in any specification language for automatic accompaniment and interactive computer music pieces.

4. PROPOSED APPROACH

The musician's performance is subject to many variations from the score. There are several ways to adapt to this musical indeterminacy based on specific musical context. The correct synchronization and error handling strategies is at the composer or arranger's discretion. To this end, we propose explicit synchronization and error handling strategies as attributes for the composer to choose, taking into account performance variations and to manage the errors of the musician and the recognition algorithm. In this paper, we provide a verbal description of the proposed approach. The formal and semantical definition of these concept are described in [5].

Accompaniment phrase and loop constructs are used to start a sequence of actions from a trigger event in the instrumental score. Once a sequence of relatively-timed actions is launched, its synchrony by default is governed by dynamic rescheduling following changes of tempo from the musician. However, as seen in section 3.1, knowledge of realtime tempo is not sufficient for precise synchronization with events played by the musician as shown in Figure 5. While this `loose` synchrony is useful in some musical context (loose phrasing of electronics for example), it is not always desirable. For a finer synchronization, we provide the composer the ability to assign a `tight` attribute to a phrase block. If a block is `tight`, its inside actions will be dynamically analyzed to be triggered not only using relative timing but also relative to the nearest event in the future. This new feature evades the composer from segmenting the actions blocks to smaller segments with regards to synchronization points and provide a high-level vision during the compositional phase. A dynamic scheduling approach is adopted to implement the `tight` block behavior. During the execution, the system attempts to synchronize the next action to be launched with the corresponding event using a hybrid strategy employing both tempo and future-event positions.

The problem of error handling, as discussed in section 3.2, boils down to the ability of attributing *scopes* to accompaniment phrases and blocks. Despite their space occupancy in the score, a block is said to be `local` if dismissible in the absence of its triggering event during live performance; and accordingly it is `global` if it should be launched in priority and immediately if the system recognizes the absence of its triggering event. Once again, the choice of an entity being `local` or `global` is given to the discretion of the composer.

The combination of the synchronization strategy attributes (`tight` or `loose`) and error handling attributes (`local` or `global`) for a group of accompaniment actions give rise to four distinct situations. Figure 6 attempts to showcase these four situations for a simple hypothetical performance setup. In this example, the score is assumed to demand for four distinct performer events (e_1 to e_4) with grouped actions whose two actions are initially specified to occur on e_1 and e_3. The figure demonstrates the simulation of the system behavior in case e_1 is missing during live performance for the four configurations discussed above. Naturally, in our realtime setup, e_1 is reported as missed once e_2 is detected.

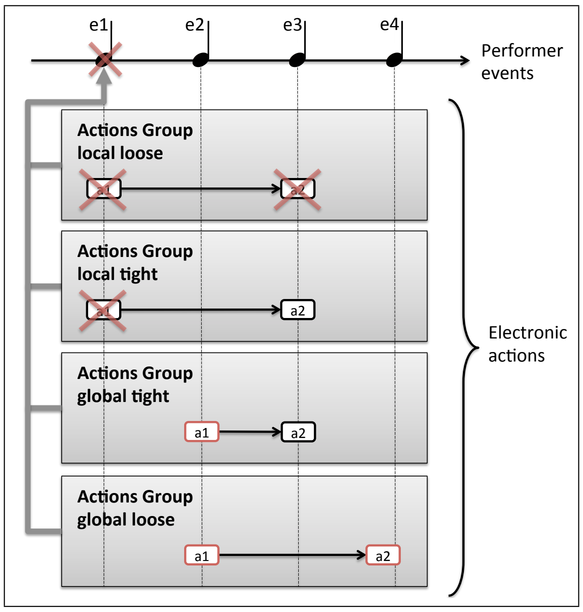

e1
e2
e3
e4
Performer events
Actions Group local loose
a1
a2
Actions Group local tight
a1
a2
Actions Group global tight
a1
a2
Actions Group global loose
a1
a2
Electronic actions

Figure 6. Accompaniment behavior in case of missed event for 4 synchronization and error handling strategies.

It is worth to note that each combination corresponds to a specific musical situation encountered in authoring of mixed interactive pieces:

`local` and `loose`: A block that is both local and loose corresponds to a musical entity with spatial independence with regards to its counterpart instrumental events, and strictly reactive to its triggering event onset (thus dismissed in the absence of this event).

`local` and `tight`: Strict synchrony of inside actions whenever there's a spatial correspondence between events and actions in the score. However actions within the strict vicinity of a missing event are dismissed. This case corresponds to an ideal concerto-like accompaniment system.

`global` and `tight`: Strict synchrony of corresponding actions and events while no actions is to be dismissed in any circumstance. This situation corresponds to a strong musical identity, strictly tied to the performance events.

`global` and `loose`: An important musical entity with no strict synchrony once launched. Such entity is similar to musical phrases that have strict starting points with *rubato* type progressions (free endings, tempo-synchronous).

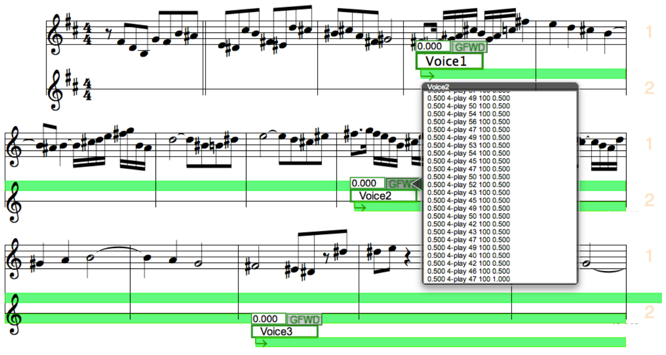
5. SAMPLE RESULTS

The above proposal has been integrated in *Antescofo*'s formal language and adopted in various new music productions involving live instrumental and electronic music. The best test for such systems is to see and use them in action. For the sake of completeness we discuss two sample results: one on a simple automatic accompaniment setting and another in case of a contemporary music production. Curious readers are referred to our website for further videos and events[3].

In the first example, we attempt to reconstruct the performance of a four-voice Fugue by J.S. Bach (in *B-minor*) where one voice is performed by a musician and others by automatic accompaniment. Figure 7 shows the score for this performance. In this example, and on purpose, the three accompanying voices are written as three distinct groups of actions (for MIDI accompaniment in this case). Each block corresponds to the accompaniment actions for each voice and the top staff represents the instrumental part. We choose `tight` synchronization strategies to ensure precise synchrony with a performer and also between different accompaniment voices. To ensure musical consistency, `local` strategies are assigned to the groups. In this way, missed event's corresponding actions will be dismissed without altering the overall performance. Despite its intuitive nature in run-time, the way the accompaniment voices are written and handled in realtime provide ease of authoring for composers willing to use such systems for their compositions.

The second example is an excerpt score from the beginning of "Tensio" (2010) for String Quartet and live electronic by French composer Philippe Manoury as illustrated in Figure 8. The left side of Figure 8 corresponds to the hand-written manuscript (excerpt) containing both instrumental (string section, four staves in bottom) and live electronic scores (top staves), and the right side of the figure correspond to its *Antescofo* equivalent used during live performance. The *Antescofo* score describes three parallel group actions entitled `arco`, `h1-trans` and `Pizzicati` corresponding to the left column. The two groups `arco` and `Pizzicati` contain *discrete* and atomic sequences

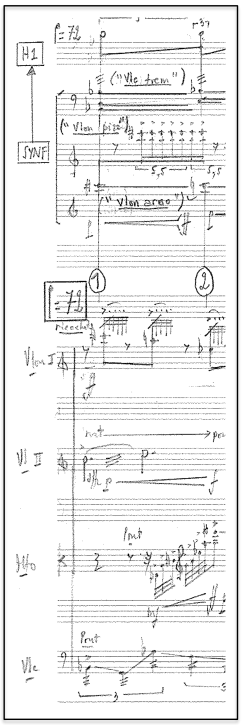
[3]`http://repmus.ircam.fr/antescofo`

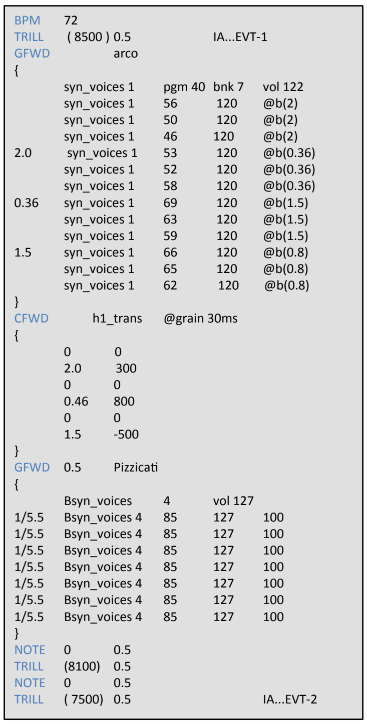


```
0.500 4-play 49 100 0.500
0.500 4-play 50 100 0.500
0.500 4-play 54 100 0.500
0.500 4-play 56 100 0.500
0.500 4-play 47 100 0.500
0.500 4-play 49 100 0.500
0.500 4-play 53 100 0.500
0.500 4-play 54 100 0.500
0.500 4-play 45 100 0.500
0.500 4-play 47 100 0.500
0.500 4-play 50 100 0.500
0.500 4-play 52 100 0.500
0.500 4-play 43 100 0.500
0.500 4-play 45 100 0.500
0.500 4-play 49 100 0.500
0.500 4-play 50 100 0.500
0.500 4-play 42 100 0.500
0.500 4-play 43 100 0.500
0.500 4-play 47 100 0.500
0.500 4-play 49 100 0.500
0.500 4-play 40 100 0.500
0.500 4-play 42 100 0.500
0.500 4-play 46 100 0.500
0.500 4-play 47 100 1.000
```

Figure 7. *Antescofo* score of Bach's *B-minor* fugue in automatic accompaniment mode, in *NoteAbility Pro*.

(defined with the keyword `GFWD`) whereas the `h1-trans` (defined as `CFWD`) contains a *continuous* sequence. The continuous process `h1-trans` correspond to the top staff in the left column. During realtime performance their polyphonic synchrony is assured despite any variations in the performance. These groups are notated as `global` and `loose` (the default behavior of *Antescofo*) and correspond as discussed to integral musical phrases launched in parallel to the instrumental world and influenced by the environment's tempi.





```
BPM     72
TRILL    ( 8500 ) 0.5                IA...EVT-1
GFWD             arco
{
        syn_voices 1      pgm 40  bnk 7     vol 122
        syn_voices 1      56       120      @b(2)
        syn_voices 1      50       120      @b(2)
        syn_voices 1      46      120       @b(2)
2.0      syn_voices 1     53       120      @b(0.36)
        syn_voices 1      52       120      @b(0.36)
        syn_voices 1      58       120      @b(0.36)
0.36    syn_voices 1      69       120      @b(1.5)
        syn_voices 1      63       120      @b(1.5)
        syn_voices 1      59       120      @b(1.5)
1.5     syn_voices 1      66       120      @b(0.8)
        syn_voices 1      65       120      @b(0.8)
        syn_voices 1      62        120      @b(0.8)
}
CFWD        h1_trans     @grain 30ms
{
        0          0
        2.0         300
        0          0
        0.46       800
        0          0
        1.5        -500
}
GFWD    0.5        Pizzicati
{
        Bsyn_voices       4          vol 127
1/5.5   Bsyn_voices 4     85         127        100
1/5.5   Bsyn_voices 4     85         127        100
1/5.5   Bsyn_voices 4     85         127        100
1/5.5   Bsyn_voices 4     85         127        100
1/5.5   Bsyn_voices 4     85         127        100
1/5.5   Bsyn_voices 4     85         127        100
1/5.5   Bsyn_voices 4     85         127        100
}
NOTE    0          0.5
TRILL   (8100)     0.5
NOTE    0          0.5
TRILL   ( 7500)    0.5                         IA...EVT-2
```

Figure 8. Score manuscript of the first bars of "Tensio" (left), and the *Antescofo* counterpart (right).

6. CONCLUSION

In this paper we attempted to formalize the problem of synchronization and coordination of actions in an automatic accompaniment setting by considering them as reactive synchronous systems on top of classical score following and machine listening. With such considerations we aimed at addressing the error handling and intelligent synchronization strategies despite human or machine listening error imposed to the system during a live performance. This is in contrast to most existing approaches where the integrity of the musical output is highly dependent on the healthiness of the artificial listening modules at work. This becomes possible by studying the language constructs of both music composition and performance within a computer setting.

The solutions proposed in the paper try to cover various musical situations that correspond to concerto-like accompaniment settings as well as compositional and performative aspects of interactive computer music pieces. They are integrated within the *Antescofo* software and have been employed in various music productions.

The coupling of high-level computer language constructions with that of low-level machine listening, at the core of this paper, is a necessity in musical practices of interactive computer music and requires more research in the computer music community. We believe that research on those lines could address complex problems with simple and elegant solutions useful for both music composition and live performance.

7. REFERENCES

[1] A. Cont, "Antescofo: Anticipatory synchronization and control of interactive parameters in computer music," in *ICMC* 2008.

[2] ——, "A coupled duration-focused architecture for realtime music to score alignment," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 6, pp. 974–987, 2010.

[3] ——, "On the creative use of score following and its impact on research," in *SMC 2011*.

[4] R. B. Dannenberg, "An on-line algorithm for real-time accompaniment," in *ICMC* 1984, pp. 193–198.

[5] J. Echeveste, A. Cont, J.L. Giavitto, F. Jacquemard, "Antescofo: a Domain Specific Language for real time musician-computer interaction," in *JDEDS* 2012, submitted.

[6] C. Raphael, "The informatics philharmonic," *Commun. ACM*, vol. 54, pp. 87–93, 2011.

[7] R. Rowe, "Interactive music systems: machine listening and composing". MIT Press, 1992.

[8] R. Shyamasundar and S. Ramesh, "Real time programming: languages, specification and verification". World Scientific Pub Co Inc, 2009.

[9] B. Vercoe, "The synthetic performer in the context of live performance," in *ICMC*, 1984, pp. 199–200.