

Corporate Open Source Governance

Prof. Dr. Dirk Riehle

Friedrich-Alexander University Erlangen-Nürnberg

FOSS B03

Licensed under CC BY 4.0 International

Governance and Compliance

- Organizational **governance**
 - Is “the way the rules, norms and [desired] actions [of an organization] are produced, sustained, regulated and held accountable” (Wikipedia)
- Organizational **compliance**
 - Is how any behavior of and within an organization is made to comply with that organization's rules, norms, and desired actions [DR]
- Software-using and producing organizations
 - Can have all kinds of governance and compliance bodies
 - One form is **open source governance and compliance**

Open Source Governance and Compliance

- **Open source governance and compliance**
 - Is the way an organization manages its open source engagement
 - License compliance is different from organizational compliance
- The three key forms of open source engagement
 - **Use** (of open source software in products)
 - **Contribute** (to open source software projects)
 - **Create or lead** (open source software projects)
- In the following, governance includes compliance

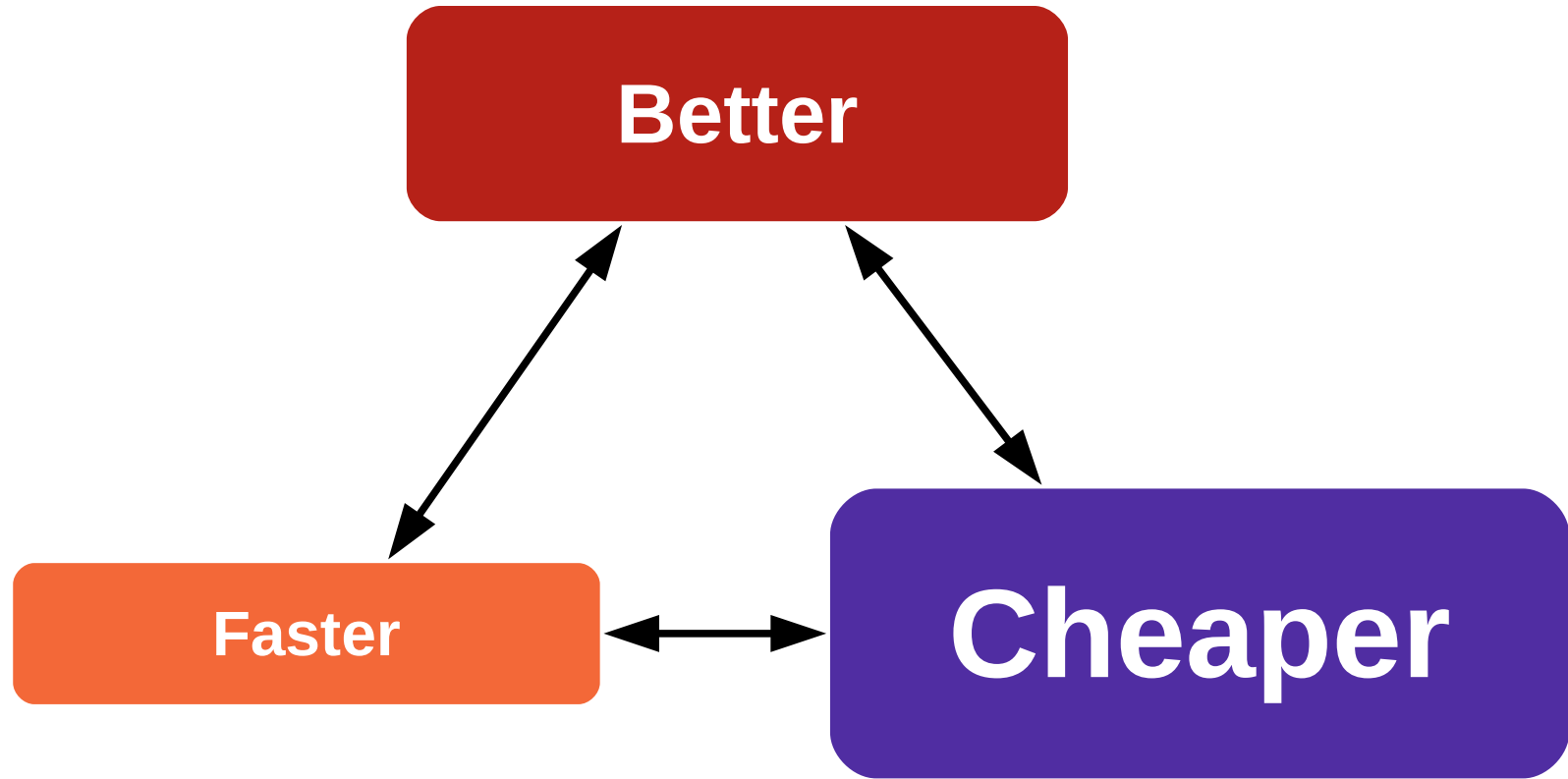
Products and Services

- Services, e.g. software-as-a-service (SaaS) are a product
 - Service provider operates product for customers, no on-premise use
 - Usually no upfront license fee, but maintenance fee (subscription)
- Open source licenses still apply!
 - The provision of the software as a service constitutes “distribution”
 - Specifically the AGPLv3 was conceived for this case
- **Product** from now on means both trad. software and SaaS

Primary Benefits of Using Open Source

- **Better**
 - Open source components can be of high quality
- **Faster**
 - Open source components are immediately available
- **Cheaper**
 - Open source components are free (no license fee)

Initial Motivation: Save Costs

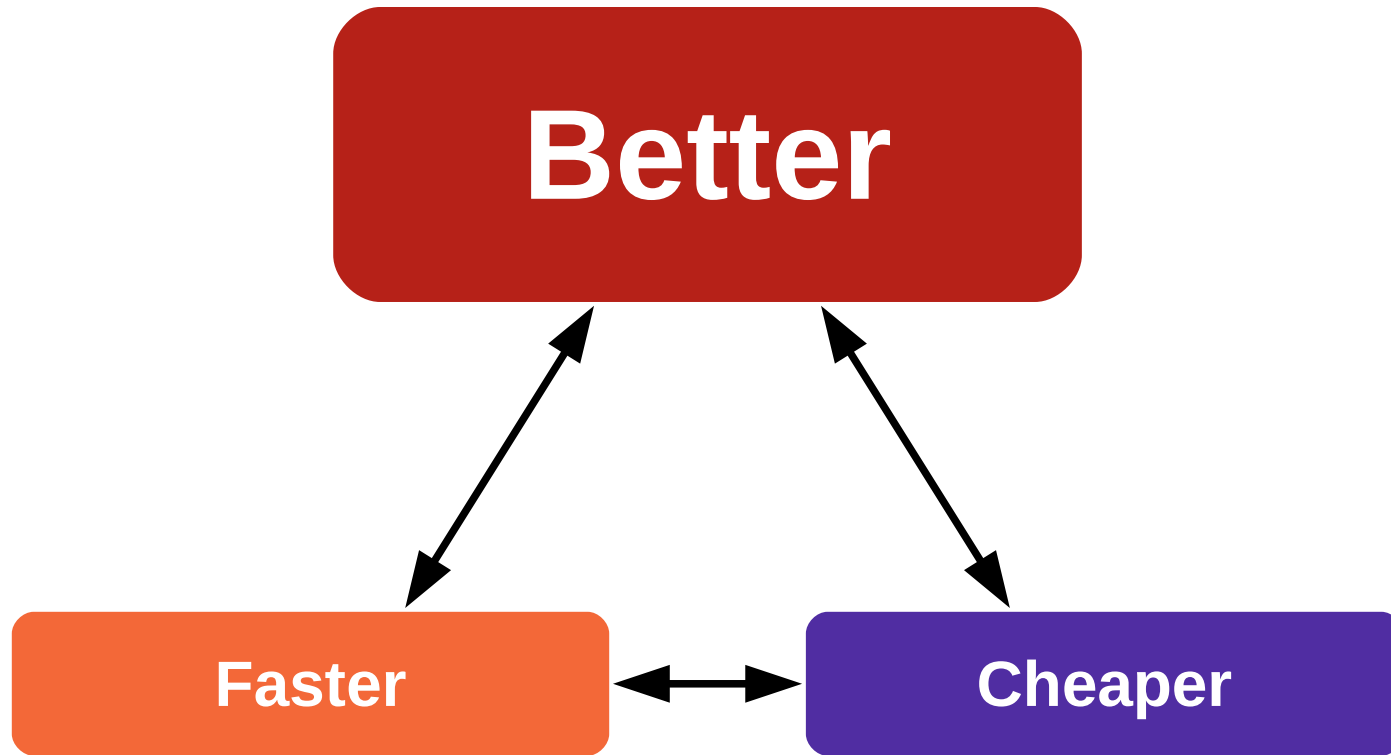


“High quality software free.”

Secondary Benefits

- Open source is open to inspect, modify
 - Faster: Users can help themselves, fix bugs
 - Faster: Users can extend the software, develop new features
- Most open source has no or little vendor lock-in
 - Faster: Innovation cannot be blocked by one company
 - Cheaper: Competition keeps service prices low
- Open source components are compatible
 - Faster, better: With standards (as reference implementations)
 - Faster, better: With platforms (as de-facto implementations)

Mature Motivation: Focus Business



“Focus on competitive differentiation.”

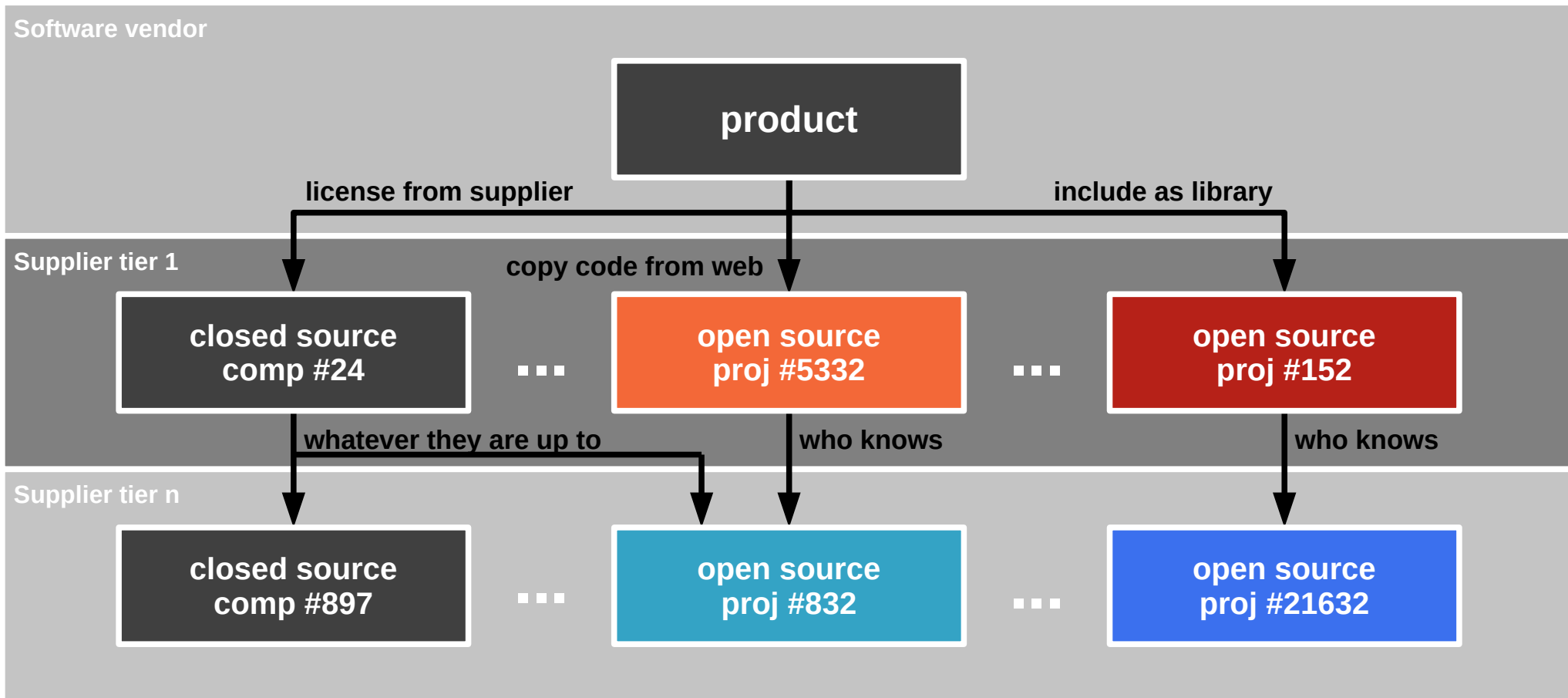
Detriments of Using Open Source

- Attackers have it a bit easier to find security holes
 - Various agencies scan open source for vulnerabilities, don't report them
 - Becomes more serious, if use of open source is signaled

Not Using Open Source is Not an Option

- “We don’t use open source” signals ...
 - They may not know what their engineers are doing
 - They may not have their business focus straight

The Software Supply Chain



The Open Source Program Office

- Open source goal and strategy formulation
 - Should be led by VP of Engineering and CTO
 - Typically leads to the creation of a program office
 - Should be done together with the program office
- Purpose of an open source program office
 - Serves as a point-of-contact
 - Manages governance processes
 - Manages governance itself
- Also sometimes called a competence center

1. **Achieving goals while**
2. **Avoiding problems**
3. **At minimal cost**

Achieving Goals for Using Open Source

- **Focus business**
 - Identification of open source opportunity
- **Save costs**
 - Selection of open source component
 - Inclusion of component in product
 - Management of open source investment

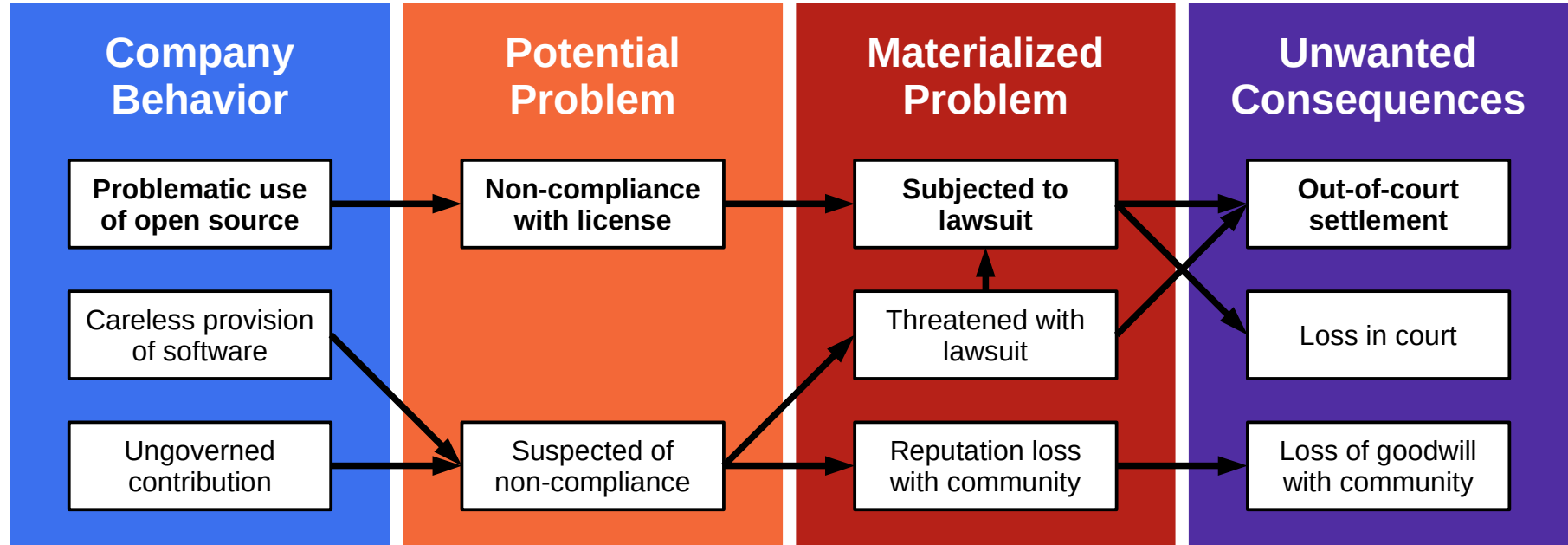
Example Governance for Achieving Goals

- **Identify open source opportunities**
 - Product manager determines business focus and engineering manager identifies correlating open source opportunities
- **Select open source component**
 - Engineering manager and software architect define criteria for open source component, select best fitting one; includes **clearance process**
- **Include open source component**
 - Software architect and software developer include open source component in product and bill-of-materials
- **Manage open source investment**
 - Engineering manager, software architect, and software developer monitor the evolution of the open source component, engage when necessary

Avoiding Problems with Using Open Source

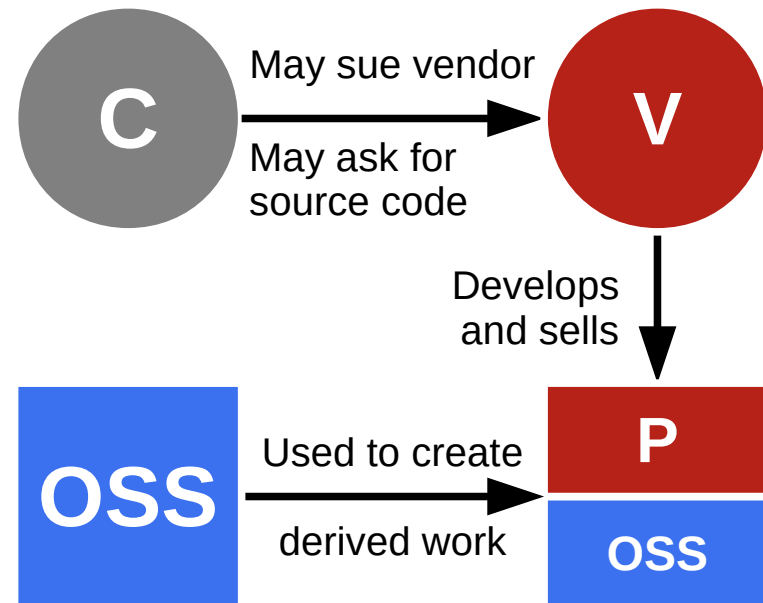
- Example of **Legal challenges**
 - Ensure usage rights as intended (license compliance)
- Example of **Quality challenges**
 - Ensure new security holes are recognized and fixed
- Example of **Process challenges**
 - Avoid maintenance burden by contributing back

Cause and Effect Chain of Legal Problems



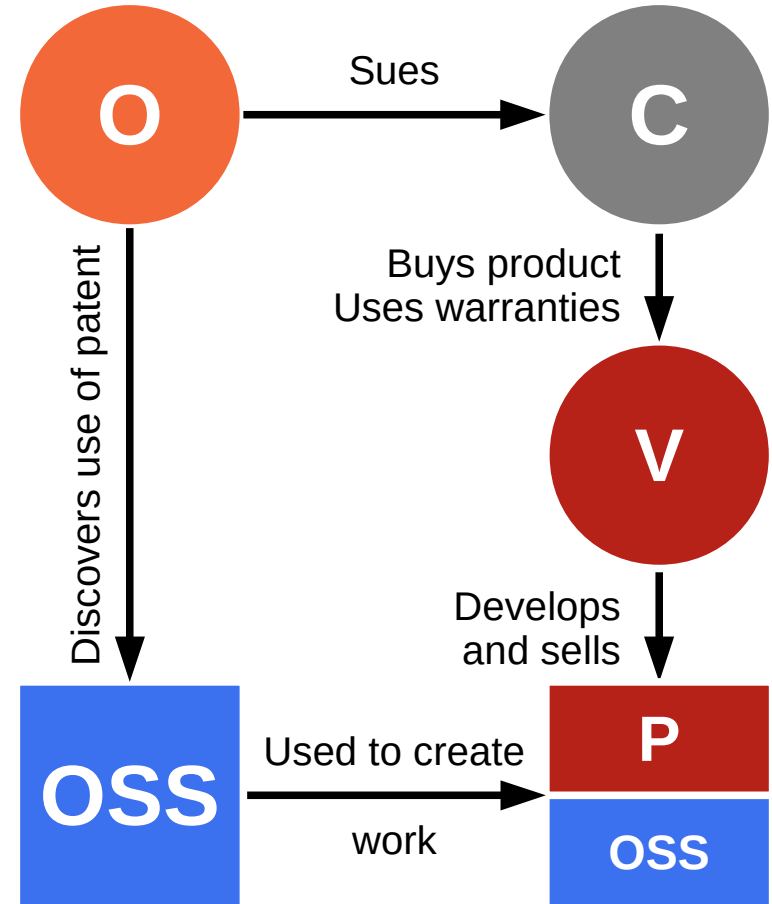
Copyright Violation Lawsuit

- Steps in process
 - Customer C acquires license for commercial product P from software vendor V
 - Customer asks software vendor for source code; in all likelihood gets denied
 - Depending on license and technical coupling, this denial is a license violation



Patent Infringement Lawsuit

- Setup of situation
 - OSS contains code that implements patent owned by patent holder O
 - Software vendor sells license for P to customer C without licensing patent
- Steps in process
 1. Patent holder sues customer (patent user)
 2. Customer turns around, sues software vendor
- Applies broadly
 - To both use-cases
 - In-house
 - In products



Costs of Materialized Problems

- When threatened with lawsuit
 - Legal costs
 - Reputation loss
 - Wasted management attention
- When subjected to lawsuit, like before +
 - **Undesired revealing of information**

Costs of Unwanted Consequences

- When settling out-of-court, like before +
 - Financial costs
 - Compliance costs
- When losing in court, like before +
 - Financial costs
 - License and other fees
 - Potential punitive damages
 - **Product revenue loss due to ...**
 - **Product recall / sales stop**
 - **Delay of product releases**
 - Compliance costs
 - Increased risks of more lawsuits

Costs of Establishing License Compliance

- In case of component replacement
 - Development costs (rework)
 - New component licensing fees
- In case of non-replacement
 - Labor costs (compliance artifacts)
 - Intellectual property loss
 - Exclusivity of source code
 - Potential free patent rights grant
 - Undesired revealing of information

Probability of Getting Sued

- Sources of lawsuits
 - GPL enforcers
 - GPL monetizers

[About](#)[News](#)[Blog](#)[Membership](#)[Sponsors](#)[Copyleft Compliance](#)[NPOAcct](#)[Donate](#)[Become a Supporter!](#)

Conservancy will continue our basic community services, thanks to our first 750 Supporters! We still need 809 more Supporters to avoid reducing licensing work & hibernating our enforcement efforts in 2017.

1.031 have joined!

809 more needed to save license compliance work.

660 matched!

August 9, 2016

Hellwig Announces He Will Appeal VMware Ruling After Evidentiary Set Back in Lower Court

In a [statement on his website](#), Christoph Hellwig announced today that he will appeal the ruling of the Hamburg District Court, which recently dismissed his case against VMware. As Christoph underscores in his statement, the ruling concerned German evidence law and the Court did not rule on the merits of the case. The ruling centered around German evidentiary rules related to documenting Christoph's contributions that appear in VMware's product. Christoph also published (in [German](#) and [English](#)) the [Court's ruling](#) which explains why the materials submitted did not satisfy German evidence rules — despite publicly available information in Linux's Git repositories. In addition, the Court chose not to seek expert testimony.

Christoph stated on his website, "I'm disappointed that the court didn't even consider the actual case of reusing the Linux code written by me, and I hope the Court of Appeal will investigate this central aspect of the lawsuit."

Conservancy publishes today its [comparison analysis between Christoph's code and VMware's code](#). This particular analysis uses a two step process: (a) use Linux's public Git logs to find Christoph's contributions from Christoph, and (b) use a widely accepted and heavily academically cited tool, CCFinderX, to show that VMware copied Christoph's code into their product.

While these evidentiary points may be new to the German courts, they have been explored in US Federal Court. Conservancy previously successfully litigated as co-plaintiff with Erik Andersen over BusyBox. Many companies who settled, and the [US Federal Court in their judgment against Westinghouse](#), ultimately accepted and agreed that Erik Andersen held copyrights in BusyBox.

The German civil legal system is not precedent-based. As such, this initial ruling creates no legally binding precedent. Our community continues our long journey to build definitive industry precedent regarding [derivative and combined works under the GPL](#).

GPL Monetization and Netfilter [KS16]

Specifically, we remain aware of multiple non-community-oriented GPL enforcement efforts [...] These “**GPL monetizers**”, who trace their roots to nefarious business models that seek to catch users in minor violations in order to **sell an alternative proprietary license**, stand in stark contrast to the work that Conservancy, FSF and gpl-violations.org have done for years.

Most notably, a Linux developer named **Patrick McHardy continues ongoing GPL enforcement actions** [...] In the last two years, we've heard repeated rumors about Patrick's enforcement activity, as well as some reliable claims by GPL violators that Patrick failed to follow the Principles.

Patrick's enforcement occurs primarily in Germany. We know well the difficulties of working transparently in that particular legal system, but both gpl-violations.org and Conservancy have done transparent enforcement in that jurisdiction and others.

Patrick McHardy has also been suspended from work on the Netfilter core team. While the Netfilter team itself publicly endorsed these Principles of enforcement, Patrick has not. Conservancy agrees that Patrick's apparent refusal to endorse the Principles leaves suspicion and concern, since the Principles have been endorsed by so many other Linux copyright holders, including Conservancy.

The CEO's Liability

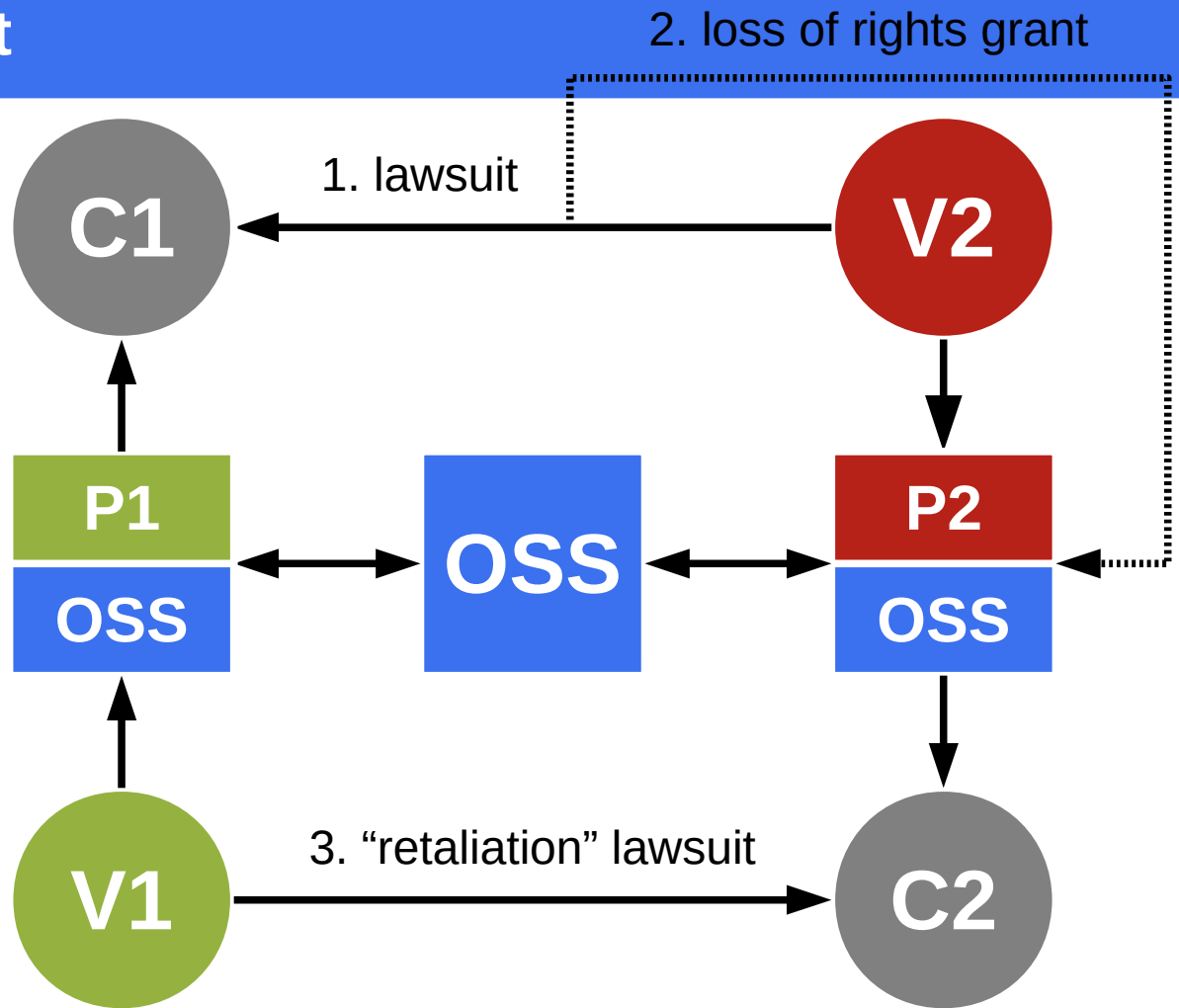
- Good open source governance is “state of the art”
 - Many large companies demonstrate how to do it
 - Therefore, good governance is responsibility of the CEO
- Leads to liability of the CEO resp. “Geschäftsführerhaftung”

Proactive Measures against Patent Lawsuits

- Open Invention Network (OIN) founded to protect Linux
 - OIN owns a large patent pool that it licenses out royalty-free
 - Requires that licensees agree not to enforce patents against Linux
- Modern open source licenses include patent retaliation clauses
 - Users of open source, who enforce patents, lose right to use open source
 - Retaliation clauses are not present in older licenses, e.g. not in GPLv2

Patent Retaliation Lawsuit

- Setup of situation
 - Both V1 and V2 contribute to OSS
- Original lawsuit
 - V2 sues C1 for patent **patent infringement**
 - C1 turns around, holds supplier V1 responsible
- Retaliation lawsuit
 - V1 sues C2 for **copyright violation**
 - C2 turns around, holds supplier V2 responsible



No risk

No reward

How to manage this risk?

Example Governance for Legal Problems

- **Developer education**
 - Educate developers, refresh education, provide handbook, ...
- **Open source clearance process**
 - Have clearance process, define point-of-contact, provide veto rights, ...
- **Supplier management**
 - Manage suppliers, define acceptable licenses, require bill-of-materials, ...

Component Analysis 1 / 2

[Home](#)[Search](#)[Browse](#)[Upload](#)[Organize](#)[Jobs](#)[Admin](#)[Obsolete](#)[Help](#)

Browse

User: demo

[logout](#)

Folder Navigation	Uploads						
Expand Collapse Refresh + Software Repository + Examples	<table><thead><tr><th data-bbox="534 534 1661 579">Upload Name and Description</th><th data-bbox="1661 534 1951 579">Upload Date</th></tr></thead><tbody><tr><td data-bbox="534 579 1661 819">gpl-3.0.txt Added by file upload: gpl-3.0.txt [View] [Info] [Download] <i>No description</i> Scheduled jobs: 3 total; 3 completed; 0 failed.</td><td data-bbox="1661 579 1951 819">2010-07-08 13:20:07</td></tr><tr><td data-bbox="534 819 1661 1046">simpletest_1.0.1.tar.gz Added from filesystem: /home/fosstester/archives/simpletest_1.0.1.tar.gz [View] [Info] [Download] SimpleTest demo upload from server Scheduled jobs: 4 total; 4 completed; 0 failed.</td><td data-bbox="1661 819 1951 1046">2010-07-12 16:15:56</td></tr></tbody></table>	Upload Name and Description	Upload Date	gpl-3.0.txt Added by file upload: gpl-3.0.txt [View] [Info] [Download] <i>No description</i> Scheduled jobs : 3 total; 3 completed; 0 failed.	2010-07-08 13:20:07	simpletest_1.0.1.tar.gz Added from filesystem: /home/fosstester/archives/simpletest_1.0.1.tar.gz [View] [Info] [Download] SimpleTest demo upload from server Scheduled jobs : 4 total; 4 completed; 0 failed.	2010-07-12 16:15:56
Upload Name and Description	Upload Date						
gpl-3.0.txt Added by file upload: gpl-3.0.txt [View] [Info] [Download] <i>No description</i> Scheduled jobs : 3 total; 3 completed; 0 failed.	2010-07-08 13:20:07						
simpletest_1.0.1.tar.gz Added from filesystem: /home/fosstester/archives/simpletest_1.0.1.tar.gz [View] [Info] [Download] SimpleTest demo upload from server Scheduled jobs : 4 total; 4 completed; 0 failed.	2010-07-12 16:15:56						

Component Analysis 2 / 2

Folder: **Software Repository/**
Test.zip/ Test

License Browser | Bucket

2 files found (2 unique) with license **Affero_v3**
1 2 ... [Next]

Count	Files	License Name
2749	Show	No_license_found
17	Show	Trademark-ref
4	Show	Apache_v2.0
2	Show	Affero_v3
2	Show	MIT-style
1	Show	Adobe
1	Show	Adobe-AFM
1	Show	Indemnity
1	Show	Public-domain
1	Show	See-doc(OTHER)
1	Show	SGI

File

Exclude this file type.

1: Folder: **Software Repository/**
Test.zip/ Test/ lib/
iText-5.0.1.jar/ com/ itextpdf/ text/ AGPL.txt

Affero_v3 ,Public-domain

Exclude this file type.

2: Folder: **Software Repository/**
Test.zip/ Test/ lib/
iText-5.0.1.jar/ com/ itextpdf/ text/ LICENSE.txt

Affero_v3

No_license_found

lib/

Adobe ,Adobe-AFM ,Affero_v3
,Apache_v2.0 ,Indemnity ,MIT-style
,No_license_found ,Public-domain
,See-doc(OTHER) ,SGI ,Trademark-ref

resources/

No_license_found

src/

No_license_found

Hint: Click on the license name to **highlight** where the license is found in the file listing.

Source Code Scanner 1 / 2

The screenshot displays the Source Code Scanner interface. At the top, there are tabs for 'My Protex', 'Manage', 'Identify' (selected), 'Review', and 'Report'. The 'Current Project' is 'Tutorial_Files'. Below this, the 'Project Status' is shown with colored squares and counts: 12 (yellow), 40 (blue), 1 (green), 0 (red), 23 (orange), 0 (white), and 2 (grey).

The left sidebar shows a tree view of the project structure under 'Tutorial_Files (12)'. It includes folders like 'lib (2)', 'licenses (3)', 'src_io', 'src_surfaces (1)', 'src_ggit', and 'tools (4)'. The 'tools' folder is expanded, showing files like 'bftest.c', 'clean-sat.c', 'fileone.c', 'filetwo.c', 'gpgsplit.c', 'mk-tdata.c', and 'mpicalc.c'.

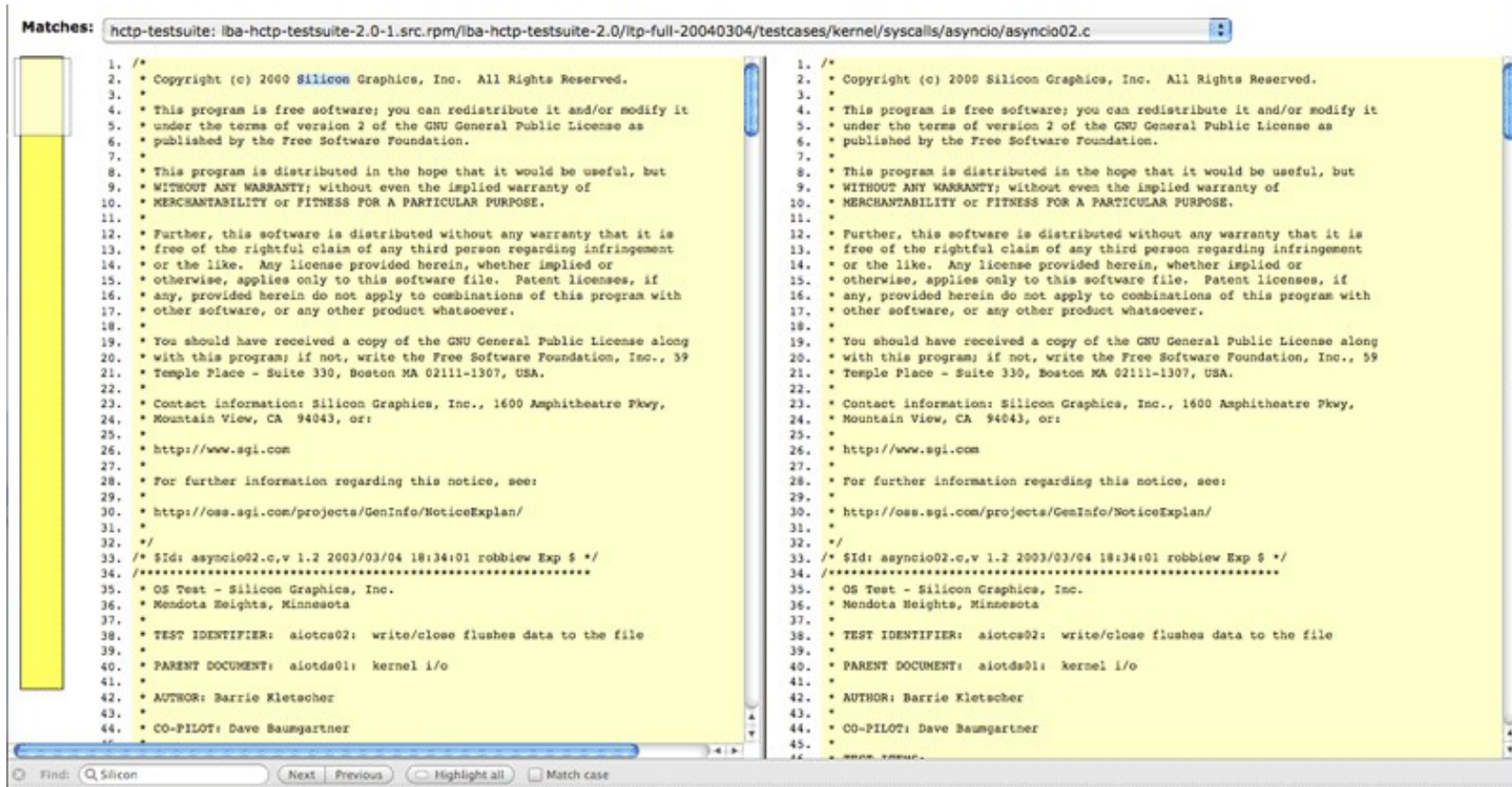
The main area has tabs for 'Bill of Materials', 'Code Matches' (selected), 'Searches', and 'Dependencies'. The 'Files' section shows '/Tutorial_Files/tools/mpicalc.c'. A search bar is present with 'Go' and 'Clear' buttons. The 'Show' dropdown is set to 'Precision'.

A table titled '3 Components (10 per page)' displays the following data:

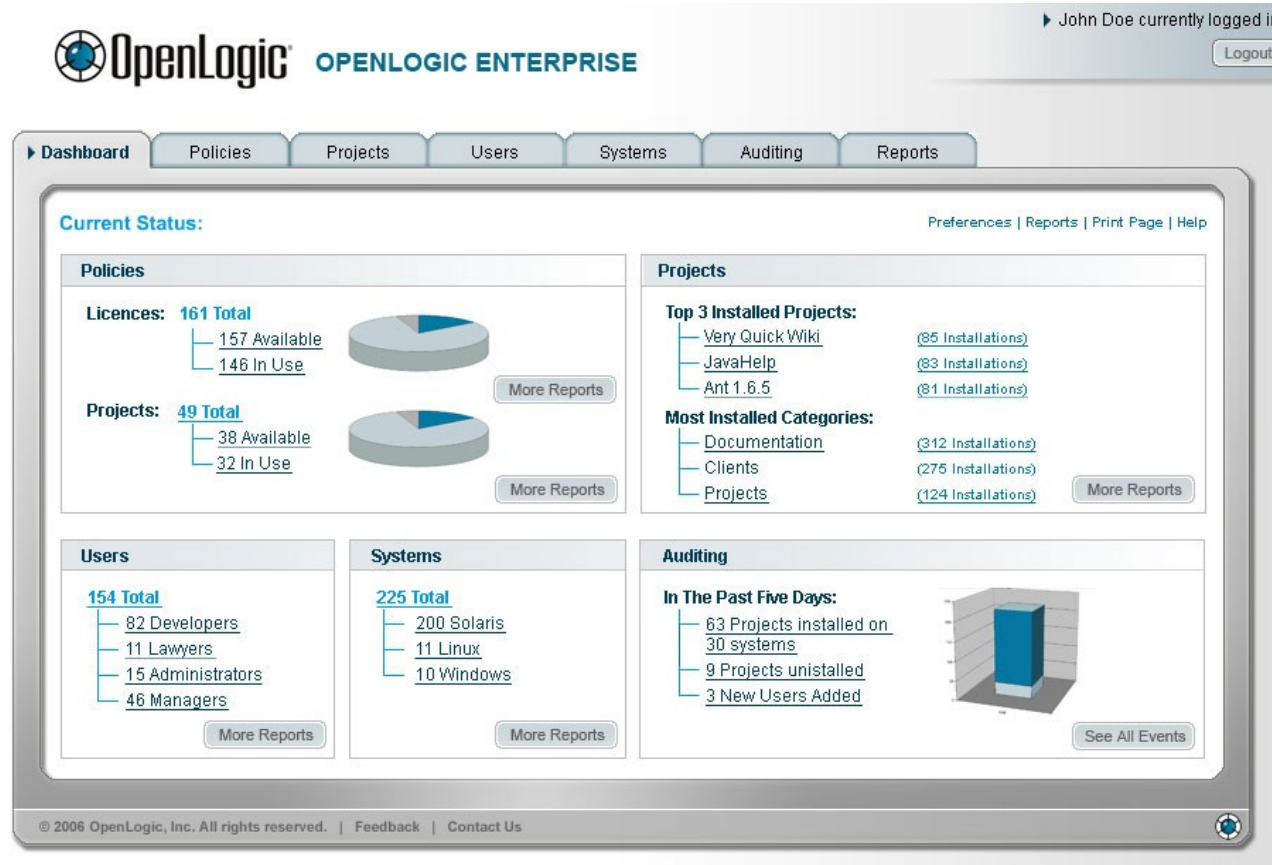
ID	Approved	Component Type	Component	Version	License	Status	%	Matched File	Line
		10	Indian Firewall Community	Unspecified	GPL 2.0	File	Precision Match 100	EFW-COMMUNITY-2-devel-SRPMs.tar.gz/EFW-COMMUNITY-2-devel-SRPMs/communit.2.2/tools/mpicalc.c	0

Below the table, there are two side-by-side code comparison windows. The left window, titled 'Your File: mpicalc.c', shows the source code of the file being scanned. The right window, titled 'Matched File: mpicalc.c', shows the source code of the matched file. Both windows display C code with comments and function definitions.

Source Code Scanner 2 / 2



Repository Management 1 / 2



Repository Management 2 / 2

The screenshot displays the Black Duck Protex web interface. The top navigation bar includes 'My Protex', 'Manage', 'Identify' (selected), 'Review', and 'Report'. The current project is 'MS+4.0'. The left sidebar shows a project tree with folders like 'bin', 'Archive', 'J2EE Application', 'lib', 'src', 'Unix Application', 'Windows Application', 'compressed', 'dep', 'multi-server', 'database', 'include', 'memory', 'persistence', 'plp', 'server', 'service', 'session', 'utils', 'project', 'main.c', 'ss', 'a0100131.gfm', 'Apache1.1 License.txt', and 'GPL2.0 License.txt'. The main content area shows the 'Identify' tab for the folder '/MS+4.0/multi-server/utils'. It displays a table of 217 components, with one component highlighted in yellow: 'Endian Firewall Community' version '2.2 RC2' with license 'GPL 2.0'. Below the table, the 'Identify' section shows the selected component details and options to 'IDENTIFY', 'EXCLUDE', or 'RESET' the component.

Black Duck Protex

My Protex Manage Identify Review Report

Current Project: MS+4.0

Project Status: 21 18 10 8 9 14

newcomer@blackducksoftware.com

BETA Report an Issue Tools Help Logout

Server: preview.blackducksoftware.com

Show: Pending Identification - ALL

New Filter...

Expand Collapse

Only Show Matches

MS+4.0 (22)

- bin (8)
- Archive (1)
- J2EE Application (2)
- lib (2)
- log4j-1.2.14.jar
- log4j.jar
- src
- Unix Application (2)
- Windows Application (3)
- compressed (1)
- dep
- multi-server (3)
- database
- include
- memory
- persistence
- plp
- server
- service
- session
- utils (3)
- project
- main.c
- ss (4)
- a0100131.gfm
- Apache1.1 License.txt
- GPL2.0 License.txt

Bill of Materials Code Matches Searches Dependencies

Folder: /MS+4.0/multi-server/utils

Search: Go

217 Components (1 - 10 10 per page) Jump...

Show: Precision

ID	Approved	Type	Component Name	Version	License	Pending	Identified
			Endian Firewall Community	Unspecified	GPL 2.0	3	0
			Endian Firewall Community	1.1-rc6	GPL 2.0	3	0
			Endian Firewall Community	1.1-rc8	GPL 2.0	3	0
			Endian Firewall Community	2.2 RC2	GPL 2.0	3	0
			Endian Firewall Community	EFW 2	GPL 2.0	3	0
			Endian Firewall Community	1.1-rc7	GPL 2.0	3	0
			Endian Firewall Community	1.1-rc5	GPL 2.0	2	0
			Endian Firewall Community	EFW 2.1	GPL 2.0	2	0
			Endian Firewall Community	EFW 2.1.1	GPL 2.0	2	0
			math-linux	Unspecified	GPL 3.0 (and others)	3	0
			Cross-Compilation Stuff for Embedded - cross-stuff	Unspecified	BSD 2.0	2	0

Identify:

*Component: Endian Firewall Community (homepage)

Version: 2.2 RC2

License: GPL 2.0 (view license)

Usage: File

Component Comment

(save comment)

IDENTIFY Snippet and File Matches (3 files pending) (show me pending)

IDENTIFY File Matches (0 files pending)

EXCLUDE Component (Recalculate Precision Matches) (3 files pending)

ORIGINAL TO: MS+4.0 (3 files pending)


RESET Identification

Create Local Component... Save Cancel

Software Package Data Exchange (SPDX)

[Home](#) [About SPDX](#) [Specifications](#) [License List](#) [Participate](#) [Documentation](#) [Tools](#)

The Software Package Data Exchange® (SPDX®) specification is a standard format for communicating the components, licenses and copyrights associated with a software package.



SPDX Specification version 2.0

The SPDX 2.0 Specification has been released. It is now the current version of the Specification.

News

From the SPDX Workgroup

Version 2.2 of the License List Released

Submitted by Jack on October 26th, 2015

SPDX License List: The year in review

by Jilayne Lovejoy, Legal team Co-Chair

Version 2.2 of the SPDX License List is now available and it seemed like a good opportunity to provide a summary of updates that have occurred over the last few releases and other related news.

The SPDX Specification

[Download the current version](#)


SPDX Version 2.0

This is the current version of the specification

[more versions...](#)

SPDX Tools

[Implement SPDX in your organization](#)



Protecode is an innovative provider of software compliance and vulnerability management systems, and an active member of the Linux Foundation's Software Package Data Exchange (SPDX) group

Events

[Learn about and participate in SPDX](#)

Security Problems with Using Open Source

- Vulnerabilities will be widely known once made public
 - But there is no supplier that informs its users about it
- Attackers have more options for finding vulnerabilities
 - Source code is public, which allows for additional attack angles
- Attackers find their targets more easily
 - Unless controlled carefully, most companies signal use

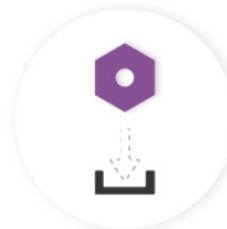
Seriousness of Vulnerabilities [S16]



10,000
new component
versions introduced daily



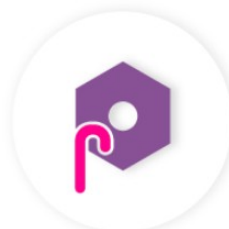
31B
component download
requests annually



229,898
average downloads per
enterprise per year



6.8%
components in apps have at
least one known vulnerability



3X
older components in
apps have 3x higher rate of
vulnerabilities



\$7.4M
estimated cost to remediate
10% of defects across
2,000 apps.

Example Governance for Security Problems

- Vulnerability identification
 - Bounty programs, vulnerability monitoring, ...
- Vulnerability triage
 - Exposure analysis, removal costs, ...
- Vulnerability removal
 - Installation tracking, update provision, ...

Process Problems with Using Open Source

- Product with project alignment
 - Schedule alignment
 - Neither schedule nor feature set may be predictable
 - New function development
 - Competitive situation may require closed development
- Avoiding maintenance burden
 - Developers want to “contribute back” bug fixes or small improvements
 - Such contribution signals company’s use of the software
 - Such signaling notifies attackers (both legal and security)

Seriousness of Process Problems

- Schedule alignment
 - Well-run open source projects have a predictable schedule
 - Feature set may not be predictable though
- New function development
 - Closed development may simply be a competitive requirement
 - Creates stress on contribution and integration (cf. Linux kernel)
- Avoiding maintenance burden
 - Can be mediated through appropriate governance

Example Governance for Maintenance Burden

- Contribution review
 - Define point-of-contact, review competitive exposure, ...
- Contribution execution
 - Appoint gatekeeper, involve external contributor, ...

Example Governance for Minimizing Costs

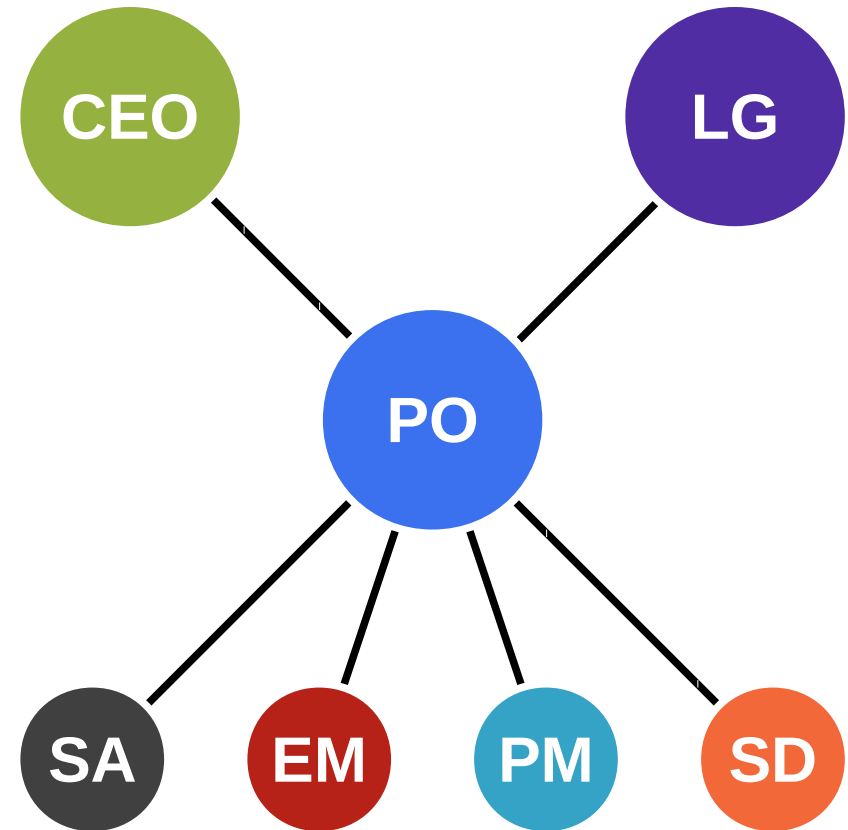
- Minimizing costs = process and resource efficiency
- Define clear roles and assign responsibilities
- Provide as much self-help as possible

Governance Domains and Processes

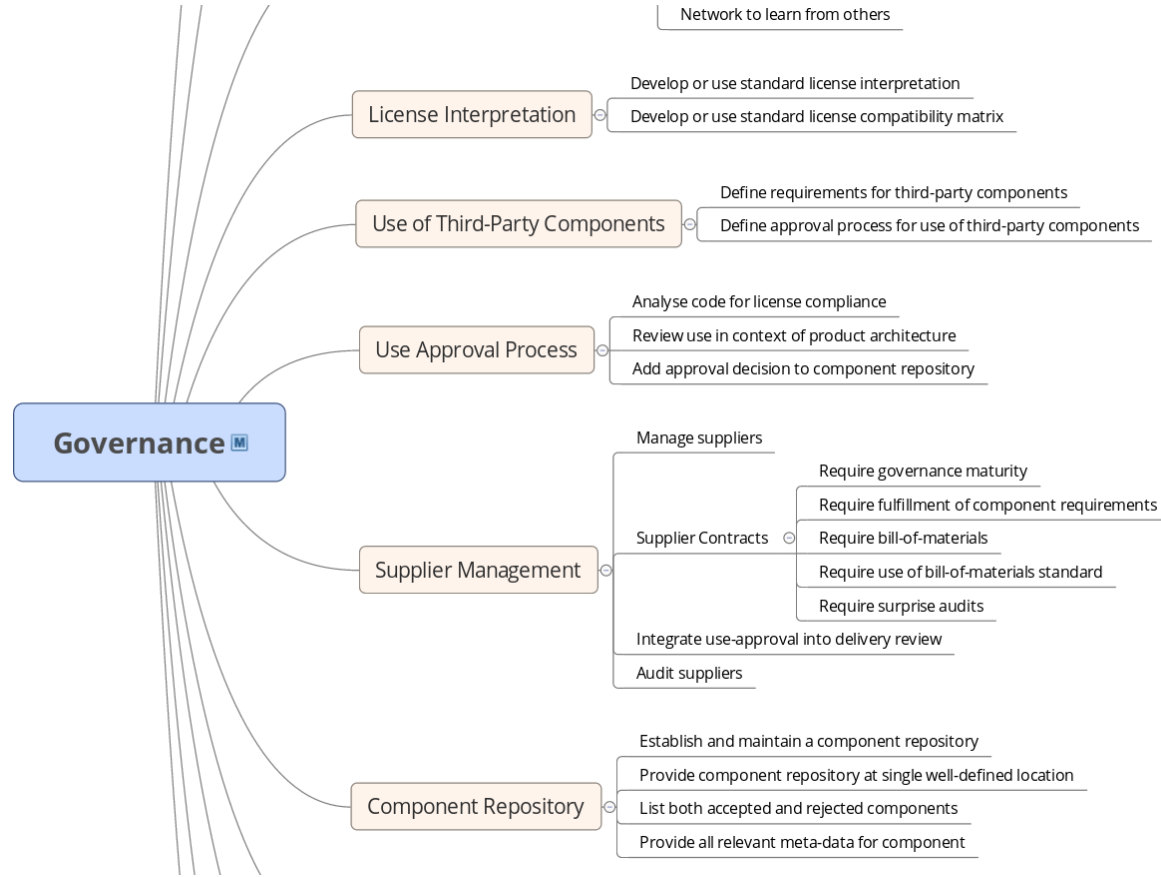
	Roles	Practices	Artifacts
Governance management	<ul style="list-style-type: none">• CEO• Program officer• Legal counsel• ...	<ul style="list-style-type: none">• Define goals• Define processes• Define roles ...• ...	<ul style="list-style-type: none">• Open source charter• Process descriptions• Practices handbook• ...
Supplier management	<ul style="list-style-type: none">• Program officer• Software architect• Engineering manager• Supplier, ...	<ul style="list-style-type: none">• Enable audits• Require bill-of-materials• Explain governance• ...	<ul style="list-style-type: none">• Audit checklist• Bill-of-materials• Educational materials• ...
Engineering management	<ul style="list-style-type: none">• Program officer• Engineering manager• Supplier,• ...	<ul style="list-style-type: none">• Review third-party• Select component• Monitor code commits• ...	<ul style="list-style-type: none">• Deliverables checklist• Component repository• Code scanner• ...
Product management	<ul style="list-style-type: none">• Program officer• Product manager• Engineering manager• ...	<ul style="list-style-type: none">• Ensure standards• Comply with licenses• Provide source code• ...	<ul style="list-style-type: none">• Platform specifications• Open source licenses• Source code browser• ...
Software development	<ul style="list-style-type: none">• Program officer• Software developer• Engineering manager• ...	<ul style="list-style-type: none">• Teach practices• Suggest component• Use comp. repository• ...	<ul style="list-style-type: none">• Practices handbook• Code search engines• Code scanner• ...

Roles and Responsibilities

- **Program officer (PO)**
 - **Responsible for governance**
 - Defines processes and practices
 - Collaborates and mediates
 - Signs-off or escalates
- Other involved parties
 - Top-level management (CEO)
 - Defines strategic goals
 - Legal counsel (LG)
 - Engineering manager (EM)
 - Product manager (PM)
 - Software architect (SA)
 - Software developer (SD)



Governance Process and Practices [1]



Best Practice Descriptions

Short	Name	
Main responsible	Actor	
(Abstract form of)	Context	of problem
(Abstract form of)	Problem	at hand
(Abstract form of)	Solution	of problem

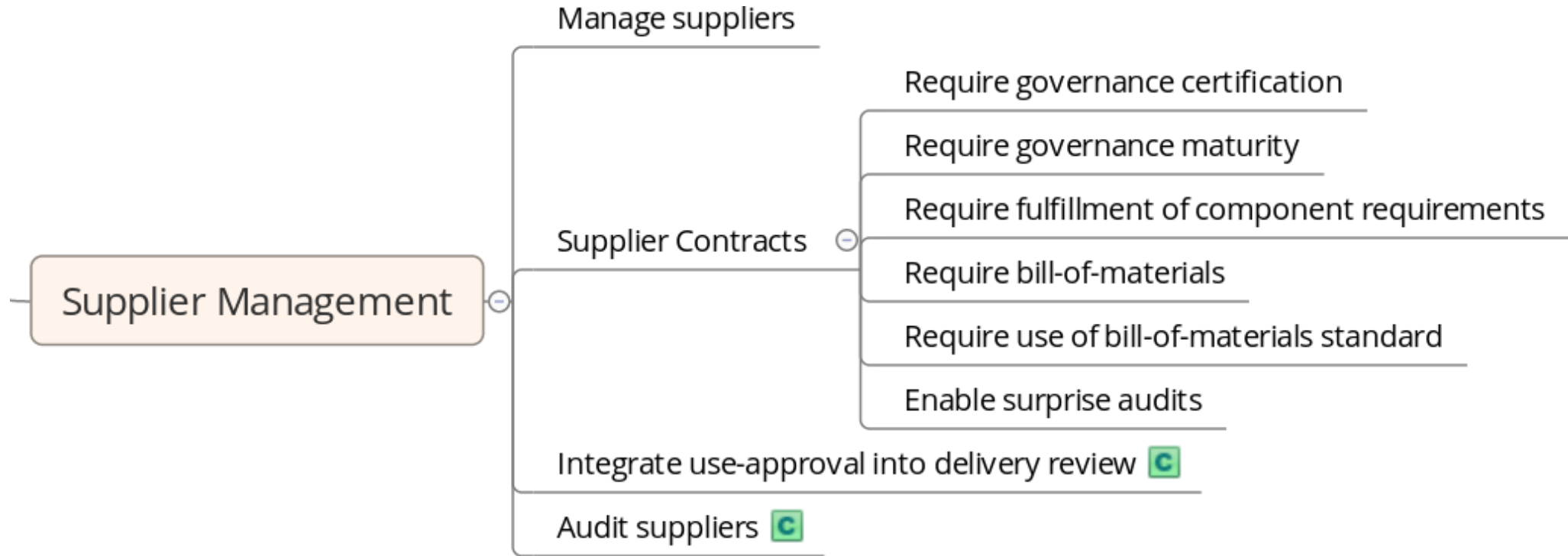
Generate Compliance Artifacts

Name	Generate compliance artifacts
Actor	Product manager
Context	The product is getting ready for release. You previously → <i>determined (all relevant) compliance artifacts.</i>
Problem	You will have to include all relevant compliance artifacts in the upcoming release.
Solution	Generate all relevant compliance artifacts from sources. Automate the generation as much as possible to avoid mistakes in an otherwise laborious but dull process. Once the product is ready for release, you'll have to → <i>provide (all relevant) compliance artifacts.</i>

Manage Suppliers

Name	Manage suppliers
Actor	Engineering manager
Context	<p>Your product includes not only open source components, but also third-party components that are supplied to you by other software vendors. In contrast to open source projects, you are paying for the component (license) and you are receiving it from a corporate entity.</p> <p>You previously → <i>defined (your) component requirements</i> and they must be met by any component, open source or not.</p>
Problem	How to ensure that a third-party component delivery meets your requirements?
Solution	<p>First, before you select a supplier, you may → <i>require governance certification</i> or at least → <i>require (a minimum) governance maturity</i> of them.</p> <p>Once you have decided for a supplier, in any delivery contract, you should → <i>require fulfillment of your component requirements</i> and you should → <i>require a bill-of-materials</i> upon delivery for which you → <i>require they use a bill-of-materials standard</i>.</p> <p>Upon delivery, you have to → <i>ensure requirements are met</i> and for this, you have to → <i>integrate use-approval into the delivery process</i>.</p> <p>If the supplier isn't certified and having to reject a component delivery is too expensive, you may want to → <i>enable surprise audits</i> and consequently also → <i>perform surprise audits</i> as to best governance practices.</p>

Manage Suppliers in Context



Quiz: Organizational Structure

- Where to put the open source program office?
 - Into product management
 - Into engineering management
 - Into legal department
 - Independent (staff function)

Review / Summary of Session

- Open source governance
 - Strategic goals (cost savings, business focus)
 - Challenges to using open source in products
 - Process and resource efficiency
- Provision and communication
 - Definition of an open source program office
 - Program office charter, roles and responsibilities
 - A handbook of processes and practices

Thank you! Questions?

dirk.riehle@fau.de – <http://osr.cs.fau.de>

dirk@riehle.org – <http://dirkriehle.com> – [@dirkriehle](#)

Credits and License

- Original version
 - © 2012-2019 Dirk Riehle, some rights reserved
 - Licensed under [Creative Commons Attribution 4.0 International License](#)
- Contributions
 - ...

Corporate Open Source Governance

Prof. Dr. Dirk Riehle

Friedrich-Alexander University Erlangen-Nürnberg

FOSS B03

Licensed under CC BY 4.0 International

It is Friedrich-Alexander University Erlangen-Nürnberg – FAU, in short.
Corporate identity wants us to say “Friedrich-Alexander University”.

Governance and Compliance

- Organizational **governance**
 - Is “the way the rules, norms and [desired] actions [of an organization] are produced, sustained, regulated and held accountable” (Wikipedia)
- Organizational **compliance**
 - Is how any behavior of and within an organization is made to comply with that organization’s rules, norms, and desired actions [DR]
- Software-using and producing organizations
 - Can have all kinds of governance and compliance bodies
 - One form is **open source governance and compliance**

Open Source Governance and Compliance

- **Open source governance and compliance**
 - Is the way an organization manages its open source engagement
 - License compliance is different from organizational compliance
- The three key forms of open source engagement
 - **Use** (of open source software in products)
 - **Contribute** (to open source software projects)
 - **Create or lead** (open source software projects)
- In the following, governance includes compliance

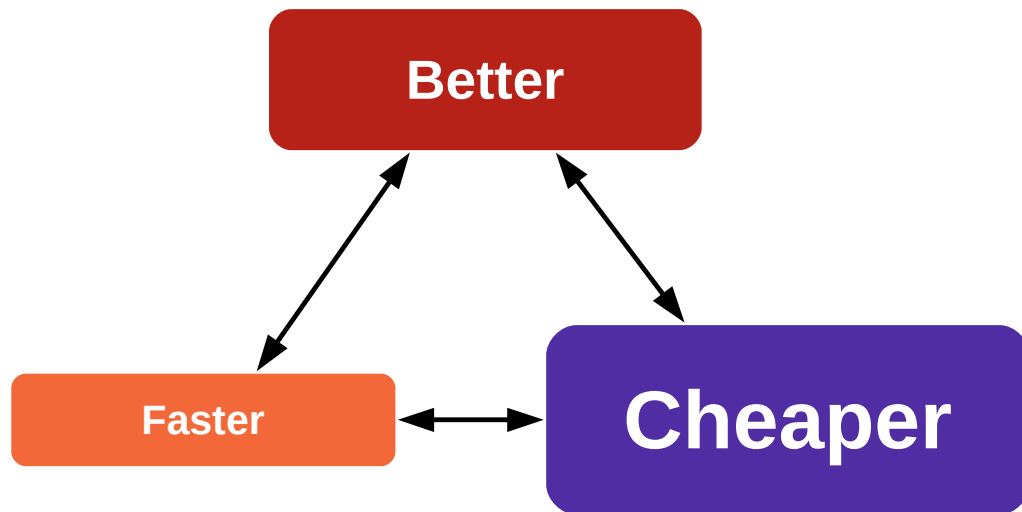
Products and Services

- Services, e.g. software-as-a-service (SaaS) are a product
 - Service provider operates product for customers, no on-premise use
 - Usually no upfront license fee, but maintenance fee (subscription)
- Open source licenses still apply!
 - The provision of the software as a service constitutes “distribution”
 - Specifically the AGPLv3 was conceived for this case
- **Product** from now on means both trad. software and SaaS

Primary Benefits of Using Open Source

- **Better**
 - Open source components can be of high quality
- **Faster**
 - Open source components are immediately available
- **Cheaper**
 - Open source components are free (no license fee)

Initial Motivation: Save Costs

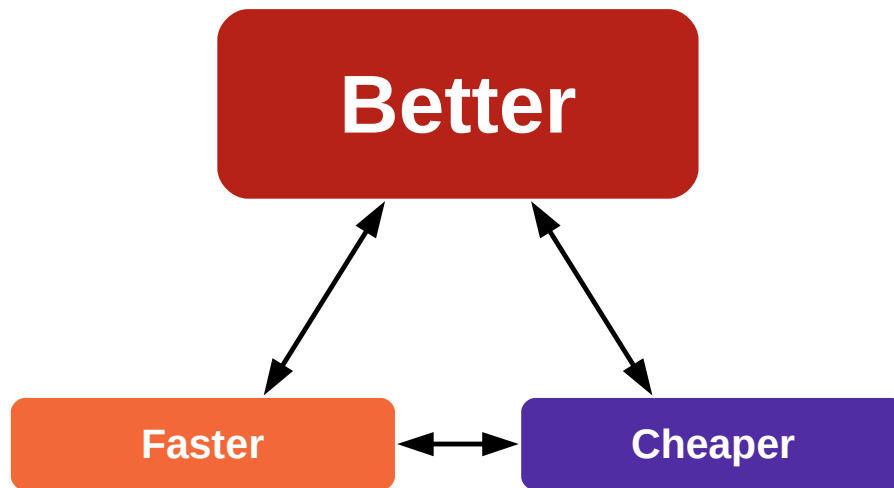


“High quality software free.”

Secondary Benefits

- Open source is open to inspect, modify
 - Faster: Users can help themselves, fix bugs
 - Faster: Users can extend the software, develop new features
- Most open source has no or little vendor lock-in
 - Faster: Innovation cannot be blocked by one company
 - Cheaper: Competition keeps service prices low
- Open source components are compatible
 - Faster, better: With standards (as reference implementations)
 - Faster, better: With platforms (as de-facto implementations)

Mature Motivation: Focus Business



“Focus on competitive differentiation.”

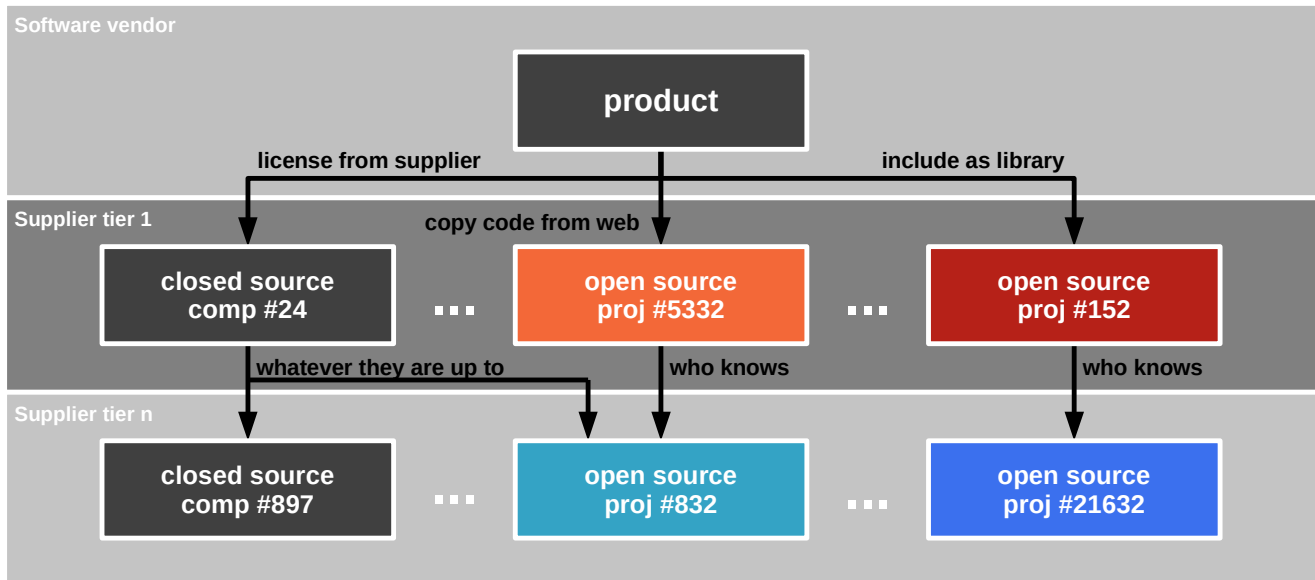
Detriments of Using Open Source

- Attackers have it a bit easier to find security holes
 - Various agencies scan open source for vulnerabilities, don't report them
 - Becomes more serious, if use of open source is signaled

Not Using Open Source is Not an Option

- “We don’t use open source” signals ...
 - They may not know what their engineers are doing
 - They may not have their business focus straight

The Software Supply Chain



The Open Source Program Office

- Open source goal and strategy formulation
 - Should be led by VP of Engineering and CTO
 - Typically leads to the creation of a program office
 - Should be done together with the program office
- Purpose of an open source program office
 - Serves as a point-of-contact
 - Manages governance processes
 - Manages governance itself
- Also sometimes called a competence center

Goals of Open Source Governance

1. **Achieving goals while**
2. **Avoiding problems**
3. **At minimal cost**

Achieving Goals for Using Open Source

- **Focus business**
 - Identification of open source opportunity
- **Save costs**
 - Selection of open source component
 - Inclusion of component in product
 - Management of open source investment

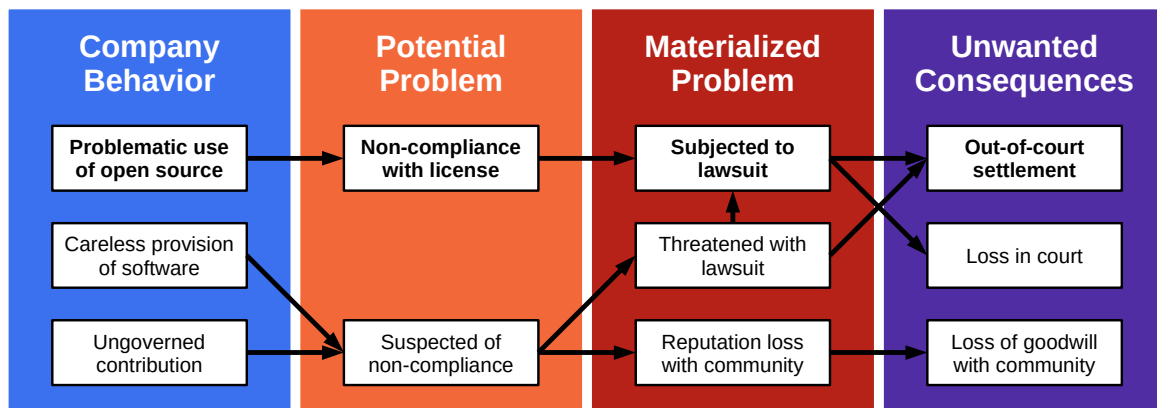
Example Governance for Achieving Goals

- **Identify open source opportunities**
 - Product manager determines business focus and engineering manager identifies correlating open source opportunities
- **Select open source component**
 - Engineering manager and software architect define criteria for open source component, select best fitting one; includes **clearance process**
- **Include open source component**
 - Software architect and software developer include open source component in product and bill-of-materials
- **Manage open source investment**
 - Engineering manager, software architect, and software developer monitor the evolution of the open source component, engage when necessary

Avoiding Problems with Using Open Source

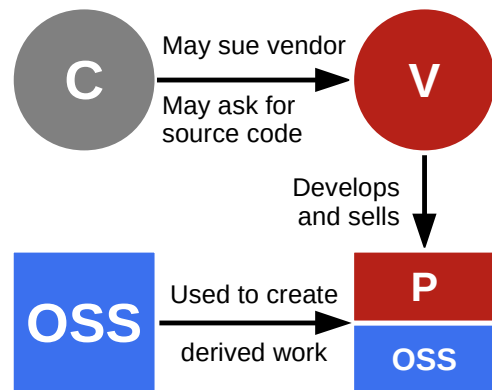
- Example of **Legal challenges**
 - Ensure usage rights as intended (license compliance)
- Example of **Quality challenges**
 - Ensure new security holes are recognized and fixed
- Example of **Process challenges**
 - Avoid maintenance burden by contributing back

Cause and Effect Chain of Legal Problems



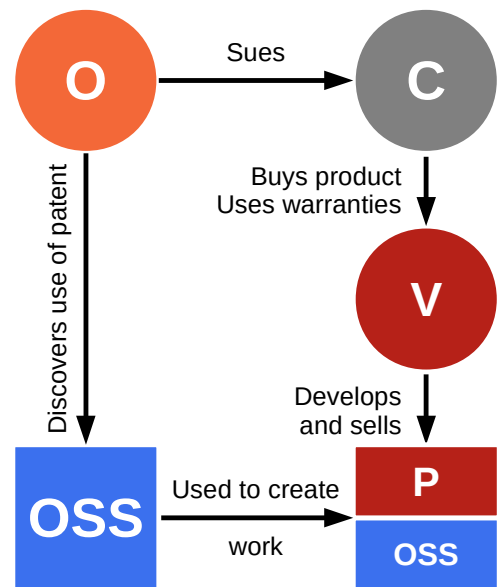
Copyright Violation Lawsuit

- Steps in process
 - Customer C acquires license for commercial product P from software vendor V
 - Customer asks software vendor for source code; in all likelihood gets denied
 - Depending on license and technical coupling, this denial is a license violation



Patent Infringement Lawsuit

- Setup of situation
 - OSS contains code that implements patent owned by patent holder O
 - Software vendor sells license for P to customer C without licensing patent
- Steps in process
 1. Patent holder sues customer (patent user)
 2. Customer turns around, sues software vendor
- Applies broadly
 - To both use-cases
 - In-house
 - In products



Costs of Materialized Problems

- When threatened with lawsuit
 - Legal costs
 - Reputation loss
 - Wasted management attention
- When subjected to lawsuit, like before +
 - **Undesired revealing of information**

Costs of Unwanted Consequences

- When settling out-of-court, like before +
 - Financial costs
 - Compliance costs
- When losing in court, like before +
 - Financial costs
 - License and other fees
 - Potential punitive damages
 - **Product revenue loss due to ...**
 - **Product recall / sales stop**
 - **Delay of product releases**
 - Compliance costs
 - Increased risks of more lawsuits

Costs of Establishing License Compliance

- In case of component replacement
 - Development costs (rework)
 - New component licensing fees
- In case of non-replacement
 - Labor costs (compliance artifacts)
 - Intellectual property loss
 - Exclusivity of source code
 - Potential free patent rights grant
 - Undesired revealing of information

Probability of Getting Sued

- Sources of lawsuits
 - GPL enforcers
 - GPL monetizers

GPL Monetization and Netfilter [KS16]

Specifically, we remain aware of multiple non-community-oriented GPL enforcement efforts [...] These “**GPL monetizers**”, who trace their roots to nefarious business models that seek to catch users in minor violations in order to **sell an alternative proprietary license**, stand in stark contrast to the work that Conservancy, FSF and gpl-violations.org have done for years.

Most notably, a Linux developer named **Patrick McHardy continues ongoing GPL enforcement actions** [...] In the last two years, we've heard repeated rumors about Patrick's enforcement activity, as well as some reliable claims by GPL violators that Patrick failed to follow the Principles.

Patrick's enforcement occurs primarily in Germany. We know well the difficulties of working transparently in that particular legal system, but both gpl-violations.org and Conservancy have done transparent enforcement in that jurisdiction and others.

Patrick McHardy has also been suspended from work on the Netfilter core team. While the Netfilter team itself publicly endorsed these Principles of enforcement, Patrick has not. Conservancy agrees that Patrick's apparent refusal to endorse the Principles leaves suspicion and concern, since the Principles have been endorsed by so many other Linux copyright holders, including Conservancy.

The CEO's Liability

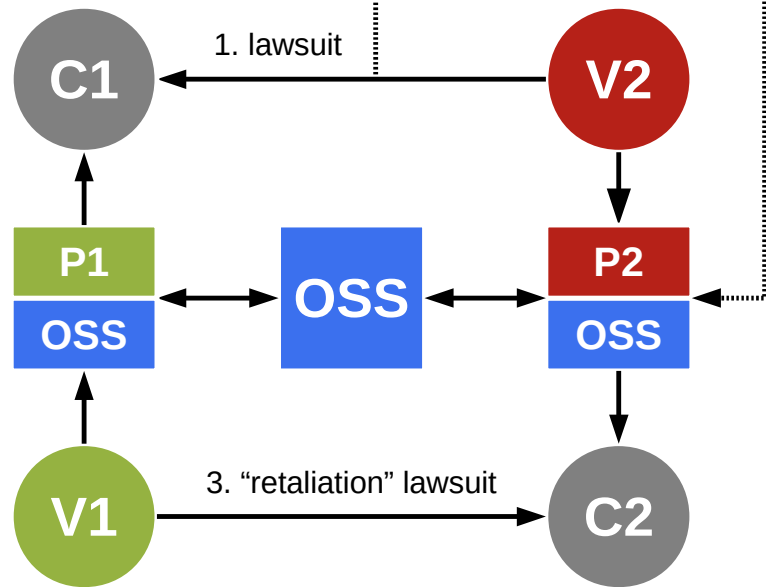
- Good open source governance is “state of the art”
 - Many large companies demonstrate how to do it
 - Therefore, good governance is responsibility of the CEO
- Leads to liability of the CEO resp. “Geschäftsführerhaftung”

Proactive Measures against Patent Lawsuits

- Open Invention Network (OIN) founded to protect Linux
 - OIN owns a large patent pool that it licenses out royalty-free
 - Requires that licensees agree not to enforce patents against Linux
- Modern open source licenses include patent retaliation clauses
 - Users of open source, who enforce patents, lose right to use open source
 - Retaliation clauses are not present in older licenses, e.g. not in GPLv2

Patent Retaliation Lawsuit

- Setup of situation
 - Both V1 and V2 contribute to OSS
- Original lawsuit
 - V2 sues C1 for patent **patent infringement**
 - C1 turns around, holds supplier V1 responsible
- Retaliation lawsuit
 - V1 sues C2 for **copyright violation**
 - C2 turns around, holds supplier V2 responsible



No risk

No reward

How to manage this risk?

Example Governance for Legal Problems

- **Developer education**
 - Educate developers, refresh education, provide handbook, ...
- **Open source clearance process**
 - Have clearance process, define point-of-contact, provide veto rights, ...
- **Supplier management**
 - Manage suppliers, define acceptable licenses, require bill-of-materials, ...

Component Analysis 1 / 2

Component Analysis 2 / 2

Software Package Data Exchange (SPDX)

Security Problems with Using Open Source

- Vulnerabilities will be widely known once made public
 - But there is no supplier that informs its users about it
- Attackers have more options for finding vulnerabilities
 - Source code is public, which allows for additional attack angles
- Attackers find their targets more easily
 - Unless controlled carefully, most companies signal use

Seriousness of Vulnerabilities [S16]

Example Governance for Security Problems

- Vulnerability identification
 - Bounty programs, vulnerability monitoring, ...
- Vulnerability triage
 - Exposure analysis, removal costs, ...
- Vulnerability removal
 - Installation tracking, update provision, ...

Process Problems with Using Open Source

- Product with project alignment
 - Schedule alignment
 - Neither schedule nor feature set may be predictable
 - New function development
 - Competitive situation may require closed development
 - Avoiding maintenance burden
 - Developers want to “contribute back” bug fixes or small improvements
 - Such contribution signals company’s use of the software
 - Such signaling notifies attackers (both legal and security)

Seriousness of Process Problems

- Schedule alignment
 - Well-run open source projects have a predictable schedule
 - Feature set may not be predictable though
- New function development
 - Closed development may simply be a competitive requirement
 - Creates stress on contribution and integration (cf. Linux kernel)
- Avoiding maintenance burden
 - Can be mediated through appropriate governance

Example Governance for Maintenance Burden

- Contribution review
 - Define point-of-contact, review competitive exposure, ...
- Contribution execution
 - Appoint gatekeeper, involve external contributor, ...

Example Governance for Minimizing Costs

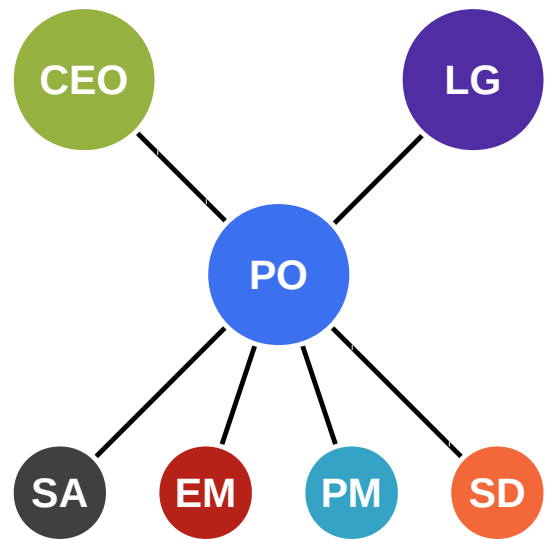
- Minimizing costs = process and resource efficiency
- Define clear roles and assign responsibilities
- Provide as much self-help as possible

Governance Domains and Processes

	Roles	Practices	Artifacts
Governance management	<ul style="list-style-type: none"> • CEO • Program officer • Legal counsel • ... 	<ul style="list-style-type: none"> • Define goals • Define processes • Define roles ... • ... 	<ul style="list-style-type: none"> • Open source charter • Process descriptions • Practices handbook • ...
Supplier management	<ul style="list-style-type: none"> • Program officer • Software architect • Engineering manager • Supplier, ... 	<ul style="list-style-type: none"> • Enable audits • Require bill-of-materials • Explain governance • ... 	<ul style="list-style-type: none"> • Audit checklist • Bill-of-materials • Educational materials • ...
Engineering management	<ul style="list-style-type: none"> • Program officer • Engineering manager • Supplier, • ... 	<ul style="list-style-type: none"> • Review third-party • Select component • Monitor code commits • ... 	<ul style="list-style-type: none"> • Deliverables checklist • Component repository • Code scanner • ...
Product management	<ul style="list-style-type: none"> • Program officer • Product manager • Engineering manager • ... 	<ul style="list-style-type: none"> • Ensure standards • Comply with licenses • Provide source code • ... 	<ul style="list-style-type: none"> • Platform specifications • Open source licenses • Source code browser • ...
Software development	<ul style="list-style-type: none"> • Program officer • Software developer • Engineering manager • ... 	<ul style="list-style-type: none"> • Teach practices • Suggest component • Use comp. repository • ... 	<ul style="list-style-type: none"> • Practices handbook • Code search engines • Code scanner • ...

Roles and Responsibilities

- **Program officer (PO)**
 - **Responsible for governance**
 - Defines processes and practices
 - Collaborates and mediates
 - Signs-off or escalates
- Other involved parties
 - Top-level management (CEO)
 - Defines strategic goals
 - Legal counsel (LG)
 - Engineering manager (EM)
 - Product manager (PM)
 - Software architect (SA)
 - Software developer (SD)



Governance Process and Practices [1]

Best Practice Descriptions

Short	Name	
Main responsible	Actor	
(Abstract form of)	Context	of problem
(Abstract form of)	Problem	at hand
(Abstract form of)	Solution	of problem

Generate Compliance Artifacts

Name	Generate compliance artifacts
Actor	Product manager
Context	The product is getting ready for release. You previously → <i>determined (all relevant) compliance artifacts.</i>
Problem	You will have to include all relevant compliance artifacts in the upcoming release.
Solution	Generate all relevant compliance artifacts from sources. Automate the generation as much as possible to avoid mistakes in an otherwise laborious but dull process. Once the product is ready for release, you'll have to → <i>provide (all relevant) compliance artifacts.</i>

Manage Suppliers

Name	Manage suppliers
Actor	Engineering manager
Context	<p>Your product includes not only open source components, but also third-party components that are supplied to you by other software vendors. In contrast to open source projects, you are paying for the component (license) and you are receiving it from a corporate entity.</p> <p>You previously → <i>defined (your) component requirements</i> and they must be met by any component, open source or not.</p>
Problem	How to ensure that a third-party component delivery meets your requirements?
Solution	<p>First, before you select a supplier, you may → <i>require governance certification</i> or at least → <i>require (a minimum) governance maturity</i> of them.</p> <p>Once you have decided for a supplier, in any delivery contract, you should → <i>require fulfillment of your component requirements</i> and you should → <i>require a bill-of-materials</i> upon delivery for which you → <i>require they use a bill-of-materials standard</i>.</p> <p>Upon delivery, you have to → <i>ensure requirements are met</i> and for this, you have to → <i>integrate use-approval into the delivery process</i>.</p> <p>If the supplier isn't certified and having to reject a component delivery is too expensive, you may want to → <i>enable surprise audits</i> and consequently also → <i>perform surprise audits</i> as to best governance practices.</p>

Manage Suppliers in Context

Quiz: Organizational Structure

- Where to put the open source program office?
 - Into product management
 - Into engineering management
 - Into legal department
 - Independent (staff function)

Review / Summary of Session

- Open source governance
 - Strategic goals (cost savings, business focus)
 - Challenges to using open source in products
 - Process and resource efficiency
- Provision and communication
 - Definition of an open source program office
 - Program office charter, roles and responsibilities
 - A handbook of processes and practices

Thank you! Questions?

dirk.riehle@fau.de – <http://osr.cs.fau.de>

dirk@riehle.org – <http://dirkriehle.com> – [@dirkriehle](#)

DR

Credits and License

- Original version
 - © 2012-2019 Dirk Riehle, some rights reserved
 - Licensed under [Creative Commons Attribution 4.0 International License](#)
- Contributions
 - ...