Fraunhofer
IML

Whitepaper

Open-Source Strategies for Companies – Insights and Guidance

Authors
Carina Culotta (Fraunhofer IML)
Estelle Duparc (TU Dortmund)

Project Team DB Schenker
Simjees Abraham
Maik Schmidt
Tilo Wiedera

Project Team Fraunhofer IML
Carina Culotta
Simon Lechtenberg

Project Team Fraunhofer ISST
Philipp Hagenhoff
Anna-Maria Schleimer

Content

1. Introduction: Why Open Source?

In today's digital economy, open-source software (hereafter OSS) plays an essential part. More than 90 % of the current software solutions are linked to open-source solutions by embedding, supplementing, or transforming them (Harutyunyan 2020). Thus, OSS has become a central building block of technological progress and is crucial for innovative and digital business models. In the platform industry, OSS is used to promote openness and to fuel the development of large ecosystems. Therefore, OSS can act as an enabler for far-reaching network effects and fuel participation from outside. Key technologies, such as cloud computing, artificial intelligence, and the Internet of Things (hereafter IoT) are also built on OSS. Clouds use open-source containerization technologies such as Kubernetes and Docker, monitoring tools such as Grafana and Instana, event brokers such as Kafka, and many other tools that enable building robust IT. But also in the consumer industry, open-source applications, such as Mozilla Firefox, OpenOffice, or LibreOffice are widely used (Schmidt et al. 2022).

Current surveys from research and industry show the increasing importance of OSS outside the digital industry to drive digitalization in traditional industries, such as the logistics or manufacturing sector (Blind et al. 2021; Gentermann and Termer 2019). More than 60 million contributors on GitHub, a platform for managing software development projects, underline the increasing importance of OSS. The popularity of OSS can be explained by the accompanying potentials that come along with the concept: OSS can act as a driver for setting new standards as it fuels the software's diffusion. It also increases the development efficiency as the modularity of OSS eases the reuse of source code for other development projects. The open development process enables the participation of external developers so that the software quality increases by external peer review and innovative ideas from outside can be incorporated into the OSS. Companies from traditional industries, such as SMEs, can use open source as strategic tool to increase their attractiveness to IT personnel and to enable cross-company collaboration. Thus, open source offers the possibility to gather and combine IT resources efficiently to address the ongoing lack of (IT) labor force in traditional industries.

Even though the benefits of open source have been recognized by traditional companies, many of them are hesitant to actively contribute or provide OSS due to the lack of an adequate strategy and a lack of business model skills (Gentermann and Termer 2019). Unlike traditional business models, revenue cannot be generated directly from the actual OSS as license fees do not apply to the software. Planning an open-source business model is crucial as companies must balance the optimal degree between openness and closedness. For example, business-differentiating intellectual property could be given away to competitors, if open sourcing essential product parts. On the other hand, companies could lose the advantage of open source by providing a mostly proprietary offering or neglecting community building. Therefore, each company needs to evaluate its resources and knowledge before choosing a suitable business model.

Closely linked to the idea of business models is the overall strategy a company can pursue. Besides creating a specific open-source business model, the usage, contribution, or provision of OSS can have a tremendous strategic impact and benefits for companies along various dimensions. Especially highly competitive but traditional industries such as logistics and engineering can benefit from OSS. Industries that are hallmarked by a scattered nature of various processes, inconsistent standards, and different customer requirements due to the individuality of the products and processes can use OSS to allow easy integration and connectivity of systems. Thereby they engage in a de-facto standard setting which helps to harmonize processes and thus creates benefits for customers and service providers alike.

The underlying Whitepaper provides a practical overview for companies that wish to build their open-source strategies and summarizes the main strategic advantages of OSS along with the three categories of "technology", "organization" and "environment". In addition, the Whitepaper provides a compact guideline regarding important key questions that companies should consider when implementing open source in the sense of using, contributing, or providing OSS.

In specific, the Whitepaper answers the following questions:

- What is OSS?
- What is the difference between using open source, contributing to open-source projects, and providing own open-source code?
- Which strategic impact can usage, contribution, and provision of OSS have from a technology, organization, and environmental perspective?
- What are open-source business models?
- What are practical guidelines for using, contributing, and providing OSS?

2. Terms and Definitions: Open-Source Software

Open Source refers to a type of software whose source code is made publicly available and that can be modified and used by external parties. The current understanding of open source can be explained by similar concepts and its historical development. The intermediate milestone in this process is, among others, the establishment of the non-profit "Free Software Foundation" by Richard Stallman in 1985 (Stallman 1999). From the very beginning, the Free Software Foundation popularized the basic idea of "Free Software" in which the user should have the opportunity to use, study, modify, improve, and distribute the software code. Until today, the free software foundation aims to maintain open-source projects and to support the resulting collective learning process as well as the exchange of new knowledge (Free Software Foundation 2020).

As the term "Free Software" has often been misinterpreted and associated with the term "free of charge", the terminology "Open Source" was introduced by the "Open Source Initiative" in February 1998 (Raymond and Perens 2018). The Open Source Initiative's main goal was to spread the open-source code's core concept and to clarify the misunderstandings arising with the free software terminology (Rajala et al. 2006) The Open Source Initiative defines ten characteristics of open source that read as follows (Open Source Initiative 2007)

1. **Free Redistribution:** The license shall not restrict any party from selling or giving away the software as a component of an aggregate software distribution containing programs from several different sources. The license shall not require a royalty or other fee for such sale.

2. **Source Code:** The program must include source code, and must allow distribution in source code as well as compiled form.

3. **Derived Work:** The license must allow modifications and derived works, and must permit to be distributed under the same terms as the license of the original software.

4. **Integrity of the Author's Source Code:** The license may restrict source code from being distributed in modified form unless the license explicitly guarantees such distribution. The license must explicitly permit distribution of software built from modified source code.

5. **No Discrimination Against Persons or Groups:** The license must not discriminate against any person or group of persons.

6. **No Discrimination Against Fields of Endeavor:** The license must not restrict anyone from making use of the program in a specific field of endeavor. For example, it may not restrict the program from being used in a business, or from being used for genetic research.

7. **Distribution of License:** The rights attached to the program must apply to all to whom the program is redistributed without the need for execution of an additional license by those parties.

8. **License Must Not Be Specific to a Product:** The rights attached to the program must not depend on the program's being part of a particular software distribution.

9. **License Must not Restrict Other Software:** The license must not place restrictions on other software that is distributed along with the licensed software.

10. **License Must Be Technology-Neutral:** No provision of the license may be predicated on any individual technology or style of interface.

The ten characteristics show that the open-source terminology matches the main idea of free software by illustrating concepts and benefits of source code sharing and software collaboration without hindering the commercial use of OSS as well (Fitzgerald 2006). There are three approaches on how to get involved with open source: usage, contribution, and provision.

3. Approaches on How to Get involved with Open-Source Software

Companies can approach OSS in different ways. They can either simply use OSS, they can actively contribute to OSS projects, or they can provide their code and put it under an OSS license.

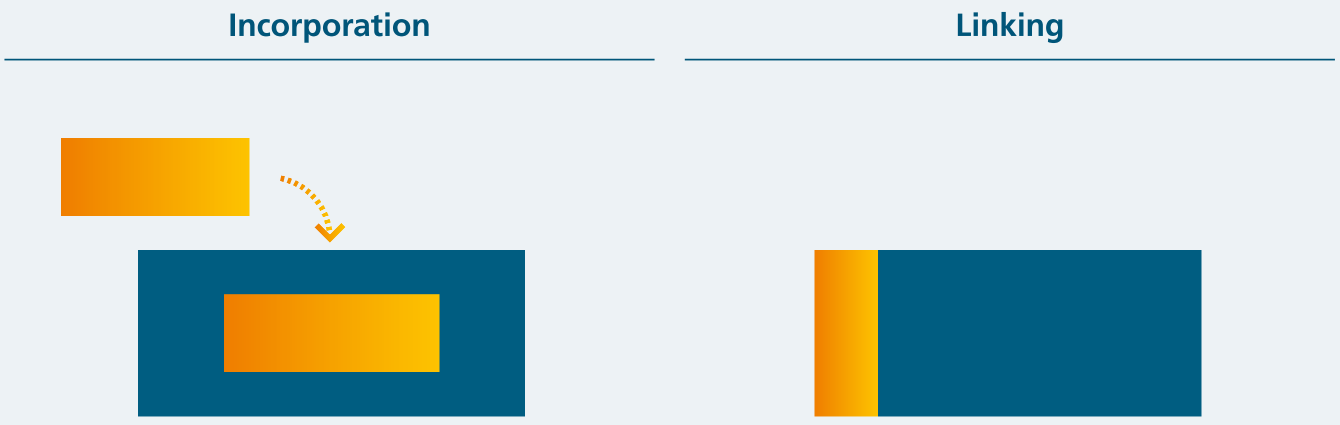
3.1 Open-Source Software Usage

The first approach focuses on the (passive) usage of the OSS. The user takes advantage of the software's specific benefits and functions without actively involving much in the community. The primary objective is the in-house usage, for example, to improve internal processes and software. The selection of OSS is similar to the one of proprietary software, as internal decision criteria, such as technical capabilities, budget, and other restrictions are used to evaluate the potential software. In addition to established decision criteria, companies should consider the OSS community as they are the driving instance of the software. The OSS is often perceived as a commodity service as it is, in contrary to proprietary software, free of license fees. However, some companies have professionalized the commercialization of OSS by selling complementary products or services. Therefore, OSS does not come "for free" if the user needs support or hosting services to implement and use the OSS. The advantage of the OSS usage approach is that users do not need profound IT skills to benefit from the OSS advantages, such as lower costs for ready-to-use software, security, and reliability as the community can provide security upgrades and patches. Therefore, this approach is suitable for users new to the open-source topic as they are similar to consumers without an active role in the community. On the other hand, the passive user has less influence on the OSS development process and its changes. This results in accompanying risks as the community is not deadline-driven and bugs may not be fixed timely. Non-technical users may face a high entry barrier as traditional OSS was developed by and for developers (AlMarzouq et al. 2005). However commercial OSS providers have adapted their product offering to non-technical users.
OSS can be used in different ways. Generally, open-source components can be:

- Merged into other software components (Incorporation),
- Connected with other software components (Linking),
- Modified (Modification) or
- Transformed (Translation)

Incorporation refers to the possibility of integrating parts of OSS into the company's software. This includes, for example, the insertion or integration of source code into the own program code. In the Linking, a developer
connects an open-source component with his own component. This link can be static or dynamic (static/dynamic linking). Also, by encapsulation (packaging), a connection can be made. Modification of OSS components involves adapting or changing the original source code of an open-source component. This also includes the insertion and removal of source code.
Such adaptations are made to optimize the components or to remove errors. Finally, open-source components may be converted (translated). This covers for example the translation of code into other programming languages and the compiling into a binary file. In the case of the various possibilities for OSS usage, organizations need to comply with the respective license obligations.

After Schmidt et al. (2022)

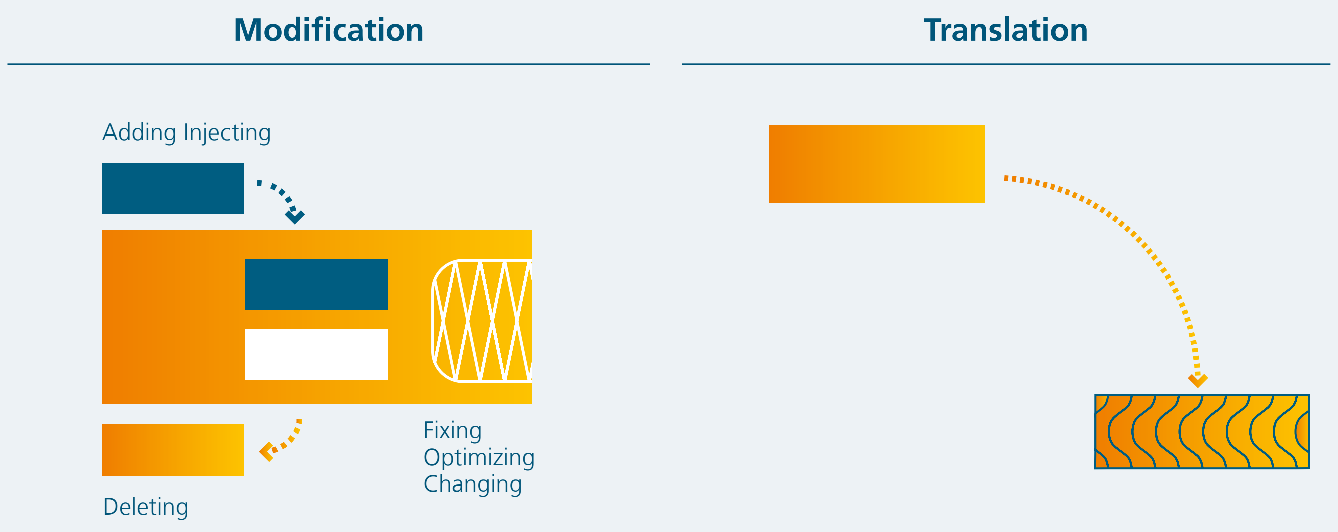


After Forschungsbeirat der Plattform Industrie 4.0/acatech (2022)

3.2 Open-Source Software Contribution

The second approach focuses not only on adopting the OSS but also on involving into the community and the accompanying benefits. For example, companies that actively participate can benefit from a broader and richer knowledge transfer and be part of development processes. The advantage over the previous approach is that the company becomes part of the community and can potentially exert some influence on the community by giving suggestions on future software improvements or key features. Thus, they can use the community as external resources to customize the software to internal needs. Instead of autonomous in-house production, the community becomes part of the development processes. In addition, companies that share their software code can benefit from the community's feedback and external developer resources. The approach is best suited for companies that have the resources and abilities, e.g., knowledge of technical development or license agreements, to participate in the community. In addition, the company needs to consider the higher time effort that comes with active community involvement (AlMarzouq et al. 2005).

Contribution thereby means, that companies actively contribute source code to existing projects. Ideally, the projects have a direct impact and benefit for the company. Software developers are communicating with other developers and contributors within the projects and engage in a joint co-developing process. Those projects are mostly publicly available on platforms such as GitHub.
Prominent projects are often sponsored and supported by OSS foundations such as the Linux Foundation or the Apache Foundation. Sometimes also big companies such as Google or Microsoft do not only publish their projects but also support individual projects if they can benefit from the current developments.



After Forschungsbeirat der Plattform Industrie 4.0/acatech (2022)

3.3 Open-Source Software Provision

The last approach describes the active provision of OSS by initiating its own open-source projects. The company can choose to either release OSS or initiate a community to develop a new solution. As an initiating instance, the company can control the license under which the source code is released, which influences the further direction of the open-source project as copyleft licenses are more restrictive than permissive ones but guarantee derivate OSS (AlMarzouq et al. 2005). In addition to the other potentials, providing OSS can spur network effects, such as in the platform economy, create new business models, enable open innovation, and accelerate the diffusion of new technology (West 2003; Okoli and Nguyen 2015). As the contributors of open-source communities tend to be customers at the same time, the company can tighten its customer relationships and leverage the customer as a resource through this approach (Hippel and Krogh 2003). However, companies need to be aware that the benefits of this approach will only unfold over a longer period as it takes time to grow a healthy and sustainable community. Therefore, companies need to actively develop and support their community to increase its attractiveness to external developers (AlMarzouq et al. 2005).

Furthermore, companies need to consider carefully, which product part they want to provide open source and how to generate revenue to avoid exploitation from outside. In the case of Elastic, the company had to relicense its open-source search engine after an opponent had launched its version based on Elastic's OSS (Banon 2021). Lastly, large companies that want to follow the provision approach should consider involving lawyers to help them choose the best open-source license to fit their requirements and objectives (AlMarzouq et al. 2005). Due to its complexity, the method is suitable for companies that are familiar with the previously mentioned open-source approaches and that want to improve their competitive position through network effects or for companies that want to enable open innovation (AlMarzouq et al. 2005).

Companies deciding that open-source provision is a beneficial business model or should be part of their business strategy can provide their open-source platforms and development environments. However, it is most beneficial to use existing and well-established platforms such as GitHub or GitLab. At the same time,
it can also be beneficial to incorporate and "donate" the project to an existing foundation that takes care of community and governance processes. In the end, this is up to the respective company and the goals that should be fulfilled.

4. Strategic Goals of Open-Source Usage, Contribution, and Provision

Companies that decide to either use, contribute, or provide OSS mostly do so, because they pursue different strategic goals within their company and respective enterprise networks. Thereby the strategic impacts differ between usage, contribution, or own provision. Providing own source code under an OSS license has the most far-reaching impact on a company and helps pursuing mostly ecosystems and market-related strategic goals. However, the contribution and usage of OSS also support various strategic decisions within a company and does not only allow for cost savings but also the co-creation of important software developments.

In the following the different impacts of using, contributing, or providing OSS are elaborated among the three dimensions of "technology", "organization" and "environment" following the technology diffusion model by (Tornatzky and Fleischer 1990). *These strategic insights are retrieved from a joint project between Fraunhofer IML, Fraunhofer ISST, and DB Schenker and are shared in the spirit of open source with the interested community.* The main advantages and benefits of OSS are summarized in Table 1.
The detailed strategic impact of the various advantages for a company is explained in the following sub-sections.

Table 1: Main strategic advantages of OSS

Technology		
Quality	Interoperability	
▪ Security ▪ Code Quality ▪ Stability and Longevity	▪ Flexibility ▪ Independence	
Organization		
Human Resources	Culture	Business Development
▪ New Human Resources and Talent Attraction	▪ Workflow ▪ Transparency ▪ Open Innovation ▪ Diversity	▪ Reputation ▪ New Business Models
Environment		
Digital Ecosystems	Market	
▪ Establishment of Digital Platforms and Ecosystems	▪ De-Facto Standards	

4.1 Technology

From a technological perspective, quality and interoperability are the main intentions to either use OSS or contribute respectively provide OSS.

From a quality point of view, OSS is meant to be secure – in the sense that code was developed openly and the developer community could find sources of error and openly address these. The original provider of the source code can implement these modifications and ensure that the software is secure and up to date. Therefore, regarding open-source projects with a lively and active community, one can assume that the code quality is high and generally in a good shape. Unlike proprietary software, that is eventually developed by just one company, the community has insights and helps to secure the quality of the code. Given a large and open community, OSS is also meant to be quite stable. Dependencies from individual, proprietary software companies are not an issue. Even though, the original developer of the OSS code would no longer be active, the community has the possibility to maintain and enhance the code. Thus, the stability of the code can be assured. This is especially relevant, e.g., in the context of machine or aircraft maintenance respectively assets that are characterized by a long lifetime and where maintenance and service providers may change over the life cycle. Moreover, the modular and transparent nature of OSS allows for high interoperability. Therefore, most OSS components allow users, given the respective license compatibility, to be combined and integrated into existing software applications flexibly. OSS entails the great potential for reducing incompatibilities of interfaces allowing the easy and hands-on creation of new products and services. These advantages can be used from a strategic perspective concerning the three different engagement levels of usage, contribution, or provisions as described in the following pages.

Technology	
Quality	Interoperability
■ Security	■ Flexibility
■ Code Quality	■ Independence
■ Stability and Longevity	

Strategic impacts of usage, contribution, and provision from a technology perspective

Quality

Strategic goals

Security

- **Usage**
  As the open-source community is checking and reviewing the source code regulary, bugs or security vulnerabilities can be quicker detected. Therefore, using OSS could be a security-relevant decision.

- **Contribution**
  By contributing to open-source projects, the company is actively engaging in the development of the respective software and therefore contributes to its security. From a strategic perspective, this can be important if the company wants to ensure the security of the software, as it uses the software on its own. Therefore, new developments and decisions can be shaped by the company.

- **Provision**
  By actively providing own open-source projects, companies can benefit from external developers that actively check and alter the code and fix eventual shortcomings. Therefore, from a strategic point of view, companies that do not possess many internal developer resourcers could benefit from the external network and thus enhance the security of their code.

Code Quality

- **Usage**
  Moreover, the usage of OSS allows to observe the general code quality in advance. Companies that wish to understand the code quality and want to make sure that the code quality is up to their standard can benefit from open source compared to proprietary software.

- **Contribution**
  The contribution to open-source projects helps to establish and secure the general code-quality of a certain projects and can be seen as generally welfare-benefitting. Moreover, the company benefits from updated projects and sustains the open-source community.
  If few companies contributed, other developers or contributors are less inclined to be engaged in the community.

- **Provision**
  The same applies to code quality. The contributions and knowledge of external parties benefits the code quality and helps to develop the software and reach a certain level of maturity. By providing their own OSS companies can benefit from the external feedback.

Stability and Longevity

- **Usage**
  In case the software is needed for long-maintenance assets, such as machines with a long lifecycle, a company should consider open-source projects to avoid dependency on proprietary software providers.

- **Contribution**
  By contributing to software that has systemic relevance for the company's products, such as machines or long-living assets, the company can reduce dependencies on software companies as they engage in the software development process themselves and thus keep "up-to-date".

- **Provision**
  From a strategic point of view, it can be beneficial to provide code under an open-source license if it becomes clear that long-term service for clients cannot be sustained or is not part of the business model. If clients, e.g., have machines with a lifespan of 30 years or more, it may not be in the interest of the software application company to provide such long-term service due to cost or personnel reasons. By providing the respective code as open source, the client can manage the code by him- or herself.

Flexibilty

- **Usage**
  If companies wish to build and provide their software products for their customers, the usage and integration of open-source components can be a valuable strategy. Combining different software elements into a new product can be efficient and cost-saving.

- **Contribution**
  Contribution to software projects can allow for co-creation rights. Thus, companies can actively shape the direction in which the software project is heading towards. Thereby companies eventually ensure an easy integration into the other software projects they are currently working on.

- **Provision**
  Providing own code under an open-source license allows for easy integration of the respective software into other systems of customers and partners. Therefore, costly alterations and time-consuming legal negotiations and contracting can be avoided. Moreover, customers may value the fact that the software is open source and use it more than proprietary alternatives - Thus a competitive advantage may result for the respective providing company.

4.2 Organization

Besides technical aspects, OSS also provides various benefits in the company's perspective as an organization. OSS has a beneficial influence on culture and business development.

Companies that provide developers the possibility to engage in open-source projects are more attractive than companies that only develop proprietary. Software developers value open-source developments as they cannot only gain reputation but also share their work and benefit from the positive aspects described. Moreover, joint developments in open-source communities may also allow for talent acquisition with respect to the overall reputation of the company. Many developers program software in their free time and if they happen to find out that the specific project they are contributing to is run by a company with open positions, they may be inclined to apply. Thus, from the perspective of a human resources department, offering the possibility to develop software code under an open-source license, can be viewed as a huge possibility for talent recruiting but also ensures the satisfaction of employees working in the field of software development.

OSS may also contribute to the culture of the company in the medium and long run. First of all, open source can be primarily understood as a collaboration model. OSS code is developed openly on platforms such as GitHub and entails certain quasi-standardized processes and rules that govern the process of software development. Thus, software developers that engage in open-source projects may be used to a structured but open and agile development process and therefore the overall workflow concerning communication and a common understanding of "how to develop software in a team" can be enhanced by OSS engagements. Along with a generally enhanced and more efficient workflow goes the aspect of transparency. As software is publicly developed in the open-source context, developers endeavor to provide high quality and contribute reliably to the software product as peer developers can immediately see their progress. This transparent process may ensure quality and motivate developers to work on the project continuously. Finally, the fact that source code is shared with a community of potential customers and partners, can be understood as a form of open innovation.

Within the concept of open innovation new products and services are not developed internally but together with customers respectively the relevant network (Chesbrough 2010). Customers or partners are integrated early in the process and thus can contribute not only knowledge but also their specific needs and demands. Therefore, products designed in an open innovation process may yield much higher acceptance rates in the markets than products entirely developed internally. The same applies to OSS: Projects that are developed under an open-source license allow customers to be directly involved in the development process and communicate their demands. The same applies to important business partners: They could immediately create application programming interfaces (APIs) to ensure that the product matches with their complementary products or services. Therefore, OSS enables open-innovation processes and fosters an open mindset towards co-creational processes.

In addition, from a business development perspective, OSS can entail various benefits. First of all, companies that engage in open-source communities or projects and that support them either by providing developers as a resource or donating money to respective open-source foundations can build a reputation in the open-source and thus software and tech community. Those companies show that they may not only use OSS, which can be viewed as a donation from private developers or other companies but also give back to the community they benefit from. Therefore, the aspect of reputation and community engagement should not be underestimated. However, of great importance from a business development perspective are the possibilities to design various open-source business models. Although companies do not generate a direct revenue stream from OSS in the sense of licensing, they can engage in different business models centered around open source. Examples are for instance the provision of additional services or dual licensing (see chapter 5). From a business perspective, contribution and provision of open source are of high relevance. However, by using OSS some aspects within the organization can also be enhanced or supported.

Organization		
Human Resources	Culture	Business Development
■ New Human Resources and Talent Attraction	■ Workflow ■ Transparency ■ Open Innovation ■ Diversity	■ Reputation ■ New Business Models

Strategic impacts of usage, contribution, and provision from an organization perspective

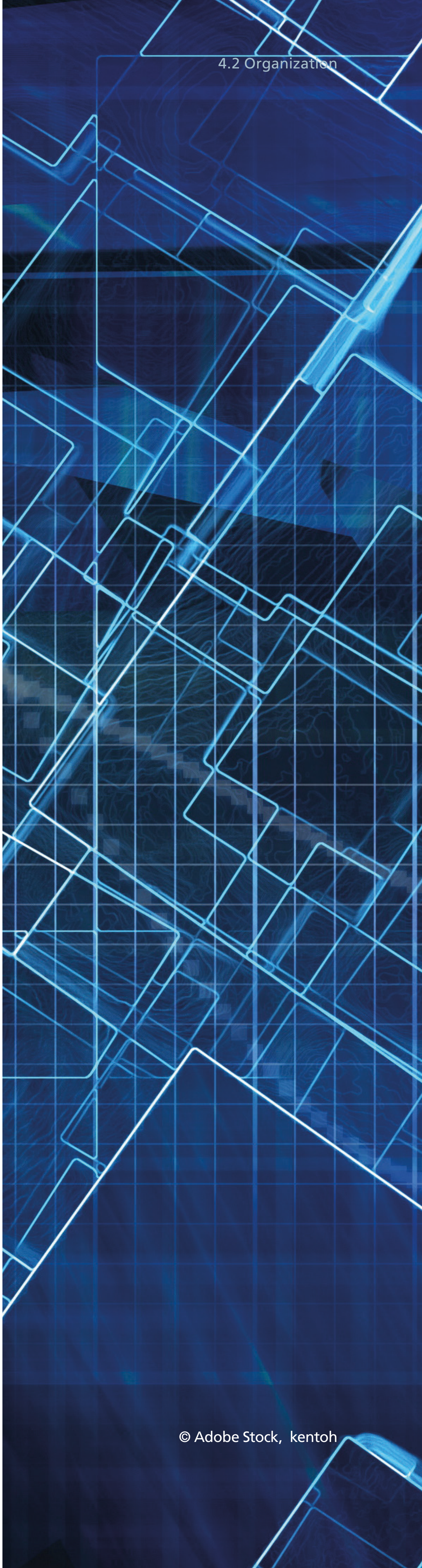
Resources
Strategic goals

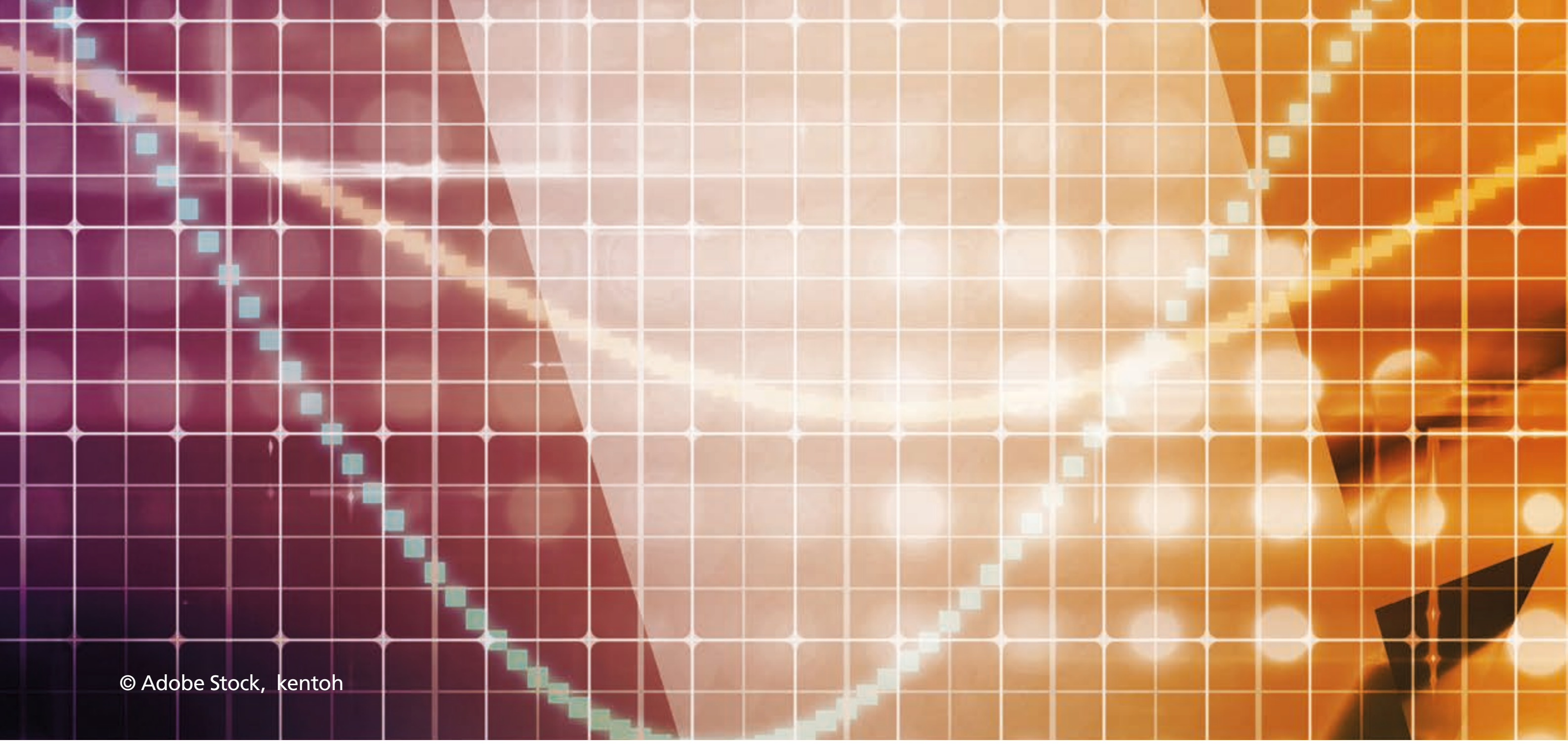
New Human Resources and Talent Attraction

- **Usage**
  Using OSS may be less resource intense in terms of developers and own IT specialists. Firms can use existing solutions and alter them in their favor.

- **Contribution**
  If a company decides, to actively contribute OSS, this may attract potential talents from the IT segment, as they find the option to contribute to open-source projects attractive.

- **Provision**
  The possibility to actively provide own OSS projects is quite attractive for many developers. Thus, they may prefer an employer who allows such freedoms over employers who are hesitant.




© Adobe Stock, kentoh

Culture

Strategic goals

Workflow

- As open source can be viewed as a cooperation and collaboration model and most open-source platforms such as GitHub define certain workflows, companies can benefit from new, agile but structured open-source development processes contrary to existing static development modes. This holds true for providing and contributing OSS. However, also using the OSS can be a first step toward an open-source culture and respective workflows.

Transparency

- **Usage**
  The use of OSS increases transparency as users within the company can easily assess where the software comes from, what it includes and check the communities' developments to stay updated.

- **Contribution**
  Developing and contributing OSS implies being transparent not only about the workflow but also about the individual development steps. This transparency often results in higher code quality and better communication between different parties. The required transparency in a contribution process, meaning that the public can see and judge what the company is contributing, enhances the general level of commitment to "good" and high-quality contributions.

- **Provision**
  Providing own software projects under an OSS license automatically increases transparency of the company's projects, interests and goals. Thus, open-source projects can fulfill an important signaling function. However, not only external parties benefit from open-source projects, but also internal stakeholders such as colleagues from other branches in other countries – as they gain insight into the projects of colleagues and collaborate in an easy manner.

Culture
Strategic goals

Open Innovation

- **Usage**
  Using OSS does not only have the advantage for the company itself but also for the customers and partners that could refer to the same solutions or systems. Consequently, joint value-creation processes can be fostered.

- **Contribution**
  By contributing their code or by collaborating within the framework of open-source projects, companies can benefit from opening their innovation- and software creation processes by getting feedback on their ideas and actively giving feedback to other projects eventually shaping and framing new developments.

- **Provision**
  The provision of own open-source projects allows the operation beyond own, fixed boundaries and fosters relationships with partners, peers, or customers. Those open innovation processes are especially vital if external knowledge is required and companies have a joint interest in working together. As provider of the code, the company can actively manage the open-innovation process.

Diversity

- **Usage**
  Depending on the ecosystem and the specific software type, the company can ensure that it uses a software that is driven and accepted by a diverse and open community.

- **Contribution**
  OSS is not subject to borders, time zones or nationalities. Thus, developers from all over the world can contribute to open-source projects. Different backgrounds - may it be educational backgrounds or simply nationalities of developers - can be beneficial for projects as the diversity of the developers lead to higher acceptance rates and diffusion of the project.

- **Provision**
  By providing open-source code, developers from all over the world can be attracted to work on the project. This can be vital for international companies that need different perspectives.

Business
Strategic goals

Reputation

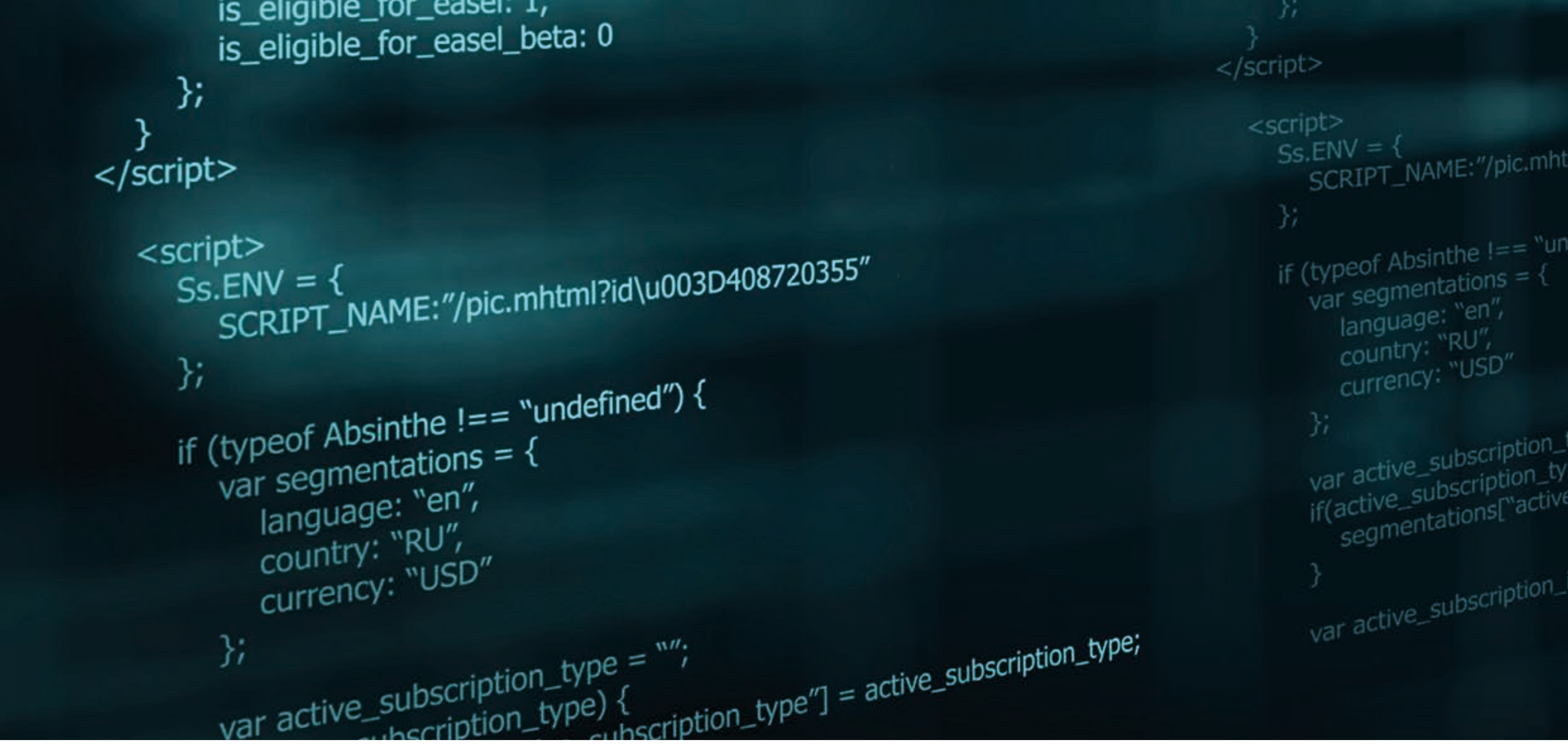
- **Usage**
  The use of OSS can be an important signal to internal developers but also to other stakeholders as the company supports the idea of OSS and its incoming benefits. Thus, the company's internal reputation is increased by allowing employees to use OSS.

- **Contribution, Provision**
  From a business perspective, companies can enhance their reputation in the developer community if they actively provide their projects or support and contribute to open-source projects. This creates a positive image for the company.

(New) Business Models

- **Usage**
  The use of OSS can be seen as part of the value creation process, as OSS saves costs and increases efficiency. Partners may use OSS products which makes the usage of the same product attractive in order to increase interoperability, for example.

- **Contribution, Provision**
  Although OSS itself does not generate a direct income stream, companies can build vital and sustainable business models around OSS (see chapter 5).


```
    is_eligible_for_easel_beta: 0
  };
}
</script>

<script>
  Ss.ENV = {
    SCRIPT_NAME:"/pic.mhtml?id\u003D408720355"
  };

  if (typeof Absinthe !== "undefined") {
    var segmentations = {
      language: "en",
      country: "RU",
      currency: "USD"
    };
    var active_subscription_type = "";
                  subscription_type) {
                    subscription_type"] = active_subscription_type;
```

```
  }
</script>

<script>
  Ss.ENV = {
    SCRIPT_NAME:"/pic.mht
  };

  if (typeof Absinthe !== "un
    var segmentations = {
      language: "en",
      country: "RU",
      currency: "USD"
    };
    var active_subscription_
    if(active_subscription_ty
      segmentations["active
    }
    var active_subscription_
```

4.3. Environment

In a complex and digitalized economy, companies do not only operate within their boundaries meaning a fixed set of partners and suppliers rather than companies operate in digital ecosystems and networks. OSS can play an essential role in building and sustaining those ecosystems and networks. Likewise, open source can also have beneficial impacts on markets by allowing and supporting de-facto standardization. Big, multinational platform enterprises, such as Alphabet (Google) or Amazon have demonstrated how they can build large ecosystems benefitting from strong network effects by using open source. The best example to illustrate this is Alphabet respectively Google: By providing an open-source mobile operating system (Android) the company was able to become the dominant operating system provider for mobile phones.
Although Android can be used without any license costs, some of the apps such as Google Chrome are pre-installed and cannot be deleted. At the same time, Google built a large ecosystem via Google Play where app developers can provide their applications based on various application programming interfaces and software development kits provided by Google respectively Android.

The idea of providing OSS in form of APIs allowing third-party developers to connect to their ecosystems can also be found in the B2B context – however, less distinct and with smaller and more specialized communities. One example is the Bosch IoT platform, providing open-source components allowing interconnectedness of sensors, devices, and gateways.
The open-source approach is helpful for building own ecosystems and connecting various actors, machines, information- and data streams via platforms (see more about business models in chapter 5).

At the same time, by providing open-source infrastructures, de-facto standards can be established. If all companies and developers as well as users "agree" on using the same tools and infrastructure a de-facto standard will emerge. Unlike a de-jure standard, a de-facto standard is a practical implementation that has been established through the widely accepted use and not through standardization committees. By providing OSS, free and easily accessible solutions, companies can help to build a de-facto-standard. The benefits of such standards are easy integration and thus the decline of costs and an increase in efficiency and interoperability amongst various, different systems.

Environment	
Digital Ecosystems	Market
■ Establishment of Digital Platforms and Ecosystems	■ De-Facto Standards

Strategic impacts of usage, contribution, and provision from a environmental perspective

Digital Ecosystems
Strategic Goals

Establishment of Digital Platforms and Ecosystems

- **Usage**
  By using OSS, companies do not directly build their ecosystems and digital platforms. However, they can actively decide which ecosystems they support by deploying the respective software. The integration of the ecosystem's, software also helps to manifest the applications into the daily practice and leads to a wider distribution.

- **Contribution**
  By contributing to OSS of certain ecosystems the respective ecosystem can be actively shaped. Own contributions help to improve the software and created traffic increases the attractivity of the ecosystem's community.

- **Provision**
  By actively providing OSS either via its platform or within foundations, companies cannot only build own communities around their software but can also pursue the business model of digital platforms and ecosystems.
  If companies provide e.g., software development kits or APIs as an add-on to their software, other developers and companies can use this software and build complementary products and services.

De-facto Standards

- **Usage**
  Using certain OSS helps to enable and establish de-facto standards. A de-facto standard lives on its wide application. By deploying the respective software, companies strengthen the standard and can help its distribution.

- **Contribution**
  The contribution of OSS to a project can lead to supporting a de-facto standard. By improving the software and making it more feasible or more adapted to the actual work environment, the software can yield higher rates of acceptance and thus a greater reach. A wide reach, high acceptance rate, and thus high level of distribution helps to establish a de-facto standard.

- **Provision**
  Companies that engage in standard setting as it brings a benefit for their industry can pursue this strategic goal by publishing OSS. Thereby, they may ensure a first-mover advantage: If a company is the first to release software under an open-source license, it may eventually loose certain income streams from licensing but can ensure a high adoption rate of the "free" solution. Thus, the company can benefit indirectly from the wide adoption of the solution resulting in a de-facto standard when complementary or coupled products depend on this "new" standard.

5. Open-Source Business Model

From a strategic point of view open-source business models can play a vital part in the company's business portfolio. The frequently occurring view of open source is often characterized by the misunderstanding that OSS cannot be commercialized. Practitioners new to the open-source business often see the concept as a collaboration tool or as means of reducing costs, without considering its potential for new business models. Well-known business model patterns include dual licensing, open core, professional services, subscription, open APIs and widget frosting.

- **Dual Licensing:** The dual-licensing business model relies on multiple licenses: the software is licensed under both an open-source license and a proprietary license. The business model often uses a copyleft license to avoid the commercialization of their OSS. Besides, a commercial version of the product is offered under proprietary license to generate revenue. Well-known solutions that follow the principle of dual licensing are, for example, MySQL or Asterisk. It is also possible to provide core functions and basic applications as open source and to place special extensions or features under a proprietary license, which describes the following business model.

- **Open Core:** Open core is a business model in which the core software is provided free of charge. Proprietary licenses for advanced features, which make up a smaller percentage, are sold by the company that owns the software. Similar to the dual licensing, this business model is sometimes criticized as some providers design the OSS offering in such a way that it is hardly usable without a proprietary offering. Regarding the proprietary code, there is no community to add value as they cannot contribute to the code.

- **Professional Services:** In addition to dual licensing, companies can provide services related to the OSS. Common service models include support, maintenance, and hosting of software, as well as custom development and customization, consulting, and training. This business model is used, for example, by RedHat which is a professional service (e.g., support, consulting, and training) provider for OSS. In addition, Red Hat offers further OSS solutions and products based on OSS, for example in the area of cloud computing or operating systems. Furthermore, business models can be built around the OSS: Collaborative projects or intermediaries can bring together different actors, similar to a platform, to create value.

- **Subscription:** Red Hat also offers the common open-source business model of subscription. In this business model, certain services, for example support or maintenance of the software, are linked to a certain period of time. As the free availability of the software in results in no dependency on manufacturers (vendor lock-in), open-source subscription models must be primarily characterized by the quality of the service. Red Hat's corporate success is built on community-driven open-source development. Therefore, the company must also have active community management: As a sponsor of the Fedora project, Red Hat supports the open-source community in the further development of the Linux distribution, based on the free package management system originally developed by Red Hat. In this way, Red Hat returns a portion of the revenue it generates based on community-driven open-source development to the community in question and ensures the continuation of its development base.

- **Open APIs:** In addition to provide business models based on OSS, companies can choose to provide software development kits or APIs as OSS. The open boundary resources are used as strategic tool for realizing overarching goals, such as indirect network effects. Especially, large B2C or B2B platforms use open boundary resources to integrate complementary products or services into their platform ecosystem.

- **Widget Frosting:** Lucrative business models can also be realized regarding the compability of hardware and software. For example, the software can be licensed under an open-source license, however, the corresponding hardware must be purchased, or vice versa. The approach is known as widget frosting and a popular approach in the printer industry: The printer's driver is often made open source as the core business is based on the physical product. In this way, the user can customize and maintain the software, which reduces the effort of the company.

In general, OSS has an accompanying function that supports the business model. The OSS is not necessarily market-differentiating and does not affect the company's core intellectual property. Instead, the OSS is an accompanying or complementary service to the core business and rather used to achieve strategic goals. The company's actual core business is, for example, consulting, mechanical engineering, the operation of a digital platform, or the design of individual software applications.
Therefore, the development and provision of OSS are less altruistic as a first glance might suggest as companies follow strategic and commercial interests. It becomes clear that OSS provides the basis for successful and sustainable business models. However, commercial interests do not contradict open-source principles as the provided OSS can be a valuable tool for users and companies could have a strategic interest in thriving open-source communities and projects. Companies such as Microsoft and Google therefore explicitly promote various open-source communities with financial, but also with human resources. Blind et al. (2021) note that in a sample of 1,151 European companies from 14 different industries, companies from the IT sector are the main force in driving open-source projects as they actively contribute to OSS development on GitHub with a share of 77 percent, followed by research institutions and public institutions with 7 percent. Companies from the mechanical engineering sector actively contribute to open-source development with only 1 percent. The smaller companies in particular invest a relatively large amount of resources in OSS development. With a share of 88 percent, SMEs are the main contributors to open-source projects.

E: 234
R: 31
E: 135
R: 22
E: 415
R: 56
E: 112
R: 12
E: 112
R: 41
E: 317
R: 89
E: 475
R: 23
E: 43
R: 65
E: 76
R: 32
E: 512
R: 64
E: 231
R: 78
E: 87
R: 111
E: 334
R: 45
DATA ANALYZING
PORT 01
PORT 02

Table 2: Open-Source Business Models

Dual Licensing / Open Core			
Description	Advantages	Disadvantages	Examples
■ Companies that provide a freely available basic offering (open source) and proprietary complements	■ Better marketing ■ Attraction of new customers ■ Achievement of high margins	■ Product complexity ■ Consideration of legal aspects (licenses etc.)	■ Elastic, Confluent, MySQL
Professional Services			
Description	Advantages	Disadvantages	Examples
■ Companies that generate revenue through professional services (e.g. development, consulting, support, etc.)	■ Low investment costs ■ Attractive in consulting-intensive industries	■ Low utilization of the service ■ Low profitability ■ Commodity service in professional OS business models.	■ RedHat, Hortonworks
Subscription			
Description	Advantages	Disadvantages	Examples
■ Companies that provide a time-limited product offering.	■ Better calculation of demand ■ Achievement of high margins	■ Consideration of legal aspects ■ Less acceptance in OS community	■ MongoDB
Open APIs			
Description	Advantages	Disadvantages	Examples
■ Companies that provide open-source boundary resources to their proprietary offering.	■ No risk of losing intellectual property ■ Easier for beginners to enter the open-source business	■ Primarily proprietary business models	■ Facebook, Apple
Widget Frosting			
Description	Advantages	Disadvantages	Examples
■ Companies that sell complementary physical products, such as printers, hardware, or machines.	■ Suitable for industry ■ Increased customer satisfaction through customization ■ Offered software is not one of the company's core competencies	■ No direct increased profit margin by opening software	■ Mercedes-Benz, Smart-Things, Clover

6. Practical Check-List for Open-Source Software within Companies

After a company has assessed and evaluated whether open-source usage, contribution, or provision is an interesting and beneficial option from a strategic point of view, it should design an open-source strategy. Thereby, it makes sense to establish an Open Source Program Office (OSPO) – a division or group of people that actively deal with and manage open-source activities. For small companies and start-ups that do not have these resources, at least one distinct person should be responsible for managing the activities within the company – regardless if it is usage, contribution, or provision. Companies and developers that wish to use OSS should thereby ask and answer the following questions:

Table 3: Checklist for OSS Usage after AlMarzouq et al. (2005) and own additions

Community	
Questions	Indicator for Usage
▪ What is the size of the community and how do the growth trends look like?	▪ Big or growing community: A large community serves as indicator for active product development and thus increased product quality.
▪ Are the users actively participating in the community?	▪ Significant portions of the community are active participants. A high participation rate indicates a high software quality and active support from members.
▪ Who are the core players of the community?	▪ The objectives of the core players should not be contrary to the organization's own goals.
License & Legal Issues	
Question	Indicator for Usage
▪ Can we accept the restrictions set up by the license?	▪ The licensing terms should be carefully reviewed before use of the software in a commercial context.
Development Process & Organization	
Question	Indicator for Usage
▪ Do we have sufficient technical capabilities to use or customize the software?	▪ As part of evaluation, the technical capabilities, such as hardware or IT personnel, that are necessary to operate the OSS should be considered.
Software & Technology	
Question	Indicator for Usage
▪ Is the software up to date and does it meet our technological and safety requirements?	▪ It is important to assess whether the OSS is frequently updated, active commits are made and security updates are available.

Companies and developers that want to actively contribute OSS in form of own code should ask following questions in addition:

Table 4: Checklist for OSS Contribution after AlMarzouq et al. (2005) and own additions

Community	
Questions	Indicator for Contribution
▪ Does the community possess broad knowledge and expertise?	▪ The community is effective in using the knowledge of its members and solving problems that arise.
▪ Does the community have a clear structure and rules of organization and even a code of conduct?	▪ The community is well organized, and the development teams are organized in a modular way. Communication is fast and professional.
▪ Who is part of the community and do the active com-mitters or sponsors align with our company's goals and objectives?	▪ The community is diverse but driven by a common spirit and vision that is in line with the own vision and goals.
License & Legal Issues	
Questions	Indicator for Contribution
▪ Do we contribute only own intellectual property or do we violate other licenses or patents by publishing our results?	▪ The contributions are based on own intellectual property and it is clearly indicated whose intellectual property it is and that no other property rights are violated.
▪ Do we agree with the license and the conditions under which the project and the code is published? Do we agree with the terms and conditions of a contributor license agreement?	▪ The terms and conditions of the license are in line with the company's policy and no negative side effects can occur from contribution. The contributor license agreement is clearly understood and accepted.
Development Process & Organization	
Question	Indicator for Contribution
▪ Do we have to change the development process, organizational structure, or routines to participate?	▪ The current organizational structure promotes the parti-cipation of our employees in open-source projects. Our employees are acquainted with open-source development. If not, a change in structure can be managed and teams or persons that are liable for setting up a new open-source process can be trained.
Software & Technology	
Questions	Indicator for Contribution
▪ Is the software design modular?	▪ A modular software design facilitates a decentralized development process.
▪ Does the software meet the firm's quality and security standards?	▪ The software project that is contributed to, is in line with the company's quality standards.
▪ Can we meet the quality standards and level of the community?	▪ The company can adhere to the quality and safety standard and ensure the required level of development of the open-source project.

Companies and developers that want to actively contribute OSS in form of their code should however in addition ask at least the following questions:

Table 5: Checklist for OSS Provision after AlMarzouq et al. (2005) and own additions

Community	
Questions	Indicator for Provision
▪ Are we able to motivate people to participate in the community?	▪ The software stimulates the interest of programmers, or the participation of customers is expected.
▪ Can we afford the time and effort to initiate the community and to participate?	▪ The values and ideals of open source are part of our culture. Therefore, we know how to promote participation in the community.
License & Legal Issues	
Questions	Indicator for Provision
▪ Can we establish appropriate licenses and contributor license agreements and its accompanying implications to ensure our benefit?	▪ The company is aware of the implications that come with copyleft or permissive licenses as well as dual-licensing strategies.
▪ Are we aware of own patents or other valueable intellectual core property that we could undermine and well as patents and intellectual property of others?	▪ No intellectual property is violated. Likewise, the contributions and the intellectual property of employees is clearly regulated
▪ Are our provisions and contributions that build on other software projects compatible with the license?	▪ All software components that are part of the open-source project are compatible with respect to their licenses.
Development Process & Organization	
Questions	Indicator for Provision
▪ Do we have sufficient technical capabilities (hardware systems, knowledge, and skills) to initiate an open-source process?	▪ Capabilities to contribute to the initiation and development are sufficient.
▪ Do we have to change our organizational structure or routines to participate?	▪ The current organizational structure promotes the participation of employees in open-source projects. The employees are acquainted with open-source development methodologies.
▪ Is the open-source provision in line with the company's strategy? Are all important divisions involved and is broad consensus and commitment given?	▪ The open-source strategy has been clearly communicated and assessed with respect to its overall fit to the company's goals.

© Adobe Stock, Maximusdn

Table 5 (continued)

Software & Technology	
Questions	Indicator for Provision
▪ Are we clear on why we want to release our software as OSS?	▪ Yes, e.g., we would like to increase our competitiveness or enable open innovation.
▪ Is the software that we provide up-to-date? Is the quality assured and are all safety concerns accounted for?	▪ Only high-quality, up-to-date products are published and no safety issues arise for the company. Other users cannot harm the company on the basis of the software neither can the software harm other users due to misfunctions or lack of quality.
▪ Will the release of the source code impact our competitive advantage?	▪ The released software or the parts that are to be released are commoditized.
▪ Does our software design allow open-source development?	▪ Software has a good modular design so that decentralized development processes, further expansion and growth is possible.
▪ Do we engage in a technology field that brings us benefits with respect to knowledge inflows, business models or sustainability?	▪ The technological perspective has been aligned with the strategic perspective on open source. It does make sense from both perspectives to publish open-source code.

© Adobe Stock, Melisa

Let’s get started: Together!

The underlying Whitepaper has provided a first stepping stone for developing open-source strategies by showing important strategic aspects along the dimensions of technology, organization, and environment. However, every good strategy-building process is followed by implementation. Especially small and medium-sized companies or companies that are “new” in the open-source community may need support or a sparring partner helping to specify the next steps.

Research Institutes such as the Fraunhofer IML and Fraunhofer ISST, but also the TU Dortmund can be vital partners not only for developing but also for implementing open-source strategies. Together with big companies but also small and medium-sized companies and start-ups we, the Fraunhofer IML, Fraunhofer ISST, and TU Dortmund, want to build a strong, vibrant open-source community around logistics, supply chain management, and Industry 4.0 applications. We started this undertaking by building the “Silicon Economy”.

The Silicon Economy is not only a publicly funded project but our mutual vision: We aim to build OSS components together with a user community coming from various industry sectors and fields of applications. The purpose of joint open-source development is to identify and realize commodity components that can ease processes through a de-facto standardization. At the same time, companies can join forces to build new and innovative components for their specific business process and integrate e. g. blockchain modules that are provided through the publicly funded partner project Blockchain Europe.

In order to create a space for this joint OSS development, the exchange of ideas and the identification of beneficial commodity applications, the Open Logistics Foundation was brought into being. The Open Logistics Foundation supports a European-driven, open-source community focusing on logistical applications. The associated Open Logistics e. V. invites all companies that are interested, to participate in this open-source community. At the same time, Fraunhofer IML and it’s partners support individual companies in their intention to develop open source – either from a strategy, business model- or technology perspective.

More information about our open-source projects and initiatives in Dortmund can be found under:

https://www.openlogisticsfoundation.org/
https://www.silicon-economy.com/
https://blockchain-europe.nrw/

Publication bibliography

AlMarzouq, Mohammad; Zheng, Li; Rong, Guang; Grover, Varun (2005): Open Source: Concepts, Benefits, and Challenges. In CAIS 16, pp. 505–521. DOI: 10.17705/1CAIS.01637.

Banon, Shay (2021): Amazon: NOT OK - why we had to change Elastic licensing. Available online at https://www.elastic.co/blog/why-license-change-aws, checked on 5/7/2022.

Blind, K.; Böhm, M.; Grzegorzewska, P.; Katz, A.; Muto, S.; Pätsch, S.; Schubert, T. (2021): The impact of Open Source Software and Hardware on technological independence, competitiveness and innovation in the EU economy, Final Study Report. Brussels (Final Study Report), checked on 10/20/2021.

Chesbrough, Henry William (2010): Open innovation. The new imperative for creating and profiting from technology. [Nachdr.]. Boston, Mass.: Harvard Business School Press.

Fitzgerald, Brian (2006): The Transformation of Open Source Software. In MIS Quarterly 30 (3), pp. 587–598. DOI: 10.2307/25148740.

Forschungsbeirat der Plattform Industrie 4.0/acatech (Hrsg.): Open Source als Innovationstreiber für Industrie 4.0, 2022, DOI: 10.48669/ fb40_2022-2

Free Software Foundation (2004-2020): What is free software and why is it so important for society? Edited by Free Software Foundation. Available online at https://www.fsf.org/about/what-is-free-software, checked on 7/31/2020.

Gentermann, L.; Termer, F. (2019): Open Source Monitor. Research Report 2019. Edited by Bitkom e. V. Berlin. Available online at https://www.bitkom.org/sites/default/files/2020-04/200420_eng_bitkom_report_openmonitor_2019.pdf, updated on 2019, checked on 4/6/2022.

Harutyunyan, Nikolay (2020): Managing Your Open Source Supply Chain-Why and How? In Computer 53 (6), pp. 77–81. DOI: 10.1109/MC.2020.2983530.

Hippel, Eric von; Krogh, Georg von (2003): Open Source Software and the "Private-Collective" Innovation Model: Issues for Organization Science. In Organization Science 14 (2), pp. 209–223. DOI: 10.1287/orsc.14.2.209.14992.

Okoli, Chitu; Nguyen, Johannes (2015): Business Models for Free and Open Source Software: Insights from a Delphi Study. In : Proceedings of the 21st Americas Conference on Information Systems. Puerto Rico, checked on 4/14/2020.

Open Source Initiative (2007): The Open Source Definition. Available online at https://opensource.org/osd, checked on 12/11/2021.

Rajala, Risto; Nissilä, Jussi; Westerlund, Mika (2006): Determinants of OSS revenue model choices. In : Proceedings of the 14th European Conference on Information Systems. Göteborg: Sweden. Available online at https://aisel.aisnet.org/cgi/viewcontent.cgi?article=1050&context=ecis2006, checked on 4/7/2020.

Raymond, Eric Steven; Perens, Bruce (2018): Open Source Initiative - History of the OSI . Available online at https://opensource.org/history, updated on Oktober 2018, checked on 7/31/2020.

Schmidt, Michael; Culotta, Carina; Nettsträter, Andreas; Duparc, Estelle (2022): Die Rolle von Open Source in der Silicon Economy. In Michael ten Hompel, Michael Henke, Boris Otto (Eds.): Silicon Economy, vol. 29. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 19–40.

Stallman, Richard (1999): The GNU Operating System and the Free Software Movement. In Chris DiBona, Sam Ockman, Mark Stone (Eds.): Open Sources. Voices from the Open Source Revolution. Sebastobol: O'Reilly & Associates, pp. 31–38.

Tornatzky, Louis G.; Fleischer, Mitchell (1990): The processes of technological innovation. Lexington, Mass.: Lexington Books (Issues in organization and management series).

West, Joel (2003): How open is open enough? In Research Policy 32 (7), pp. 1259–1285. DOI: 10.1016/S0048-7333(03)00052-0.