FOSSA

TL;DR

Understanding the different types of open source licenses is only the beginning. You also have to manage all of the open source software used by your company.

The management of your open source software can be divided into six basic areas, with responsibilities shared across engineering, legal and DevOps:

- **Open Source Management Basics**
- **Open Source Management Strategy**
- **Open Source Management Program Implementation**
- **Open Source Policy**
- **Open Source Processes**
- **Tools for Open Source Management**

These focus areas impact open source license compliance, security vulnerabilities, code quality, and risk management while your organizations scales its usage of open source.

The implementation of an automated Open Source Management Platform can streamline the management of your open source and give you a competitive advantage.

OS Management Fundamentals for Engineering, Legal and DevOps

Open Source Management

Open source is no longer the plucky underdog. It's the dominant force in the Enterprise market.

Just half a decade ago, open source companies were Rocky to big-name proprietary software's Apollo Creed. Sure, Red Hat Linux was a big name, but it was an outlier due to its crucial part in many enterprise's infrastructures.

Fast forward to today and you find that IBM acquired Red Hat for $32 billion. Other open source companies have sprung up and are becoming the software upon which enterprises depend.

Do you want to run microservices in a fleet of containers?
You need to use Docker.

Do you need quick-access storage? You'll be using MongoDB.

Is Java the language of choice for your business applications?
Then your entire software stack depends on open source tools such as Spring, Apache, and Jenkins.

The increased dependence on open source software opens up a new responsibility—you have to keep track of all of the software you're using.

How Do You Manage Your Open Source Inventory?

This guide dives into the fundamentals of open source management, so you can reduce risk and maximize efficiency. We'll divide open source management into six essential components:

- Open Source Management Basics
- Open Source Management Strategy
- Open Source Management Program Implementation
- Open Source Policy
- Open Source Processes
- Tools for Open Source Management

An even more impressive shift is in the mindset of the enterprise. They no longer begrudgingly accept open source as a necessary evil. A recent survey of IT leaders by Red Hat shows that 69% feel that open source is a "very" or "extremely" important part of their IT infrastructure software strategy.

Red Hat's survey shows that enterprises are using open source solutions to drive innovation and give themselves a competitive advantage. You'd be hard-pressed to find successful companies that aren't using open source in some capacity.

The increased dependence on open source software opens up a new responsibility—you have to keep track of all of the software you're using.

Why? Because open source software has licenses attached to them. There are many different licenses, each with different terms and permissions. Some allow you to change and redistribute software at will, while others expect you to follow specific guidelines or adhere to certain obligations. Open source is often free, but there are still responsibilities associated with its use.

UNDERSTANDING THE DIFFERENT TYPES OF OPEN SOURCE LICENSES IS ONLY THE BEGINNING. YOU NOW HAVE TO MANAGE ALL OF THE OPEN SOURCE SOFTWARE USED BY YOUR COMPANY.

OPEN SOURCE SOFTWARE AND OPEN SOURCE MANAGEMENT BASICS

Managing open source software licenses is equally as crucial as managing proprietary software licenses. If the great Oracle/Google battle over the last decade has taught us anything, it's that companies that own or license open source software have requirements you need to meet.

In May of 2017, a federal court ruled that an open source license is an enforceable contract. There have been several lawsuits brought against companies who violate open source licenses.

Open source management is a severe concern for enterprises that depend on it.

To begin the journey of open source management, let's quickly review what kinds of open source licenses you can expect to see in the software you use.

FOSSA

Common Types of Open Source Licenses

Copyleft licenses allow you to change the software for your purposes and distribute your modified copy. However, copyleft licenses require you to "pay it forward" and release all of your code as open source under the same license.

Arguably the most famous copyleft license is the GNU General Public License or GPL. There are three versions of the GPL, and all three are copyleft licenses.

Permissive licenses, as you may guess by the title, allow you to do pretty much whatever you want with the software. Examples of permissive licenses are the Apache 2.0 and MIT licenses. They require you to include the license and the copyright statements, and Apache requires you to state significant changes made. Otherwise, you're free to change and redistribute to your heart's content.

Understanding the different types of open source licenses is only the beginning. You now have to manage all of the open source software used by your company. There are several challenges to effective management.

Challenges With Open Source Management

The complexity of enterprise environments leads to several problems in managing open source software.

Licensing Compliance

First, licensing compliance is challenging to maintain. Non-compliance is a real threat to your company, not only financially, but also reputationally. Open source companies aren't content to sit on the sidelines and hope no one is violating their licenses. The GNU project actively solicits information about those who violate their license.

Licensing compliance is challenging for three main reasons:

- Understanding the requirements of every license attached to open source code you use or plan to use is difficult for a full-time lawyer. Imagine the difficulty for a developer or business manager who wants to use the software.
- Companies often embed open source code within other applications. Open source libraries often depend on other open source libraries. A web of open source license dependencies is the result, creating major headaches for those tasked with managing them.
- Those who are likely to use or choose open source software are not fully equipped to interpret the licenses and ensure compliance.

THERE ARE TWO FLAVORS OF OPEN SOURCE LICENSES: COPYLEFT AND PERMISSIVE. THERE ARE COUNTLESS VARIATIONS IN THE DETAILS, BUT MOST OPEN SOURCE SOFTWARE WILL USE THESE TWO TYPES.

OPEN SOURCE MANAGEMENT CHALLENGES

- ✔ Licensing Compliance
- ✔ Security Vulnerabilities
- ✔ Code Quality
- ✔ Scale
- ✔ Scheduling
- ✔ Programming Language Diversity

FOSSA

Security Vulnerabilities

Second, security vulnerabilities within open source software are difficult to track and manage, primarily if used within a broader application. As your dependence on open source grows, your ability to stay on top of vulnerabilities, and how they impact your codebase (let alone fixing them) becomes a monumental task.

Code Quality

Third, code quality may begin to suffer if you depend on open source software that isn't maintained well. Developers want to use third-party code to solve problems because they don't have the time to solve them on their own. However, it's challenging to make sure that the open source code used matches the standards of your organization.

Scale

The fourth challenge is scale. Your use of open source will eventually grow to the point where you want to keep track of what you have. And you'll likely start by tracking things manually. But manual tracking of open source doesn't scale well. It won't be long before the time used to keep spreadsheets up to date will significantly and negatively impact the time spent developing critical business software. The problem with scale "scales" very well.

The larger your company, the larger the headache. Multiply your manual tracking issues by the number of divisions, projects, or business units independently using open source software. You'll need an aspirin before long.

Scheduling

Scheduling is another challenge because open source projects follow their development roadmaps and delivery schedules. It's challenging to keep your software schedules in sync with the various projects you depend on to run it. This challenge often leads to a feeling of continually playing catch-up with new versions of software and security vulnerabilities.

Programming Language Diversity

Finally, programming language diversity among open source projects can increase the risk of using them. Software written in programming languages your developers aren't familiar with makes it difficult to ensure compliance with code and security standards. Once you understand the challenges you're likely to face, you can begin building a strategy for managing open source software.

FOSSA

OPEN SOURCE MANAGEMENT STRATEGY

Don't give up on using open source because the above challenges seem daunting. A clear management strategy will make day-to-day execution much more straightforward. Once you have a plan in place, you can manage these challenges and overcome them.

There are five fundamental goals of practical open source management:

- **Know where all of your open source code is within your environment**
- **Avoid licensing compliance problems**
- **Don't waste your developers' time manually tracking open source software**
- **React quickly to security vulnerabilities found within open source components**
- **Align the code quality of open source libraries with your standards.**

All strategies should do their best to address these goals. The best practices we've discovered revolve around four key areas — all of which represent a "shift left" in the software development lifecycle to handle any challenges early and effectively:

- Automation
- Real-time auditing and tracking
- Integration with the SDLC (Software Development Lifecycle)
- Collaboration between teams

Let's explore these pieces in more detail so you'll know how to implement them in your business.

FOSSA

AUTOMATION IS THE ONLY RELIABLE WAY TO TRACK YOUR DEPENDENCIES.

OPEN SOURCE MANAGEMENT PROGRAM IMPLEMENTATION

Our first concern is automation. You'll need to find a way to automate the discovery and capture of the open source software you depend on to run your systems.

You may have already started tracking open source dependencies using spreadsheets. Perhaps your developers are helping you to identify the software they use. Unfortunately, this process is not ideal and won't work for long.

Manual open source tracking has two drawbacks. First, it takes away from your developers' limited time. Your software engineers should be building valuable business software, not digging through dependency trees looking for every version of open source software they use. They won't likely be able to support the effort and get their everyday work done. Delays will begin to pile up, causing tension between the legal and development groups.

Second, manually tracking dependencies doesn't scale well. If you're only using ten libraries, then a spreadsheet may work. But what happens when you're using 100, 1,000, 5,000? It'll soon become impossible to keep track of everything you're using, especially if developers can choose open source programs to use (more on that later).

Automation is the only reliable way to track your dependencies. Machines are much more suited to digging into every nook and cranny to retrieve absolutely everything you're using. The reporting and storage will be consistent. And you can still build in escalation paths where humans need to take a look.

Integration with the SDLC (Software Development Lifecycle)

We live in a world of DevOps and agile development practices. Software is constantly changing, and open source software is no exception. As more downstream processes, such as functional and security testing, are being "shifted left" in the development cycle, so should open source management.

There are two sides to the integration coin. Real-time auditing and tracking capabilities allow you to review every code change for compliance. Every build should have its dependencies scanned and checked for new security vulnerabilities automatically. Then the proper teams can be notified, and remediation can begin.

Integration with the SDLC also means playing nicely with the tools your development teams are already using. Significant problems should appear in their development environments. Enter escalations into project management software, such as Jira or Asana, where your development teams can respond and prioritize amid their current work. Make your open source management visible.

FOSSA

Collaboration Is Necessary

Your legal department will be concerned with which licenses are present and whether you're keeping in compliance with them.

Developers will be trying to find the easiest-to-use libraries to help speed up development.

The C-Suite will want to know how to manage the risks of open source software.

Security teams will be worried about security vulnerabilities found in third-party code.

But all will benefit from the competitive advantage open source can bring. So it's essential to maintain excellent communication between all of these groups. Align your goals and find common ground. Use software where possible to reduce friction.

Keep focused on the goal, the successful and low-risk use of open source software, so that all are on the same page.

OPEN SOURCE TOUCHES ON SEVERAL DISCIPLINES WITHIN YOUR COMPANY

- ✔ **Legal Department**
- ✔ **Developers**
- ✔ **C-Suite**
- ✔ **Security**

FOSSA

OPEN SOURCE POLICY

Policies define the "rules of engagement" for a given endeavor. Companies of all sizes benefit from guidelines on how the development team should use open source software. Without clear policies in place, your risk in using open source will blow up over time.

What are some smart open source policies to think about if you don't have them yet?

- Define the open source licenses that are acceptable for use
- All open source libraries must be approved before use, or...
- Guidelines on which libraries can be used without approval or with a simple tap on the shoulder to legal and compliance.
- Is it okay to use more than one open source library that performs the same function?
- Do libraries have to be clear of vulnerabilities before they are approved?
- Can your developers contribute back to the open source projects? If so, how?
- What level of support is acceptable for use (as in, how often the software is updated)?
- Do your developers fix bugs or wait for the project supporters?
- How often should your policies be reviewed?

These are some questions to get you started, and there could be many more. Only you understand your unique risk posture and exposure through open source. You'll have to decide over time which policies make sense for you.

And once you have your policies in place, an automated tool that catches open source policy violations in real-time will reduce your risk and save you loads of time.

FOSSA

OPEN SOURCE PROCESSES

Your policies will give birth to the processes necessary to carry them out. An automated tool can help with this part, but some procedures will require steps that a human must take.

An approval process for new open source libraries will help you to reduce the risk of a developer introducing a dangerous piece of software into the ecosystem. But what does such a process look like?

First, you'll want to review open source libraries for architectural compatibility. If you're creating a website front end using Angular, you may not want to bring in an open source library written using ReactJS. Make sure that languages and code quality match your standards.

You'll next want to make sure that no other software you're currently using serves the same purpose as the proposed open source library. Developers want to get their code completed as quickly as possible and may balk at having to explain why they need something. But there shouldn't be 20 different open source calendar libraries in use when only one or two are required.

An inventory of all of your software assets is essential in this process. Your developers should be able to check ahead of time if there is a component available to them to serve their purpose. This process reduces friction and time wasted in a manual review.

A formal review will allow all interested parties to weigh in and decide whether a piece of software will provide value. Legal can check for license compatibility and support options. Security can make sure there aren't glaring security holes present. Fellow developers or software architects can make sure the library will work within the environment.

TOOLS FOR OPEN SOURCE MANAGEMENT

A transparent process is needed when an escalation to a human is required. Automated tools can help here since different people will need to be contacted depending on the nature of the problem.

If a critical open source security vulnerability crops up, a system should notify both the developers and the security team. Policies will tell them whether to fix the software themselves or wait for a fix from the project supporters.

An open source license violation may also be present. Legal should know of any licenses that are causing issues for the development team. Developers need to understand how to resolve license issues without contacting the legal department every time.

Another possible escalation is a violation of your open source policies themselves. Perhaps a developer begins to use an open source library that hasn't been reviewed or is out of compliance in some other way.

Let the developer know why they can't use it and how to get the software approved.

The process of escalation should not be feared. We're not talking about bringing developers before the "Legal Inquisition" to be punished. The right people should be contacted at the correct times to resolve the issue to everyone's benefit.

Open Source Inventory Management: Essential And Possible

Since we've covered quite a few guidelines here, save (and share) this content for when you're ready to implement different pieces of open source management. In the meantime, what next steps can you take?

First, review your current open source management practices. Are you using spreadsheets to track libraries? Who is doing it? How much time are they taking to keep up with the demand?

Then decide on a few starting policies. Review some of the questions we raised above and a few of your own. You don't have to have everything nailed down right now. Just get started.

Then check out an automated open source management platform like FOSSA to help take some of the load off of your shoulders. Once you've made the decisions, FOSSA can implement and enforce them for you.

Using open source software gives you a competitive advantage. Harness that advantage wisely with proper open source management practices.

ABOUT FOSSA:

FOSSA is the world's first Modern Open Source Management platform. Designed for development and legal teams alike, FOSSA provides component intelligence, continuous compliance, and cross-team collaboration solutions that enable engineering excellence and accelerate market capture while mitigating business risk.

FOSSA.com | **Sign up with Github** | **tldrlegal.com**

FOSSA, Inc. | Modern Open Source Management
114 Sansome Street, Suite #201, San Francisco, CA 94104