

ELISA

Enabling Linux in Safety Applications

Open Source Best Practices as a Foundation for Quality Management

14th exida Symposium, Spitzingsee, Oct 27-28 2026

Philipp Ahmann, ETAS GmbH

Work in Progress - License: CC-BY-4.0

whoami - Philipp Ahmann



Sr. OSS Community Manager
Automotive OSS Process Lead

Chair of the Technical Steering Committee
Lead of the Systems Working Group

Advisory Board Member of LF Europe

Eclipse Safe Open Vehicle Core Committer

OSS enthusiast and promoter

Work in Progress - License: CC-BY-4.0

Agenda

- Motivation & Teaser

- Literature examples and further reading
- Quality Management & ISO 9001
- Best practices programs/frameworks

- Project examples from the wild
- The special role of Eclipse S-CORE like projects

- Summary

Work in Progress - License: CC-BY-4.0

Motivation & Teaser

Work in Progress - License: CC-BY-4.0

Based on ELISA Lighthouse-OSS

Setting the scene

- Traditional software development methods were **built for proprietary software**
- No **existing quality standard** matches established open source development practices
- But OSS Communities have demonstrated **mature processes and consistent quality**
- OSS is a viable choice for regulated industries, **but there is a lack of a matching standard**



Photo by William Bout on Unsplash

ELISA Enabling Linux in Safety Applications

Work in Progress - License: CC-BY-4.0

Real world statements from OSS contributors

“I cannot upstream this software, it is not good enough.”

(Engineer wihin an automotive company asked to upstream some of their work)

Being publicly exposed, may put pressure on you.

Work in Progress - License: CC-BY-4.0

Real world statements from OSS contributors

**”Technically, my patch remained the same,
but I had to adopt a lot on documentation and style.”**

*(Engineer within an automotive company after upstreaming some of their work.
Work was delivered internally following ASPICE processes)*

Documentation and coding style are practice and important to keep maintainability

Work in Progress - License: CC-BY-4.0

Literature examples & further reading

Work in Progress - License: CC-BY-4.0

Look into existing evidences in literature

Examples of literature which focuses on Open Source practices:

[1] 2017 - Evaluation of open-source OS for Safety-Critical Applications (P. Berntsson)

[2] 2020 OSS Development Process Systematic Review (B. Napoleao, F. Petrillo, S. Hallé)

[3] 2025 - Towards a more sustainable and secure software tooling in free/libre open source software environments (S. Tatschner)

Work in Progress - License: CC-BY-4.0

Based on ELISA Lighthouse-OSS

Categories and practices described in literature [2]

Category	Example Practices in OSS	What it means
Governance	• Meritocratic leadership • Transparent decision-making • Clearly defined roles	Governance provides predictability and accountability. It helps external users, contributors, and assessors understand who can make decisions, how decisions are recorded, and how project direction is established.
Development Practices	• Self-assigned tasks • Code ownership • Version control	Development Practices provide the technical foundation for reliable OSS development. They make it possible for distributed contributors to work together, maintain the codebase, review changes, and preserve traceability.
Quality Engineering (Maturity model)	• Clean commit history • Coding conventions • Structured peer review & testing	Quality Engineering provides evidence that the project does not merely define good practices, but applies them consistently. It supports defect prevention, maintainability, regression control, and release confidence.
Documentation	• User-facing documentation • Developer documentation	Documentation describes how the project records and communicates information needed to use, develop, maintain, integrate, and assess the software. It enables adoption, contribution, maintenance, and assessment. It reduces reliance on tacit knowledge and helps external users understand whether the project is suitable for their context.
Sustainability (Risk management)	• Secure sustainable funding models • Monitor and improve Bus Factor	Sustainability describes whether the project can continue to operate, maintain itself, respond to issues, and support users over time. A project may be technically excellent but still risky for downstream adoption if it depends on too few people, lacks funding, or cannot sustain maintenance.

Work in Progress - License: CC-BY-4.0

What a Quality Management System expects

Objectives:

- Leadership
  (Customer Focus, Quality policy, Roles, …)
- Planning
  (Risks, opportunities, quality, changes, …)
- Support
  (Resources, Competence, Awareness, Communication, …)

Execution:

- Operation (execution of above)
- Performance evaluation
  (internal monitoring, customer satisfaction)
- Improvement

- Many parts are covered in the practices mentioned on the previous slides and has evidence in literature [1][2][3]
- As the QM systems are based on company structures not everything is relevant/applicable to Open Source
- Some elements may be good to be added to OSS projects *(but OSS projects do not have access to these standards)*

Work in Progress - License: CC-BY-4.0

Quality Management & ISO 9001

Work in Progress - License: CC-BY-4.0

What Quality Management Actually Needs

- Defined and controlled processes
- Responsibilities and authority
- Competence
- Evidence of execution
- Monitoring and evaluation
- Continual improvement

A quality-management system does not require every organization to use the same toolchain or development lifecycle.

Why is ISO 9001 important?

With more than one million certificates issued to organizations in 189 countries, ISO 9001 is the most widely used quality management standard in the world. Within the ISO 9000 family, which defines seven quality management principles including a strong customer focus and continual improvement, ISO 9001 is the only standard that can be certified to (though certification is not mandatory).

Business benefits include:

- **Customer confidence**: The standard ensures that organizations have robust quality control processes in place, leading to increased customer trust and satisfaction.
- **Effective complaint resolution**: ISO 9001 offers guidelines for resolving customer complaints efficiently, contributing to timely and satisfactory problem-solving.
- **Process improvement**: The standard helps identify and eliminate inefficiencies, reduce waste, streamline operations, and promote informed decision-making, resulting in cost savings and better outcomes.
- **Ongoing optimization**: Regular audits and reviews encouraged by ISO 9001 enable organizations to continually refine their quality management systems, stay competitive, and achieve long-term success.

https://www.iso.org/standard/9001

Work in Progress - License: CC-BY-4.0

Example: Change Control

OSS:	**Typical quality questions:**
• Issue • Proposed modification	• Why was the change needed?
• Automated gates • Review • Approval	• Who reviewed it? • What verification was performed? • Which criteria were applied?
• Merge • Release note	• Which released artifact contains it?

Repositories can become part of the quality record, if workflows are defined and consistently applied.

ELISA
Enabling Linux in Safety Applications

Work in Progress - License: CC-BY-4.0

Example: Competence and Responsibility

- Maintainer and reviewer roles
- Code ownership
- Contribution history
- Approval permissions
- Documented governance
- Nomination processes

- Rights are given in repositories.
- The maintainer owns the code.
- Every commit is traceable.
- Without an approval nothing gets merged.
- Contribution guidelines are put in repos.
- New maintainers are added by merit.

Good to know:

“Contribution volume is not the same as competence.”

Getting code into an OSS project may be much harder than getting something into a commercial proprietary product behind closed walls.

Work in Progress - License: CC-BY-4.0

Best practices programs/frameworks

Work in Progress - License: CC-BY-4.0

Based on ELISA Lighthouse-OSS

Best practices programs/frameworks

- ELISA Lighthouse OSS Checklist
- **Apache Project Maturity Model**
- **OpenSSF: Scorecard**,
  Best Practices Badge, OSPS Baseline
- **LFX Insights**
- CNCF TAG Security Self-Assessment
- Eclipse Trustable Software Framework
- OpenChain ISO/IEC 5230
- TODO Group Guides
- CHAOSS (community health)
- LF - AI Responsible AI Framework

Other (non-OSS) quality guidelines:

- NIST Software Development Framework
- INCOSE Systems Engineering Body of Knowledge" (SEBoK)
- ISO 42003 - Implementation Guide for AI Management
- AMMITEC - Maritime Software Quality Guidelines

Work in Progress - License: CC-BY-4.0

Apache Project Maturity Model

Work in Progress - License: CC-BY-4.0

What it is:

- The Apache Project Maturity Model provides a suggested framework for evaluating the overall maturity of an Apache project community and the codebase that it maintains. While this model is specific to projects of the Apache Software Foundation (ASF), many of these factors apply to any open source project.

handout slide

https://community.apache.org/apache-way/apache-project-maturity-model.html#overview

Work in Progress - License: CC-BY-4.0

What you could potentially also find in standards:

- The project is independent from any corporate or organizational influence.
- The project maintains a public list of its contributors who have decision power. The project's PMC (Project Management Committee) consists of those contributors.
- The project's PMC (Project Management Committee, see CS10) approves each software release in order to make the release an act of the Foundation.
- Decisions require a consensus among PMC members and are documented on the project's main communications channel. The PMC takes community opinions into account, but the PMC has the final word.
- The project strives to answer user questions in a timely manner.

Work in Progress - License: CC-BY-4.0

What you would typically NOT find in standards:

- The community strives to be meritocratic and gives more rights and responsibilities to contributors who, over time, add value to the project.
- The project documents how contributors can earn more rights such as commit access or decision power, and applies these principles consistently.
- “Dictators, benevolent or not, are not welcome in Apache projects.”

Work in Progress - License: CC-BY-4.0

handout slide

Specifics of the model

- Compared to automated tools like in OpenSSF Scorecard, the rating is done manually
- This allows the judgement of soft elements, but also requires higher effort
- Results are a self assessment.



Photo by Randy Jacob on Unsplash

https://opendal.apache.org/community/maturity/

Work in Progress - License: CC-BY-4.0

OpenSSF Scorecards



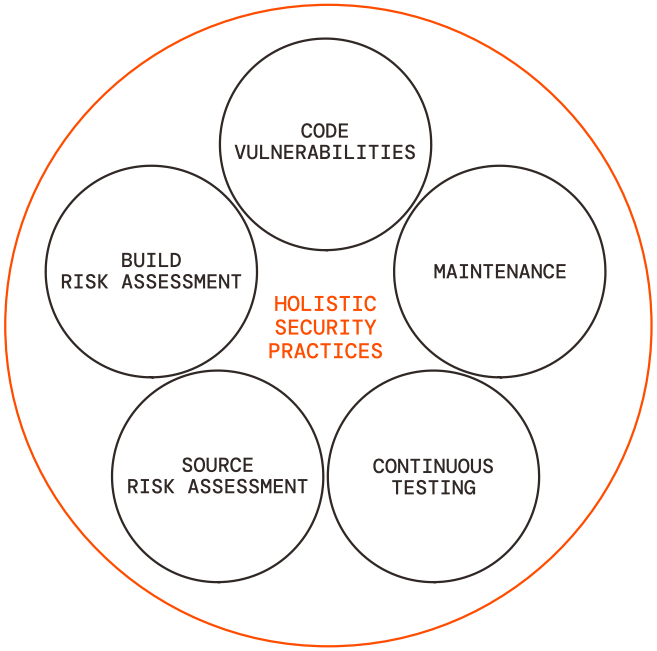
Work in Progress - License: CC-BY-4.0

What it is:

- Scorecard assesses open source projects for security risks through a series of automated checks.
- It was created by OSS developers to help improve the health of critical projects that the community depends on.
- You can use it to proactively assess and make informed decisions about accepting security risks within your codebase.
- You can also use the tool to evaluate other projects and dependencies, and work with maintainers to improve codebases you might want to integrate.
- Scorecard helps you enforce best practices that can guard against:

Work in Progress - License: CC-BY-4.0

Practices which are checked:



https://scorecard.dev

- Does the project have unfixed vulnerabilities?
- Does the project use tools to help update its dependencies e.g. Dependabot
- Does the project contain a security policy?
- Does the project run tests in CI, e.g. GitHub Actions, Prow?
- Does the project use fuzzing tools, e.g. OSS-Fuzz?
- Does the project use Branch Protection?
- Does the project require code review before code is merged?

Work in Progress - License: CC-BY-4.0

Specifics of the framework

- Automated tools which can be executed locally or be added as GitHub actions
- Take about 10 minutes to get started from the command line.
- Focus on security, less on overall quality.
- Not taking soft elements into account. *(in contrast to Apache)*

Holistic security practises

Code vulnerabilities	Description	Risk
Vulnerabilities	Does the project have unfixed vulnerabilities? Uses the OSV service.	High

Maintenance	Description	Risk
Dependency Update Tool	Does the project use tools to help update its dependencies e.g. Dependabot, RenovateBot?	High
Maintained	Is the project maintained?	High
Security Policy	Does the project contain a security policy?	Medium
Licence	Does the project declare a licence?	Low
CII Best Practices	Does the project have a CII Best Practices Badge?	Low

Continuous testing	Description	Risk
CI Tests	Does the project run tests in CI, e.g. GitHub Actions, Prow?	Low
Fuzzing	Does the project use fuzzing tools, e.g. OSS-Fuzz?	Medium
SAST	Does the project use static code analysis tools, e.g. CodeQL, LGTM, SonarCloud?	Medium

Source risk assessment

Name	Description	Risk
Binary Artifacts	Is the project free of checked-in binaries?	High
Branch Protection	Does the project use Branch Protection?	High
Dangerous Workflow	Does the project avoid dangerous coding patterns in GitHub Actions?	Critical
Code Review	Does the project require code review before code is merged?	High
Contributors	Does the project have contributors from multiple organizations?	Low

Build risk assessment

Name	Description	Risk
Pinned Dependencies	Does the project declare and pin dependencies?	Medium
Token Permissions	Does the project declare GitHub workflow tokens as read only?	High
Packaging	Does the project build and publish official packages from CI/CD, e.g. GitHub Publishing?	Medium
Signed Releases	Does the project cryptographically sign releases?	High

ELISA
Enabling Linux in Safety Applications

Work in Progress - License: CC-BY-4.0

LFX Insights



Work in Progress - License: CC-BY-4.0

handout slide

What it is:

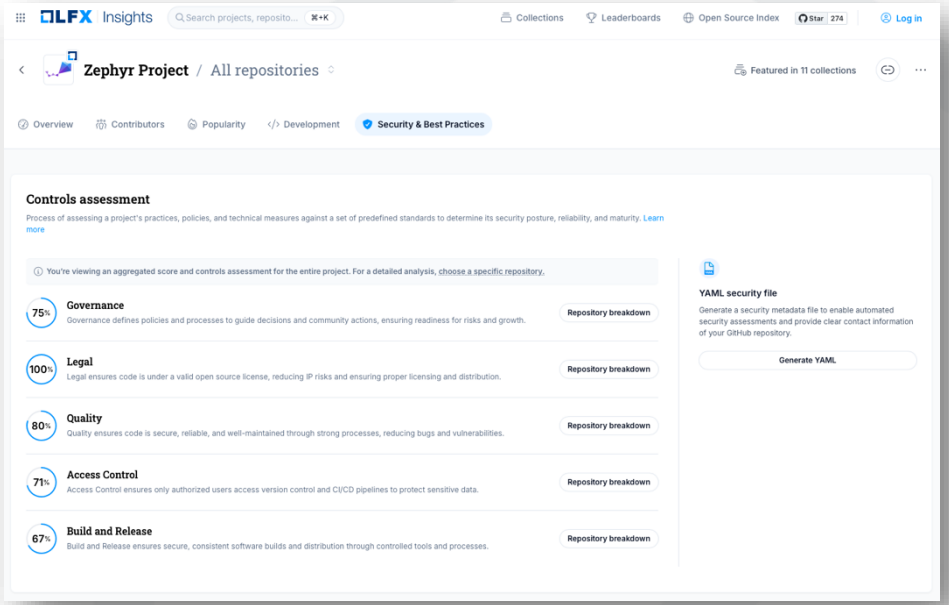
- LFX Insights helps developers and their organizations make smarter decisions about the open source projects they depend on.
- Assess a project's overall health, with a summary of key activity, community, and security signals.
- See contributor affiliations and organization-level insights
- See whether a project follows security best practices, including vulnerability management, documentation quality, and governance signals.

https://insights.linuxfoundation.org/docs/introduction/what-is-insights/

Work in Progress - License: CC-BY-4.0

LFX Insights – based on Zephyr project

- **Governance**
  defines policies and processes to guide decisions and community actions, ensuring readiness for risks and growth.
- **Legal**
  ensures code is under a valid open source license, reducing IP risks and ensuring proper licensing and distribution.
- **Quality**
  ensures code is secure, reliable, and well-maintained through strong processes, reducing bugs and vulnerabilities.
- **Access Control**
  ensures only authorized users access version control and CI/CD pipelines to protect sensitive data.
- **Build and Release**
  ensures secure, consistent software builds and distribution through controlled tools and processes.

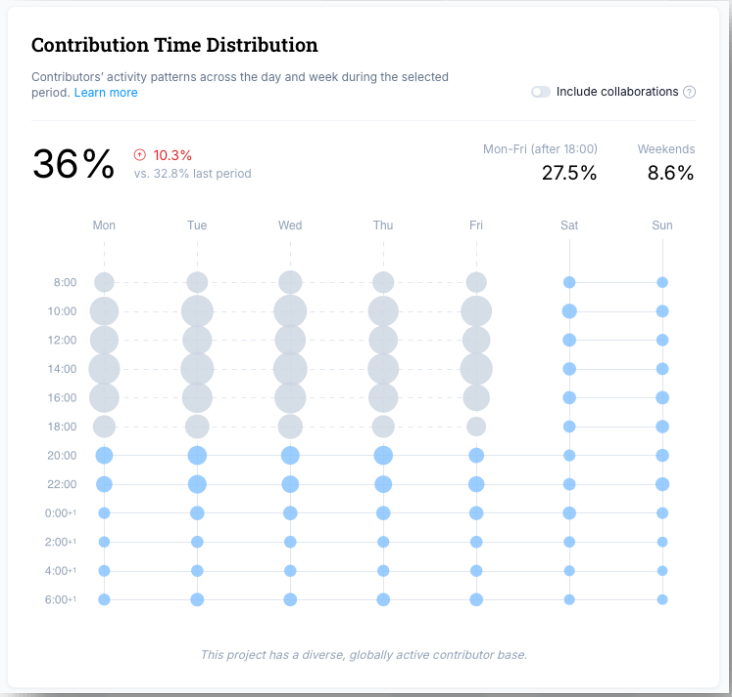


https://insights.linuxfoundation.org/project/zep

Work in Progress - License: CC-BY-4.0

Example: Heat map of contribution times

- Contribution time and number can be an argument that the open source project is or is not a hobbyist activity.
- Example:
  - Most Zephyr contributions happen Monday to Friday between 8:00 and 20:00.
- Note:
  - The data has weakness as it depends on profile settings in GitHub and just provides an indicator



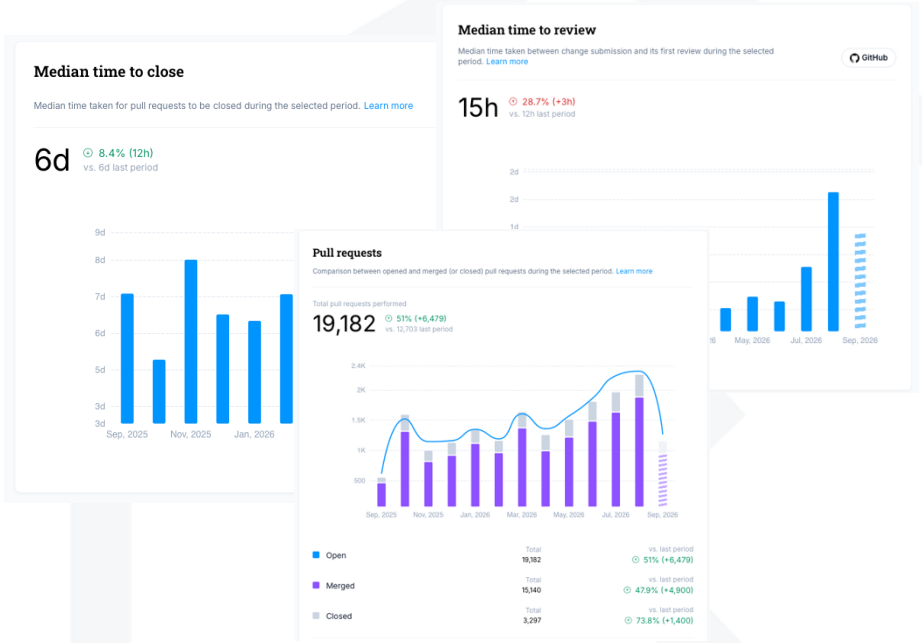
https://insights.linuxfoundation.org/project/zep/development?timeRange=past365days&start=2025-09-17&end=2026-09-17&widget=contributions-outside-work-hours

ELISA
Enabling Linux in Safety Applications

Work in Progress - License: CC-BY-4.0

Not every number let you directly derive something.

- The time it takes to close a PR is not telling you something about quality.
- Compare:
  - 100% Code coverage also does not tell you something about the code quality.
- Note:
  - Set things into context and do explicit checks.
  - As everything is public you are able to perform checks
- Frameworks are hints to best practices, but measurements can be wrong.



https://insights.linuxfoundation.org/project/zep/development?timeRange=past365days&start=2025-09-17&end=2026-09-17&widget=median-time-to-close

Work in Progress - License: CC-BY-4.0

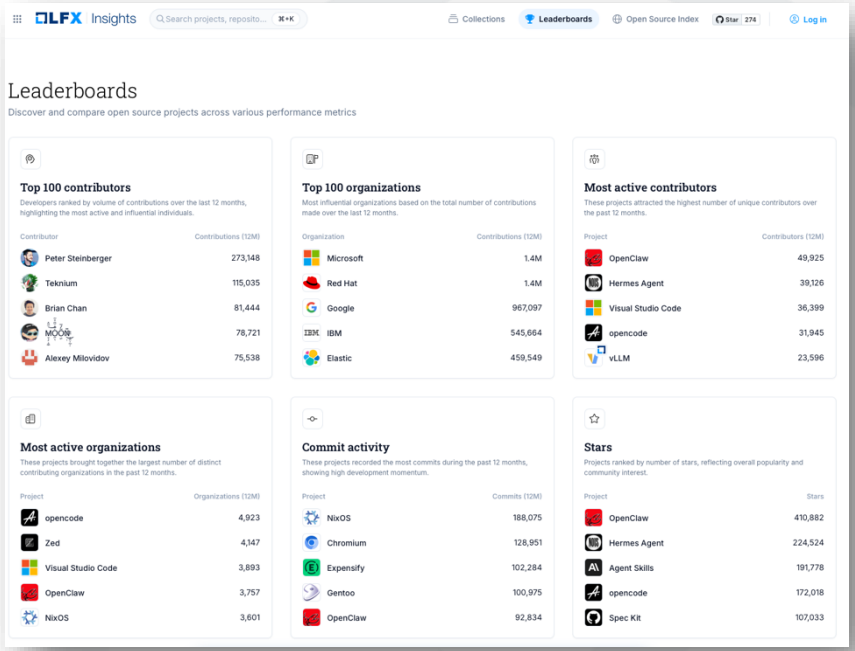
handout slide

Specifics of the tooling

- Wide knowledge base and many projects listed.
- Projects beyond Linux Foundation covered.
- Documentation and conference talks exist around the tooling
- Also pulls in OpenSSF Scorecards

Good to know:

- Provide collections, statistics and more on companies, contributors, not just best practices on individual projects



https://insights.linuxfoundation.org/leaderboards

ELISA
Enabling Linux in Safety Applications

Work in Progress - License: CC-BY-4.0

Project examples from the wild

Work in Progress - License: CC-BY-4.0

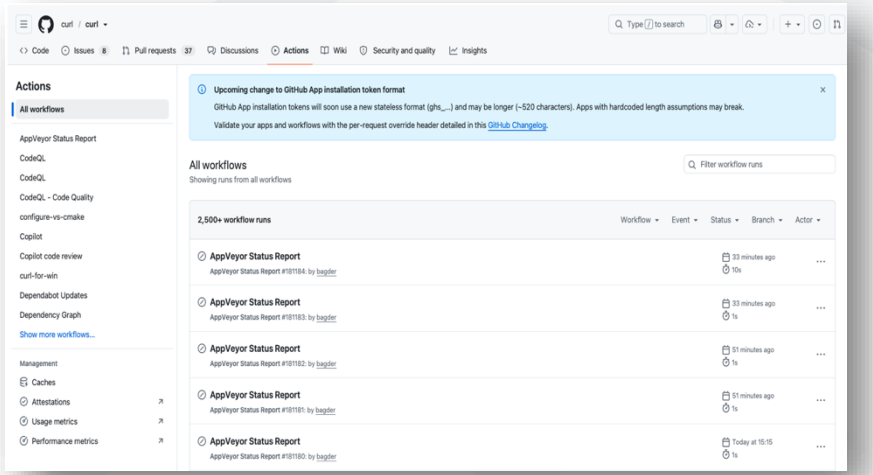
What OSS projects may already provide to you...

- peer review,
- automated testing/verification,
- continuous integration,
- traceable and transparent contribution processes,
- and collaborative governance

Work in Progress - License: CC-BY-4.0

Automation: Pipelines, actions, testing…

- Projects need to be efficient
  (as maintainer resources are limited)
- Scripting (&AI) help to automate where possible
- Automated testing and static analysis are performed (e.g. via CodeQL)

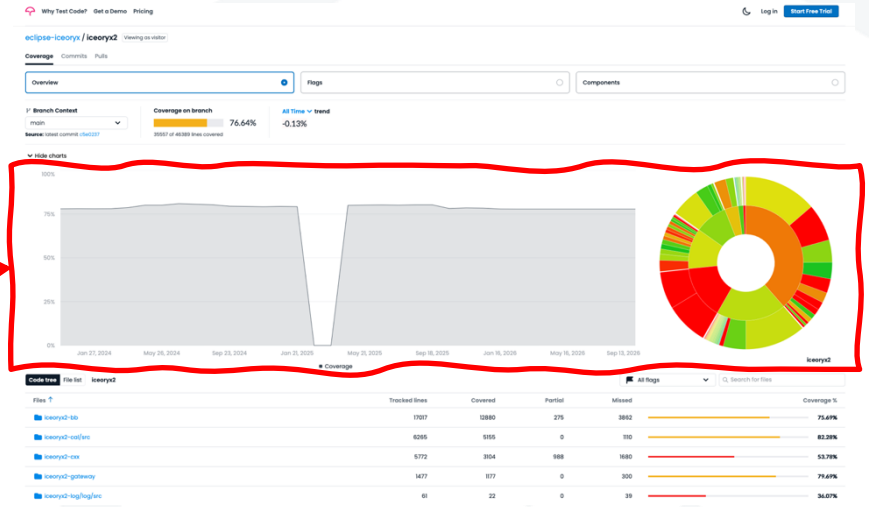


https://github.com/curl/curl/actions

Work in Progress - License: CC-BY-4.0

Code coverage measurements





iceoryx2 - Zero-Copy Lock-Free IPC with a Rust Core

https://app.codecov.io/gh/eclipse-iceoryx/iceoryx2?search=&trend=all%20time

https://github.com/eclipse-iceoryx/iceoryx2

ELISA
Enabling Linux in Safety Applications

Work in Progress - License: CC-BY-4.0

Contribution guides to reduce frustration

- Poor submissions simply delay patches and frustrate both sides
- Clear contribution guidelines help to have proper submissions
- Automated scripts can support formatting and formalities to lower effort of maintainers.

“When you get a friendly response on your mail, it was most likely one of my bots.”

- A kernel maintainer



What is in a patch?

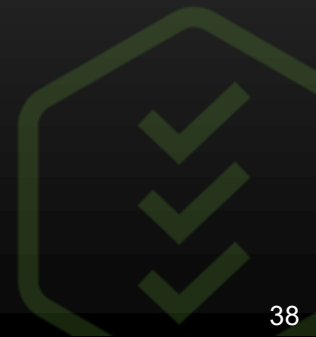
Sending patches

Review

https://wiki.xenproject.org/wiki/Submitting_Xen_Project_Patches

Work in Progress - License: CC-BY-4.0

General Best Practices Aspects



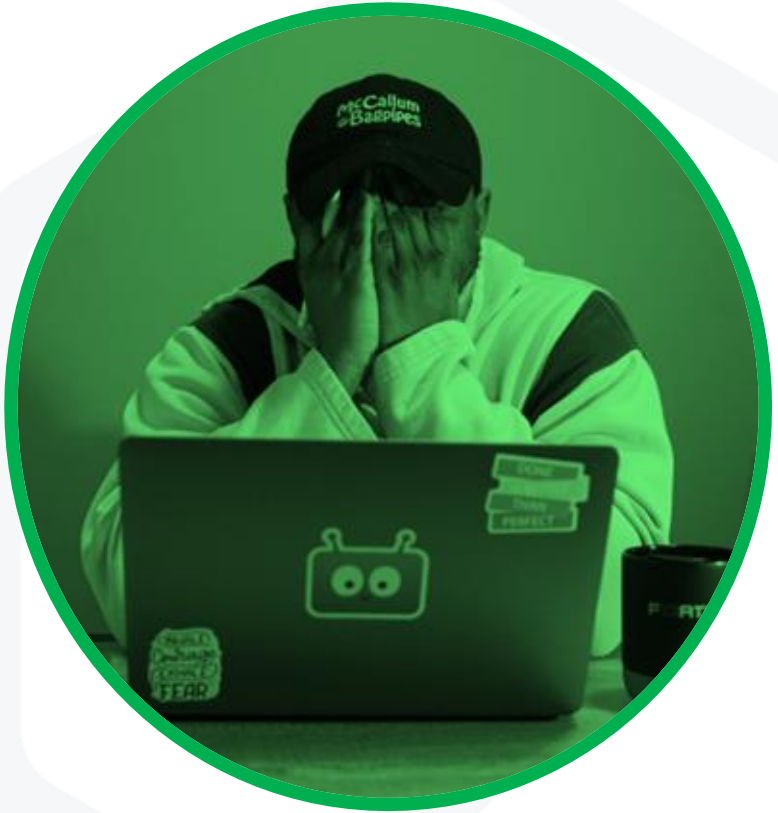
Work in Progress - License: CC-BY-4.0

The role of maintainers

- Committed to their code
- No easy replacement

- Follow their own schedule, quality culture
- Always observed by the public
- Strong entrepreneurship

- Often paid by a company
  (for many “hardened” projects industry relies on)



Work in Progress - License: CC-BY-4.0

Photo by Edemekong Andem on Unsplash

Foundations/Projects provide role definitions

Common roles:

- Maintainer / Committer / Project Lead
- Contributor / Community member

Extended roles:

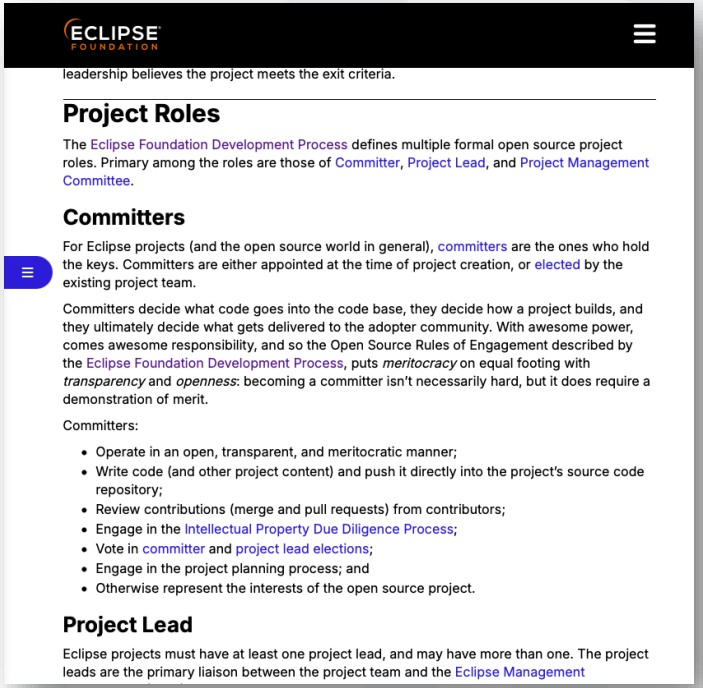
- Governing Board member
- (Technical) Steering Committee member
- Project / Working Group Leader

Seldom/Uncommon roles:

- Quality/Security/Safety Manager

Good to know:

- Roles vary by project or foundation
- Maintainers file is very common



ECLIPSE FOUNDATION

leadership believes the project meets the exit criteria.

Project Roles

The Eclipse Foundation Development Process defines multiple formal open source project roles. Primary among the roles are those of Committer, Project Lead, and Project Management Committee.

Committers

For Eclipse projects (and the open source world in general), committers are the ones who hold the keys. Committers are either appointed at the time of project creation, or elected by the existing project team.

Committers decide what code goes into the code base, they decide how a project builds, and they ultimately decide what gets delivered to the adopter community. With awesome power, comes awesome responsibility, and so the Open Source Rules of Engagement described by the Eclipse Foundation Development Process, puts *meritocracy* on equal footing with *transparency* and *openness*: becoming a committer isn't necessarily hard, but it does require a demonstration of merit.

Committers:

- Operate in an open, transparent, and meritocratic manner;
- Write code (and other project content) and push it directly into the project's source code repository;
- Review contributions (merge and pull requests) from contributors;
- Engage in the Intellectual Property Due Diligence Process;
- Vote in committer and project lead elections;
- Engage in the project planning process; and
- Otherwise represent the interests of the open source project.

Project Lead

Eclipse projects must have at least one project lead, and may have more than one. The project leads are the primary liaison between the project team and the Eclipse Management

https://www.eclipse.org/projects/handbook/#roles

ELISA
Enabling Linux in Safety Applications

Work in Progress - License: CC-BY-4.0

Role of Eclipse S-CORE process

Work in Progress - License: CC-BY-4.0

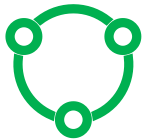
Process model and its implementation

The Project Management Plan:

- Describes the process implementation („makes a meta-model concrete“)
- It provides the tailoring to practices
- It mentions the tools which are used

The Process Description (a meta-model):

- Tool reference kept out wherever possible
- Links standards to the process artifacts (ISO 26262, ISO 21434, ASPICE)
- Defines work products and provides guidance as well as templates



The Docs-as-code tooling implements the process with traceability, checks, gating, and more.

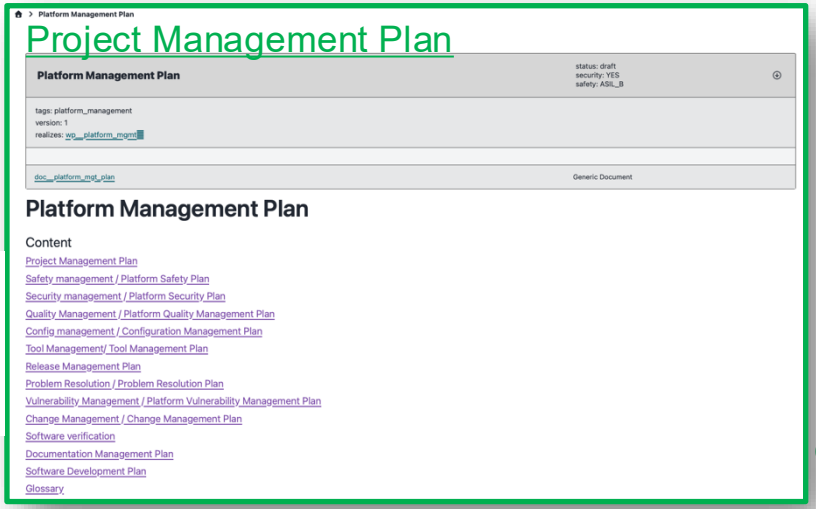
Good to know: Docs-as-code and process_description can be used standalone

Work in Progress - License: CC-BY-4.0

Process model and its implementation

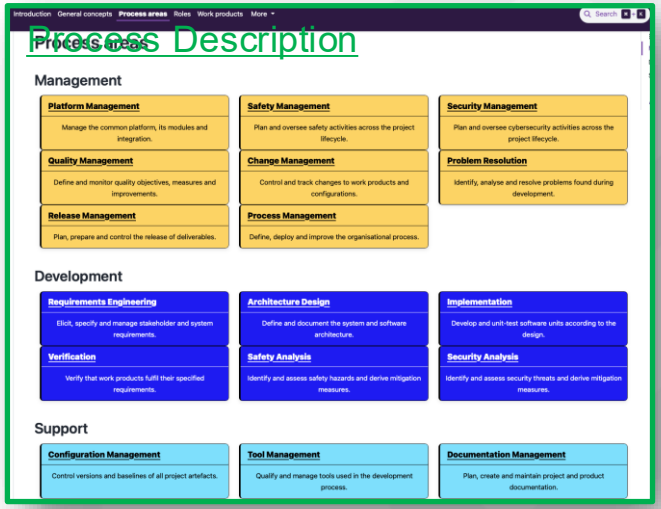
handout slide

Project Management Plan



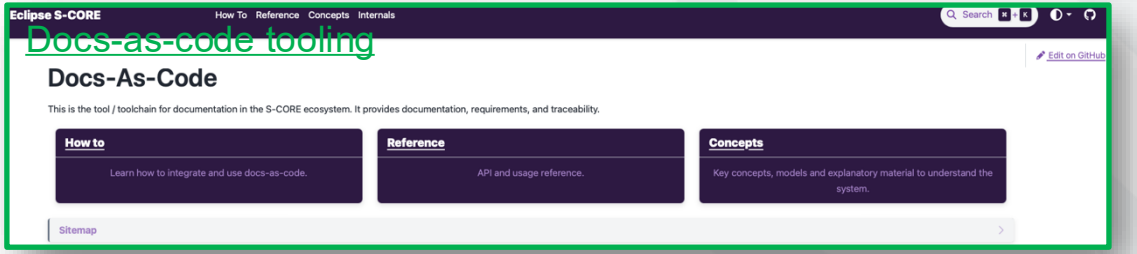
https://eclipse-score.github.io/score/main/platform_management_plan/index.html

Process Description



https://eclipse-score.github.io/process_description/main/standards/index.html

Docs-as-code tooling



https://eclipse-score.github.io/docs-as-code/main/

ELISA
Enabling Linux in Safety Applications

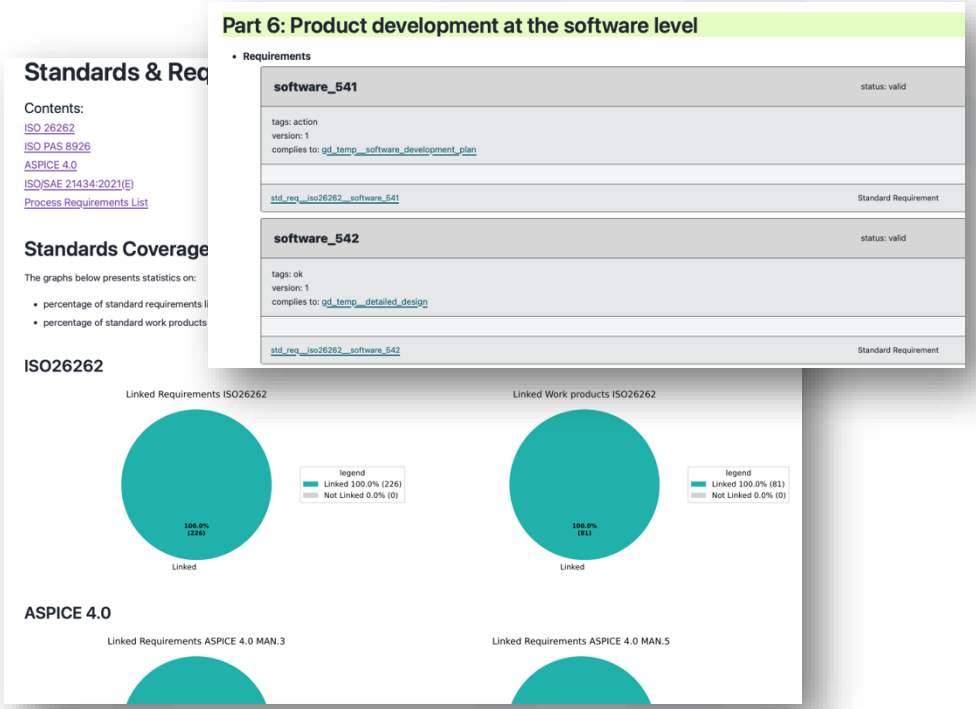
Work in Progress - License: CC-BY-4.0

Dealing with standards

- Standards hold a copyright
- Simply taking the text from standards is not possible
- Discussed with ISO on how to reference best
- Abstract reference to chapters possible: software_541 = Part 6 chapter 5.4.1

Good to know:

Creating work products and following the process enables a standard compliant development

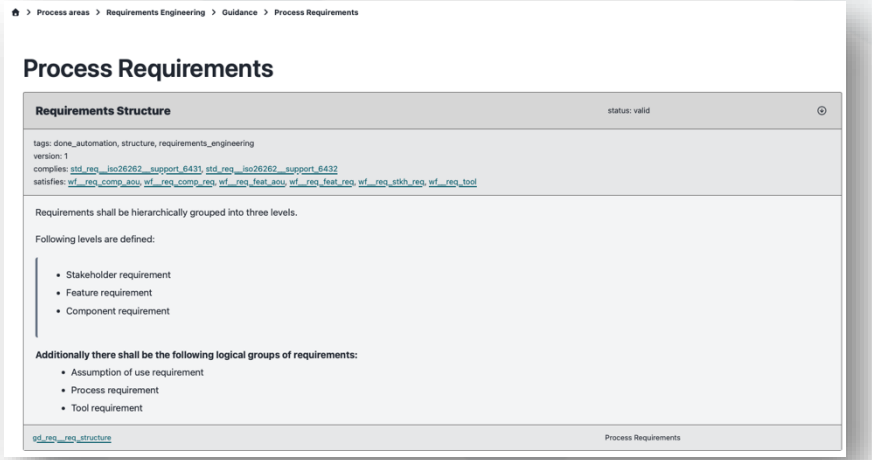


https://eclipse-score.github.io/process_description/main/standards/index.html#

Work in Progress - License: CC-BY-4.0

The role of tools (requirements)

- Manual inspections/reviews are limited to a bare minimum
- Automation is used wherever (reasonably) possible
- Docs-as-code is preferred over any binary format document
  (this benefits also further data processing)

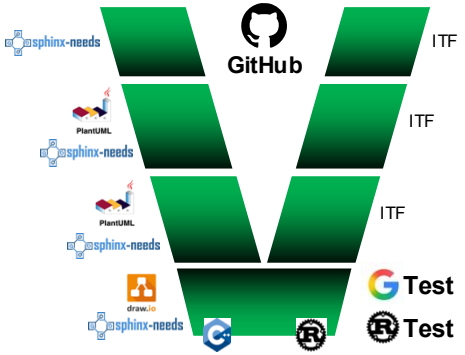


https://eclipse-score.github.io/process_description/main/process_areas/requirements_engineering/guidance/requirements_process_reqs.html

Work in Progress - License: CC-BY-4.0

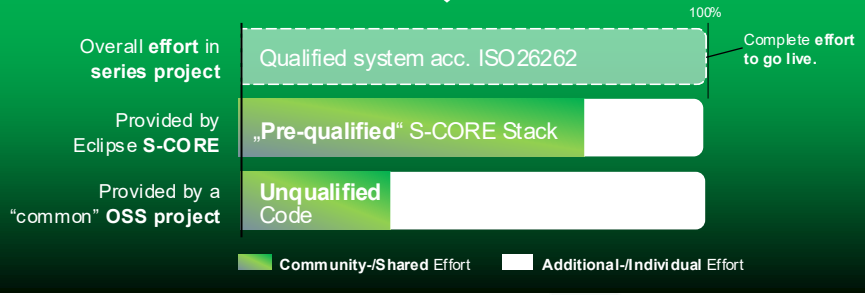
Tight coupling of process & tools + regular audits

Fully open-source toolchain established.



7 external process audits completed/ findings implemented,
1 more audit to go to cover **complete SW-development process**.





!

*“With S-CORE, the **shared content** for the industry is **increased by far**,…*

*… nevertheless, **some effort** is still **needed** to reach a fully qualified system.*

Work in Progress - License: CC-BY-4.0

How Eclipse S-CORE differs to traditional OSS

What you mostly will not see in a random (but mature) OSS project:

- Strong Automotive influence from the inception.
- Mostly driven by industrial & paid contributors.
- Heavy tool support to enforce processes (not only CI/CD)
- Respect various requirements from ISO 26262, ISO 21434 and ASPICE as part of the development
- Regular accompanied auditing

Good to know: Eclipse S-CORE is an automotive project under Foundational Open Source Stewardship by the Eclipse Foundation!

ELISA
Enabling Linux in Safety Applications

Work in Progress - License: CC-BY-4.0

Summary

Work in Progress - License: CC-BY-4.0

From OSS projects into companies

- Open source practices can provide substantial quality evidence.
- The evidence must be translated, evaluated, and placed into context.
- Open source practices represent modern software development
  and can be incorporated into a company's quality management system
- The goal is not to force open source into a V-model,
  but to make its proven practices understandable and usable for quality management.

Do not ask only whether an open source project follows your process.
Ask whether its practices provide the evidence needed for your quality argument.

Work in Progress - License: CC-BY-4.0



Join our efforts!

Aerospace · Automotive · Linux Features · OS Engineering Process · Safety Architecture · Systems · Tools · Lighthouse · Space Grade Linux

Get Involved

- Lighthouse SIG meets every other Friday 12:00 PM
  https://elisa.tech/community/meetings/
- Mailing list
  https://lists.elisa.tech/g/lighthouse
- Discord
  https://discord.gg/TnWZpDUu
- Contributions in GitHub welcome!
  https://github.com/elisa-tech/lighthouse-oss/
  https://github.com/elisa-tech/lighthouse-oss/wiki



ELISA
Enabling Linux in Safety Applications

Work in Progress - License: CC-BY-4.0