



Single-Vendor Open Source: It's Not a Project, It's a Feature!

Dirk Riehle¹, Friedrich-Alexander-Universität Erlangen-Nürnberg

This article examines the outcry over open source relicensing (the so-called "rug pull") by vendors and argues that it is a counterproductive reaction. The relicensing of single-vendor commercial open source software is a watershed event and not necessarily one to spill tears over.

A common, but wrong, belief is that all open source software is born equal. That is not true. Software, including open source software, has an owner, and the owner can determine its fate. If the owner is a community, the community can decide its fate. If the owner is a single vendor, the vendor can decide.

SINGLE-VENDOR VERSUS COMMUNITY OPEN SOURCE

Ownership in this context is mostly determined by who owns the copyright to the code. Community open source software is owned by all the community members who contributed to the software. If the community wants to change the license, pretty much most of the community has to agree. So it rarely happens.

Single-vendor open source software is software owned by a single vendor who develops the software for commercial purposes.¹ The vendor chooses to make the software available under an open source license in addition to the commercial license they typically sell.

The vendor performs 99% of all development because users rarely contribute, and if they do, the vendor requires a copyright transfer to maintain 100% ownership of all product-critical code.

There are a large number of strategic reasons why providing a commercial product in part or fully under an open source license in addition to the commercial license. Some of these have been discussed in a previous article.¹ If the vendor wants to change the license of the open source software going forward, it is easy because they are the single sole owner and can decide alone.



FROM THE EDITOR

Commercial and in particular single-vendor open source firms are companies that provide some or all of their product under an open source license to the world. Increasingly, over recent years, these companies have decided to change the license going forward, often to a nonopen source license. In this article, I explain the logic of such relicensing, from the vendor's perspective, and how affected users can and perhaps should think about it. Happy reading, and stay healthy!—Dirk Riehle

Examples of community open source software projects are PostgreSQL, OpenSearch, and Valkey. Examples of single-vendor open source projects are Yugabyte DB, System Initiative, and Nubus.

The open source community at large is often critical of single-vendor open source. It is not true open source, it argues, because the community of users does not have a say in where the software is headed. These voices would like to extend the open source *software* definition with an open *collaboration* definition, asking that open governance be added to the definition.¹ However, the definition of open source software is very clear and limited to the artifact and the rights to it and does not say anything about collaboration and governance.

With the ownership rights in hand, single-vendor open source firms in recent years have decided to drop the open source license from future releases of the software, either switching to alternative somewhat open licenses (so-called source-available licenses) or to no-free-usage licenses at all. Not surprisingly, the users of the single-vendor open source software as well as the open source community at large were not amused.

WHY HAVE AN OPEN SOURCE STRATEGY?

Most people, when they try to understand single-vendor open source, look at the open source software first. This is the wrong perspective. You need to look at the commercial product first. Then it becomes clear that the open source version of a commercial product is just a feature of that product. It does not have a life of its own; from the vendor's perspective, it is just an appendix to the commercial offering.

Three immediate and obvious feature-like aspects are as follows:

1. The open source software creates trust in the security and quality because customers can look at the code.
2. The open source software removes the need for an escrow contract between vendor and customer.
3. The open source software acts as a promise against price gouging because customers of the commercial product can always switch to the open source version.

I've been interviewing CEOs of single-vendor open source firms for years. It is common practice today that the only thing open source is the source code. Already, build scripts are not available. Binaries are made available only under a commercial license: free of charge to use, sometimes, but not under an open source license. Still, there is value in providing a commercial product under a (free-to-use) open source license, in addition to the (to-pay-for) commercial license and associated services, to both the vendor and the open source users.

Users can get significant value out of the open source software because, if done well, it is simply good software, free of charge to use. As a consequence, users repay the vendor by providing bug reports, word-of-mouth marketing, and documentation, among other things. These and many other benefits often

help the single-vendor open source firm disrupt a market dominated by a closed-source incumbent.

THE ASSUMED SOCIAL CONTRACT

Users who don't or can't or won't distinguish between community open source and single-vendor open source often transfer their assumptions from community open source to single-vendor open source. Assumptions include the following:

1. The company, acting as a benevolent dictator of the open source software, will keep the community's best interests at heart.
2. The company will respond to community needs like new feature requests, honor them, and realize them.
3. The company will not make any changes to the software that go against the wishes of the community.

These and other assumptions, taken together, form an assumed social contract. Evangelists and developer relations personnel of the vendor often support the belief in this contract, either explicitly or by not objecting. However, CEOs generally don't. Or if they do, chances are high they'll eventually be replaced by someone who doesn't. This social contract exists in the minds of some or most users—but not in the mind of the vendor.

THE RELICENSING EVENT OR "RUG PULL"

Against the interests of their open source user community, vendors in recent years decided to stop providing their commercial product under an open source license, foregoing some or all of the benefits of the strategy mentioned previously. This so-called relicensing event applies only to the software going forward; old software versions remain under the open source license. [Figure 1](#)

illustrates how vendors focus their development work over time.

The main official argument by vendors is that they need to stop the competition from using their software against them. Often, they blame the hyperscalers (for example, Azure, AWS, or GCP) for eroding their business using their software. To stop them, they are taking away the open source license that allows the competition to eat their lunch.

Not surprisingly, this relicensing is upsetting to the user community of the vendor's open source software. With many cases known today, the complaints are usually the same; the vendor made a promise by engaging in the social contract outlined previously, decided not to honor the contract, and effectively lied about their intentions. Mailing lists are scrutinized for any statement by a past CEO that might indicate a "forever open source" promise.

It is all for naught. Vendors don't lightheartedly make this decision; they know full well about the expected community push-back and bad marketing. Still, they make the decision.

I find it somewhat hypocritical to assume that a commercial vendor, driven by a for-profit motive, may put a community's interests over its own. My recommendation to users of single-vendor open source is to always be aware that a relicensing may happen and prepare for it.² Preparation, among other options, includes being ready to migrate to alternative software or to simply pay the vendor for the commercial version.

THE COMMUNITY FORK IS A GREAT OUTCOME

When relicensing occurs, multiple different reactions may happen. In most cases, users will either migrate away or pay the vendor. In some (the most high-profile cases), however, the community may decide to fork and keep developing the last good open source version of the software.

Three well-known examples are the community forks of Elasticsearch to OpenSearch, Terraform to OpenTofu, and Redis to Valkey. These three originally single-vendor open source projects were forked by a group of high-profile users, predominantly the hyperscalers, then put under an open source foundation, and are now being developed as community open source software.

How is it not a fabulous outcome if the community steps up to keep developing the open source software? Venture capital funded the development of great software, made available under an open source license and in use by many. Eventually, a community of developers took over ensuring the sustainability of the software and its contribution to common benefit. Venture capital is typically interested only in its own profit, but in this scenario, it contributed directly to the creation of significant common value in the form of widely used quality open source software.

WHY COMPLAINING IS COUNTERPRODUCTIVE

The complaints of the open source community about relicensing events are bad

marketing and hurt the single-vendor open source firms. As a consequence, many vendors moved to adopt a new type of license, the so-called source-available licenses, from the beginning. Source-available licenses are effectively noncompete licenses; they allow the free use of the software, but not for the purposes of competing with the vendor.

Source-available licenses typically do not allow for the community fork that could happen in response to a single-vendor open source project behaving in ways disliked by the open source user community. The user community is locked in, and in the case of disagreement with the vendor, can only opt to abandon the software. Existing single-vendor open source firms have opted to relicense to source-available licenses, and many new vendors that would have been single-vendor open source firms are starting out with a source-available license directly, not providing the potential open source benefit of forking to its community.

The move from open source licenses to source-available licenses may have been brought about by creative vendors trying to curtail the competition, but the complaints by the open source community certainly added to it. As a consequence, we see less venture capital being invested in single-vendor open source, withholding the benefits and contribution to the commons of the original software or the later community fork discussed previously. If the open source community at large would like to see more venture capital funding open source development, it should applaud the single-vendor open source model, support it against the source-available model, and accept that a relicensing may happen at some point in time.

Relicensing is a coming-of-age event, in which the single vendor offers to hand over the software to a community for long-term sustainability. It is up to the community to react, recognize the value or not, and step up.

AN ALTERNATIVE ROUTE, BUT IT IS NOT FOR ALL

Relicensing is not a foregone conclusion. For all that I can say, the primary

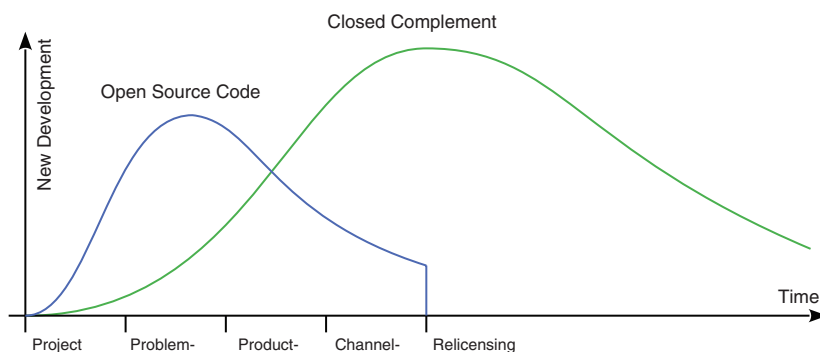


FIGURE 1. Open source code and closed complement over time.

predictor is the amount of venture capital driving the single-vendor open source firm. Venture capital, seeking an exit, is hastening the growth of the single-vendor open source firm. This includes tightening the screws to push users, then considered freeloaders, to become paying customers.

Raising venture capital is not a foregone conclusion; it is typically a choice of the founding CEO. In my research work, most of the companies that I interviewed and that raised venture capital see relicensing in their future or started out with a source-available license right away. They are often on a high-growth trajectory, fueled by the venture capital they raised.

In contrast to these, I also found many open source companies that dominate the software their business is built on but that raised no venture capital. Examples are BigBlueButton and Blindside Networks, the company behind it, Samba and SerNet, and XWiki by the company of the same name. These are established and profitable companies, but they

provide nothing of the returns that the venture-capital-funded companies provide to the founders and their investors. Also, it took them much longer to build the software than it would have taken venture-capital-funded startups.

It is unlikely that these companies will ever relicense, and despite their dominant position, it may not even be possible. They have found a mostly comfortable arrangement with their communities. Still, they run the risk of being overtaken and displaced by venture-capital-funded competitors.

I don't have an answer to the question of whether venture-capital-funded single-vendor open source firms must relicense at some point in time. It should also be possible to keep going without relicensing. However, as this article showed, if the vendor decides to relicense, it should be viewed as a positive event in the life of the underlying software—it is given the chance to mature from being the appendix of a commercial product to becoming a

true community open source project of its own .

REFERENCES

1. D. Riehle, "Single-vendor open source firms," *Computer*, vol. 53, no. 4, pp. 68–72, 2020, doi: [10.1109/MC.2020.2969672](https://doi.org/10.1109/MC.2020.2969672).
2. D. Riehle, "The Innovations of open source," *Computer*, vol. 52, no. 4, pp. 59–63, 2019, doi: [10.1109/MC.2019.2898163](https://doi.org/10.1109/MC.2019.2898163).
3. D. Riehle, "How to think about a dependency on commercial open-source software," *dirkriehle.com*, Sep. 24, 2022. [Online]. Available: <https://dirkriehle.com/2022/09/24/how-to-think-about-a-dependency-on-commercial-open-source-software/>

DIRK RIEHLE is the professor for open source software at Friedrich-Alexander-Universität Erlangen-Nürnberg, 91058 Erlangen, Germany. Contact him at dirk@riehle.org.



www.computer.org/cga

IEEE Computer Graphics and Applications bridges the theory and practice of computer graphics. Subscribe to CG&A and

- stay current on the latest tools and applications and gain invaluable practical and research knowledge,
- discover cutting-edge applications and learn more about the latest techniques, and
- benefit from CG&A's active and connected editorial board.

