Composing a piece for piano and electronics on Linux

Lorenzo Franco Sutton
Rome, Italy
lsutton@libero.it

Abstract

In this paper the author reports his experience about a rather complex music-creation scenario using Linux: successful composition of a piece for piano and electronics using Free/Libre and Open Source Software. The whole workflow, from composition to recording and final production of a high-quality printed score, is presented describing the chosen tools, adopted strategies, issues and the overall experience.

Keywords

electronic music, music composition, piano, FLOSS, Linux

1 Introduction

In 2003 Daniel James concluded an overview on Linux audio software in *Sound On Sound* magazine stating that those were probably still “early days for Linux desktop audio applications”, nonetheless he was also optimistic about the future of Free/Libre and Open Source Software (FLOSS) in the music and audio domains [1]. His prediction seems to have proven true: today Linux appears mature enough for supporting music creation and production and is now widely utilised by a wide spectrum of users ranging from home musicians to professional music studios.

In the field of electronic art music it seems that on the one hand many academic institutions dealing with computer music – such as research centres, universities and conservatories – are fully encouraging the use of Linux and Open Source. On the other hand it appears that, from the author's experience, the majority of Italian conservatoire teachers and students are still using other operating systems and closed-source software, especially in the composition domain. In 2009 the author started working on a piece for piano and electronics[1] for a conservatoire assignment in electronic music composition. He initially started working on Windows but was soon determined to undertake a challenge and decided to exclusively use Linux for the entire composition and creation workflow, even though he was the only Linux user in his class. The objective was successfully achieved by dividing the whole workflow into sub-tasks using specific software for specific jobs, addressing arising issues in a precise and focused manner. The described approach is quite common in the FLOSS world and related to the Unix philosophy of having “each program do one thing well” [2], as opposite to the 'one big software does it all' concept sometimes seen in the multimedia domain.

2 Background: the piece

Open Cluster [3] is a piece for live piano and electronics composed in 2009 and partly revised in 2010. It started in 2009 as an assignment under the guidance of Alessandro Cipriani of the Conservatoire of Frosinone and was further developed in 2010 by the author.

In astronomy an open cluster is a group of stars loosely bound to each other by gravitational attraction [4]. In music a cluster is a chord which has three or more consecutive notes. The initial idea for the piece was to freely explore 9-note series, called “constellations”, on the piano. These series, often presented in clusters, are the main components and formal construction pieces of the piano part. The piece is conceived for a live player interacting with a fixed electronic part, the latter being created by using exclusively sounds from the piano part. The author's idea was to enable a performer to engage in an interplay between the part he/she plays and the electronics, with the performer always being encouraged to “play” (in the broadest meaning of the word).

[1] As opposite to 'live electronics' this is still often referred as the 'tape' mostly for historical reasons, meaning that the electronic part is fixed and played back along with the live performance.

3 Workflow for the composition

In the creation workflow for *Open Cluster* the four main tasks were: 1. Composition and scoring of the piano part. 2. Production of a good quality MIDI performance of the piano part for creation of the electronic part,[2] rendered to audio. 3. Audio recording of the whole composition (piano and electronics). 4. A final, complete score with both the piano and electronics ready for high quality printing.
In the following details on how each step was tackled are described. Figure 1 shows a diagram of the general workflow, and the main software interactions within it.

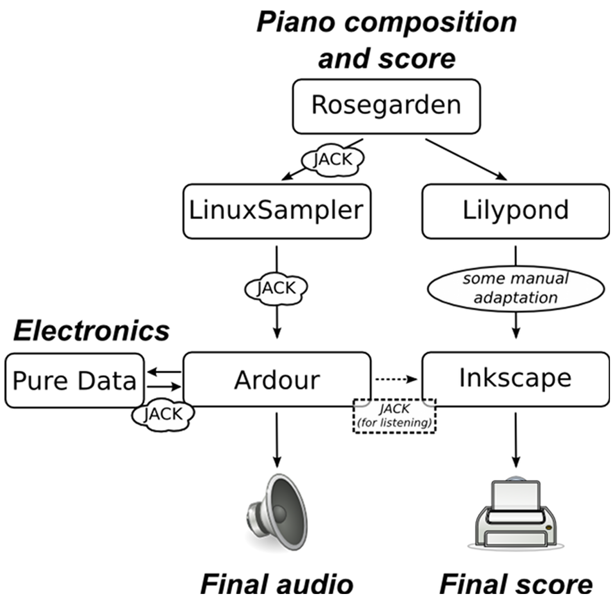

Figure 1 General workflow diagram with the main software interactions

4 Composition and scoring of the piano part

The author chose Rosegarden MIDI sequencer [5] as a composition tool eventually using LinuxSampler [6] and the 'Maestro Concert Grand' sample library for the sampled piano library [7]. Rosegarden was chosen because of its rich notation-editing features, MIDI sequencer capabilities and the ability to export to Lilypond – a high-quality music engraving system [8]. In fact in this situation the author felt the need to have a tool that could on the one hand offer effective notation writing – through the QWERTY keyboard – and on the other hand capable of playing the results, as well as providing rich MIDI editing features. Rosegarden does offer the possibility to use many soft-synths internally and the Fluydsynth DSSI was initially used for early experimenting with SoundFonts. Eventually LinuxSampler was used together with QSampler as a graphical front-end: in this way a high quality piano sample library, chosen as the preferred 'virtual instrument',[3] could be used since the beginning. Rosegarden easily connects to LinuxSampler through JACK [9].[4] JACK is a very efficient software for handling audio and MIDI connections among different music software, essentially allowing one to interconnect them and communicate with one another. Additionally it offers a transport mechanism to synchronise playback operations.[5]

Because Rosegarden doesn't natively support two-staff piano scoring [10] the chosen approach was to use two separate tracks for left and right hand, and then undertake full piano notation directly in Lilypond once the composition process was completed. To ease synchronisation with the electronic part, the piece is written in 4/4 with a metronomic tempo of 240 BPM for the crotchet, which results in one measure per second. The piano 'constellations' had been chosen in advance by the author and the whole composition process took place in Rosegarden. The setup was very adequate and comfortable.

5 Creation of the electronic part

Once the piano part was finalised a full performance was recorded in Ardour, which had been chosen as the main environment to create the electronic part. Ardour is a full-featured Digital Audio Workstation allowing for professional grade multi-track audio editing [11]. Recording into Ardour was easily achieved by directly connecting QSampler's audio outputs to a stereo audio track in Ardour, again through JACK. As explained earlier, the author wanted to use exclusively sounds from the piano performance for the electronic part so

[2] Ideally a live performance and recording, but this was not possible due to practical constraints.

[3] In fact the author is not a fully trained pianist and didn't have the opportunity to work with a performer.

[4] Controlled via QjackCtl: http://qjackctl.sourceforge.net/

[5] For a more precise and technical in-depth description refer to the JACK homepage in the references.

many of Ardour's editing features were put to work to layer, cut, collage, etc. pieces of the piano part and processes them with the many effects Ardour offers.[6] Pure Data, a "real-time graphical programming environment for audio, video, and graphical processing" [12], was extensively used for manipulation of the sound both by connecting the software directly through JACK and working with it separately. For example a patch for Pure Data developed by the author [13], which enables minimalistic granulation of audio files, was used for creating material of the electronic part.

The advantage of using JACK to seamlessly connect all the various applications is evident: all audio and MIDI could easily be routed from one software to the other in a very flexible and efficient manner.

6 Audio Rendering of the complete piece

Once the electronic part was concluded, both the piano recording and the electronics were saved to a separate Ardour session, so as to have a kind of master session, and simply exported to a final wave file. This was the final recording of the complete piece.

7 Creation of the full score

The full score for Open Cluster consists of a piano part and an 'expressionistic' representation of the electronic part. The author decided to use this representation because on the one hand the piano performance should be precise enough to match specific events in the electronic part, on the other hand some degree of liberty is foreseen, especially in moments were the piano part is prominent or where the electronics constitute more of a background.

Because of the mixed nature of the score, comprising both traditional music notation and graphics, the author decided to use specialised tools for each of the tasks: Lilypond for the music notation, Inkscape for the graphics. The jEdit text editor [14] with the LilyPondTool plugin [15] was used for editing of the LilyPond source file. The left and right hand parts were kept in two separate files for easier editing and future adaptation.

[6] Ardour natively supports the Linux Audio Developer's Simple Plugin API (LADSPA) effects, which are a de facto standard in Linux as well as other plugin formats such as LV2. See www.ladspa.org

Because the electronics representation was to be stacked vertically below the piano staff, enough space below each staff had to be ensured. No straightforward way of achieving this was found so, after digging into the excellent Lilypond documentation, the author came up with the solution of adding a dummy staff to the overall score: this is an additional staff added three times, with all notes hidden through the `\hideNotes` directive and all staff symbols, such as Clef. TimeSignature. etc., set to be transparent through a series of override commands. The general structure of the Lilypond `\score` section is the following:

```
\score
 {
<<
 \new StaffGroup
 <<
   % Right Hand
   \newStaff {\include "rightHand.ly"}
   % Left Hand
   \newStaff {\include"leftHand.ly"}
  >>
  % dummy space below the piano part
  \new Staff
  {
   % includes the file 3 times
   ...
  }
>>
```

LilyPond is able to generate scores in SVG format [16].[7] These in turn can be opened by Inkscape. Two issues arose when opening the generated SVG file in Inkscape. Firstly a known bug in Inkscape 0.46 (which was being used at the time) caused some elements not to show up properly [17]: the issue was solved by systematically correcting the SVG source as suggested by the Lilypond documentation. Secondly, at the time of score creation Lilypond was exporting to multipage SVG,[8] which Inkscape doesn't support [18]; this was resolved by following a suggestion from the Lilypond mailing list [19]: the pages were manually split to multiple files by editing the SVG XML source and eventually a unique page created by importing the separate files in Inkscape and having them all on the same drawing area. Clearly this is not a very

[7] The current Lilypond SVG back-end underwent a series of changes since the version used for this work.

[8] This behaviour seems to differ in different versions: in fact some versions create a file per page.

straight-forward procedure, but the recent enhancements to the Lilypond SVG backend and possible changes to the Inkscape multi-page issue status may improve the situation.

During creation of the graphics for the electronic part, the final Ardour session was kept open in the background and controlled via QjackCtl through the Jack Transport mechanism. This allowed to control Ardour's playback and quickly move through measures, replay sections etc. In fact the author was drawing the part while listening to it and precisely synchronising some of the graphical elements with the piano part.

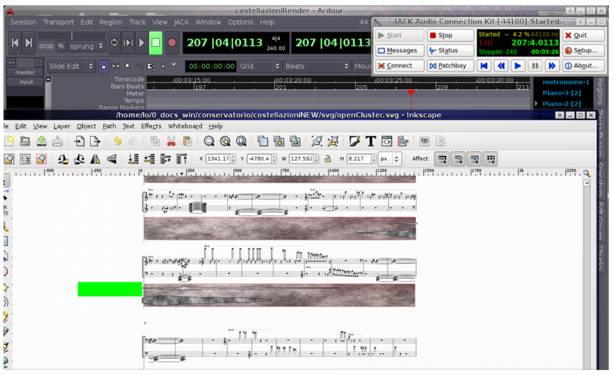

Figure 2. A screenshot with the score in Inkscape. At the top QjackCtl and in the background Ardour playing

As a usability note the ability in the GNOME desktop environment to put any window “Always On Top” was very useful, as QjackCtl (which consumes small screen estate) was always visible and used as playback interface while working in Inkscape.

Once the complete score was ready each page was exported to a single PNG file at 600 DPI (A3 paper size). Combining these into a single PDF file was easily achieved with the ImageMagick graphics manipulation suite using the *convert* command. The PDF was then taken to a print shop for final printing.

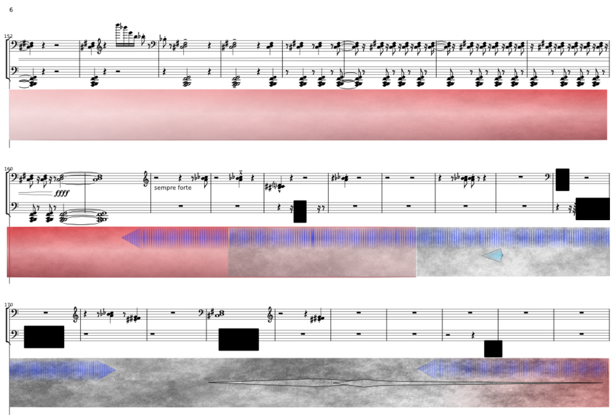

Figure 3. A page from the final score. The black 'rectangles' are clusters

8 Conclusions

The successful accomplishment of a complex music creation task using Linux and Free/Libre and Open Source Software tools was presented. Clearly, this is only one possible path the author chose as particularly suited to his needs. The presented workflow shows that a modular approach, using specific software for the specific jobs versus the 'one software does it all' paradigm, proves to be an effective strategy enabling one to concentrate on each task and tackle possible issues separately. Linux as an operating system and the Free/Libre and Open Source Software presented show to be mature enough to support such kind of tasks. Some issues arose especially in the graphics-related activities for score creation, but it's fair to say that this isn't a particularly standard task in music creation: additionally the issues were overcome thanks to good documentation and community support (e.g. one of the software's mailing lists). The presented scenario is rather complex and certainly non-standard compared to other music production and composition ones, and will hopefully be of inspiration and use for anyone working in similar fields, such as electronic music or non-standard score production, who is considering Linux as an operating system for their creative needs.

9 Acknowledgements

The author would like to acknowledge M. Alessandro Cipriani for support and helpful advice during the composition of the piece. He would also like to thank the Linux Audio Users and

Developers community for continuous support and inspiration as well as the developers of all the great software mentioned in this paper. A special thanks goes to the *GnuFunk* members for their support.

References

[1] Daniel James. 2003. Open Source Music Software & The AGNULA Project. In *Sound On Sound*. www.soundonsound.com/sos/feb03/articles/linuxaudio.asp

[2] The Bell System Technical Journal. Bell Laboratories. M. D. McIlroy, E. N. Pinson, and B. A. Tague. “Unix Time-Sharing System Forward”. 1978. 57 (6, part 2). p. 1902. cited in Eric S. Raymond. 2003. *The Art of Unix Programming,* chapter 1 section 6 www.faqs.org/docs/artu/ch01s06.html

[3] Lorenzo F. Sutton. 2009-2010. Open Cluster - For piano and electronics. Score and audio available at:

http://lorenzosu.altervista.org/music_sound/openCluster/

[4] “Open Cluster”. *Wikipedia, The Free Encyclopedia*. Wikimedia Foundation - http://en.wikipedia.org/wiki/Open_cluster Retrived on 14 Feb 2011

[5] Rosegarden – www.rosegardenmusic.com

[6] LinuxSampler - www.linuxsampler.org

[7] Mats Helgesson, Maestro Concert Grand v2 Gigasampler bank. Available at www.linuxsampler.org/instruments.html

[8] LilyPond - http://lilypond.org/

[9] JACK – Jack Audio Connection Kit - http://jackaudio.org/

[10] D. Michael McIntyre. 2008. Piano Notation by Example.

[11] Ardour – Digital Audio Workstation - www.ardour.org/

[12] Pure Data - http://puredata.info/ www.rosegardenmusic.com/tutorials/supplemental/piano/

[13] Granita – minimalist granular synthesis for Pure Data. http://lorenzosu.altervista.org/pd/granita/

[14] jEdit – Programmer's Text Editor - http://www.jedit.org/

[15] *LilyPondTool for jEdit* - http://lilypondtool.blogspot.com/

[16] *SVG backend*. Lilypond Wiki. Retrived on 14 Feb 2011 - http://wiki.lilynet.net/index.php/SVG_backend

[17] Software support in *Inkscape Wiki SVG Backend* Retrived on 14 Feb 2011 http://wiki.lilynet.net/index.php/SVG_backend#Software_support

[18] Multipage support. *Inkscape Wiki*. http://wiki.inkscape.org/wiki/index.php/Multipage - Retrived 14 Feb 2011

[19] Multi-page SVG file. LilyPond user discussion mailing list. Message 27334 of 14 Jan 2007 by Vincent. http://www.mail-archive.com/lilypond-user@gnu.org/msg27334.html - Retrived on 14 Feb 2011