Practical and ethical aspects of FLOSS

Mgr. Michal Kohútek

May 14 2019

1 Introduction to FLOSS licenses

Most regular users of computer software do not recognize various different software licences, nor do they know of the existence of neither Open Source nor Free Software movements. It is understandable, as even though they permeate both work and personal lives of almost everyone in the developed or developing countries, computers are still seen as arcane and deeper knowledge of their inner workings is restricted to IT personnel, computer scientists and enthusiasts. It is also true that neither Open Source nor Free Software have widespread equivalents in the physical world. It isn't a trivial task to explain the mechanisms by which such movements survive, or even thrive in our modern society. In this essay, I will try to accomplish the task of explaining what Open Source is and why it has taken the computer world by storm.

There is a distinction to be made between Free Software and Open Source, as they both describe almost the same range of programs, however they differ in a (mostly) philosophical dimension. To prevent confusion of the users and creators alike, an umbrella term for software belonging to the both categories has been coined - FOSS[1]. It means that anyone is freely licensed to use, copy, study and change the software in any way, and the source code is openly shared so that people are encouraged to voluntarily improve the design of the software [1]

Free Software is any program that provides users with four essential freedoms:

- The freedom to run the program as you wish, for any purpose (freedom 0).
- The freedom to study how the program works, and change it so it does your computing as you wish (freedom 1). Access to the source code is a precondition for this.
- The freedom to redistribute copies so you can help others (freedom 2).
- The freedom to distribute copies of your modified versions to others (freedom 3). By doing this you can give the whole community a chance to benefit from your changes. Access to the source code is a precondition for this.

Notice, that in this context, "Free software" doesn't mean "non-commercial". In fact, a free program must be available for commercial use, its development can be paid for, but once you get your copy of it, you always have the freedom to copy and change it and even to sell copies. According to GNU.org, to understand the concept, you should think of "free" as in "free speech," not as in "free beer".

The difference between Free Software and Open Source Software is mostly in different underlying values. As Richard Stallman, founder of GNU explains: "Roughly, it [the concept of free software] means that the users have the freedom to run, copy, distribute, study, change, and improve the software." On the other hand, "Open source" focuses on the practical consequences enabled by these licenses: surprisingly effective collaboration on software development. Free software came first. Later, it became apparent that free software was leading to remarkable collaboration dynamics. In 1997, Eric Raymond's seminal essay "The Cathedral and the Bazaar" focused attention on the implications

[1] Free and Open-Source Software

that free software has for software development methodology [2]. The closest to a neutral term would be FOSS (free and open source software) or FLOSS (free/libre/open source software), which have had limited success fulfilling that value-neutral role.

There is a plethora of licenses covering not only software licences, but also art, multimedia, hardware designs, etc. In context of FLOSS, these usually fall in two categories: permissive and "copyleft" licences. The main difference is that permissive licences usually require little more than attributing the original portions of licensed code to the developers and exemption from liability. As of 2018, the most widely used permissive licence is permissive MIT license. Copyleft licences stipulate that the same rights be preserved in derivative works created later on. The most widely used copyleft licence is GNU GPL [2], originally written by Richard M. Stallman.

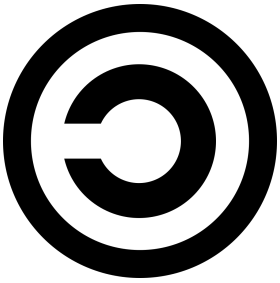

Figure 1: Copyleft - Zscout370, Sertion, e.a.

2 FLOSS projects are on the rise

In recent years there has been a worldwide push towards development and promotion of FLOSS movements across various academic fields and industry. Dr. Giorgio F. Signorini, PhD in his paper "Open Source and Sustainability: The Role of University" argues, that IPRs [3] do not fulfill their role of disseminating technical innovation by securing a form of reward for the research investments. In his paper, he highlights many studies suggesting that a different paradigm may be more effective in fostering innovation. Signorini also identifies two main ways IPRs hinder development of poorer nations: By limiting people's access to knowledge through copyright and by restricting the use of novel technologies through patents.

Signorini references the differences between "Free" and "Open Source" as they were perceived in the beginning of such movements. He notes, that despite important semantic distinctions, "Open Source" has now come to assume a much broader meaning than the words themselves encompass, and OSI's [4] now widely recognized definition of Open Source Software closely resembles the definition of Free Software by the FSF:

"Generally, Open Source software is software that can be freely accessed, used, changed, and shared (in modified or unmodified form) by anyone [3]"

[2] General Public License
[3] Intellectual Property Rights
[4] Open Source Initiative

Although there are still fine differences between FSF, OSI and other definitions, a portmanteau "Free (Libre), Open Source Software" (FLOSS), effectively transmits the notions of both freedom and openness and thus will be, for the sake of brevity, used thereafter.

Signorini lists the features of FLOSS, such as quality, reliability, flexibility and independence from vendor and asserts that they are the direct consequences of the two basic rights : the right to access and the right to actively use the software.

Quality - There used to be a time when there was a widespread view that "since Open Source software is free, it must be of low quality". Contrarily, studies of the last two decades shown many FLOSS products to be highly **reliable** and in many cases outperform proprietary systems. It also is flexible to a high degree, which means that it can be easily customized to meet new needs, and that it can be very resilient to changes in the environment.

Independence from vendors - Proprietary software vendors are incentivized to create an environment in which clients are forced to keep using the same software even when it no longer meets their original needs, either via proprietary formats or tools incompatible with other platforms. As the user has complete access to all the algorithms or data formats, FLOSS has no such need and inherently leads to the formation of standards.

Signorini further describes the OS model and its possible applications outside of the software development, such as Open Source Hardware, Open Access and Open Education. When thinking about these applications, Signorini asks himself:

- can the OS model be exported to hard technologies? and perhaps, in a broader sense, to the domains of content publishing and education?
- which of the defining properties of FLOSS can also be applied to these areas?
- what are the differences?

In conclusion, Signorini proclaims that in order for the world to be sustainable and fair, the essential features of Open Source model can be transferred with minor changes to other fields of human activity, such as hardware and intellectual work in general. This scheme can be shown to be economically sustainable and in some cases, such as academic publishing, more sustainable than restricted-access scheme. Its potentially revolutionary impact on the current society needs to be a subject of further research.[4]

2.1 Open Source in Neuroscience

An interesting case of open source proliferation is a recent movement in neuroscience. Gleeson et al. have written a paper encouraging the neuroscientific community to embrace Open Source model in their research. They claim that although replicating neuroscientific experiments is hard for the general public (due to it requiring specialist hardware, access to transgenic animals or reagents, etc.), publishing the code and the algorithms allows greater review of computational analyses carried out on the data, thus increasing the transparency of the research.

Increasing number of researchers have made an active, public commitment to open sharing of code. Those who made this pledge commit to making all software tools, libraries, etc. they develop for experimental data analysis or model construction open source at the time of publication, whether or not the software is the main subject of the paper. If and when asked to serve as peer reviewers, they will henceforth ask authors about the availability of any code they have developed for data analysis and modelling that is essential to reproducing the results of their paper and require it to be shared publicly.

One barrier that prevents many researchers to share code is their belief that it isn't sufficiently well-written or documented to be useful to others. Authors aren't obliged to provide support for the code they've created, they only claim that it can be used to reproduce the results of that specific publication.

Open sourcing the code allows others to find the parameters, algorithms and/or assumptions used in the analysis or model that may be missing from the paper. Moreover, it also allows early feedback from the community and can increase the usability and quality of the code. That way, authors receive more instant recognition for their contributions without having to wait until the publication of their work.

That being said, not all software can be shared in this way. Some software, such as drivers for custom hardware, would be of little use in other context. There may be a valid reason why source code for a software developed for commercial purposes cannot be released, not least if there has been industry funding that restricts the rights to the code. For most cases, however, open sourcing the code can improve the scientific worth of a publication as well as benefit the scientific community in general [5].

In November 2017, a workshop was organized at University of Warsaw, Poland. This hackathon was organized by the Brainhack organization in order to promote interaction between researchers and to encourage open neuroscience. The event had nine projects on different topics, including functional connectivity research, white matter tractography, etc. The report authored by Andre Maia Chagas describes one such project, in which the participants used a poster child of open-source development - Raspberry Pi single-board computer for conducting experiments helping the understanding of neuronal communication of various species. The teams used an affordable, open source neurobiology lab, "the FlyPi", based on the Raspberry Pi. The device allows the ability to use several imaging techniques, such as fluorescence imaging, calcium imaging and high-resolution movement tracking. Apart from that, it also allows the modulation of neuronal activity using opto- and thermogenetics. The instructions to build the FlyPi can be found online [5] [6]

3 Importance of preventing vendor lock-in

One of the features of FLOSS, as stated by Santorini is prevention of vendor lock-in. Vendor lock-in is a mechanism by which the vendor makes customer dependent on him for products and services by implementing barriers preventing the customer from

[5] `https://zenodo.org/record/1486176#.W-sNfhCnzuQ`

switching to other vendor without significant cost, time investment or loss of features. Many large software companies that sell proprietary software or services attempt to a greater or lesser degree to lock their customers in their ecosystem. A few of the most well known examples include Microsoft, Apple and Google.

In 2004's ruling on Microsoft's business practices, the European Commision quotes an internal memo drafted by Aaron Conterer, Microsoft's general manager for C++ development for then CEO Bill Gates, stating that: "[Windows API] is so deeply embedded in the source code of many Windows apps that there is a huge switching cost to using a different operating system instead. It is this switching cost that has given customers the patience to stick with Windows through all our mistakes, our buggy drivers, our high TCO, our lack of a sexy vision at times, and many other difficulties.[...] Customers constantly evaluate other desktop platforms, [but] it would be so much work to move over that they hope we just improve Windows rather than force them to move. In short, without this exclusive franchise called the Windows API, we would have been dead a long time ago. The Windows franchise is fueled by application development which is focused on our core APIs"[7].

There are many other examples of vendor lock-in being a transgression to the users' rights, perpetrated by Apple (bundling of iTunes with iPod media player, locked marketplace of applications for iOS devices), Google (replacing open standard Google Talk by a proprietary Google Hangouts thus cutting off third-party clients). Google's Android is an interesting case, since while the core project is Open Source, increasing proportion of features are locked away in bundled proprietary Google services and applications. Apart from software, vendor lock-in is rampant in hardware as well. Notable perpetrators are printer manufacturers locking away the use of cheaper third-party ink cartridges [8] or glucose meters manufacturers causing needless deaths of diabetic patients [9]. As demonstrated by these examples, vendor lock-in has significant economic and health ramifications for the society and should be considered unethical.

4 Conclusion

We have come a long way from the time when software giants like Microsoft and Oracle fought teeth and nails against anything related to FLOSS. The market has changed and even them have slowly came to realize the economic benefit of Open Source development model. Microsoft, whose then CEO Steve Ballmer famously proclaimed GNU/Linux, a paragon of FLOSS, to be cancer is now a member of Linux Foundation, Apple has released its new programming language Swift under the Apache 2.0 licence with Runtime Library Exception and European Union has updated renewed it's OSS strategy, that puts a special emphasis on contribution to open source software projects [10]. Reuven M. Lerner wrote an article on the progress FLOSS has made in an article for Linux Journal. He applauds the changes in public perception of FLOSS, but points out the importance of teaching next generations not only the economic value, but also ethics concerning development of new technologies [11]. The fight is not over, but the progress is obvious.

In his paper, Vijai Dharmamony has collected and summarized a varied set of open source tools for many tasks usually done with proprietary (and often expensive) appli-

cations. He lists an alternative for working with photos, vector images, office tasks, 3D graphics, statistics, etc. In many cases, the selected software has almost full feature parity with their proprietary alternatives.[12]

FLOSS projects have come a long way to provide viable alternatives to many proprietary products. In the table below, I present the reader with a few such alternatives.

Purpose	Proprietary software	FLOSS alternative
Operating System	Microsoft Windows	Ubuntu (GNU\Linux)
Web Browser	Internet Explorer, Google Chrome	Mozilla Firefox
Office Suite	Microsoft Office	LibreOffice
Raster graphics	Adobe Photoshop	GIMP, Krita
Vector graphics	Adobe Illustrator	Inkscape
3D graphics	Autodesk Maya, Cinema4D	Blender
Video editing	Adobe Premiere, Windows Movie Maker	Pitivi, OpenShot, KdenLive
Sound editing	Adobe Audition	Audacity

Table 1: FLOSS alternatives

There are many more use cases for general-purpose computers and it is out of scope for this essay to list all the alternatives, but for most use cases there is a FLOSS alternative to proprietary software. Often there isn't exact feature-parity, but adjusting the workflow may be an acceptable compromise when taking vendor lock-in into consideration. For this reason, it is my recommendation to always include and prefer open alternatives when selecting a right tool for the task.

References

[1] The GNU Project GNU.org. *What is free software? The Free Software Definition.* 2019. URL: https://www.gnu.org/philosophy/free-sw.html.

[2] Scott K Peterson. *What's the difference between open source software and free software?* 2017. URL: https://opensource.com/article/17/11/open-source-or-free-software.

[3] Open Source Initiative. *Open source initiative: Frequently asked questions: what is open source software?* 2018. URL: https://opensource.org/faq#osd.

[4] Giorgio Signorini. "OPEN SOURCE AND SUSTAINABILITY: THE ROLE OF UNIVERSITY". In: (Oct. 2018). DOI: 10.13140/RG.2.2.18724.73608.

[5] Padraig Gleeson et al. "A Commitment to Open Source in Neuroscience". In: *Neuron* 96 (Dec. 2017), pp. 964–965. DOI: 10.1016/j.neuron.2017.10.013.

[6] Simone Monachino, Eric James McDermott, and André Chagas. "Building and hacking open source hardware". In: *Research Ideas and Outcomes* 4 (Dec. 2018). DOI: 10.3897/rio.4.e31701.

[7] European Commision. *Commission Decision of 24.03.2004 relating to a proceeding under Article 82 of the EC Treaty (Case COMP/C-3/37.792 Microsoft)*. 2014.

[8] Declan McCullagh. *Lexmark invokes DMCA in toner suit.* 2003. URL: http://news.cnet.com/2100-1023-979791.html.

[9] Aisling Babaria Palav; O'Riordan. *A Haitian Boy's Needless Death From Diabetes.* 2013. URL: https://www.nytimes.com/2013/11/15/opinion/a-haitian-boys-needless-death.html?_r=0.

[10] European Commision. *Open source software strategy.* 2018. URL: https://ec.europa.eu/info/departments/informatics/open-source-software-strategy_en.

[11] Reuven M. Lerner. *Open Source Is Winning, and Now It's Time for People to Win Too.* 2019. URL: https://www.linuxjournal.com/content/open-source-winning-and-now-its-time-people-win-too.

[12] Dharmamony Vijai. "Free and open source software for everyone". In: Dec. 2018, pp. 14–19. ISBN: 0036-8512.