"terminal rasa" - every music begins with silence

Frank EICKHOFF
Media-Art, University of Arts and Design
Lorenz 15
76135 Karlsruhe,
Germany,
feickhof@hfg-karlsruhe.de

Abstract

An important question in software development is: How can the user interact with the software? What is the concept of the interface? You can analyze the "interface problem" from two perspectives. One is the perspective of the software developer. He knows the main features of the software. From that point he can decide what the interface should look like, or which areas should be open for access. On the other side is the perspective of the user. The user wants an interface with special features.

The questions for audio software designed for live performance are: What should the program sound like? If software for live performance should have features like an instrument, what features does an acoustic instrument have, and what features should a computer music instrument have? The first part of this paper is concerned with music and sound, the special attributes of acoustic instruments, and the features of an average Personal Computer. The second part of the paper presents the audio software project "fui" as a solution to the aforementioned questions and problems.

Keywords

computer music instrument, interface problem

1 Introduction

The "interface problem" is a very important aspect in audio software development. The interface of a machine is not the device itself, but rather the parts of the machine which are used for interaction and exchange; the area between the inner and the outer world of the machine. The "interface problem" is the problem of interaction between human and machine. For an instrument, it is the area between sound production and sound modulation. Sound is produced through specific methods or physical phenomena. These methods of sound production can be controlled through the manipulation of various parameters. These parameters are the values which should be open for access by the user. It is possible to draw conclusions from the analysis of sound to the possibilities of sound modu-lation, which is interaction. Therefore, the first item to consider is music.

2 Computer Music Instrument? Music - Instrument - Computer

Silence, noise, and sound are the most basic elements of the phenomenon that is music. What music one wants to hear is an individual decision. Each has his own likes and dislikes. In the first place, music is a matter of taste. An instrument (acoustic or electronic) is a tool or a machine to make music. Any of these tools are designed with a special intention. The basis of this intention is a certain idea of sound and timbre. One can say that the instrument is a mechanical or electronical construcion of a sound idea. What should my instrument sound like? How can I construct this sound?

2.1 Acoustic Sound

Sound is nothing more than pressure differences in the air. One can hear sound, but one can not easily see it or touch it. The behavior of sound in a space is complex and depends on the physical properties of the space. Thus, any visual representation of sound must remain abstract, and is necessarily a simplified model of the real situation. The special character of sound is that one can NOT see it.

2.2 Digital Sound

A computer calculates a chain of numbers which can be played by a soundcard. Acoustic waves are simulated by combinations of algorithms. Such mathematical processes are abstract and not visible. An audio application can run within a shell process or even as a background process. It does not require any visual or even statistical feedback.

2.3 Instrument = Sound + Interface

At the point where one wants direct access to the sound manipulating parameters of his software or instrument, one needs some sort of an

interface. The construction of the interface is derived on one hand from the timbre of the sound. On the other hand, the interface has influence on the playability of the instrument and, thus, on the sound aesthetic. The instrument is the connection of sound with an interface.

2.3.1 Classic, Acoustic Instruments

A classical instrument like the violin or the piano is very old compared to the computer. The structure and operation of acoustic instruments has been optimized through years of usage. One could say, then, that the instrument has a balance between optimized playability and a characteristic tone colour / timbre. Every instrument has its own unique sound.

2.3.2 Universal Computer

From the start the computer was developed as a universal flexible calculating machine. The "universal computer" works with calculating operations and algorithms. Alan Turing proved with his invention of the "Turing machine" that every problem which could be mechanized can be solved by a computer calculation[1]. Otherwise the computer ends up in an infinite loop and without result. The "Turing machine" does not stop. It is obvious that the computer can solve a huge amount of problems.

The computer interface is divided into hardware and software interface. The hardware setup of an ordinary personal computer is a keyboard, a monitor and a mouse. Software interfaces are programs which can interact with such hardware. The clarification of this concept makes it easy to deal with the complex possibilities of the computer.

2.4 Computer Music Instrument

When one wants to use the computer as an instrument, one must combine the features of an instrument with the features of a computer. One needs to create a balance between playability, unique sound and the special character of the computer, that is flexibility:

$$\boxed{\text{Sound vs Playability vs Flexibility}}$$

3 The "fui" Audio Application

The audio application "fui" is a sample loop sequencer. The program is designed for "live

performance", as such it is playable like an instrument. It is a simple tool to create short rhythmic loops. It has a minimal sound characteristic and serial or linear rhythmic aesthetic. The user has two different interfaces. One is a terminal for keyboard commands. The other is a graphic window with a GUI (Graphical User Interface) for the interaction with the mouse.

3.1 Short Description

The user can load audio samples into a sequence. Such sequences are played in a loop directly. He can move such samples to a specific point in time within a sequence. Samples are dragged and moved multiple times until the music gets interesting. With this method it is easy to construct rhythmic patterns. Every sample can be modulated through the control of different parameters (Filter, TimePitch, PitchShift, Incremental or Decremental Loop). It is possible to create multiple sequences, and to switch between them in a song like manner. Because of the playback of the loops, the user gets a direct response to all the changes he or she makes. The music develops through improvising and listening.

3.1.1 Sound Effects

Every sample can be modulated with different effects. The effect parameters are both static and random. The "pitch" control allows the user to manipulate the pitch of the sample. The "position" is the playback start value within a sample. "Loop" restarts the sample at the end and "count" is the number of repeats. "Incremental loop" or "decremental loop" starts the sample at the "position" point, and the sample length gets shorter or longer after every repeat.

3.2 Interaction - Interface

The "fui" application uses the standard interfaces of an ordinary computer. Every interface has its advantages and disadvantages. The terminal program is specialized on keyboard control. The GUI is specialized on mouse control. The "fui" application uses both features (see Figure 1).
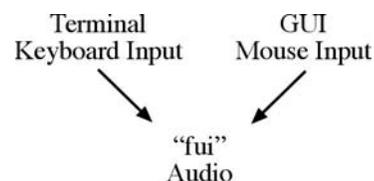

Figure 1: "fui" Interface

---

[1]"On computable Numbers and an Application to the Entscheidungsproblem", Alan Turing, 1947

3.2.1 Terminal

Before the invention of the desktop computer with GUI control there was only a terminal. The terminal is one of the oldest software interfaces to the processes of the computer. The terminal works perfectly as an interface because it is incorporated on many operating systems. It operates simply on keyboard input and text output. It is difficult to implement a comparable interface in a mouse orientated GUI. When there is a terminal anyway, why shouldn't we use it?

3.2.2 "pet" - Pseudo emulated Terminal

The first thing which is launched by "fui" is "pet" (pseudo emulated terminal). The idea behind "pet" is to use the terminal as a keyboard input and text output interface during the runtime of the program. This object uses the standard streams (stdout, stdin) for reading and writing. The user can type in simple UNIX like commands (see Table 1) for file browsing, changing the working directory, loading files into the "fui" software.

The "ls" command prints out a list of the current directory. The "pet" object numbers all files and folders in the directory (see Figure 2).

```
pet: /Users/f/ > ls
1 Desktop/
2 Documents/
3 test.txt
pet: /Users/f/ > █
```

Figure 2: list Directory

The "get" command loads a filename into the "pet" command parser. The command argument is the filename or the number printed out with "ls" (see Figure 3). Some other commands like "cd" use the same method of file identification. This method provides a simple and fast way to load files or browse directories.

```
pet: /Users/f/ > get 3
getting "test.txt"
pet: /Users/f/ > get test.txt
getting "test.txt"
pet: /Users/f/ > █
```

Figure 3: get Filename

3.2.3 "pet" and "fui"

The "fui" software uses "pet" file loading and file browsing. The "pet" object numbers all file in a directory chronologically. When the user loads a sample or creates a new sequence "fui"

creates index numbers. The sample-ID is "ID" and the sequence-ID is "SID". For example, when the user wants to call a specific sample he has to know the sample-ID. The command "la" prints out a list with all sample filenames, information about position, pitch and IDs of the current sequence.

cd PATH or NUM	change directory
ls	list directory,
	files are numbered
start	start audio
stop	stop audio
open	open GUI
load 'name'	load sequence
save 'name'	save sequence
new	new sequence,
	generates SID
dels	delete sequence
la	list all samples,
	with ID and SID
get NAME or NUM	load sample
del ID	delete sample
seq SID	set current sequence
loop TIME	set loop time (ms)
loff ID	loop off
lon ID	infinite loop on
lr ID	random loop
ld ID POS NUM	decremental loop
li ID POS NUM	incremental loop
pi ID VALUE	set pitch value

Table 1: "pet" Commands

3.2.4 Graphic User Interface

Sometimes the possibility of visualization, graphical feedback of statistical values, or interaction is very useful. The "fui" GUI is rendered in OpenGL and has a very simple design. There are text buttons (strings which function like a button), text strings without any interactivity and variable numbers to adjust parameters with the mouse. Every control is listed in a simple menu (see Figure 4). Some text buttons have multiple states. Active GUI elements are rendered in black, inactive elements are grey (see Figure 5).

A vertical, dotted line is the loop cursor. The cursor changes the position from left to right, analog to the current time position of the loop. Audio samples are drawn as rectangles (see Figure 6).

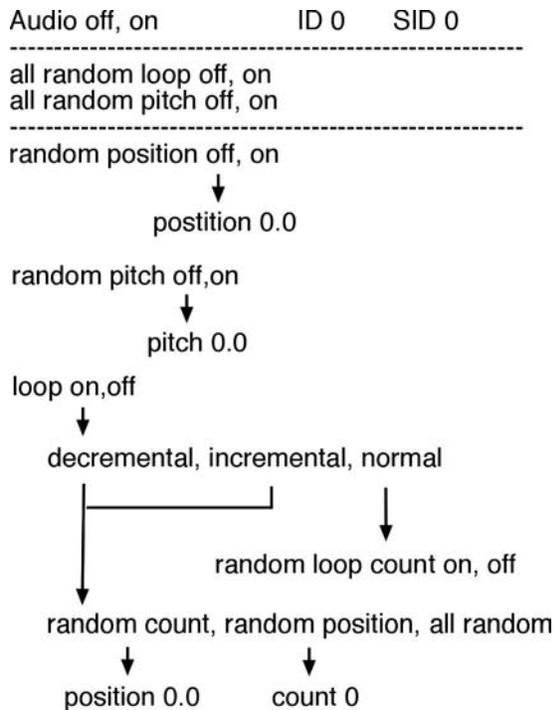The width of the rectangle is proportional to the length of the sample and the length of the

```
Audio off, on                    ID 0      SID 0
-----------------------------------------------------
all random loop off, on
all random pitch off, on
-----------------------------------------------------
random position off, on
                         ↓
                   postition 0.0

random pitch off,on
                         ↓
                   pitch 0.0

loop on,off
             ↓
      decremental, incremental, normal

                                        ↓
                       random loop count on, off

      random count, random position, all random
             ↓                    ↓
      position 0.0            count 0
```

Figure 4: GUI Menu

Figure 5: GUI

sample loop in seconds which is the width of the window. The sample can be moved with the mouse within a two dimensional area (like a desktop, table or "tabula").

Every new sequence has a blank area, a blank table ("tabula rasa").

3.3 Example Usage

Every music begins with silence. "fui" starts as a simple terminal application without any other window or GUI ("terminal rasa"). After startup the software waits for command line input (see Figure 7).

The "open" command opens the GUI window. The "new" command creates a new, empty sequence. "fui" adds a new number to the "SID LIST" in the GUI window. This number is the new ID for the current sequence. The "start" command starts the audio playback. The loop cursor starts moving over the
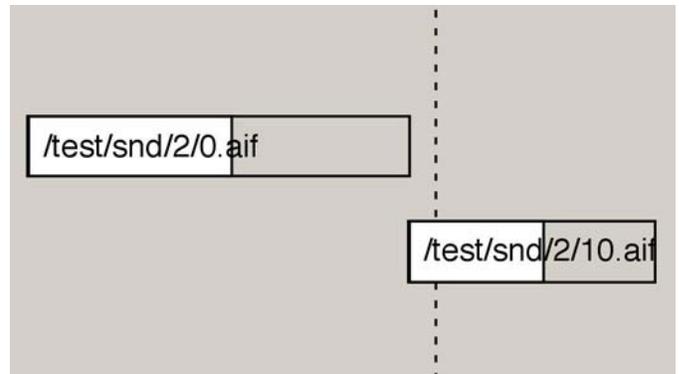

Figure 6: Vertical Cursor and two Samples

```
starting fui...
------------------------------------------------
init RtAudio API
device: in 0, out 0
chans 2, srate 44100, bsize 512, nobuff 16
------------------------------------------------
"pet - pseudo emulated terminal" (try 'help')
change to current directory '/Users/f'
pet: /Users/f/ > ■
```

Figure 7: Start Screen - "terminal rasa"

window. Now, the user can browse the harddisk for suitable samples. The "get" command loads a sample into the sequence. The user can move the sample to a position within the GUI window. Every time the cursor reaches the sample, the sample will be played (see Figure 8).

3.4 Sound - Playability - Flexibility

Sound, playability and flexibility have a mutual influence on each other. The sound is determined by the implementation of audio playback and audio manipulation. Many interesting rhythms can be found by improvising and playing with the software. Different interfaces and the implementation of a powerful audio engine enhance the flexibility of "fui".

3.4.1 Sound

The characteristic "fui" sound comes from the combination of short samples into rhythmic loops. All samples are freely arranged within the time frame of one sequence. There are no restrictions imposed by time grids or "bpm" (beats per minute) tempo parameters. The user has a simple visualization and a direct audio playback.

3.4.2 Playability

The use of the UNIX like terminal and the simple GUI provide a simple and playful access to the software. Different sound effects with

static or random modulation vary the sound. All changes are made intuitively by the user through listening. For example, a combination of two samples which might sound boring at first, can become very interesting with slight changes to the position of one sample within the time frame of the loop. A simple change of one parameter can have an interesting result in the music.

3.4.3 Flexibility - Audio API?

In the first place the source code should be portable. This project was developed on an Apple Macintosh PISMO, G3, 500 Mhz, OSX 10.3 using the gcc compiler. Later it was ported to Linux. The whole project was written in ANSI C/C++ with OpenGL for graphic rendering. The "Software Toolkit"[2] from Perry Cook and Gary Scavone is used for realtime audio file streaming. The platform independent window class from "plib"[3] is used for creating the render window.

Different audio engines are tested for the main audio streaming:

"RtAudio"[4] from Gary Scavone, "Portaudio"[5] from Ross Bencina, "FMOD"[6] from Firelight Technologies and "JACK"[7] from Paul Davis "and others".

The "JACK" API works as a sound server within the operating system. Completely different audio applications, which are compiled as "JACK" clients, can share audio streams with each other. Now the developer does not need to think about implementing some kind of plug-in architecture in the software. Audio streams can easily be shared in a routing program. It is simply perfect for audio software developers. From that point the use of the "JACK" API is the most flexible solution for the "fui" audio project.

4 Conclusions

Music is meant to be listened to. The idea of "fui" is to establish a balance between the interface and the characteristic sound of the computer as a musical instrument. When one is familiar with the special features and the historical background of acoustic instruments and computers in music AND the general differences between the two, it is possible to say that the ideal combination of both media is a hybrid and open environment. The design of the interface is simple, minimal and experimental. The sound aesthetic is linear with nested rhythmic patterns. The user deals with the program in a playful way and the music is created through listening.

5 Acknowledgements

6 Project Webpage

http://www.theangryyoungcomputers.de/fui

---

[2]http://ccrma.stanford.edu/software/stk/

[3]http://plib.sourceforge.net/

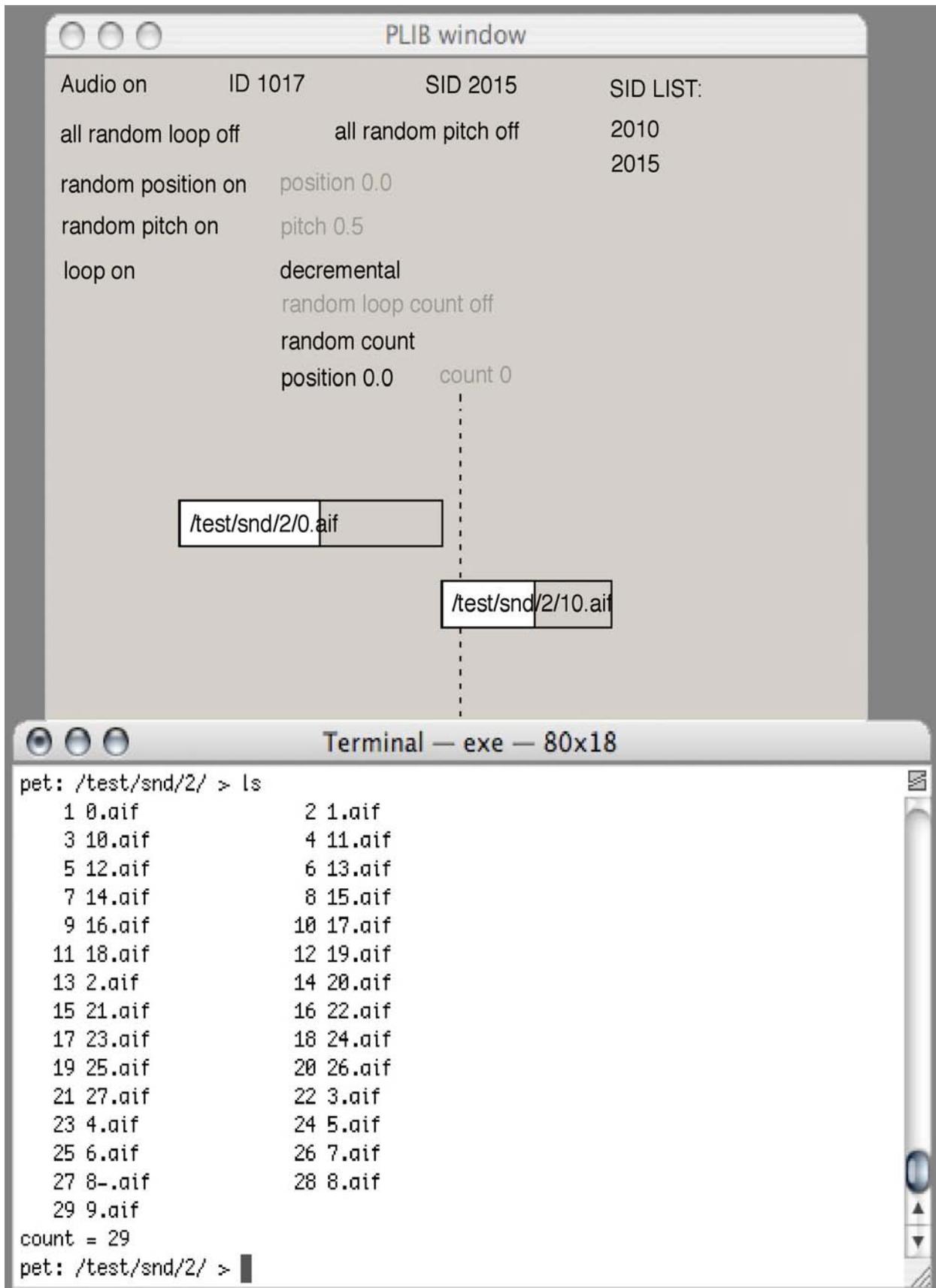[4]http://music.mcgill.ca/

[5]http://www.portaudio.com

[6]http://www.fmod.org

[7]http://jackit.sourceforge.net/

Figure 8: "fui" Screenshot