embeddedworld
Exhibition&Conference

Open Source for Safety-Critical Systems: A Landscape Exploration

Philipp Ahmann
Sr. OSS Community Manager, ETAS GmbH

whoami - Philipp Ahmann

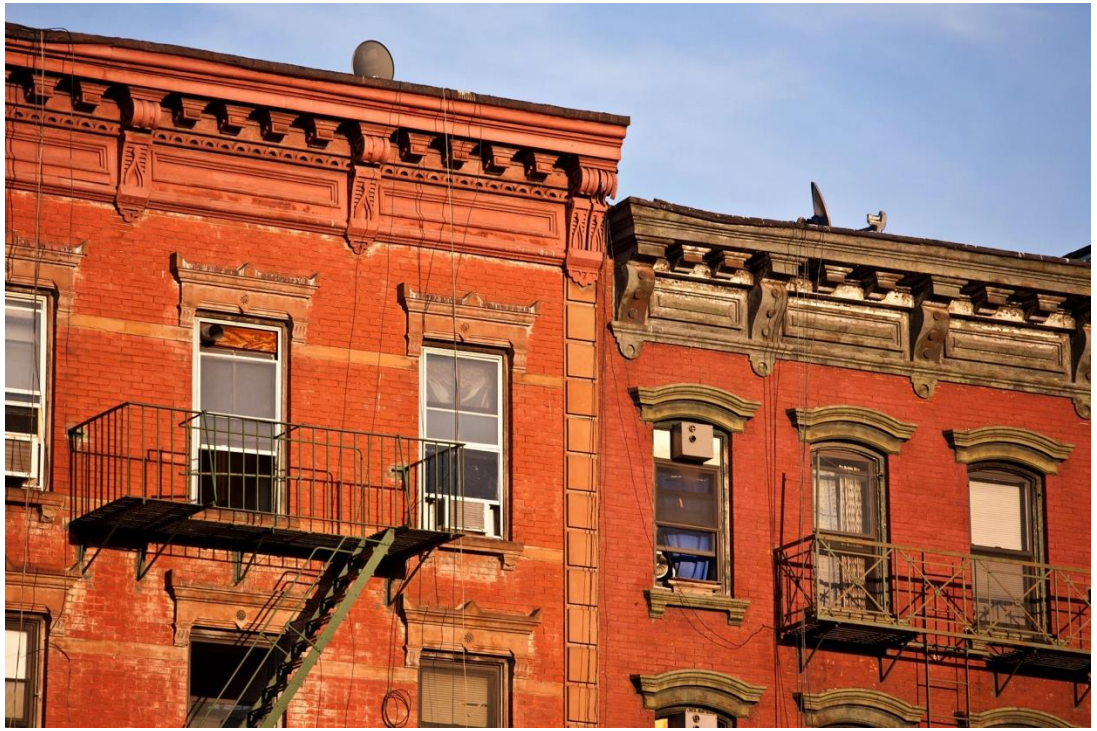
ETAS	Sr. OSS Community Manager Automotive OSS Process Lead
ELISA Enabling Linux in Safety Applications	Chair of the Technical Steering Committee Lead of the Systems Working Group
THE LINUX FOUNDATION | Europe	Advisory Board Member of LF Europe
Eclipse SDV	Eclipse Safe Open Vehicle Core Committer
	OSS enthusiast and promoter

What is functional safety?

And what is the difference to cyber security?

What is functional safety?

In some languages safety & security are the same word!

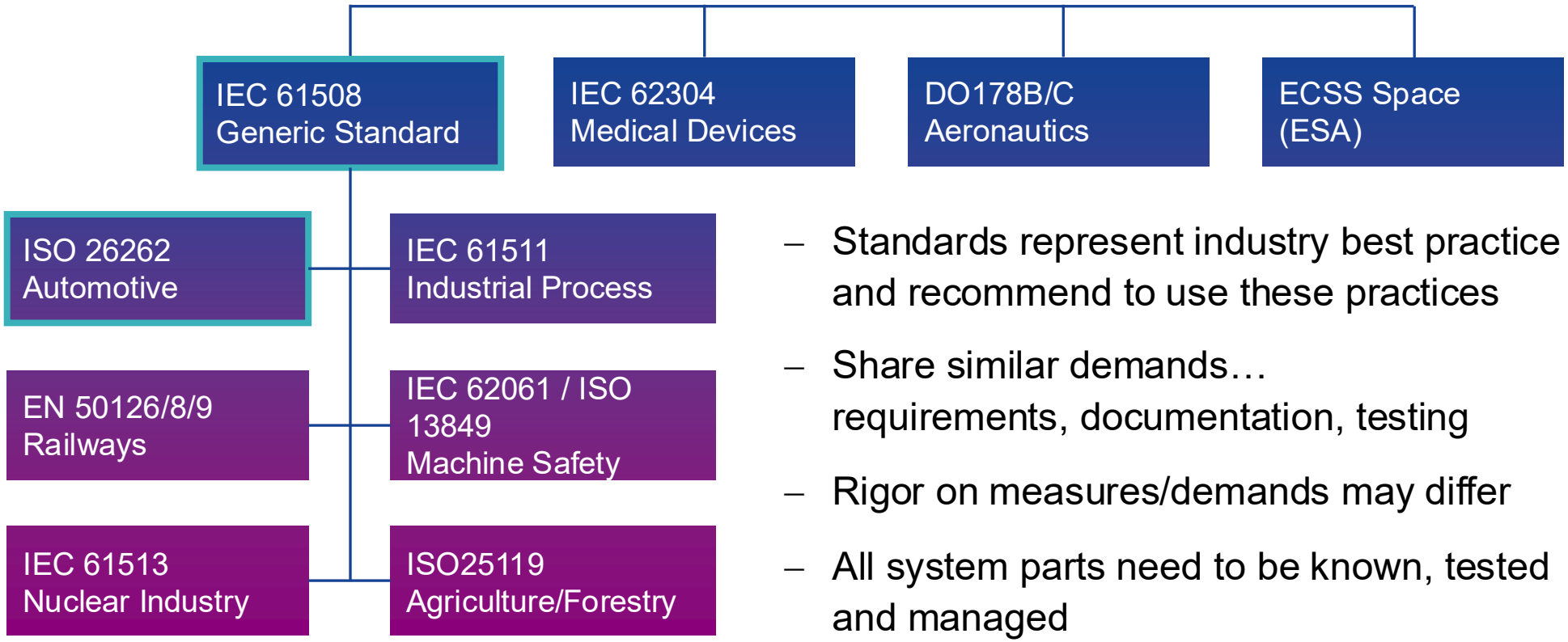


Safe, but not secure!

Secured, but not safe!

What is functional safety?

Samples of safety (integrity) standards



- Standards represent industry best practice and recommend to use these practices
- Share similar demands… requirements, documentation, testing
- Rigor on measures/demands may differ
- All system parts need to be known, tested and managed

The Fundamental Challenge

When adopting/applying these standards to Open Source

Traditional development processes / v-model

VS

code centric open source development

Standard checklist-based approaches

VS

collaborative development

The need for (formal) traceability

and

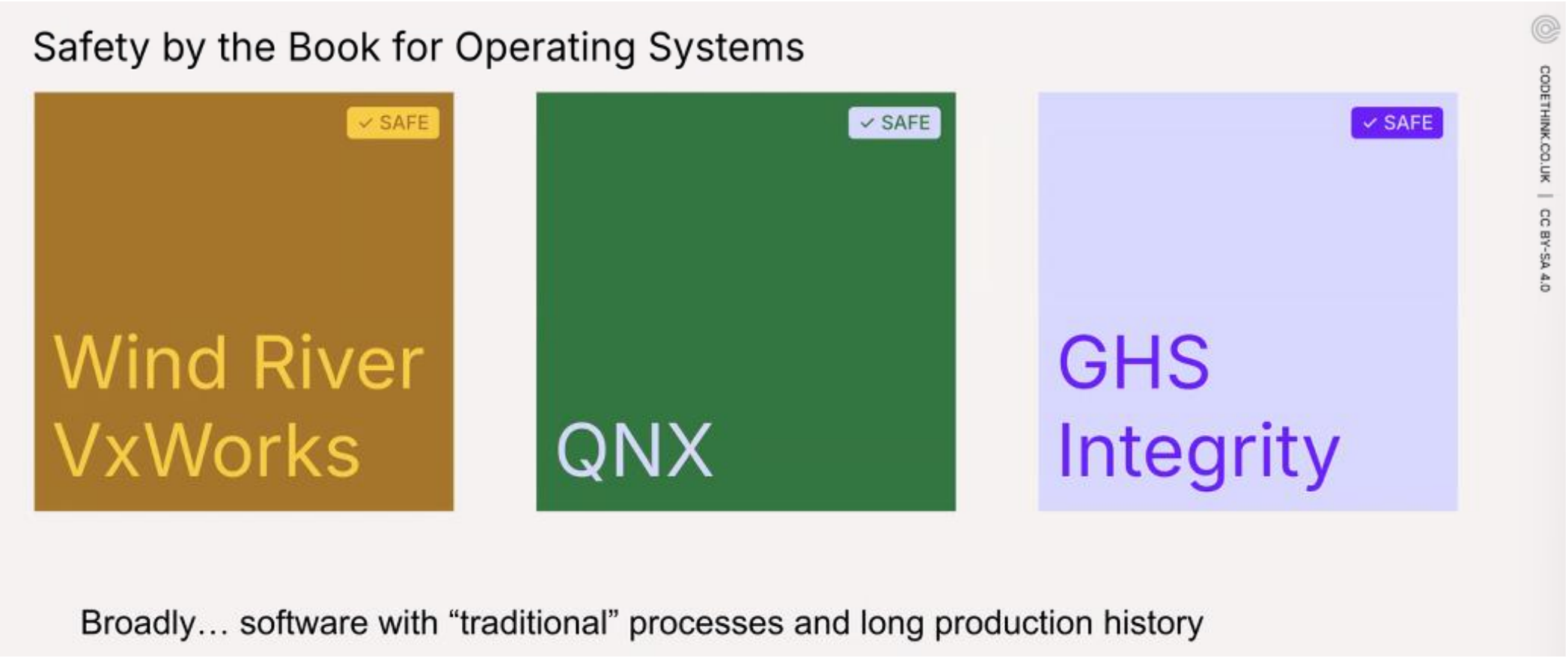
documentation in safety-critical systems

Setting the scene

The evolving landscape

Where do we come from?

Credits to Codethink, where I took the slide from!



Introduction & Motivation

Processes described by standards vs.

- Safety integrity standards need to adopt to increasing complexity of products
- Safety requires a robust fundament based on processes, technical measures and statistical analysis
- Growing industry interest in open source for safety-certified applications
- Current challenges in integrating open-source solutions with safety standards

(China is already making heavy use of Open Source in Automotive systems)

Where the hype may have started… - The Linux world

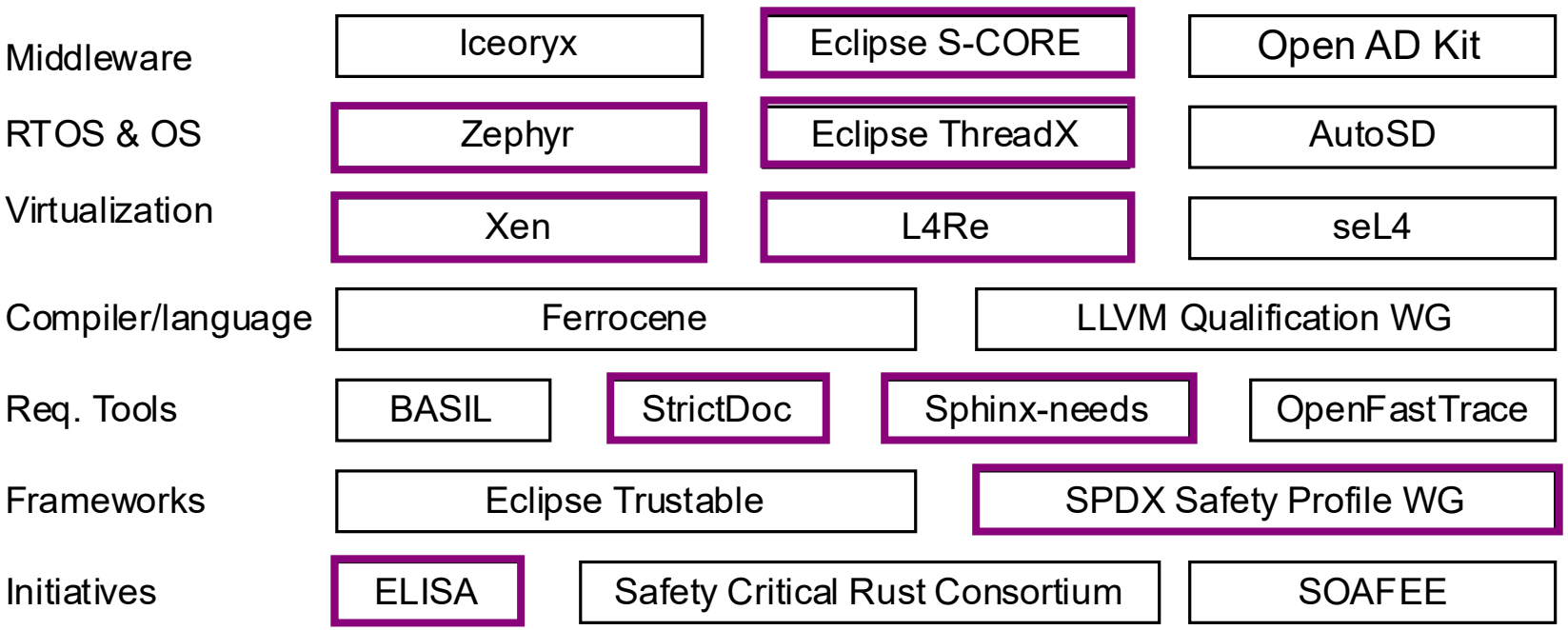
Safe <x>, Safety <x>, <x> for safety, SIL qualified…

- Open source software superlative.
- Largest community, largest source base.
- Made for flexibility and wide use cases.
- Spread over whole world and in space.
- Several attempts with certification path.
- Gains again momentum for high performance products (e.g. SDV*)
- Prominent open space examples: SIL2LinuxMP and ELISA

SDV: Software-Defined-Vehicle

Example of the wider safety open source landscape

It is not all about Linux … spot check



Note: Mapping may have some (smaller) inconsistencies

Example of the wider safety open source landscape

Common theme

- Argument of „OSS development is not organized like commercial software“
- Less influence on maintainers (positive & negative – no traditional supplier management)
- Harder to train/direct developers
- Liability of a community? (but commercial provider may be liable – insurance)
- Development process: Requirements, traceability, v-model,… mapping safety integrity standards

Towards safety certification

Ways provided by standards and used by projects

Route to safety certification

The most common/typical approaches for pre-existing OSS software?!

- IEC 61508 Route 3S for pre-existing software
- ISO 26262-8 clause 12 for (less complex) automotive applications
- ISO PAS 8926 as a bridge for complex software (on its way towards ISO 26262 3rd edition)
- SEooC: ISO 26262-10 clause 9 + ISO 26262-6 for standalone software components

Route to safety certification

… and some more options to get OSS into safety critical systems (handle with care)

– Decomposition
  → OSS becomes QM

– Mixed-criticality (not always allowed)
  → OSS for QM parts in SIL system

– Tool qualification
  (questionable for device software)

– ISO 26262-8 clause 14 aka Proven in use
  (very questionable)

Proven in use (PIU) on the example of Linux

One version of X used in same way taken over to same/comparable use case

– Linux is PIU in many industries and use cases?
→ True, but ISO 26262 PIU is not about popularity or maturity - it's about demonstrable evidence for the same item in a comparable safety context.

– Linux can be found in ADAS L2+ systems today?
→ True, but even if Linux is used in ADAS L2+ systems, those implementations are highly customized and not representative of your specific system.

– Linux distributions vary widely (kernel versions, patches, configurations, drivers, HW platforms).
→ **Use this as an argument for diversity.**

Tool qualification on the example of Qt Safe Renderer

Confidence in the use of software tools

- Carefully read the certificate as first indication.
- Try to get the full assessment report (e.g. mentioned, but not listed at QT website)

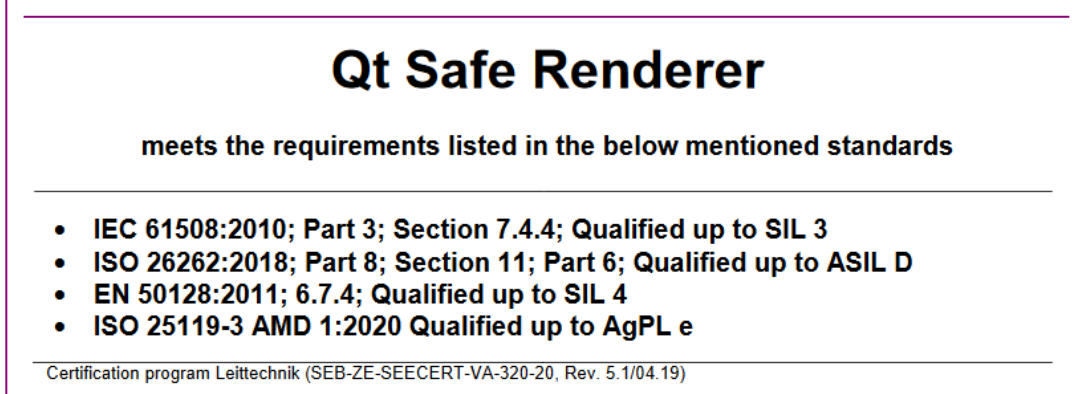
- ISO 26262 has part 6 mentioned up to ASIL-D
- The other standards refer to tool qualification

- Question:

Is the certification really relevant for me?

"The Qt Safe Renderer provides a UI rendering component that can be used to render safety-critical items, such as warning indicators, in functional safety systems."

https://doc.qt.io/QtSafeRenderer/
https://doc.qt.io/QtSafeRenderer/qtsr-delivery.html



Qt Safe Renderer

meets the requirements listed in the below mentioned standards

- **IEC 61508:2010; Part 3; Section 7.4.4; Qualified up to SIL 3**
- **ISO 26262:2018; Part 8; Section 11; Part 6; Qualified up to ASIL D**
- **EN 50128:2011; 6.7.4; Qualified up to SIL 4**
- **ISO 25119-3 AMD 1:2020 Qualified up to AgPL e**

Certification program Leittechnik (SEB-ZE-SEECERT-VA-320-20, Rev. 5.1/04.19)

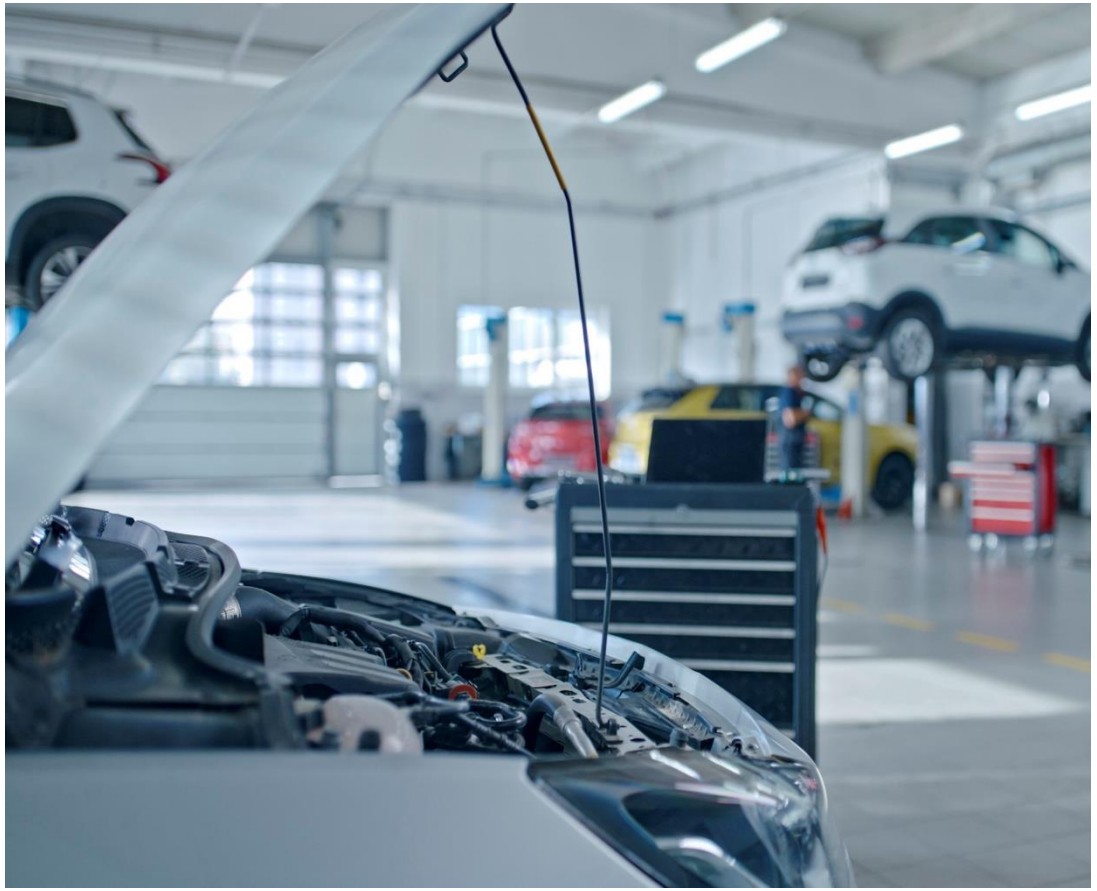
SEooC is a SEiaC

You always assume a context. This is why there are assumptions of use. Check them!

What is the probability
that a car will fall on your head?

SEooC is a SEiaC

You always assume a context. This is why there are assumptions of use. Check them!



Understanding your Hardware is crucial (incl. FFI)

A practical example: ARM Trusted Firmware can stop CPUs for security reasons…

Project examples

Product Software

Project approaches

Various starting points exist…

– „Enable Safety afterwards“ → touched in previous slides

– Start from “Security footprint & Critical industries” project

– “Certification in mind” (from beginning)

– Control the development environment & OSS project e.g. by single vendor
“Certified and Accredited Open Source”

– Certification before going open source

→ Solution needs to be viable

Security footprint & Critical industries

Example: Xen Project

- Situation: Since Xen for embedded, security WG was started in parallel (in 2010)
  - Widely adopted in critical production environment (Data center, Desktop & Embedded)
  - Community brings quality awareness

- Implemented measures:
  - Rigorous Quality Process. Full commit traceability.
  - Security & isolation are project's top priority
  - Commits are tested with 2 CI loops.
  - Misra compliance

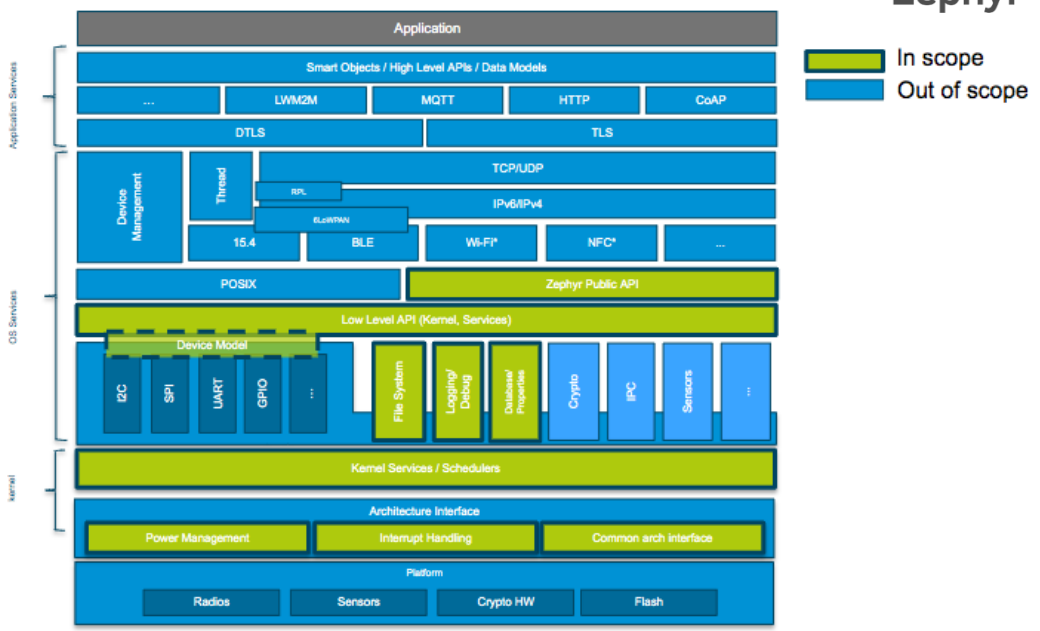
Certification in mind (wide use cases)

Example: Zephyr RTOS

– Situation: Targeting wide use cases beyond safety & security, but consider safety certification from the beginning

– Resulting challenge: heterogenous community with manifold non-safety-critical use cases

– Implemented measures:
tools and processes established early
  – Traceability & Requirements → StrictDoc
  – MISRA checks
  – Certification of code subsets.
  – IEC 61508 Route 3S



https://www.zephyrproject.org/introduction-of-coding-guidelines-for-zephyr-rtos/

Certified and Accredited Open Source

Example: L4Re

KERNKONZEPT

– Situation:
  – Ownership by a single vendor eases certification of the software

– Benefits by projects:
  – Easy adoption of processes to new demands

– Resulting challenges:
  – Balance community and certification/accreditation demands
  – Less sharing of development cost

→ L4Re association path

Hypervisor: L4Re by Kernkonzept
https://l4re.org/overview.html

L4 Re

– Planned for 2026: L4Re will get its own home in the L4Re association, with the goal of securing its long-term independence and creating a foundation for a community.
– Certified and accredited versions of L4Re are also available from Kernkonzept
– VS GEHEIM, NATO SECRET, COMMON CRITERIA EAL 4+, ASIL D is coming in Q4 2026, L4Re Safety Certification Kit

Doing the “hard thing” before

Example: ThreadX (former Azure RTOS)

– Situation/Benefits:
  – Certified when going open source

– Risks/Challenges for the project:
  – Onboarding community
  – Add community contributions to the established process
  – Liability of an OSS foundation
  – Certification cost covering

– Measures taken:
  – Certification organization
  – Commercial model

RTOS: ThreadX at Eclipse (Microsoft)
https://threadx.io/

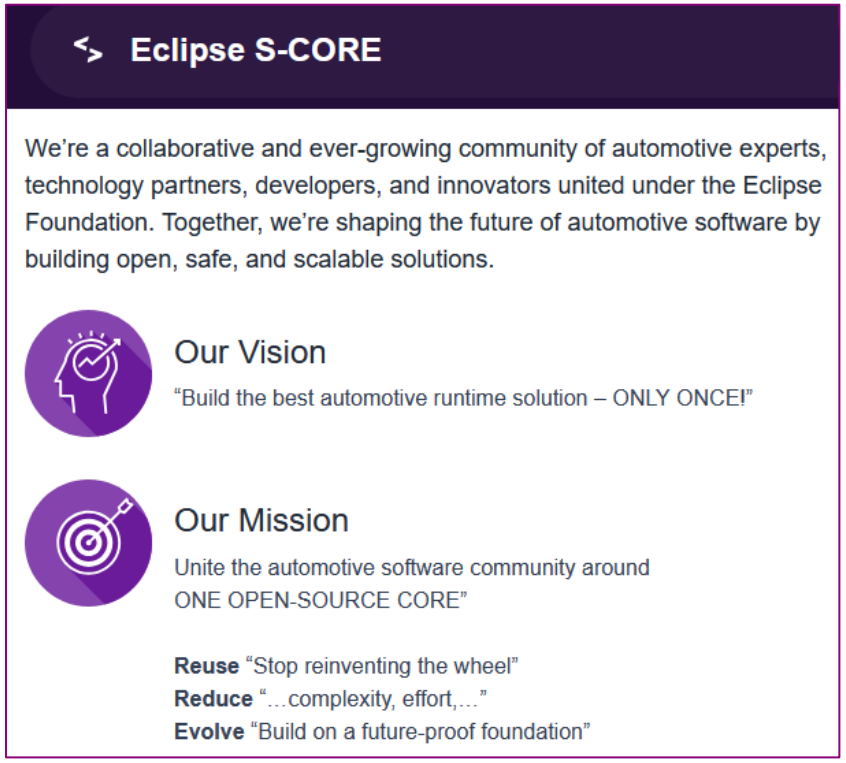
– This RTOS is designed for deeply embedded applications. It provides advanced scheduling facilities, message passing, interrupt management, and messaging services.
– Eclipse ThreadX RTOS has many advanced features, including picokernel architecture, preemption threshold, event chaining, and a rich set of system services.

Combining various approaches

Example: Eclipse Safe Open Vehicle Core (S-Core)

– Situation:
  – Security footprint & Critical industries *Xen*
    → Automotive focus
  – Certification in mind *Zepyhr*
    → Develop process & SW in parallel
  – Overcome single vendor OSS *L4Re*
    → Execute project in Eclipse Foundation
  – Doing the hard thing before *ThreadX*
    → Liability and certification cost transferred.

– Challenges:
  – Clash of Automotive & OSS world
  – Community focus on Automotive professionals
  – “Certification ready”, but no actual certification



Repeating theme: Not fully open everything*

Open Core – your business case

- Source code is typically available

- Limited view on tests, requirements, traceability…

- Safety artifacts often behind a paywall
  - By commercial company
  - Through membership fees

- Secure business with:
  - Commercial extensions and add-ons
  - Alternative & compatible implementations (safety & non-safety derivative)

Business: Liability/Certification, Add-Ons, SDK/IDE, Customer support & maintenance

**Ferroscene Certified Rust Compiler opens everything*

Project examples

Tools and Infrastructure

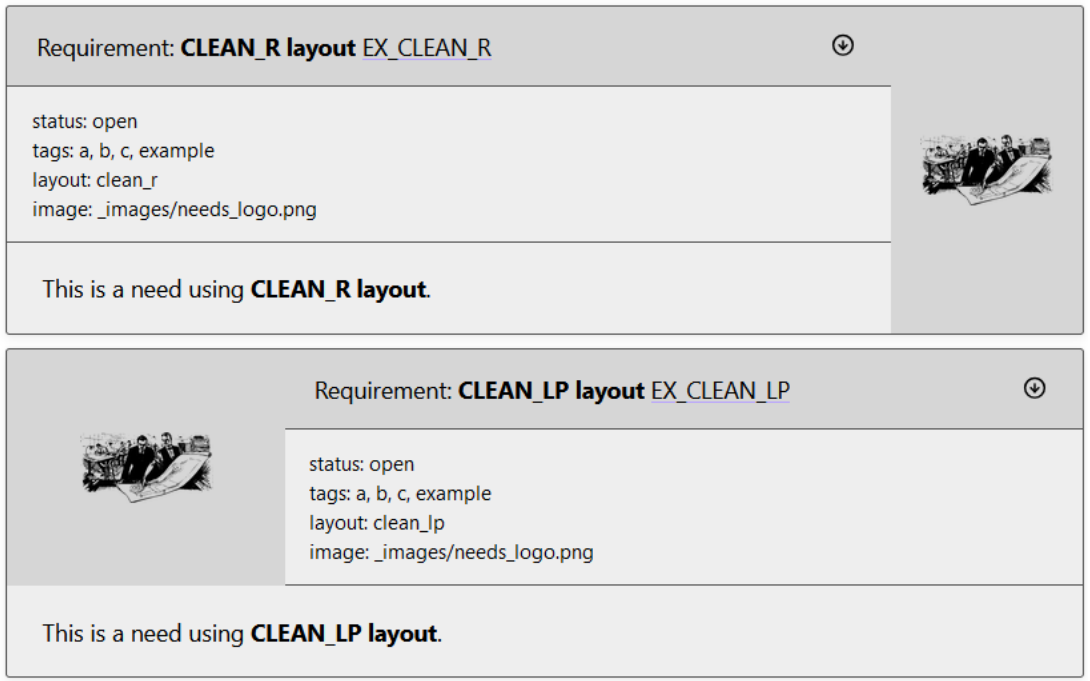
Sphinx-needs

Examples of OSS Requirement Tools

Sphinx-needs

- https://sphinx-needs.readthedocs.io
- Combine Docs-as-Code with Application Lifecycle Management, to track requirements, specifications, test cases, and other engineering objects in your documentation

Used e.g. by: Eclipse S-Core project

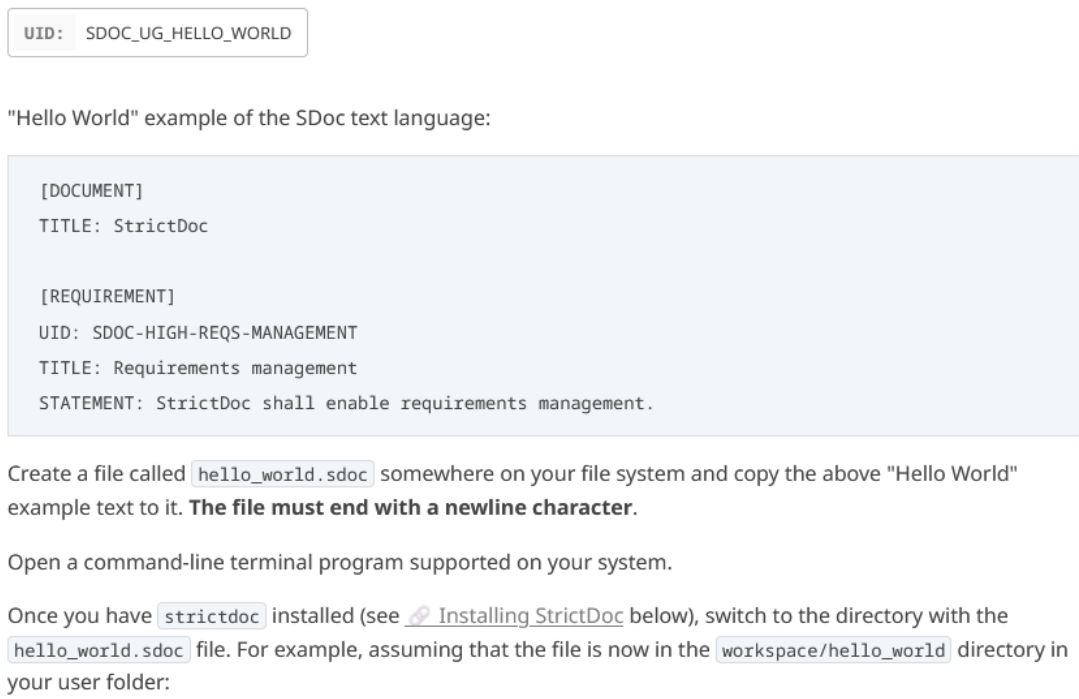


StrictDoc

Examples of OSS Requirement Tools

- https://strictdoc.readthedocs.io
- StrictDoc efficiently manages requirements and specifications using a human-readable DSL (SDoc), generating output in multiple formats (HTML, PDF, etc.) via a web UI. Key features include traceability, customizable fields (e.g., ASIL, priority), and fast, incremental generation. See limitations for details.
- Developed with “safety in mind”

Used e.g. by: Zepyhr project



`UID:` SDOC_UG_HELLO_WORLD

"Hello World" example of the SDoc text language:

```
[DOCUMENT]
TITLE: StrictDoc

[REQUIREMENT]
UID: SDOC-HIGH-REQS-MANAGEMENT
TITLE: Requirements management
STATEMENT: StrictDoc shall enable requirements management.
```

Create a file called `hello_world.sdoc` somewhere on your file system and copy the above "Hello World" example text to it. **The file must end with a newline character**.

Open a command-line terminal program supported on your system.

Once you have `strictdoc` installed (see Installing StrictDoc below), switch to the directory with the `hello_world.sdoc` file. For example, assuming that the file is now in the `workspace/hello_world` directory in your user folder:

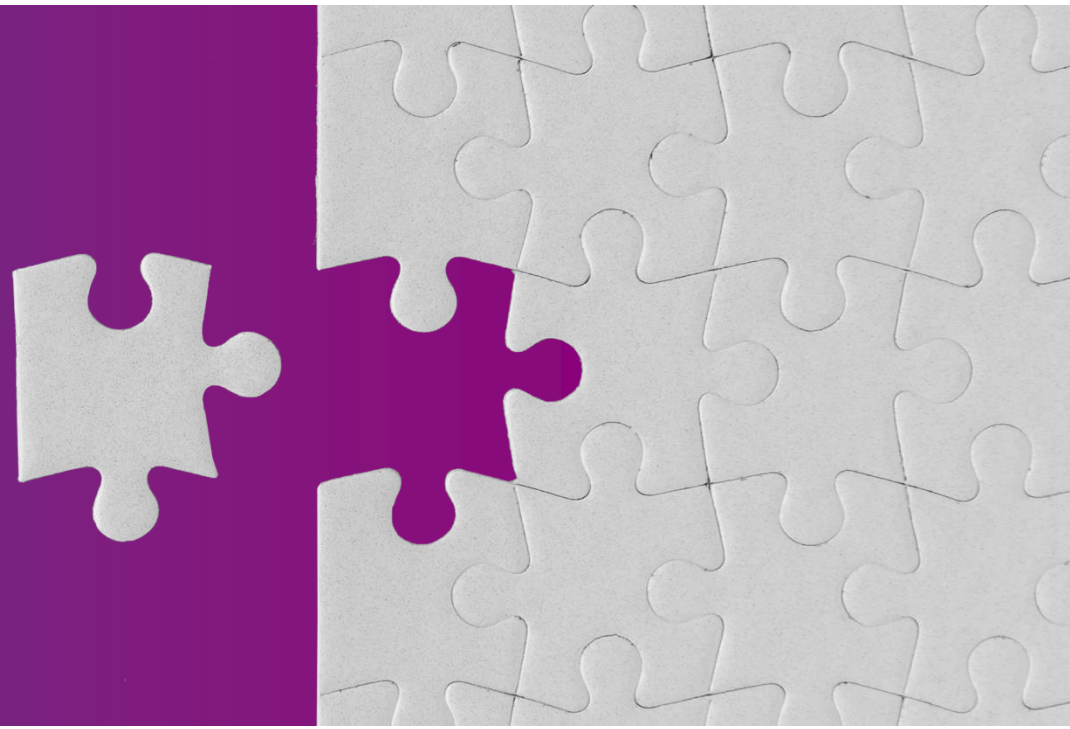
The broader requirements tools ecosystem

Examples of OSS Requirement Tools

There are far more OSS requirements tools out there than presented!

This is just a small selection used in safety critical (OSS) projects!

(It does not mean that others are less suitable. It is your choice!)



Quick hits:
https://github.com/itsallcode/openfasttrace | https://basil-the-fusa-spice.readthedocs.io/ | https://doorstop.readthedocs.io
https://github.com/osrmt/osrmt | https://goeb.github.io/reqflow/ | https://github.com/topics/requirements-tracing

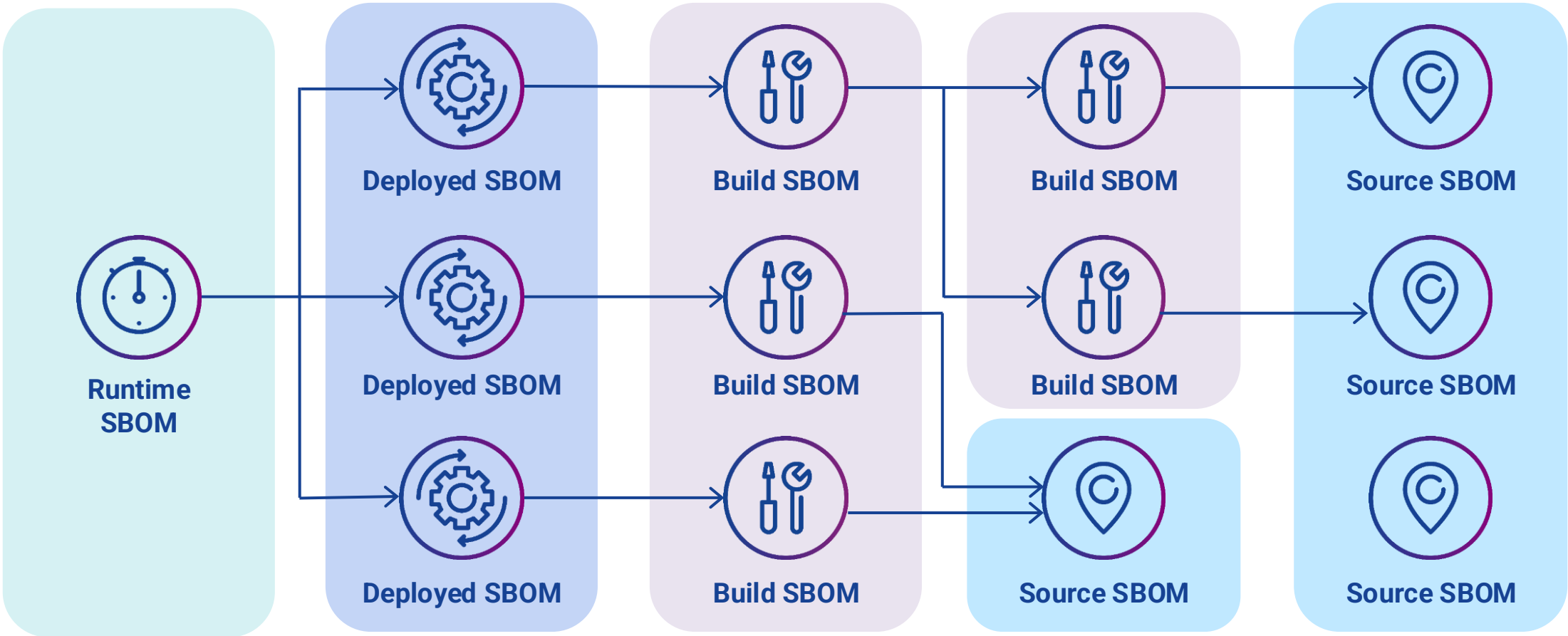
SPDX Safety Profile

Safety SBOM

- Profile team formed in August 2022
- Mailing list: https://lists.spdx.org/g/spdx-fusa
- Scope:
  - Provide a complete model of dependencies in a safety related project
  - Support effective impact analysis methodologies
    (input information for FMEA, Ishikawa Analysis, GSN/SACM etc.)
  - Provide reproducible results in both impact analysis and evidence generation
  - Formal way to demonstrate completeness after project tailoring and for different scopes

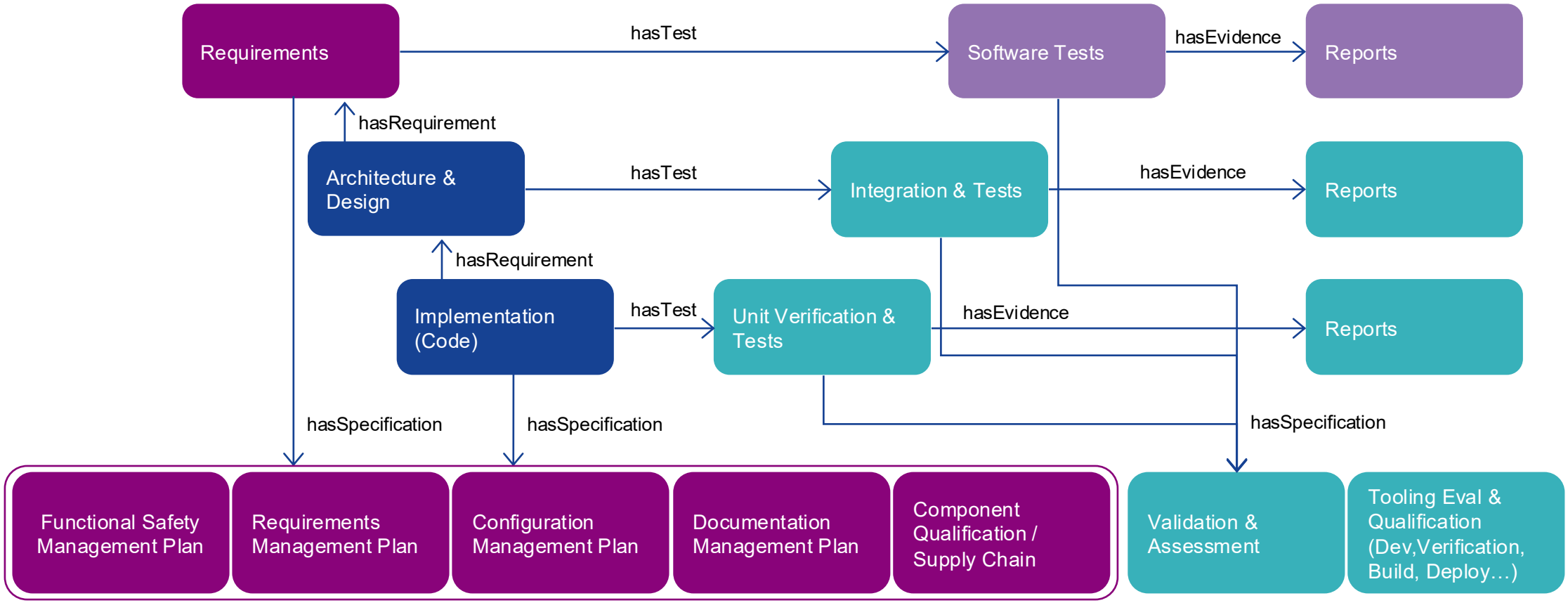
SPDX Safety Profile

Understanding Safety Critical System: Traceability



SPDX Safety Profile

SPDX Safety Dependencies in a FuSa Project



Concluding thoughts

Safe <x>, Safety <x>, <x> for safety, SIL qualified…

Wording does not tell you what it is!

What we may have learnt:

While we are in regulated industries with lots of definitions within safety integrity standards…

…nobody clearly defines/limits the words being used by marketing!

Practical steps to conduct:

- Understand the system & its context sufficiently.
- Check where safety is actually allocated.
- Check reports & certificates carefully.
- Proof the safety net/watchdog effectiveness.

Thank you