QUARTERMASTER
OPEN SOURCE COMPLIANCE TOOLING

KDE AKADEMY, AUGUST 2018

@fosscompliance (Quartermaster)
@mirkoboehm (me)
What is Quartermaster?

- Quartermaster is an integrated FOSS toolchain that implements industry best practises of license compliance management.

- Quartermaster runs adjacent to a software build in CI or development environments. It collects build graphs, performs analysis and generates compliance reports (to developers, reviewers, upstream).

- Quartermaster focuses on fact finding and accurate, complete and up-to-date compliance documentation.

- Quartermaster is FOSS and developed under a collaborative model.
Why?
There is still **no industry standard** for FOSS compliance tooling. The management of software copyright and license compliance in FOSS **needs to improve**.

Consensus
“Hygiene factors … do not give positive satisfaction or lead to higher motivation, though dissatisfaction results from their absence.”

–Two-factor theory (Wikipedia)
FOSS Compliance is a **hygiene factor**. Uncertainty and litigation **undermines** the fabric of Open Source.
For whom?

- FOSS Communities: Deliver compliance documentation with your packages.
- Software vendors: Certify own compliance checks along the supply chain (see OpenChain spec).
- Distribution channels: Verify compliance documentation for products in your store/on your distribution/…
Who makes it?

- Quartermaster is an Open Source project by licensing and governance.
- Endocode is currently driving it.
- Siemens, Google support it.
- Quartermaster should become an independent project under a neutral umbrella (LF?)
Workflow (Phases and Tasks)

Workflow Phases:
- Construction
- Analysis
- Reporting

Functions:
- Configuration
- Metadata
- Build Graph
- History
- Contributors
- Copyright
- Licenses
- Policies
- Metrics
- Knowledge
- BOM
- CI Feedback
- Alerts
- License Catalog

Contributors
Metadata
Configuration
Copyright
Licenses
Policies
Metrics
BOM
CI Feedback
Alerts
License Catalog
Knowledge
Workflow (Sample Modules)

**Workflow Phases**
- Construction
- Analysis
- Reporting

**Functions**
- cmake
- Config file
- make
- Git
- ScanCode
- Cregit
- (Commercial)
- Custom policy check
- (Code complexity?)
- ClearlyDefined
- HTML
- SPDX
- Slack
- Github Badge
- ClearlyDefined
Step by step…
(Demo Time…)

THE LINUX KERNEL
Architecture

- Master process

- Toolchain specific build system instrumentation (gcc, clang, go build, …)

- gRPC/protobuf module APIs

- No file formats

- Modular command line toolchain

- Integration API in master

- Linux/OSX/(Windows) client side, master runs in container
License Model

- Data Model: Open Data License
- Core Toolchain: GPL3
- Modules: separate processes, communicating with the master
- Paradigm: Toolchain is FOSS. Core QMSTR modules are FOSS. Proprietary integrations possible, all relevant data becomes part of Open Data model.
Take-aways: *Lessons learned* from the Quartermaster prototype
Facts vs Opinions

• Compliance Documentation:
  • Authors, copyright, license information is project metadata and belongs into the “package” (repository and commit history).

• Approval, Guidance, Supply Chain:
  • Approvals, reviews, judgement calls are business-specific and belong into a knowledge base.
Inbound vs Outbound Licenses

- Source package SPDX files document *inbound licenses*.
- Outbound license *cannot be deduced*.
- If outbound license is specified by vendor, license compatibility can be *algorithmically evaluated*.
Upstream vs Data Pools

- FOSS compliance **data** belongs upstream.

- Default: The inbound licenses of a module are deduced from the content of the repository.

- **Opinions** (reviews, approvals, …) are not generic.

- In-house “Open Source Handbook”.

- Relevant metadata not available upstream should be curated and centralised.

- **ClearlyDefined**.
Build time is the right time.
Build Time vs Static Code Analysis

- A **Concrete Build Dependency Graph** associates referenced source files and dependencies to a (binary) target.

- Source code analysis (code scanning) detects attributes of source files (licenses, authors, copyright holders).

- The **combination of build time and static code analysis** allows reasoning about outbound licenses.
Quality Issues with Unmanaged Code Repositories

- Environments that assemble programs clients-side from unknown sources defeat quality assurance mechanisms.

- FOSS Compliance documentation is possible, but unreliable and costly until this quality problem is resolved.
Improving FOSS Compliance is a process.

We need to improve all aspects over time: Supply chain management, up to date and accurate documentation, reliable knowledge bases, …
Community and Business
Open Governance

- Public Website: qmstr.org
- Public sprint and milestone planning (see blog).
- Regular development updates
- Collaborative requirements development
- Show me the code: github.com/QMSTR
- Open Slack channel: qmstr.slack.com
- Follow @fosscompliance :-)
- Legal Advisory Committee (collaboration with REUSE? FSFE Legal Network?)
QMSTR is commercially supported FOSS

- Separation of product and services.
- Endocode will begin offering professional services with the release of QMSTR v0.1.
  - Support Contracts
  - Training
  - Custom Development
  - Consulting
- No Open Core: 100% FOSS.
Summary
QMSTR creates an integrated Open Source toolchain that implements industry best practices of license compliance management.
Project Roadmap

• Q2/2017: Proof of Concept. (✅ check)

• Q4/2017: Minimum viable prototype. (✅ check)

• Jan 17, 2018: QMSTR 0.1 requirements workshop (✅ check)

• April 2018 (LLW 2018): QMSTR v0.1 release. (✅ check)

• July 2018: QMSTR v0.2 release. (✅ check)
  
  • New features: Git analyser, SPDX parser, Python QMSTR modules, …

• Ongoing: A major release every three months.
What could the KDE Community do?

• Contribute to language and KDE-style toolkits and workflow support.
• Adopt Quartermaster for releases, packaging, checks, …
• …?
Next opportunities to get involved!

- Next sprint community hangout: August 22
- Q4 milestone planning workshop October 2018 (possibly co-located with Open Source Summit Europe)
- We need: coding. feedback. knowledge. adoption. funding.
QUESTIONS?

QUARTERMASTER
OPEN SOURCE COMPLIANCE TOOLING

KDE AKADEMY, AUGUST 2018

@fosscompliance (Quartermaster)
@mirkoboehm (me)
Credits

- Kristian Fagerström, “Earth”, https://www.flickr.com/photos/147764143@N07/32995070824, CC BY-SA 2.0
- Clay Gilliland, “Inbound”, https://www.flickr.com/photos/26781577@N07/12104785406, CC BY-SA 2.0
- Darwin Bell, “red lock”, https://www.flickr.com/photos/darwinbell/275662601, CC BY 2.0
- Alex Ermolin, “We’re Open”, https://www.flickr.com/photos/alexermolin/4974314835, CC BY 2.0