

# Zephyr: Journey to Safety Certification

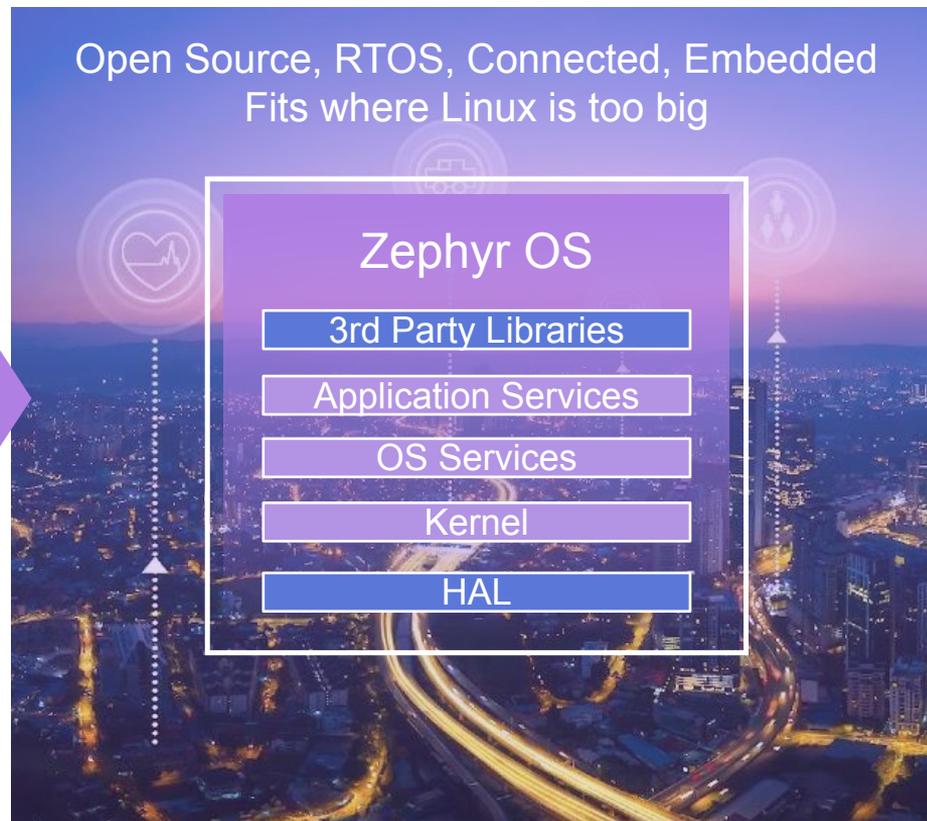
Kate Stewart, Zephyr Project Director, The Linux Foundation  
Nicole Pappler, Zephyr Functional Safety Manager, AlektoMetis



# Zephyr Project



- **Open source** real time operating system
- **Developer friendly** with vibrant community participation
- Built with **safety and security** in mind
- **Broad SoC, board and sensor support.**
- **Vendor Neutral** governance
- **Permissively licensed** - Apache 2.0
- **Complete**, fully integrated, highly configurable, **modular** for **flexibility**
- **Product** development ready using LTS includes **security updates**



# Products based on Zephyr



**Oton More  
Hearing Aid**



**Lildog & Lilcat  
Pet Tracker**



**Livestock Tracker**



**Moto Watch 100**



**Samsung Galaxy  
Ring**



**Proglove**



**Adhoc Smart Waste**



**Google  
Chromebook**



**Framework laptop**



**Keeb.io BDN9**



**Hati-ACE**



**Safety Pod**



**BLiXT solid state  
circuit breaker**



**Aethero Deimos  
Satellite**



**PHYTEC Distancer**



**Laird Connectivity  
sensors & gateways**



**BeST pump  
monitoring**



**Vestas Wind  
Turbines**

# 700+ supported boards... and growing



**Arduino Portenta H7**



**ESP32**



**Sipeed HiFive1**



**nRF9160 DK**



**STM32F746G Disco**



**M5StickC PLUS**



**TDK RoboKit 1**



**BBC micro:bit v2**



**Blue Wireless Swan**



**Arduino Nano 33 BLE**



**Intel UP Squared**



**Dragino LSN50 LoRA Sensor Node**



**Microchip SAM E54 Xplained Pro Evaluation Kit**



**Raspberry Pi Pico**



**Altera MAX10**



**NXP i.MX8MP EVK**



**Adafruit Feather M0 LoRa**



**u-blox EVK-NINA-B3**



[docs.zephyrproject.org/latest/boards](https://docs.zephyrproject.org/latest/boards)

# 220+ Sensors Already Integrated



adt7420  
adx1345  
adx1362  
adx1372  
ak8975  
amg88xx  
ams\_as5600  
ams\_iAQcore  
apds9960  
bma280  
bmc150\_magn  
bme280  
bme680  
bmg160  
bmi160  
bmi270  
bmm150  
bmp388  
bq274xx  
ccs811

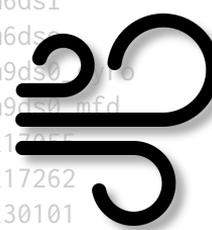
dht  
dps310  
ds18b20  
ens  
esp8266  
fdd  
fxos8560  
fxos9560  
grove  
grow\_r502a  
hmc58831  
hp206c  
ht1221  
i2c-gpio  
i2c-gpio-50c  
i2c-gpio-605  
i2c-gpio-670  
i2c-gpio-72  
icp-125  
iis2dh  
iis2dlpc



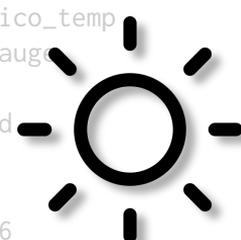
iis2iclx  
iis2mdc  
iis3dhhc  
ina219  
ina230  
isl2935  
ism330dhc  
ite\_tach\_it8xxx2  
ite\_vcmp\_it8xxx2  
lis2dh  
lis2ds12  
lis2dw12  
lis2tr  
lm75  
lm77  
lps22  
lps22hh  
lps25hb  
lsm303dlhc\_magn



lsm6ds0  
lsm6dsl  
lsm6dsx  
lsm9ds0  
lsm9ds0\_mfd  
max17055  
max17262  
max30101  
max31875  
max44009  
max6675  
mchp\_tach\_xec  
mcp9804  
mcpu\_acmp  
mhz-mpz  
mpr121  
mpu6050  
mpu9250  
ms5607  
ms5837



nrf5  
nuvoton\_adc\_cmp\_npcx  
nuvoton\_tach\_npcx  
nxp\_kinco  
opt3001  
pcnt\_encoder3  
pms7003  
qdec\_mcp  
qdec\_nrfx  
qdec\_sam  
qdec\_stm32  
rpi\_pico\_temp  
sbs\_gauge  
sgp40  
sht3xd  
sht4x  
shtcx  
si7006  
si7055  
si7060



si7210  
sm3511t  
stm32\_temp  
stm32\_vbat  
stmemsc  
stts751  
sx9500  
th02  
ti\_hdc  
ti\_hdc20xx  
tmp007  
tmp108  
tmp112  
tmp116  
vcnl4040  
vl53l0x  
wsen\_hids  
wsen\_itds

# Supported Hardware Architectures



Cortex-M, Cortex-R  
& Cortex-A



x86 & x86\_64



32 & 64 bit

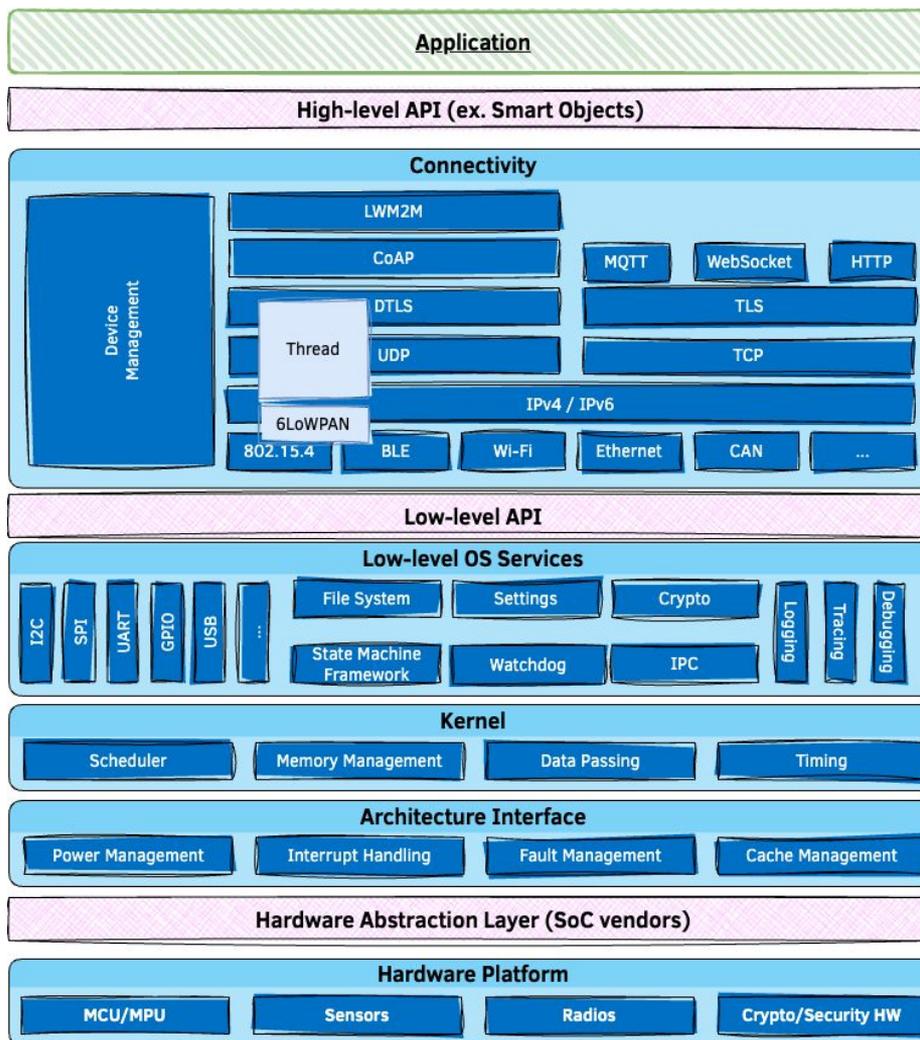


Xtensa



[docs.zephyrproject.org/latest/hardware/index.html#hardware-support](https://docs.zephyrproject.org/latest/hardware/index.html#hardware-support)

# Software Architecture



# Zephyr in the wild... 6.9K Forks!



**About**

Primary Git Repository for the Zephyr Project. Zephyr is a new generation, scalable, optimized, secure RTOS for multiple hardware architectures.

[docs.zephyrproject.org](https://docs.zephyrproject.org)

iot real-time microcontroller  
embedded bluetooth bluetooth-le  
mcu rtos zephyr zephyros  
embedded-c zephyr-rtos

Readme  
Apache-2.0 license  
Code of conduct  
Security policy  
Activity  
Custom properties  
11.3k stars  
407 watching  
**6.9k forks**  
Report repository

**Releases** 127

Zephyr 4.0.0 Latest  
on Nov 15, 2024

+ 126 releases

**Vestas Wind Systems A/S**

3 followers <http://www.vestas.com>

**Popular repositories**

**zephyr** Public  
Forked from [zephyrproject-rtos/zephyr](https://github.com/zephyrproject-rtos/zephyr)

Primary Git Repository for the Zephyr Project. Zephyr is a new generation, scalable, optimized, secure RTOS for multiple hardware architectures.

C 2 1

Source: <https://github.com/vestas-wind-systems>

Source: <https://github.com/zephyrproject-rtos/zephyr>

# Building on POSIX

- **Zephyr apps can run as native Linux applications**
  - Easier to debug/profile with native tools
  - Connect to real devices using TCP/IP, Bluetooth, CAN
  - Helps minimize hardware dependencies during the development phase
- **Re-use existing code & libraries by accessing Zephyr services through POSIX API**
  - Easier for non-embedded programmers
  - Implementation is optimized for constrained systems
  - Supported POSIX subsets: PSE51, PSE52, and BSD sockets

# Safety & Security Focus From the Start



## Exhibit B

### **Zephyr Project Charter (the “Charter”)**

The Linux Foundation  
Updated August 21, 2023

#### **1. Mission of the Zephyr Project (“Zephyr,” or, alternatively, the “Project”).**

The mission of the Project is to:

- a. deliver the best-in-class RTOS for connected resource-constrained devices, built to be **secure and safe.**
- b. maintain an auditable code base, while taking advantage of community participation; this auditable code base is open source;
- c. include participation of leading members of this ecosystem, including micro-controller manufacturers, hardware developers, software developers and other members of the ecosystem; and
- d. host the infrastructure for the open source Project and sub-projects, establishing a neutral home for community meetings, events and collaborative discussions and providing structure around the business and technical governance of the Project.

Source:

<https://zephyrproject.org/wp-content/uploads/2023/08/LF-Zephyr-Charter-2023.08.21.pdf>

# Safety and Security Focus From the Start



## Exhibit B

### Zephyr Project Charter (the “Charter”)

The List  
Updated

#### 1. Mission of the Zephyr Project (“Ze

The mission of the Project is to:

- a. deliver the best-in-class RTOS to be secure and safe.
- b. maintain an auditable code base with community participation; this auditable code base shall be open to all members of the ecosystem; and
- c. include participation of leading controller manufacturers, hardware manufacturers, and other members of the ecosystem; and
- d. host the infrastructure for the open source project as a neutral home for community members, providing structure around the

#### 7. Safety Committee

a. Composition – the Safety Committee members shall consist of:

- i. one appointed voting representative from each Platinum Member, plus
- ii. non-voting Silver Member representatives who shall not count towards quorum.

b. Responsibilities – the Safety Committee shall be responsible for:

- i. the definition of the processes to ensure an auditable code base, as well as any associated certification artifacts (“Safety Artifacts”);
- ii. annually elect a Representative on the Safety Committee to serve as chair of the Safety Committee; and
- iii. annually elect a safety architect (the “Safety Architect”) , who may be different from the chair of the Safety Committee.

# So what does Zephyr mean by Safety?



**Safety** – the freedom from unacceptable risk of physical injury or of damage to the health of people, either directly, or indirectly as a result of damage to property or to the environment

# So what does Zephyr mean by Safety?



**Safety** – the freedom from unacceptable risk of physical injury or of damage to the health of people, either directly, or indirectly as a result of damage to property or to the environment

## **Zephyr will focus on "Functional Safety"**

- the part of safety that depends on a system or equipment operating correctly in response to its inputs
- Detecting potentially dangerous conditions, resulting either in the activation of a protective or corrective device or mechanisms to prevent hazardous events or in providing mitigation measures to reduce the consequences of the hazardous event.

# So what does Zephyr mean by Safety?



**Safety** – the freedom from unacceptable risk of physical injury or of damage to the health of people, either directly, or indirectly as a result of damage to property or to the environment

## Zephyr will focus on "Functional Safety"

- the part of safety that depends on a system or equipment operating correctly in response to its inputs
- Detecting potentially dangerous conditions, resulting either in the activation of a protective or corrective device or mechanisms to prevent hazardous events or in providing mitigation measures to reduce the consequences of the hazardous event.

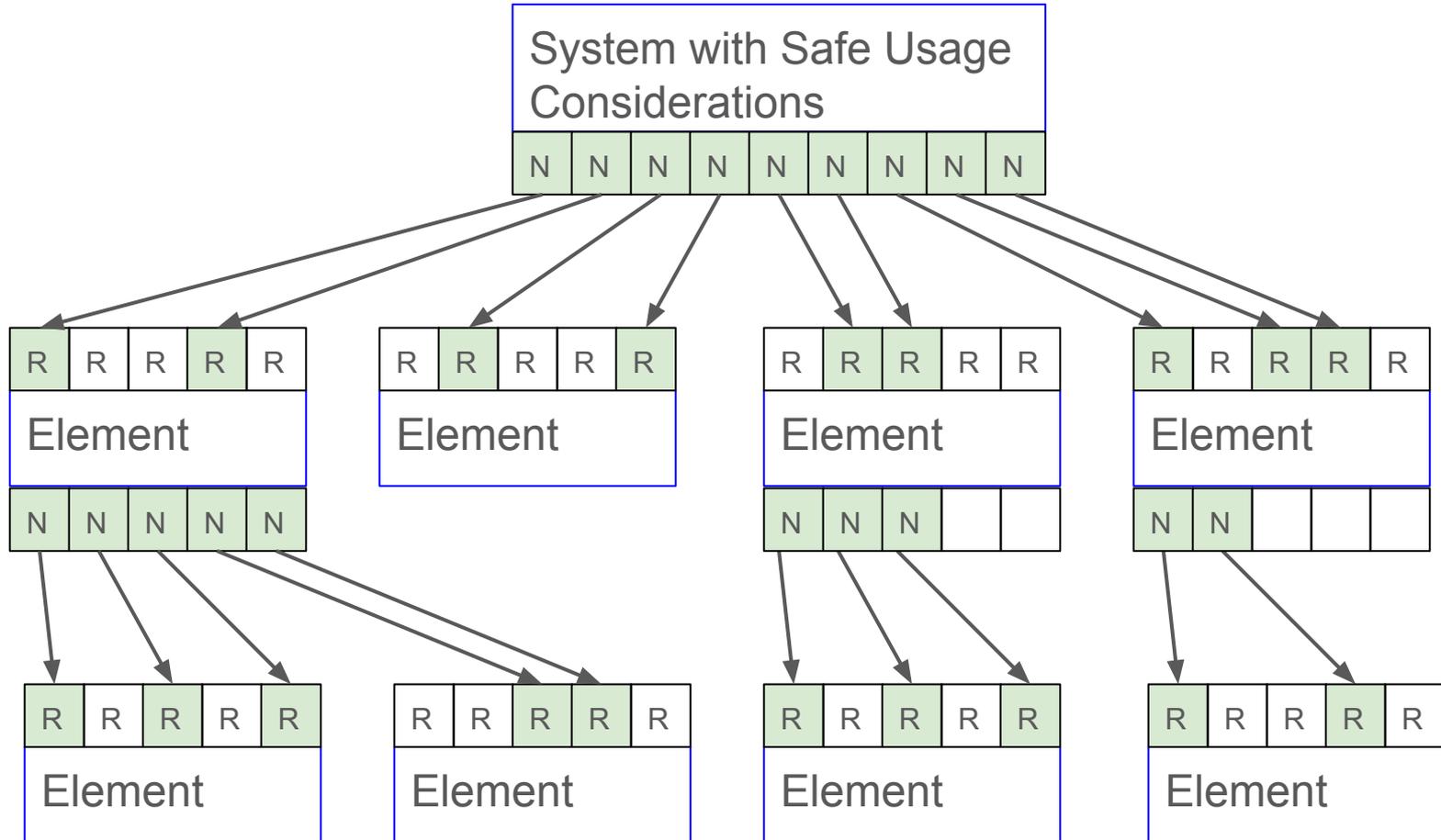
# "depends on a system" ?

**Systematic capability** is the general assumption, that

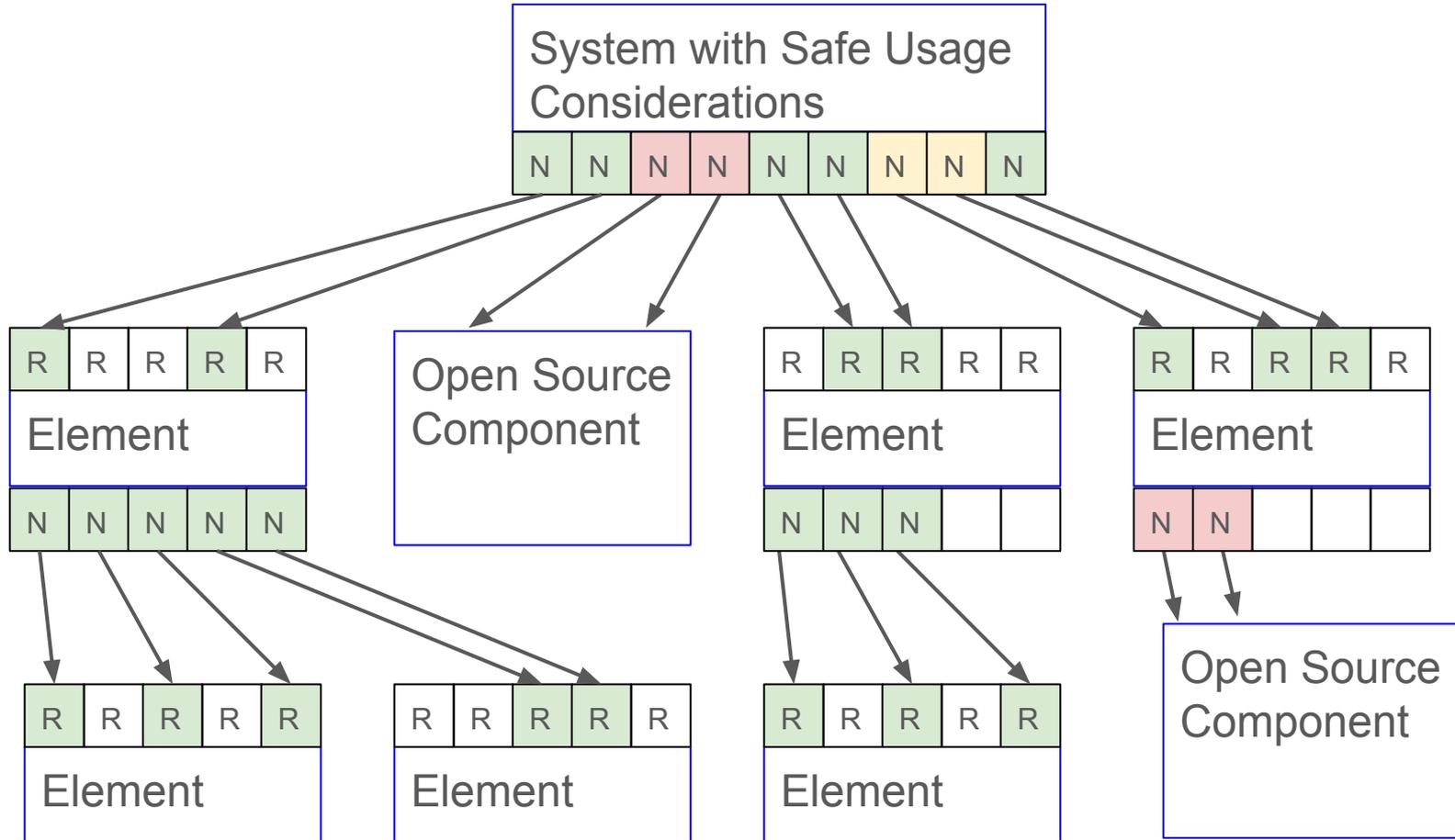
- if development, test and deployment of a system follow a specific set of tasks and
- there is evidence for adherence to these tasks
- (and under the assumption that the system architecture supports safety)

⇒ **Software is capable of performing as intended**

# Analyzing a System for Safe Usage

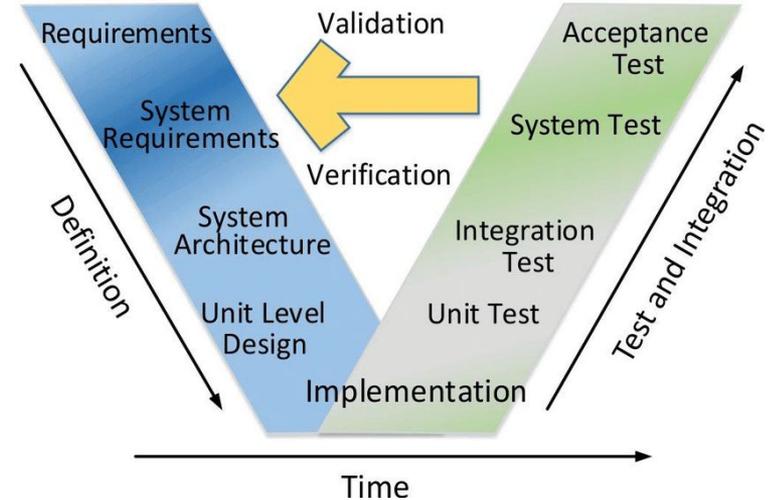


# Analyzing a System with Open Source?



# Safety Engineering 101: the "V-Model"

- The V-model for functional safety development originated from systems engineering practices to structure the process of designing and validating complex systems.
- It was later adapted and widely adopted in the automotive industry and other safety-critical sectors as a framework for ensuring the systematic integration of safety requirements and processes throughout each stage of the development lifecycle.
  - **ISO 26262** in the automotive industry
  - **IEC 61508** for industrial systems
  - **DO-178C** in aerospace

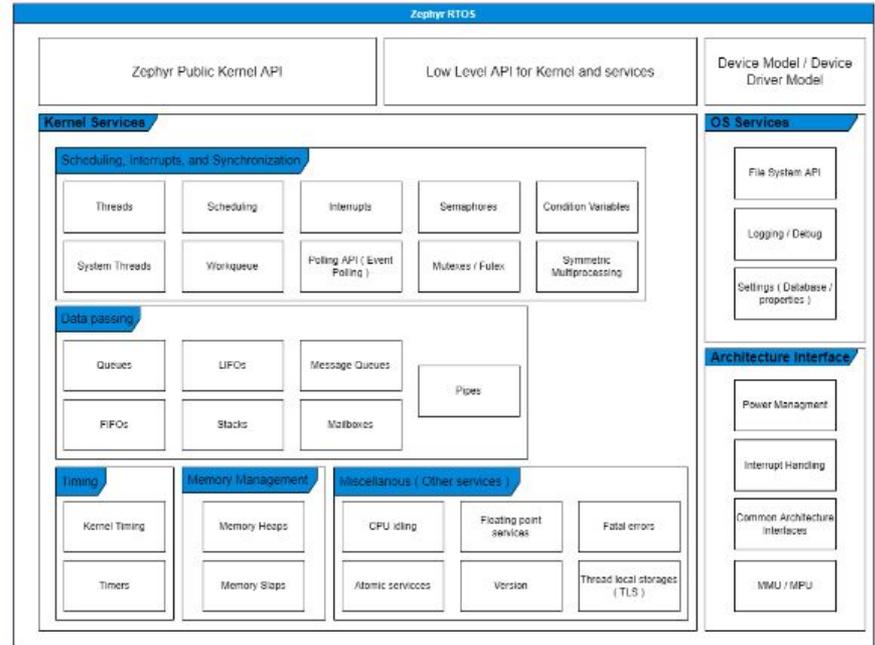


Source Image image provided under CC-4.0  
[https://www.researchgate.net/figure/The-functional-safety-development-via-V-model-14\\_fig4\\_362572593](https://www.researchgate.net/figure/The-functional-safety-development-via-V-model-14_fig4_362572593)

# Zephyr Initial Certification Focus



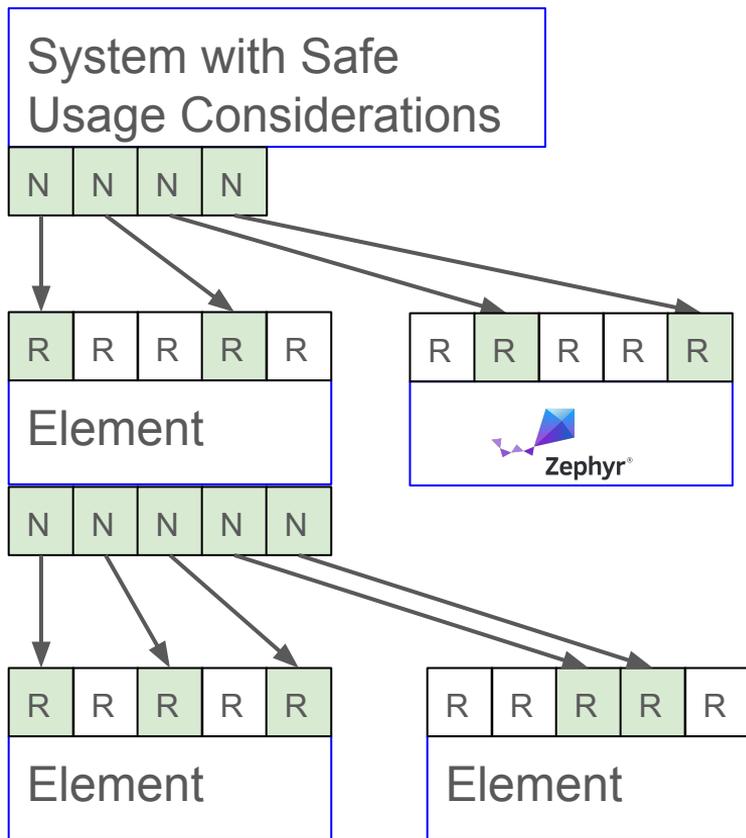
- Start with a limited scope of kernel and interfaces
- Initial target is IEC 61508 SIL 3 / SC 3 (IEC 61508-3, 7.4.2.12, Route 3s)
- Option for 26262 certification has been included in contract with certification authority should there be sufficient member interest



Starting scope

**Scope** can be **extended** to include **additional components** with associated **requirements** and **traceability** as determined by the safety committee

# Safety Element out of Context - SEooC



Development and verification independent of a specific context or application

Provides integration and operation information for safe system integration

Comes with sufficient evidence, that it can be integrated to a safety relevant system.

# Systemic Capability for Safety



## IEC 61508-3, Clause 7.4.2.12

“Where a pre-existing software element is reused to implement all or part of a safety function, the element shall meet both requirements a) and b) below for systematic safety integrity:

- a) Meet the requirements of one of the following compliance routes:
  - Route 1s: compliant development. Compliance with the requirements of this standard for the avoidance and control of systematic faults in software;
  - Route 2s: proven in use. Provide evidence that the element is proven in use. See 7.4.10 of IEC 61508-2;
  - Route 3s: assessment of non-compliant development. Compliance with 7.4.2.13

# Systemic Capability for Safety



## IEC 61508-3, Clause 7.4.2.12

“Where a pre-existing software element is reused to implement all or part of a safety function, the element shall meet both requirements a) and b) below for systematic safety integrity:

- a) Meet the requirements of one of the following compliance routes:
  - Route 1s: compliant development. Compliance with the requirements of this standard for the avoidance and control of systematic faults in software;
  - Route 2s: proven in use. Provide evidence that the element is proven in use. See 7.4.10 of IEC 61508-2;
  - Route 3s: assessment of non-compliant development. Compliance with 7.4.2.13



# Systemic Capability for Safety



## IEC 61508-3, Clause 7.4.2.13

To comply with Route 3s a pre-existing software element shall meet all of the following requirements a) to i)

- a) The software safety requirement specification for the element [...] shall be documented to the same degree of precision as would be required by this standard for any safety related element of the same systematic capability [...]



⇒ Creation of System and Software Specification

# Systemic Capability for Safety

## IEC 61508-3, Clause 7.4.2.13

To comply with Route 3s a pre-existing software element shall meet all of the following requirements a) to i)

- b) The justification for use of a software element shall provide evidence that the desired safety properties specified [...] have been considered [...].



⇒ Definition of the Safety Claims

⇒ Traceability to requirements, code and tests

# Systemic Capability for Safety



## IEC 61508-3, Clause 7.4.2.13

To comply with Route 3s a pre-existing software element shall meet all of the following requirements a) to i)

- c) The element's design shall be documented to a degree of precision, sufficient to provide evidence of compliance with the requirement specification and the required systematic capability. [...]The justification for use of a software element shall provide evidence that the desired safety properties specified [...] have been considered [...].



⇒ Reuse of the information provided by the [docs.zephyrproject.org](https://docs.zephyrproject.org)

# Systemic Capability for Safety

## IEC 61508-3, Clause 7.4.2.13

To comply with Route 3s a pre-existing software element shall meet all of the following requirements a) to i)

- d) The evidence required in 7.4.2.13 a) and b) shall cover the software's integration with the hardware. [...]



- ⇒ use the existing tests
- ⇒ establishing traceability
- ⇒ enhancing coverage as needed

# Systemic Capability for Safety



## IEC 61508-3, Clause 7.4.2.13

To comply with Route 3s a pre-existing software element shall meet all of the following requirements a) to i)

- e) There shall be evidence that the element has been subject to verification and validation using a systematic approach with documented testing and review of all parts of the element's design and code [...]



- ⇒ use the existing tests
- ⇒ establishing traceability
- ⇒ enhancing coverage as needed

# Systemic Capability for Safety



## IEC 61508-3, Clause 7.4.2.13

To comply with Route 3s a pre-existing software element shall meet all of the following requirements a) to i)

- f) Where the software element provides functions which are not required in the safety related system, then evidence shall be provided that the unwanted functions will not prevent the E/EE/EP system from meeting its safety requirements.



⇒ Providing a defined safety scope definition

# Systemic Capability for Safety



## IEC 61508-3, Clause 7.4.2.13

To comply with Route 3s a pre-existing software element shall meet all of the following requirements a) to i)

- g) There shall be evidence that all credible failure mechanisms of the software element have been identified and that appropriate mitigation measures have been implemented.



⇒ creating requirements

⇒ establishing traceability to code & tests

# Systemic Capability for Safety

## IEC 61508-3, Clause 7.4.2.13

To comply with Route 3s a pre-existing software element shall meet all of the following requirements a) to i)

- h) The planning for use of the element shall identify the configuration of the software element, the software and hardware run-time environment and if necessary the configuration of the compilation / linking system.



⇒ information that will be discussed in the Safety Manual

# Systemic Capability for Safety



## IEC 61508-3, Clause 7.4.2.13

To comply with Route 3s a pre-existing software element shall meet all of the following requirements a) to i)

- i) The justification for use of the element shall be valid only those applications which respect the assumptions made in the [...] safety manual [...]



⇒ creation of the Safety Manual

# Systemic Capability for Safety



## IEC 61508-3, Clause 7.4.2.12

“Where a pre-existing software element is reused to implement all or part of a safety function, the element shall meet both requirements a) and b) below for systematic safety integrity:

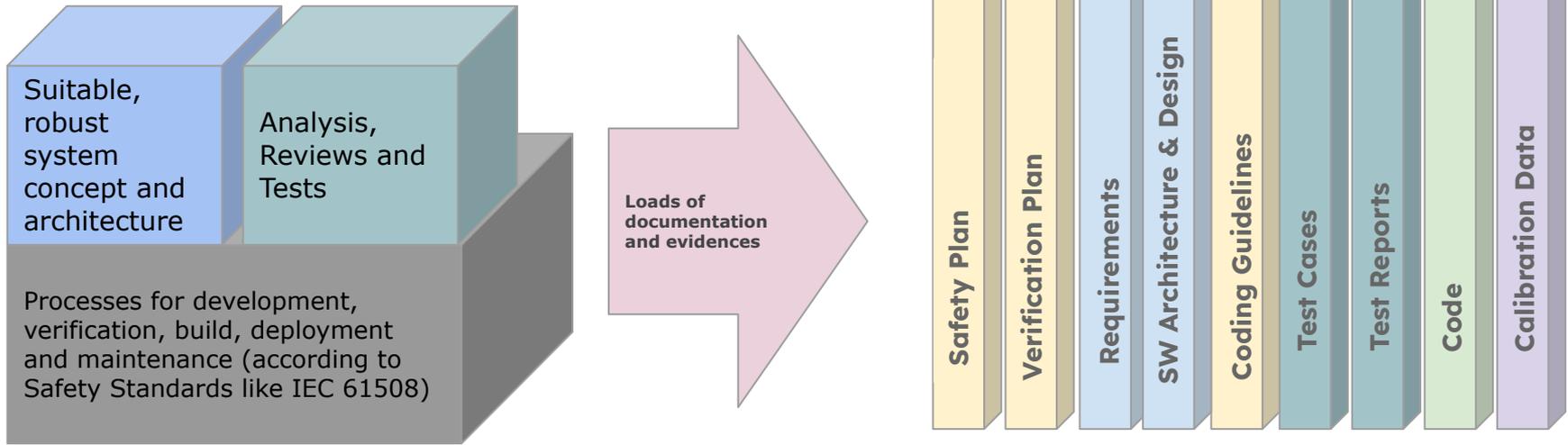
b) provides a safety manual [...]

⇒ creation of the Safety Manual



# What is Functional Safety aiming for ?

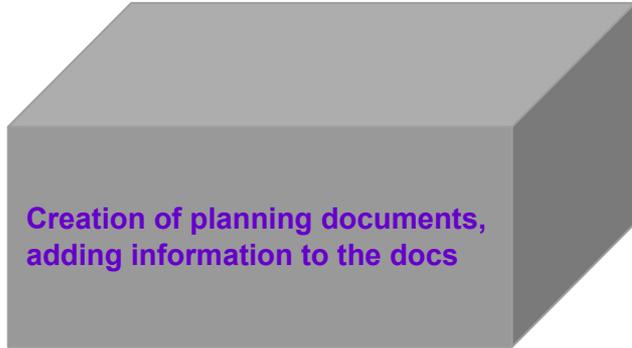
## Safety Architecture and Documentation



# How is the Zephyr Project addressing this?



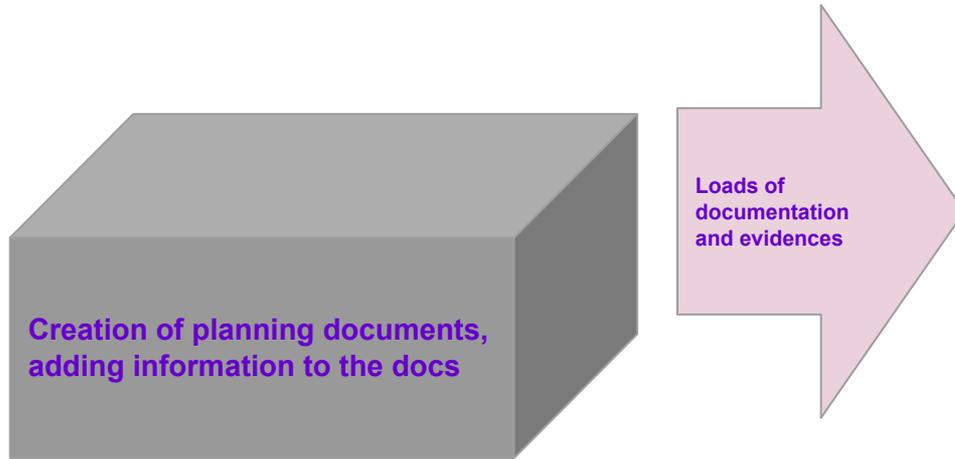
Following Route 3s by creating the missing work products



# How is the Zephyr Project addressing this?

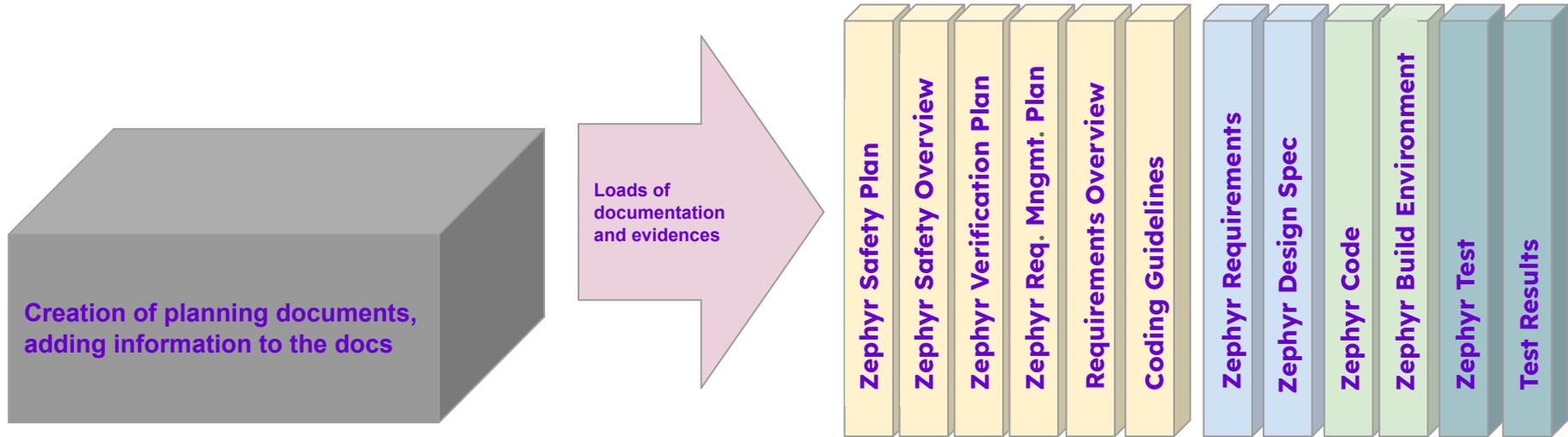


Following Route 3s by creating the missing work products



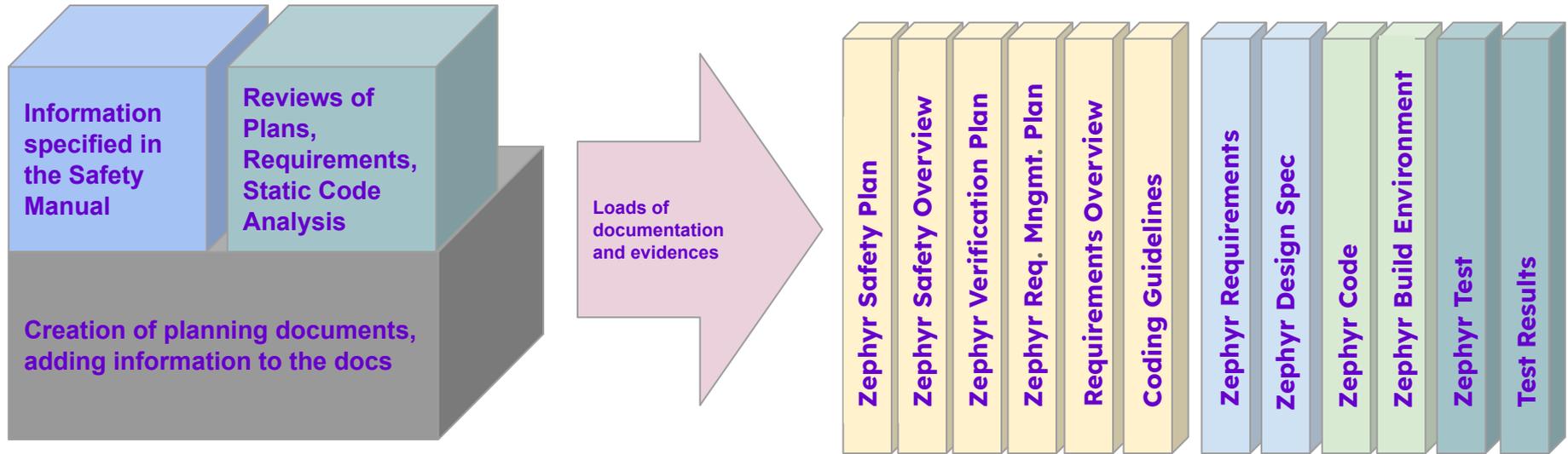
# How is the Zephyr Project addressing this?

Following Route 3s by creating the missing work products



# How is the Zephyr Project addressing this?

Following Route 3s by creating the missing work products



# Creating the Work Products



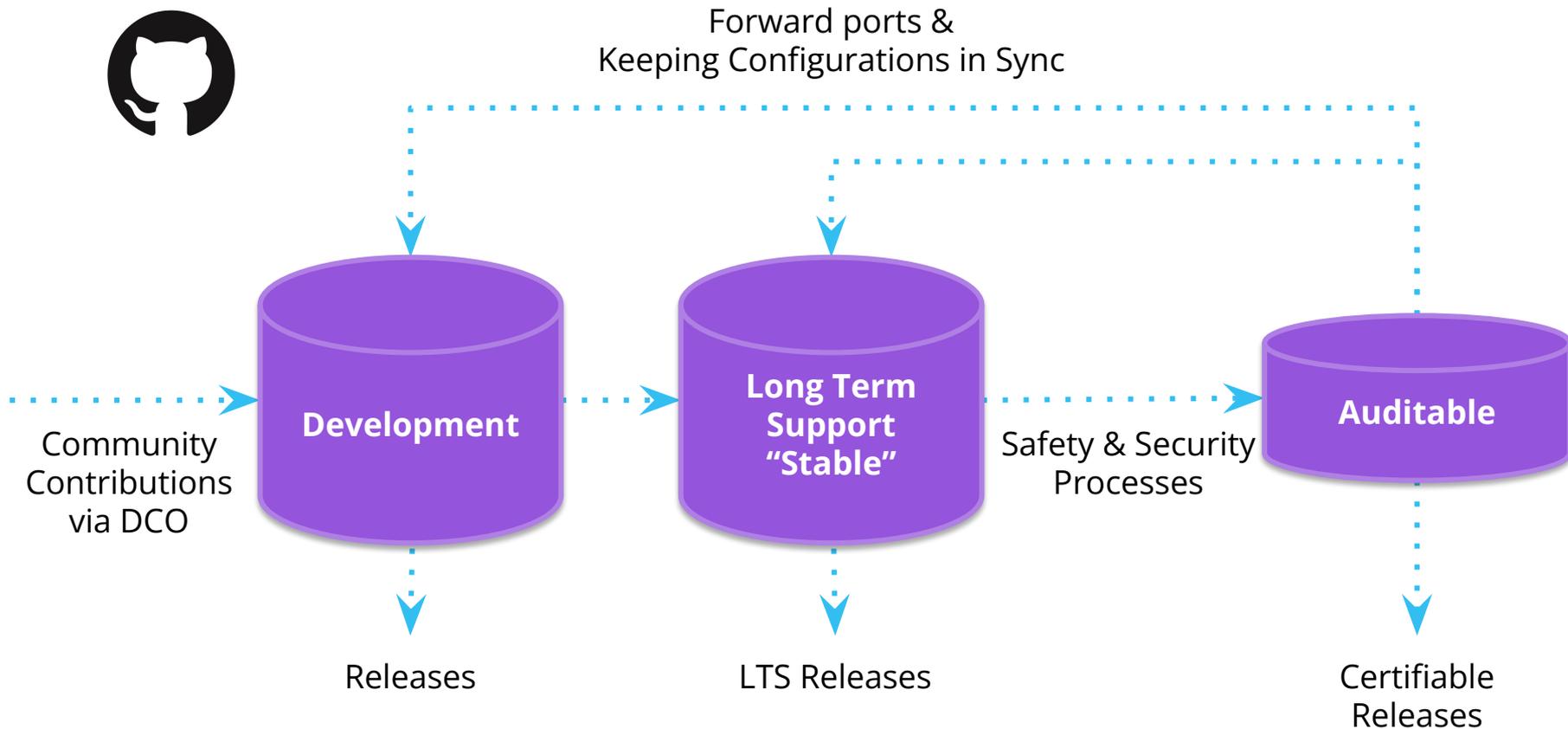
## Safety Committee Role

- Safety Certification strategy decisions
  - Scope of certification
  - Certification standards
  - Certification timeline
- Assessment and audit specific tasks
- Owner of certification artefacts
- **Participation limited** to the project's platinum members, the safety architect, the safety chair, functional safety manager and project staff

## Safety Working Group Role

- Creating/deriving and documenting the requirements for project code
- Establishing traceability between requirements, code and relevant tests
- Extending testing coverage as needed
- Setting up requirements management tooling
- Working on the creation of the required documentation and evidence
- [Open to everyone to participate](#)

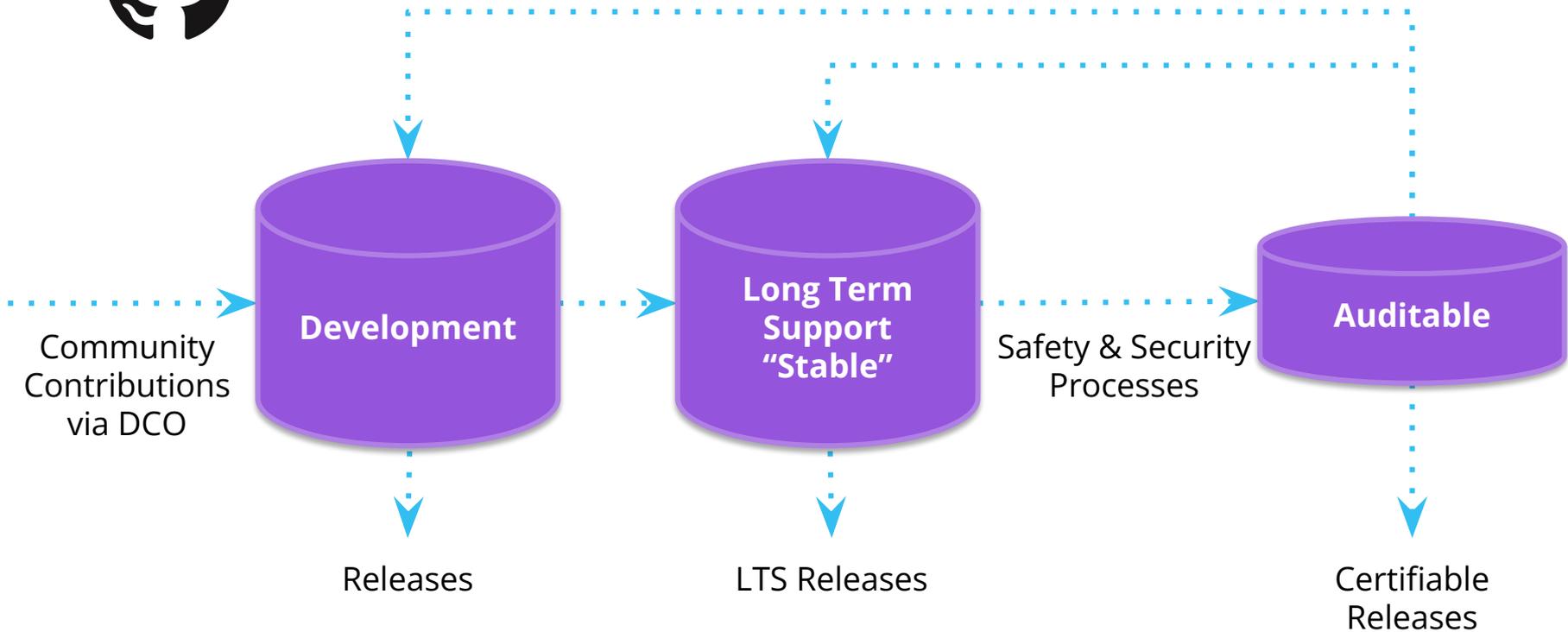
# Code Repositories



# Traceability on "Development"



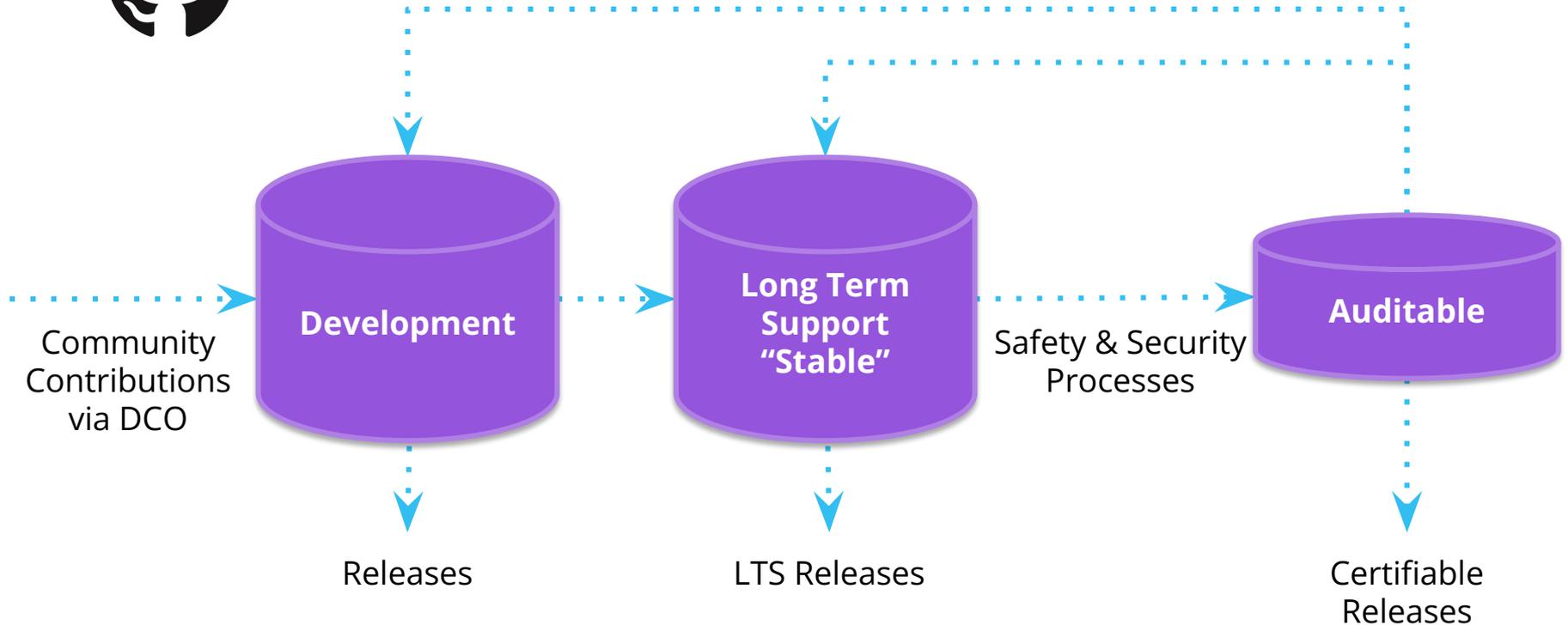
Forward ports &  
Keeping Configurations in Sync



# "Auditable" is subset of "Development"



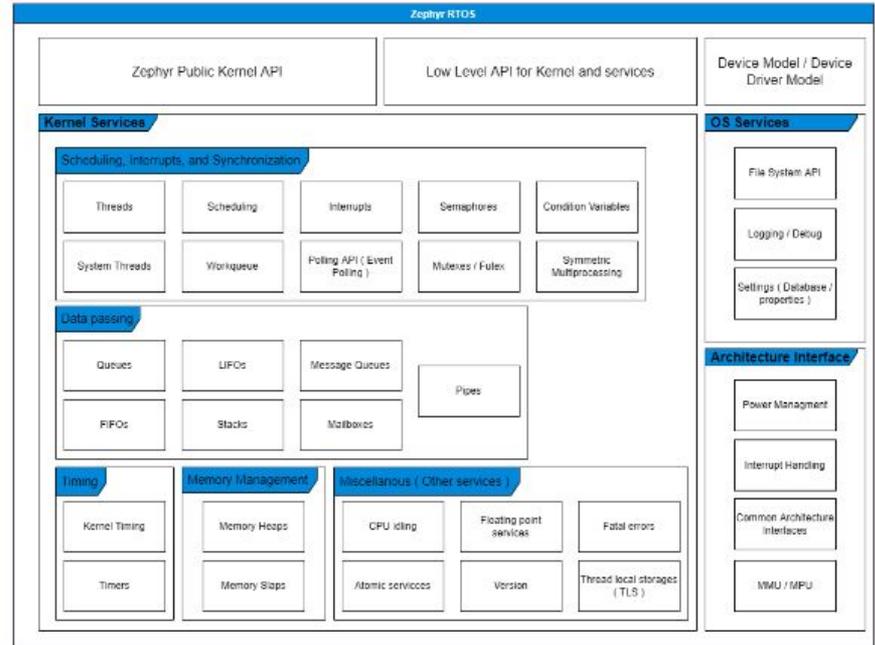
Forward ports &  
Keeping Configurations in Sync



# Zephyr Initial Certification Focus



- Start with a limited scope of kernel and interfaces
- Initial target is IEC 61508 SIL 3 / SC 3 (IEC 61508-3, 7.4.2.12, Route 3s)
- Option for 26262 certification has been included in contract with certification authority should there be sufficient member interest



Starting scope

**Scope** can be **extended** to include **additional components** with associated **requirements** and **traceability** as determined by the safety committee

# Work Product Strategy

Strategy for artefact creation:

- use developer friendly tooling
- use known workflows on GitHub
- reuse as much as we can from the [docs.zephyrproject.org](https://docs.zephyrproject.org)

## Enhancements to project documentation

- Sphinx in [docs.zephyrproject.org](https://docs.zephyrproject.org)

## New artefacts ⇒ [StrictDoc](#)

- Requirements
- Safety Plan
- Safety Manual
- ...

## Evidence ⇒ [StrictDoc](#)

- Assessments
- Checklists
- ...

# Compliance with Coding Standards



Project already has defined [Coding Guidelines](#) derived from MISRA rules and CERT-C references

Identification of Coding Guideline violations and adaption of the code

- Initially done by Bugseng on a separate branch
- Work merged to the main branch of "Development" in 2024

**Coming soon:** Static Analysis in the CI to check for adherence on each patch

## Coding Guidelines

### Main rules

The coding guideline rules are based on MISRA-C 2012 and are a **subset** of MISRA-C. The subset is listed in the table below with a summary of the rules, its severity and the equivalent rules from other standards for reference.

#### Note

For existing Zephyr maintainers and collaborators, if you are unable to obtain a copy through your employer, a limited number of copies will be made available through the project. If you need a copy of MISRA-C 2012, please send email to [safety@lists.zephyrproject.org](mailto:safety@lists.zephyrproject.org) and provide details on reason why you can't obtain one through other options and expected contributions once you have one. The safety committee will review all requests.

Main rules

Zephyr rule	Description	MISRA-C 2012 rule	MISRA-C severity	CERT C reference
1	Any implementation-defined behaviour on which the output of the program depends shall be documented and understood	<a href="#">Dir 1.1</a>	Required	<a href="#">MSC09-C</a>
2	All source files shall compile without any compilation errors	<a href="#">Dir 2.1</a>	Required	N/A
3	All code shall be traceable to documented requirements	<a href="#">Dir 3.1</a>	Required	N/A
4	Run-time failures shall be minimized	<a href="#">Dir 4.1</a>	Required	N/A
5	All usage of assembly language should be documented	<a href="#">Dir 4.2</a>	Advisory	N/A
6	Sections of code should not be "commented out"	<a href="#">Dir 4.4</a>	Advisory	<a href="#">MSC04-C</a>
7	Identifiers in the same name space with overlapping visibility should be typographically unambiguous	<a href="#">Dir 4.5</a>	Advisory	<a href="#">DCL02-C</a>
8	typedefs that indicate size and signedness should be used in place of the basic numerical types	<a href="#">Dir 4.6</a>	Advisory	N/A

# Critical Path: Requirement & Traceability



- Established hierarchical structure of requirements
- Capturing the requirements in StrictDoc which is working towards import/export of SPDX
- Capturing plans in StrictDoc where each planning item (like Safety Plan) is tracked as a requirement
- Capturing Assessment checklist in StrictDoc where each checkpoint is a requirement, tracing to the Zephyr's evidences

A screenshot of a web-based code editor interface. The top navigation bar shows the repository name "zephyrproject-rtos / reqmgmt" and a search bar. Below the navigation, there are tabs for "Code", "Issues", "Pull requests", "Actions", "Projects", "Wiki", "Security", and "Insights". The left sidebar displays a file tree with the path "main > docs > software\_requirements > condition\_variables.sdoc" selected. The main editor area shows the content of "condition\_variables.sdoc" with a line-by-line view. The code is a StrictDoc document for condition variables, including sections for [DOCUMENT], [GRAMMAR], [REQUIREMENT], and [RELATIONS]. The [REQUIREMENT] section contains two entries: one for dynamic initialization (UID: ZEP-SRS-21-1) and one for static initialization (UID: ZEP-SRS-21-2). The [RELATIONS] section lists parent requirements: ZEP-SYRS-20.

```
1 [DOCUMENT]
2 TITLE: Condition Variables
3 REQ_PREFIX: ZEP-SRS-21-
4
5 [GRAMMAR]
6 IMPORT_FROM_FILE: software_requirements.sgra
7
8 [REQUIREMENT]
9 UID: ZEP-SRS-21-1
10 STATUS: Draft
11 TYPE: Functional
12 COMPONENT: Condition Variables
13 TITLE: Dynamic initialization of condition variables
14 STATEMENT: >>>
15 The Zephyr RTOS shall provide a mechanism to define and initialize a condition variable dynamically (at runtime).
16 <<<
17 RELATIONS:
18 - TYPE: Parent
19   VALUE: ZEP-SYRS-20
20
21 [REQUIREMENT]
22 UID: ZEP-SRS-21-2
23 STATUS: Draft
24 TYPE: Functional
25 COMPONENT: Condition Variables
26 TITLE: Static initialization of condition variables
27 STATEMENT: >>>
28 The Zephyr RTOS shall provide a mechanism to define and initialize a condition variable statically (at compile time).
29 <<<
30 RELATIONS:
31 - TYPE: Parent
32   VALUE: ZEP-SYRS-20
```

# Safety Assessment & Certification Plan



## Phase 1 - Concept Phase

Approval of the overall plans and strategy, scope and high level requirements & architecture

## Phase 2 - Detailed Phase

Filling the gaps:

- Completeness of requirements
- Safety Analysis
- Traceability
- Verification & Test Coverage

# Status Today



- Coding Guidelines established based on MISRA rules and applied
- Static Analysis tooling to check for adherence to Guidelines selected for future contributions
- Reference Requirements and Traceability started in the code base
- Automatic human readable documentation of requirements from StrictDoc rules is available

A screenshot of a web-based documentation interface for Zephyr software requirements. The breadcrumb trail at the top reads "Zephyr Project Requirements / Zephyr Software Requirements / Document". The left sidebar shows a tree view of the requirements structure, with "Zephyr Software Requirements" expanded to show sub-sections like "Hardware Architecture Interface", "C library", "Device Driver API", "Exception and Error Handling", "System Initialization", "File system", "Interrupts", "Logging", "Memory protection", "Memory Objects", and "Data Passing". The main content area displays a specific requirement titled "1.2. Thread Context Switching".

**1.2. Thread Context Switching**

UID:  
ZEP-SRS-19-2

STATUS:  
Draft

TYPE:  
Functional

COMPONENT:  
Hardware Architecture Interface

PARENTS:  
← ZEP-SYRS-1 Architecture Layer Interface

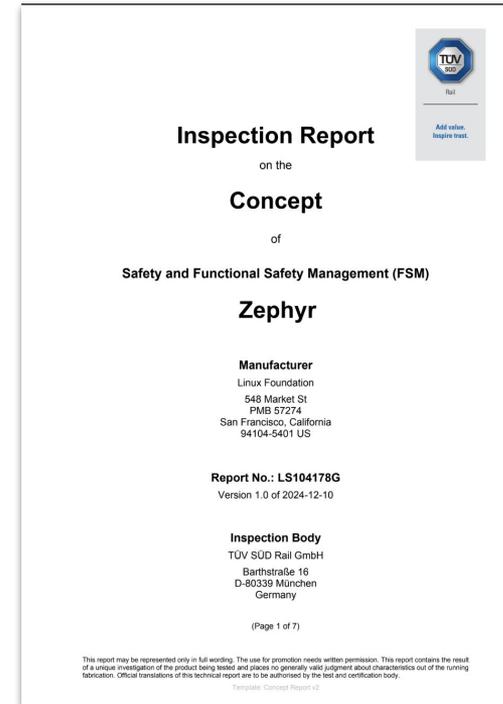
STATEMENT:  
The Zephyr RTOS shall provide a mechanism for context switching between threads.

USER\_STORY:  
As a Zephyr RTOS user I want to execute code concurrently in one or more threads and when interrupted at a code location in a thread, to continue at the very same location.

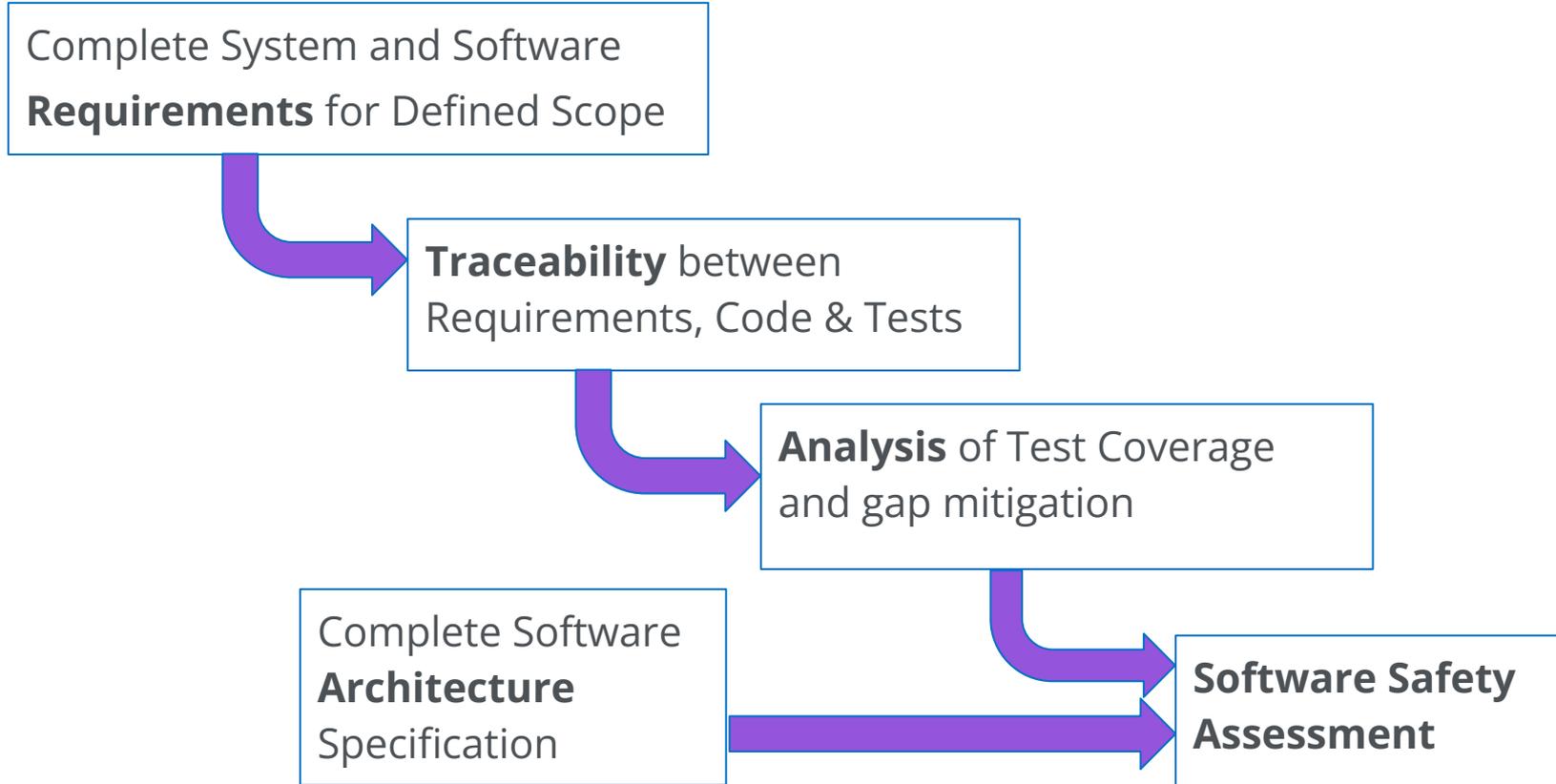
# Status Today



- Coding Guidelines established based on MISRA rules and applied
- Static Analysis tooling to check for adherence to Guidelines selected for future contributions
- Reference Requirements and Traceability started in the code base
- Automatic human readable documentation of requirements from StrictDoc rules is available
- **Formal concept approval and Phase 1 is complete ... on to Phase 2**



# Critical Path to Phase 2



# Want to help make it happen faster?

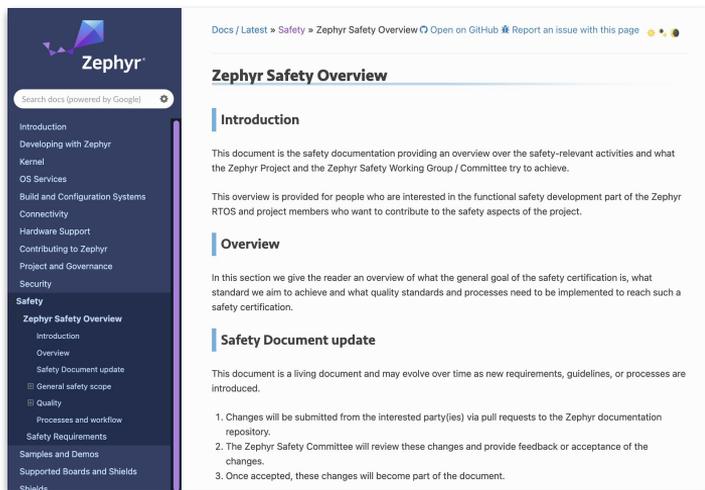


Read the docs to start ...

## Safety Overview



## Requirements Guideline



Docs / Latest » Safety » Zephyr Safety Overview [Open on GitHub](#) [Report an issue with this page](#)

## Zephyr Safety Overview

### Introduction

This document is the safety documentation providing an overview over the safety-relevant activities and what the Zephyr Project and the Zephyr Safety Working Group / Committee try to achieve.

This overview is provided for people who are interested in the functional safety development part of the Zephyr RTOS and project members who want to contribute to the safety aspects of the project.

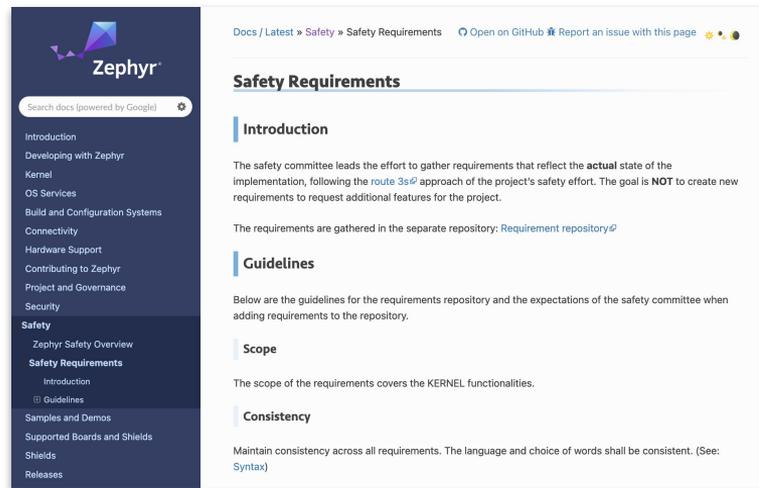
### Overview

In this section we give the reader an overview of what the general goal of the safety certification is, what standard we aim to achieve and what quality standards and processes need to be implemented to reach such a safety certification.

### Safety Document update

This document is a living document and may evolve over time as new requirements, guidelines, or processes are introduced.

1. Changes will be submitted from the interested party(ies) via pull requests to the Zephyr documentation repository.
2. The Zephyr Safety Committee will review these changes and provide feedback or acceptance of the changes.
3. Once accepted, these changes will become part of the document.



Docs / Latest » Safety » Safety Requirements [Open on GitHub](#) [Report an issue with this page](#)

## Safety Requirements

### Introduction

The safety committee leads the effort to gather requirements that reflect the **actual** state of the implementation, following the [route 3s](#) approach of the project's safety effort. The goal is **NOT** to create new requirements to request additional features for the project.

The requirements are gathered in the separate repository: [Requirement repository](#)

### Guidelines

Below are the guidelines for the requirements repository and the expectations of the safety committee when adding requirements to the repository.

#### Scope

The scope of the requirements covers the KERNEL functionalities.

#### Consistency

Maintain consistency across all requirements. The language and choice of words shall be consistent. (See: [Syntax](#))

# Want to help make it happen faster?

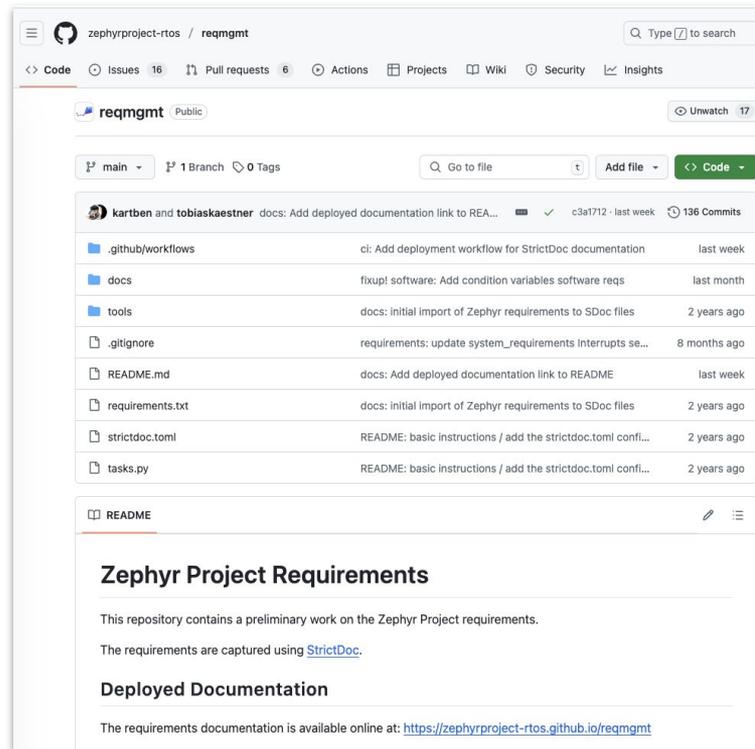


Contribute in our repos:

## Requirements Repository

- Grab a PR and give some feedback
- Read through the existing requirements and submit an issue if needed
- Get familiar with [StrictDoc](#) and start creating new requirements :-)

<https://github.com/zephyrproject-rtos/reqmgmt>



# Join us!



## Participate in the Working Group:

### Safety Working Group Project

- Have a look at the tasks
- Grab an existing task
- Or submit a new task



<https://lists.zephyrproject.org/g/safety-wg>

- Biweekly meeting calendar information and mailing list subscription.



<https://discord.gg/mgZkSmq2>

A screenshot of a GitHub project board titled "Safety WG". The board is organized into three columns: "No Status" (2 items), "Todo" (8 items), and "In Progress" (5 items). The "No Status" column contains a task "Governing Board 2025 Dependability Goals" with an "RFC" label and a task "Mailbox requirements". The "Todo" column lists tasks such as "Establish a way to trace existing testcases back to design specification" (with "Enhancement" and "Safety" labels), "Improve coding guidelines descriptions" (with "Coding Guidelines", "Enhancement", and "Safety" labels), and "Extend documentation with diagrams". The "In Progress" column shows tasks like "Add Requirements for the Zephyr project" (with "Meta" and "Safety" labels), "Coding guidelines fixes migration from auditable branch back to main (forward port)" (with "Coding Guidelines" and "MISRA-C" labels), and "Define a Coding Guideline enforcement strategy" (with "Coding Guidelines" and "Enhancement" labels). The board also includes a search bar, a "Board" dropdown, and a "Requirements" tab.

Source: <https://github.com/orgs/zephyrproject-rtos/projects/23/views/1>