Introducing "Business Source": The Future of Corporate Open Source Licensing?

Michael "Monty" Widenius and Linus Nyman

> *"The license is the constitution for the community."*
>
> Eben Moglen
> Director-Counsel and Chairman
> Software Freedom Law Center

The benefits of the open source development model have been proven by the test of time; however, making this development model economically feasible can be challenging. In this article, Monty Widenius puts forth a suggestion for a new type of license, which is the result of the lessons learned from decades of work on open source, both as programmer and entrepreneur. The result, "business source", is a license that seeks to enable a business model that can both harness the benefits of open source while generating sufficient income for the program's continued development. The business source license automatically changes terms after a given period: an initial non-open source license with specific usage restrictions that transforms to a fully open source license on a specific future date.

In this article, we contribute a proposal for a novel license, a set of recommendations for managers, and a sample text of a business source license. This work will be of relevance to four main groups: i) those developing or managing a closed source program but who are interested in the benefits open source offers; ii) those managing open core programs; iii) projects in development; and iv) investors interested in funding open source projects.

Introduction

Open source is more than free software: it is a powerful tool that can be leveraged by companies to appropriate value (e.g., Carbone, 2007; timreview.ca/article/93). Open source software is increasingly commercially developed and supported (Wheeler, 2009; timreview.ca/article/229); in fact, a majority of open source development today is carried out by companies (Weiss, 2011; timreview.ca/article/436). However, choosing to "go open source" offers both advantages and challenges. Although proprietary software may, in the long run, be hard pressed to compete successfully in the same market with a complementary open source product (Lindman and Rajala, 2012; timreview.ca/article/510), maintaining a quality open source product requires contributors that are both skilful and knowledgeable. Establishing a strong community is considered vital to success (Byron, 2009; timreview.ca/article/258); however, it is unrealistic to expect the sporadic contributor to achieve complete knowledge of an entire codebase. To train up and maintain in-house programmers, however, requires a project to generate sufficient income to meet these demands.

In days past, there was something of an unspoken agreement that a company that used a lot of open source programs would also purchase services or assign developers to contribute to the program. This, in turn, supported the program's further development. However, over time, it became more and more common for companies to use open source without contributing to its development (Asay, 2013; timreview.ca/article/650). Whether due to a greater familiarity with open source as a concept, market instabilities and quarterly profit demands, or any other reasons, this approach is short-sighted in that it does nothing to ensure

Introducing "Business Source": The Future of Corporate Open Source Licensing?

Michael "Monty" Widenius and Linus Nyman

that the program in question can continue to evolve and improve over time. Continued success requires harnessing the power of open source while at the same time generating sufficient income to ensure the program's development and well-being. Finding the right business model and license are important preconditions for success.

Any business model that seeks to leverage the benefits of open source should maintain – to as great an extent as possible – the key elements of the open source development model. Indeed, it is in light of these benefits that open source business models must be examined. Therefore, we begin this article with a reminder of the central benefits of an open source development model. After this, we briefly discuss different types of open source projects, the more common business models, and the impact of licensing decisions. Finally, we introduce business source, a new type of license aimed at securing the benefits of open source while still enabling the generation of necessary income to fund its continued full-time development.

The Benefits of Open Source

From a developer's point of view, going open source is beneficial in that it helps spread the word about a product because it is easy to try out. A further benefit is community contributions, which can lower development costs; provide innovative solutions (sometimes even offering solutions the developing company would not have thought of); and may result in development in areas that are important to contributors but that the company might not have prioritized or realized the importance of including. Also, open source projects generally get more feedback and better bug reports than closed source projects, and have a faster average time from discovery to solution (e.g., see Schindler's [2007; tinyurl.com/l35oetx] comparison), thereby improving quality. The benefits of open source result in a more useful product, more market recognition, feedback, leads, partners, and sales opportunities as well as a strong trademark.

From a user's point of view, open source offers much in the way of sustainability. Given that users have the right to fork the code at any time, vendor lock-in, planned obsolescence, and similar initiatives are all but impossible to implement (Nyman and Lindman, 2013; timreview.ca/article/644). If a supplier removes important features, one can add them back in oneself; if the supplier stops supporting the version of the product being used, or abandons the program altogether, it is safe to assume that someone will fork the code and continue its maintenance and development. (For more on open source sustainability see the January 2013 issue of the *Technology Innovation Management Review*: timreview.ca/issue/2013/january) Furthermore, there is little risk for hidden trap doors or unexpected features (e.g. Amazon's ability to delete customers' Kindle books (tinyurl.com/9eewrw5) and Microsoft's ability to have Windows collect and send usage information) because one can examine the product's code. Vendors can generally be considered trustworthy because they depend on trust to survive.

From a developer's point of view, using open source software (as a customer) is beneficial in that it is easy to get access to, examine, and use open source code. A developer also has complete freedom to examine and change any part of the code to satisfy business demands, fix bugs, or port to other systems, either themselves or by hiring someone else to do it. Finally, open source offers the freedom to use (read, build, and change) the code and redistribute it in an open source environment.

Types of Open Source Projects and Business Models, and the Impact of Licensing

It is useful to distinguish between different kinds of open source projects given that they can have different goals, requirements, and possibilities regarding licensing as well as profitability. West and O'Mahoney (2008; tinyurl.com/5zl4uc) distinguish between *sponsored* (i.e., corporate) and *autonomous* (i.e., community-developed) projects. In sponsored projects, one or more corporate entities control the project and employs most of the developers (MySQL was such a project); in community-developed projects, governance and control are shared widely among the community. Some community-developed projects have a non-profit foundation created to support the project; however, these foundations have little authority over their members (O'Mahoney, 2005; tinyurl.com/l5xzbva).

Although there is much interesting discussion and debate around business models as well as their content, focus, and definition, for the purpose of this article we will define a business model simply as the way in which a company delivers value to a set of customers at a profit (Johnson, 2010; tinyurl.com/m9uf6xe). The benefits, or value, of open source described earlier are universal to all open source projects; there are, however, differ-

Introducing "Business Source": The Future of Corporate Open Source Licensing?

Michael "Monty" Widenius and Linus Nyman

ences in approach regarding the means of achieving profitability. Among the most common approaches are *the services model, open core,* and *dual licensing*. The services model is one in which the product is given away for free and income is generated by offering support, services, training, etc. around the product. In open core, part of the content (the "core") is open source, with additional closed source features provided for a fee. Dual licensing means offering a program under two separate licenses, commonly one version under a viral, GPL-style license and another under a commercial, closed source license allowing for proprietary use. Traditionally, the source code for both versions is identical, except for changes in the copyright. (For more information on business models and open source, see Bailetti [2009; timreview.ca/article/226]; Daffara [2009; timreview.ca/article/277]; and Shanker [2012; timreview.ca/article/534]. For an introduction to business models that summarizes popular business model frameworks and proposes a modified framework for technology entrepreneurship, see Muegge [2012; timreview.ca/article/545].)

Finally, it is important to include a brief mention of the importance of licensing, which is a significant factor in open source adoption decisions (Daffara, 2011; timreview.ca/article/416). Finding a license that meets the needs of both corporations as well as the open source community is crucial to the continued well-being of open source software development: being too restrictive will harm community growth, while being too permissive will harm business growth.

Introducing: Business Source

Here, we introduce business source: a new type of license that seeks to address the previously discussed challenges of licensing as well as profitability by using two different licenses with a time delay. The source code is made visible and editable to all from the start; however, for a set amount of time, a pre-defined segment of users have to pay to be allowed to use it. After this initial time period, the license automatically changes to an open source license. To clarify the concept, let us break it down into two phases, examining each individually.

Phase 1: Source Code Available

The software begins under a license that makes the code visible to all. The license gives the user the right to modify and redistribute the code. However, it is not an open source license: the license sets specific requirements for who is allowed to use the program free of charge and who must pay for it. In other words, for the vast majority of users, it will be indistinguishable from an open source program, while a small minority of users will have to pay for it for a limited time. The license used in phase 1 is valid for a set amount of time, and the specific date when the license changes is stamped directly into the source code.

The goal of business source is to facilitate the generation of income without alienating the open source community. Trust is generated through the knowledge that it is only a matter of time before the code is automatically re-licensed under an open source license. Another benefit with business source is that most of the benefits that users and developers expect from open source – and which were described earlier in this article – are open to them: there is no vendor locking, they are in control of the source code, they have the right to free redistribution, etc.

Business source raises three main implementation questions: what timeframe should the developers choose?, what segment should pay for the program?, and how much should the developers charge? These are questions that the developer needs to answer based on their knowledge of their specific industry; however, we will discuss them briefly to offer some guidance on the matter, based on Monty Widenius' experiences with open source in general and the database industry in particular.

What timeframe should developers choose?

With business source, the balance that must be struck here is one of being reasonable to the company on one hand and to the customers and community on the other hand. From the company's point of view, the timeframe needs to be long enough to make money on the existing program while developing improvements. From the customer's and community's point of view, the issue is one of risk management: if the company begins to behave unreasonably, how long will they have to pay for licenses for original code (that they are not using as such anymore)?

A license duration of just one year would prompt many users to just decide to wait for the open version, whereas any duration over five years would, for all intents and purposes, make the program open core. Three years seems a good balance: people will not want to wait too long to be free of a vendor that misbehaves (such as one that stops developing their product), but it is still a

Introducing "Business Source": The Future of Corporate Open Source Licensing?

Michael "Monty" Widenius and Linus Nyman

reasonable timeframe for a developer to know that the program will soon become open source, regardless of any potential unfavourable actions of the company. As noted, this is a suggestion based on the database industry; the length can be decided individually for each project depending on industry (and investor) criteria.

What segment should have to pay?
Given that this article seeks merely to outline the business source approach on a conceptual level, it is impossible to define "who" should have to pay; instead, we will speak to "how many". Again, there is a balance to be struck between generating enough community interest and trust versus generating enough income. A ratio that worked well for MySQL was approximately one per thousand users paying for the software. In general, having between one per one hundred to one per one thousand users pay should be a good range for any product. It is important not to have too many people that have to pay because one wants to ensure that the product gets maximum spread in order to reach all the people that are prepared to pay. Generally, it is a good thing to arrange it so that those that cannot afford to pay or would not be willing to pay do not have to pay! The criteria for defining which segment to charge for the product will depend on the software and industry, but some examples of metrics that could be used are customers who use the product in the cloud or customers with more than X workers in either the entire company or in some specific department.

How much should developers charge?
The price should be low enough to both encourage people to switch from closed source and also to not fork the product. Being somewhere between one tenth to one third of the price of closed source competitors should be reasonable to all. The entrepreneur needs to ensure a sufficient income for both the staff and the entrepreneur to be able to work full-time on the product without having to do consulting or training on the side. Payment should be made easy (e.g., by offering several payment methods, such as PayPal, credit cards, bank transfers, or cheques. Among the ways MySQL initially grew was by accepting cheques and handling credit cards on the website).

Rather than attempting to increase the percentage of paying customers or maximize the money generated from a customer that has already bought a license, we recommend concentrating on increasing the total customer base. (MySQL's attempts to increase the percentage of paying customers were only marginally successful; growth came primarily from increasing total customer volume.) In practice, this means that one license should cover one copy of the product, including all future versions. (However, these guidelines can and should be adapted to fit the developers needs.) The user should have rights to make any changes to the copy they are licensing. Furthermore, the license should also be transferable. Having such a broad license will both discourage people from forking the product and increase its adoption.

It is important to find a proper balance between the time limit and the license price to avoid a situation where a large-enough group decides it easier to fork and wait for the license to change than to pay for the licenses. One should strive to be the leader, with a community that assists in the development of one's product. To achieve this, the license must seem reasonable. Offer something better than the alternative and companies will be more willing to aid in the development of the software.

Phase 2: Open Source

In phase 2, the license automatically changes to an open source license on a pre-defined date, making the code available to all, free of charge. In practice, each file is stamped with a statement of when – on which specific day – the license automatically changes to an open source license. A practical question here is what license to choose. If one wants to make the code freely usable by all, BSD version 2 (which is compatible with the GPL) or Apache are the easiest, though GPL is also an option. (The pros and cons of license choice is a topic for another article; it is a question of how much control one will have over possible forks.)

Decisions about contributor licensing are also up to the company implementing business source. One option, preferred by the Free Software Foundation (fsf.org), is to first receive the code and then license it back to the contributor; however, some consider this a bit difficult to explain. Another option is to accept contributions under either the BSD version 2 or a shared copyright. (For more on license selection and business models, see Daffara [2011; timreview.ca/article/416]; for an open access journal on issues related to open source licensing, see the International Free and Open Source Software Law Review (ifosslr; ifosslr.org); and for a list of open source licenses, see the Open Source Initiative [opensource.org/licenses].)

Introducing "Business Source": The Future of Corporate Open Source Licensing?

Michael "Monty" Widenius and Linus Nyman

Managerial Prescriptions: Who Should Consider Business Source?

Business source is neither designed nor suggested to be the correct license for all projects. A requirement common to all projects considering business source is that, given the time-based license change, the program must continue to evolve to ensure that there are new releases with new end-dates for the automatic license change. Further advice and discussion regarding when business source should be considered is categorized by type of project: closed source, open source, and projects that are still in development. We conclude with a brief discussion for investors.

Closed source projects
Business source is primarily intended for closed source projects and as a better alternative for open core projects (see below for details on open core). In short, business source is ideal for all those closed source projects interested in the idea of contributing open source code, opening their product up to the development potential, and other benefits (covered earlier in this article) that open source offers, while at the same time enabling sufficient income to continue development and growth. Specifically, business source is ideal for:

1. Projects that are considering going open source, or projects that are interested in the benefits of open source, but are concerned with its lessened potential for income.

2. Projects that have already decided to make the switch to open source but have not yet implemented it. Business source is particularly well suited for such a scenario, because they can try a move to business source first and, if it is not satisfactory, take the further step to make the project open source later.

Open source projects
To be able to implement business source, a project must own the code being licensed, must be able (and allowed) to handle the generation of income, and must allow the use of the phase 1 license that is only partially compliant with the Open Source Definition (OSD; opensource.org/osd). In practice, it is the so-called sponsored projects (i.e., corporate projects) for which business source would be possible. To handle the practicalities of an income, a community-developed project would need a company, turning it (for all intents and purposes) into a sponsored project; and, a community-developed project governed by a foundation to guard the openness of the code would not allow the use of the first, only partially OSD compliant, phase of the business source license.

Of the main open source business models in use, business source is mainly relevant to open core projects. We urge all those with an open core project to examine the possibility of switching to business source. Such a move would maintain the potential for income, while improving community image and, thereby, increasing the size of the project and the number of contributions. Programs using a services model are likely to find that community and licensing concerns may make business source difficult or impossible to implement. (It can, however, be considered if additional income is essential for project survival; this is a situation the community may well accept as a reason for a switch). The specific set of requirements under which dual licensing works best (e.g., embedded programs) do not always lend themselves to business source if the dual licensing generates a sufficient income. In summary:

1. Business source can be considered for sponsored projects, but will not be feasible for community-developed projects.

2. Open core projects should consider business source.

3. For at least the vast majority of projects focused on services or dual licensing business models, business source will not be ideal.

Projects in development
Any project that is still in development should consider business source because it will be easier to gain funding and achieve growth with a business source license than with an open source license. (However, license choice naturally depends on the type of project and its goals: a company that aims to remain small can do well with a services approach; a company that seeks strong growth should consider business source.)

Investors
If you are an investor and come across an interesting project (whether open or closed source), consider suggesting business source. As discussed, such a move can offer benefits to both open and closed source programs. (The first author, Monty Widenius, has suggested business source to startups that have approached the investment company Open Ocean Capital

Introducing "Business Source": The Future of Corporate Open Source Licensing?

Michael "Monty" Widenius and Linus Nyman

[openoceancapital.com] with an interesting idea, but that would not generate sufficient income as an open source project. The suggestion has been well received, and development projects that will implement business source are underway.)

Conclusions

Being too restrictive in one's licensing will harm community growth, while being too permissive will harm business growth. The challenge with open source business models is finding one that simultaneously harnesses the power of open source as a development tool and enables a revenue stream that makes continuing product development possible.

Business source, based on Monty Widenius' decades of experience with open source entrepreneurship and licensing, addresses this challenge by implementing a time-based, automatic license change. Initially, the code is made available for everyone to view, but a segment of users must pay to use the product. After a set number of years, the license automatically changes to an open source license, freeing the code for all to use freely. Business source seeks to allow for the best of both worlds: maximizing contributor potential through guaranteeing the openness and freedom of the code (an important concern to would-be contributors), while making it possible to generate income.

The license can be tuned and tweaked to target any segment of one's choosing for the generation of income, while being free to everyone else. As long as the software continues to evolve and delivers value to customers, the developers will maintain a steady income, while (with a delay of a few years) new and improved open source software will continue to be generated.

Monty Widenius has presented the business source idea at conferences and universities in several countries and continents. It has consistently been well received by lawyers, academics, open source practitioners, and entrepreneurs alike.

Acknowledgements

The authors would like to thank S. Keith Moulsdale, Partner at Whiteford, Taylor & Preston LLP, for his kind help in reviewing the example business source license included in this article.

About the Authors

Michael "Monty" Widenius is the founder and original developer of MySQL and MariaDB. He has been an entrepreneur since 1979 and founded MySQL Ab, Monty Program Ab, SkySQL, and Open Ocean capital.

Linus Nyman is a doctoral student at the Hanken School of Economics in Helsinki, Finland. The topic of his PhD is code forking in open source software, and he lectures on corporate strategy and open source software. Other areas of interest include freemium and microtransaction business models in gaming. Linus has a Master's degree in Economics from the Hanken School of Economics. Regarding this article, he would like to note that business source is Monty's idea; Linus merely got involved to help put the idea into article form.

Citation: Widenius, M. and L. Nyman 2013. Introducing "Business Source": The Future of Corporate Open Source Licensing. *Technology Innovation Management Review.* June 2013: X–Y.

Keywords: business source, open source business models, software licensing, open source software development, open core

Introducing "Business Source": The Future of Corporate Open Source Licensing?

Michael "Monty" Widenius and Linus Nyman

Appendix: An Example of a Business Source License

The following is an example of a business source license for a fictional NoSQL product. It should be altered to fit the users' specific requirements. This example was drafted by Monty Widenius based on his considerable experience with dual licensing, and it has been vetted by a lawyer with expertise in software licensing.

XYZ Business Source License

Copyright © 2013, XYZ Corporation

This license ("License") grants rights in specified software code (the "Code") under a business-source-style license that applies one set of terms and conditions (the "Pre-Change Terms") to the Code and all modified Code before a specified date (the "Change Date"), and another set of terms and conditions (the "Post-Change Terms") on and after the Change Date. The Change Date for this license is 01 January 2015.

More about this License can be found at http://company-name/Business_source.

A. Pre-Change Terms: License, before 01 January 2015:

Prior to the Change Date, you have the non-exclusive, worldwide rights under this License to copy, modify, display, use, and redistribute the Code solely under the following conditions:

[Insert business source limitations appropriate to your business here, such as: "The database size used by the Code is less than 1 Gigabyte, and the Code is used in non-commercial contexts where neither you, the user nor any distributor or service provider makes money, directly or indirectly, from using or otherwise exercising your licensed rights in the Code or modified Code".*] [The foregoing limitations are for illustrative purposes only. When designing your business-specific, Pre-Change limitations, carefully consider such things as: i) the differences between source and object code; ii) copyright and patent rights; and iii) the impact on your business of all possible uses of the code, including distribution, the creation and use of derivatives and collective works, and the provision of cloud-based and other services that do not require distribution of the Code.]*

All copies and uses of original and modified Code are also subject to this License. When copying or distributing original or modified Code, you must conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License applies to the original or modified Code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Code.

If your desired use of the Code or modified Code does not meet all of the above requirements, you MUST purchase a separate, commercial license for the Code prior to all conflicting installations or other uses of the Code. You can buy support/licenses from: _____.

Any attempt to use the Code outside the permitted scope of the Pre-Change Terms will automatically terminate your rights under this License to this and all future versions of the Code.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, THE CODE OR ANY SERVICES OR WORK PRODUCT PROVIDED UNDER OR IN CONNECTION WITH WITH THIS LICENSE ARE PROVIDED ON AN "AS IS" BASIS. YOU EXPRESSLY WAIVE ALL WARRANTIES, WHETHER EXPRESS OR IMPLIED, INCLUDING (WITHOUT LIMITATION) WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NON-INFRINGEMENT, SYSTEM INTEGRATION, AND ACCURACY OF INFORMATIONAL CONTENT.

On the Change Date, the Pre-Change Terms shall automatically terminate and shall be replaced with the Post-Change Terms described in Section B, below.

Introducing "Business Source": The Future of Corporate Open Source Licensing?

Michael "Monty" Widenius and Linus Nyman

Appendix: An Example of a Business Source License (continued)

B. Post-Change Terms: License after, and including, 01 January 2015:

On and after the Change Date, the software code is licensed to you pursuant to version 2 or later of the GNU General Public License, as follows:

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; version 2 or later of the License.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA.