Sandbox Culture

Aymeric Mansoux

June 11, 2013

Introduction

Is there an app for that? There is an app for that! These two popular expressions sums it all: the writing and execution of software is reduced today to the most banal consumerist process that we can possibly imagine. Whatever your needs are, simply rub a finger on the mirror lamp of your mobile device and the app store genie will grant any of your wishes: no time to take care of your children, here are a few educational apps; your cheap $0.99 plastic cooking timer is broken, plethora of cheap $0.99 timing apps await you; no idea what to eat, drink, watch or listen to, surely an app will tell you better than your friends and family, who might have actually convinced you to purchase such advisory apps via a social network app. Truth is, apps have found their way in every aspect of most people life, thanks to a software industry that is leading us to believe that everything can be mediated and

facilitated by apps. The question we should ask ourselves is in which way this marketed and simplified view on software, that is the app, is transforming our understanding of the production, distribution and the use of software. Indeed, lost in the fog of a metaphorical cloud, we are left to create new and wander in all sort of allegories, waiting for a posthumanity to clean up the mess we are making of all the links between living things, technology, ideas and concepts. Meanwhile, and before this day to come, these deceptive rhetorical figures of speech are expanding their territory. Existing as a result of a dialogue between the analogue and the digital, they open doors to new playgrounds to inhabit and dream of, yet for which the metaphorical coating is preventing us from grasping their underlying code, literally. Software as app is one of these problematic playgrounds. In this essay I am arguing that in order to deconstruct these technological allegories and shortcuts we need to turn them against themselves in order to highlight the way they change our thinking about cooperation, collaboration, trust and communities. To do so I am using the analogy of the sandbox that is deeply anchored in modern computer history, more precisely in this text with the Unix chroot program. Here, I am exploring the idea of a sandbox culture as a filter to look at how apps are encoding social and commercial transformations in popular mobile operating systems such as Apple's iOS and Google's Android. Then, taking the example of the iPhone RjDj app, I am exploring how similar processes of *sandboxing* also exist at the level of free and open source software communities, ultimately suggesting the existence of a sandbox

culture emerging from the multiple properties and nested production processes happening within the growing network of humans, software and the contracts that bind them all.

Geek speak metaphors

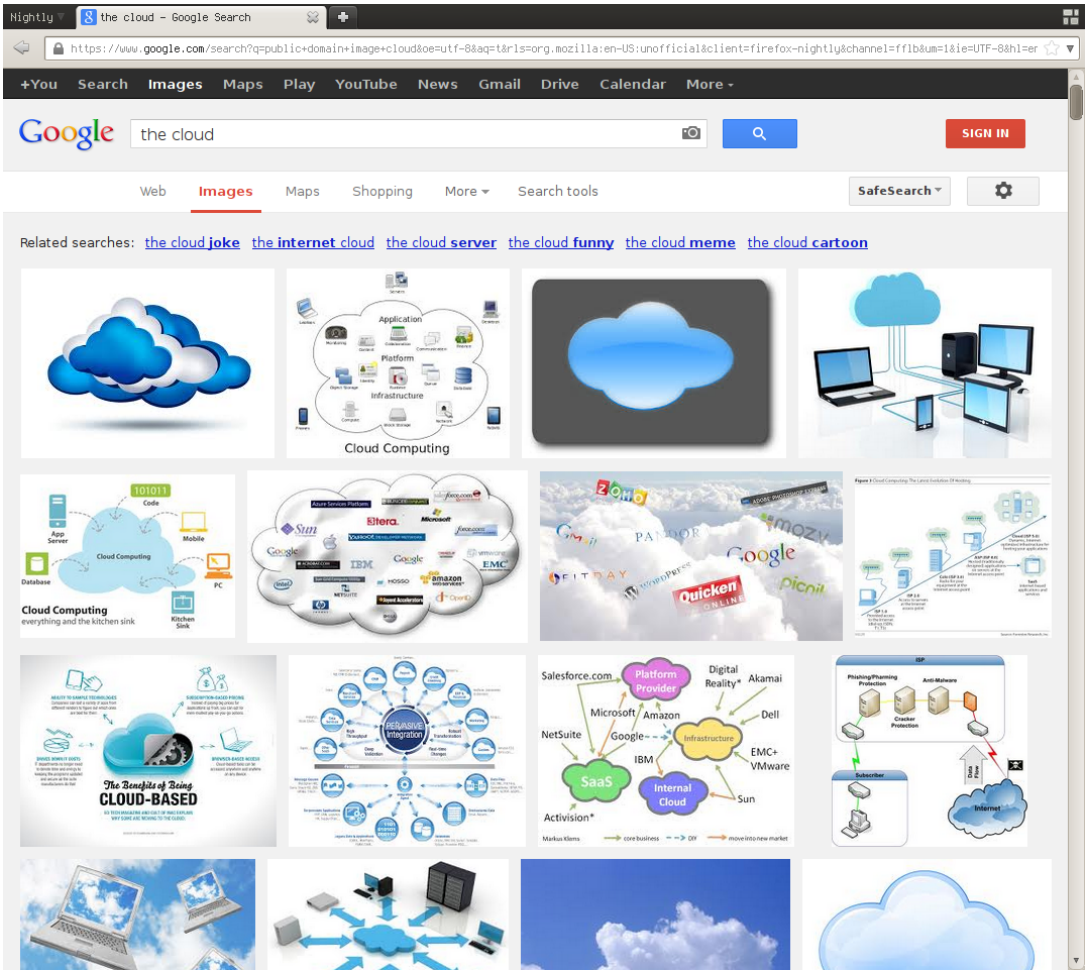

Figure 1: Clouds from the cloud.

Figures of speech, and most notably analogies and simple metaphoric mechanisms have been used in the context of op-

erating systems for a long time.[1] It can be seen in the way the Graphical User Interface (GUI) of computer operating systems can refer to models of windows and desktops and how digital photo manipulation software are emulating the tools found in the darkroom to name a very few examples. Some of these shortcuts can also be used as a general umbrella to hide and help swallow whole infrastructures and the businesses that rely on them, as best exemplified with the web 2.0, that enabled the transformation of websites into software as services, and more recently the cloud, that allows a simplified discourse around ways to interface with large centralised or decentralised network storage and processing systems. What is less known though, is the fact that such terms are powerful catalysts. They are not just instruments of a global corporate conspiracy, they are also the very fertile ground for new technological developments and also operate an important role in the process of programming itself.[2] Indeed, regardless how the whole idea of the "cloud" might be at the same time flawed and manipulative it is nonetheless feeding an imaginary that will influence all its actors and as a consequence shape tomorrow's technology whether we like it or not.

These technology related allegories, metaphors and figurative shortcuts are not unidirectional. They can also be used in turned as a conceptual tool to look at cultural phenomenons to

1. Neal Stephenson, *In the Beginning...Was the Command Line* (New York City: HarperCollins, 2009), 18.

2. Christopher Alex McLean, "Artist-Programmers and Programming Languages for the Arts" (PhD diss., Goldsmiths, University of London, 2011), 32-35.

highlight some properties that would not be easily visible otherwise. For instance Lawrence Lessig uses "geek-speak metaphors" of file system permissions, read-only (RO) and read/write (RW), to illustrates the mechanisms of remix culture.[3] The RO versus RW narrative works nicely to make a good case of remix culture because it relies on technological elements that are familiar to anyone who used a computer. Growing from the *versioning* of Jamaican reggae songs into dub music, to its systematisation in the nineties music industry, the remix has been increasingly used to demonstrate the boundless power of cultural expressions beyond the realm of music. Building a critique of remix culture is far from trivial as the later carries positive and popular, maybe populist, messages about the needs to access and generate information for the benefit of humanity: RW. The message is strong as it relies on the obvious cultural mechanisms in which any objects is a cultural product, specifically an object deriving from existing ideas and technologies and therefore, yes indeed, of course, everything is remix. With this idea, those who are preventing or limiting such a flow can only be against culture itself and are often impersonated by evil inhuman corporations.[4] However at no point in these David vs Goliath tales it is questioned how such remixes come into existence, what kind of technological, social, political frameworks permit their production and what impact they have both on their aesthetics and more importantly how

3. Lawrence Lessig, *Remix: Making Art and Commerce Thrive in the Hybrid Economy* (London: A&C Black, 2009), 28.

4. Brett Gaylor, dir., *RiP!: A Remix Manifesto* (Montreal, CA: EyeSteelFilm, 2008).

such frameworks influence the groups that inhabit them. While remix culture helps us to flatten down culture into one happy-go-lucky single linear stream of exhilarating tribute to the genius of mankind, we need to find another analogy to understand more precisely what might be going on at another level, that is beyond the usual stereotypes of good communities vs evil media, gift economy vs exploitation of free labour, etc. So even if the popularity of remix culture has greatly contributed to demystify the role of the author as an isolated genius by making visible how originality is relative to both the different inspirational sources and building blocks it relies upon to exist, one aspect still remains to be debunked: the context in which the act of creation takes place.

For instance remix on its own is not enough to understand the network and the links between the different objects that constitute it, thus missing an opportunity to understand how control and rules affect the dynamism of creation. Back to the RO and RW file permission approach, other questions arise, who owns the file, where is it located, how can it be accessed, who benefits from reading from or writing to it, etc. How does such question can be mapped back to the remix? Coming back to the birth of dub music, looking at it from the perspective of the remix will allow us to understand its formal aesthetic and its sonic qualities, but it will at the same time overshadow other properties. Namely, how originally the dub plate functioned as an addictive product sold to competing Jamaican sound systems who constantly needed new stocks of these self deteriorating wax plates in order

to attract an audience and sell them alcohol. The same sound systems were ultimately advertisement platforms for the tracks used as source material for the dub versions. Such original tracks were never played during these gigs and were only made available to purchase in music stores owned by the same studios that were producing the dub versions.[5] In this case remixes functions as a well defined branch of a tree and not so much as a node in a vague metaphysical network without boundaries as it is too often presented and understood in the remix culture narrative. So is there a better way to look at the act of creation so that some context can also be made visible? Let's see. Instead of RO and RW, the file system inspired "geek speak metaphor" that I suggest to use is the chroot, a simple Unix command that I will explain in details in the next section. Using the chroot metaphor assumes however some historical and technical understanding as we need to zoom out from the simple file permission perspective to look instead at the whole operating system. In order to do that, I first want to provide a quick summary of the history of Unix and time-sharing so as to understand the evolution of permissions and user directories in today's operating systems.

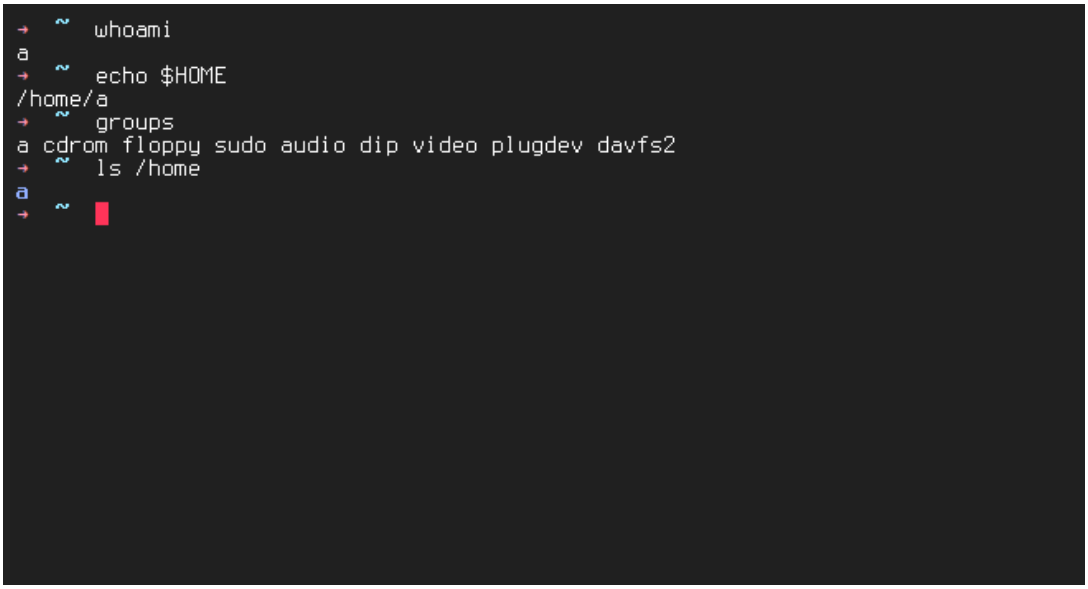

Figure 2: A terminal on my laptop

Chroot jails

Time-sharing, introduced in the late nineteen fifties, enables the sharing of computation time amongst several users, so that one person does not have to wait for someone else's calculation to complete so they can start theirs..[6] This, nowadays obvious feature found in multi-tasking and multi-users environments, was popularised in the 1964 Multiplexed Information and Computing Service (MULTICS) operating system, that eventually led in 1969 to the birth of the simplified Uniplexed Operating and Comput-

5. Michael Veal, *Dub: Soundscapes and Shattered Songs in Jamaican Reggae* (Middletown, Connecticut: Wesleyan University Press, 2007).

6. Katie Hafner and Matthew Lyon, *Where Wizards Stay Up Late: The Origins Of The Internet* (New York City: Simon & Schuster, 1999), 25.

ing System (UNICS) eventually renamed Unix.[7] The latter paved the way for a large family of operating systems found today in servers, network devices, video game consoles, desktop and laptop computers, and of course smart phones, tablets and all sorts of fondleslab. One important aspect of a Unix-like environment is its organisation as a hierarchical model in which everything is represented by files and arranged in a tree of nested directories. In such a system every files and processes have a single owner. Users belongs to different groups which give them different permissions to navigate in some parts of the tree structure as well as read, write and execute files in the system. They are also given a home directory in which they can manage their own files. Sitting on top of the mountain, a superuser, called the *root* user, possesses all the permissions in the machine. As we see here, this quick overview is already projecting us in a richer imaginary than the one found in the RO versus RW comparison. This description is however problematic as this top-down authoritative hierarchical organisation is not without issues. However, and while indeed this description "is deeply indebted to culturally determined notions such as private property, class membership, and hierarchies of power and effectively,"[8] it has nevertheless spawned a very rich network of machines built and inhabited around principals of mutual help, sharing, decentralisation and

7. Eric Steven Raymond, *The Art of UNIX Programming* (Boston: Pearson Education, 2003), 31.

8. John Unsworth, "Living Inside the (Operating) System: Community in Virtual Reality," in *Computer Networking and Scholarly Communication in the Twenty-first-century University*, ed. Teresa M. Harrison and Timothy Stephen (Albany, New York: SUNY Press, 1996), 142-142.

cooperation, literally turning this capitalist-like file system organisation into a constellation of networked communities, from hackerspaces to artist-run servers and activist infrastructures.

Figure 3: Connecting to a couple of Unix-like machines, hostname obfuscated.

Now, represented with Figure 2, if I open a terminal on my laptop, running the free Unix-like operating system Debian, I can see a textual representation of the file system and pass some commands in order to know: what is my username (a); where is my home directory (/home/a); which groups I belong to (too many);

and if there are other people having account on this machine by simply looking what other files are in the *home/* directory (no, I am the only one). Using a Secure Shell (SSH) communication, it is then possible for me to connect to other Unix-like machines on which I already have an account. Such machines are currently connected to the Internet and have different purposes, some are storage servers, some are website hosts, others are streaming platforms, it does not matter because they are all computers running a Unix-like operating system. What is interesting for the demonstration is that, thanks to SSH, I can type the same commands on these distant machines, as the ones I just executed on my laptop. Some results of this can be seen in Figure 3. By doing this I am aiming at showing you that: I have different identities on different machines; each of these machines provides homes for other users, friends, peers, students, etc; on some machines I have superuser access, on some others I am regular user. This social network as file system implies that in these micro communities I am known under different contexts, and that I am either trusted to take good care of the homes that I am maintaining for others or, alternatively, that I trust others to take care of mine. This mechanism of trust is also built in this system to avoid leaving the door open to any malicious software or user. It is in this particular situation that the command chroot, that I referred to earlier, becomes useful. Added in 1979 to the seventh edition of Bell Labs' Unix,[9] the chroot program manipulates the way the

9. Ted Timar, "Unix - Frequently Asked Questions (6/7) [Frequent posting]," comp.unix.questions, unix-faq/faq/part6_1084272547@rtfm.mit.edu, May 11, 2004.

file system is "seen" from a user or process perspective. It does so by moving the apparent uppermost directory of the file system to another location, thus preventing the *chrooted* users and processes to access anything outside of this metaphorical jail. Said differently, the sub-folder you are stuck inside appears to you as being the base and starting point of all the other folders in the system, while other users and processes can see you constrained in the space and resources you are being allocated: you really are in jail. More specifically when applied in the context of security, the technique is literally called a chroot jail.[10] It's not by accident that the act of getting administrator rights on an iPhone, which operates a Unix based system, is called jail breaking as this hack is a form of privilege escalation that aims at liberating phone users from their jail, and subsequently give them access to the full Unix machine hidden behind the touchy-feely-shiny-sticky golden cage GUI. Similarly, on Android phones the term "rooting" refers to the process in which the phone user can modify the operating system so as to gain superuser permissions: that is becoming *root*.

If RO and RW are used to support a remix culture, I am arguing that using the chroot to talk about cultural processes leads us instead to acknowledge the existence of a sandbox culture. Indeed, sandboxes are deeply rooted, no pun intended, in software and hardware history as we have seen with the chroot program. At the same time the term is generally used to describe all

10. David A. Wheeler, "Secure Programming for Linux and Unix HOWTO" (1999).

sorts of testing and prototyping practices that requires the creative potential offered by such an isolated and malleable place. This aspect makes our digital sandbox similar to the children playground's physical box full of sand, where things can be invited, bounced, created, abandoned, contained, constrained, interpreted, experimented, censored, populated and grown, said differently it's a world on its own. Most importantly it implies the existence of a higher level structure, and therefore context, in which all these actions are ultimately nested within. The sandbox is within the playground, that is within the park, that is within the city, that is within the state, etc.

With sandbox culture in mind, let's now try another experiment. This time, I am opening a terminal on an Android phone and type the same commands as with the other Unix-like systems shown above. Looking at Figure 4, the results are quite surprising. When asking the system to display my username, I am given as response something quite cryptic, *u0_a94*, which is certainly not the username that I gave when I configured the phone. Then, when requesting the system to tell me the location of my home directory, the latter is not only strangely named *data*, but I do not have the permissions required to enter it, as if I was … locked outside. In short, I am homeless. Of course, this point fits rather well with today's praise for mobility and a technological nomadism that grant us the right to work anywhere, at any time, all the time. Meanwhile these homeless devices help us get on with our productive and efficient times by maintaining the collective illusion of a one-size-fit-them-all lifestyle sold by ad-

Figure 4: Who am I, indeed.

vertisement companies and the software industry. In a way there is some eerie similarity between this mobile culture and the life of workers envisioned by Philip K. Dick in the “The Three Stigmata of Palmer Eldritch” in which a cult emerges around a drug used in combination with props called “layouts”, all that so as to escape from reality and share their common experiences in an

idealised and simulated narrative.[11] The good news is: we too have layouts, they are called apps.

User as app

In the field of computing, an application is a type of software that is directly relevant to the computer user, like an Internet browser, as opposed to a system library that is a software not directly beneficial to the users but nonetheless necessary to the functioning of other software, including applications. The term app is therefore a shorthand for application. With the increasing organisation of software piracy in the nineties, apps or appz became a specific category of pirated software,[12] generally desktop and system utilities, however today the term is generally used to describe mobile and web applications. The term has been greatly popularised with the 2008 launch of the App Store,[13] Apple's very own software distribution channel, that quickly led to the creation of similar efforts from other companies. Enter a profusion of Google Play, Microsoft Windows Store, Amazon Appstore, Canonical Ubuntu Software Centre, Mozilla Firefox Mar-

11. Philip K. Dick, *The three stigmata of Palmer Eldritch* (New York: Vintage Books, 1991).

12. The Inner Circle, "FAQ," 1997, accessed May 10, 2013, `http://web.archive.org/web/19970508133415/http://www.aditom.se/~inner-circle/13.html`.

13. Apple Inc., "Apple - Press Info - iPhone App Store Downloads Top 10 Million in First Weekend," 2008, accessed May 10, 2013, `http://www.apple.com/pr/library/2008/07/14iPhone-App-Store-Downloads-Top-10-Million-in-First-Weekend.html`.

ketplace, and all the indie efforts made to provide commercial alternatives to the famous corporations. That said, the idea of helping users to manage and install software by the means of a dedicated application is not a new idea. Nearly all Unix-like systems comes with a package manager, such as aptitude on Debian based OS,[14] or a centralised repository of files facilitating the compilation and installation of software on a local machine, such as the Ports Collection on FreeBSD.[15] In these examples the whole software ecosystem and its infrastructure is flatten down and any software is no more important than any others: the obscure command line tool sits proudly next to a full blown office suite and a venerable system library. By promoting the application as some sort of *übersoftware*, what the app store model proposes instead is the metaphor of the shopping mall with its curated and sponsored selection of products and a market of over populated stands where hidden gems are lost in an endless stream of 0.99$ pieces of junk, all fighting for the attention of a user turned into a customer. Strangely enough, the whole market metaphor is even transforming the landscape of software publishing and distribution, as alternative app stores can be used to install pirated apps without having to pay for them,[16] and free

14. Osamu Aoki et al., "Chapter 2. Debian package management," 2012, accessed May 10, 2013, `http://www.debian.org/doc/manuals/debian-reference/ch02.en.html`.

15. The FreeBSD Project, "5.6. Using the Ports Collection," 2013, accessed May 10, 2013, `http://www.freebsd.org/doc/en/books/handbook/ports-using.html`.

16. Beijing YouRanTianDi Technology, "kuaiyong," 2012, accessed May 10, 2013, `http://www.kuaiyong.com/eg_web/announcement.html`.

and open source software apps can be found under the same roof in a dedicated store.[17] The app trend is literally a digital urban planning project for turning the software architecture of the OS into an electronic commercial city, from the main shopping centre and its well lit partitions to the dodgy sub directory black market and the messy hard-to-find niche boutiques binaries.

Anyhow, looking closer at the Android platform, the trick to get non Google blessed apps running on these Unix-like systems is bewildering as the system only favours the addition of new apps via its official channel. It means that you do not really install such alien apps, you *sideload* them.[18] Of course it is the same as installing, but the fact that a special word is being used to describe this trivial process is quite revealing once more of the diminished position of the user in this OS. With this semantic treachery, the freedom of installing software on your own machine is turned into a mechanism in which you are given the impression that what you are doing is not morally correct, you are loading something into the system via the back alley, instead of just installing it directly. You are put in a position where you are breaking a code of conduct, breaking a morale that has been embedded into the software architecture and the end-user license agreement (EULA) you quickly accepted when using the device for the first time. It does not feel right because this process make

17. F-Droid Limited, "F-Droid · About," 2013, accessed May 10, 2013, `https://f-droid.org/about/`.

18. CyanogenMod Wiki, "Basic concepts - CyanogenMod," CyanogenMod, 2013, accessed May 10, 2013, `http://wiki.cyanogenmod.org/w/Basic_concepts#.22side-loading.22`.

us realise that we do not own such places, they are are borrowed from an unknown party. Next to that, if you want to sideload apps you need to specifically say that you want to install from unknown sources. Besides the irony of having to accept that the software you are more likely to know better than those found in Google Play is in fact unknown, it means that the mechanism of trust that would have been the outcome of a long human process, like in the case of the network of Unix-like systems that I connected to earlier, is now digitally implied and implicit. By using the device, it is taken for granted that an unknown *thing* will act as a surrogate to manage who you should trust or not by the use of digital signatures and certificates. Talking about trust, in the Android system, and for security purposes, individual apps run as unique UNIX users generated when the app is installed, and these apps run in their own sandbox with permissions granted upon their installation.[19] Effectively in this Unix-like OS there are no humans as users owning software processes, but instead apps as users owning software processes. Your own interaction with the system, like demonstrated with an Android terminal in Figure XXX is as a consequence also *sandboxed*, it is an app.\

Truth is, from the system perspective, you are an app and apps need no home but sandboxes.

19. Google Inc., "Android Security Overview | Android Open Source," Google Inc., 2013, accessed May 10, 2013, `https://source.android.com/tech/security/#the-application-sandbox`.

Nested inside legal sandboxes

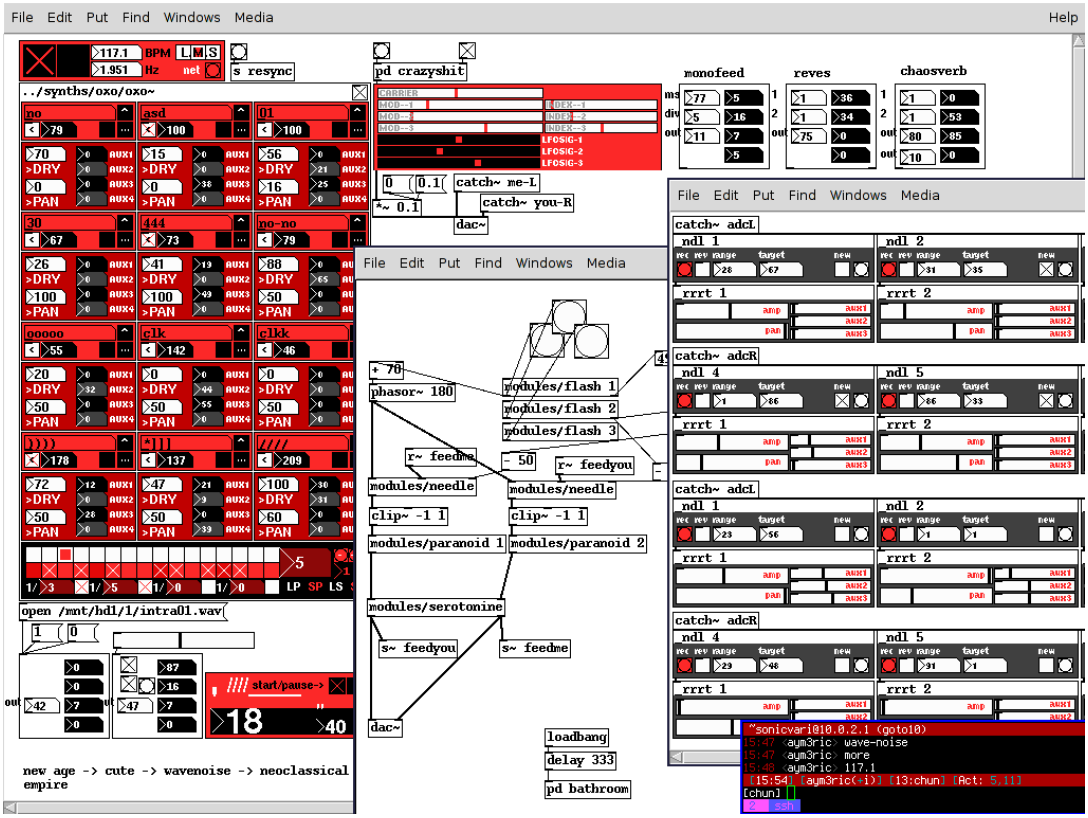

Figure 5: Screenshot of the Pure Data interface during a performance from the band 0xA.

Given that one is willing to take a closer look at the software infrastructures of these popular UNIX-derived operating systems, it is relatively easy to highlight the issue of ownership and control, and therefore make more visible boundaries that are often disappearing within the realm of a so-called transparency within computer systems. However, things can get much more complicated once we understand that these sandboxes often exists nested within one another through their legal apparatus. To illustrate this point I will now take the example of RjDj. RjDj

was an iPhone app released in 2008[20] that promised to change the way we consume music by bringing to the masses a generative and interactive sonic experience, optionally taking advantages of the different sensors present on the phone. It was developed originally as a platform, a new type of music label to some extent, for composers to contribute and distribute such pieces.[21] RjDj was not developed from scratch, its core component was in fact another software, Pure Data (Pd), which is a popular[22] cross-platform visual programming language used by artists, musicians and designers to write 'patches,' that are graphical representation of real-time multimedia processes used for live performances, installations, audiovisual creations, etc.[23] Said differently, from the perspective of existing Pd users, RjDj was a mobile Pd patch player. For the Pd developers, and without entering into technical details that are beyond the scope of this paper, RjDj was an inspiration and a new impulse to revisit an old desire[24] of decoupling the "engine" part of Pure Data from its user interface, thus creating a software library that could be used by other applications such as games, embedded systems and other audiovi-

20. Günter Geiger, "[PD] [PD-announce] RJDJ released," e-mail to the Pd-announce mailing list, October 10, 2008, accessed October 28, 2012, `http://lists.puredata.info/pipermail/pd-announce/2008-10/001314.html`.

21. Andy Farnell, "Re: Questions Pd/RjDj," e-mail to the author, May 28, 2013.

22. Ernesto Romero, dir., *FLOSSOFíA: El Software Libre en el Arte* (México: Centro Nacional de las Artes, 2009).

23. Frank Barknecht, "Pure Dataflow - Diving into Pd," in *Digital Artist's Handbook*, ed. Marloes de Valk (Lancaster: folly/GOTO10, 2007).

24. Chun Lee, "Art Unlimited: An investigation into contemporary digital arts and the free software movement" (PhD diss., MiddleSex University, 2008).

sual frameworks. After the combination of several efforts, most notably on the Android platform, the project libpd is released towards the end of 2010 and also find its way into the iPhone toolkit of the RjDj developers.[25] In February 2012 the website libpd.cc is launched next to a book on the topic published by open source best friend, O'Reilly Media.[26] A link to a web forum, distinct from the existing Pd list and bulletin boards community, is provided for discussions regarding the use of this library. From this point, there would not be much to say, this is just open source business as usual. However, in the "about" page of the site, the following statement can be read:

> libpd is Pure Data. It is not a fork of Pure Data, not a different flavor of Pure Data. It is simply a way of using Pd in a new way that can be more convenient and allows compatibility with mobile app development, game development, embedding into sophisticated 3D visualization tools, and lots of other applications. As such, it adds to Pd, without taking away anything from Pd Vanilla's DSP core. It has the same license as Pd, too. It is every bit as free and open source as Pd. As such, the project is hugely indebted to the entire Pd community, and to Pd's original creator, Miller Puckette. Those of us working with libpd have done

25. Peter Brinkmann et al., "Embedding pure data with libpd," in *Proceedings Pure Data Convention* (2011).

26. P. Brinkmann, *Making Musical Apps* (Sebastopol, California: Oreilly & Associates Incorporated, 2012).

> so because we're excited to see Pd patches running in more places than ever before, doing things they've never done before, and we trust you're just getting started.[27]

A close reading of the text reveals some obvious emphasising regarding: the fact that the project is not representative of a schism within the Pd community; freedom and openness of the software remain unchanged; large credits and kudos are given to the Pd community as a whole. Said differently: do not panic, let's make awesome apps. This way to positively promote and position libpd in relation to the Pure Data project while paying extensive tribute to its existing community, as well as focusing on the creative potential of the software is not accidental. Behind the technological innovation of RjDj and libpd there is a another story, that is the one of a community who, at the time of the arrival of these new projects, managed for more than a decade to grow and develop itself in some sort of autonomous and sometimes messy sandbox inhabited by academic researchers, musicians and artists, yet all very much sensitive to the subject of free software and free culture due to the openness of Pd's source code and its tight links with several free and open source operating systems. The sudden appearance of RjDj is the wake-up call that there is something else in this Pd sandbox, and it is unclear if this thing invited itself in or was a built-in feature. Before going

27. Peter Kirn, "libpd » About," 2012, accessed May 10, 2013, `http://libpd.cc/about/`.

further, let's make a quick detour via the retelling of the context in which Pd was given birth in the first place. Indeed, one of the initial motivation for Miller S. Puckette to start working on Pd, back in the mid nineties, is to depart from frustrations he has with his former employer, Institut de Recherche et Coordination Acoustique/Musique (IRCAM), which made it very hard for him to disseminate his further research on the software Max that he wrote while employed by the institute. Therefore one of the other meaning of the Pd acronym is purposefully synonym of liberation from the IRCAM intellectual property sandbox: Public Domain.[28] So when Pd is announced to the world in 1996, it is in this spirit of dissemination of knowledge that Puckette concludes his paper, musing and wondering about the future of Pd, acknowledging the community aspect being as important as the software itself.[29] As it turned out the community that emerged around the software took Pd far beyond its author's wildest dreams.[30]

When RjDj is introduced in July 2008, it is done via one of the founders of the Pd community, in the form of an invitation to join intensive week-end working sessions, where selected Pd users would be flown over, fed, accommodated, yet unpaid, to contribute hacking and reflection on a new form of interactive music for mobile devices thought to be the next generation of Walkman or mp3 player in which the consumption of such algorithmic mu-

28. Miller Puckette, "Who owns our software? —- a first-person case study," in *Proceedings ISEA* (2004).

29. Miller Puckette, "Pure Data," in *Proceedings International Computer Music Conference* (1996).

30. Puckette, "Who owns our software? —- a first-person case study."

Figure 6: Promotional digital flyer for RjDj.

sic would result in effects similar to the one of taking drugs.[31] At the time of the first announcement it is stated that the project was built using several open source components and that most parts of the project would be released as open source software in return. It is only a few months later, and after few more sprints around Europe, that the definitive form of the project becomes clear. RjDj is not yet another artistic use of Pd, it's a project from a technology startup: Reality Jockey, Ltd. So while the project got a very good mainstream media attention, it left perplexed

31. Günter Geiger, "[PD] Announcement Kickstart RJ," e-mail to the Pd mailing list, July 2, 2008, accessed May 10, 2013, `http://lists.puredata.info/pipermail/pd-list/2008-07/063497.html`.

some members of the Pd community seeing their favourite creative sandbox suddenly exposed in mainstream tech and gadget blogs, while still puzzling about how exactly these exciting developments and playful hacking sessions around Pd led within just three months to the apparition of a company selling tens of thousands of $0.99 Pd-derived apps. To complicate things further some active Pd developers and contributors who were essentially doing voluntary work within the Pd community got invited into the RjDj project, via other developers already involved, and for which they were offered a job to do what they loved: play in the sandbox canvas of Pd, which for anyone who has been seriously involved in a free and open source software project as unpaid contributor is an offer that is very hard to refuse. Chris McCormick, one these lucky developers, recalls: "the RjDj thing seemed like it would be pretty wild and the pay was good so I went with that."[32] The sandbox would not be a holiday destination anymore but an alluring main residency where a new home could be built, financed by this startup.

Most importantly what this event will demonstrate is the two different, conflicting, approaches to free software licensing within the Pd community. Such conflicts returns whenever the topic of licenses and freedom is brought back and can be summed-up as the opposition between the pragmatism of open source software practice and the philosophical, possibly political stand taken by free software supporters. The debate is old and

32. Chris McCormick, "Re: question RjDj," e-mail to the author, May 31, 2013.

tiresome but is always revived every time someone either link technical discussions with their legal context,[33] or somehow addresses an implied political or ethical property in the openness of a software's source code.[34] To go back to the analogy of the sandbox, the enthusiasm to build together a whole world led to some positive feedback loop of generosity and mutual help within the Pd community that might have made everyone forget to read or ignore the very rules of the sandbox, literally putting trust and ethics before their legal reality. Yes indeed, making sand castles together is fun, reading licenses is awfully boring. In fact what RjDj and libpd highlighted was the lack of understanding of these rules that were without much consequences as long as the community was partly isolated from the rest of the world. With RjDj, Pd users and supporters understood that Pd was not a free software with a copyleft license[35] like the emblematic GPL that requires modifications to be shared back under the same conditions, but it was released instead with a permissive modified open source BSD license which allows for the incorporation of Pd code within any proprietary and closed source systems without the need to share anything back. After all Pd has been in the past used to provide the building blocks of the real time au-

33. Marvin Humphrey, "[PD] Keyboard shortcuts for "nudge", "done editing"," e-mail to the Pd mailing list, September 24, 2011, accessed May 10, 2013, `http://lists.puredata.info/pipermail/pd-list/2011-09/091294.html`.

34. Damian Stewart, "RjDj presentation - FAQ with audience," make art, GOTO10, November 29, 2008.

35. Scott R. Looney, "[PD] rjdj is gone, robotcowboy is coming ...," e-mail to the Pd mailing list, November 3, 2012, accessed May 10, 2013, `http://lists.puredata.info/pipermail/pd-list/2012-11/098781.html`.

dio synthesis objects of the proprietary closed source software Max/MSP[36] and was also used as sound engine in Electronics Arts game spore.[37] Why this was not seen as negatively as with RjDj is open to speculations: maybe the sandbox fun worked as a smoke screen; maybe the fact that the interleaving between Pd and Max stories created a shared respect towards their original author, Miller Puckette, thus making the search for skeletons in the closet rather distasteful; maybe the use of Pd in spore was perceived by users as a naive software bundling and distribution as a whole, without consequences. Alternatively, the fact that RjDj appeared to grow partly from the community brought a less romantic transparency on such mechanisms, and highlighted the necessary public deconstruction of software, both from a technical and legal aspect. The latter is certainly the element that made all these issues more comprehensible for the sandbox citizens. As it turns out due to conflict between Apple developer's agreement and the General Public License (GPL), it is not possible to distribute copyleft free software such as GPL software in the iPhone app store, which means that the developers of RjDj and libpd made explicit that such licensed software, which is the case for some popular externals (plugins) distributed as part or next to Pd, should be therefore either avoided or re-licensed by their respective authors prior to their use for the creation of iPhone

36. Cycling '74, "FAQ: Max 4 « Cycling 74," 2013, accessed May 10, 2013, `http://cycling74.com/support/faq_max4/`.

37. Mark Danks, "[PD] Pd in video game Spore," e-mail to the Pd mailing list, November 11, 2007, accessed May 10, 2013, `http://lists.puredata.info/pipermail/pd-list/2007-11/056300.html`.

apps. The result of that is two fold: first, the penny drops, the Free/Libre/Open Source Software (FLOSS) ecosystem is not one unified sandbox but a constellation of different worlds with their own rules mirrored in the many externals and derived projects surrounding Pd; second, to accept contribution to its project from the Pd community RjDj has to find a legal construction in which such contributions would have to be provided with a permissive license so it can be used in an iPhone app. Practically speaking, those willing to provide copyleft GPL work instead, would have to assign the copyright of these files to Reality Jockey. In layman's terms, it means that the authors willing to participate in the development of some core components of the app must give up their rights to the company who can then re-license the outcome of such participation the way they want so as to incorporate them into an iPhone app. It is this particular trick that triggered most criticism, namely how the GPL was therefore used as "a firewall to protect commercial interests on a closed platform, while exploiting the work of a free software community."[38] Next to that public tension, similar issues were going on within the project itself. Requests from Reality Jockey's CEO to remove the license and copyright notice of Pure Data to further close the project, was "one drop too much"[39] for Paul Brossier, who at the time had contributed essential work on the audio engine and GUI. As

38. Christopher Alex McLean, "Alex McLean | The iPhone and toilet paper freedom," 2009, accessed May 10, 2013, `http://yaxu.org/the-iphone-and-toilet-paper-freedom/`.

39. Paul Brossier, "Re: question RjDj," e-mail to the author, June 4, 2013.

a result of this conflict the developer left the project. Regardless if these discords were the result of a well executed business plan of action or a clumsy attempt to satisfy several agendas, and their sandboxes, the process affected deeply some Pd/RjDj developers who witnessed the whole process:

> I was very ambivalent. Always cheerful and enthusiastic about the team, the technical aspects of the project and its potential, but always on edge and suspicious. [...] There are last-minute meetings that some people do not get to hear about because they "are only technicians". Suits begin to appear that nobody knows. We hear about "great opportunities" which fortunately were raised, but how "compromises will have to be made" And the good people who believe in something more than money, smell the wind and start leaving. This was turning point for me because in it I saw a lot of my fears about the driving of a tech startup come true. The crisis and growing-up, for me, was to see that these things are how it works for this kind of people and are not reflections of my own cynicism or paranoia. Many companies use the idealism of young people as a weakness. Interest in openness, software freedom, innovation, passionate creativity exist as long as they are useful to recruit and train the team. Once the company starts making

business transactions all that goes out the window.[40]

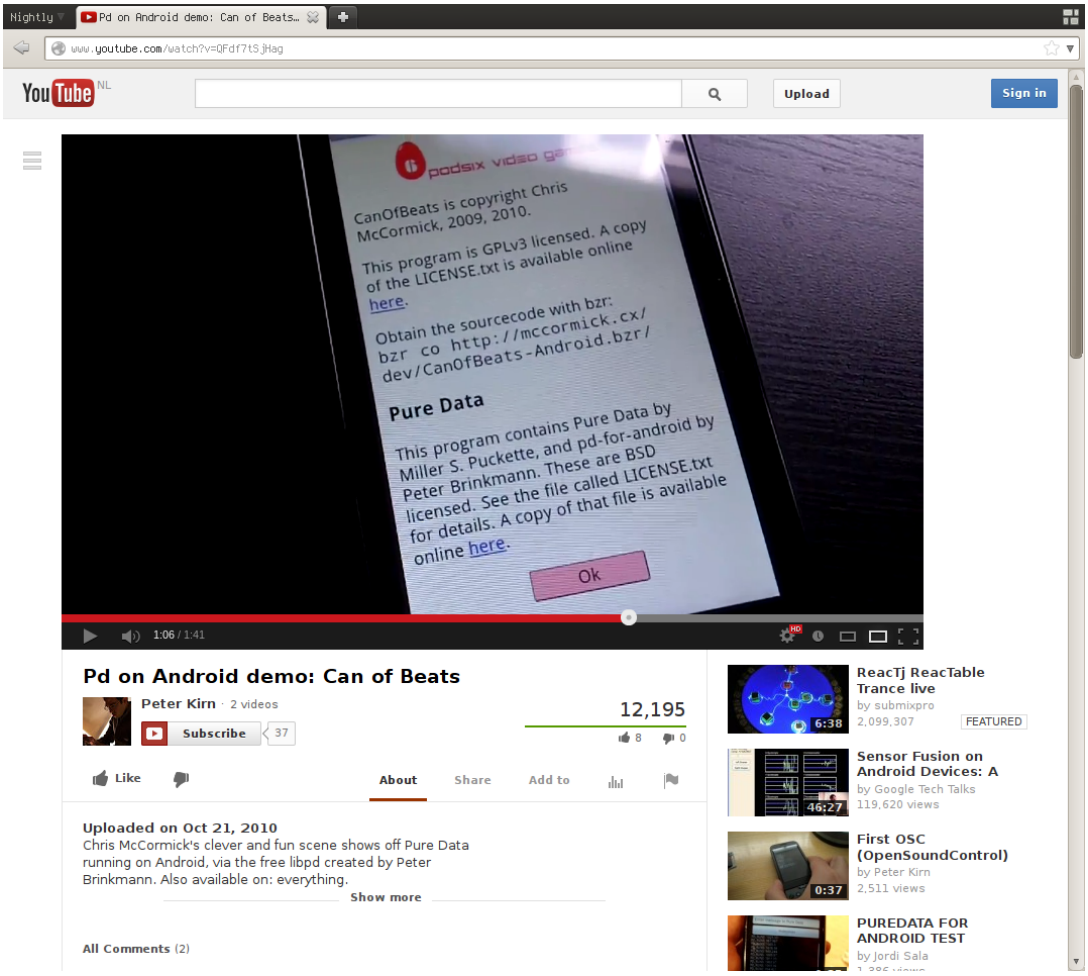

Figure 7: Credit screen of Chris McCormick's CanOfBeats Android app as demonstrated by Peter Kirn. The commercial app was developed during McCormick's free time while employed at Reality Jockey. It uses libpd and is licensed under the free software copyleft GPLv3, allowed on Google Play.

This perspective highlights precisely the impact that such sandboxes have on the affect of its participants and the damages they can potentially make that are neither visible in the "awesome" sugar coating of the positivist and productivist loop of so-

40. Anonymous, "Re: question regarding RjDj," e-mail to the author, May 14, 2013.

cial media driven tech startup scene, nor in the pragmatic "it just works" slogan from Apple. Of course let's not reduce this to yet another argument of free labour and exploitation for in this story, we are dealing with nested sandboxes, and things can never be black or white as Chris McCormick reminds us:

> [...] I think I always knew it was a proprietary company and I was paid to do a job within the confines of that. I am not even sure it's accurate to say that it was any less transparent than other proprietary companies, especially startups which are notoriously secretive. In the end I was happy that we got some things released under Free Software licenses let alone the whole stack. Actually I think if you look at other startups in the music space, we were releasing a lot more stuff as Free Software than others did, so I feel good about that. Some of it is even in use today and it really inspired some cool projects that wouldn't have happened without RjDj.
>
> If I had have felt like we were actively violating any Free Software licenses that would have been a different thing, but I felt like myself and a couple of other people in the company worked hard to make sure that wasn't happening[.][41]

Still, RjDj was also established within Apple's kingdom. The

41. McCormick, "Re: question RjDj."

latter's growing presence in the project, and its influence to dictate indirectly the form and conditions of the link with the Pd community, eventually dragged all the energy left in the project.[42] Finally in October 2012 the RjDj app is removed from the App Store. From a software perspective, the RjDj project is far from being anecdotal as it provided new perspectives and horizons for Pd users and developers and finally unified efforts around the creation of one library that can expand Pd's territory to other sandboxes. At the same time the project inspired new derivative works, frameworks, players, apps and whatnot either closed, or open, or in some sort of legal grey zone. It also provided new commercial opportunities for Reality Jockey Ltd. that is now reusing parts of the RjDj software in other commercial apps.[43] However, the whole story had arguably mainly an effect on the people who witnessed and participated in this process and it is unclear how this will transform Pd's social landscape in the long term, now that the sandbox participants have seen the cracks in the wall, seeing how the source code of an artistic iPhone app can be turned into a hub for different opinions, ideologies, philosophies and practices to collide, not always in a pleasant way.

42. Anonymous, "Re: question regarding RjDj."

43. Farnell, "Re: Questions Pd/RjDj."

Conclusion

Sandbox culture is a term sometimes used in the context of open ended games or virtual world platforms to describe the different activities happening within and around these environments. However, as I demonstrated in this paper, the idea of a sandbox culture can go much beyond the boundaries of software rendered virtual realities and can easily manifest itself through any software, licenses and their users. Moreover, rarely it occurs that the relationship with these objects happens also to be sandboxed at another level. In fact, sandbox culture gives the impression that it borrows properties both from the nesting effect of Matryoshka dolls, for the relationship it creates outside and inside the sandbox, and from the topology of a Möbius strip, for the circularity of the cultural chain of dependence it generates. While this might sound like some hard science-fiction one-liner, what I am talking about here bares little to no resemblance with the questioning of reality, whether it is virtual or hyper in the *simulacresque* sense. It deals in fact with the much more prosaic process in which technology and its legal apparatus, by the means of manipulation or misunderstandings, generates a new imaginary, a magical thinking that inspires novel forms of organisation and production and in return calls for the creation of more technology to support the newly bootstrapped culture. What is interesting here is not so much the process itself but how it affects our relationship with others and with the said technology. On their own, all these sandboxes have the potential to be perfect

friction-less standalone universes and being able to detect their existence is far from being as trivial as taking a fictional drug or compiling a snippet of C code. The awareness of a sandbox is most of the time accidental. It appears through the tensions and conflicts that raise from the inability to conciliate the animal flesh and the programmatic rationalism of the rules, the codes, that are written to govern it. Whenever they occur, such tensions and conflicts produces a temporary tearing of the sandbox fabric, the artificiality of its decor becomes visible and through the rupture of its emulated transparency, information bleeds and leaks through. Most importantly, it is precisely at this point that the effect of the formatted and constrained sandbox mindset becomes suddenly tangible for those who have for too long inhabited it and suddenly realise the penalty of sedentary lifestyle in such fragile constructions.

If looking at cultural processes through the lens of the remix allows us to envision a generous gift economy. The lens of the sandbox gives us as a counterpoint a much bleaker vision in which there is no empowerment, no individuation, just a constant replay of the same mechanisms that permits a debt to grow with time. Yes, a creative debt economy arises from the sandbox culture and spoils the gift, for its victims become literally in debt towards the playground they are given and without the knowledge to truly appropriate and reconfigure it, to reprogram it, they are left jailed and condemned to contribute to a culture that is linked and interlocked within one or several walled gardens they have no control over. In the sandbox culture, the only

possibility that seems offered to cancel the debt is therefore to create a new sandbox, from scratch or based upon another one, in the hope of displacing the debt towards those who will hopefully populate it and keep alive this grand cultural pyramid scheme. Is there any escape possible that does not result in elevating our status from slaves to masters? Maybe the prince of simulacra can help us here. In the opening of La prise d'otage, Jean Baudrillard states that it is impossible to destroy the system by a direct revolution, whether it is through implementing a logical contradiction or via overturning the balance of forces in place, because everything that can produce energy just feeds back into the system,[44] the latter is certainly true in the context of sandbox culture. He suggests that the system must be given a gift to which the only response possible will lead to its own death. Yes, however, I seriously doubt that there is an app for that. What is worrying here is that the phenomenon that I am expressing with the allegory of the sandbox does not solely concern groups exclusively busy with apps and artistic use of software and licenses. In fact it could be easily stretched to collectives, practices, tools, beliefs, education, any objects. In a society that is increasingly programmed and scripted for efficiency and productivity, the binary nature of the software apparatus gives less and less room for negotiation, hesitation and reflection, no room for trust to be explored and grown at a human pace once it is outsourced and mediated by techno-legal infrastructures. Looking

44. Jean Baudrillard, "La Prise d'Otage," in *Le ludique et le policier : et autres textes parus dans Utopie, 1967/78* (Paris: Sens / Tonka, 2001), 335–344.

at the social component of proto-, past and current derived Unix systems, and in the true spirit of cybernetics, we can see that by building digital infrastructures to inhabit we have slowly moved some fundamental social mechanisms into these architectures to the point where they became indistinguishable. We should not be surprised if today we are at their mercy. As Ken Thompson, author of the original Unix system, mentioned once: "[y]ou can't trust code that you did not totally create yourself. (Especially code from companies that employ people like me.)"[45] Maybe the way to work around this issue is counter intuitive at first and would require us to fully embrace a nomadism that is not forced upon us but chosen consciously, almost in a cynical, yet meaningful, way to go beyond and further the opportunistic behaviour of the free market: the creation of meta communities of travelling parasites who will develop a practice not within sandboxes but across them, making sure they do not fall in love with any of them and who will hopefully succeed in contaminating others, encouraging and teaching them how to trust again, to travel light, and forever avoid the traps that all these so-called homes[46] have turned into.

45. Ken Thompson, "Reflections on trusting trust," *Communications of the ACM* 27, no. 8 (1984): 761–763.

46. Colin Dixon et al., "The home needs an operating system (and an app store)," in *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks* (ACM, 2010), 18.

References

’74, Cycling. “FAQ: Max 4 « Cycling 74.” 2013. Accessed May 10, 2013. http://cycling74.com/support/faq_max4/.

Anonymous. “Re: question regarding RjDj.” E-mail to the author. May 14, 2013.

Aoki, Osamu, et al. “Chapter 2. Debian package management.” 2012. Accessed May 10, 2013. http://www.debian.org/doc/manuals/debian-reference/ch02.en.html.

Apple Inc. “Apple - Press Info - iPhone App Store Downloads Top 10 Million in First Weekend.” 2008. Accessed May 10, 2013. http://www.apple.com/pr/library/2008/07/14iPhone-App-Store-Downloads-Top-10-Million-in-First-Weekend.html.

Barknecht, Frank. “Pure Dataflow - Diving into Pd.” In *Digital Artist’s Handbook*, edited by Marloes de Valk. Lancaster: folly/GOTO10, 2007.

Baudrillard, Jean. “La Prise d’Otage.” In *Le ludique et le policier : et autres textes parus dans Utopie, 1967/78*, 335–344. Paris: Sens / Tonka, 2001.

Beijing YouRanTianDi Technology. “kuaiyong.” 2012. Accessed May 10, 2013. http://www.kuaiyong.com/eg_web/announcement.html.

Brinkmann, P. *Making Musical Apps*. Sebastopol, California: Oreilly & Associates Incorporated, 2012.

Brinkmann, Peter, Peter Kirn, Richard Lawler, Chris McCormick, Martin Roth, and Hans-Christoph Steiner. “Embedding pure data with libpd.” In *Proceedings Pure Data Convention*. 2011.

Brossier, Paul. “Re: question RjDj.” E-mail to the author. June 4, 2013.

CyanogenMod Wiki. “Basic concepts - CyanogenMod.” CyanogenMod. 2013. Accessed May 10, 2013. http://wiki.cyanogenmod.org/w/Basic_concepts#.22side-loading.22.

Danks, Mark. “[PD] Pd in video game Spore.” E-mail to the Pd mailing list. November 11, 2007. Accessed May 10, 2013. http://lists.puredata.info/pipermail/pd-list/2007-11/056300.html.

Dick, Philip K. *The three stigmata of Palmer Eldritch*. New York: Vintage Books, 1991.

Dixon, Colin, Ratul Mahajan, Sharad Agarwal, AJ Brush, Bongshin Lee, Stefan Saroiu, and Victor Bahl. "The home needs an operating system (and an app store)." In *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks*, 18. ACM, 2010.

Farnell, Andy. "Re: Questions Pd/RjDj." E-mail to the author. May 28, 2013.

F-Droid Limited. "F-Droid · About." 2013. Accessed May 10, 2013. https://f-droid.org/about/.

Gaylor, Brett, dir. *RiP!: A Remix Manifesto*. Montreal, CA: EyeSteelFilm, 2008.

Geiger, Günter. "[PD] Announcement Kickstart RJ." E-mail to the Pd mailing list. July 2, 2008. Accessed May 10, 2013. http://lists.puredata.info/pipermail/pd-list/2008-07/063497.html.

———. "[PD] [PD-announce] RJDJ released." E-mail to the Pd-announce mailing list. October 10, 2008. Accessed October 28, 2012. http://lists.puredata.info/pipermail/pd-announce/2008-10/001314.html.

Google Inc. "Android Security Overview | Android Open Source." Google Inc. 2013. Accessed May 10, 2013. https://source.android.com/tech/security/#the-application-sandbox.

Hafner, Katie, and Matthew Lyon. *Where Wizards Stay Up Late: The Origins Of The Internet*. New York City: Simon & Schuster, 1999.

Humphrey, Marvin. "[PD] Keyboard shortcuts for "nudge", "done editing"." E-mail to the Pd mailing list. September 24, 2011. Accessed May 10, 2013. http://lists.puredata.info/pipermail/pd-list/2011-09/091294.html.

Kirn, Peter. "libpd » About." 2012. Accessed May 10, 2013. http://libpd.cc/about/.

Lee, Chun. "Art Unlimited: An investigation into contemporary digital arts and the free software movement." PhD diss., MiddleSex University, 2008.

Lessig, Lawrence. *Remix: Making Art and Commerce Thrive in the Hybrid Economy*. London: A&C Black, 2009.

Looney, Scott R. "[PD] rjdj is gone, robotcowboy is coming ..." E-mail to the Pd mailing list. November 3, 2012. Accessed May 10, 2013. http://lists.puredata.info/pipermail/pd-list/2012-11/098781.html.

McCormick, Chris. "Re: question RjDj." E-mail to the author. May 31, 2013.

McLean, Christopher Alex. “Alex McLean | The iPhone and toilet paper freedom.” 2009. Accessed May 10, 2013. http://yaxu.org/the-iphone-and-toilet-paper-freedom/.

———. “Artist-Programmers and Programming Languages for the Arts.” PhD diss., Goldsmiths, University of London, 2011.

Puckette, Miller. “Pure Data.” In *Proceedings International Computer Music Conference*. 1996.

———. “Who owns our software? —- a first-person case study.” In *Proceedings ISEA*. 2004.

Raymond, Eric Steven. *The Art of UNIX Programming*. Boston: Pearson Education, 2003.

Romero, Ernesto, dir. *FLOSSOFíA: El Software Libre en el Arte*. México: Centro Nacional de las Artes, 2009.

Stephenson, Neal. *In the Beginning...Was the Command Line*. New York City: HarperCollins, 2009.

Stewart, Damian. “RjDj presentation - FAQ with audience.” make art, GOTO10. November 29, 2008.

The FreeBSD Project. “5.6. Using the Ports Collection.” 2013. Accessed May 10, 2013. http://www.freebsd.org/doc/en/books/handbook/ports-using.html.

The Inner Circle. “FAQ.” 1997. Accessed May 10, 2013. http://web.archive.org/web/19970508133415/http://www.aditom.se/~inner-circle/13.html.

Thompson, Ken. “Reflections on trusting trust.” *Communications of the ACM* 27, no. 8 (1984): 761–763.

Timar, Ted. “Unix - Frequently Asked Questions (6/7) [Frequent posting].” comp.unix.questions. Unix-faq/faq/part6_1084272547@rtfm.mit.edu. May 11, 2004.

Unsworth, John. “Living Inside the (Operating) System: Community in Virtual Reality.” In *Computer Networking and Scholarly Communication in the Twenty-first-century University*, edited by Teresa M. Harrison and Timothy Stephen, 137–150. Albany, New York: SUNY Press, 1996.

Veal, Michael. *Dub: Soundscapes and Shattered Songs in Jamaican Reggae*. Middletown, Connecticut: Wesleyan University Press, 2007.

Wheeler, David A. “Secure Programming for Linux and Unix HOWTO” (1999).