Model-Based Testing of an Interactive Music System

Florent Jacquemard  Clément Poncelet
clement.poncelet@ircam.fr

UMR SMTS 9912
Mutant Team

Team leader: Arshia Cont

3 permanents and 3 Phd students

Realtime Recognition and Extraction of musical data from audio signals.

Realtime Synchronous Programming in Computer Music.

Publications [http://repmus.ircam.fr/mutant/publications](http://repmus.ircam.fr/mutant/publications)
Outline

**Introduction**

- Antescofo an Interactive Music System
- A Model-Based Testing application

**A running method for testing**

- Offline and Automatic timed conformance Testing
- An IMS Model: The Intermediate Representation

**Conclusion**

- Conclusion & Future works
Instrumental Music
A play between the composition and the performance time

Anthèmes 2
pour violon et dispositif électronique (1997)

Libre
brisque
($J = 92$)

Violon

($J = 92$) rall.
batt. (archet normal)
($J = 66$)

Pierre Boulez
(1925)
Instrumental Music
A play between the composition and the performance time

Anthèmes 2
pour violon et dispositif électronique (1997)

Pierre Boulez
(*1925)

Libre
brasque
(\( \text{\( \text{\( J \)} \) = 92} \))

Violon

Régie informatique

(\( \text{\( \text{\( J \)} \) = 92} \)) rall.
batt. (archet normal)

(\( \text{\( \text{\( J \)} \) = 66} \))
Mixed music
realtime interaction between acoustic and electronic instruments
Mixed music
realtime interaction between acoustic and electronic instruments
Automatic Accompaniment
polyphonic score following and audio accompaniment
Automatic Accompaniment
polyphonic score following and audio accompaniment
IMS

Unpredictable inputs

Hardly Timed reactions
Antescofo an Interactive Music System

2 phases
Composition
Performance

mixed score
DSL
Antescofo an Interactive Music System

2 phases

Composition

Performance

mixed score DSL

Environment

MAX/MSP

inputs

outputs

discrete message passing
Antescofo an Interactive Music System

Environment

MAX/MSP

audio signal
inputs

outputs

discrete message passing

Listening machine

tempo

position

Reactive engine

mixed score DSL

Antescofo

Antescofo an Interactive Music System
Piece creation

Ideas → Score → Concert ?
Testing a piece
Prepare the IMS for a concert

realtime
interpretation

rehearsal
human checking
of outcome

no time for debugging!
Testing a piece
Prepare the IMS for a concert

Manual method
Test for one performance
Tedious (perform the whole piece)
Not precise

Model Based Testing
Covering generation
Automatic and Virtual Clock
Formal (accurate and informative)
Model Based Testing Antescofo

Environment
inputs
sound processing
outputs
Listening machine
tempo
discrete events
Reactive engine
position
Model Based Testing Antescofo

1 Introduction

This report presents procedures for testing Antescofo, a score-based Interactive Music System (IMS). It describes common Model-Based Test (MBT) vocabulary and presents methodology and use cases.

2 Model Based Testing

Figure 1 depicts in its upper half a reactive system’s Implementation Under Test (IUT) interacting with an environment RealENV, and in its lower half, two formal specifications of the latter, resp. $S$ and $E$.

The environment model $E$ is a formal description of what can be expected from the environment. In our case, it is the definition of the set of all possible input traces, i.e., all the potential interpretations of musicians to be tested on a given mixed-score.

Note that since IMS are realtime systems, we need to express time in $E$ and $S$, like in: "one message $m$ has to be emitted one beat after the first event $e_1$ of the musician".

We present the application of model based testing (MBT) techniques to a score based IMS called Antescofo. Roughly, our method proceeds with the following steps, depicted in Figure 2. First, a given mixed score is compiled into an intermediate representation (IR). This formalism can be presented as a table of finite state machines extended with delays and asynchronous communications.

\[ E \]
\[
\text{Environment model}
\]

\[ S \]
\[
\text{System specification}
\]
Model Based Testing Antescofo

1 Introduction
This report presents procedures for testing Antescofo, a score-based Interactive Music System (IMS). It describes common Model-Based Test (MBT) vocabulary and presents methodology and use cases.

2 Model Based Testing
Figure 1 depicts in its upper half a reactive system's Implementation Under Test (IUT) interacting with an environment RealENV, and in its lower half, two formal specifications of the latter, resp. $S$ and $E$.

The environment model $E$ is a formal description of what can be expected from the environment. In our case, it is the definition of the set of all possible input traces, i.e. all the potential interpretations of musicians to be tested on a given mixed-score.

Note that since IMS are real-time systems, we need to express time in $E$ and $S$, like in: "one message $m$ has to be emitted one beat after the first event $e_1$ of the musician".

We present the application of model based testing (MBT) techniques to a score based IMS called Antescofo. Roughly, our method proceeds with the following steps, depicted in Figure 2. First, a given mixed score is compiled into an intermediate representation (IR). This formalism can be presented as a table of finite state machines extended with delays and asynchronous communications.

![Diagram of Antescofo system](image)
Model Based Testing Antescofo

1 Introduction

This report presents procedures for testing Antescofo, a score-based Interactive Music System (IMS). It describes common Model-Based Test (MBT) vocabulary and presents methodology and use cases.

2 Model Based Testing

Figure 1 depicts in its upper half a reactive system’s Implementation Under Test (IUT) interacting with an environment RealENV, and in its lower half, two formal specifications of the latter, resp. $S$ and $E$.

The environment model $E$ is a formal description of what can be expected from the environment. In our case, it is the definition of the set of all possible input traces, i.e. all the potential interpretations of musicians to be tested on a given mixed-score.

Note that since IMS are real-time systems, we need to express time in $E$ and $S$ like in: "one message $m$ has to be emitted one beat after the first event $e_1$ of the musician".

We present the application of model based testing (MBT) techniques to a score based IMS called Antescofo. Roughly, our method proceeds with the following steps, depicted in Figure 2. First, a given mixed score is compiled into an intermediate representation (IR). This formalism can be presented as a table of finite state machines extended with delays and asynchronous communications.

Figure 1: Specification: reality (top) and models (down)
Outline

Introduction

- Antescofo an Interactive Music System
- A Model-Based Testing application

A running method for testing

- Offline and Automatic timed conformance Testing
- An IMS Model: The Intermediate Representation

Conclusion

- Conclusion & Future works
Test Workflow
Design of a running method to test using existing MBT tools

Timed traces: \( (a, t) \) with \( a \in \Sigma_{\text{in}} \cup \Sigma_{\text{out}} \) and \( t \in \mathbb{R}^+ \)

Mixed Score
Test Workflow
Design of a running method to test using existing MBT tools

**Timed traces:** \((a, t)\) with \(a \in \Sigma_{\text{in}} \cup \Sigma_{\text{out}}\) and \(t \in \mathbb{R}^+\)
Test Workflow
Design of a running method to test using existing MBT tools

Timed traces: $(a, t)$ with $a \in \Sigma_{\text{in}} \cup \Sigma_{\text{out}}$ and $t \in \mathbb{R}^+$
Test Workflow

Design of a running method to test using existing MBT tools

Timed traces: $(a, t)$ with $a \in \Sigma_{in} \cup \Sigma_{out}$ and $t \in \mathbb{R}^+$
Test Workflow

Design of a running method to test using existing MBT tools

Timed traces: \( \langle a, t \rangle \) with \( a \in \Sigma_{\text{in}} \cup \Sigma_{\text{out}} \) and \( t \in \mathbb{R}^+ \)
Test Workflow

Design of a running method to test using existing MBT tools

**Timed traces:** \( \langle a, t \rangle \) with \( a \in \Sigma_{\text{in}} \cup \Sigma_{\text{out}} \) and \( t \in \mathbb{R}^+ \)
Test Workflow
Design of a running method to test using existing MBT tools

Timed traces: \(\langle a, t \rangle\) with \(a \in \Sigma_{\text{in}} \cup \Sigma_{\text{out}}\) and \(t \in \mathbb{R}^+\)
Outline

Introduction

- Antescofo an Interactive Music System
- A Model-Based Testing application

A running method for testing

- Offline and Automatic timed conformance Testing
- An IMS Model: The Intermediate Representation

Conclusion

- Conclusion & Future works
An IMS Specification
Antescofo Domain Specific Language

; Antheme 2 INTRO
BPM 92 ; tempo a la croche attention

[...] ; PFWD 0.1 sl-t1-trait2 bang

; BPM 46 ;

TEMPO OFF
TRILL ( 7400 7500 ) 8.0 Q2 ;second_trait ; old_e2; Trill_1
cue_nb 2;
ir2-param reverb tr0 25
A2_samples_in 74 120 15
0.1 A2_samples_in 73 120 15
0.1 A2_samples_in 70 120 15
0.1 A2_samples_in 69 120 15
0.1 A2_samples_in 68 120 15
0.1 A2_samples_in 67 120 15
0.1 A2_samples_in 66 120 15
0.1 A2_samples_in 63 120 15

fd_1_del 25;
fd_1_fre -347;
fd_1_db 0;
ir1 99 500; start decrescendo reverb inf 1
An IMS Specification
Antescofo Domain Specific Language
An IMS Specification

Antescofo Domain Specific Language

input events expected

output actions triggered by events sent to environment
An IMS Specification
Antescofo Domain Specific Language

input events expected

output actions triggered by events sent to environment
An IMS Specification
Antescofo Domain Specific Language

+ High level features: Synchronisation, error management
dynamic processes …
Intermediate Representation

the syntax

symbols \( \Sigma = \Sigma_{in} \cup \Sigma_{out} \cup \Sigma_{sig} \)

- input events \( \Sigma_{in} \)
- output messages \( \Sigma_{out} \)
- internal signals \( \Sigma_{sig} \)

Representable with **Finite State Machine** + variables & time
Intermediate Representation

the syntax

symbols \( \Sigma = \Sigma_{\text{in}} \uplus \Sigma_{\text{out}} \uplus \Sigma_{\text{sig}} \)

- input events \( \Sigma_{\text{in}} \)
- output messages \( \Sigma_{\text{out}} \)
- internal signals \( \Sigma_{\text{sig}} \)

Representable with **Finite State Machine** + variables & time

- Synchronous transitions set (instantaneous)
  - emit
  - fork
  - conditional

\[ a \in \Sigma_{\text{out}} \uplus \Sigma_{\text{sig}} \]
Intermediate Representation

the syntax

symbols \( \Sigma = \Sigma_{in} \cup \Sigma_{out} \cup \Sigma_{sig} \)

input events \( \Sigma_{in} \)
output messages \( \Sigma_{out} \)
internal signals \( \Sigma_{sig} \)

Synchronous transitions set (instantaneous)
emit
fork
conditional

Asynchronous transitions set (delaying)
receive
delay
asap concurrency

\( a \in \Sigma_{out} \cup \Sigma_{sig} \)

\( b \in \Sigma_{in} \cup \Sigma_{sig} \)
\( tu \) is a time unit

\( \text{asap}\)
\( a_n ? \)
Intermediate Representation

the semantics with example

NOTE e1 d1
d11 msg11
d12 msg12
...
NOTE e2 d2
d21 msg21
...

Antescofo score

Executable bytecode

IR representation
Intermediate Representation

the semantics with example

NOTE e1 d1
d11 msg11
d12 msg12
...
NOTE e2 d2
d21 msg21
...

Antescofo score

Executable bytecode

Before and after unfolding

IR representation

Network of parallelized machines

time (pulsations)
Intermediate Representation

the semantics with example

**NOTE** e1 d1
d11 msg11
d12 msg12
...
**NOTE** e2 d2
d21 msg21
...

**Antescofo score**

**Executable bytecode**

**Before and after unfolding**

**IR representation**

**Network of parallelized machines**
Intermediate Representation

the semantics with example

NOTE e1 d1
d11 msg11
d12 msg12
...
NOTE e2 d2
d21 msg21
...

Antescofo score

Network of parallelized machines

Executable bytecode

Before and after unfolding

IR representation

Network of parallelized machines

time (pulsations)
Intermediate Representation
the semantics with example
Intermediate Representation

Automatic construction

Mixed Score
(AntescofoDSL)

Inference rules

\[
\frac{\text{env } \text{sc} : \mathcal{E}}{\vdash_{\text{env}} \text{sc} : \mathcal{E}} \quad \frac{\text{proxy } \text{sc} : \mathcal{P}}{\vdash_{\text{proxy}} \text{sc} : \mathcal{P}} \quad \frac{\text{evt } \langle [], \text{sc} \rangle \vdash G}{\vdash_{\text{all}} \text{sc} : \mathcal{M}}
\]

\[
\frac{\vdash_{\text{all}} \emptyset : \mathcal{A}_\emptyset}{\text{TOP}}
\]

\[
\frac{\langle \text{sc} : \text{ev}, \text{sc'} \rangle \vdash G}{\vdash_{\text{evt}} \langle \text{sc} : \text{ev}, \text{sc'} \rangle : G}
\]

\[
\frac{\langle \text{sc} : \text{ev}, \text{sc'} \rangle \vdash G \quad 0, \text{ev} : \text{sc'}, \tilde{e} \vdash_{\text{top}} \text{sa} : G_{\text{sa}}}{\vdash_{\text{evt}} \langle \text{sc}, \text{ev} : \text{sc'} \rangle : G \parallel T_e + G_{\text{sa}} + F_i}
\]

\[
\frac{\text{acc} + d, \text{sc}, \tilde{e} \vdash_{\text{al}} \text{sa} : G_{\text{sa}}}{\vdash_{\text{al}} \text{al} \vdash \text{sa} : G_{\text{sa}}'}
\]

\[
\frac{\text{acc}, \text{sc}, \text{al} \vdash \text{act}(d, \text{sa'}, \text{al}') : \text{sa} : G + G_{\text{sa}} \parallel T(g') + G_{\text{sa}}'}{\text{GRP}}
\]

\[
\text{acc}, \text{sc}, \text{e} \vdash_{\text{al}} \text{act}(d, \text{sa'}, \text{al}') : \text{sa} : G + G_{\text{sa}} \parallel T(g') + G_{\text{sa}}'
\]

\[
\frac{\text{acc} + d, \text{sc}, \text{clos}est(\text{acc} + d, \text{sc}) \vdash_{\text{al}} \text{sa} : G_{\text{sa}}}{\vdash_{\text{al}} \text{al} \vdash \text{sa} : G_{\text{sa}}'}
\]

\[
\frac{\text{acc}, \text{sc}, \text{e} \vdash_{\text{al}} \text{act}(d, a, \text{al}') : \text{sa} : G(a, d, \text{al}, e) + G_{\text{sa}}}{\text{MSG}}
\]

\[
G(\tau, d, \text{[loose, local]}, \text{sc}, e) = 1_{\text{c}} \ell \xrightarrow{d} \ell_i \xrightarrow{\tau!} \ell'_{\text{c}}
\]
Environment model

Universal environment

Environment that can miss $e_1$

Environment modeling an interpretation
Conclusion & Perspectives

Conclusion

Model-Based Testing an IMS with automatic model construction
Automatic generation of test cases with CoVer and alternative methods

Future Works

Visualisation for IR (Assistance for composition)
Using Ascograph, Ptolemy, SpaceEx

IR
VM = IR interpreter.
Analyse robustness, race conditions, time safety…

Online
TRON-like framework: On-the-fly test suite generation
Online Testing

Uppaal/Tron [Larsen, Mikucionis, Nielsen 04]

Methods of conformance testing for Antescofo

Clement Poncelet
January 18, 2015

1 Introduction
This report presents procedures for testing Antescofo, a score-based Interactive Music System (IMS). It describes common Model-Based Test (MBT) vocabulary and presents methodology and use cases.

2 Model Based Testing

Figure 1 depicts in its upper half a reactive system's Implementation Under Test (IUT) interacting with an environment RealENV, and in its lower half, two formal specifications of the latter, resp. \( S \) and \( E \).

The environment model \( E \) is a formal description of what can be expected from the environment. In our case, it is the definition of the set of all possible input traces, i.e. all the potential interpretations of musicians to be tested on a given mixed-score.

Note that since IMS are realtime systems, we need to express time in \( E \) and \( S \), like in:

"one message \( m \) has to be emitted one beat after the first event \( e_1 \)."

We present the application of model based testing (MBT) techniques to a score based IMS called Antescofo. Roughly, our method proceeds with the following steps, depicted in Figure 2. First, a given mixed score is compiled into an intermediate representation (IR). This formalism can be presented as a table of finite state machines extended with delays and asynchronous communications.

Figure 1: Specification : reality (top) and models (down)
<table>
<thead>
<tr>
<th>Reference</th>
<th>Title</th>
<th>Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>[Honing 01]</td>
<td>From time to time: The Representation of timing and tempo.</td>
<td>2001</td>
</tr>
</tbody>
</table>
Automatic construction of models

Mixed Score (AntescofoDSL)

Intermediate Representation

Under restrictions

Timed Automata
Generation of Input traces

**Goal:** create a test case $(t_{in}, t_{out})$ where $t_{in}$ in $\mathcal{E}$ and $t_{out}$ in $S(t_{in})$
Generation with **Uppaal** & **CoVer**

Uppaal extension for Offline generation [Blom, Hessel, Jonsson, Peterson 04]

---

**User defined coverage criterion on model items**

- **Environment model** (E)
- **System specification** (S)

**Generation**

- *observers* = FSA motoring the simulation
- exploration of the symbolic state set
- keep the trace with maximal number of Observer’s final states

**Conclusion Cover**

- Well suited for debugging
- Shortest delays → tempo explosion
- Scalability problem (small scores/extracts)
Fuzzing the ideal trace

Performance Models (aka fuzz) [Dannenberg 97, Honing02]

Timing function (TIF)

\[ f \equiv \langle f_{\text{shift}}, f_{\text{tempo}} \rangle \]

- time shifting for individual events
- tempo curve

Conclusion Fuzzing ideal trace
+ Scalable
+ Preparation of concerts
- Lack of coverage criteria
From recorded audio file

Conclusion User File
+ Well suited for Composition assistance
- Input score specifiable
Black-Box execution

3 Scenarios

bounds:
reactive machine

Antescafo: standalone

input trace

<i,d,T>

Internal adapter

Listening machine

position

tempo

Reactive engine

expected test trace

real output trace
Black-Box execution

3 Scenarios

Reactive engine

Listening machine

bounds:
reactive machine + tempo management

standalone

expected test trace

real output trace

input trace

Internal adapter

position

tempo

<i,d>

Listening machine

<Anteosco>
Black-Box execution

3 Scenarios

bounds: whole Antescofo

audio signal

input trace

external adapter

MAX/MSP

Antescofo standalone

Listening machine

position

tempo

Reactive engine

expected test trace

real output trace
Timed conformance

\(r_{tioco_e}\)

Relativized timed conformance input/output relation

conformed iff \(\forall\) test cases verdict OK

---

Testing Real-Time Systems Using Uppaal
[Hessel, Larsen, Mikucionis, Nielsen, Pettersson, Skou08]