IanniX

Aesthetical/Symbolic visualisations for hypermedia composition

Thierry Coduys
Director, La kitchen
78, Avenue de la République
75011 Paris, France
Tel: + (33) 1 56 79 02 89
Thierry.Coduys@la-kitchen.fr

Guillaume Ferry
Researcher, La kitchen
78, Avenue de la République
75011 Paris, France
Tel: + (33) 1 56 79 02 89
Guillaume.Ferry@la-kitchen.fr

ABSTRACT

IanniX proposes a graphical representation of a multidimensional and multi-formal score, a kind of poly-temporal meta-sequencer.
It allows a multi-topology of the space and relays on the principals of UPIC. IanniX proposes at a macro level of the graphical score a symbolic representation of complex objects still existing in the micro level.
The development of IanniX started in 2002, through collaboration between Thierry Coduys (La kitchen), Adrien Lefevre (Adlef), and Gérard Pape (CCMIX). This software continues to privilegiate graphic representation for hypermedia score.
An evolution of IanniX is currently under development at La kitchen, bringing new features, especially on aesthetic and graphical customization aspects. Composers can use either GUI (Multiple Document Interface, with direct interactions) or XML scripts directly (advanced edition) to create their scores.
IanniX provides network interfaces to communicate with processing software's (PureData, Max/MSP, SuperCollider, VirChor...) using standard UDP protocol.
IanniX is cross-platform software working on Linux, Mac OS or Win32 systems as well.
IanniX has been conceived in honour to Iannis Xenakis, conceptor of UPIC.
This project is sponsored by the French Ministry of Culture (DMDTS).

Keywords
Graphical hypermedia composition, Symbolic & Aesthetical visualisation, Multi-formal and Poly-temporal score.

1. NEW CONCEPTS & FEATURES

The usage of the former versions of IanniX, has been very helpful in the development of this version of the software with new concepts and features, as follows:

- Evolution of the architecture, introducing a client/server structure.
- IanniX is given a new GUI (Graphical User Interface), allowing interactive composition in real-time, directly on the graphical visualization window. Interactions are controlled by **mouse/keyboard combination**, or even by **external controllers** (sensors, MIDI controllers...).
- Qt environment (Trolltech, under GPL license) was chosen to implement the new interface. Qt is a high-level C++ application framework, with many advantages:
  - Cross-platform system (X11, MacOS, Win32 with GPL specific version)
  - Special module to manage OpenGL graphics.
  - Interface descriptions are saved in an XML-like format, directly editable with Qt Designer.

An external 3D rendering is available; using XML script rendered Virtual Choreographer (VirChor, http://virchor.sourceforge.net), providing aesthetic & artistic visualization of scores.
Render styles customizable through XSL style sheets.
Since dedicated softwares such as PureData or Max/MSP exist, it is possible to communicate with them easily; IanniX is now especially focusing on graphical visualization.
However, IanniX still contains its own audio processor.

2. ARCHITECTURE EVOLUTION

This new version was developed using the following environments:

- C++ for kernel implementation (Server, Augmented Mapping)
- Qt for GUI design
- OpenGL library for interactive 3D edition & viewing
- MySQL for database management.

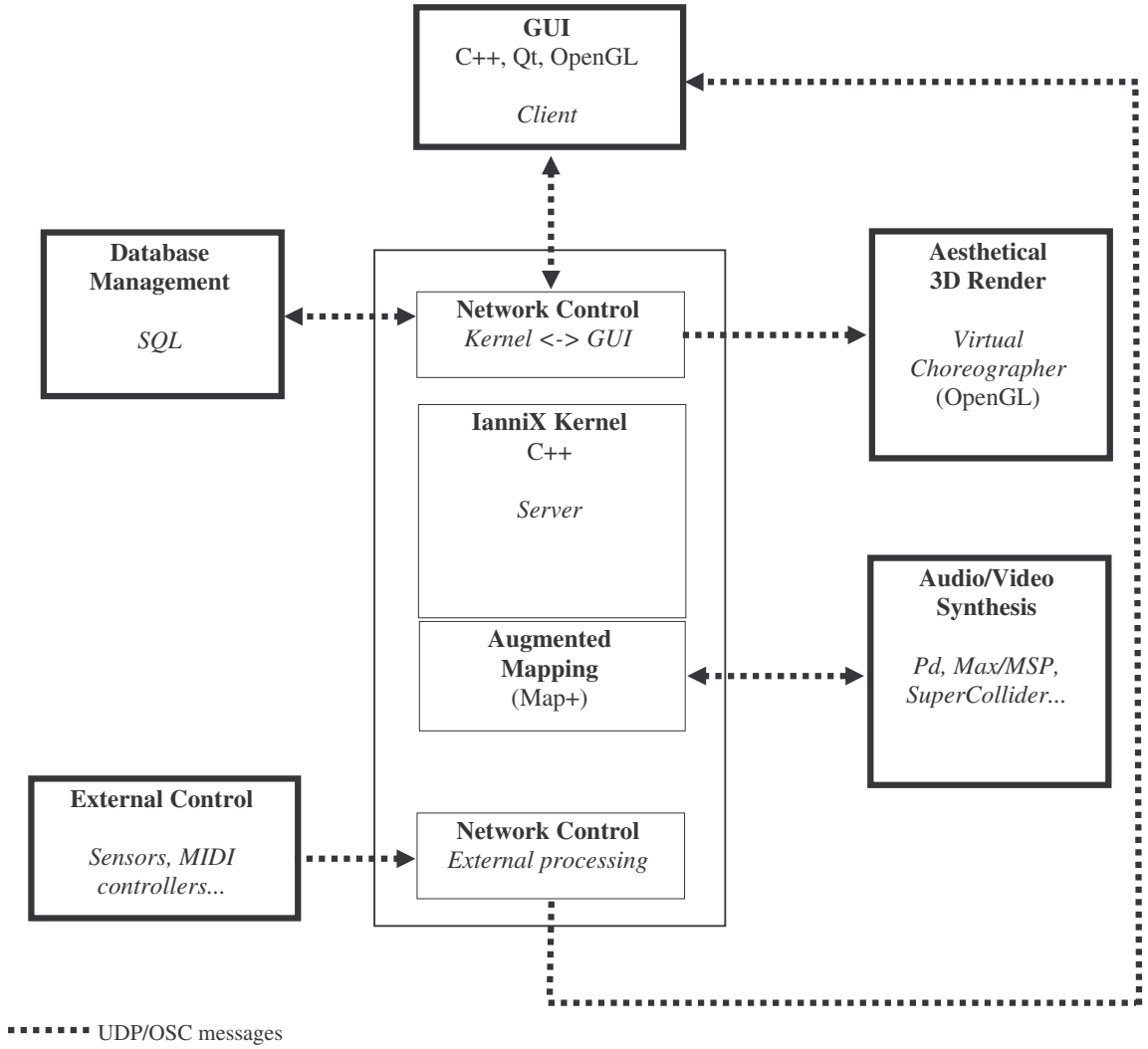


Figure 1. Architecture

3. IANNIX BASIS

3.1. Score structure

A score is a 3-dimensional space (X, Y, Z), designed to contain many objects.
It is defined by two parameters : height and width.
Both parameters represent score's global duration in X/Y axis.
They are given in milliseconds, millisecond being the time unit for scores.

Objects can be placed anywhere on the score, even in depth, along Z-axis.
By the way, objects size (curves and trajectories) corresponds to their self-duration. This representation provides a precise visual feedback for timing.

Four object classes can be found in IanniX.
Each object has a specific function, and some of them are linked (for example, cursors must be linked to a single trajectory).

3.2. Objects definition

A generic object is defined with:

- Unique identifier & group identifier (optional)
- Name (optional)
- 3D Position (x, y, z)
- A specific colour
- Graphical properties (for aesthetic render)
- IP address & port number (for UDP connections)

3.2.1. Curves

IanniX curves are B-Splines, which provides many advantages:

- A curve is defined by a set of control points
- Curve precision is not affected by scaling large panel of shapes available
- A curve can be graphically created and edited, through control points placing & moving.
- Random curves can be generated, too.

3.2.2. Trajectories

A trajectory is a single curve with a set of cursors on it. Trajectories can be linear (horizontal, vertical, with angle), circular (for loops) or user-defined with B-splines.

3.2.3. Triggers

Triggers are basically graphical links to external media files. These files will be processed by external software, as specified in network parameters. File types can be video, audio, pictures and text... As soon as a cursor crosses a trigger, the associated file is played.
A trigger is defined with duration and a playback mode (loop, single playback)

3.2.4. Cursors

These are the objects dedicated to playback control. A cursor is assigned to a single trajectory and will evolve in time following trajectory's shape, many cursors can follow the same trajectory. In former versions of IanniX, cursors had an infinite width, now this width can be specified.
Cursors are controlled with two parameters, width and playback speed. Both of them can be constant through time, or variable, following a temporal function. Temporal functions are editable through timeline editor (cf Fig.3)

3.3. Score Interpretation

First of all, IanniX must be connected to an audio processing software supporting OSC protocol, for example PureData, to initialize its playback.

When a score is interpreted, cursors will start moving on their trajectories at a certain speed, given by their own speed parameter, and the global tempo of the score (tempo can be changed in real-time through a slider).

By default, cursors start altogether, but delays can be applied to each of them.

As soon as a cursor reaches an object, it triggers a specific event:
Curves are interpreted at cursor's speed, and regular interpolated values are sent to PureData (Pd) via UDP. Curve interpretation depends on cursor's incidency.
These values will be used by Pd to generate a sound, or as a control parameter.
Triggers will send UDP signal to the concerned host, to initialize playback of their assigned media file.

Playback can be finite, or infinite, in which case cursors will go forward/backward on their trajectories, or loop in circles/ellipses.

3.4. GUI features

3.4.1. Graphical composition

IanniX offers a rich graphical interface for composing, leaving scripting part in background. IanniX has a MDI (Multiple Document Interface), which allows the editing of various score in a same window. A tabbed system has been settled to manage it. The graphical composer consists in an OpenGL window, with special interactions on it. Those interactions can be controlled through mouse /keyboard combination, external devices (sensors, MIDI controllers...) and human gesture, through Augmented Mapping application.

The main interactions in graphical composition are: Interactive Edition on objects (Passive/active selection, Copy/Cut/Paste/Delete, Drag & Drop, Edit properties, B-Spline curves edition through control points).
Guided Edition, a feature aiming to keep the user «on track», because a score can be huge (in space) and it would be easy to get lost in it. By default, the selected object automatically appears in the center of the screen.

3.5. GUI Screenshots

Some screenshots of the first GUI implementation are given here as an example:

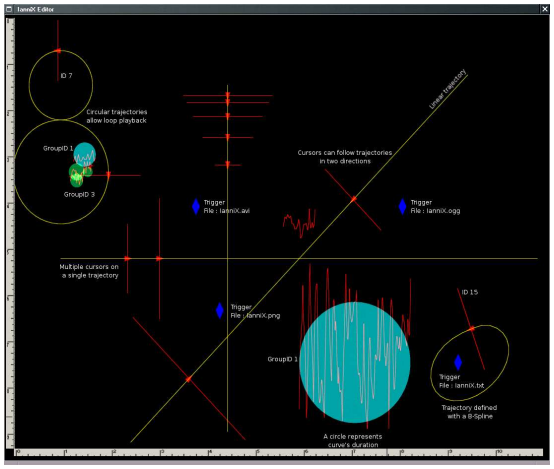


Figure 2. Score

This score showcases all kind of IanniX objects, with many comments on it.
It shows various shapes of trajectories:

- Three linear trajectories, with various angles
- Two circular trajectories (for loop playback)
- A B-Spline curve (bottom right), user-defined with control points.

On these trajectories, many cursors have been assigned with various widths and in both playback directions. Some cursors identifiers are displayed. Score contains four triggers, each of them pointing to a multimedia file (wav, ogg, txt, png). IanniX doesn't play those files; it only starts their playback by sending an UDP message to the concerned host. Six curves are also displayed:

- Four curves on the left, displayed with their symbolic circle, a big one in bottom right.
- Group identifiers are displayed (those groups are also determined by circles colour)
- A single curve, without any circle.

Those curves have been randomly generated

The object editor window controlling object's creation. Every object's property is filliable here: identifiers, graphical properties, positioning, network properties.

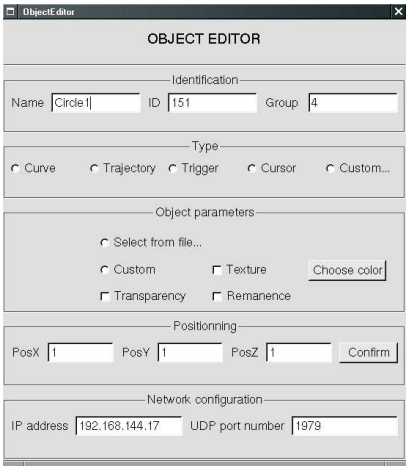


Figure. 3 Object editor

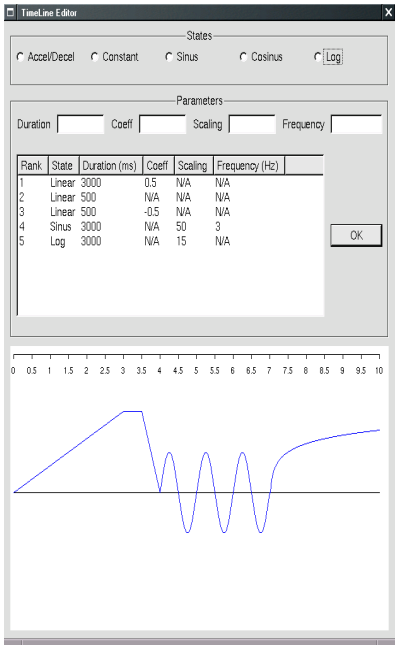
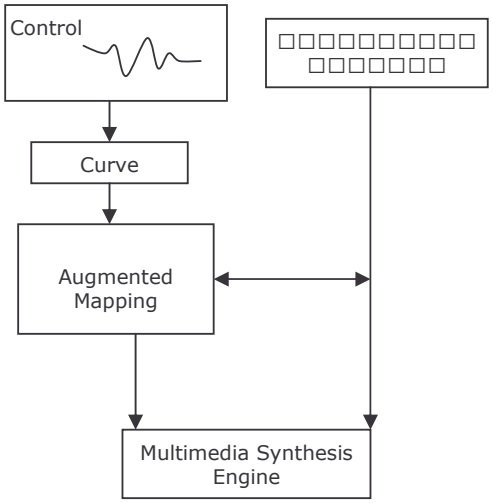


Figure 4. Timeline editor

4. AUGMENTED MAPPING



One of the most delicate problems in such a complex system as IanniX is to allow a satisfying mapping scheme between the controllers and the controlled media. Augmented Mapping is the black box between the user-interface and event generators. The ideal mapping system in IanniX would recognize its user's mapping preferences, by structuring them the user would attempt to build its own mapping scheme.

The system takes as an input the object's properties (which are described in a XML file) and controls information (typically a graph representing the "artistic gesture") and must be able to generate relevant information for the control of multimedia material. The first module will be processing the graphical information. By analyzing the structure of the curve, properties such as periodicity, stiffness, regularity, etc. will be extracted. Once the "meaning" of the gesture has been learned, this control information must, in turn, correspond to relevant synthesis parameters. This is the role of the mapping module. It will take into account both the properties from the score object and the curve description. Finally, the system will use the OSC protocol to communicate with the Multimedia Synthesis Engine (Pd or Max-MSP), and everything will be taken care of to allow for an easy insertion of pre-existing patches in the IanniX scheme.

In this manner and contrary to existing software, IanniX would not impose a certain aesthetic to the user and would rather define itself corresponding to its user.

5. OPEN APPLICATION

5.1. Client / Server Architecture

A kernel program, managing data and networking, is running as a server. When this server is launched, it is possible for it to communicate with third-party programs, through the same protocol (UDP / OSC) (for example, a IanniX GUI client, or VirChor). The GUI client handles only display and graphical interactions, while the kernel controls data structures, network streams and scripting.

5.2. Customization: Import / Export of Graphical Profiles

A score's appearance can be fully customized with a special configuration menu, allowing users to choose object properties (shape, colour), or to define classes of objects (using some criteria's). These customizations can be saved or exported in « profiles » which can be applied to any other IanniX score. Those profiles are saved in XSL style sheets. A user can also import profiles from other users and apply them to his personal documents. This aspect gives the graphical editor a great flexibility, and may also develop exchanges between composers. Besides, XSL style sheets are also used to customize VirChor render. They can also be saved, imported or exported.

5.3. Scripting

A score is edited through the graphical editor and in the meantime an XML script is generated automatically in background. Advanced users will be able to edit this script manually through an editor. XML was chosen because of its portability and it is a real standard, nowadays. It alos simplifies compatibility with VirChor, also using XML scripts.

5.4. UDP / OpenSoundControl (OSC) for networking

This is a quite powerful protocol managing connections between computers, synthesisers or multimedia devices. As programs like VirChor or PureData were already using it, IanniX was also designed for using OSC.

5.5. Database

IanniX provides a database module, allowing access to remote data stored in a SQL database. As a consequence, these data could be then imported in an object definition.
The databases can be local or remote, accessible via UDP connections.

6. CONCLUSIONS

The future major direction for IanniX is of course the Augmented Mapping, an aspect which is currently in development at La Kitchen.
Augmented Mapping is currently the center of interest of a 3-year thesis, lead by researcher Sylvain Le Groux.

7. PERSPECTIVES

Here are the scheduled perspectives for IanniX, which may be available soon :

- **Augmented Mapping** module, available within 3 years.
- Evolution of the architecture, in order to allow **plug-ins.**
- IanniX Win32 Release, using an alternative Qt SDK version, made by KDE developers. Effectively, native Qt SDK is non-GPL, and we had to use a GPL version, which is basically a Win32 port of Linux distribution.
- Easy installation packages, for each platform (Linux, MacOS, Win32).
- Adding a IanniX repository on SourceForge.net.

8. REFERENCES

[1] Iannis Xenakis http://www.iannis-xenakis.org

[2] CCMIX http://www.ccmix.com/indexfr.shtml

[3] Marc Battier http://homestudio.thing.net/revue/content/asr1p20.html

[4] Philippe Caillaud http://bibliofr.info.unicaen.fr/bnum/jelec/Solaris/d07/7caillaud-aiff.html

[5] La kitchen http://www.la-kitchen.fr/IANNIX/iannix.html

[6] Corlaix Omer, Gallet Bastien, « Entretien avec Iannis Xenakis », Musica Falsa n°2, Paris, 1998, p. 29.

[7] Colyer Cornelia, "Centre d'Etudes de Mathématiques et d'Automatiques Musicales", ICMC Proceedings, 1986, p. 317-319.

[8] Marino Gérard, Raczinski Jean-Michel, Serra Marie-Hélène, "The New UPIC System", ICMC Proceedings, Glascow, 1990, p. 249-252.

[9] Raczinski Jean-Michel, Marino Gérard, "New UPIC System Demonstration", ICMC Proceedings, Montreal, 1991, p. 567-570.