Legal and practical aspects of Open Source software in industry

Legal Heidelberg OSADL Talks, September 29, 2020, Online Session 2

How can OSADL help with license compliance – Part I
- License compliance as part of company compliance
- OSADL Open Source Policy project
- What is software license scanning and when is it needed?

What is "Free and Open Source software" (FOSS)?

- Software whose **license** fulfills specific requirements is called "Free", "Open Source" or "Free and Open Source".
- Unrestricted and unconditional permission to **run, analyze and modify** the software
- **Copying and distribution** are permitted provided that **license conditions** are complied with

What is company compliance?

- Compliance with legal provisions and regulations
- Compliance with standards
- Compliance with ethical requirements

What is company compliance?

- Compliance with legal provisions and regulations
- Compliance with standards
- Compliance with ethical requirements

Copyright Law

To prevent copyright infringement, protected works may only be copied and distributed when a valid license is obtained.

Who is responsible in a company to take care that copyright law is obeyed?

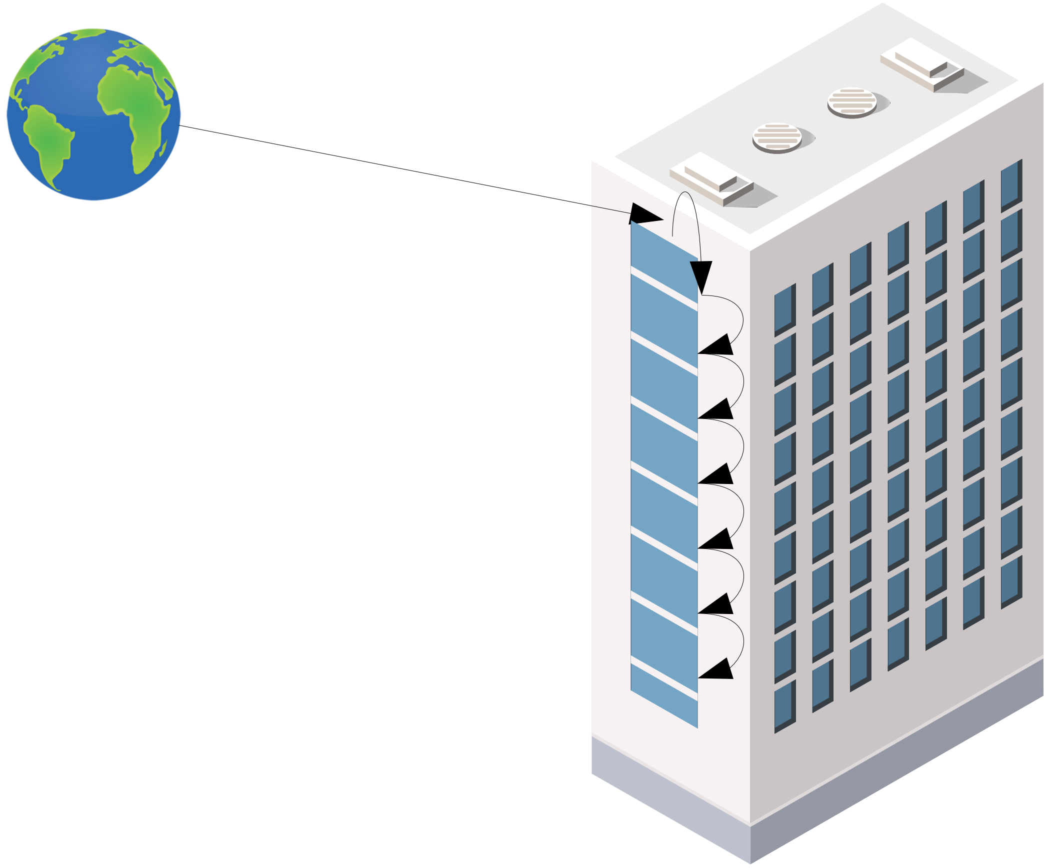
- Not any employee.
- But the management!

Who is responsible in a company to take care that copyright law is obeyed?

- Not any employee.
- But the management!

COMPLIANCE IS A MATTER OF THE BOSS!

How do *new strategies* normally enter a company?



Management

Strategic management

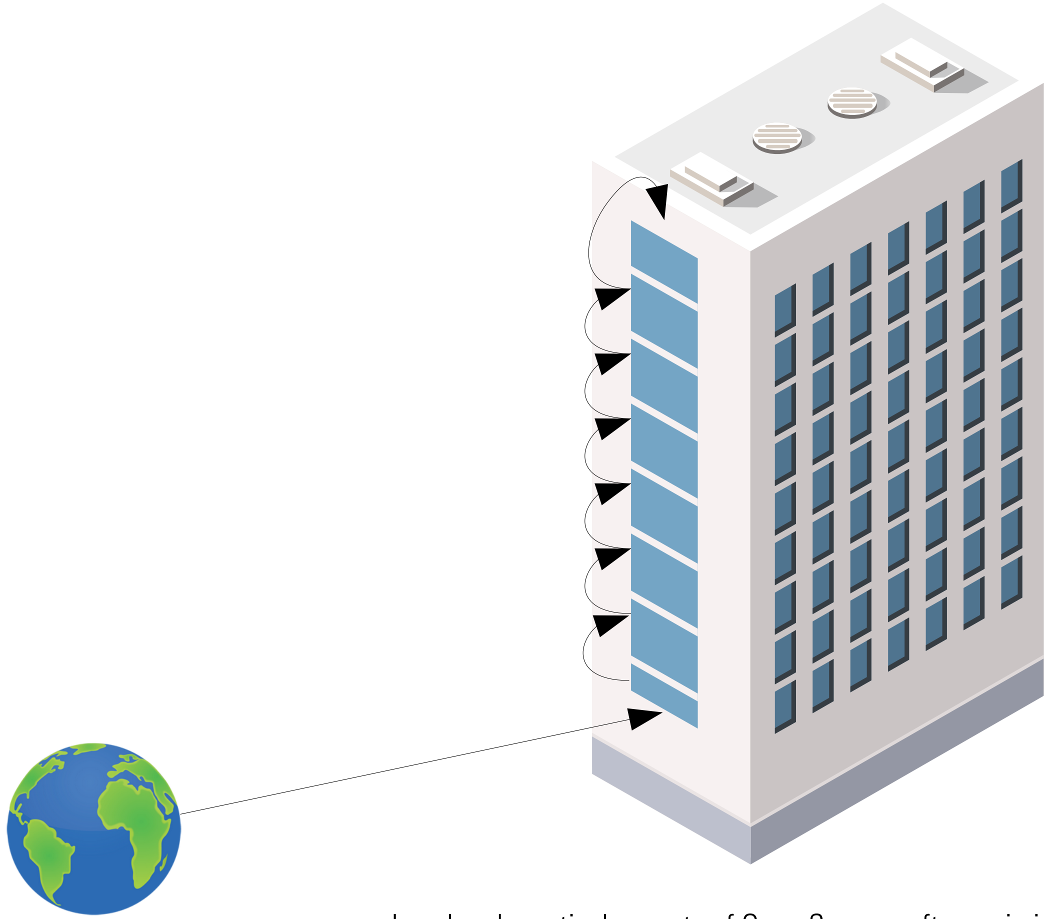
Development dept.

Product management

QA dept.

Software development

How does *Open Source* normally enter a company?



Management

Strategic management

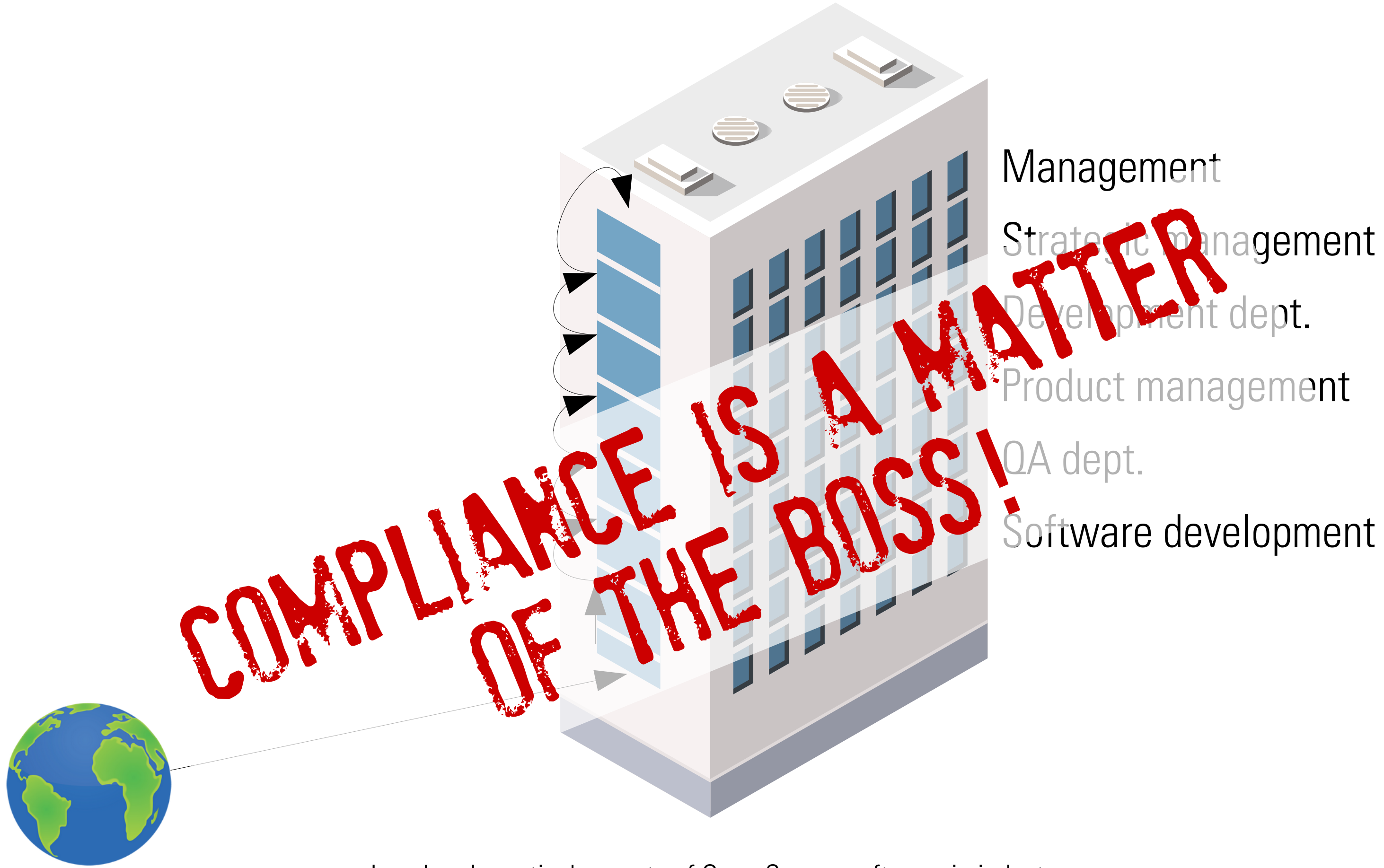
Development dept.

Product management

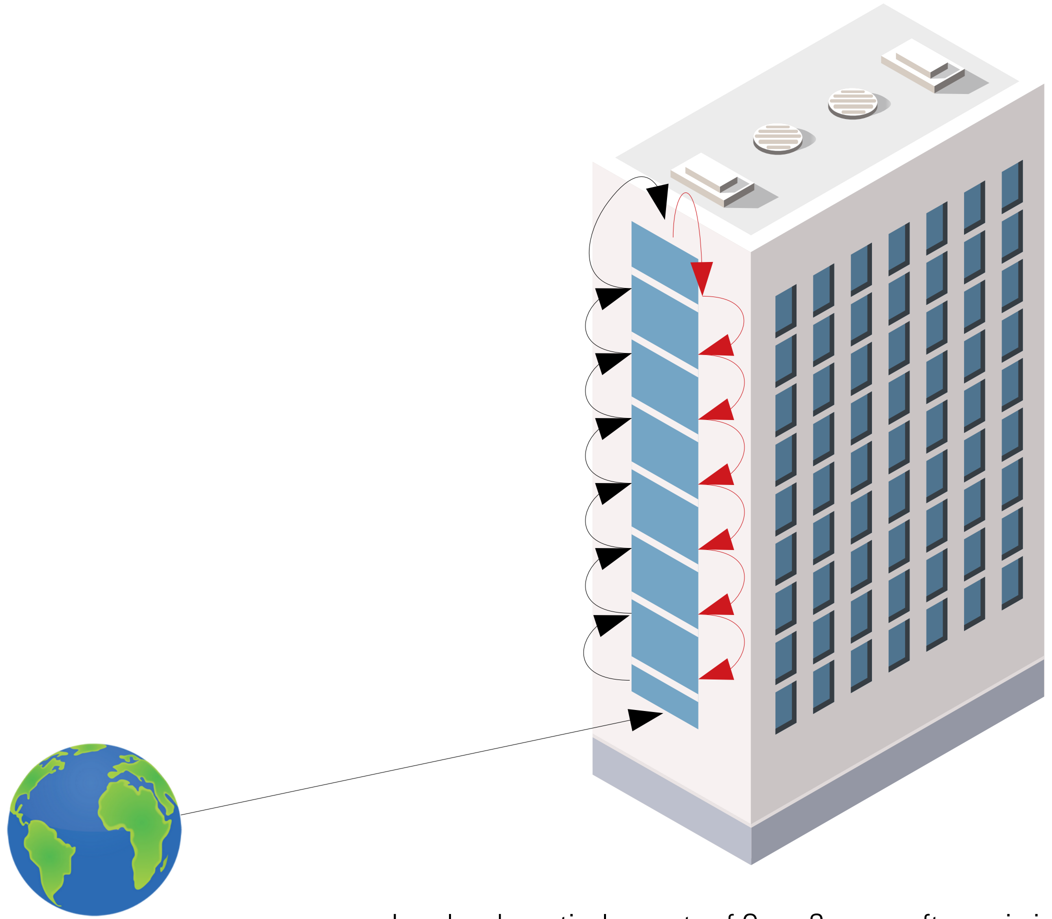
QA dept.

Software development

How does *Open Source* normally enter a company?



Open Source needs *both ways*



Management

Strategic management

Development dept.

Product management

QA dept.

Software development

How can OSADL help?

OSADL provides a wide variety of tools and services to help companies use and distribute FOSS compliantly:

- Database of Frequently Asked Questions (OSADL FAQ)
- Comprehensive legal assessments
- Open Source License Obligations Checklists
- License compliance audit (LCA)
- In-house and public seminars and workshops

How can OSADL help?

OSADL provides a wide variety of tools and services to help companies use and distribute FOSS compliantly:

- Database of Frequently Asked Questions (OSADL FAQ)
- Comprehensive legal assessments
- Open Source License Obligations Checklists
- License compliance audit (LCA)
- In-house and public seminars and workshops

The OSADL Open Source Policy Template: The Basis for License Compliance

The Basis for License Compliance

A FOSS policy is needed ...

... to create and maintain **processes** within a company,

... to establish **understanding** of concepts related to FOSS,

... to avoid copyright infringements,

... to provide **control** about licensing of a company’s IP,

... to meet customer requirements.

Open Source Policy Template

- Different companies take different approaches to license compliance, a company's FOSS policy must reflect these
- Creating a policy requires **understanding and expertise**
- Using a policy requires it to be **brief and specific**

➔ The OSADL Open Source Policy Template is structured to take these requirements into account

Structure of the Open Source Policy Template

- Various chapters with template texts as basis for an individual policy
- Motivations and explanations for the creator of the company policy

☑ *Options to choose from where there are alternative possibilities of interpreting or handling a situation*

- text blocks to modify contracts and other documents
- *Placeholders to be filled out individually*

→ Annexes providing processes and forms for legal information on products

→ Supplements providing technical, legal and practical background on copyright law and license compliance.

Content of the Open Source Policy Template

- Introducing and **defining terms** such as "Free and Open Source Software (FOSS)"
- Establishing the **goals** of the policy
- Assigning **responsibilities and tasks** to employees that handle FOSS
- **Communicating** the policy within a company
- Establishing sustainable processes to **detect, analyze and handle FOSS**
- Equipping products with **FOSS license information**
- **Contributing** to the FOSS ecosystem
- Organizing **audits and certification**

Software flow in a company

Software flow: Incoming software

Employee

Freelancer

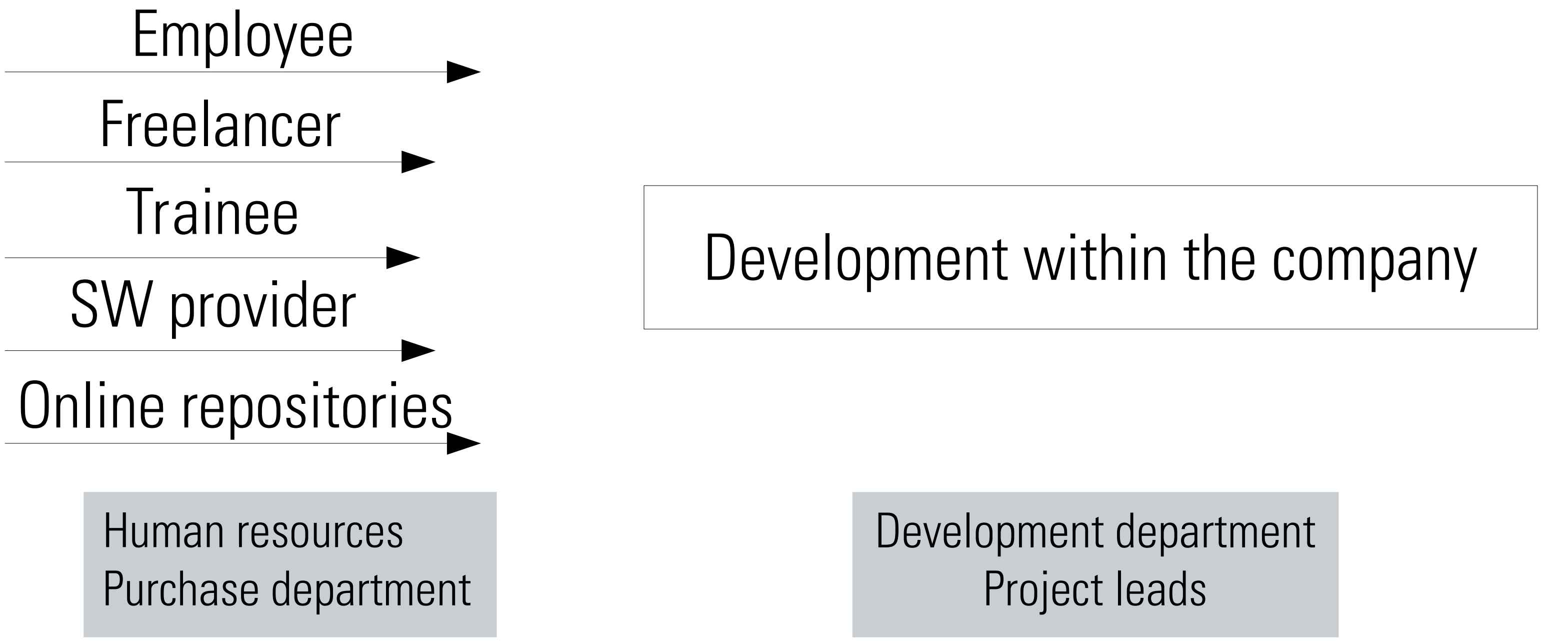
Trainee

SW provider

Online repositories

Human resources
Purchase department

Software flow: Own development



Software flow: Outgoing software

Employee

Freelancer

Trainee

SW provider

Online repositories

Development within the company

Delivery

- Electronically
- Installed on HW
- On a medium
- Via Internet

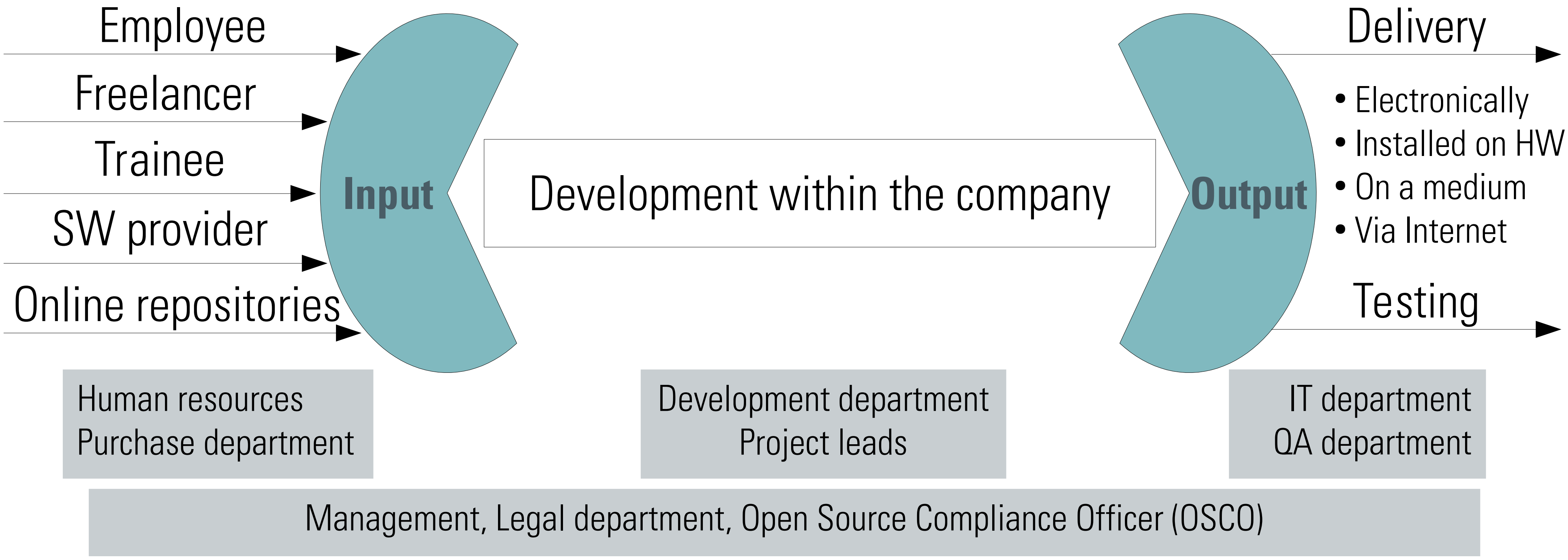
Testing

Human resources
Purchase department

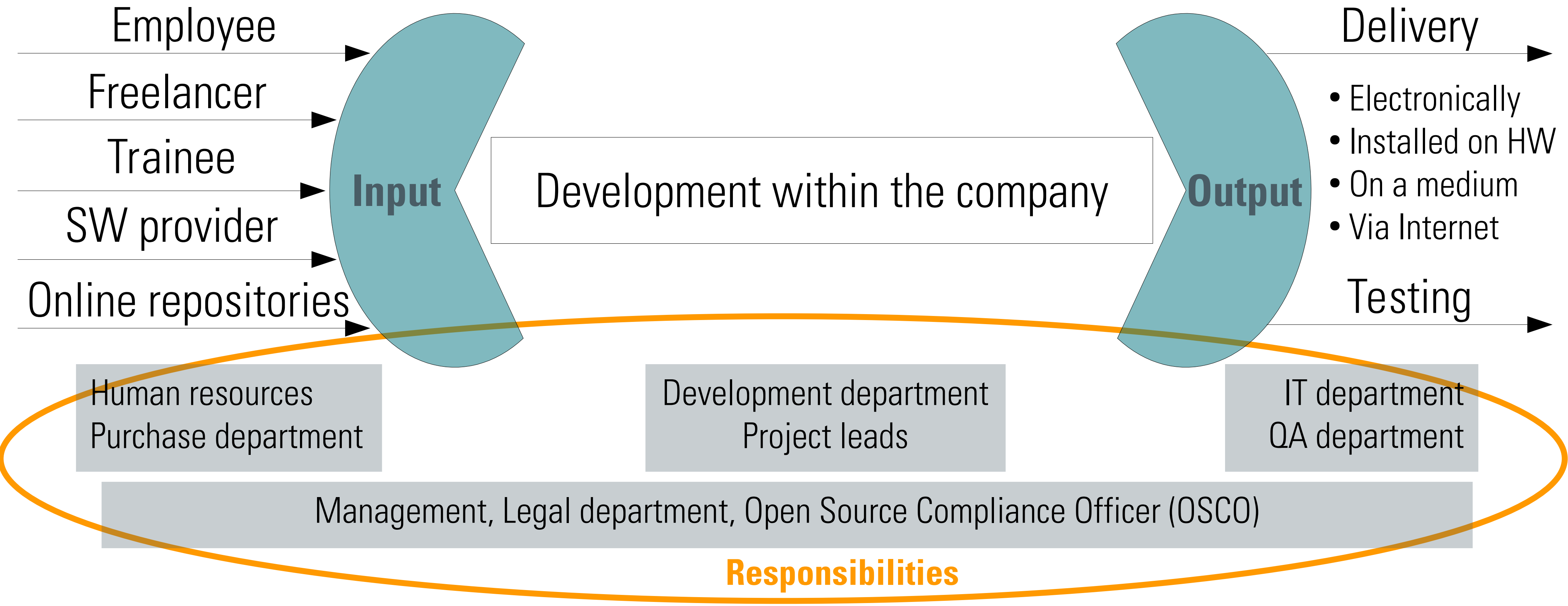
Development department
Project leads

IT department
QA department

Software flow: Input/output gateways



Software flow (1)

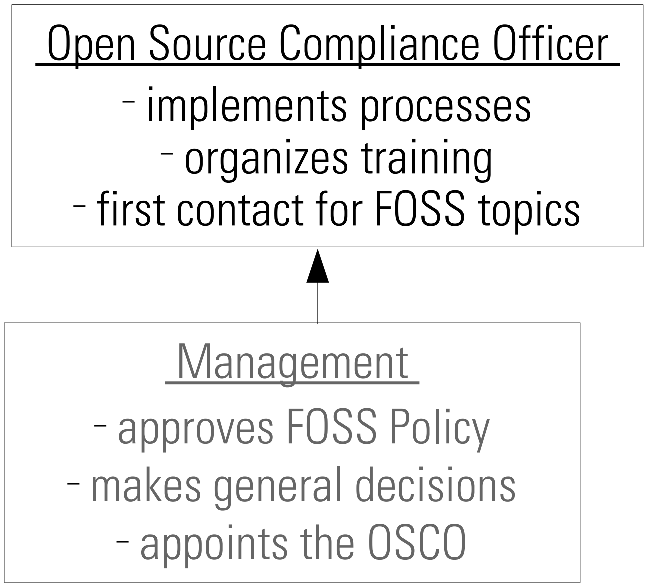


Responsibilities

Management

- approves FOSS Policy
- makes general decisions
- appoints the OSCO

Responsibilities



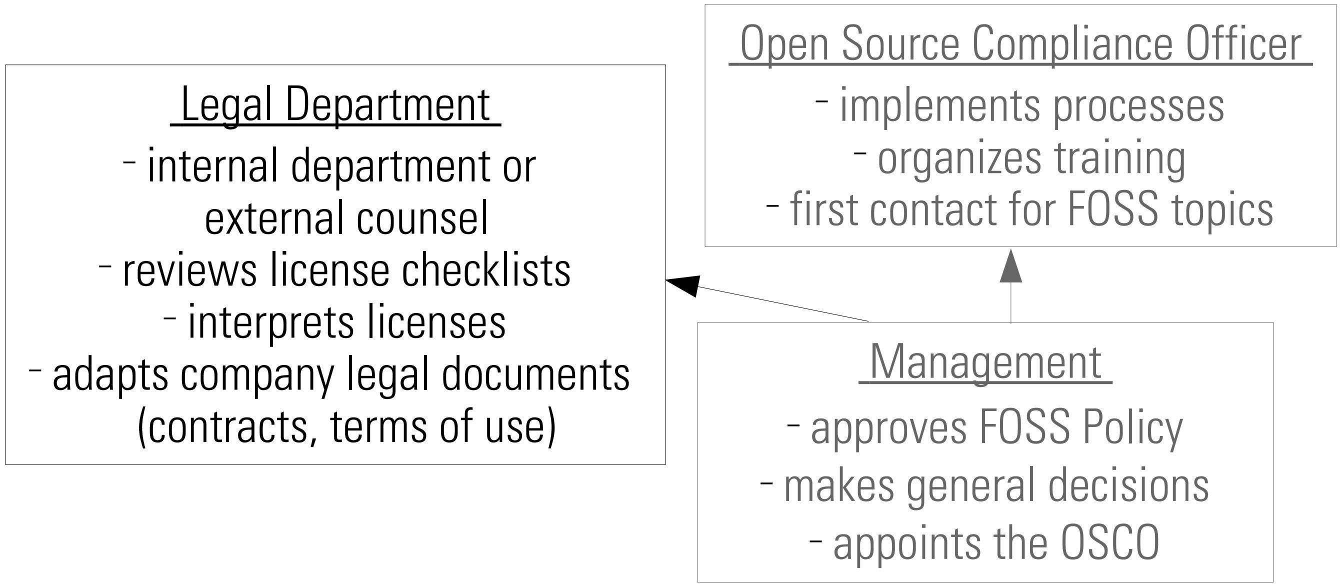
Open Source Compliance Officer

- implements processes
- organizes training
- first contact for FOSS topics

Management

- approves FOSS Policy
- makes general decisions
- appoints the OSCO

Responsibilities



Open Source Compliance Officer

- implements processes
- organizes training
- first contact for FOSS topics

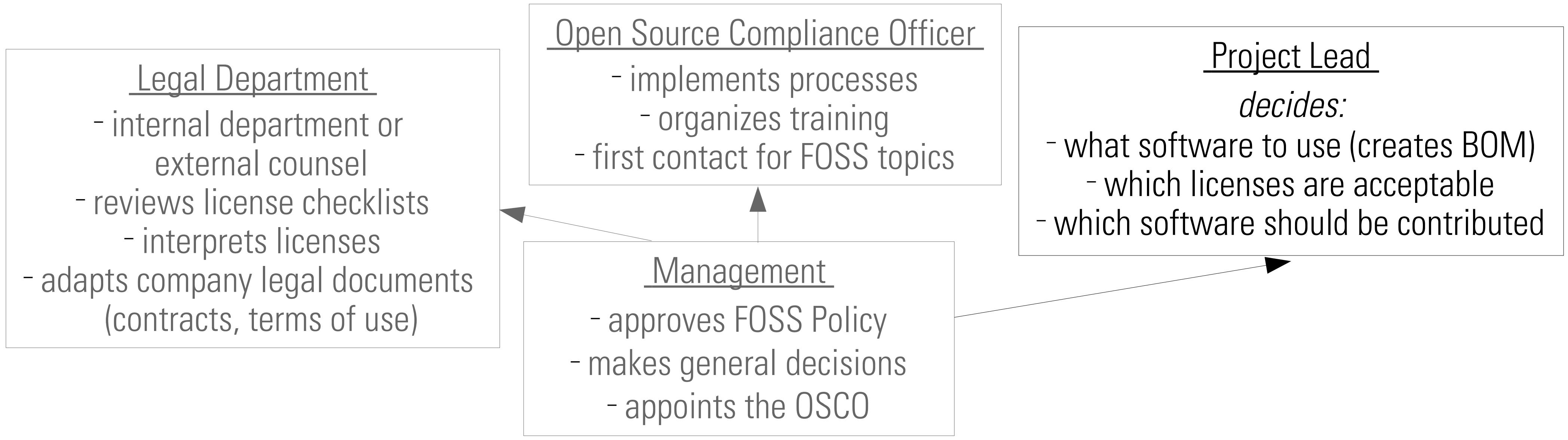
Legal Department

- internal department or external counsel
- reviews license checklists
- interprets licenses
- adapts company legal documents (contracts, terms of use)

Management

- approves FOSS Policy
- makes general decisions
- appoints the OSCO

Responsibilities



Open Source Compliance Officer

- implements processes
- organizes training
- first contact for FOSS topics

Legal Department

- internal department or external counsel
- reviews license checklists
- interprets licenses
- adapts company legal documents (contracts, terms of use)

Management

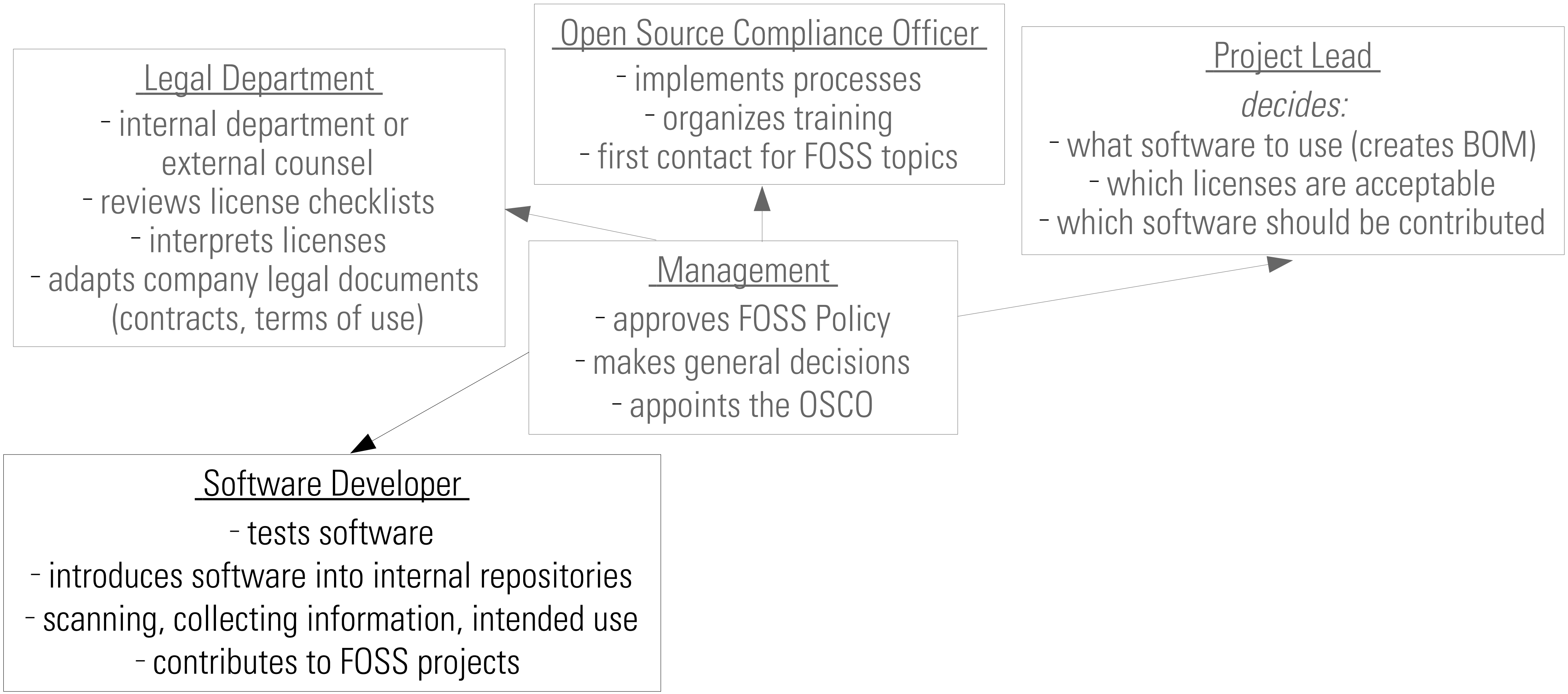
- approves FOSS Policy
- makes general decisions
- appoints the OSCO

Project Lead

decides:

- what software to use (creates BOM)
- which licenses are acceptable
- which software should be contributed

Responsibilities



Open Source Compliance Officer

- implements processes
- organizes training
- first contact for FOSS topics

Legal Department

- internal department or external counsel
- reviews license checklists
- interprets licenses
- adapts company legal documents (contracts, terms of use)

Project Lead

decides:

- what software to use (creates BOM)
- which licenses are acceptable
- which software should be contributed

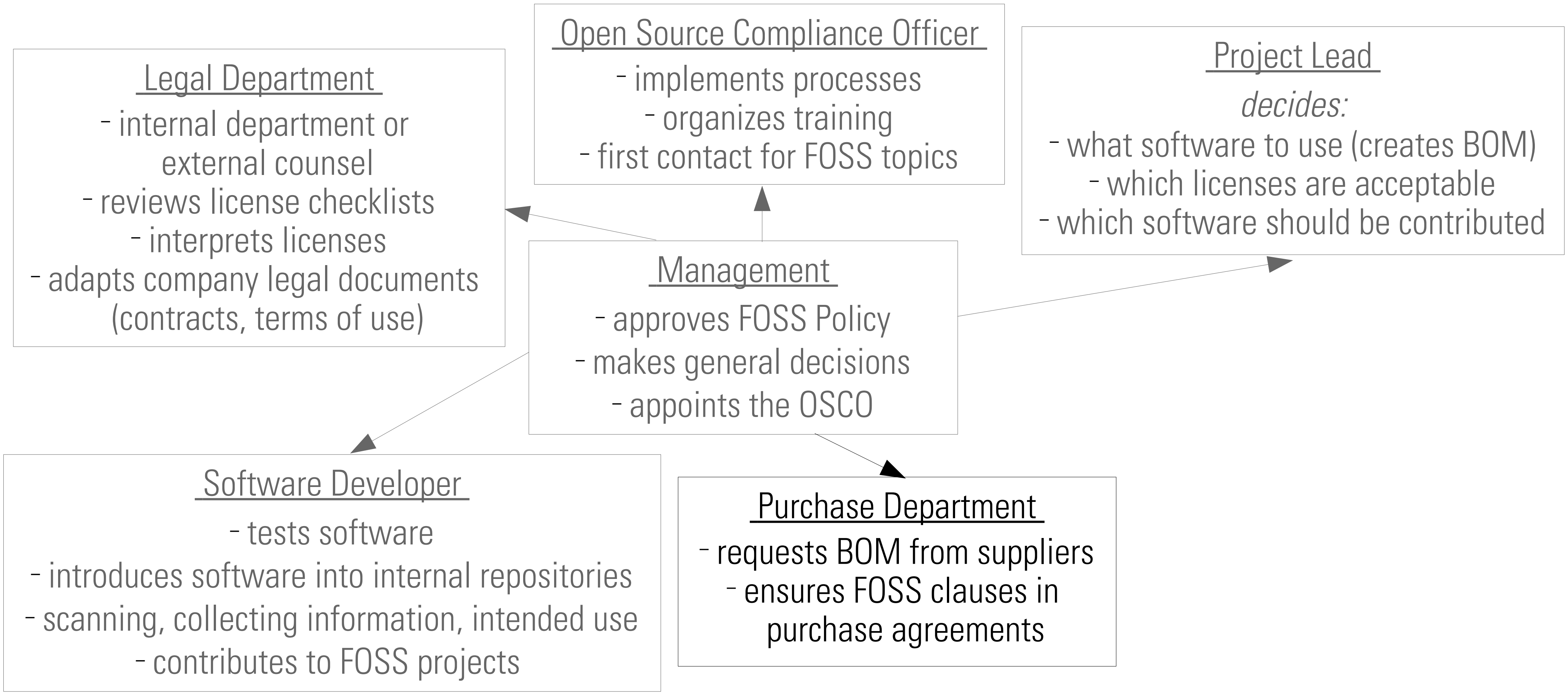
Management

- approves FOSS Policy
- makes general decisions
- appoints the OSCO

Software Developer

- tests software
- introduces software into internal repositories
- scanning, collecting information, intended use
- contributes to FOSS projects

Responsibilities



Open Source Compliance Officer
- implements processes
- organizes training
- first contact for FOSS topics

Legal Department
- internal department or external counsel
- reviews license checklists
- interprets licenses
- adapts company legal documents (contracts, terms of use)

Project Lead

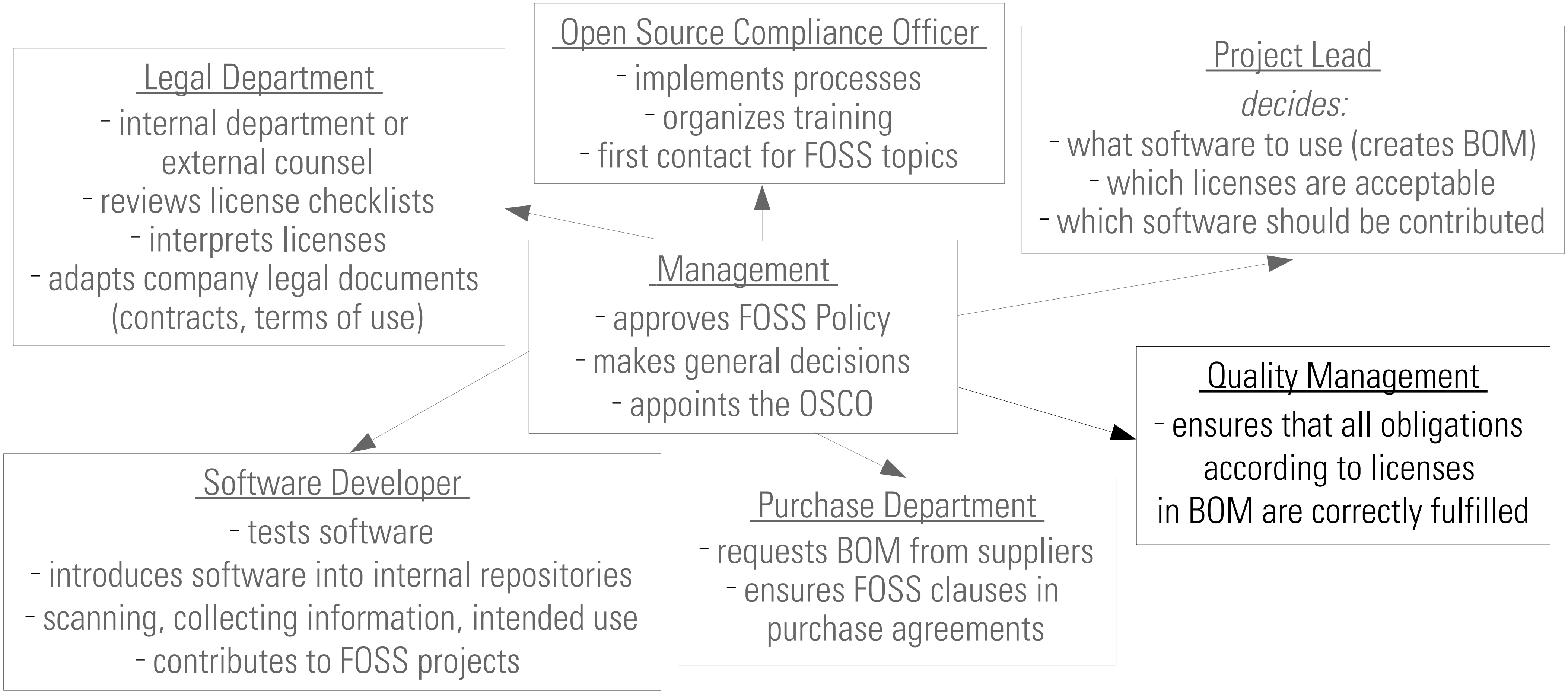
decides:
- what software to use (creates BOM)
- which licenses are acceptable
- which software should be contributed

Management
- approves FOSS Policy
- makes general decisions
- appoints the OSCO

Software Developer
- tests software
- introduces software into internal repositories
- scanning, collecting information, intended use
- contributes to FOSS projects

Purchase Department
- requests BOM from suppliers
- ensures FOSS clauses in purchase agreements

Responsibilities



Open Source Compliance Officer

- implements processes
- organizes training
- first contact for FOSS topics

Legal Department

- internal department or external counsel
- reviews license checklists
- interprets licenses
- adapts company legal documents (contracts, terms of use)

Project Lead

decides:

- what software to use (creates BOM)
- which licenses are acceptable
- which software should be contributed

Management

- approves FOSS Policy
- makes general decisions
- appoints the OSCO

Quality Management

- ensures that all obligations according to licenses in BOM are correctly fulfilled

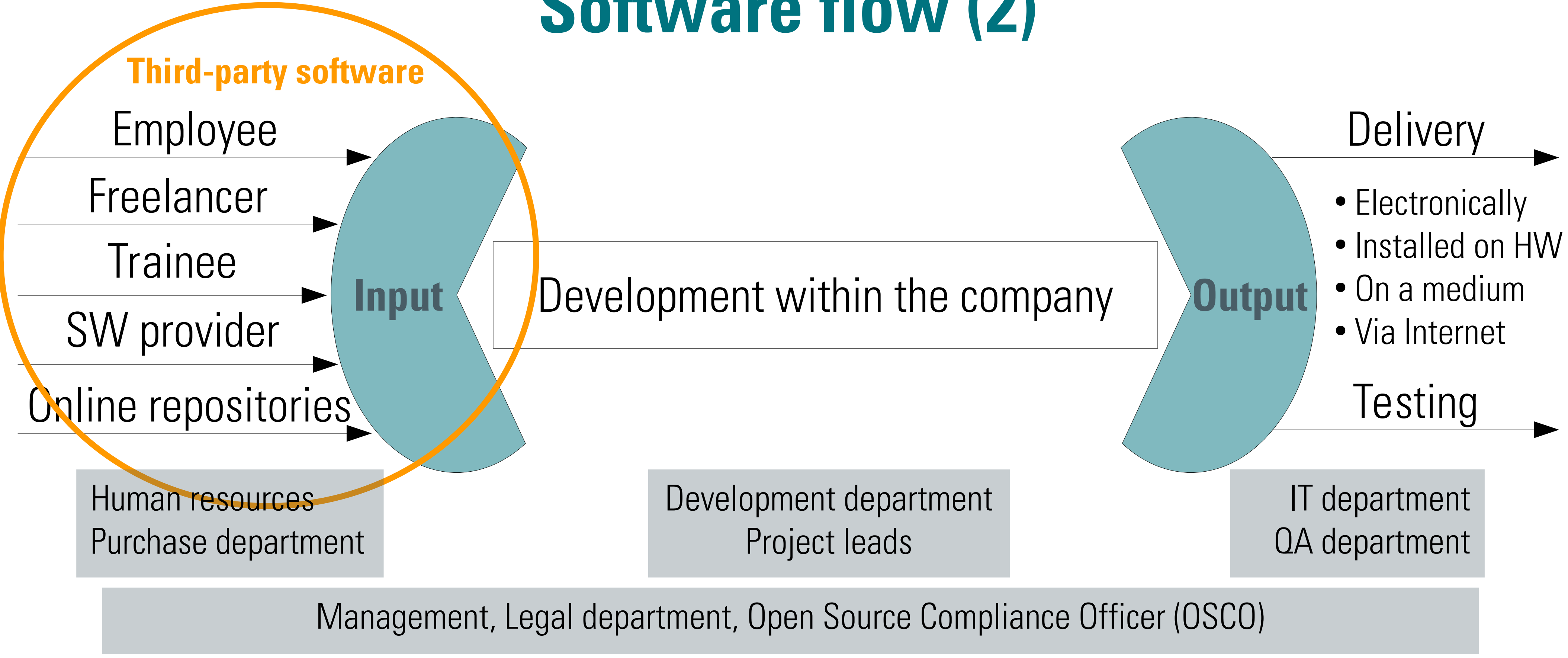
Software Developer

- tests software
- introduces software into internal repositories
- scanning, collecting information, intended use
- contributes to FOSS projects

Purchase Department

- requests BOM from suppliers
- ensures FOSS clauses in purchase agreements

Software flow (2)



Detection and analysis of third-party software (1)

- **Goals**
  - Knowing what external software is used
  - Avoiding unlicensed software
  - Basis for creating a BOM → Annex: Bill of Material
- **Categorization**

☐ *Option 1: All FOSS is treated as equal*

Taking into account future mergers and acquisitions or a change from internal use to distribution a company may be well advised to establish identical procedures for in-house and distributed FOSS.

☐ *Option 2: No rules apply to in-house FOSS*

FOSS can be used within a legal entity and shared between all employees without any restrictions or license obligations. The advantages of this approach are avoiding expenses for processes and administration. It may however be advisable for large companies to maintain a company-wide repository where all internally used FOSS is listed to bring users together and avoid a large number of various versions.

Detection and analysis of third-party software (2)

Testing for technical suitability

- All freely downloadable software may be tested if no license agreement or terms of use need to be accepted. Otherwise, the Software developer must submit a request to the legal department at *legal@company.tld* with the following information: Name of the software, download link, legal text
- The legal department approves or declines and the decision is archived at *https://intranet.company.tld/directory*
- Checking into internal repositories is not allowed without a prior approval according to the approval process outlined in this policy.

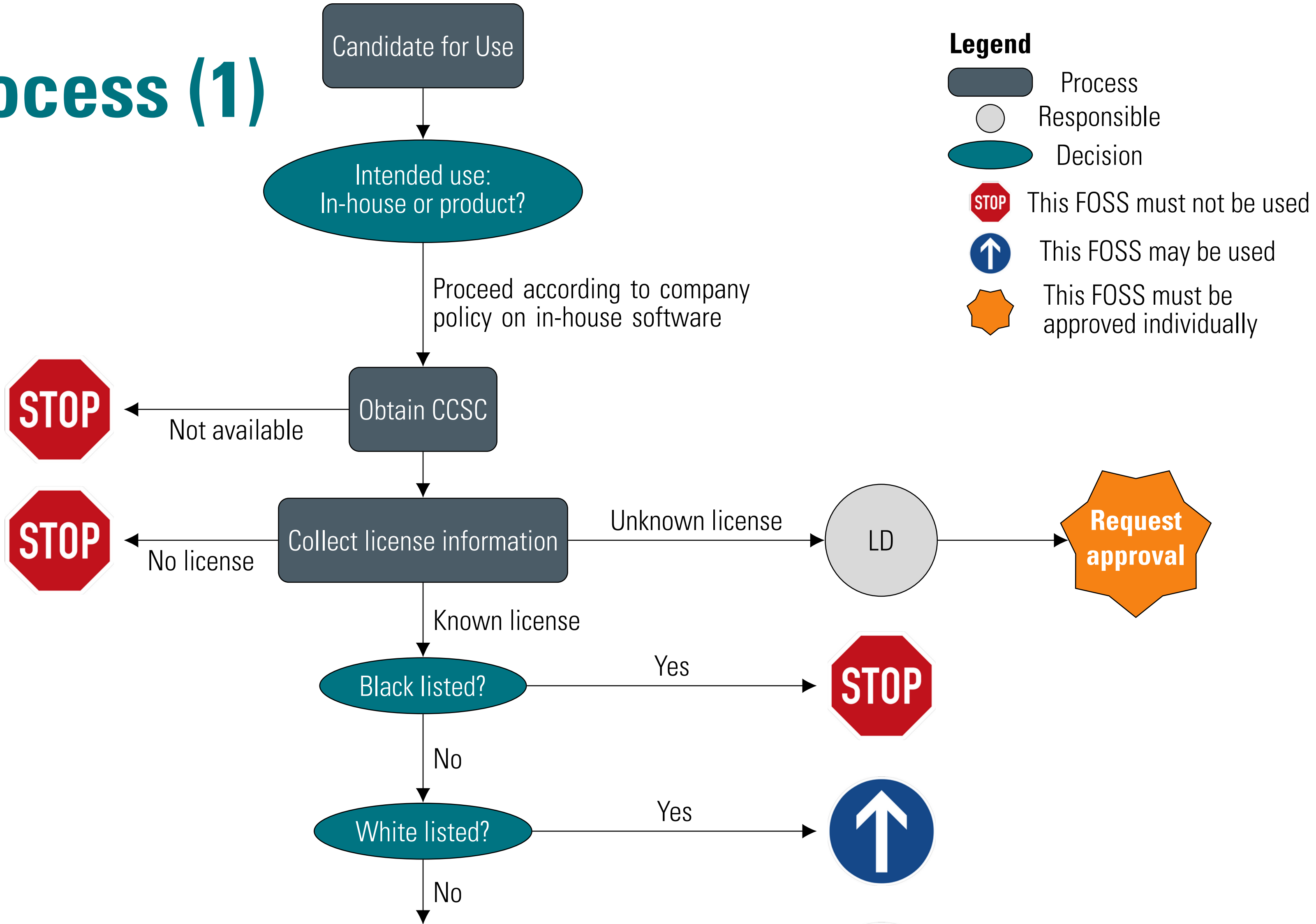
Detection and analysis of third-party software (3)

Candidate for Use → Annex: Approval process

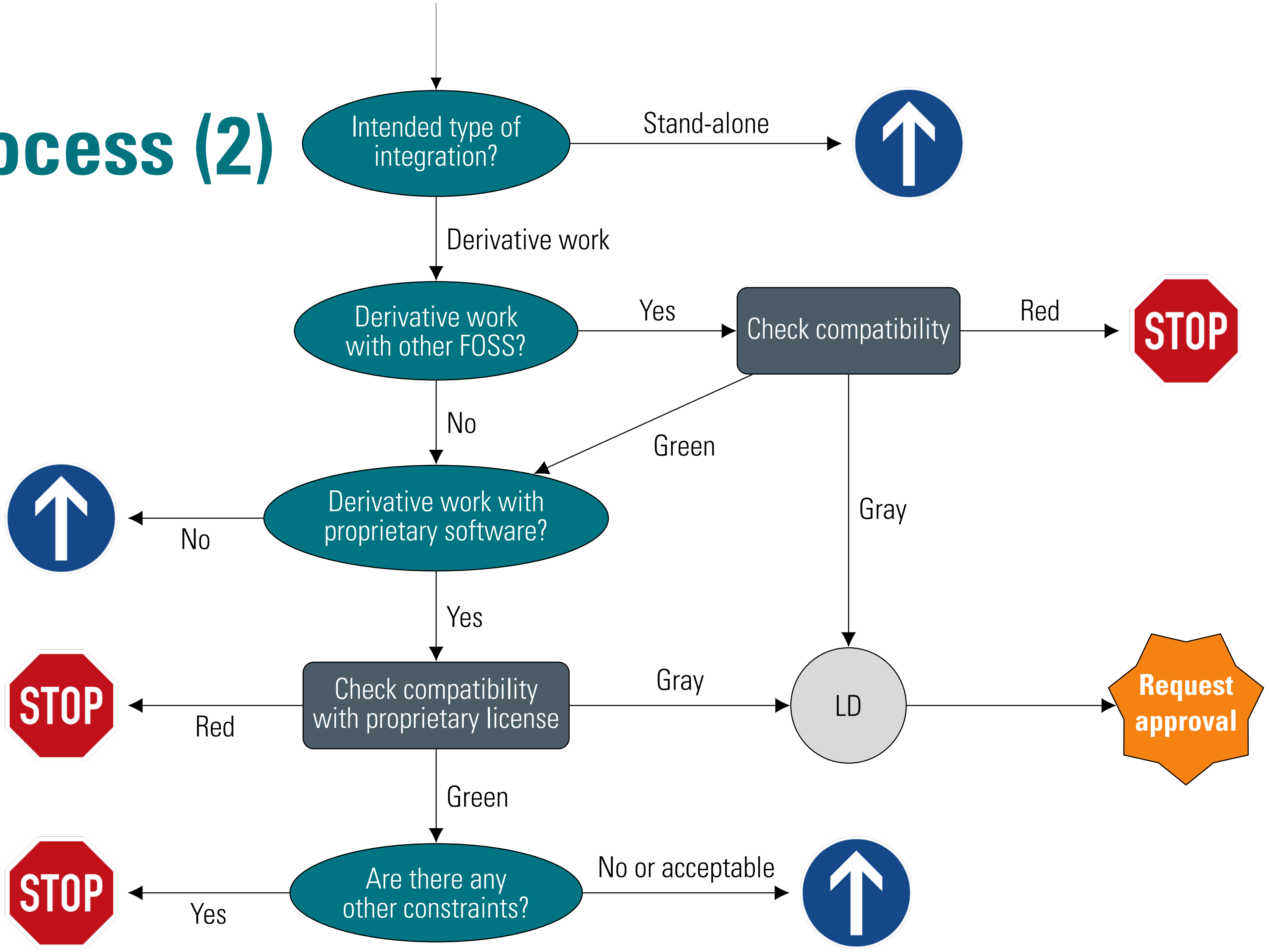
- Collect complete corresponding source code
- Extract license information → Supplement: How To Scan
- Consult → Annexes: List of known licenses, Black List, White List
- Determine intended use → Supplement: What is a derivative work?
  → Annex: Copyleft policy
- If applicable, check license compatibility → Annex: OSADL Compatibility Matrix

Providing an approval process that every employee who wants to use a particular software can follow aims to reduce the requests that need to be evaluated individually.

Annex: Approval process (1)



Annex: Approval process (2)


Intended type of integration?
Stand-alone
Derivative work
Derivative work with other FOSS?
Yes
Check compatibility
Red
STOP
No
Green
Gray
Derivative work with proprietary software?
No
Yes
Check compatibility with proprietary license
Red
STOP
Gray
LD
Request approval
Green
Are there any other constraints?
Yes
STOP
No or acceptable

Detection and analysis of third-party software (4)

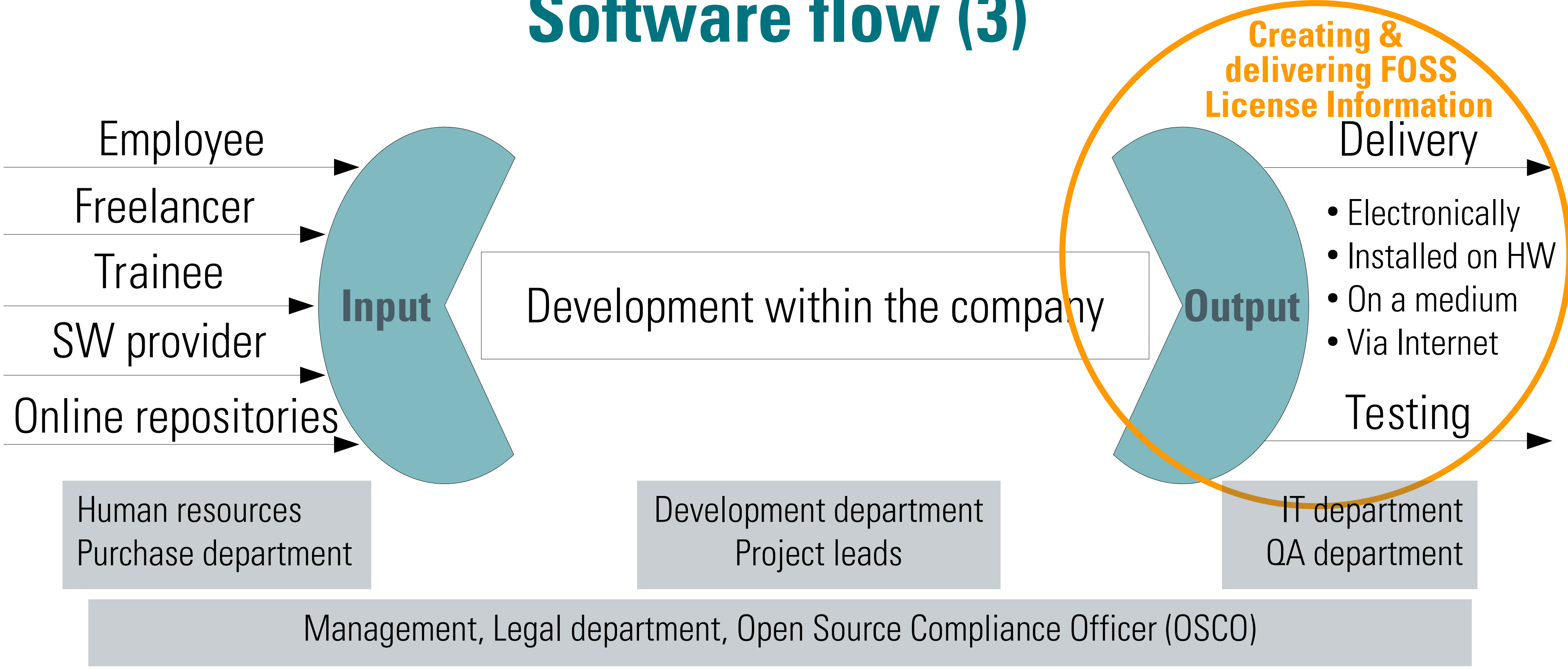
The Candidate for Use may be entered into the company's repository:

- software name and version
- complete corresponding source code
- (re)build instructions
- legal information

STOP The Candidate for Use is not suitable for use in the company's products. This information must be archived and an alternative must be found.

Request approval An individual request for approval must be submitted to the legal department → Annex: Approval request

Software flow (3)



FOSS License Information

The OSADL Open Source License Obligations Checklists help to determine what information, documentation and other material must be delivered together with the software:

- **Information obligations:** delivering license texts, copyright notices, modification notices, warranty disclaimers, acknowledgments, …
- **Disclosure obligations:** delivering or offering the complete corresponding source code and build and installation instructions
- **Licensing obligations:** adapting company documents (*e.g.* EULA and Terms of Use), licensing own development correctly if a derivative work with software under a copyleft license is created

FOSS License Information: Use cases

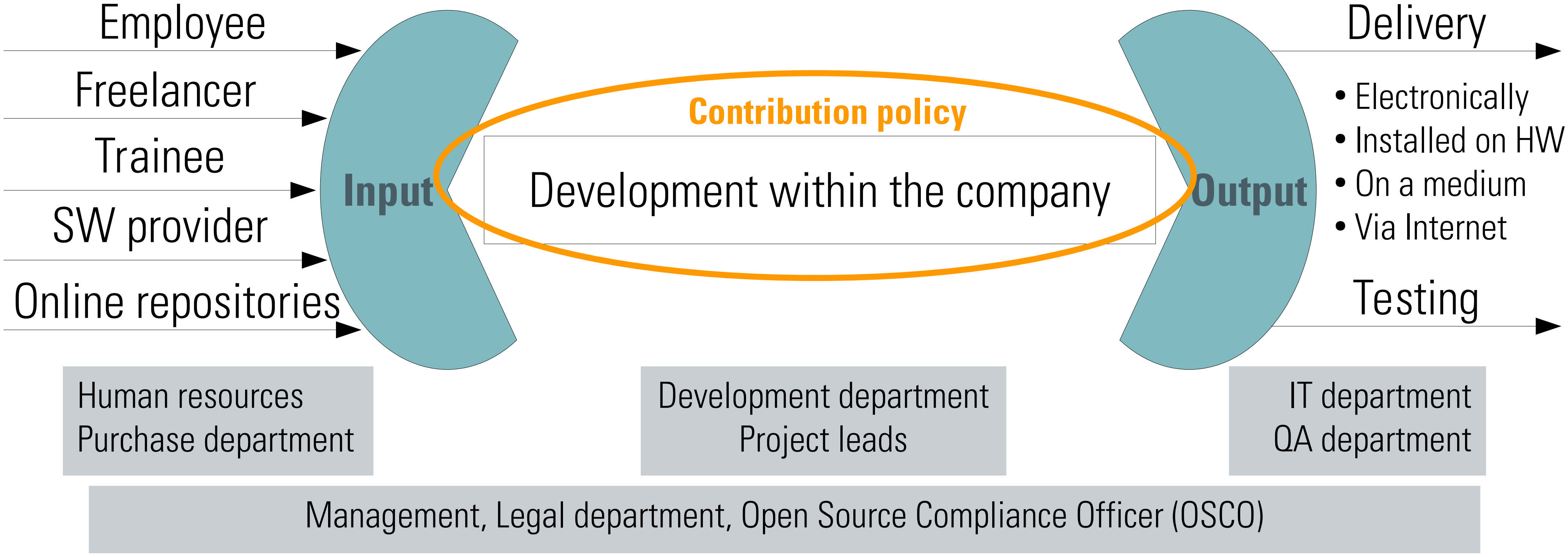
Different use cases → different aspects to be considered:

- Unmodified or modified source code
- Unmodified or modified binaries
- Software as a Service (SaaS)
- Linux kernel in an embedded system
- Entire Linux distributions
- Updates

FOSS License Information: Delivery

- Choosing a **delivery medium**
- Determining **QA process**:
  - Timing: before distribution starts
  - Compliance check: for every FOSS component listed in the BOM the License Information is checked for completeness according to the applicable checklist
- **Correcting**, if necessary
- **Releasing** the product for distribution together with FOSS License Information

Software flow (4)



Contribution to FOSS projects

When a company uses FOSS in their products, they may sooner or later decide to contribute to the FOSS projects they are using because [. . .]

- **Approval of contributor:** training and experience in programming, community etiquette, FOSS licensing, separation of private and company development
- **Approval of FOSS project:** license (copyleft, patents), software quality, reputation, Contributor License Agreements
- **Approval of contribution:** may the contained IP be published, code quality, conflicting agreements (*e.g.* NDA), third-party content, safety and security vulnerabilities

→ **Annex: Contribution permission**

Additional topics (1)

- Communication of the FOSS Policy
  - on the Intranet at *https://intranet.company.tld/FOSS-Policy.pdf*
  - additional provisions in employment contracts:

> [. . .] The employee is obliged to take note of and follow the employer’s FOSS Policy immediately after taking up his or her duties. [...]

- Own FOSS projects
  → Supplement: Selecting a FOSS license

Additional topics (2)

- Audits and certification
  - OpenChain conformance

> The OpenChain specification provides guidelines on what must be done to be able to distribute FOSS compliantly rather than giving specific instructions on processes. So far, conformance with these guidelines is assessed in self-evaluation. [. . .]

  - OSADL License Compliance Audit (LCA)

> The OSADL LCA aims to support companies to organize the compliant distribution of Linux-based embedded systems. It focuses on a specific product and in particular on the Linux kernel under the GNU General Public License (GPL-2.0) and the C library, usually the GNU C library under the GNU Lesser General Public License (LGPL-2.1). [. . .]

- Patent considerations:
  - all FOSS licenses require licensing implemented patents
  - Open Invention Network (OIN) and License on Transfer (LOT) Network

Build your own FOSS Policy

- The OSADL Open Source Policy Template is intended as a basis for companies to create their individual FOSS policy
- The document offers the possibility to remove all explanation boxes, select options and fill out placeholder texts
- Alternatively, particular sections may be copied into a company's own document and maintained locally

Build your own FOSS Policy

- Unpack the archive "OSADL-OS-Policy.tgz"
- Open "OS-Policy_Master.pdf" with your favorite PDF reader to read the Open Source Policy Template with all explanation boxes included
- Annexes and Supplements are linked from the main document or separately available in the respective directories
- Open "OS-Policy_Master-editable.pdf"/"OS-Policy_Master-editable-noexplanations.pdf" with your favorite PDF editor (*e.g.* commercial PDF Studio) to create your own FOSS policy with all explanation boxes included/removed
- Select options and fill out placeholders; if desired, customize the header image with your company logo
- Save the file to freeze your entries while keeping links alive

Disclaimer

Implementing a FOSS policy requires

- legal expertise
- (professional and legal) decision competence for the company

The template does not replace these.

Supplement: How to Scan

- **Informational software scanning**
  - **Extracting** license, copyright and other relevant **notices** from source code.
  - Plain text information
- **Forensic software scanning**
  - **Discovering** third-party **software snippets** incorporated into source code.
  - Snippets may have been used maliciously or negligently and are **not licensed** correctly.
  - "Finger prints" of suspicious software are matched against large data bases of respective finger prints of known software components.
  - Information is not obvious, but hidden or even obfuscated.

Informational *vs* forensic scanning

Scanning	Effort	Duration	Needed by everybody?	Examples
Informational scanning	Relatively small	Minutes/hours	Probably yes	Grep, Nomos, Fossology, Scancode
Forensic scanning	Very big	Days/weeks	No, not necessarily	Black Duck, Palamida/Flexera, BANG

Informational scanning

- **Scan tools** (*e.g.* Scancode, Nomos, Monk) search source code text for **keywords** such as *Copyright, license, (c)* :
- Fuzzy algorithms are used to account for spelling and formatting variants
- Mostly, some manual modifications are needed
- A **license management tool** (*e.g.* FOSSology) stores the scanning results and manual completions in a company-wide database.
- Information can then be retrieved for various conditions of use.

Scanning and beyond ...

- Source code administration
  - Comprehensive table of licenses in use
  - History of licenses, documentation of license changes
  - Hints to obligations of detected licenses
  - Evaluation of license compatibility
- Batch-Processing
  - License scanning integrated into tool chain and build processes
  - Alerts (e.g. via email), if critical change detected
  - Documentation as a proof of implemented license compliance

BTW: Why do we need scanning?

GPL-2.0 Section 1:

```
 1. You may copy and distribute verbatim copies of the Program's source code
as you receive it, in any medium, provided that you conspicuously and
appropriately publish on each copy an appropriate copyright notice and
disclaimer of warranty; keep intact all the notices that refer to this
License and to the absence of any warranty; and give any other recipients of
the Program a copy of this License along with the Program.
```

Fulfill obligation „Publish copyright notice(s)"

- Challenges
  - **Formal presentation not specified:**
    `Copyright © 2019 Employer LLC, author John Doe`
    could have been written as
    `Owned by Employer LLC, written by John Doe`
  - **Possible large number of copyright holders and authors**
    At the time when some licenses were created, there were no large communities of distributed software development with more than thousand developers.

Formal presentation of copyright not specified

Other sources of information may need to be consulted such as the file „MAINTAINERS“ of the Linux kernel:

Formal presentation of copyright not specified

Other sources of information may need to be consulted such as the file „MAINTAINERS“ of the Linux kernel:

Under GPL-2.0

Formal presentation of copyright not specified

Other sources of information may need to be consulted such as the file „MAINTAINERS“ of the Linux kernel:

Step #1: Building a list of authors:

```
# grep "^M:" MAINTAINERS | sed 's/^M:[\x09 ]*//' | cut "-d<"
-f1 | tr -d '"' | grep -v @ | sort | uniq >maintainers
```

Formal presentation of copyright not specified

Other sources of information may need to be consulted such as the file „MAINTAINERS“ of the Linux kernel:

Step #1: Building a list of authors:

```
# grep "^M:" MAINTAINERS | sed 's/^M:[\x09 ]*//' | cut "-d<"
-f1 | tr -d '"' | grep -v @ | sort | uniq >maintainers
```

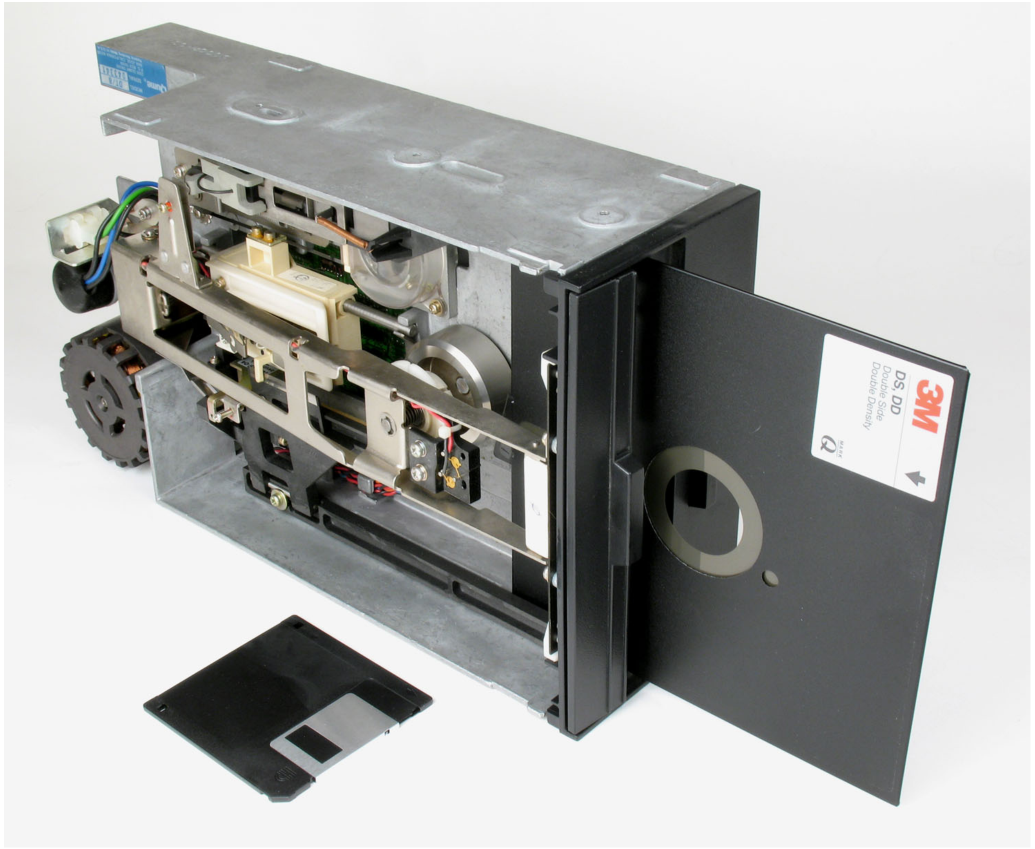
Step #2: Searching for authors and formal descriptors:

```
# grep -ir -f maintainers -e "copyright.*[12][90][0-9][0-9]"
-e "(c).*[12][90][0-9][0-9]" . | grep -v -e _AUTHOR -e
^./Documentation -e ^./tools -e ^./samples -e ^./patch -e
^./.git -e ^./.pc -e ./MAINTAINERS: >copyright-notices
```

A storage medium used to look like that

8-inch floppy disk:

- 1,6 MByte unformatted data
- 1,2 MByte formatted data
- 500 Kbit/s data transfer
- Few authors
- More than 400 cm² area for copyright notices



A storage medium may look like that today

For example USB storage:

- Up to 1 TByte capacity
- Up to 100 MByte/s data transfer
- Possibly more than thousand authors
- Very little area for copyright notices

Large number of copyright holders and authors

- The file `copyright-notices` (Linuxkernel 5.2.21-rt13):

```
# wc -l copyright-notices
65141 copyright-notices
```

- As normal text document printed in 12 pt:
  1595 pages
- Can only be forwarded in electronic media
- Document cannot be created manually
- Instead of a self-made script (as in our example) established tools must be used.

Large number of copyright holders and authors

- The file `copyright-notices` (Linuxkernel 5.2.21-rt13):

```
# wc -l copyright-notices
65141 copyright-notices
```

- As normal text document printed in 12 pt:
  1595 pages
- Can only be forwarded in electronic media
- Document cannot be created manually
- Instead of a self-made script (as in our example) established tools must be used.

This is "Informational Scanning"

The file copyright-notices, page #1

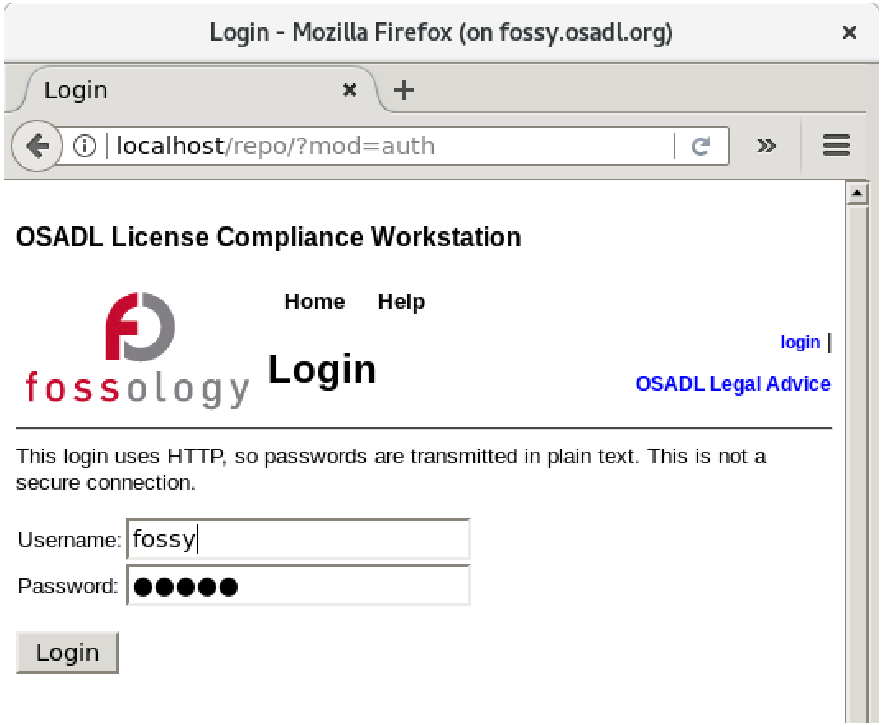
```
./kernel/softirq.c: * Copyright (C) 1992 Linus Torvalds
./kernel/futex.c: * (C) Rusty Russell, IBM 2002
./kernel/futex.c: * (C) Copyright 2003 Red Hat Inc, All Rights Reserved
./kernel/futex.c: * (C) Copyright 2003, 2004 Jamie Lokier
./kernel/futex.c: * (C) Copyright 2006 Red Hat Inc, All Rights Reserved
./kernel/futex.c: * Thanks to Thomas Gleixner for suggestions, analysis and fixes.
./kernel/futex.c: * PI-futex support started by Ingo Molnar and Thomas Gleixner
./kernel/futex.c: * Copyright (C) 2006 Red Hat, Inc., Ingo Molnar <mingo@redhat.com>
./kernel/futex.c: * Copyright (C) 2006 Timesys Corp., Thomas Gleixner <tglx@timesys.com>
./kernel/futex.c: * Copyright (C) 2007 Eric Dumazet <dada1@cosmosbay.com>
./kernel/futex.c: * Requeue-PI support by Darren Hart <dvhltc@us.ibm.com>
./kernel/futex.c: * Copyright (C) IBM Corporation, 2009
./kernel/futex.c: * Thanks to Thomas Gleixner for conceptual design and careful reviews.
./kernel/irq/resend.c: * Copyright (C) 1992, 1998-2006 Linus Torvalds, Ingo Molnar
./kernel/irq/resend.c: * Copyright (C) 2005-2006, Thomas Gleixner
./kernel/irq/affinity.c: * Copyright (C) 2016 Thomas Gleixner.
./kernel/irq/affinity.c: * Copyright (C) 2016-2017 Christoph Hellwig.
./kernel/irq/autoprobe.c: * Copyright (C) 1992, 1998-2004 Linus Torvalds, Ingo Molnar
./kernel/irq/debugfs.c:// Copyright 2017 Thomas Gleixner <tglx@linutronix.de>
./kernel/irq/dummychip.c: * Copyright (C) 1992, 1998-2006 Linus Torvalds, Ingo Molnar
./kernel/irq/dummychip.c: * Copyright (C) 2005-2006, Thomas Gleixner, Russell King
```

The file copyright-notices, page #1595

```
./fs/affs/dir.c: * (c) 1996 Hans-Joachim Widmaier - Rewritten
./fs/affs/dir.c: * (C) 1993 Ray Burr - Modified for Amiga FFS filesystem.
./fs/affs/dir.c: * (C) 1992 Eric Youngdale Modified for ISO 9660 filesystem.
./fs/affs/dir.c: * (C) 1991 Linus Torvalds - minix filesystem
./fs/affs/inode.c: * (c) 1996 Hans-Joachim Widmaier - Rewritten
./fs/affs/inode.c: * (C) 1993 Ray Burr - Modified for Amiga FFS filesystem.
./fs/affs/inode.c: * (C) 1992 Eric Youngdale Modified for ISO9660 filesystem.
./fs/affs/inode.c: * (C) 1991 Linus Torvalds - minix filesystem
./fs/affs/file.c: * (c) 1996 Hans-Joachim Widmaier - Rewritten
./fs/affs/file.c: * (C) 1993 Ray Burr - Modified for Amiga FFS filesystem.
./fs/affs/file.c: * (C) 1992 Eric Youngdale Modified for ISO 9660 filesystem.
./fs/affs/file.c: * (C) 1991 Linus Torvalds - minix filesystem
./fs/affs/super.c: * (c) 1996 Hans-Joachim Widmaier - Rewritten
./fs/affs/super.c: * (C) 1993 Ray Burr - Modified for Amiga FFS filesystem.
./fs/affs/super.c: * (C) 1992 Eric Youngdale Modified for ISO 9660 filesystem.
./fs/affs/super.c: * (C) 1991 Linus Torvalds - minix filesystem
./LICENSES/preferred/LGPL-2.0:Copyright (C) 1991 Free Software Foundation, Inc.
./LICENSES/preferred/LGPL-2.1:Copyright (C) 1991, 1999 Free Software Foundation, Inc.
./LICENSES/preferred/GPL-2.0: Copyright (C) 1989, 1991 Free Software Foundation, Inc.
./LICENSES/deprecated/GPL-1.0: Copyright (C) 1989 Free Software Foundation, Inc.
./LICENSES/deprecated/X11:Copyright (C) 1996 X Consortium
```

Fossology

Web based license management



Login - Mozilla Firefox (on fossy.osadl.org)

Login

localhost/repo/?mod=auth

OSADL License Compliance Workstation

Home Help

Login

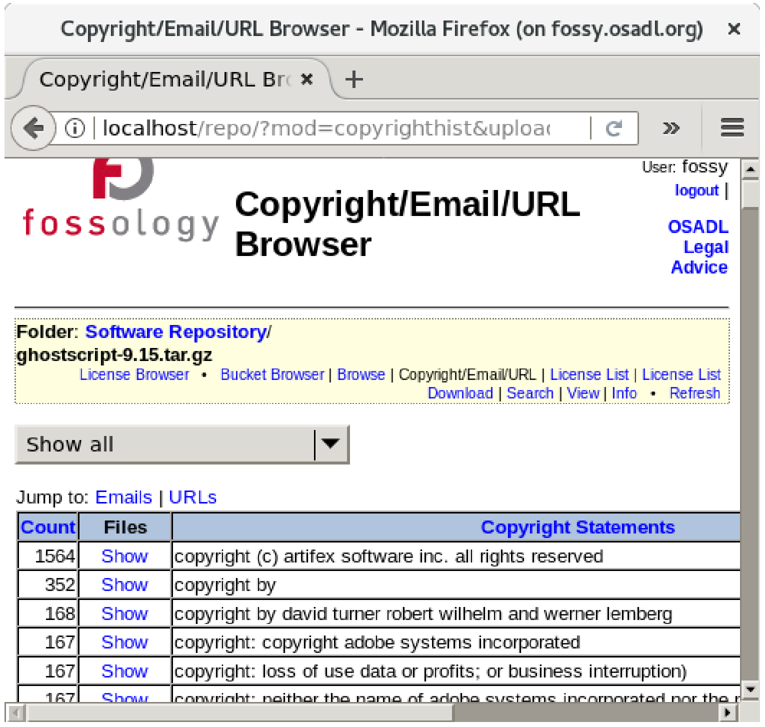
login |

OSADL Legal Advice

This login uses HTTP, so passwords are transmitted in plain text. This is not a secure connection.

Username:	fossy
Password:	●●●●●

Login



Copyright/Email/URL Browser - Mozilla Firefox (on fossy.osadl.org)

Copyright/Email/URL Br

localhost/repo/?mod=copyrighthist&uploa

User: fossy
logout |

Copyright/Email/URL Browser

OSADL Legal Advice

Folder: Software Repository/
ghostscript-9.15.tar.gz
License Browser • Bucket Browser | Browse | Copyright/Email/URL | License List | License List Download | Search | View | Info • Refresh

Show all

Jump to: Emails | URLs

Count	Files	Copyright Statements
1564	Show	copyright (c) artifex software inc. all rights reserved
352	Show	copyright by
168	Show	copyright by david turner robert wilhelm and werner lemberg
167	Show	copyright: copyright adobe systems incorporated
167	Show	copyright: loss of use data or profits; or business interruption)
167	Show	copyright: neither the name of adobe systems incorporated nor the

Scancode (command line tool)

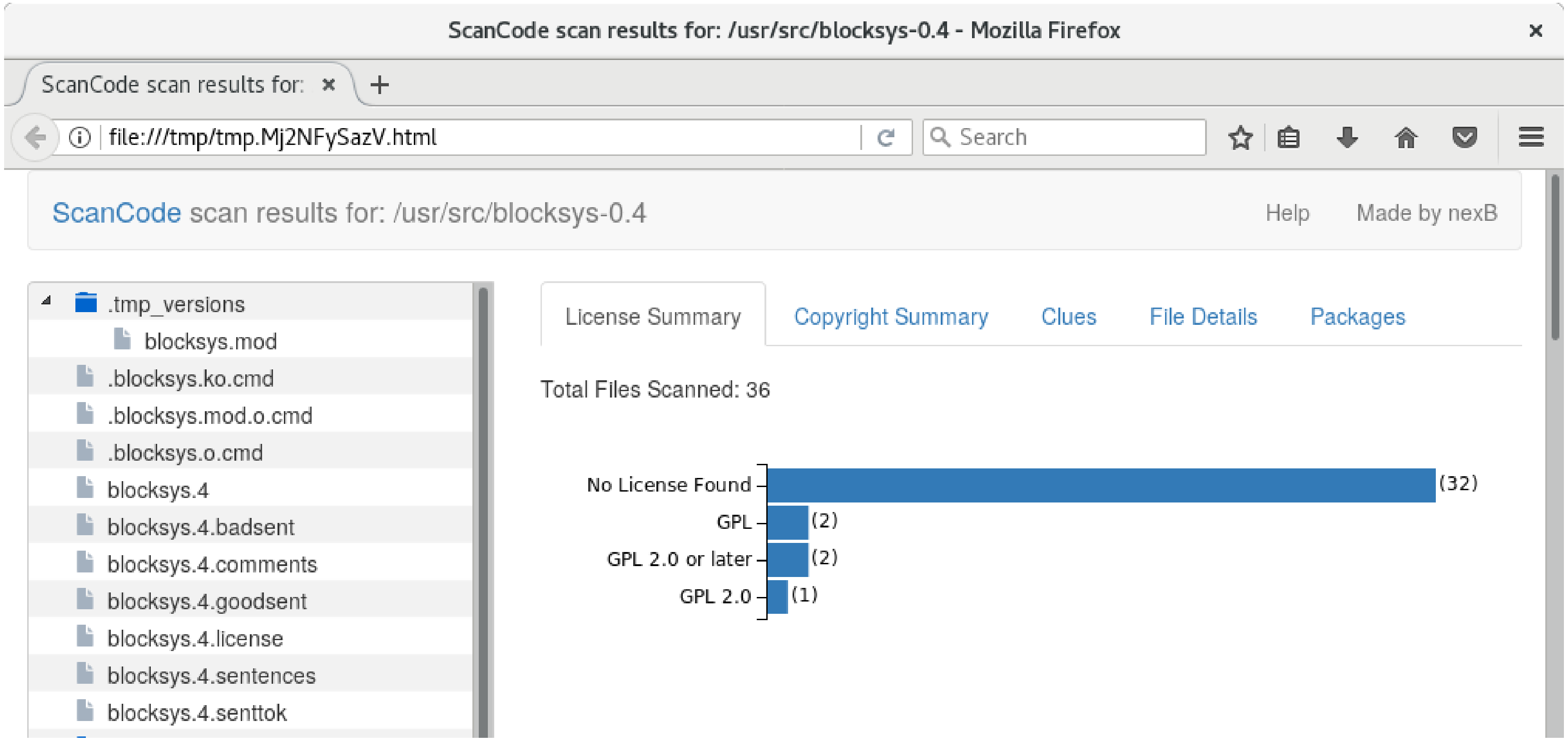
```
Usage: scancode [OPTIONS] <input> <output_file>

  scan the <input> file or directory for origin clues and license and save results to the <output_file>.

  The scan results are printed to stdout if <output_file> is not provided. Error and progress is printed to stderr.

Options:
  -c, --copyright          Scan <input> for copyrights. [default]
  -l, --license            Scan <input> for licenses. [default]
  -p, --package            Scan <input> for packages. [default]
  -e, --email              Scan <input> for emails.
  -u, --url                Scan <input> for urls.
  -i, --info               Include information such as size, type, etc.
  --license-score INTEGER  Do not return license matches with scores lower than this score. A number between 0 and 100.
                           [default: 0]
  --license-text           Include the detected licenses matched text. Has no effect unless --license is requested.
  -f, --format <style>     Set <output_file> format <style> to one of the standard formats: json or json-pp or html or
                           html-app or spdx-tv or spdx-rdf or the path to a custom template  [default: json]
  --verbose                Print verbose file-by-file progress messages.
  --quiet                  Do not print summary or progress messages.
  -n, --processes INTEGER  Scan <input> using n parallel processes.  [default: 1]
  -h, --help               Show this message and exit.
  --examples               Show command examples and exit.
  --about                  Show information about ScanCode and licensing and exit.
  --version                Show the version and exit.
  --diag                   Include additional diagnostic information such as error messages or result details.
  --timeout INTEGER        Stop scanning a file if scanning takes longer than a timeout in seconds.  [default: 120]
  --max-memory INTEGER     Stop scanning a file if scanning requires more than a maximum amount of memory in megabytes.
                           [default: 1000]
```

Scancode (result formatted for browser)



ScanCode scan results for: /usr/src/blocksys-0.4 - Mozilla Firefox

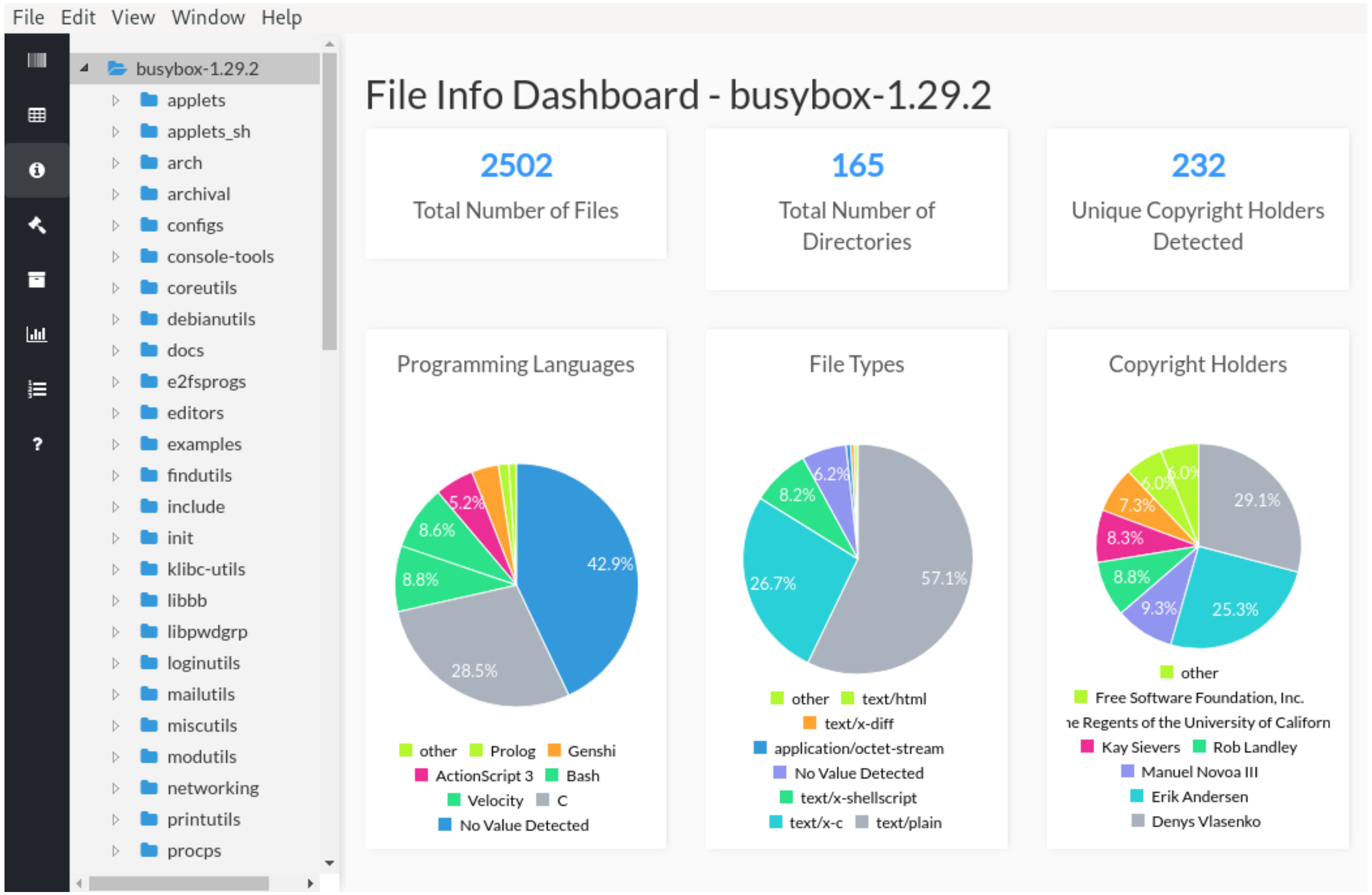
ScanCode scan results for: /usr/src/blocksys-0.4

.tmp_versions
blocksys.mod
.blocksys.ko.cmd
.blocksys.mod.o.cmd
.blocksys.o.cmd
blocksys.4
blocksys.4.badsent
blocksys.4.comments
blocksys.4.goodsent
blocksys.4.license
blocksys.4.sentences
blocksys.4.senttok

Total Files Scanned: 36

No License Found	(32)
GPL	(2)
GPL 2.0 or later	(2)
GPL 2.0	(1)

Scancode (workbench)


File Edit View Window Help

- busybox-1.29.2
  - applets
  - applets_sh
  - arch
  - archival
  - configs
  - console-tools
  - coreutils
  - debianutils
  - docs
  - e2fsprogs
  - editors
  - examples
  - findutils
  - include
  - init
  - klibc-utils
  - libbb
  - libpwdgrp
  - loginutils
  - mailutils
  - miscutils
  - modutils
  - networking
  - printutils
  - procps

File Info Dashboard - busybox-1.29.2

2502	165	232
Total Number of Files	Total Number of Directories	Unique Copyright Holders Detected

Programming Languages

🟩 other 🟩 Prolog 🟧 Genshi
🟥 ActionScript 3 🟩 Bash
🟩 Velocity ⬜ C
🟦 No Value Detected

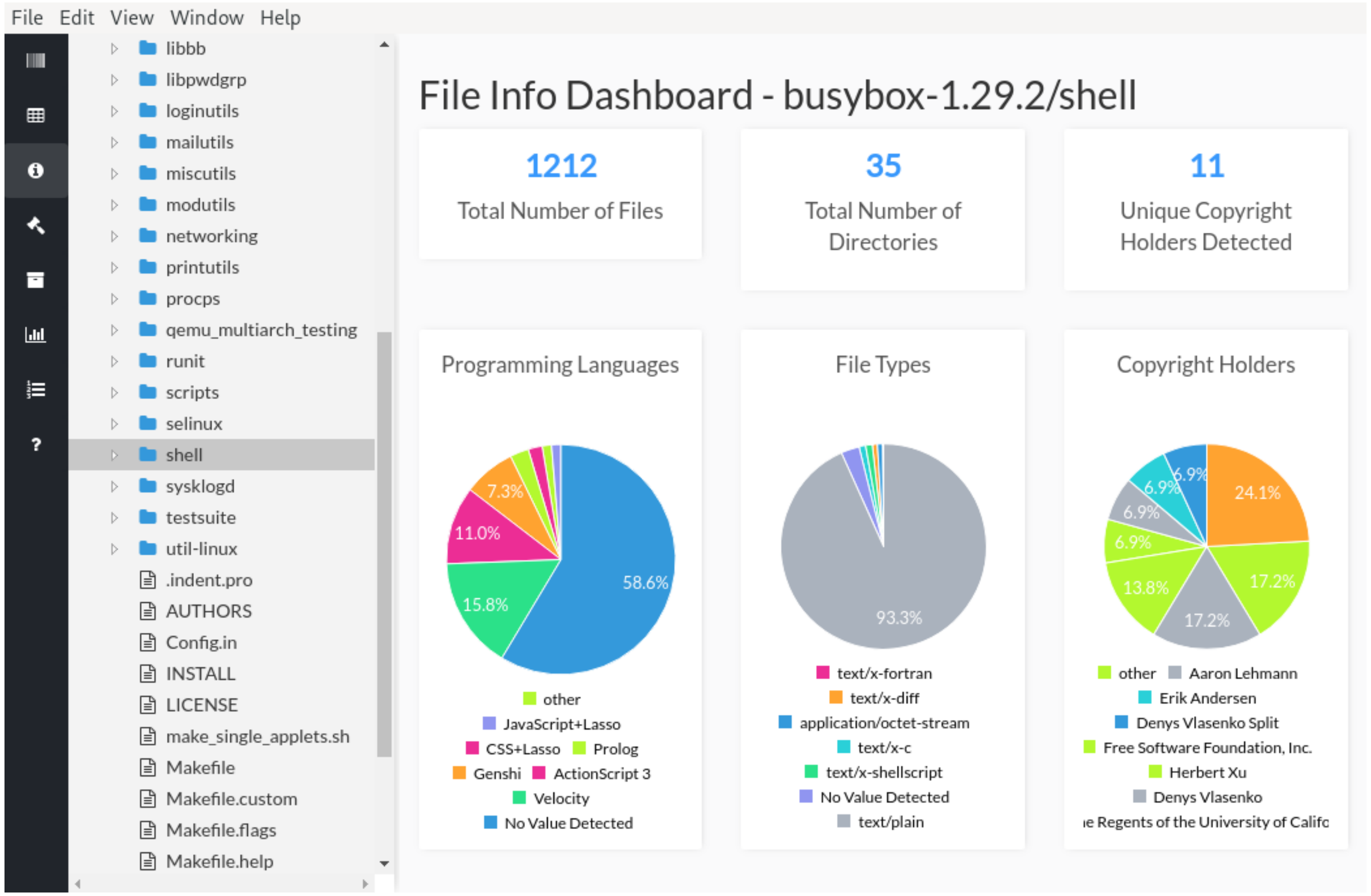
File Types

🟩 other 🟩 text/html
🟧 text/x-diff
🟦 application/octet-stream
🟪 No Value Detected
🟩 text/x-shellscript
🟦 text/x-c ⬜ text/plain

Copyright Holders

🟩 other
🟩 Free Software Foundation, Inc.
ıe Regents of the University of Californ
🟥 Kay Sievers 🟩 Rob Landley
🟪 Manuel Novoa III
🟦 Erik Andersen
⬜ Denys Vlasenko

Scancode (workbench)



File Edit View Window Help

- libbb
- libpwdgrp
- loginutils
- mailutils
- miscutils
- modutils
- networking
- printutils
- procps
- qemu_multiarch_testing
- runit
- scripts
- selinux
- shell
- sysklogd
- testsuite
- util-linux
- .indent.pro
- AUTHORS
- Config.in
- INSTALL
- LICENSE
- make_single_applets.sh
- Makefile
- Makefile.custom
- Makefile.flags
- Makefile.help

File Info Dashboard - busybox-1.29.2/shell

1212 Total Number of Files

35 Total Number of Directories

11 Unique Copyright Holders Detected

Programming Languages

other
JavaScript+Lasso
CSS+Lasso Prolog
Genshi ActionScript 3
Velocity
No Value Detected

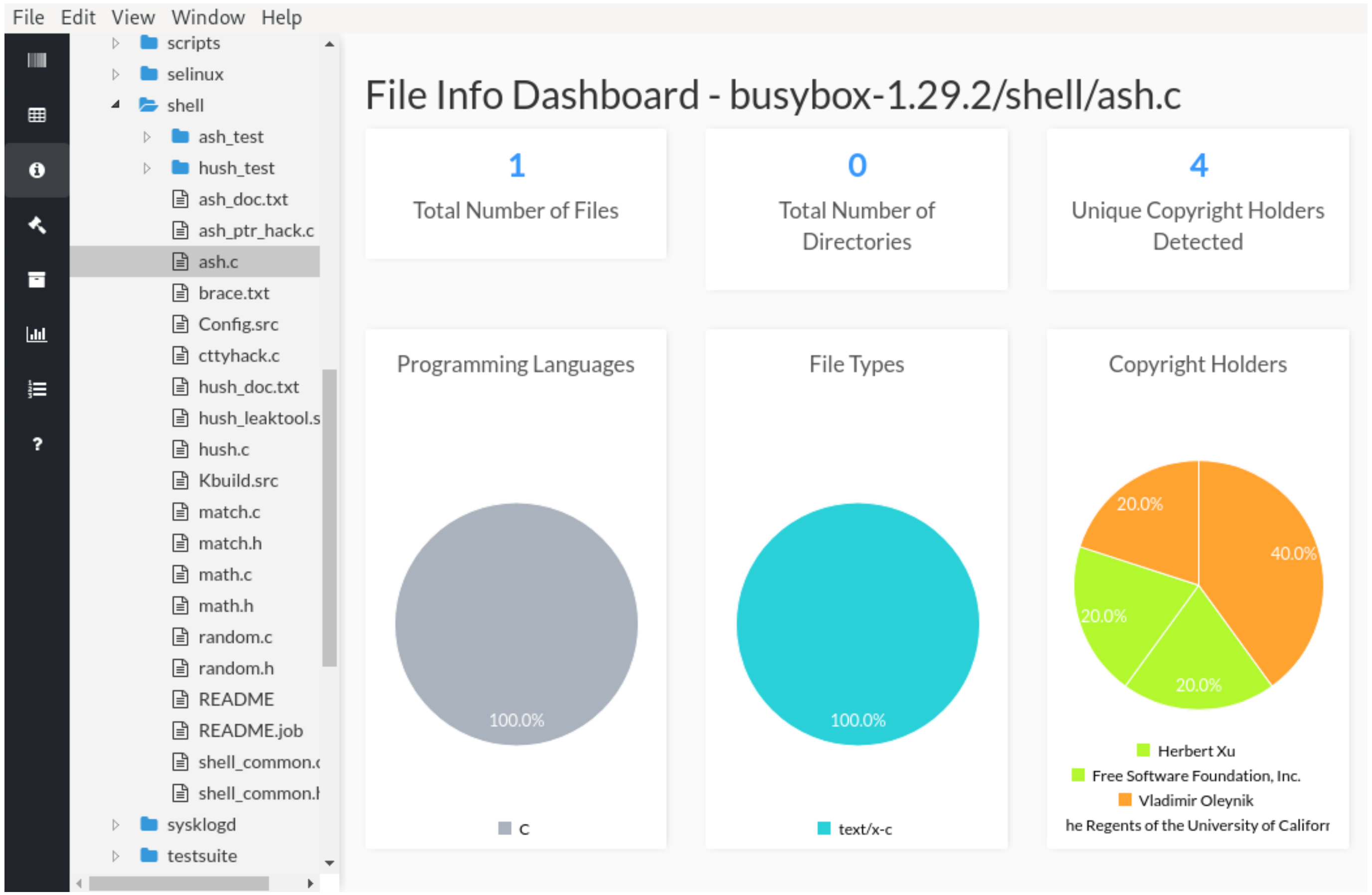
File Types

text/x-fortran
text/x-diff
application/octet-stream
text/x-c
text/x-shellscript
No Value Detected
text/plain

Copyright Holders

other Aaron Lehmann
Erik Andersen
Denys Vlasenko Split
Free Software Foundation, Inc.
Herbert Xu
Denys Vlasenko
ıe Regents of the University of Califc

Scancode (workbench)



File Edit View Window Help

- scripts
- selinux
- shell
  - ash_test
  - hush_test
  - ash_doc.txt
  - ash_ptr_hack.c
  - ash.c
  - brace.txt
  - Config.src
  - cttyhack.c
  - hush_doc.txt
  - hush_leaktool.s
  - hush.c
  - Kbuild.src
  - match.c
  - match.h
  - math.c
  - math.h
  - random.c
  - random.h
  - README
  - README.job
  - shell_common.
  - shell_common.
- sysklogd
- testsuite

File Info Dashboard - busybox-1.29.2/shell/ash.c

1	0	4
Total Number of Files	Total Number of Directories	Unique Copyright Holders Detected

Forensic scanning (1)

Step #1: Create data base and tool for forensic scanning:

- Collect each and every piece of Open Source software ever published (could be up to several hundreds of TBytes)
- Determine "meaningful" source code snippets and create hashes of them
- Store hashes along with original source code information in a data base

Forensic scanning (2)

Step #2: Use the tool to discover yet unidentified code in own software (by negligence or by fraud):

- Determine "meaningful" source code snippets of own software and create hashes of them (same procedure as with foreign code in step #1)
- Search for the hashes in the data base
- Manually check the matches and remove false positive ones (this may be labor intensive)
- Take care of the correct findings (license/remove/rewrite code)

Conclusion

- **Informational scanning** is feasible with limited effort and provides all information that normally is needed to compliantly copy and distribute Open Source software. It, therefore, is generally recommended (**"knowing your files"**).
- **Forensic scanning** usually requires a big effort, but certainly may provide crucial information, if needed. Forensic scanning, thus, should only be employed, if the individual conditions of software procurement let this appear meaningful (**"knowing your enemy"**).

What is the OSADL Scanbook?

- Standard scan tools such as Scancode and Fossology
- Armijn Hemel's Linux kernel delta scan
  - Trust *kernel.org*, but do not trust other code
  - Generate a hash data base of all original Linux code
  - Only scan code that does not belong to a valid hash
- Available as image or in a ready-to-use notebook
- Example of a license compliant generic redistribution of a Linux distribution