



HAL
open science

VIRAGE: Designing an interactive intermedia sequencer from users requirements and the background

Antoine Allombert, Raphaël Marczak, Myriam Desainte-Catherine, Pascal Baltazar, Laurent Garnier

► To cite this version:

Antoine Allombert, Raphaël Marczak, Myriam Desainte-Catherine, Pascal Baltazar, Laurent Garnier. VIRAGE: Designing an interactive intermedia sequencer from users requirements and the background. International Computer Music Conference 2010, Jun 2010, New-York, United States. pp.1-1. hal-00516335v1

HAL Id: hal-00516335

<https://hal.science/hal-00516335v1>

Submitted on 9 Sep 2010 (v1), last revised 14 Dec 2010 (v2)

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

VIRAGE : DESIGNING AN INTERACTIVE INTERMEDIA SEQUENCER FROM USERS REQUIREMENTS AND THEORETICAL BACKGROUND

A. Allombert, R. Marczak, M. Desainte-Catherine

LaBRI

allomber, marczak, myriam@labri.fr

P. Baltazar

GMEA

pb@gmea.net

L. Garnier

Blue Yeti

laurent@blueyeti.fr

ABSTRACT

We present the research process of the *Virage* project and its main achievement : a sequencer for authoring and performing interactive intermedia scenarios. This two year long project addressed the question of authoring and controlling interactive scenarios dealing with several heterogeneous digital media, in the context of the performing arts. We show here how we adapted a theoretical background, thought for the interpretation of musical scores, to specific requirements in collaboration with practitioners in the field,. We also describe how these exchanges about the features of the sequencer guided the software development.

1. INTRODUCTION

The *Virage* project attempted to address the questions of authoring time and interaction in the context of media management (sound, lighting, image, machinery...) for the performing and interactive arts. Since several years, media managers and artists have used more and more digital contents and tools. For this purpose the practitioners use software and tools designed for each type of content, or adapt softwares coming from other fields (e.g. computer music). However, more and more complex cases challenge the possibilities of scripting the time organization of heterogeneous contents, as well as ways to interact with them.

The *Virage* project proposed some answers to these questions, by designing an interactive sequencer for scripting time and interaction with heterogeneous digital contents. The results of the study and the participation of practitioners fueled the the discussion of the theoretical model. In the same manner, each step of the implementation was approved by every members of the project.

We first present some elements of the study. Then we focus on the theoretical background of the project, and how the field requirements influenced the design of the model and its implementation. At last, we propose some applications and perspectives.

2. CONTEXT AND CHALLENGES

This project can be considered as a practical reflection on time-based media scripting in digital artistic creation, par-

ticularly in relation to live performance. Previously to the project, a study has been carried out and published [?] in order to analyze the limits of existing software systems for interactive performing arts situations. This preliminary study was then completed by a field survey¹ conducted at the very beginning of the project, that led to write initial specifications that were updated at the very term of the project (February 2010). The final objective of the project was indeed not to produce effective software, but rather to write these specifications -in order to allow future development- through the development of usable prototypes, and feedback from their confrontation to real-world cases

2.1. Interoperability, modularity

The insularity between the existing multi/intermedia software environments, has been stated through the preliminary studies as a major bottleneck for creative workflow in the context of the performing arts.

The *Virage* sequencer² has then been proposed as an experimental way to overcome this insularity by acting as a hub between these environments, dedicated to the design of temporal parameter management. This has been achieved by combining the *Virage* sequencer with the existing software and hardware environments that are actually used in production by the practitioners in the field. In order to do that, a plugin interface has been developed, that will be further explained in 4 and allows the coexistence and simultaneous usage of several protocols, including : Open Sound Control, Minuit³, CopperLan⁴, potentially any network or device protocol can be added to the list by developing third-party dedicated plugins.

In respect to this concern of interoperability, the *Virage* sequencer focuses on the scripting of behaviors of abstract parameters, and not of the media themselves (as in all existing sequencers, except Iannix [?], which has been reviewed and didn't show as relevant to the studied field, namely the

¹http://www.platforme-virage.org/wp-content/uploads/2009/11/Virage_Analyse_des_Usages.pdf

²downloadable at:<http://www.platforme-virage.org/?p=1297>

³specifications at: <http://www.platforme-virage.org/?p=1444>

⁴<http://www.copperlan.org/>

performing arts) This design choice allows a certain independence from the medias' intrinsic durations, similarly to what was experienced with MIDI sequencers before the introduction of DSP.

2.2. Flexible time

Existing creative multi/intermedia software environments generally address time-scripting by one of the next two strategies : a fixed timeline (DAWs and Video editing softwares), that allows a very precise scripting of media behaviors or cue lists (Theater Cue Managers such as Qlab⁵ or SFX but also in Ableton Live⁶ which is very frequently used in performing arts projects)

Time is then managed as a monolithic fixed flow (as in the timeline model), or as a set of unrelated discrete events (as in the cue list model). It is to note that Ableton Live proposes both of these models in the software, but that they are completely unrelated temporally.

The practitioners in the field expressed a need for a third strategy that might be a mix of these ones because, on one hand, the fixed timeline representation appeared to be too strict to adapt to the live character of performing arts. On the other hand, the cue representation is lacking of possibilities of designing complex time structures.

Thus we developed a system in which one can express complex and precise time organization : temporal relations can be drawn between events of the scenario, due to artistic choices or technical constraints, with optional duration for the intervals between the events. Additionally, the behavior of the live action can be followed by changing some characteristics of the temporal organization during a performance, including the execution speed of the scenario. For this purpose, the system should adapt the scenario by respecting the time relations and accepting real-time modifications. We use the term "flexible time" to describe this type of behavior.

In such a system, a scenario is no more the representation of a single performance, but it represents a set of possible performances that share temporal properties. Then, our system can be compared to other works as *Harp* [?] or *DoubleTalk* [?], which considered a musical score as the representation of several possible executions with shared properties.

3. THEORETICAL MODEL

The *Virage* sequencer is based on an hybrid temporal paradigm that has been initially designed for musical composition

For a specific musical approach and a complete presentation, one can refer to [?].

We assume that an interactive scenario is the temporal organization of a set of *events*, in which the temporal characteristics are partly specified to allow some modifications during the performance. During the authoring phase, the user (the media manager or artist) can specify some temporal properties that he wants to be respected during each performance : temporal orders between some events and ranges

of value for the time intervals between the events. He also specifies what can be changed during a performance. This way of scripting the temporal organization is close to "constraints programming", since a performance constitutes an instance among possible scenarios that respect some given rules.

3.1. The authoring model

We present here the graphical language used for authoring the interactive scenarios. An example of such a scenario is given in figure ??.

Since the system is thought as a hub that can send abstract data to other applications, the authoring formalism is based on *processes* able to compute and produce abstract values. The user can define how the processes will run during the performance. For this purpose, he defines the temporal organization of his scenario by using *temporal objects*.

The *temporal objects* are the basic element of the temporal model which is inspired from the hierarchical models proposed by Balaban [?]. A *temporal object* with no children, called a *texture*, represents the execution in time of a given *process*. A *temporal object* with children, a *structure*, represents the temporal organization of its children. Each *structure* has its own timeline, scaled with its own time unit. A scenario is represented by a root *structure*. Each object is associated with a set of *control points* which represent particular moments of its execution.

On the figure ??, the *objects* "sound", "red" and "green" are *textures* that represent the execution of processes, which read tables in order to send values to specialized applications ; "lights" and "My scenario" are structures.

3.1.1. Temporal relations

The user can define the temporal properties of his scenario through temporal relations (taken from points algebra) between the *control points* of the *temporal objects*. There are 2 qualitative relations : precedence (*Pre*) and posteriority (*Post*). In addition, the user can specify quantitative constraints, by giving a range of possible values for the time interval between two points bound by a relation.

The system will maintain the properties imposed by the temporal relations, during the authoring process as well as during the performance. In the first case they can help the editing operations, in the second case, they define the limits for the modifications introduced by the performer through the *interaction points*.

3.1.2. Interaction Points

During the authoring process, the user can define some *control points* to be dynamically triggered by the performer during the execution. These *control points* are said *dynamic* (as the beginning of the "lights" *structure* in figure ??), while the other *control points*, the *static* ones, will be triggered by the system. The written date of a dynamic *control point* is indicative. The possibilities are limited by the temporal relations. Then, in the same example, the system will refuse the dynamic triggering of the point L_α until the duration $\Delta_{1_{min}}$ has elapsed, in order to respect the relation between M_α and

⁵<http://figure53.com/qlab/>

⁶<http://www.ableton.com/>

L_α . Similarly, the system will automatically trigger L_α if the performer did not do it until the duration Δ_{max} has elapsed.

3.2. The execution model

On the execution side of the system, we propose a generic abstract machine, the *ECO* Machine, that runs the scenarios written with the formalism.

Inside this machine, we represent the temporal organization of the scenario with Petri nets structures (precisely hierarchical time stream Petri nets [?]), which are runnable place/transition representations of partially ordered set of events ; they have been used several times in computer music applications [?]. Other representation could be explored such as concurrent constraints [?].

4. IMPLEMENTATION

In this section we present the *IScore* library, an implementation of the formalism described in section 3. Thus, the library is the union of 2 parts corresponding to both sides of the model : an editor engine to write the temporal constraints based scenarios and an execution engine to compile and run the scenarios. This library, developed in C++ and currently working on *Mac OSX* and *Linux*, is open source⁷.

4.1. Architecture

The architecture of the implementation corresponds to the theoretical model.

4.1.1. Authoring

For the authoring part, we use a constraints solver library (*GeCode*⁸). The dates of the *control points* and the temporal relations are turned into a constraints problem. Each time a date is changed, the solver is called to find new values for the other dates and the scenario is updated.

4.1.2. Execution

On this side, we directly implemented the *ECO* machine as it is theoretically described. Thus, we implemented a Petri nets runner. During the execution, the Petri net is ran according to its temporal characteristics. When triggering an event, the runner sends a message to a part of the software which launches and stops the processes associated to the *temporal objects*.

4.1.3. DeviceManager library

The *DeviceManager* library is implemented to manage network communications. Indeed the *Virage* software has to control many different technologies by sending messages. This library hides the protocols used to communicate with other applications. Communication protocols quickly evolve so we had to think about an extendable solution based on a plugin system conceived in the *Tulip* project⁹, and develop dedicated plugins for each of the supported protocols.

⁷<http://github.com/raphaelmarczak/libIScore>

⁸<http://www.gecode.org/>

⁹<http://www.tulip-software.org>

4.2. Users involvement

Users have been implied in the research process all along the project, and at all its stages : from the initial definition of features through directed experimentations to the conception stage by giving constant feedback and defining priorities of the development tasks. Then, once the first stable prototype had been produced, all successive steps of the development were the opportunity to confront the prototypes to a diversity of real-world situations generated by experimental workshops led by the 6 artistic partners of the research platform, ranging from the performing arts to digital interactive arts. These experimentations of the prototypes have been done within the existing creative environments of the practitioners in the field, and developed around actual artistic questions, in actual production situations, but out of production time, in order to avoid stressing the research process with production deadlines and constraints.

4.3. Graphical interface

The main space of the graphical interface is dedicated to the edition of the temporal scenario, using the forthcoming elements : *Temporal objects* are represented by boxes on the time line, that can be connected at each end by *Relations*, may they be fixed (bold lines), flexible (dashed lines) or semi-flexible with bounds (combination of bound and dashed lines representing the respective fixed and flexible durations). *Trigger Points* can be created and attached to the temporal objects, allowing the user to interactively trigger them. They are currently placed on an upper rail for more straightforward ordering and triggering by the operator, when executing the scenario. Though, it has been noticed that this representation was in some way restrictive to the creative temporal possibilities of the model. This placement of Trigger Points will be alternatively moved against the objects or on the time line rail depending on the editing mode (authoring or execution) in a future version.

Apart from this main space, the user has access to a Namespace browser for exploring the remote environments.

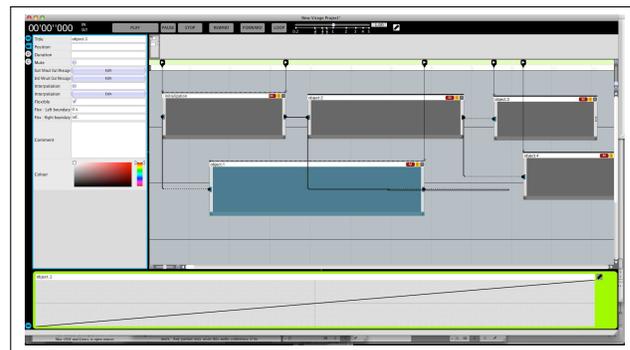


Figure 1. A screenshot of the graphical interface.

5. PERSPECTIVES

The consolidation of the prototype as a usable cross-platform open source software is currently contemplated by a fur-

ther consolidation of the project's consortium and associated partners as a cooperative in order to keep the collaborative spirit developed throughout the research project

This consolidation phase will mostly focus on robustness (including porting the graphical interface to the Qt framework), ergonomics and usability

A few features that were not developed in the project will be added, including routing and conditioning of data flows or events from and to remote addresses, in order to allow to script continuous interaction. Also, the edition of hierarchical scenarios will be added to the features of the graphical interface.

Further research has been submitted to focus on the integration of conditional objects, non-linearity and multi-user scenarios.

6. CONCLUSION

We presented a research project that consisted in gathering researchers and practitioners in the field, to propose solutions to real problems in the field of interactive media arts. The constant communication along the way allowed us to fuel the research process with new questions that led to adapt the theoretical background, but also to improve it for new developments. The research situation helped the field agents to clarify their point of view on their own constantly-evolving practices, by stepping back from their production-oriented habits to a more experimental situation questioning the very nature of the temporal aspects of their creation.

This partnership showed successful because both parts were enough mature and brought along existing models for scripting interaction on the one hand, real cases and newly established practices on the other hand. Finally, the software prototype gave an experimental tool and initiated the development of a really efficient software.

7. ACKNOWLEDGMENTS

This research project has been funded by French National Agency, in the "Audiovisual and Multimedia" 2007 program. The authors would also like to thank all of the partners of the Virage platform¹⁰ that have been contributing by feedback, advising and workforce in order to achieve this project.

¹⁰http://www.plateforme-virage.org/?page_id=14