TDT4705 Software Engineering, Depth Study
Fall 2006

Digital art and open source

How is Open Source utilised in the demoscene?

Nicolas Mendoza
Supervisor: Letizia Jaccheri

Table of Contents

1. Abstract

The project is carried out as part of the 9th semester of a Master degree in Computer Science at the Norwegian University of Science and Technology. The project is categorized under the «TDT4705 Software Engineering, depth study» subject. The formal description that the project needs to adhere to is:

> Candidates will have to participate to one open source artistic software project in order to learn about open source and software art, improve programming skills, and contribute to the discussion around open source and software art. The candidate can influence the research focus, as well as the project to work with, the research questions will have to be grounded in the literature.

The project uses the creative subculture coined the "demoscene" as the arena to see how open source is utilised in the creation of "demos". The research questions are a result of seeing challenges open source has had in regular software areas and the evolution of software art that have been discussed in the literature for the courses. Participation although slightly less focused on, is carried out by investigating existing software, the usage of open source, and open source software that are relevant to the process of demo making. Hopefully the project can lay a foundation for future participation in a relevant art related open source project.

2. Motivation

I've been involved in the demo scene in one way or another since I got my first computer. Mostly as a bystander, collecting and enjoying demos and other creative works made by individuals and groups participating in the so-called demoscene. I have however participated in scene groups as an ascii-artist and making graphics, and the occational code contribution in groups like «Megastyle», «Braindead», «Save our Souls» and «TUFS». The latter, a demoscene group which started out as a pure open source centric group releasing all their productions as open source.

Over the years I have also been involved in the regular open source community, everything from reporting bugs, suggesting patches and porting software from one platform to another. I have used open source and modified it for different projects both privately and as work assignments for various employees. I am familiar with the nature of gift economy, distributed development, the various hierarchies and administration. I have been using typical open source systems like open bug tracking systems, public forums and mailing lists, and have adhered to the various processes and styles that different projects follow.

With this background I felt strongly motivated to do a research project combining this two subjects to do a research in how open source is being utilised in an artistic environment that the demoscene is. Having aquintances and friends in both areas hasn't been a negative factor, although it has been challenging to draw a line between friendships and proper research.

3. The demoscene

3.1. Introduction

The demoscene is a "A subculture in the computer underground culture universe, dealing with the creative and constructive side of technology ... " [32xDemoFAQ]. It's basically a virtual group of people doing creative things with their computers, like creating images, music and programs in an artistic fashion. The works seldom have any other purpose than their mere being, having a strong resembling with «real life» art, that be conventional paintings, sculptures, multimedia installations and similar.

Defining what art is, is not a task I will undergo, like Paul Fishwick states in his "An introduction to Aesthetic Computing", the term is yet to be defined and hence there are and will probably always be discussions and disagreements on what classifies as art. I therefore choose to mainly illustrate the similarities between demoscene activities and products compared to what traditionally has been accepted as art in the non-digital world.

The demoscene is a vague term, since people participating in it don't sign up anywhere or enroll in any way to be part of it. Anyone can claim they are part of "the scene", even if only participating as an observer, for instance by attending at demo parties, voting, discussing releases and similar. However the creating part of the demoscene is what I focus on here and that would be the persons working as graphic or music artists, 3d modelers, script writers, programmers, designers and the like.

3.2. History

The origins of the demo scene can be dated back to the early 80's when home computers started to be common and the game industry, and therefore the piracy communities, started to flourish. Given the nature of the media programs were distributed on, they were easy to copy and re-distribute without loss of quality. Games and applications were eventually copy-protected with various schemes, and "crackers" (hackers that cracked the protections) removed such protections and re-distributed it adding their initials or code names on the software.

This activity became a competition between the various crackers, and their cracker groups. so labelling their «work» with their initials, logos and slogans become more common to build up a name or reputation for themselves. Many times this labels were added in front of the programs for everyone to see. Slowly, more advanced intros started to unfold, moving letters, scrolling text, simple animations and later also music and sound were added. One can easily see the similarities to another emerging sub culture at that time, namely the grafitti community writing their «handles» (pseudonyms) on walls. organizing themselves in groups and becoming more advanced with more colours, and more effort into the process itself.

At some point the making of these intros became a goal itself, and for some, more interesting than the cracking of games. The demos themselves become more advanced, more graphics, more scenes and more music. People joined together in groups, cooperating together making demos. When computer users normally gathered together in clubs or copyparties, to exchange ideas, software and have fun, they also started having competitions for best demos, graphics, music etc. made with computers. The participants competed in stretching the hardware limits as much as possible in regards of nice-looking or realistic effects, as many objects on the screen at the same time, the fastest routines and so forth. The demoscene was born.

3.3. Demos

In this project I concentrate on demos, which is used as a term describing executable programs, that might include music, graphics and sound to form an artistic product. There are many names on different kinds of demos, like intros, dentros, inv(i)tros and similar, but I will consider a demo any running executable that is intended to be a contribution to the demo scene and defined as such by either their makers or the arrangers of demo competitions.

Demos have changed over time, from simple text displayed, or scrolling over the screen in the early eighties to realistic 3D animations with full-fledged music with different scenes, and in some cases careful design and directing decisions. From being something one person could do in a couple of hours to projects spanning months with several contributors working together. Just like the game industry has evolved, so has the demo scene. And the participants have naturally many strong connections between these two areas of computing.

Just like paintings, demos have had various eras with different styles, although ranging over a much smaller period. If one examines demos from various periods, one can see how they steal from eachother, and follow rules or styles that happen to be in fashion at the time. Demos have been coded in assembler, but now it's more common to use more high level languages like C and C++. Starting out by «banging the hardware» (directly modifying hardware registers) demos now also tend to use highlevel graphics and input APIs like OpenGL, DirectX, SDL and others that for instance the underlying operative system provides.

A demo group normally consists of at least one coder, or programmer, a graphician, and a musician. Many times groups have more members though, and the various releases can be done by all of them, or just part of the groups, depending on availability, time, abilities and other factors. Depending on geographical or social factors,there is also cooperation between groups and their members. In the early days, such productions were maybe made and put together over a weekend where the members met physically, while one now benefit fromthe Internet in means of communication. In many areas demoscene development resembles the methods open source software teaches.

In the beginning small demos or intros were something that were released together with applications, but later they became larger and released on different occations. Over the years demos have been released mainly at parties participating in demo competitions at demo parties like Breakpoint, Assembly, The Gathering, Kindergarden and many more. The demoscene has become even more competitive and the various groups protect their work to avoid other people using their tricks and effects. This so that one keeps its advantage and increase one's chances to win at later competitions. Prizes at this demo competitions vary but many use them as a nice side-income by spending months before a party putting together demos and hoping they win up to 4000 Euro (Assembly 2006, Prize Combined demo) [AssemblyPrizes2006].

Demos released at parties are by de-facto considered as public domain, and some parties even force participants to give up their rights or at least give the people behind the party the rights of its use for any purposes. In my experience they never demand source code or similar so it normally only affects binaries released.

4. Open source

Most people are familiar with the term open source and the open source community. The term itself is however used broadly. I will try to describe it and its history.

Traditionally open source has been used to depict software that had available source, preferably bundled together with the software, or simply because the programming language used was of an open source nature. (interpreted scripts etc.). It meant the opposite of closed source which was the traditional way of releasing software first and formost in respect of commercial and proprietary software development.

The Open Source Initiative (OSI) was started as a reaction to the Free Software Foundation and its problems reaching out to the conservative business. Raymond and Perens set out to form OSI defining Open Source as a new term bsaed on FSF's initiatives. They created a guideline that open source licenses should adhere to be labeled as such. The guidelines were based on the previously defined Debian Free Software Guidelines [Perens1999]. They were criticized for using the term open source which was watered out, but one of the arguments in favour was that the term «free» used by FSF had a much worse duality in that it could mean free of charge in addition to the meaning that is used in free software.

The guidelines set out contained a set of criteria that an open source license should meet. They focused on that the licenses should allow redistribution, modification, be non-discriminatory, source code should be available, licenses must not be restricted to aproduct or a brand, licenses should not contaminate other software and more. [OSIDefinition].

Like Perens states, the terms free software and open source has been used to mean various things. In this project I try to follow the OSI's definition which includes licenses like GPL, BSD, MPL, Apache, X and more. However if a release has viewable source I would take that into consideration, as the reasons for not licensing it with a proper license might be as simple as forgetfulness, assuming it went into public domain by releasing it and other reasons.

4.3. Open source methods

The open source and free software movements have affected the society in other ways than just the ideology. The bazaar style of doing development have been taken up by many conventional companies. Dual licensing schemes, distributed development, off-location development, collaborative methods and tools, public bug trackers, community sites, incremental and frequent updates, releases of beta software and other aspects that have always been utilised by the FOSS communities have been adopted by non-FOSS originating or -based companies and groups. I will try to see if there are identifiable signs of this inspiring the ways the demoscene operate.

5. Research hypothesises

To design a proper research goal I have proposed the following hypothesises "The usage of open source in the demoscene is marginal." and "Open source software is mainly used for cross platform development". This two hypothesises are based on years of observation by being part of the demoscene community.

I want to research the validity of the claims by investigating how open source actually is utilised when developing demos in the demoscene community. These two statements are different in that the first one suggests a usage as a percentage of a total, and the other is about a specific use of OSS. The first one is of a quantitative nature, while the second one leans towards a more qualitative nature. The question "How is open source software utilised in the demoscene?" will therefore not only encompass in what degree such software is used, but also in what way open source is being used.

By utilitization I refer to and have focused on open source software used as part of demo productions when doing the quantitative research in measuring actual usage of open source software. When I have interviewed people I have also focused on usage of tools to generate demos, like compilers, image and sound creation tools and other utilizations like open source methodology.

6. Research methods

Measuring the usage of open source can be done in several ways. I have chosen to do a two-way approach, one using a purely quantitative method by gathering data about existing productions, inspect them and analyze the data which should give an understanding on the degree of usage.

This project will mainly focus on demoproductions released the last two years. This to get a result that mirrors the current situation Since the widespread of open source methods, processes and software has been rising since it's introduction, it's natural to not go too far back in time to get a representational result. The approach will be to inspect the productions from the most popular competitions in the time period. The productions used will be the ones listed in each competition's result lists, but still depending on availability of the productions themselves. I will present the data, first off based on wether the demo uses open source or not, then what kind of open source software is used.

The second research method will be set forth by conducting a set of interviews with different subjects participating in the demoscene community and try to analyse their answers to see wether it corrolates with the quantitative findings and possibly base some conclusions on them. Originally a questionnaire sent out to as many demo scene participants as possible was planned, but due to several reasons I decided to go for an interview approach instead, even though it is harder to analyze data based on that.

6.1. Choosing a set

I have selected a set of 139 demos during the last two years. The timeframe used is to get a snapshot of the current situation. If we assume that the open source usage in the demoscene has only been increasing, and is at its peak, similar to its influence on the rest of the software area, choosing a recent period will give us an as favourable picture of open source usage as possible, leading to more material to research on. The demos are from the three biggest parties measured in amount of attendants and competition participants over a period of two years, making it 6 in total. In addition I have looked at a Norwegian party held earlier this year and its releases.

The parties are: Breakpoint '05, The Gathering '05, Assembly '05, Breakpoint '06, Solskogen '06, The Gathering '06 and Assembly '06. I have used the web site scene.org's FTP repository which contains virtually all demo releases from parties the last 20 years. I downloaded the contributions to the demo competitions in all of the mentioned parties and analyzed them by going thru their contents and looking for licenses mentioning open source, or actual source code accompanying the productions.

In addition I have looked at a web site called Unix Scene. It is an effort to create a Unix-centric demoscene. The chance of having open source material is naturally much bigger and I have used the material listed to be able to do some more in-depth research in the actual usage of open source. The site contains 234 demos listed, where some are open sourced and some closed source despite made for open source operative systems.

6.2. Designing the questionnaire

Given the suspicion that analysing existing open source demos would yield a poor result, ie. a very small perecentage of existing demos are not using open source, it was clear to me early in the process that intervieweing participants of the demo scene of their experience with open source would be beneficial. The questions are identical on all the interviews listed in the Appendices 10.1-10.6.

I tried to design questions that were open-ended so that possible unknown material could arise, instead of presenting the interviewees with alternatives and an «easy-route» when answering questions. I also tried to design them in a neutral way to avoid influencing the subject in possible ways. For instance, directly asking why open source is so little used, might both, not let the subject think about whether there actually are parts of the processes that utilize it, and possibly put the subject in a defense or attack position. Other considerations have been in mind too, and I'll try to illustrate that by explaining the thought behind the questions.

First I start out by getting some personalia, to establish who the person is and what rolehe has in the demoscene. First in a formal matter, then a more informal question [Q1] letting the subject present himself and explain his relation to the demoscene.

Then, I go quickly on to asking what kind of software the subject uses [Q2], this question is open in that it might be tools to generate material like pictures, movies or sound, in addition to be compilers and development environments orstrictly familiar libraries or APIs incorporated in a final product.

The third question [Q3] is in after-thought a bit vague, it should probably have suggested a percentage or a scale but it works currently as setting the mood both for the intervieweers and myself when analysing the answers.

When making questions that want to know more in case a previous question was answered in certain way is not easy, however I tried to use the fourth question [Q4] as a follow-up of the previous one to get a list of software I could investigate myself and of course to see i there were any trends or typical tools used.

As just mentioned, the fifth question [Q5] also would rely on a positive answer to the third question if I hadn't designed it so that it could be answered both if the subject uses open source software or not. I try to find out if there is any conscious reasoning behind the use of open source software or if it's coincidental. If not, the question gives us a bridge to the next question which is possibly a bit prejudiced in nature.

Question 6 [Q6] might be interpreted as a question to get a solution to a problem, as my supervisor has pointed out, it might be due to the background as engineer that I want to create a problem which I can then provide a solution for. However, that's not been the main point of the question. It has rather been to get the interviewees point of view on open source software and why she might think it has challenges or not.

In addition I wanted the subject to think about possible solutions that I might be able to provide in the future for instance for a thesis project, though. It might have been dangerous to declare that open source software has lack of acceptance thereby tainting the subject or (mis)guiding her to a certain direction, but I think it's evident and also proven that the acceptance is very low both looking at the numbers and the answers given, so it's acceptable to use that as a premise.

As mentioned, it might be kind of an «I will save you» approach, but it s also a «why» question in that it might be interesting to see why open source is not used according to the person interviewed. We can later see if there are common misunderstandings, perfectly valid reasons for not using it or some concrete and «fixable» reasons why.

When doing this questionnaire I was also curious in wether the subjects were using other aspects of the open source community not restricted to software usage and licenses. Could it be that the demoscene has been affected by open source development processes or methodologies just like proprietary software arenas as we've been thought in OSS lectures and seminars? [Q7]

The OSS community have from the start had a bazaar approach when it comes to development processes, both in less planning, different release processes and cycles, incremental and continuous updates, off-location distributed development, focus on collaborative tools (CVS, Subversion, Arch etc.), open bug and issue tracking systems. Are any of these in use, and could the question itself make the subject thinking about open sourceusage in a different way?

One of the suspected reasons why open source software hasn't beenwidley used in the demoscenehas been due to the competitive nature of the demoscene. Making the fastest algorithms and most impressive effects has been the goal, and the source code have mostly been guarded from others. Is this a true obervation from my side, or maybe showing off clever methods might have helped getting status in the communty? I try to pinpoint this with question 8. [Q8]

The last question [Q9] is basically a summary to investigate consistency in answers, but also get ideas and information on beneficial ways forthe demoscene to use open source software. Could there be areas that people haven't thought of using open source? Is open source already being used in areasI haven't thought of?

6.3. Investigating relevant open source

Since the project is supposed to be about being part of open source and being involved in a project I did take some time to look at some demo scene relevant software and even provide some patches back. Some of the work was done beforethe actual project started, but its experience is relevant and added to the list of open source projects that I have been involved with.

6.4. Research process

The research process spanned over the fall of 2006. It involved course lectures in the subject TDT!0 Software Technology: COTS and Open source software [TDT10] and TDT69 Artistic software: processes and products [TDT69]. The latter held by Letizia Jaccheriwhich is also my supervisor for this project. We spent most time dicsussing the syllabus articles and how software took part of digital art, in addition we visited a local event arrangedby TEKS (Trondheim Electronic Art Centre) [TEKS] called MASKIN. It had a number of lecturers and exhibitions focusing and art and machines. This was a nice experienceto see art and machine hand in hand in practice. In addition each of the TDT69 students presented our projects as partof a workshop by Trondheim matchmaking which washeld during the event. At the particular workshpo Anna Notaro talked about Digital authorship which gave us insight in challenges of publishing and reusing material. [MatchMaking]

In TDT10 we met regularly disussing syllabus articles and had guest lecturers talking about the topics related to the articles. Since I've been actively following the open sourcearena many of the issues, challenges and topics presented were familiar. However, getting face-to-face time with knowledgeable lecturers and discussing with fellow students gave new insight and raised interesting topics. During the fall I was also privileged enough to participate at TEKNAs open source seminar held in Oslo. It lasted for two days and consisted of a number of lecturers from various aspects of open source usage. Everything from government representatives to open source based businesses. Many success stories were presented, future plans and challenges regrding the use of open source were presented. A more in-depth report about that event can be found online. [TeknaOpenSourceReport]

I quickly started to focus on an area I could take advantage ofmy previous experience and that would be relevant for the project. Writing about digital art and open source combined is a challenge, but it seemes to be that using the little known creative community called the demoscene would be an interesting setting for others in addition to myself. The demoscene is small and unknown by most people, and judging by the literature available not considered as art, not even by the demoscene community themselves in my experience. Maybe this project can reveal some aspects that have been overlooked both by the demoscene and by anybody else not familiar to it.

I have seen the open source community and their methods grow and spread over the years. It's seems it continues growing and getting adapted with nothing stopping it. Even the most conservative companies like big blue (IBM) and others seem to have embraced it and others like Microsoft have done what they can to counter its widespread. I wanted to see if the same was true in the demoscene. The commercial software area has used non-disclosure and obfuscation as their main «weapons» to guard their secrets and keep advantages, similar to how the demoscene have guarded their secrets to achieve fame and money.

By experience though, I was fairly confident that the open source community haven't had a similar impact on the demoscene and I also wanted to find out why. I designed a questionnaire that could answer some of my wonderings. The initial aim was to send out such a questionnaire to a large amount of participants to get some quantitative data to measure. This proved harder than I thought as many demoscene contributors only operate using handles (nicknames) and contact details are hard to get. Others were simply not interested and never got back, and also the designing of the questionnaire itself took more time than I anticipated. I decided to do a qualitative approach and seek out relevant subjects which ended up less than desireable, but their answers have been quite interesting and varied to be able to take note of them.

Given my suspects of little usage I wanted to measure this in some way and decided to get some hard numbers by looking at released demos and in what degree they used open source. To make the project more interesting though I added later a part that concentrated on demosthat actually used open source and try to draw some conclusions based on that.

7. Research results

7.1. Open source usage in demo making

Pouët.net, a site dedicated to collect all demoscene related release and in addition provide a community site around them, lists approximately 10700 productions in the category demo. Unfortunately the site doesn't provide a mean of sorting or filtering by licenses or wether there is open source involved in a demo or not, but at least it gives us a rough number of the number of demo releases since 1980 upto december 2006. There are probably more demos released out there, and also likely some duplicates in Pouët's database. In addition the border between wether something is a full fledged demo or another smaller production like intro, dentro, 64k, 32k and similar is somewhat fuzzy. I have decided though to use their classification, and when picking out a representative representation from the last two years I have used the demo category in the parties.

7.1.1. Overall representative usage

Of the 139 demos chosen from the three biggest parties the last two years only one of them was actually open source based. The demo is called Inductiemachine and made by the group Limp Ninja. It was released at Breakpoint 2006 and came in at 14th position in the demo competition. The demo contains a creative commons license and the executable file is provided with its source as the file main.cpp. The license goes as follows:

> This work is licensed under the Creative Commons Attribution-NonCommercial-ShareAlike 2.5 License.
>
> To view a copy of this license, visit http://creativecommons.org/licenses/by-nc-sa/2.5/ or send a letter to Creative Commons, 543 Howard Street, 5th Floor, San Francisco, California, 94105, USA.

This means that the software is licensed under the CC by-nc-sa license. This is a license originally made for content and not necessarily source code of software. It's still a valid use though. [CreativeCommonsLicenses].The legal team of the Debian GNU/Linux project has stated that none of the Creative Commons license are compatible with Debian's own guidelines for free software

[Evans1999], including the combination of attribution-noncommercal-sharealike that this demo uses. Neither of the CC licenses are listed under OSI's approved licenses either. Yet, I still want to take the production into consideration.

Looking further it seems that the actual sourcecode included does have a license included in the file itself, notably a GPL v.2.0 license. I assume then that the CC license refers to the data files included. The demo is bundled with a set of DLL files (Dynamic libraries for the Win32 architecture), yet they are binary only and there doesn't seem to be any text indicating that they are GPL, even though they most likely are. The demo should at the very least include a written offer for the source code for the GPL or LGPL parts included, it doesn't.

The binary files included are fmod.dll, jpeg.dll, libpng13.dll, SDL.dll, SDL_image.dll, zlib1.dll all of which are (L)GPL libraries reely available elsewhere,except for the fmod library. This is a dual-licensed library but in no way open source compatible [FmodLicense]. However it's not problematic to include it.

The above mentioned issues aside, we note that the demo is using SDL, Simple DirectMedia Layer library. According to its website SDL is a «Simple DirectMedia Layer is a cross-platform multimedia library designed to provide low level access to audio, keyboard, mouse, joystick, 3D hardware via OpenGL, and 2D video framebuffer.» [SDL]. The jpeg and png libraries are popular open libraries to gain easy access to images and are used by the SDL framework if requested. The SDL_image library is a sub library of SDL handling images as the name reveals.

Some other interesting findings were done when looking into the 139 productions. Even though all but one production was closed-source, several of the demos included viewable source code. There is no mention of licenses, but there are several reasons why this code is exposed. The source code found is so-called shader code. Shaders is described as the following on the 3DDrome website:

> «Simply put, shaders are small programs that are executed on a per-vertex or per-pixel basis. Vertex Shaders (also sometimes called Vertex Programs) are executed on each vertex that's processed by the graphics API. Pixel Shaders (also sometimes called Fragment Shaders or Fragment Programs) are executed on each pixel that's rendered.
>
> The powerful flexibility enabled by shaders is probably the biggest advance in real-time graphics programming in the past several years. Most modern GPUs, particularly ones

> made by NVIDIA and ATI, support vertex and pixel shaders. » [Shaders].

The reason why these set of instructions are provided as viewable sourcehas little to do with licenses, ideologies or similar, but are provided in that form due to the nature of how shaders work. The various GPUs are not binary compatible in how they treat shaders, whereas the shader language itself is more common. In addition, compilation of shader code happens at runtime in an efficient manner optimized for the GPU in question. In other words, the code is «open» merely for practical reasons.

Another demo worth mentioning is The dolphin dream by Remembrance which uses Cal3D [Cal3D] to handle character animation. It is used as a dynamic library. The license is LGPL. A quote from the readme of the demo is interesting to take notice of:

> «To animate the dolphin this demo uses spherical linear interpolation of quaternions directly controlling the animation library's quaterions for each bone. Thus i could control the cal3d library which is used for the character animation. (http://cal3d.sourceforge.net)
>
> I prefer to animate inside VSXu to be able to test and fine-tune the synchronization with the music, something i would find difficult to do if having to re-import everything from the 3d software every time, and even slower if i had to work with someone else (3d animator). Some of you might think that using and modifying an opensource library such as cal3d is lame - and yes, from a demoscene perspective I strongly agree. However VSXu is more than just a demotool. In the light of making stuff ossible for others to use on a larger scale than just for demos (VJ's and other artists), the choice of cal3d is obvious.»

The author clearly is convinced that other demoscene members might think the use and alteration of an open source piece of software is «lame», a derogatory term, referring to bad and possibly also fresh programmers of little knowledge. He also mentions its usage, in his demotool,

A demo tool is a set of pre-created tools, preferably bundled as one with various amount of features that easiens the task of creating demos. Some features include timelines, object rendering, 3d engines, scene-exchanging effects and more. More or less they are build over time as the demo programmer get experienced and builds up a library of methods and algorithms and similar, and wants to spend little time on repetitive tasks and all the «in-between» the unique code and effects for each demo.

Some groups have been known to use demo tools and other pre-made material in an extensive way so that their demos appear almost identical with little new material from one production to another. Such a behaviour is frowned upon in the demoscene and has led to disqualification at demo competitions. Earlier, using such tools for aiding the creation of demos was seen as amateurish, but then mostly if the demo tools were public and little effort was required to make a production, they were so-called «demo makers». In-house, tailored demo tools that are developed over time are totally acceptable, specially if they only aid in getting the «routine» work done.

7.1.2. Analyzing selected releases

To get a better picture of what kind software were actually used and how it was utilised in open source demos I did some research on the UnixScene website. UnixScene is an effort to gather demos released for Unix and to have a striving demoscene based on the Unix platform. It lists 234 productions from 1990 to 2006. Included here are intros and closed source software. By downloading their listing, concatenating the results and running something like this on the result:

```
$ cat unixdemos.html | perl -le 'while(<>) { $a .= $_; } $a =~ s/\n//; while($a =~
/left">\s+<a href=".*?">(.*?)<\/a> by
(.*?)(\(.*?\)).*?<strong>source:<\/strong>(.*?)<tr>.*?<strong>platform:<\/strong>(.*?)<t
d/msgi) { $n = $1; $d = $3; $s = $4; $p = $5; $g = $2; $g =~ s/<a.*>(.*?)<\/a>/$1/; $o =
"$p, $g, $n, $d, $s"; $o =~ s/ +/ /g; $o =~ s/&nbsp;//; print $o; } ' | grep demo | grep open |
sort -f
```

We get a nice result sorted on the platform and open source libraries used, like listed in Appendix 10.7.2.
As we see only 70 of the productions are open source and in the demo category, but still we get some interesting numbers. A majority of the demos are meant to run on Linux, 54 of them. 11 of them are labelled as multi-platform, and of those 8 of them are using SDL. Only 18 of the Linux ones are using SDL. In total 27 of the demos are using SDL, and 22 OpenGL, 29 are meant to run on X11, or utilizing special APIs from it not using higher level abstraction libraries, 4 draw directly to screen using SVGAlib. There are other frameworks and APIs used like GGI, Aalib and PTC, but this is only one or two tops.

One conclusion to draw is that for multi platform open source demos, SDL is the choice. Also we see that most of the UnixScene demos are meant for Linux, including the multi-platform ones, leaves only 6 dedicated to other Unixes than Linux. OpenGL is a natural choice for many demos, and this is no surprise as 3D demos that are to run on Unix will get the easiest 3D acceleration by using that language. Wether the user running the demos is using a software implementation like Mesa or have proper drivers to his graphic hardware doesn't really matter, except speedwise.

There is a total of 47 different groups that together have made the 70 productions. This shows us that are some variety out there in regards of who is making open source demos.

7.2. Interviews

The interviews provided some useful date, although not agreggatable. The subjects questionned provide some insight in a demoscene members relation to open source. The group of people asked have different position in the demoscene; three coders, two are graphic artists, two are musicians, and incidently the two latter are also organizers in their respective groups.

One of them is no longer active in the demoscene but was member in the past. Two of them have current jobs that include work grounded on their demoscene experience. One working making Amanith and AmanithVG, a hardware accelerated vector graphics library for desktop and for handsets. The other works for a graphics hardware manufacturer as a software developer making drivers for low-end devices.

Two of the subjects are part of a group started out as a open source only demo group (TUFS – TUFS Uses Free Software (following the tradition of recursive acronyms).

When asked [Q2] most of them list a set of closed-source products, 2 of them however use mostly open source software, like C++, Lame, OpenGL, Audacity, SDL and Blender. OpenGL and C++ are not strictly pieces of software. Both are open standards though. And the one listing C++ and OpenGL uses the GNU Compiler Collection as a tool. OpenGL is implemented as open source on the platform he uses, except for the low-level drivers that may be closed-source. Audacity and Lame are programs for manipulating music and both are open source. Blender is an open source 3D modeler and renderer.

In [Q3] I ask to what degree they use open source software, and some additional examples are given. Still, though, the answers are very different. One argues that there is little use of using open source software for assembly development, while another says that he uses open sourcesoftware when developing for thr GBA, a low-end portable console which relies on assembly and heavy optimization to make great effects possible. It's notable that when asked specificaly they admit using more open source software than in the previous question. Many of them seem to have favourite applications that they list also, and another mentions various libraries that he can recommend for open source development.

[Q4] shows us that there is a big difference in the amount of open source software used, when asking about regular usage, as end-user application, compared to when asking for usage for demoscene and development related work. Web browser, content-manipulating programs and even operating systems are listed.

The ones answering [Q5] because they actually do use open source software, say that OSS is critical for small platforms to survive, that they provide everything needed, that the philosophy bhind it is part of the reason and that the platform of choice provides mostly open source software. The ones answering negatively claim that they need niche applications that need to be closed to survive, or argue that the closed-source tools they use are more integrated and provide streamlined solutions and are of better quality.

When askes about challenges faced in [Q6] the majority lists lack of quality as the main challenge. In additionn some say that on MS Windows, the integrated graphics and 3D solutions are better than its counter-part OpenGL. Some argue that demoscene members are used to the tools they use at work and that makes it easy for them to continue using those for development of demos, implying that the software used at work are closed-source. The last one starts imaging how a shift to open source could actually happen, and thinks that demo parties might lose out due to the sharing of code. Another says there are no challenges, but only on non-Windows platforms given the reasonstated above. Two argue that if one were to use open source to a larger extent, there would be a change in the focus from programming fast routines to concentrating more on design and looks.

[Q7] Seemed to be a bit badly explained. Half of the group said they didn't use any other aspects, but the rest admitted they personally used a versioning control system or knew people who did. Such systems are not strictly open source methods, although the increased focus on them and having them publicly available are. Another mentions the use of mailing lists and wikis to exchange information with group members.

The penultimate question [Q8] identifies a challenge for open source becoming a wide spread method in the open source, all of the subjects agree that there is a conflict between the secrecy and importance of hiding «trade secrets» and OSS's open nature. However some of them turn the tables and suggest that revealing tricks and code might increase the status of the authors. Another one rules any advantages of using open source for educational purposes due to its «hacky» nature. (badly written code with little or no documentation.) This question seemed to overlap slightly with [Q6] something one of them point out.

The common answer to [Q9] seems to be cost related. They all argue that there are advantages of having gratis software for making art and similar works. Some argue that it lets the user alter the code to its needs, but another one finishes by saying that this is also possible in closed-source software that have script or plugin functionality.

7.3. Involvement in open source projects

I have been involved in several open source projects in different levels and aspects. However the most common way I have participated is either to port some open source software to another platform where the software wasn't available, or by reporting bugs and suggesting fixes. I have also been part of projects in an idle state observing the course and changes over the years, something that has been very educational.

Relevant projects for this research has been the porting and participation of SDLBasic, a basic interpreter with cross-platformness and simplicity as a goal using the SDL framework as a base. I ported the interpreter to Amiga OS 4 and submitted patches and recompiled and fixed bugs when new releases came. In addition I have worked on the Amanith project to get it working on the same platform [Amanith]. Amanith is a 2D graphical vector library similar to Gnome's Cairo. It is made in C++ and relies on OpenGL to achieve hardware acceleration. Something that has normally been reserved for 3D applications. The code was very clean and platform independent. The build system uses qmake and I had to work around that and do some cross-compiling to get it work on the target platform, as it doesn't have TrollTechs QT nor the Qmake system available.

For this project I had planned to implement a simple demo using strictly open source software, but unfortunately I didn't prioritize well enough, leading to lack of time to achieve that task. However, the conclusions of the project might reveal an interest project to set up and work hands-on to get additional relevant coding experience.

8. Discussion

Both the demoscene area and open source has challenges and interesting aspects. I have been able to verify some suspiscions I had before starting the project and got someinteresting input from people participating in the demoscene with different perspectives.

If we go back to the statements I made initially, it's evident that the use of open source is very modest in the demoscene. It wouldn't be wrong to state that it is marginal, both based on the actual numbers, but also judging by the answers from the people I interviewed. It's also interesting to see what kind of problems or challenges the subject see when mixing the two together.

The second statement might have been a bit bolder, and based on the material I have gathered it's difficult to draw any conclusions. The investigation of open source demos did show that multi-platform demos did indeed use SDL as a framework. However there were many other open source demos that were designed for use for Linux. If they happened to use things like OpenGL and SDL, the ability to be cross-platform increases, but there is nothing showing that this has been in mind when choosing to go the open-source route.

A conclusion one might drawhowever is that open source is used extensively for making content, and as regular end-user software. But when it comes to integrating it into one's own production or releaseing softwareas open source things look worse. An important observation though is that some of the reasoning behind not using open-source from the demoscene members are similar to the ones used by conventional closed-source companies. We've seen that area change the last few years, maybe there is a hope for the demoscene too from an open source advocate's point of view?

Although it probably isn't wise to suggest solutions to a problem or invent a problem where there might not be one. I propose that for my spring thesis I work on setting up a project, gather a team, and design and implement a plug-in based and scriptable demotool. This would hopefully provide a powerful tool to demosceners that arecurious about open source, but are afraid of giving away their own things. Such an open source tool that would let people contribute to it, but would let them still be able to utilize it to create content that isn't open source might help the adaption and possibly show that open source demos might be a feasible alternative.

8.1. Evaluation of process

All in all the project prosess didn't turn out that bad, but there are things I will do differently next time. One of the major problems in my opinion is my own lack of communication with my supervisor. In addition to be able to show pro[1]gress, I also didn't utilize the aid a supervisor may give, and worked on my own too much time. This lead sometimes to questions and issues I spent lot of time thinking of and trying to figure out rather than get help from experienced people.

I would try to get writing of the actual project started earlier, even if just a crude preposition and structure. Trying to find a proper research question and define it clearly from the start would help in getting focus during the course of the project. I realize I also fell in the trap of getting lost in the data gathering phase starting to read and investigate subject and matters that in the end didn't necessarily fit the project.

In addition I would seek out more literature in the art of writing a project, how to structure it properly, and the learn the formal processes required from the start. Other things I would do differently is to make a plan and stick to it, although make small changes as the project evolves, write down ideas, thoughts and places to find info immediately to avoid it be forgotten during the process and be more structured about time usage, for instance spending a certain amount of hours at a certain time of the day to the project and stay focused then rather than do bits and pieces here and there.

Getting in touch with possible subjects at an early stage, and maybe doing a pre-study wouldn't hurt either. Seeking help in analysing of data and input in a larger extent would've probably made some clearer conclusions too. However, the end result seems valuable and well-structured, and hopefully the project led to findings interesting to others too.

[1]

9. References

- [Silverman1997] Qualitative research – Theory, Method and Practice, edited by David Silverman, SAGE Publications Ltd., 1997. ISBN: 0-8039-7666-6.
- [Holme&Solvang1986] Metodevalg og metodebruk, Idar Magne Holme, Bernt Krohn Solvang, TANO Achehoug, 1986, ISBN: 82-518-3427-9.
- [Presser&al2004] Methods for testing and evaluating survey questionnaires, Stanley Presser, Jennifer M. Rothger, Mick P. Couper, Judith T. Lessler, Elizabeth Martin, Jean Martin, Eleanor Singer, John Wiley & Sons, Inc., 2004, ISBN: 0-471-45841-4.
- [Yin1994] Case study research - design and methods, second edition, Robert K. Yin, SAGE Publications, 1994, ISBN: 0-8039-5663-0
- [Bell1999] Doing your research project – a guide for first-time researchers in education and social science, Judith Bell, Open University Press; 3 edition (August 1, 1999), ISBN: 0-335-20388-4
- [Fishwick2006] Aestetic Computing, edited by PaulFishwick , Leonardo Books, 2006. ISBN: 0-262-06250-X. Chapter 1, An Introduction to Aesthetic Computing, Paul Fishwick, 25 pages. This chapter contains an introduction to the field of aestetic computing which combines art and computing. The references are important.
- [Craig1999] Craig Harris. Art and Innovation: The Xerox PARC Artist-in-Residence Program , Leonardo Books, 1999, ISBN: 0-262-08275-6. An Archeology of Sound: An Anthropology od Communication, Paul De Marinis, pages 164 - 184
- [Greene2004] Rachel Greene. Internet Art. Thames and Hudson, World of Art, June 2004. ISBN: 0500203768. Introduction pages 8 - 19, Chapter 2, Isolating the Elements, pages 73 - 118.
- [Sedelow1970] Sally Yeates Sedelow. The Computer in the Humanities and Fine Arts. ACM Comput. Surv., 2(2): 89 - 110, 1970.
- [Software-art-wikipedia2006] From Wikipedia, the free encyclopedia. Software art . January 2006, 1 page
- [UnixSceneKameli] http://unixscene.kameli.net/?choice=demos
- [Kameli] http://www.kameli.net/fit/
- [OSIDefinition] OSI: http://www.opensource.org/docs/definition.php
- [FSFFSOSS] Free software != Open Source: http://www.fsf.org/licensing/essays/free-software-for-freedom.html/view?searchterm=open%20source
- [SlengPung] http://kg.slengpung.com/

- [Ogre3D] http://ogre3d.org
- [Mesa3D] http://mesa3d.org
- [LibSDL] http://libsdl.org
- [Amanith] http://www.amanith.org
- [Ogles2] http://ogles2.sourceforge.net/
- [OpenGL] http://www.opengl.org/
- [ThePartyDemo] http://209.85.129.104/search?q=cache:dti8-T0skN8J:theparty.dk/pages/scene/what_is_a_demo.html+what+is+a+demo&hl=no&ct=clnk&cd=1&client=opera
- [32xDemoFAQ] http://tomaes.32x.de/text/faq.php
- [SceneOrg1] http://www.scene.org/showforum.php?forum=11&topic=61743
- [ScheibDemos] http://www.scheib.net/play/demos/what/borzyskowski/
- [SceneOrg] http://scene.org/
- [32xDemoScene] http://tomaes.32x.de/text/pcdemoscene_faq.txt
- [EvenLakeBooks] http://www.evenlakestudios.com/books/
- [Freax] http://www.freax.hu/
- [AssemblyPrizes2006] http://www.assembly.org/2006/compos/prizes
- [MathGames] http://www.maa.org/editorial/mathgames/mathgames_08_16_04.html
- [MindCandy] http://www.mindcandydvd.com/
- [TDT10] http://www.idi.ntnu.no/emner/tdt10/
- [TDT69] http://www.idi.ntnu.no/~letizia/tdt69/
- [TEKS] http://teks.no
- [MatchMaking] http://matchmaking.no
- [TeknaOpenSourceReport] http://polarboing.com/search.php?type=stories&query=tekna Nicolas Mendoza, 2006
- [Shaders] http://www.3ddrome.com/articles/cgshaders.php
- [Evans2005] http://people.debian.org/~evan/ccsummary debian-legal Summary of Creative Commons 2.0 Licenses, Evan Prodromou, April 2005
- [CreativeCommonsLicenses] http://creativecommons.org/licenses/
- [FmodLicense] http://www.fmod.org/ifmodlicense.html
- [SDL] http://sdl.org
- [Cal3D] http://download.gna.org/cal3d/documentation/guide/
- [Perens1999] http://perens.com/Articles/OSD.html The Open Source Definition, Voices from the Open Source Revolution by Bruce Perens, ISBN: 1-56592-582-3

10. Appendix

10.1. Interview with Johan Samuelsson

Name: Johan Samuelsson

Age: 28

Handle: Spot

Group: Up Rough (Amiga)and Triad (C64)

Position: Leader, Graphics, Music

1.What is your role/position in the demoscene?

I am the leader of the music oriented democrew Up Rough Soundsystem. My main function in the group is to get our members motivated, and keep them informed on what's going on. But also to scout for new talents, in and outside the scene. I am also often the one that comes up with ideas for demos/intros, and then we develop the ideas furthertogether. I guess you could call me an Art Director in Up Rough as well, as I often design our products and ask other members to provide fitting music and code. Occasionally I also make music in various oldschool formats such as Protracker and the likes. But I see my music making more like an hobby, the graphics is my main interest. Lately people have started to appreciate my music a lot, which is encouraging. I ended up at 2:nd place in the music competition and 4:th place in the graphics competition on a demo party earlier this year.

2.What kind of software are you utilizing to produce or author demoscene material?

For my graphics work I usually use the DPaint successor PPaint. That's pretty much it. I am a scener from the old generation and a hardcore pixel fetischist, pusing pixels is what I love, and no tool beats PPaint when it comes to that. I also use my Amiga and PPaint when I do C64 graphics, as I can't stand using a joystick for pixeling. I guess I've gotten a bit lazy lately. I have also used the Amiga when doing commercial work (Games, Icons etc.).

3.In what degree are you utilizing Open Source software for demo scene related work? (creative tools, engines, link libraries, etc.)

Well, PPaint ain't exactly open source, it's freeware nowadays, but back in the days it was a commercial product. So was DPaint that I used before that. So no open source apps are used when making graphics. When making music on the other hand I've used the open source tracker Protracker almost exclusively. As for link libraries and such, we don't use open source stuff, and it wouldn't make sense as all our demos are written in Assembler and thus are not very suited to be open sourced nor very useful on other platforms as they are hitting the hardware directly. The first thing we do is to shut down the system and talk directly to the hardware to gain speed. That's the way it was done back then, and that's the way it's still being done on the classic range of Amigas today.

4.What Open Source software are you utilizing if any (names)?

Wow. That's many. This last year or so I have ported about 100 or so open source apps and games to my platform (AmigaOS 4.0) just for the fun of it. When I need something done, for example when converting graphics from the Amiga ILBM format into the C64 Koala format i just port some tools that does the job for me. The stuff I use range from converters to cross compilersto arvhivers to music editors to games. I enjoy the challenge of porting games more than playing them though.

5.If you are using or not using OSS, what are the main reasons behind it?

It's free, and it's out there. In this early days of AmigaOS 4.0 (Which isn't released to the general public yet) OSS really helps to move the platform forward and to keep the users from being bored. OSS is crucial for small platforms to survive.

6.What challenges do you think OSS faces to get a wider acceptance in the demo scene?

The driving force in the scene has always been that 'I can do that better than you!' And it has always been about who pushes the hardware the furthest, therefore people like to stick to their magic hw pushing sourcecodes. However this has changed lately, as there is no challenge in pushing the HW anymore, as the HW does everything for you and it's blazing fast compared to the computers of the 80's and 90's. 10 years ago a coder could spend months optimizing a single routine. You don't see that today. Todays demos are more about the content, more like art, and less about the hardware. That has lead to coders being less protective of their sources, and by the use of libraries such as SDL demos are suddenly easy to compile for multiple platform, and then it makes sense to include the source. This doesn't mean that it's common to include the source, but it happens. And I get the feeling it's getting more common. Check out the site called www.unix-scene.org for a lot of open source demos.

7.Are you using other aspects from OSS, rather than the software itself? (collaborative methods,

shared source repository, educational purposes, etc.)

No.

8.Does the competitive nature of the demoscene conflict or help with the use of open source?

No, it doesn't help at all. It surely conflict. Coders like to show off, so they often tell less talented coders how to do certain tricks, but it's not often actual code are shared. But ofcourse there are exceptions. Some diskloaders for the C64 are open source, music replayers are almost always open source, and some low level graphics conversion libs are open source.

9.In what ways can open source software be beneficial for artistic purposes (create digital art content like images, music, demos, interactive presentations etc.)?

I don't know if the fact that they are open source is the key here, it's the fact that they are free that makes them successful i think.

10.2. Interview with Matteo Muratmat

Name: Matteo Muratori

Age: 29

Handle: Turbo

Group: Apocalypse Design

Position: Main coder

1.What is your role/position in the demoscene?

Idler at the current state :)

2.What kind of software are you utilizing to produce or author demoscene material?

When i was scener, my favourite tools were Visual Studio 6.0, and 3DSMax. For the sound we used FMOD library

3.In what degree are you utilizing Open Source software for demo scene related work? (creative

tools, engines, link libraries, etc.)

Very low, some nice opensource libraries, useful for the scene, are GLEW, GLFW, libPNG and libJPEG

4.What Open Source software are you utilizing if any (names)?

For my work i use several opensource softwares. Abakt for the backups, OpenOffice for all office tasks, 7ZIP for packer/archiver, Firefox to browse the web, Inkscape to do some SVG :)

5.If you are using or not using OSS, what are the main reasons behind it?

For my work i use only two main closed source softwares:Windows XP and Visual Studio .NET. I must admit that they are at another (much higher) level, i can't waste time for common tasks at OS level (browse directory, create .ZIP on the fly, and so on), and overall i need a professional development environment with a powerfull debugger.

6.What challenges do you think OSS faces to get a wider acceptance in the demo scene?

The general quality level. It's not good to say, but 95% of opensource softwares are "unusable" or they don't work :)

7.Are you using other aspects from OSS, rather than the software itself? (collaborative methods, shared source repository, educational purposes, etc.)

Code repository, i'm a big fan of SVN.

8.Does the competitive nature of the demoscene conflict or help with the use of open source?

The demoscene has been (and i think it still is) for a long time a "closed source" movement, only recently we see some demogroups that releases the sources oftheir demos. In general i can't see such kind of conflict.

9.In what ways can open source software be beneficial for artistic purposes (create digital art content like images, music, demos, interactive presentations etc.)?

Well, i think opensource software, talking about artistic purposes, could be a good start point. A so called "newbie" can start producing contents with no fee or money! From a professional point of view, i think closed source software, epsecially for gfx contents, are too superior respect to their open source counterparts.

Please, attach some AmanithVG links to your article ;DDDD

10.3. Interview with Erlend Hamberg

Name: Erlend Hamberg

Age: 22

Handle: Morbuz

Group: Tufs

Position: Coder

1.What is your role/position in the demoscene?

Lazy coder in a demo group. Not too active in the scene, but watches from the outside.

2.What kind of software are you utilizing to produce or author demoscene material?

C++, OpenGL, SDL, Blender

3.In what degree are you utilizing Open Source software for demo scene related work? (creative tools, engines, link libraries, etc.)

With the exception of libfmod, all software used is free.

4.What Open Source software are you utilizing if any (names)?

vim, gcc, SDL, Blender, subversion

5.If you are using or not using OSS, what are the main reasons behind it?

Provides the best tools on my preferred platform (Linux).

6.What challenges do you think OSS faces to get a wider acceptance in the demo scene?

None, on other platforms than Windows. On Windows, Direct3D delivers great performance and is tightly integrated with the operating system and Microsoft's development tools.

7.Are you using other aspects from OSS, rather than the software itself? (collaborative methods, shared source repository, educational purposes, etc.)

I often use subversion/cvs to collaborate with others. I also often include source with myproductions.

8.Does the competitive nature of the demoscene conflict or help with the use of open source?

It could be both. People can prefer hiding their 'secrets', but a good production with source included could also be a show-off of coding skills and a 'proof' that no cheap tricks are used.

9.In what ways can open source software be beneficial for artistic purposes (create digital art content like images, music, demos, interactive presentations etc.)?

The advent of freely licensed music andart (perhaps especially CC-licensed) can be of great help for artists and musicians who no longer have to steal samples and textures.

10.4. Interview with Gjøran Sæther

Name: Gjøran Sæther
Age: 23
Handle: Proteque
Group: Scarab (Amiga), Tufs (PC)
Position: Graphician

1.What is your role/position in the demoscene?

Making fullscreen graphics for demos and graphic competitions in Scarab and Tufs.

2.What kind of software are you utilizing to produce or author demoscene material?

Most of the time I use Photoshop and Corel Painter. I also use Amicapaint on Commodore 64

3.In what degree are you utilizing Open Source software for demo scene related work? (creative tools, engines, link libraries, etc.)

I did use The Gimp earlier but have stopped using it in favour of Photoshop. I do use some opensourced emulators like UAE for emulating Amiga.

4.What Open Source software are you utilizing if any (names)?

Ah a lot. Firefox, Linux, Adium, Vim, X11 etc etc

5.If you are using or not using OSS, what are the main reasons behind it?

The main reason is lack of functionality. Lets say the gimp. It lacks proper way of handling brushes (resizing etc).

6.What challenges do you think OSS faces to get a wider acceptance in the demo scene?

To be honest I think they need to work on the qualty of the software. Make proper ports of stuff to lets say MacOS X og the gimp so that people don't need to use X11.

7.Are you using other aspects from OSS, rather than the software itself? (collaborative methods, shared source repository, educational purposes, etc.)

Nope

8.Does the competitive nature of the demoscene conflict or help with the use of open source?

Yes. When using gpl-libs it forces them to open demos. Demos is about secrets and tricks. You don't want to show the world how you managed to do the tricks that looks impressive (but doesn't have to be impressive)

9.In what ways can open source software be beneficial for artistic purposes (create digital art content like images, music, demos, interactive presentations etc.)?

Well if you think of it like the common man sees open source it is free of charge. Artists are broke and cant afford to buy software. However in the real terms of free it helps artists to be able to choose platforms as their favourite programme can easily be ported.

10.5. Interview with Erik Faye-Lund

Name: Erik Faye-Lund
Age: 25
Handle: kusma
Group: shitfaced clowns
Position: coder

1.What is your role/position in the demoscene?

I'm a demo-coder, writing demos for the PC and GBA. I also pretend to be a graphics artists when nobody can see me.

2.What kind of software are you utilizing to produce or author demoscene material?

Mostly commercial grade software. I use Visual Studio (PC) and devkitPro + UltraEdit (GBA) for development, and Photoshop + Illustrator + 3D Studio MAX to create demo content.

3.In what degree are you utilizing Open Source software for demo scene related work? (creative tools, engines, link libraries, etc.)

To a large degree for my GBA projects, almost to no degree in my PC projects.

4.What Open Source software are you utilizing if any (names)?

devkitPRO, Console 2, Pimpmobile, Python, Swig

5.If you are using or not using OSS, what are the main reasons behind it?

The commercial grade tools are in general of higher quality than the open source alternatives, and they have mostly better support, both through having larger user-bases (and hence communities) and direct customer support.

6.What challenges do you think OSS faces to get a wider acceptance in the demo scene?

The commercial development tools are of high quality, and as more and more demo-coders have jobs that gives one access to good tools, it's easy to stick to those you work with every day. And demo sceners are usually focused on productivity rather than idealism.

7.Are you using other aspects from OSS, rather than the software itself? (collaborative methods, shared source repository, educational purposes, etc.)

I'm not sure if I understand the question. Source control is by no means an OSS aspect, but we do have a wiki-page and a mailing-list in Shitfaced Clowns used for exchange of ideas without having to have close contact all the time.

8.Does the competitive nature of the demoscene conflict or help with the use of open source?

Absolutely. But this often limits itself to specific routines, not frameworks or libraries. But the competive nature of the demoscene result in fewer complete source-releases. But then again, a demo is a one-time product, they don't evolve over time. The only use of demo source code would be bug-fixing for external people. (Let's face it, demo-sources are usually hacks -- there's not too easy to learn from source code with finnish curse words for variable names, and random broken #ifdefs around all the place and no comments)

9.In what ways can open source software be beneficial for artistic purposes (create digital art content like images, music, demos, interactive presentations etc.)?

Technical minded artists can modify applications to do exactly what they want. But then again, this feature is also present in commercial applications like 3DS MAX through user-scripts.

10.6. Interview with Ruairi Fullam

Name: Ruairi Fullam
Age: 24
Handle: rc55
Group: UKScene
Position: Organiser / Musician

1.What is your role/position in the demoscene?

I organise the Sundown demoparty, the only multi-platform demoscene party in the UK since 2005. I have recently started releasing music.

2.What kind of software are you utilizing to produce or author demoscene material?

Renoise for production. Audacity for Sample Editing. Lame for Mp3 Encoding. Foobar2000 for listening. FLAC for archiving.

3.In what degree are you utilizing Open Source software for demo scene related work? (creative tools, engines, link libraries, etc.)

Generally within my role I only use FLAC and LAME for archival and transport purposes. I use Audacity for sample editing occasionally.

4.What Open Source software are you utilizing if any (names)?

OpenOffice, Ubuntu Edgy Eft (distribution), Audacity, Lame, Flac, Firefox.

5.If you are using or not using OSS, what are the main reasons behind it?

Renoise is a closed source niche application with a very low chance of being open sourced. The developers revenue stream simply wouldn't exist if it was free due to its nature. I do try to use OSS wherever I can because I agree with the philosophy (to a large degree however I believe there is cases where proprietary software is reasonable).

I am an enthusiast of all technology, and Open Source software has the highest level of intimacy with producer to consumer, so that's why I favour it.

6.What challenges do you think OSS faces to get a wider acceptance in the demo scene?

From what I see the demoscene has never massively been about sharing code, so it would be a shift in ideology. The biggest problem from my point of view is the divide in quality between more experienced groups and newcomers and how their positions are acheived and maintained.

It would be very interesting to see what would happen. I imagine that the demoscene would possibly end up becoming even less concerned with technical challenge and more with visuals, artwork and production. It isn't a bad thing though, but the demoscene audience is a fickle one and any substantial shift in philosophy could potentially alienate a large number of people.

The parties ultimately would be the people to suggest this, but they risk their incomes by suggesting such things.

In regards to Open Source in it's current state, Microsoft's DirectX always seems to be leaps ahead of any OpenGL offering, and it's the same for development tools, support, technology developments and career prospects. A lot of sceners work or want to work in the games industry - and DirectX really seems the only viable option. (Of course, there are exceptions to the rule).

7.Are you using other aspects from OSS, rather than the software itself? (collaborative methods, shared source repository, educational purposes, etc.)

Not intentionally. I know some high end demo groups use versioning systems for project management.

8.Does the competitive nature of the demoscene conflict or help with the use of open source?

See Q6.

9.In what ways can open source software be beneficial for artistic purposes (create digital art content like images, music, demos, interactive presentations etc.)?

The only benefit I can see over existing proprietary solutions is the cost. Despite open source being code editable, most commercial software offerings appear to be extensible to a large degree anyway. It all depends on context.

10.7. List of productions examined

10.7.1 Demos from the two last years at the biggest parties

From: ftp://ftp.scene.org

195_95_by_plastic.zip
1995_by_kewlers_mfx.zip
afy-bloodshot.zip
amigasp.zip
andromeda.software.development-ambiencefor.the.masses.zip
andromeda_software_development_-_animal_attraction.zip
ans-silvae.zip
artcity-party-version.zip
asd-aphorism_for_the_masses.zip
asd-captive.zip

bacardi_cola_by_moonhazard.zip

bauknecht_trocken.zip

bombman_by_matt_current.zip

boozing.kupoz-the.lonely.tower.zip

bulk_darklite.zip

butterflythoughts.zip

bx_elixir.zip

bzh_megahitler.zip

chickenvictory.zip

clrsrc-1sb.zip

club_null_by_wamma.zip

contamination_of_the_present_governmental_system_by_matt_current.zip

contraz_-_i_eleni-the_stars.zip

couldnt_sleep_v100.zip

ctz-trashin.zip

dark.codex.crew-well.it's.crab.01.timemachine05.zip

da_scene_by_deagrace.zip

dd_ican.zip

demoen.zip

digitalangels-digitalnoize.zip

dihalt2006.zip

diskoalifight_by_wamma.zip

division0_aadn2-invit.zip

don_t_stop_by_portal_process.zip

dope-reflections.proper.zip

dutch_posse_deathtripping.zip

faktafan-raanebil.howto.zip

flo_airu.zip

formal_obsessions_by_adapt.zip

fr-045_life_after.zip

guideline_kukulkan.zip

iconoclast_by_andromeda_software_development.zip

ifb-fjellbrod_med_rug_og_hvete.zip

ilmapeppu_-_airass_by_taat_2006.zip

imaginations_by_dope-party_final.zip

imaginations_by_dope.zip
imbecility_-_infected.zip
imbecility-ubern00bs.zip
indremisjonens.frivillige.bakeri-fjellbroed.med.rug.og.hvete.zip
indremisjonens_frivillige_bakeri_-_kneipp.zip
ineffable-tribute.to.c64.zip
into_the_0_0_255_by_unknown_artists.zip
intracks-2.years.late.zip
intracks-final.zip
iterate_by_imbusy_xerxes.zip
iterate_-_imbusy_xerxes_final.zip
jml-scie.zip
kbd_modus_fix.zip
keyboarders-the_usual.zip
key_dark-pigbender.zip
kikiprods!_wrong-heaven-final.zip
kp!2005_electronic-warfare.zip
light_amplification_by_stimulated_emission_of_radiation_by_ananasmurska.zip
light_amplification_by_stimulated_emission_of_radiation_(final)_by_ananasmurska.zip
lightcore_-_sacre_bleu.zip
lilil_-_executable_universe.zip
limp_ninja_-_inductiemachine.zip
lovefearthenumbers_by_kooma.zip
mc-rem.zip
mfx_athr.zip
mfx_deit.zip
miserae_by_timescratchers.zip
monorail_party.zip
moonstone.zip
my_diamond_mind_broken_by_outbreak_1.zip
n_4fxsndtrk.zip
nazareth_-_the_beginning.zip
nce-dac.zip
nce-intl.zip
new_horizons-ac.zip

nocturnal-4.effects.and.a.soundtrack.zip

no_trace_by_rre.zip

outracks_-_genesis.zip

outracks_nokelvin.zip

out_summer.zip

(pc-demo)-45secs_by_activator~ex-yaphan.zip

pgs_glaucoma.zip

playpsyco.and.ifb-d'frm.zip

playpsyco_no_cave_in.zip

playpsyco_-_treble_and_bass.zip

portal.process-meet.the.biots.zip

pp_meet_the_biots_final.zip

primate_theory_by_unknown_artists.zip

pure1d_v100.zip

pure1d_v102.zip

pyromanssi_-_pyromance_by_taat.zip

radiate.zip

rbs_asminv06.zip

[sd]_tnsse_party.zip

sgz_cafre2.zip

silicon_skies_-_die_chefrocker.zip

simuinvi.zip

sklutti_-_paranoia.zip

sklutti-skluta.2.zip

solenoid.zip

sp6-invit.zip

spank_da_bitch_by_diewissenden.zip

spontz_80_demosound.zip

starstruck_by_tbl-placeholder.zip

structured_noise_-_xanubis.zip

sts-02.zip

sts-03.zip

sts-04__instant_zen_by_synesthetics.zip

tannh_user_gate_by_cubicle_1.zip

tbl-starstruck-partyversion.zip

telos_by_portal_process.zip

textmode_v_by_kvasigen.zip

the_apparatus_by_division.zip

the_barythymia_by_division.zip

the_dark_codex_-_barkebilleboogie.zip

the_line_age_by_trauma.zip

thenever-asciigefickteansischnitten.zip

the_winter_party_by_it-forge.zip

tlotb_ifparty05.zip

track_one_by_fairlight.zip

traction_25_final.zip

traction_25_party.zip

traction_explosive.zip

traction_ud_nowayback.zip

trocken_by_bauknecht_final.zip

tsq_rudimentary.zip

tufs-pobelfro.zip

unknown_entity_by_promille_dezign.zip

unleeb_-_color_parade.zip

vvd-rem.zip

xplsv_tube.zip

yaphan.vs.excess-dihydrogenmonoxide.zip

yourroom.zip

zokum_and_timo_-_tryggvification_3.zip

139 in total

10.7.2 UnixScene open source demos

From: http://unixscene.kameli.net/archive/

Linux/GGI , ECFh , Bizarre'99 invitro, (demo), open

Linux/GGI/PTC , Thaumaturge , NYCD, (demo), open

Linux/GLUT/GL , Radiant , Lost in the underflow, (demo), open

Linux/GL , xAngle , xa-005: Spike, (demo), open

Linux , Knights , Mindlink 1.1, (demo), open

Linux , Knights , Styler 1.1, (demo), open

Linux , Knights , Styler 2.1, (demo), open

Linux/PTC , Aspirine , Life is a bizz, (demo), open

Linux/SDL , cosmo , My First Demo, (demo), open

Linux/SDL , CSR , Loop, (demo), open

Linux/SDL , Glass , GNUdemo, (demo), open

Linux/SDL/GL , eL , Past mosfncitioo, (demo), open

Linux/SDL/GL , The Lab , Summer Hack, (demo), open

Linux/SDL/GL , xAngle , Beatbox 2005 invitation, (demo), open

Linux/SDL , Hydri&Spruce , \n, (demo), open

Linux/SDL , Introx , Oj!, (demo), open

Linux/SDL , mfx , Dose 2, (demo), open

Linux/SDL , mfx , Surge, (demo), open

Linux/SDL , Particle system , Celtic particle, (demo), open

Linux/SDL , Perplexity , Teatro, (demo), open

Linux/SDL , Rastasoft , THK intro, (demo), open

Linux/SDL , s3k7lunch , Gløgg, (demo), open

Linux/SDL , Topcrew , Stallinator, (demo), open

Linux/SDL , Tran , Timeless, (demo), open

Linux/SDL , tAAt , tAAt 2001, (demo), open

Linux/SDL , xAngle , xa-001, (demo), open

Linux/SVGAlib , Perplexity , Mega-o-matic, (demo), open

Linux , tAAt , Laatukauraa, (demo), open

Linux/X11 , Alge & Irvin , Ich bin ein demoscener, (demo), open

Linux/X , != , +1-1, (demo), open

Linux/X11/GL , Posercummers , Sykt mange jur, (demo), open

Linux/X11/GL , The Lab , Eternal, (demo), open

Linux/X11 , xAngle , xa-002: Scratch, (demo), open

Linux/X11 , xAngle , xa-003: Midsummer, (demo), open

Linux/X , Aspirine , Magnus effect, (demo), open

Linux/X , BlockoS , CrackTros, (demo), open

Linux/X , co-operation , Meeting, (demo), open

Linux/X/GL , Excess , Amoeba, (demo), open

Linux/X/GL , Fadeout , Triple five tsunami, (demo), open

Linux/X/GL , Flimmer , Refreshment, (demo), open

Linux/X/GL , Michael and Dirk , NiX, (demo), open

Linux/X/GL , Poor groupless sceners , Unsigned, (demo), open

Linux/X/GL , Popsy team , VIP2, (demo), open

Linux/X/GL , TUFS , Claustrophobic Sting, (demo), open

Linux/X , Lame over , Xdemo 2, (demo), open

Linux/X , Lame over , Xdemo 3, (demo), open

Linux/X , Lame over , Xdemo 7, (demo), open

Linux/X , Lame over , Xdemo, (demo), open

Linux/X , No! , LTP4 invitro, (demo), open

Linux/X , Posercummers , Superscroller 5, (demo), open

Linux/X/SVGAlib , SED , Serious?, (demo), open

Linux/X/SVGAlib , SED , Trankil, (demo), open

Linux/X/SVGAlib , Skal , Hard Rox, (demo), open

Linux/X , Syn[rj] , Space delirium, (demo), open

Multiplaform/SDL , Fit , Stercus Accidit, (demo), open

multiplatform , Bomb , State of mind, (demo), open

Multiplatform/SDL , Da Jormas , Hata, (demo), open

multiplatform/SDL , Fit , Chrysler, (demo), open

multiplatform/SDL , Fit , Dr.Fungi, (demo), open

Multiplatform/SDL , Fit , Hex Pistols, (demo), open

multiplatform/SDL , Fit , Jenny thinks, (demo), open

multiplatform/SDL/GL , Fit , Dr.Fungi goes Africa, (demo), open

multiplatform/SDL/GL , Fit , Micery, (demo), open

multiplatform , Sleber eid , The Director, (demo), open

Multiplatform/X/? , Artwork , Megademo IV 2, (demo), open

Unix/Allegro , Lou , Shmuptro, (demo), open

Unix/SDL , QMG , Qrash, (demo), open

Unix/X/GL , NAN5 , Yuri nation, (demo), open

Unix/X , QMG , HElliZER, (demo), open

Unix/AA-lib , AA , BB, (demo), open

70 in total