FOSSology: Hands On Demo
Overview: Contents

1. **Motivation License Clearing**
   The challenges today

2. **Introduction FOSSology**
   What FOSSology is designed for

3. **Automation Scenarios**
   What is going to be automated

4. **Introduction SW360**
   A basic introduction to SW360

5. **Automation Example: SW360Antenna**
   To get FOSSology running on your machine
# Schedule

<table>
<thead>
<tr>
<th>Time</th>
<th>Topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>3:15 pm</td>
<td>Motivation and Introduction</td>
</tr>
<tr>
<td>3:45 pm</td>
<td>Automation</td>
</tr>
<tr>
<td>4:00 pm</td>
<td>Introduction</td>
</tr>
<tr>
<td>4:30 pm</td>
<td>Break and Questions (15 mins)</td>
</tr>
<tr>
<td>4:45 pm</td>
<td>SW360 Introduction</td>
</tr>
<tr>
<td>5:15 pm</td>
<td>Automation Example</td>
</tr>
<tr>
<td>5:45 pm</td>
<td>End</td>
</tr>
</tbody>
</table>
FOSSology: Run on Your Own

1. **Docker:**
   
docker run -p 8081:80 fossology/fossology

2. **Vagrant**
   1. checkout source code
   2. vagrant up

3. **Packages:**
   1. go to https://github.com/fossology/fossology/releases
   2. Install packages from there

4. **Source code:**
   https://github.com/fossology/fossology/wiki/Install-from-Source
FOSSology: Motivation

Why we need to look at licenses?
What is Open Source Software Licensing?

**Basics about Licensing**
- Obligations
- Restrictions
- Rights

**Example for GPL version 2.0 (selection)**
- **Obligations**
  - Include original source, copyrights
  - Include license
- **Restrictions**
  - Disclaiming warranty
- **Rights**
  - Modify
  - Distribute

**Further reading:**
- The Linux Foundation provides a public training including basics about licensing [https://training.linuxfoundation.org/linux-courses/open-source-compliance-courses/compliance-basics-for-developers](https://training.linuxfoundation.org/linux-courses/open-source-compliance-courses/compliance-basics-for-developers)
- The TLDR Legal pages at [https://tldrlegal.com/](https://tldrlegal.com/) provide **OUTLINES** about license obligations, restrictions, rights
Why is Open Source Software Licensing?

What are the goals?

*It is about telling the software developers what to care for:*

1. **Identify obligations to fulfill, including providing for example**
   a) Credits (copyrights, prominent notice)
   b) Information about licensing
   c) Source code

2. **Check for license compatibility**
   a) Simple example: GPL version 2 and CC-BY-SA (copyleft effect examples)

3. **Be able to check desired usage**
   a) Does your business case match the licensing?
   b) Is the context of usage envisaged from the OSS publishers

4. **Be a good OSS Citizen**
## Open Source Software Licensing

<table>
<thead>
<tr>
<th>License Texts</th>
<th>License Headers</th>
<th>Licensing in Metadata</th>
<th>Prosaic Licensing</th>
<th>Copyright</th>
</tr>
</thead>
<tbody>
<tr>
<td>• GPL Text</td>
<td>• Standard header of licenses</td>
<td>• SPDX-License identifier</td>
<td>• Developer writes a text about</td>
<td>• “Copyright 2018 Software</td>
</tr>
<tr>
<td>• MIT Text</td>
<td>• Or written in own words</td>
<td>• Package metadata</td>
<td>how the component is licensed</td>
<td>Bauer”</td>
</tr>
<tr>
<td>• Etc.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Mostly only on top level</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>of distro</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### License Texts
- GPL Text
- MIT Text
- Etc.
- Mostly only on top level of distro

### License Headers
- Standard header of licenses
- Or written in own words

### Licensing in Metadata
- SPDX-License identifier
- Package metadata

### Prosaic Licensing
- Developer writes a text about how the component is licensed

### Copyright
- “Copyright 2018 Software Bauer”
Examples for Licensing – Permission Notice Variants

Permission to use, copy, modify, and distribute this software for any purpose and without fee is hereby granted, provided that this copyright and permission notice appear on all copies and supporting documentation, the name of Roaring Penguin Software Inc. not be used in advertising or publicity pertaining to distribution of the program without specific prior permission, and notice be given in supporting documentation that copying and distribution is by permission of Roaring Penguin Software Inc.

Roaring Penguin Software Inc. makes no representations about the suitability of this software for any purpose. It is provided "as is" without express or implied warranty.'

- See SPDX Id HPND
- Looks in general normal and in many occurrences like the “official“ license text
- However, for documentation, you need this text, not the reference texts
Examples for Licensing – HPND Another Example

Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of Silicon Graphics not be used in advertising or publicity pertaining to distribution of the software without specific prior written permission.

Silicon Graphics makes no representation about the suitability of this software for any purpose. It is provided "as is" without any express or implied warranty. SILICON GRAPHICS DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS …
Examples for Licensing – Rare OSS Licenses

1. Jabber Open Source License Version 1.0 “This Jabber Open Source License (the "License") applies to Jabber Server and related software products as well as any updates…”

2. Arphic Public License: “Copyright (C) 1999 Arphic Technology Co., Ltd. 11Fl. No.168, Yung Chi Rd., Taipei, 110 Taiwan All rights reserved except as specified below…”

3. The Catharon Open Source LICENSE: “2000-Jul-04 Copyright (C) 2000 by Catharon Productions, Inc. Introduction This license applies to source files distributed by Catharon:”

There are (non-permissive) license texts occurring rarely
Thus, they are not covered by common license scanners
These examples maybe not part of the SPDX license list
If they are covered by the SPDX license list, there will be more in future …
Examples for Licensing – Evaluation License

License for Evaluation Purposes. *** *** hereby grants you a fully-paid, non-exclusive, non-transferable, worldwide, limited license (without the right to sublicense), under *** ***’s applicable intellectual property rights to view, download, use and reproduce the Specification only for the purpose of internal evaluation.

• Such licenses just slipped into the open source code
• Cannot be used in a commercial application
• Contacting the copyright holder required
Examples for Licensing – No License

This software is the confidential and proprietary information of *** ***, Inc. ("Confidential Information").

You shall not disclose such Confidential Information and shall use it only in accordance with the terms of the license agreement you entered into with ***. *** MAKES NO REPRESENTATIONS OR WARRANTIES ABOUT THE SUITABILITY OF THE …

- Sometimes confidential code slipped into open source
- Maybe it was by accident
- But it must be avoided in own distribution
Examples for Licensing – Hard Restrictions

The firmware this driver downloads into the *** card is a separate program and is not GPL'd source code, even though the Linux side driver and the routine that loads this data into the card are.

This firmware is licensed to you strictly for use in conjunction with the use of *** *** *** adapters. There is no warranty expressed or implied about its fitness for any purpose.

- OSS means that there are not restriction for using this
- Actually, this license is not an OSS license then
- Usage maybe possible, but needs to be checked
Examples for Licensing – Hard Restriction Camouflage

François Loizos

Another real world example:

• It is actually based on an MIT license text
• MIT license: very popular and permissive
• Added two conditions inside the original license text
  • (not so permissive)
• Very hard to identify with regular expression matching
License Analysis of a Component - Summary

1. **Overall goals**
   a) Mitigate risk
   b) Help the engineering with definitive instructions
   c) Building a list of reusable assets, requires usage independent clearing of component

2. **How to perform a component analysis for licensing**
   a) Reviewing file notices
   b) Reviewing license texts
   c) Determining the exact text for obligations (rights, restrictions)
   d) Identify new licenses
   e) License clarification required?
      ▪ OSS Expert group available in organisation
      ▪ Legal advice required

- The FOSSology project enables the tool-based identification of licenses:
  • Finding license relevant texts
  • Highlighting text occurrences
  • Identifying wording differences compared with reference texts
  • Searching for licensing phrases
  • Aggregation in a hierarchy
  • Reporting of found licenses
FOSSology: Introduction

FOSSology, a Linux Foundation Collaboration Project
It is about finding licenses

Finding Licenses

- License texts
- References to licenses
- Written texts explaining licensing
- License relevant statements
The Example

- (Apache projects are known for homogenous licensing)
- Such a project declares its licensing on its Website
- However, Open Source implies using other open source
- As such, a projects can contain also parts from other open source projects
Open Source and Reuse

- It is natural that an OSS project reuses available
  https://github.com/fossology/fossology
- Likely OSS from other projects is found
- For example, FOSSology will find 25 other licensing relevant text occurrences in Apache thrift
Using FOSSology with this Example

Another Example: Linux Kernel Project

- Looking at the Linux kernel will result in thousands of files containing license relevant information
- Analyzing them yields about 30 distinctive licensing statements

![FOSSology License Browser](image)
What is FOSSology?

A Web server application for license and copyright compliance of software components.

FOSSology Project
https://www.fossology.org/

- Published first in 2008, GPL-2.0
- 2015: Linux Foundation collaboration project
- Web server based and command line interfaces
- Scanning agents searching for license and copyright relevant hits (and more …)
- A multi-user / multi-tenant Web UI for review organizing clearing job

FOSSology Development
https://www.github.com/fossology/fossology

- Standard Web application stack:
  - Linux, Apache 2, PostgreSQL, PHP,
- Web-based UI in PHP, but scanners written in C / C++
- Two ways to interact:
  - Web user interface
  - Command line utilities
How does FOSSology work? – Overview 1 of 2

See more details the Basic Workflow Description: https://www.fossology.org/get-started/basic-workflow

- Upload OSS Package
  - Upload an open source package to the server
  - Select scan agents that analyze the software

- Review and Adjust (“Clearing”)
  - Review what scanners have found
  - Review license occurrences and correct findings if necessary

- Generate
  - Generate report output
  - For example list of licenses or SPDX
How does FOSSology work? – Overview 2 of 2

- Uploading source code archive (*.zip, *.tar.gz, etc)
- Agents scan for license relevant text
- Copyrights, Export Control (ECC), your keywords to look for etc.
- Review scanner results for wrong license classification
- Review other scanner findings (copyrights, ECC)
- Result of the “clearing”
  - SPDX reporting
  - Generated notice or readme file
  - debian-copyright

Upload Component
Agents Scanning
Review Results
Generate Reporting
Pass Report to Client
# Hands-On: Basic End-to-End Workflow

## Functionality

1. **Using FOSSology End-to-End**
   - From uploading …
   - … to generating report: SPDX

2. **Uploading** - offers a variety of selections

3. **Review the uploaded file in the license browser**

4. **Review the found licenses in the aggregated view**

5. **Do the clearing work**

6. **Review the copyrights**

7. **Review the Export Control and Customs (ECC)**

8. **Generate desired report output**

## Example

- **Upload <OSS Component>**
  - Go to license browser by clicking upload name or selecting clearing from the action menu
  - Select “go to all …. with licenses”
  - Review licenses and apply decisions

- Select the copyright, e-mail, url section from the yellow menu bar area
  - Review copyright statements and correct in case

- Select ECC from the yellow area and review

- Go to Browse main view
  - Select the pop-up menu of the ionicons upload
# Features: Two License Scanners: Nomos and Monk

<table>
<thead>
<tr>
<th><strong>Nomos Keywords</strong></th>
<th><strong>Nomos Reg. Expressions</strong></th>
<th><strong>Bulk Phrase Matches</strong></th>
<th><strong>Monk Full text Matches</strong></th>
</tr>
</thead>
</table>
| • Finds all kind of license relevant texts  
• Finds unknown Licenses | • Finds most license relevant texts  
• Identifies also derivatives of licenses | • Good for finding actual licenses  
• Identifies also derivatives of licenses | • Certainty that known license text is actually found and wording is exactly reproduced |

**Flexibility**

<table>
<thead>
<tr>
<th><strong>Nomos Keywords</strong></th>
<th><strong>Nomos Reg. Expressions</strong></th>
<th><strong>Bulk Phrase Matches</strong></th>
<th><strong>Monk Full text Matches</strong></th>
</tr>
</thead>
</table>
| • Very imprecise  
• Does not identify license  
• High number of false positives | • Only limited precision for identifying actual licenses  
• False positives | • Limited to known phrases only  
• Does not provide certainty about original or derivative | • Works only on known license texts  
• Actual occurrences are minority |

**Precision**
# FOSSology Feature Overview

A **Web server application for license and copyright compliance of software components.**

<table>
<thead>
<tr>
<th>License Scan features</th>
<th>Other features</th>
</tr>
</thead>
<tbody>
<tr>
<td>· Regular expression scanner</td>
<td>▪ Copyright, authorship statements scanner</td>
</tr>
<tr>
<td>· Text similarity scanner</td>
<td>▪ Export control and customs scanner</td>
</tr>
<tr>
<td>· License (text) management</td>
<td>▪ Command line interfaces</td>
</tr>
<tr>
<td>· Aggregation of licenses in hierarchical view</td>
<td>▪ Reporting</td>
</tr>
<tr>
<td>· License histogram</td>
<td>▪ SPDX RDF and tag-value</td>
</tr>
<tr>
<td>· Supporting concluded vs. found license</td>
<td>▪ Debian-copyright</td>
</tr>
<tr>
<td>· Bulk processing of files with same licensing</td>
<td>▪ Plain text output</td>
</tr>
<tr>
<td>· Reusing of license conclusions</td>
<td>▪ Files sorting in buckets</td>
</tr>
<tr>
<td></td>
<td>▪ User, group and upload management</td>
</tr>
</tbody>
</table>
FOSSology is Open Source

What happens actually?

- As an organization distributing software there is responsibility for license compliance.
- This work must be actually provided by 3rd parties as well!

- With FOSSology, a tool is freely available supporting all kind of organizations with their license compliance efforts.
  - GPL-2.0 licensed
You like it, and would like to contribute?

- See the [FOSSology.org](http://fossology.org) website for the most recent list of public supporters.
- *Not on this list?* – Consider supporting the project with putting your logo there. To be included, send email to fossology-steering@fossology.org.
- Report issues on Github: [https://github.com/fossology/fossology](https://github.com/fossology/fossology)
- And of course… code contributions to extend and improve FOSSology are welcome!
Automation Summary

Where you can automate
FOSSology – What Do we Have

OSS Component Analysis

- Licensing
- Obligations
- Acknowledgements
- Copyright Statements
- Export Control
- More Analysis
FOSSology – What Do we Have

OSS Component Analysis

- Licensing
- Obligations
- Acknowledgements
- Copyright Statements
- Export Control
- More Analysis

- License statements either from component or metadata
- Instructions related to individual licenses,
- Requirements to acknowledge use, related to component
- Copyright statements related to component
- ECC relevant statement
- Org Specific: e.g. Non-commercial use
FOSSology – What Do we Have

OSS Component Analysis

- Licensing
- Obligations
- Acknowledgements
- Copyright Statements
- Export Control
- More Analysis

License statements either from component or metadata
Instructions related to individual licenses,
Requirements to acknowledge use, related to component
Copyright statements related to component
ECC relevant statement
Org Specific: e.g. Non-commercial use
Automation possible for processed cases
Automation possible for processed cases
Automation possible for processed cases
Automation possible for processed cases
Automation possible for processed cases
Corrections required
Corrections required
SW360 Introduction

Eclipse SW360 – An Open Source Component Hub
Introduction

SW360 is a 3\textsuperscript{rd} party software component catalogue
Assigns 3\textsuperscript{rd} party components to products or projects

Goals and Benefits

- Reuse information about components
- Coordinate product documentation process
- Support software clearing
SW360 and Liferay Portal Community Edition

**SW360 is an application based on Liferay Portal Community Edition**

- SW360 uses Liferay Portal CE as portal server
- Liferay Portal community edition, LGPL-2.1-or later
- Currently at 6.X, upgrade to 7.X planned

**Advantages**

- Many Liferay-based solutions exist
- Proven user, session and data management
- Further technologies ready for future adoption:
  - Mobile device support
  - Marketplace of add-ons
  - Collaboration and Social tools
Main Use Case 1: Component Inventory Database

Collect Information about Components

- **It is about Components in use**: for all others, Internet can do better

- **OSS Licensing**: collect analysed licensing information (and reuse analyses)

- **Not OSS only**: internal components, commercial, freeware

- **More information**: ECC, vulnerabilities, statistics, static code analyses, etc.
Main Use Case 2: Bill-of-Material Management

**BOM. Inventory Management**

- Understand which software component is used in which products
- **Product / Project**: holds relation to releases of components
- **Component Catalogue**: captures organisation information of components and releases
Main Use Case 3: Product Documentation

**Readme OSS / NOTICE Generation**

- **Create Component Releases**
- **Upload SPDX file**: exchange file for Licenses, Copyrights and Acknowledgements
- **Create Project**: and add the component releases
- **Generate documentation**: For all linked releases, license information is collected for generation
Basic Data Model

**Goals and Motivation**

- Clean Component Catalogue: Reduction of duplicate entries.
- Separating vendor and version from the name of the components brings clarity to component identification.
- Interoperation with other systems: support the CPE standard which also implement this 3-parts separation.
- Having the clear modeling of data enables better search and filtering abilities.
SW360 Hands On

A Brief Example of the SW360 Structures
How it works 1 - Create a Component Entry

Create Component Entry

Add Component Entry

Add Release Entry

Add Package (Upload) to Release

- A component is a container for releases
- Just creating a component alone creates an empty container entry with no releases in it.
- Projects cannot reference a component, but a release

- Release = Version
- A component can have multiple releases
- It makes sense to group them
- Now a vendor can be assigned

- Multiple open source packages of the same releases: should be multiple releases actually
- There should be just one source code upload per release
# Hands-On: Create components and releases

<table>
<thead>
<tr>
<th>Functionality</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. <strong>Creating first component</strong></td>
</tr>
<tr>
<td>• In the basic list view, click on add component</td>
</tr>
<tr>
<td>• Perform the mandatory entries</td>
</tr>
<tr>
<td>2. <strong>Create a release entry</strong></td>
</tr>
<tr>
<td>• Release corresponds to a version number</td>
</tr>
<tr>
<td>• Review fields</td>
</tr>
<tr>
<td>• Upload source file with source file attachment type</td>
</tr>
<tr>
<td>3. <strong>Repeat for second and third component and release</strong></td>
</tr>
<tr>
<td>• Repeating the same steps</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Create entry as new component</td>
</tr>
<tr>
<td>• Click “Add” in component list</td>
</tr>
<tr>
<td>• Enter required information, for example “library” as type</td>
</tr>
<tr>
<td>• Add a release, version X.X</td>
</tr>
<tr>
<td>• Update the release</td>
</tr>
<tr>
<td>• Go to attachments</td>
</tr>
<tr>
<td>• Upload source file</td>
</tr>
<tr>
<td>• Create another component / release</td>
</tr>
<tr>
<td>• Apply same steps</td>
</tr>
</tbody>
</table>
How it works 2 - Create a Project Entry

- A project in sw360 can be also called “product”. It is a structure to maintain a Bill-of-Material of software components
- In fact, can represent product, service, delivery

- Release = Version
- Projects can contain other projects
- Allows for building blocks

- Send to clearing if no clearing report exists
- Integration with another tool, Fossology
## Hands-On: Create projects

<table>
<thead>
<tr>
<th>Functionality</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1. Create a project entry</strong></td>
<td>- Click “Add project” on main view</td>
</tr>
<tr>
<td>- Review the fields</td>
<td>- Button is on the top right</td>
</tr>
<tr>
<td>- Link two releases to project</td>
<td>- Enter a simple name, for example “project-main”</td>
</tr>
<tr>
<td><strong>2. Create second project entry</strong></td>
<td>- Click on “Add projects”</td>
</tr>
<tr>
<td>- Link the other release to it</td>
<td>- (this creates the data structure)</td>
</tr>
<tr>
<td><strong>3. Link a sub project</strong></td>
<td>- Add the two releases</td>
</tr>
<tr>
<td>- For the first project, link the</td>
<td>- They were just created before</td>
</tr>
<tr>
<td>second project</td>
<td>- …</td>
</tr>
<tr>
<td>- Review the project hierarchy in</td>
<td>- Go on for the second project</td>
</tr>
<tr>
<td>linked projects</td>
<td>- Choose a simple name, auch as “project-sub”</td>
</tr>
<tr>
<td></td>
<td>- Click on save project</td>
</tr>
<tr>
<td></td>
<td>- Add an remaining release…</td>
</tr>
</tbody>
</table>
### Hands-On: Create README_OSS

<table>
<thead>
<tr>
<th>Functionality</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. <strong>Go to project main view</strong></td>
<td>• Go to the main project that was created first</td>
</tr>
<tr>
<td>• Select the project from the list</td>
<td>• Go linked releases and projects</td>
</tr>
<tr>
<td>• View the details</td>
<td>• Click Generate License info</td>
</tr>
<tr>
<td>2. <strong>Create the README_OSS</strong></td>
<td>• Review the license information from the SPDX files</td>
</tr>
<tr>
<td>• Go to clearing information</td>
<td>• Download the README_OSS to review files in desktop</td>
</tr>
<tr>
<td>• Hit the generation button</td>
<td></td>
</tr>
<tr>
<td>3. <strong>Review the generated README_OSS</strong></td>
<td></td>
</tr>
<tr>
<td>• (remove inappropriate licenses)</td>
<td></td>
</tr>
</tbody>
</table>

**Example**

- Go to the main project that was created first
- Go linked releases and projects
- Click Generate License info
- Review the license information from the SPDX files
- Download the README_OSS to review files in desktop
SW360Antenna Example

Brief example to facilitate automation with SW360
Example Automation Setup

Identify dependencies

- Software-Bill-of-Material, S-BOM

Identify involved license compliance information

- Licenses, license relevant statements, acknowledgements

Optional steps

- ECC, own policies, aggregation on product view

Generate compliance „bundle“

- Licenses, copyrights, acknowledgements, source code, written offer, further reporting, etc.
Example Automation Setup

- Identify dependencies
  - Software-Bill-of-Material, S-BOM

- Identify involved license compliance information
  - Licenses, license relevant statements, acknowledgements

- Optional steps
  - ECC, own policies, aggregation on product view

- Generate compliance „bundle“
  - Licenses, copyrights, acknowledgements, source code, written offer, further reporting, etc.

Tools Used:
- FOSSology
- SW360, SW360antenna
Automation with SW360, Antenna, FOSSology

- **Identify dependencies**
  - Software-Bill-of-Material, S-BOM

- **Identify involved license compliance information**
  - Licenses, license relevant statements, acknowledgements

- **Optional steps**
  - ECC, own policies, aggregation on product view

- **Generate compliance „bundle“**
  - Licenses, copyrights, acknowledgements, source code, written offer, further reporting, etc.

SW360antenna (also: O.R.T., Tern, etc.)

FOSSology

SW360, SW360antenna

SW360, SW360antenna
Example for Integration into DevOps: SW360antenna

**https://github.com/eclipse/antenna**

- **Maven**: Plugin for maven projects
- **Gradle**: Plugin for gradle projects
- **Command Line**: CLI tools to run the same, not embedded into a maven or gradle run (but project files are read)
Demo

Showing this on screen

- **antenna/example-projects/mvn-test-project:** Example integration with a maven-based project

- **Review Configuration:**
  Run as maven target, setup in pom.xml, configuration, workflow

- **Run Targets:**
  See component, release and project records created in SW360 web application
Thank you for your attention!