LINUX FOUNDATION

How to Build Open Source Competency in Your Company

A Linux Foundation publication

www.linuxfoundation.org

Businesses today have strong motivation to get more involved in open source communities and collaborative development.

According to the Linux Foundation's 2014 Collaborative Development Trends Report, the benefits of using open source software are now well established. The managers surveyed cited shorter product development cycles and faster time to market (77 percent); reduced engineering and development costs (66 percent); and better relationships with customers and business partners (55 percent), among many other advantages.

Combined they create a strong economic force that's transforming diverse industries. No area of technology is untouched. From cloud and hyperscale computing, to big data and storage, web infrastructure, embedded systems, mobile technology, drones and robotics, and the Internet of Things—open source projects like Linux, Hadoop, OpenDaylight, and OpenStack, are dominating or disrupting industry ecosystems.

Open source technology creates a competitive advantage for the companies that use it. It's no longer economical for one company to create its own infrastructure, such as a new database, operating system, or Platform-as-a-Service (PaaS), from scratch. "Why would they? Instead they build atop the shared research & development (R&D) investment of an entire industry," says Jim Zemlin, Executive Director of The Linux Foundation. "2014 marked a tipping point for open source in which companies decided there was too much software to write for any one company to do it by themselves."

Companies are shedding commodity software development and investing in "external R&D," using open source projects as a base on which to build their own tailored software. They're also beginning to understand that to keep pace with fast changing markets they need not only to use open source software, but to participate in its development.

But integrating into open source communities takes time and effort and requires a new approach to product development. Whereas traditional, proprietary development requires secrecy and a management hierarchy, collaborative development requires openness and values consensus. Code contributions, not title or position, are what determine influence and technical direction in an open source project.

Collaborative development takes place in diverse and geographically dispersed communities that have their own rules, conventions, tools, and processes. Simply put, each community has its own unique culture and it takes time to establish the trust, ways of collaborating, and cultural understanding required to be effective in open source. Companies that embark on this process, however, and commit time and resources to open source development see phenomenal results.

“Those who master the game have a compelling advantage. Those who don’t are getting left behind,” says Zemlin.

As the nonprofit consortium behind Linux, the largest open source project in the history of computing, The Linux Foundation has a deep knowledge of the open source development process, its culture and business applications. Founded in 2000, The Linux Foundation supports its member companies through every step of open source involvement, helping them to navigate the world of open source communities, and enabling them to begin using open source software, contribute code to the projects, influence their technical direction and finally to initiate their own projects. This paper provides an overview of that process as well as seven best practices, gleaned from years of collaboration with open source leaders, that will help companies proceed along the path to open source mastery.

The Four Stages of Open Source Mastery

A company's open source participation typically follows a spectrum that begins with consumption—using open source tools and components—and progresses to code contributions with increasing levels of commitment. Contributors start by fixing bugs, then adding features, and finally starting and leading new projects. The process typically involves four stages: consumption, participation, influencing, and initiation.

Consumption—Companies look for open source projects that offer a benefit to products, either now or in the future. They use these projects as a base on which to differentiate their own products, services, or infrastructure.

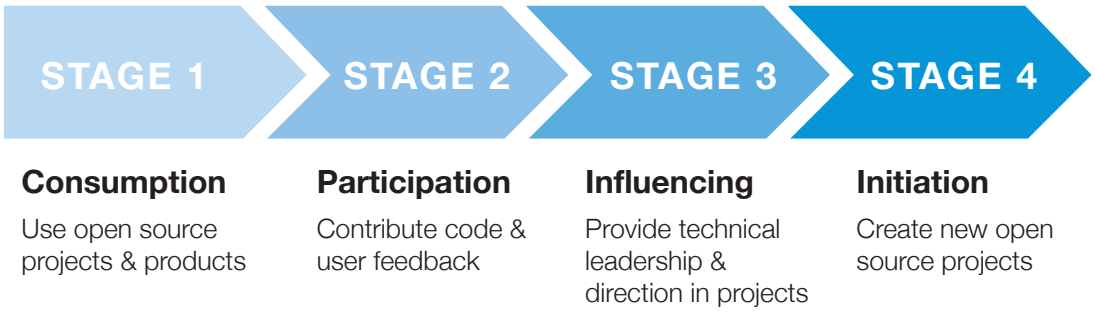
Participation—Once a company is consuming an open source project, its developers begin finding and fixing bugs in that project. When this happens, they can commit those fixes back upstream to the central hub of the code so that the company's code then becomes a permanent part of the open source project.

Influencing—A company can leverage open source by identifying which project they are using that needs additional functionality for their own purposes. They will begin to make changes to that project to change its technical direction and advance it in a different way or take it further.

Initiation—After participating in open source communities long enough to build a reputation, a company is in a position to benefit from shared R&D (see #6 in the best practices section, below) and/or commoditize a competitor's technology. It can share, or open source, its proprietary projects that could be of use to the community. Or it can create new open source projects from scratch and benefit from collaboration among external developers.

The 4 Stages of Corporate Open Source Participation

Where's your company on the spectrum?



Best Practices for Building Open Source Competency

Companies that have been using and participating in open source for a while and have found success—innovating through open source—typically follow many of the same best practices. Open source programs can be structured many ways – from one or two advocates on an engineering team to entirely separate R&D divisions. Regardless of the specific path, companies adopt and evolve these practices at their own pace to suit their own needs. The most successful programs eventually achieve a level of mastery in each.

Below are seven key best practices, accompanied by anecdotes from technology industry leaders that are currently working to build their internal open source programs.

When taken individually, each practice helps address a common issue companies face when integrating open source technologies and practices. Collectively, they are the building blocks for establishing open source mastery within a company.

1. Hire an open source program manager

An open source manager is an executive-level position that provides hands-on management of corporate activities and involvement in open source projects. Many companies have found that having a manager for their open source efforts is a crucial way to grow their open source programs. The program manager is dedicated to and responsible for increasing open source participation within a company, ensuring legal compliance with open source licenses, and building the company's reputation within open source communities.

Last year, SanDisk committed to increasing its strategic use of open source, recognizing that open source is an engine for innovation in many of the markets in which it plays, including mobile, cloud, enterprise, and connected consumer devices. The flash-storage giant opened a new Open Source Strategy Office and hired Nithya Ruff to be its open source director. Ruff, a veteran Linux and open source product manager, was hired to manage the program after the company started receiving an increasing number of questions from developers regarding the use of open source and license compliance.

SanDisk has since become the 7th largest contributor to the open source Ceph project. And it's been working on flash optimization of drivers, file systems, and memory with various SoC vendors and on projects such as Android, Hadoop, and Apache Cassandra, as well as contributing upstream to the Linux kernel. Such contributions put SanDisk, a newcomer to open source, on the map as a company that's serious about working with other companies and projects, Ruff says.

Qualities of an Open Source Program Manager

- Management experience in the technology sector
- High level of comfort with and passion for technology and open source
- Legal understanding of open source licensing and compliance
- Well-known and respected within at least one major open source community
- Highly organized with the ability to set, manage, and implement priorities
- Able to build consensus from public discussions, both internal and external
- Travels several times a year to attend major open source events
- Excellent communication skills, in writing and public speaking

To help build successful open source practices, a program manager has a wide range of qualities and qualifications. He or she should be a generalist and a people person, a passionate believer in open source who is technical enough to earn developer respect. He or she should also understand the legal implications of collaborative development, including licenses and compliance issues. The job also requires business acumen. Open source development is still thought of by some as "giving code away for free." The open source program manager should be able to explain the business benefits of collaborative development and raise awareness of its contributions to the company's revenues. Further, the job requires marketing inside and outside the company to increase the program's visibility and create momentum.

The more a company draws attention to its open source efforts, says Ruff, the more interest it will receive from the open source community.

2. Identify critical open source projects

The open source program manager can work across teams and departments to identify which open source projects are critical to a company's products or services. He or she can then work to integrate the selected open source code into the technology stack by creating an "open source map," which will provide strategic direction for a company's open source participation. To go beyond consuming open source projects, a company needs to establish processes to facilitate their developers making contributions to those projects and communities. Once a company understands its IP portfolio and the open source contributions its developers want to make, it can work to ensure its developers are empowered to make those contributions.

3. Establish a compliance program

In proprietary software development, tight control over source code is maintained. It's the sole right of the proprietor to add features or fix problems. Licenses simply permit users to run the software on their computers. In contrast, open source software is released under

Types of Open Source Licenses

- **General Public License, or GNU**
  Allows use of the software on condition that the resulting code is open-sourced with the same license (copyleft license)
- **Lesser General Public License (LGPL)**
  Allows usage of libraries without release of source code (weak copyleft)
- **Permissive License**
  Allows use as part of programs distributed under other licenses, including proprietary licenses (non-copyleft)
- **Apache License**
  A permissive license that also grants patent rights to users from contributors
- **BSD License**
  A permissive license that allows free use of code with proper attribution and without warranty

licenses that allow source code to be used, modified, and shared freely. Licenses include the developer's name as the originator and owner of the code and often require projects that modify the code to release changes back into the open source community.

When open source code is used for commercial products, questions of license compliance can arise. Companies that use open source software in their products have established open source compliance programs to address these issues. Most companies create a FOSS compliance program and set up a core team, usually called the Open Source Review Board (OSRB) to ensure proper compliance. The OSRB often consists of representatives from engineering, product teams, and legal in addition to the Compliance Officer (sometimes called Director of Open Source). In addition to the core team, an extended team that consists of various individuals across multiple departments (Documentation, Supply Chain, Corporate Development, IT, Localization, etc.) also contributes on an on-going basis to the compliance efforts.

A good open source compliance program brings together processes, training, tools and personnel to ensure the company's usage respects copyrights, complies with license obligations, and protects the company's own intellectual property and that of its customers and suppliers (whose source code is included in the product). For more information, see our Open Compliance Program at https://www.linuxfoundation.org/programs/legal/compliance.

The open source community operates on the basis of reputation in addition to technical expertise, so compliance problems can damage a company's standing within the community. For example, last year Samsung violated the GPL when it didn't release software back to the community that contained some Linux code. The company fixed the problem within two weeks, according to Ibrahim Haddad, Vice President and Head of the Open Source Innovation Group at Samsung, but the distrust still lingers.

Samsung gets software from thousands of partners and ships millions of devices per quarter; ensuring compliance is not an easy task. Over the past three to four years, their compliance program, led by the company's Open Source Innovation Group, an R&D division of Samsung, has significantly reduced the number of issues.

They began by identifying the common points of failure—where compliance issues were likely to arise. This included failures in policy, process, tooling, the software procurement process, and other minor areas such as failure to produce the proper binaries or a website access error. Then they systematically addressed every failure, coming up with a resolution.

For example, to reduce confusion and errors in their compliance policy among employees, Samsung reduced the policy to essentially one, easy-to-remember sentence: Any code you get from the outside has to go through a process.

The next step was making the compliance process very light weight. They track all incoming software from internal as well as third-party sources through a ticketing system. And there's a 5-step approval process for creating or participating in open source projects, with product managers responsible for smaller requests (such as contributing minor patches to a project) and an open source review board taking larger requests such as forming a new open source project.

They also have a common set of tools for open source projects now, and a set of courses for developers on open source compliance and development. Developers who want to earn the prestigious title of "software architect" at Samsung are required to take those two courses as part of their overall training for advancement, for example.

The Linux Foundation offers certification programs and a comprehensive lineup of training courses in all aspects of open source software development, including compliance. For more information, visit http://training.linuxfoundation.org.

4. Foster open source culture

In open source software development, design decisions, strategies, and discussions happen in public by means of collaborative tools such as version control systems, mailing lists, web forums, and bug trackers. Use of these tools fosters the establishment of a more open and transparent culture for collaborative software development. The tools enable communication—code sharing, discussion, patches—among members of the community and provide an authoritative record of how decisions were made, why they were made, and who signed off on them.

Common collaborative tools include:

Bug Tracking System—Allows developers to keep track of outstanding issues in a project. Examples are Bugzilla and JIRA.

Internet Relay Channel—Provides a discussion forum for collaboration. Also allows private chats, as well as file sharing and data transfer.

Mailing list—Members of an open source community receive and share information regarding the project via email. Provides a searchable record of all activity.

Version Control System—Tracks and controls changes to the source code. Examples include Git and Subversion.

Web Forum—Provides a central location for questions and discussions about a project.

Wiki—A web site that allows editing of its content and structure by its users. Open source projects often have their own Wiki.

One of the most commonly used collaborative tools is GitHub, which boasts more than 9 million users. Based on Git, a distributed version control system created by Linux creator Linus Torvalds in 2005, GitHub is a comprehensive source code repository designed to facilitate and track collaborative development. It supports many useful collaborative features such as version control, mailing lists, bug tracking, release management, and wiki-based documentation.

Choosing collaborative tools is a very personal thing for communities. A community might revolve around a mailing list alone, like the Linux kernel community, or it might use a combination of tools. Tizen, a Linux Foundation Collaborative Project, has a Wiki, a mailing list, and web forums, among other development tools.

For companies just beginning an open source program, using collaborative tools for internal functions can give developers experience with the open source development process that will make participation in external open source projects easier. Using common open source development tools internally also helps to establish a culture of openness and can aid in recruiting open source developers accustomed to working with the tools and related methodologies.

The Top 5 Ways Developers Participate in the Open Source Community

Source: Linux Foundation Collaborative Development Trends Report, 2014.

1. Use open source development tools
2. Use open source in commercial products
3. Participate in open source events
4. Test and submit bugs
5. Actively contribute code

5. Hire or grow open source developers

Qualified open source developers—those experienced in the unique practices of collaborative development—are scarce and in demand. It can take months for developers to achieve contributor status on a project, and they need to be active at high levels on a project for even longer to achieve the status of key contributor or maintainer. Status is earned not only through code contributions and technical savvy, but through a contributor's ability to work within the community.

"Project to project, the structure differs and so you really need to know how to work with Apache versus how to work with the kernel versus how to work with the Android group and which mailing list to be on and how to communicate with the people," says SanDisk's Nithya Ruff. "Who's the maintainer of the project? What's the etiquette for submitting a patch? What's the coding style? It's very hard to find people who have not just the technical skills, but the skill of the ecosystem of Open Source and how to work with it."

This scarcity has turned the old way of hiring on its head. Because developers are mobile and in short supply, they have a multitude of opportunities and can pick and choose between the companies and projects they'd like to work on. So how do companies find and recruit developers to build their teams?

One solution is to look inside the company. Find developers who are already doing open source internally and bring them together, create an open source working group that shares information, discusses open source issues, creates best practices, and mentors those just starting out. SanDisk implemented this strategy and less than one year later, Ruff says, she had internal developers approaching her asking to be trained in open source. Now she has developers from outside the company approaching her as well.

Developers can also participate in mentoring programs outside the company. For example, The Cloud Foundry Foundation offers a mentoring program for developers, which allows programmers from the Cloud Foundry community to participate in pair programming with experienced committers on a Cloud Foundry project. Cloud Foundry, a Linux Foundation Collaborative Project, is a successful open source cloud PaaS that enables developers to build, deploy, run, and scale applications on public and private clouds. Through the

mentoring program, developers get experience with the core technology and the development environment, and can make direct contributions to the project.

The other way to build an open source team is to hire well-known open source developers as a way of anchoring the program, bringing in expertise and attracting other developers to the company. Guy Martin, Senior Strategist in Samsung's Open Source Group, recommends "hiring people of caliber that other developers want to work with." His group uses networking and personal connection to recruit open source developers.

A company's level of participation in open source projects can also help attract developers. Martin says developers don't want to work for a company that's just a consumer of open source. They want to work for one that's contributing back or initiating projects.

In other words, hiring or training the right people and contributing to open source projects creates a virtuous cycle that also builds a company's reputation within the open source community and allows them to begin to influence the direction of projects and to initiate projects as well. For more detailed information on this topic, download the other two white papers in this series: How to Recruit and Hire Open Source Developers and How to Reach and Influence Open Source Developers.

6. Leverage external R&D

Open source software development allows companies to take advantage of external R&D. External R&D is the practice of using open source projects as a base on which to build a company's own tailored software, and its benefits are many. It greatly speeds time to market by allowing a company to leapfrog over some of the development time. It saves money because the pre-existing software projects are already completed and are free. It expands a company's horizons by exposing it to new and different ways of doing things. And it aids in recruitment, allowing a company's developers to get to know and work with external developers that might be potential new hires. Experience in open source development is transferable from company to company, so it saves companies training time for new hires as well because the developer has already been working on a company's project before being hired.

The first step in leveraging external R&D is for a company to survey the pre-existing open source projects for software that fits their needs. An abundance of open source projects is easily accessible in public repositories on sites like GitHub. Developers can search these repositories, choosing projects that will provide a base for the product they are developing. This practice can take months or even years off the development cycle. And it's becoming standard practice. "Nobody makes anything these days without open source software," says Zemlin. "You can't build a phone or a television or anything without leveraging a lot of open source software."

Corporate Benefits of Using External R&D

- Speeds time to market
- Saves development time
- Encourages innovation
- Aids in recruitment
- Reduces training time for new hires

James Pearce, Head of Open Source at Facebook, says, “Before we build anything, we look at the open source projects that would help us fulfill those requirements.” Then they test the project to see if it can scale to suit Facebook’s needs. They improve the project, often modifying it to make it scale to the level needed or if it can’t be modified, they replace it entirely with something that does work for them. In addition to building on top of existing open source projects, Facebook also releases software to the open source community. Projects that have been open sourced can attract individual developers and users to contribute improvements as well, another gain for Facebook and for the open source community.

A final important step in Facebook’s approach is to continue to use the projects they’ve released. Pearce says, “If you want an open source project to be successful, continuing to use it yourself is of utmost importance.” That way, projects will continue to be maintained, updated and modified when necessary.

7. Join a Foundation

Open source foundations are key enablers of networking in the open source community at large. Joining organizations such as the Apache Software Foundation, Eclipse Foundation, The Linux Foundation, and Outercurve Foundation, helps companies in all stages along the open source spectrum of participation. Foundations offer a mix of resources and services from legal protection and advice, to financial management, training, and events, all of which strengthen and support their members in the open source community.

By attending, speaking at, and sponsoring open source events, companies can gain access to developers and maintainers in their key open source projects, get oriented to the open source way of collaborating and learn how to participate, influence, and initiate. They can share tips and success stories and gather valuable information and advice on how to improve their programs. Events help companies publicize their programs, recruit open source developers, and build competency. It’s a way of showing thought leadership, illustrating that the organization is open source friendly, and has a business strategy that features open source development.

The Linux Foundation offers comprehensive training to help companies learn how to participate in open source projects, and best consume them, in a way that builds competence internally to eventually influence and initiate projects. The Foundation starts by providing a common baseline of technical competency in Linux and open source. It helps companies find open source talent both from outside and from within, providing training for developers who want to increase their open source expertise. For more information on open source participation, see our book, How to Participate in the Linux Community.

The Linux Foundation also sponsors ongoing collaborative projects. Companies can initiate projects and then build ecosystems around them in a neutral environment provided by the Foundation. This structure enables collaboration of competing parties and provides a voice for the project, which accelerates public awareness and project momentum. Successful projects produce code that becomes a de facto industry standard.

Open Source Projects Hosted by The Linux Foundation

ALLSEEN ALLIANCE	AUTOMOTIVE GRADE LINUX	CLOUD FOUNDRY	Dronecode
IoTivity	Let's Encrypt	node.js	OPEN CONTAINER PROJECT
OPNFV	OPEN VIRTUALIZATION ALLIANCE	bel	OPEN DAYLIGHT
openMAMA	R consortium	TIZEN	Xen Project
yocto PROJECT			

Conclusion

Companies that have achieved mastery in open source have much in common. They use best practices while gaining experience with collaborative development through participation in open source communities. They've created open source programs, run by dedicated managers, that include a system for legal compliance. They have gathered developers with open source expertise either from within the company or hired from outside. They make use of collaborative tools for internal as well as external development. And they actively build a network with others in the open source community by joining a foundation, attending events, and training their team.

Their participation in open source projects follows a pattern of increasing engagement, from consumption, to participation, gaining influence and initiating projects. Consuming can teach a company how to leverage external projects and developers to build their products. Participation can bring more fluency in the conventions and culture of a community. Increased contributions to a project lead to a growing role within the community, enabling greater ability to influence the direction of a project. Initiating projects or releasing existing projects as open source to the community strengthens the sense of give and take, which can further build a company's reputation in open source, and make the company even more attractive to open source developers.

Companies that have made a commitment to collaborative development will continue to be rewarded as the use and importance of open source continues to expand. "We are experiencing an innovation renaissance that is largely driven by open source software that powers distributed, scale-out systems," Zemlin says. Companies that actively participate in the open source community are on the cutting edge of that renaissance. Companies that ignore this trend will be left behind.

The Linux Foundation promotes, protects and standardizes Linux by providing unified resources and services needed for open source to successfully compete with closed platforms.

To learn more about The Linux Foundation or our other initiatives please visit us at www.linuxfoundation.org