EDITOR **DIRK RIEHLE**
Friedrich Alexander-University of Erlangen Nürnberg;
dirk.riehle@fau.de

# The Open Source Distributor Business Model

**Dirk Riehle,** Friedrich Alexander-University of Erlangen Nürnberg

*This article defines and discusses one particular commercial open source business model, called the (open source) distributor model. It can attract significant venture capital, thereby contributing to the long-term sustainability of open source.*

An open source distribution is a software that combines a typically large set of open source components into one larger open source software project (community distribution) or software product (commercial distribution). A distribution is typically formed around a core piece of software that is enhanced and made practical by the added components.

A Linux distribution, for example, has the Linux kernel as its core to which it adds various other components like file systems or graphics drivers to form a viable operating system. Prime examples are the Debian, SUSE, or Red Hat Linux distributions. There are also smaller domain-focused Linux distributions such as Univention Corporate Server, a Linux distribution for the public sector.

The core focus of a distribution, whether by a community or company, is on making the components in the set work together well so that users can use the whole with as little problems as possible. Distributions typically emerge when component integration complexity is too much for users to handle themselves.

The general public is mostly aware of Linux distributions. However, there are many other distributions. Table 1 shows some examples.

## TYPES OF BUSINESS MODELS

A business model describes how a company operates and achieves its goals. Open source itself may not be a business model, but it can be an important strategy to help a company reach those goals. While each firm has its own distinct

*Digital Object Identifier 10.1109/MC.2021.3112318*
*Date of current version: 17 November 2021*

### FROM THE EDITOR

In this month's column, we take a short pause from our current theme of open source communities to let authors catch up on promised articles. This month's article is on a rarely discussed yet prominent open source–based business model, the distributor model. It is part of my side theme on how and why open source has become sustainable and such an important force, both for company profit and good in the world. With that, happy hacking and be safe and healthy! — *Dirk Riehle*

business model, there are naturally distinguishable types of business models.[1–3]

In our research, we have found three distinct coarse-grain types of open source business models, based on their value proposition and the intellectual property that supports it.[4] These three models are

1. open source service and support firms
2. open source software distributors
3. single-vendor open source firms.

Service and support firms typically do not own specific intellectual property but rather service a small number of existing community open source projects. Single-vendor open source firms provide a product that they typically develop themselves and of which they own the key intellectual property; unlike traditional software vendors, however, they make some or all of the software available under an open source license, in addition to a commercial license and services.[5]

## THE OPEN SOURCE DISTRIBUTOR MODEL

Open source distributor firms are software vendors that build a cohesive product from existing open source components that they typically don't own. The corresponding core intellectual property of open source distributors therefore is not the software itself but rather centers on component integration and provision. It includes build systems and processes, compatibility matrices, configuration databases, and test suites.

**TABLE 1.** Examples of community distributions, commercial distributions, and complex products (a commercial distribution with significant amounts of proprietary software).

| | Growth stage | Mature stage |
|---|---|---|
| Complex products | » Kafka: **Confluent Platform (Confluent)**<br>» Lucene/Solr: **Elasticsearch (Elastic)** | » Hadoop: **Cloudera Distribution (Cloudera)**<br>» Drupal: **Acquia Lightning (Acquia)** |
| Commercial distributions | » Kubernetes: **OpenShift (Red Hat), Mesosphere Kubernetes Engine (D2IQ), Kubermatic Kubernetes Platform (Kubermatic)**<br>» OpenStack: **VMware Integrated OpenStack (VMware), Mirantis Cloud Platform (Mirantis), Fusionsphere OpenStack (Huawei)** | » Linux: **Red Hat Enterprise Linux (RHEL), SUSE Linux Enterprise Server (SUSE), Univention Corporate Server (Univention)**<br>» Python: **ActivePython (ActiveState), Anaconda (Continuum Analytics)** |
| Community distributions | » ROS: **ROS** | » TeX: **TeX Live, MacTeX, MiKTeX**<br>» Linux: **Debian, Fedora, OpenSUSE**<br>» OpenJDK: **Amazon Corretto**<br>» Python: **CPython, WinPython** |

### Value proposition

Venture capital funding only flows if a firm can believably promise significant returns to its investor's money, and open source distributors achieve this by promising many of the same revenue streams that classic software vendors promise. A distributor firm's value proposition consists of but is not limited to the following:

1. services such as hotline support and remote or on-site servicing
2. access to software updates, including new features and bug fixes
3. guarantees such as warranties, indemnification, and certification
4. commercial license for proprietary tools and trademark use
5. operational services such as hosting the software for customers
6. complementary materials for documentation, training, and so on
7. access to self-help services such as forums, chat bots, and so on
8. reduced lock-in, if the software is all open source.

None of this should be surprising to practitioners; with the exception of the reduced lock-in through open source, this value proposition is common to traditional vendors and distributors alike.

Distributors can often sell a commercial license, even if the product consists of open source software. This is possible because, next to source code, other forms of intellectual property are being integrated. The most noteworthy intellectual properties are trademarks that the distributor owns and that are being displayed in various places to inspire user confidence in the

quality of the software. Other intellectual properties can be scripts and configuration files that tie the system together and make it function well and that are not necessarily provided under an open source license.

Such mixing of proprietary intellectual property with open source would probably allow a distributor to prevent the use of the distribution by nonpaying users. In practice, this rarely happens. For example, until recently, the CentOS project, an independent community project, provided the most recent Red Hat Enterprise Linux (RHEL) release, stripped of the Red Hat trademarks, to the world for free. The CentOS release was often available within a day after the corresponding RHEL release.

A main difference between distributors and traditional software vendors is that distributors generally forego an initial license fee and jump straight to a software subscription. A software subscription is a bundle of services and rights, as described before, provided to customers for a defined time period, for example, a year. Traditionally, this was called and managed as maintenance.

### Business functions

Distributor firms do not own the source code of most of what they are integrating into a viable product. Rather, they use original open source components, build them from source, and integrate them with many others, so that taken together they become the intended product. There are three main aspects of the distributor model, which are different from traditional product vendors and make this model unique and successful:

1. A distributor participates in the development of the open source software components from which the product is built. They do so, however, not exclusively but rather together with other companies. This can easily be the largest engineering cost of the distributor, yet it does not provide a unique competitive advantage to the distributor.
2. The actual product, the distribution, is not a set of components but rather the well-integrated software built from components, complemented with various services. From this observation follows a necessarily capability-based competitive advantage as well as the unique intellectual property that a distributor typically does not open source:
   - build processes for building the product from its components
   - compatibility matrices and configuration data
   - knowledge databases for support
   - tests and test suites.
3. The distributor almost always provides a free version of the distribution (without any of the commercial services listed earlier) to drive adoption by nonpaying users.

The first two aspects are unique to the product development process of distributors and the competitive advantage they can build. A well-working free version of the distribution not only creates goodwill, but also it helps the distributor grow a large community of nonpaying users. Unlike trials or other restricted forms of software, open source licenses permit users to use the software as long as they want to, for whatever purpose. As a consequence, as long as the distribution fulfills the need, users will happily keep using it.

A large community of nonpaying users helps the distributor have more efficient and effective business functions; it improves sales and marketing, product management and engineering, and support. It comes with a cost, though: the added need for scalable community management. We will discuss these business functions next.

A software subscription is a bundle of services and rights, as described before, provided to customers for a defined time period, for example, a year.

**Marketing.** The marketing function and processes of most commercial open source distributors include those of traditional vendors but also go beyond them. Following traditional patterns, distributors use the usual channels, for example, trade magazines, conferences, and mass mailings, to generate leads for sales.

A well-working free version of the distribution, however, can tap into the power of word-of-mouth marketing using a much larger community of users than a traditional vendor can do. High-quality software for free is a great value proposition. Happy users like to talk about the software that makes their work life easier. Happy users also make good reference material to the extent that they are willing to talk about their use of the software.

It is important for a distributor to be visibly involved with some or all of the components of the distribution. Such involvement, for example, by employing key developers of the core components of the distribution, inspires trust among potential customers that this distributor will be able to resolve any problems that might arise with the software.

**Sales.** The sales function and process of most commercial open source distributions include those of traditional vendors but also go beyond them. As is traditionally done, marketing primes

the sales funnel and salespeople pick up any leads and try to sell the product.

However, a well-working free version of the distribution can have a significant impact on the sales process. From a sales perspective, having an installed base of nonpaying users is not a problem but rather an opportunity. A distributor may want to track who downloads the free version and gather their email addresses. Email addresses at corporations are a good indicator of a potential customer. Multiple email addresses of different users at the same corporation are a clear indicator that a sales call may be fruitful. This way, a free version allows a sales organization to prioritize where to spend their efforts.

The actual sales process also becomes more effective. The initial users of the free version are often line-of-business (LoB) users who did not want to go through a purchasing process with their IT department. With multiple LoB users in one company, the IT department may want to rein in the use of the software and purchase it centrally. In a comparative evaluation of products, the distributor has a leg up on its competitors. Its product is already in use and has champions inside the buying organization, while its competitors do not. Asking rhetorically: What would you rather buy: a product from a vendor that you have yet to test and evaluate, or a product already in use at your organization where you can ask actual users about it?

**Product management.** A large user base, including nonpaying users, is a great resource for product managers to draw on for product feature discovery. Product managers can learn from users by tracking problems and requests in forums. They can make an issue tracker publicly available so that users can report on problems, and they can create user polls to prioritize upcoming features right, among other things. Due to the large user base, this takes place on a level of scale that traditional vendors at the same stage cannot meet.

The product management of distributions has its challenges, though. As mentioned, developers of the company are typically involved in the development of the underlying open source components. Product managers need to watch out that such developer engagement does not have negative effects such as early revealing of product strategy or intellectual property that should not have got out. Developers are often not aware of the side effects of what they are doing and need to work with product and engineering management to avoid such effects. Employing developers to work on the open source components is often also the main way to get features implemented that customers want but that aren't available yet.

> Unlike trials or other restricted forms of software, open source licenses permit users to use the software as long as they want to, for whatever purpose.

**Engineering management.** A large user base finds problems faster than a small user base. A user might not only file a bug report; they might provide the patch that fixes the problem as well. This helps mature the components of a distribution and thereby the overall distribution faster.

Engineering managers, on the one hand, have to manage those software developers who work on the open source components of relevance to the distribution, and, on the other hand, they have to manage those developers who create the actual distribution. Usually, these tasks fall to different engineering managers and different developers. Those developers working on open source also often serve as a conduit for contributions from other developers within the company who cannot or don't want to be public about their contribution.

Creating the actual product, the distribution, involves significant integration work. The engineering focus is therefore on the aforementioned capabilities of build processes as well as the intellectual property of configuration data, compatibility matrices, test suites, and so on. If certification for specific hardware and software is important, the corresponding processes and documentation need to be considered as well. All of this needs to be performed at a nontrivial scale.

**Product support.** Nonpaying users of a free version typically understand that open source software does not come with a right to support. As a consequence, and for many in the spirit of open source, users are willing to help each other. If the distributor provides appropriate tools like forums and wikis, users may become active to create documentation and self-help materials. Product support can benefit from understanding user problems and utilizing the materials users develop.

**Community management.** As explained, most of the benefits of a free version accrue due to a large but nonpaying user base. Creating this user base comes at a cost, though: the company actively needs to manage this community. Community management requires labor and budget for travel and events. Community managers need to engage with the community, and should provide and operate, for example, a website, forums and wikis, and a software forge. Often, they organize one or more user conferences.

An important aspect of community management is continuity. Community managers cannot be exchanged at random because users and the user community often build relationships

with specific community managers rather than the role. The less turnover, the better for the distributor because established community managers can use their relationships for the distributor's benefit more easily.

Most of these costs are variable that scale with the number of users. Given that these are nonpaying users and that the hope is that its size grows as quickly and to as large as possible, the primary efficiency consideration of community management is how to scale with as little effort as possible.

Community managers achieve this using various best practices. Mostly, they try to establish a community (of nonpaying users) that helps itself and in which company resources are only the helper of last resort. If, for example, a new nonpaying user asks a question in a forum, a community manager will not answer directly. They will first wait for another user to answer. If nothing happens, they will nudge existing users to help the newcomer, and only if that doesn't work, they may decide to provide an answer themselves.

### Intellectual property

Contrary to popular belief, distributors can own significant intellectual property that they don't necessarily make available under an open license. A distributor, like any other commercial company, both creates, uses, and protects its trademarks. The main use of trademarks corresponds with their raison d'être: to identify a product and company and inspire trust in an assumed level of quality associated with and represented by the trademark. Distributors do not grant broad usage rights to nonpaying users if they grant usage rights at all.

A distributor may also own software patents. However, they are either licensed openly, typically by way of open source licenses to the code in which the patents have been realized, or solely for defensive purposes, for example by way of patent networks whose goal it is to protect the network members from lawsuits over other party's patents.

A distributor may own various copyrightable materials, most notably, copyright to source code, configuration data, and knowledge databases. The source code is usually licensed in accordance with the project being contributed to. Sometimes, distributors hold back particular extensions or tools as proprietary software that they make available only in binary form and through a paid subscription.

The installer (and later, the update service) plays a particularly important role because in this process the distributor's knowledge on how to create a well-working system from disparate components is being applied. The source code of the installer is typically not as important as the configuration and rule data that adjust the configuration of various components as the user is making choices during system installation. It is this type of intellectual property at scale, separate from the underlying open source code, that is not easy to create and evolve and that allows a distributor to maintain a strategic advantage over potential competitors seeking to enter the market.

The distributor business model, while not entirely new, is nevertheless a model that has reached significance mostly because of the size and complexity of the open source world. Some of the world's most important software companies are built on this model, providing significant returns on investments to entrepreneurs and investors. Therefore, this model is making an important contribution to the long-term sustainability of open source software and the good it does for the world.

> Employing developers to work on the open source components is often also the main way to get features implemented that customers want but that aren't available yet.


### ACKNOWLEDGMENTS

I would like to thank Ann Barcomb, Michael Dorner, Matthias Eckermann, Peter Ganten, Andreas Jaeger, Nikolay Harutyunyan, and Markus Rex for feedback on this article.


### REFERENCES


1. E. Capra and A. I. Wasserman, "A framework for evaluating managerial styles in open source projects," in *Proc. IFIP Int. Conf. Open Source Syst.*, 2008, pp. 1–14.
2. C. Daffara, "Business models in FLOSS-based companies," in *Proc. Workshop Presentation 3rd Conf. Open Source Systems (OSS 2007)*, 2007.
3. P. J. Ågerfalk and B. Fitzgerald, "Outsourcing to an unknown workforce: Exploring opensourcing as a global sourcing strategy," *MIS Quart.*, pp. 385–409, 2008.
4. D. Riehle, "The innovations of open source," *Computer*, vol. 52, no. 4, pp. 59–63, 2019. doi: 10.1109/MC.2019.2898163.
5. D. Riehle, "Single-vendor open source firms," *Computer*, vol. 53, no. 4, pp. 68–72, 2020. doi: 10.1109/MC.2020.2969672.



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