Business Models in FLOSS

Daniel Heger, Matthias Reisacher

February 9, 2020

Abstract

There are many open source software companies that generate large amounts or revenue based on a product for which the code is openly accessible and documented. This disproves the notion of earlier days that having a closed source product is the only way to keep being competitive. There are a number of business models applied by companies that publish or maintain open source programs to still earn profits and develop a good product despite making their code openly published - or even because of it. In this paper, we first give an insight about the differences between closed source and open source, between copyright and copyleft, and we also provide a very short overview of the historical context of the creation of open source and copyleft, as well as the benefits the companies or an individual can get out of publishing a software as open source or copyleft. Then, we give an overview of possible business models that are used and operate on its products code being open source. Finally, we give some examples of popular companies or products that operate with one or multiple of these business models.

1 Introduction

When we think of business models in the context of free, libre and open source software (FLOSS), it's usually not the first thing to think about business models based on this kind of software: The source code is available for free, usually automatic installers and useful documentation as well, and the term itself already denotes that the software is free. How can money then be generated regardless of this fact? How can companies survive, or strive, based on *free* software?

This is were open source business models come into play. Open source is typically viewed more as a cooperative approach to developing a product and less as a business model in itself [13]. However, in recent times, there were a number of large companies that were formed around the concept of FLOSS (e.g. Canonical) or that shifted their focus to the utilisation of and contribution to open source software (e.g. Microsoft). Considering that companies that base their business operations on open source software generate large revenues, it is clear that there are many other factors to sell your product besides just "selling your code".

Furthermore, providing a software product as open source can create a community of developers that spans the whole world. Collaboration on a large project where everyone all over the world with only a computer and access to the internet is able to - besides using it - improve the product, fix bugs, add new features, etc., not only makes the product a better one, but also gives everyone who wants to be a part of it a chance to bring themseleves into the project.

2 Copyright and Copyleft

When starting to develop a piece of software with the intent to release it to the public at some point, one of the first questions that usually has to be solved is: Which license should be applied to the software? This is the most crucial step that determines, whether a piece of software is proprietary or copyleft is considered to be applicable to it. This section gives some historical context on copyright and copyleft, outlines the difference between the concepts of copyright and copyleft/open source, and gives some insights of how copyleft can benefit software.

2.1 Historical context

Back in the early days of computers, when writing software still was mostly done at universities, and only research institutes or large companies had expensive hardware and software still was something only those large institutions would produce to suit their own needs. Back then, writing software was more of a profession and less of a hobby. Furthermore, in those times, source code was usually shared to be able to fix problems in the program or run it on different hardware configurations.

The proprietary way of writing and publishing a piece of software came about with the decision of the US government that source code are subject to copyright. From this decision on, the traditional way of publishing a piece of software has been the type of proprietary software which was owned and maintained by one company exclusively, namely the company that programmed or published it. The software was only published as precompiled binary and sold by means of purchasing licenses to be able to install, use or modify/improve it.

This approach was not liked by the "hacker culture" that preferred to fix bugs in programs or modify or add new features to them themselves. Those continued to share source codes in online communities (in some cases illegaly) to still be able to tailor some programs to their needs.

In 1983, Richard Stallman launched the GNU project. The main idea of the project was to create a package of software that doesn't put restrictions onto users. Everyone could modify the source code of those software products. And in 1985, the concept of *copyleft* was invented and implemented in the first license, the GNU General Public License, in 1989.

2.2 Copyright and proprietary software

The proprietary model is the standard for everything published that doesn't include any different license. Copyright automatically applies to all intellectual property for a certain period of time. During this period, the author of the piece of software, in this case the company that publishes the software, has the monopoly on its usage, modification, distribution, etc., and can sell licenses for those rights to third parties and ban the distribution of unlicensed uses, modifications, etc. As copyright applies automatically, this is the default case for anything that's "without a license", not just software.

Many companies still use this approach to generate their revenue. They sell licenses for using the software they provide, guard their source code behind closed doors, and prohibit modifications to or usage of their code that became known to the public ("leaked") for other projects. Some of those companies can still publish their product as *Freeware* with optional services or functions that have to be bought, however the inability to view and therefore also modify the source code defines the product as a proprietary software [17].

2.3 Copyleft and Open Source

In contrast to copyright, copyleft does not need for the user of the software to buy a license from the developer or similar things, in fact it provides much more freedom, especially to developers. Due to this fact, no money is gained from licensing fees, this has to be mitigated somehow. We'll discuss the business models arising from this challenge in section 3.

Copyleft uses copyright law "to remove restrictions on distributing copies and modified versions of a work for others and requiring that the same freedoms be preserved in modified versions" [1]. This means that anyone can modify a version of a product and release it themselves without any repercussions they would face if the traditional closed source model applied. Some open source licenses even permit open source code to be included and released in closed source software. These licenses are *permissive* licenses, as opposed to *restrictive* licenses.

Permissive licenses, like e.g. the MIT or BSD license, don't put any restrictions on the usage of the source code. It may be modified and can even be included into closed source projects, as long as the conditions, usually the inclusion of the license or mentioning of the author(s) of the code, are adhered.

Restrictive licenses, however, restrict the usage of open source code insofar that every modification or derived product, a product where (some of) the code was included, cannot add any additional restrictions to the use of the software, like e.g. the restriction to modify the code. It doesn't mean that the software has to be published under the *same* license, but it has to be a compatible license, one that doesn't add any further restrictions onto the software [2]. This is the reason why copyleft licenses are known as "reciprocal" or "viral" licenses [1].

What all of these licenses have in common is that they take care that the possibility to modify code exists. This means, in every product that incorporates an open license, the source code has to be publicly available. This is due to the fact that without openly available code, there can be no modification.

2.4 The benefits of copyleft

As can be seen from the widespread use and popularity of open source software, copyleft licensing provides many benefits to the end user as well as the creator or publisher of a piece of software.

- It can create an avid community around the software, where people work together to fix bugs, add documentation or new features without them only doing it for money. This means for companies, it can be a huge gain of cost-efficient resources, and for users of the software that they can incorporate features they need into it. By doing this they contribute to further development of the software.

- It's easier to publish software and to maintain it for individuals. They don't have to found a company and hire people who take care of license management, distribution, etc., but instead they gain the benefit of having the whole internet as potential contributors.

- For companies, it can be a way "to signal that the company wants this particular software to stay open". [14]. Lindman et al. argue that the license is a tool for the company to gain trust among potential members among the community.

- For individuals, it can be a worthwhile way to gain experience working on or contributing to a larger open source project. Furthermore, it contributes to their exposure as a developer or contributor, especially for future projects or applications.

- Open source generally improves the code quality by having multiple independent reviewers [16].
- Open source can set its own standards that are much easier accessible and better defined than the competition. This may prove a huge advantage against competing closed source standards.

3 Business Models

In the following subsections, we will give an overview over possible business models that are used in the context of open source software. Note that a company can use several of these together to maximize their revenue.

3.1 Support – The "pure open-source" model

The basic idea of this business model is to sell services and not software rights. The revenue is acquired by selling attendances such as deployment, production-oriented and integration services, training and workshops, bug fixes, or consulting services. However, the actual selling point of those companies is quality. Maintenance and support are not restricted to a single company. Nevertheless, customers choose to get these services from a particular vendor because they believe they will get a certain level of quality. Therefore, a customer's trust in the company is a critical factor.

3.2 SaaS - Hosting and professional services

The need for some software applications to keep the service, its infrastructure, and the network running around the clock, every day of the week can be an essential requirement. This task can be challenging and time-consuming for companies or individual persons due to the presence of hardware and software failures. Therefore, the software may be open source, but some people or businesses decide not to run a service themselves.

The business model of *Software as a Service* (SaaS) consisting among others of online hosting applications, custom adaptations, and the maintenance of services, sells convenience. Fully-managed versions of the software allow the user to experiment with the application, or to deploy it in production with a few clicks. Additionally, customers do not have to worry about operating the product in a steady-state, applying software upgrades, creating backups, or problematic downtimes.

3.3 "Dual-licensing" model

Dual licensing emerged as an open-source business model in the early 2000's, and describes a software component which is released under two licenses simultaneously: a traditional proprietary license and an open-source one, usually from the *GNU General Public License* [3] family. So the basic idea of this business model is to generate profit by monetizing intellectual property by providing customers a way to buy their way out of having a GPL-like license applied to their code. Since the pain point for companies comes when the GPL code is mixed with proprietary code, the concept only works when the software is not an entire program but a component. Otherwise, the copyleft license would not constitute a more significant problem, and the users could use and distribute the whole software under GPL.

The concept was pioneered by the company *MySQL* as the '*identical twin*' model, where the software was exactly the same under both licenses. That started to evolve into the '*fraternal twin*' model, where the open-source version differed from the commercial edition.

3.4 Open core model

The fraternal twin model, part of the dual-licensing model, evolved into the 'open core' model as the difference between the open-source version and the commercial version got more significant. The idea behind the open core business model is that a 'core', or feature-limited version, of a software product is operated as open-source software and provided freely, while additional features are provided commercially as proprietary add-ons plugins or extensions. But the open core model is not binary and can be implemented in many ways. It can vary in the mount of code that is actually open-source software, how disruptive plugins are, how the product is delivered, if commercial fixes are upstreamed or not, or the amount of user control.

3.5 Widget frosting

Widget frosting is a model where a company is selling a hardware device and releases an open-source toolkit or SDK that allows others to add functionality to the device. Here, most revenue is generated through sales of the hardware. At the same time, the free and open-source software is used in order to get better drivers and interface tools. While the downside for moving to open source is minimal for the company, it gives their customers the advantage of possible endless support. While hardware products have a finite production and support lifetime, customers in this business model have access to the driver source and can apply patches themself.

3.6 Founding and voluntary donations

Donationware is a business model that supplies fully operational unrestricted software to the user and requests an optional donation be paid to the provider. The author may set the amount of the donation, or it may be left to the user's perception of the software value. Software authors either use some variation of a 'Donate' button on their website in hopes of attracting the user's attention, or they are part of a funding initiative. Although funding platforms are based on the user's initiative and will to aid a software project financially as well, they often offer more details and transparency about the particular project.

3.7 Advertising-supported software

Another way to commercialize free and open-source software is to move towards the economic model of advertising-supported software. The concept is based on the popularity of the service and consists of 'selling' their users to other companies. For instance, companies can sell advertising banners on their website, they can sell the value of a default setting for specific services, or they can whitelist other company's software and advertisements. The amount of generated profit depends heavily on the popularity of the own website since the significance of the own product correlates with the total number of active users.

3.8 Branded merchandise

Some open-source organizations sell branded gifts and merchandise articles like t-shirts and coffee mugs. Branded merchandise is used as promotional material at conventions, but also sold in offices and online. Companies can use this activity as a branding exercise and as a way to make more money for the business.

3.9 Crowdfunding

Even though crowdfunding is not an actual business model, it represents a possible way for an open-source project to get started before they move on to one of the other business models. Crowdfunding is s model in which individuals or organizations contribute an arbitrary amount of money to a project. Online platforms allow collecting significant financial sums by raising small amounts of money from a large number of people.

4 Practical Applications

There is a wide variety of companies that operate on one or multiple of the business models mentioned in section 3. In the following subsections, we will give some practical examples for a couple of companies that operate on open source business models.

4.1 Red Hat

Red Hat is an American multinational software company and the most significant open-source company using the *support model*. The company offers multiple products, but their core technology is *Red Hat Enterprise Linux* (RHEL) and *Red Hat Directory Server*. Whereby, the latter one had a market share of 33.1% of the worldwide server operating environments in the year 2018 [11]. They also provide *CentOS* (Community Enterprise Operating System), a community-supported, free, and enterprise-class Linux distribution, which is based on RHEL. Furthermore, Red Hat has more than $1,300$ paying customers using *Ansible* to automate their IT procedures [9], and more than $1,000$ organisations that are using *OpenShift*, Red Hat's application container platform based on Docker [10]. They acquired a deferred revenue balance of \$3.4 billion in the fiscal year 2019, consisting mostly of subscription revenues from infrastructure-related offerings, which constitutes about \$2.1 billion of income. [12].

4.2 Android

Android is an operating system for mobile devices created by Google and released as an open-source to generate a platform audience and to receive contributions and feedback. The mobile operating system had a worldwide market share of $74,3\%$ [7] in 2019. Although this level of distribution offers Google diverse possibilities to generate revenue from this open-source product, we want to focus on one specific way, where Google uses the *open core model* to make a profit. Even though the Android mobile operating system is free to install, manufacturers need a special certificate to install the Google Mobile Services (GMS), which consists of Gmail, Google Maps, and the Google Play store. The *Google Play store*, a digital distribution service for Android apps, represents a key position since it is the only convenient way for users to install and update external apps. Therefore, manufacturers cannot refrain from the GMS products without drastically limiting the possibilities of extending the system for their customers. Google does not charge for a GMS license. Still, every Android device produced by a company needs a certificate from an authorized testing facility in order to apply for the free license. The fee for those certificates varies and is negotiated on a case-by-case basis between a company and a Google-approved, licensed manufacturer and can range from $\$40,000$ to $\$75,000$ [8].

4.3 Github

Github is a company that provides software development version control and is owned by Microsoft. It uses the SaaS model to generate profit by hosting distributed version control and source code management systems in combination with its own features, and is with almost 40 million users the largest host of source code in the world [5]. The offer a basic plan for free use with limited functionality, and and multiple plans with larger feature sets at various costs. This business model enabled GitHub to make $140 million in annual recurring revenue in 2016 [6].

4.4 WordPress

WordPress is a popular open-source content management system licensed under GPL, based on a template system and a plugin architecture. WordPress automates many tedious tasks involved in web publishing and includes a '5-minute installer', which allows any non-technician to launch a WordPress site with just a few clicks. The theme system allows users to change the look and functionality of a WordPress website without altering the code or site content. It offers a possibility for the community to make money as well. One of the largest commercial theme repositories is *Themeforest*, which currently has 48, 225 themes for sale [15]. On the other hand, plugins allow the users to tailor their site to their specific needs by extending the features and functionality of a website. Wordpress.org currently has 55,596 plugins available, which does not include premium plugins [18].

WordPress is split into the following two fundamentally different products, both with different owners:

- **Wordpress.org**
  The WordPress trademark and the Wordpress.org domain are owned by the *WordPress foundation*, a non-profit organization. It contains the open-source content management system, which is free to use and available under the GPL license, and is often referred to as 'self-hosted WordPress'. Since Wordpress is a non-profit organization, the primary source of revenue is through donations.

- **Wordpress.com**
  Wordpress.com is owned by a private company called *Automattic*, and they use a mixture of the SaaS model and advertising-supported software to generate profit. On the one hand, they sell convenience by offering web hosting of stripped-down versions of WordPress to their customers. They sell related services such as hosting, backup, and others to make the product suitable and exciting for non-technicians. On the other hand, users can choose between multiple packages. The basic plan for free use allows Wordpress.com to show advertisement banners on your site.

4.5 Wikipedia

Wikipedia is the largest digital encyclopedia in the world. Probably everyone, including you, has already used it at least once in their lives. And it is a very well-known brand. The Wikimedia Foundation that stands behind Wikipedia and many other Wiki-projects is a non-profit charitable organisation that runs only on donations. Some of these donators are themselves large IT companies that have a charity fund and donate parts of it to the Wikimedia Foundation to keep their services running [4].

References

[1] Aintzane Cabañes. Copyright vs. copyleft: a short introduction to these licenses. `https://aintza.wordpress.com/2010/01/28/copyright-vs-copyleft-a-short-introduction-to-these-licenses/`, February 2020.

[2] Ben Cotton. What is copyleft? `https://opensource.com/resources/what-is-copyleft`, February 2020.

[3] Free Software Foundation. Gnu general public license. https://www.gnu.org/licenses/gpl-3.0.en.html, February 2020.

[4] Wikimedia Foundation. Wikimedia benefactors. https://wikimediafoundation.org/support/benefactors/, February 2020.

[5] Github. Showing 39,954,951 available users. https://github.com/search?q=type:user&type=Users, February 2020.

[6] Github. Showing 39,954,951 available users. https://web.archive.org/web/20191026111345/https://medium.com/@moritzplassnig/github-is-doing-much-better-than-bloomberg-thinks-here-is-why-a4580b249044, February 2020.

[7] GlobalStats. Mobile operating system market share worldwide. https://gs.statcounter.com/os-market-share/mobile/worldwidea, February 2020.

[8] The Guardian. The hidden costs of building an android device. https://www.theguardian.com/technology/2014/jan/23/how-google-controls-androids-open-source, February 2020.

[9] Red Hat. Ansbile. https://connect.redhat.com/explore/ansible, January 2020.

[10] Red Hat. Openshift. https://www.redhat.com/en/about/press-releases/more-1000-enterprises-across-globe-adopt-red-hat-openshift-container-platform-power-business-applications, January 2020.

[11] Red Hat. Results 2019. https://www.redhat.com/en/blog/red-hat-leading-enterprise-linux-server-market, February 2020.

[12] Red Hat. Results 2019. https://www.redhat.com/en/about/press-releases/red-hat-reports-fourth-quarter-and-fiscal-year-2019-results, February 2020.

[13] Sandeep Krishnamurthy. An analysis of open source business models. 2005.

[14] Juho Lindman, Matti Rossi, Anna Puustell, et al. Matching open source software licenses with corresponding business models. *IEEE software*, 2011.

[15] Envato Market. Wordpress themes & website templates. https://themeforest.net/, February 2020.

[16] Antonio Onetti and Fabrizio Capobianco. Open source and business model innovation. the funambol case. In *Proceedings of the first International Conference on Open source Systems*, pages 224–227, 2005.

[17] Richard T Watson, Marie-Claude Boudreau, Paul T York, Martina E Greiner, and Donald Wynn Jr. The business of open source. *Communications of the ACM*, 51(4):41–46, 2008.

[18] WordPress. Plugins. https://wordpress.org/plugins/, February 2020.

About this document. This seminar paper was written as part of the lecture *Free and Open Technologies*, held by Christoph Derndorfer and Lukas F. Lang at TU Wien, Austria, during the winter term 2019/2020. All selected papers can be found online.[1]

[1] `https://free-and-open-technologies.github.io`