The basics of Functional Safety in the automotive domain

Fabrizio Tronci

Functional Safety Manager

2022-02

HUAWEI

Who is this guy?

Current Role:

Functional Safety Manager in Huawei Pisa RC

Work Experience:

7+ years of experience in automotive and Functional Safety field

Head of Safety Competence Center @Marelli (Venaria site)

Functional Safety Lead Engineer @AVL Italia (Cavriago site)

Contact Info:

Email: fabrizio.tronci@huawei.com

Job location: Via Pietro Nenni, 24 - Pisa

WHAT DOES SAFETY MEAN?

Brainstorming

What is Functional Safety in your opinion?

Let’s guess the main key words

Functional Safety Definition

Risk = P(damage) x Severity

Safety is freedom from unacceptable risk

- Of physical injury
- Of damage to the health of people
- To the environment

Functional Safety

Absence of risk, judged to be unacceptable in a certain context due to hazards caused by failures of E/E systems or unintended behavior of an item with respect to its design intent – resulting from:

- Specification, implementation or realization errors
- Failure during operation
- Reasonably foreseeable operational errors
- Reasonably foreseeable misuse

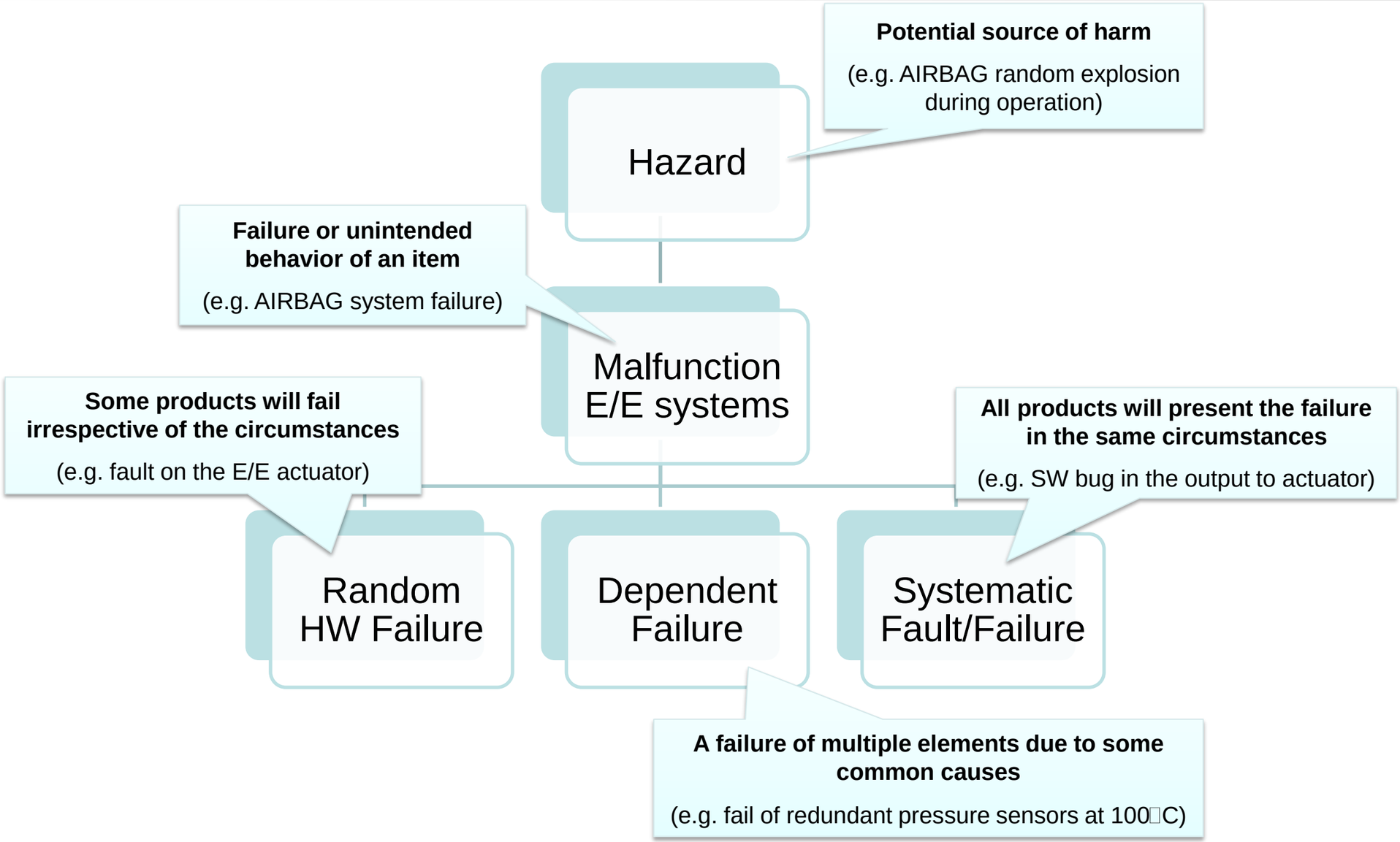
Functional Safety: Objective

Functional Safety Objective

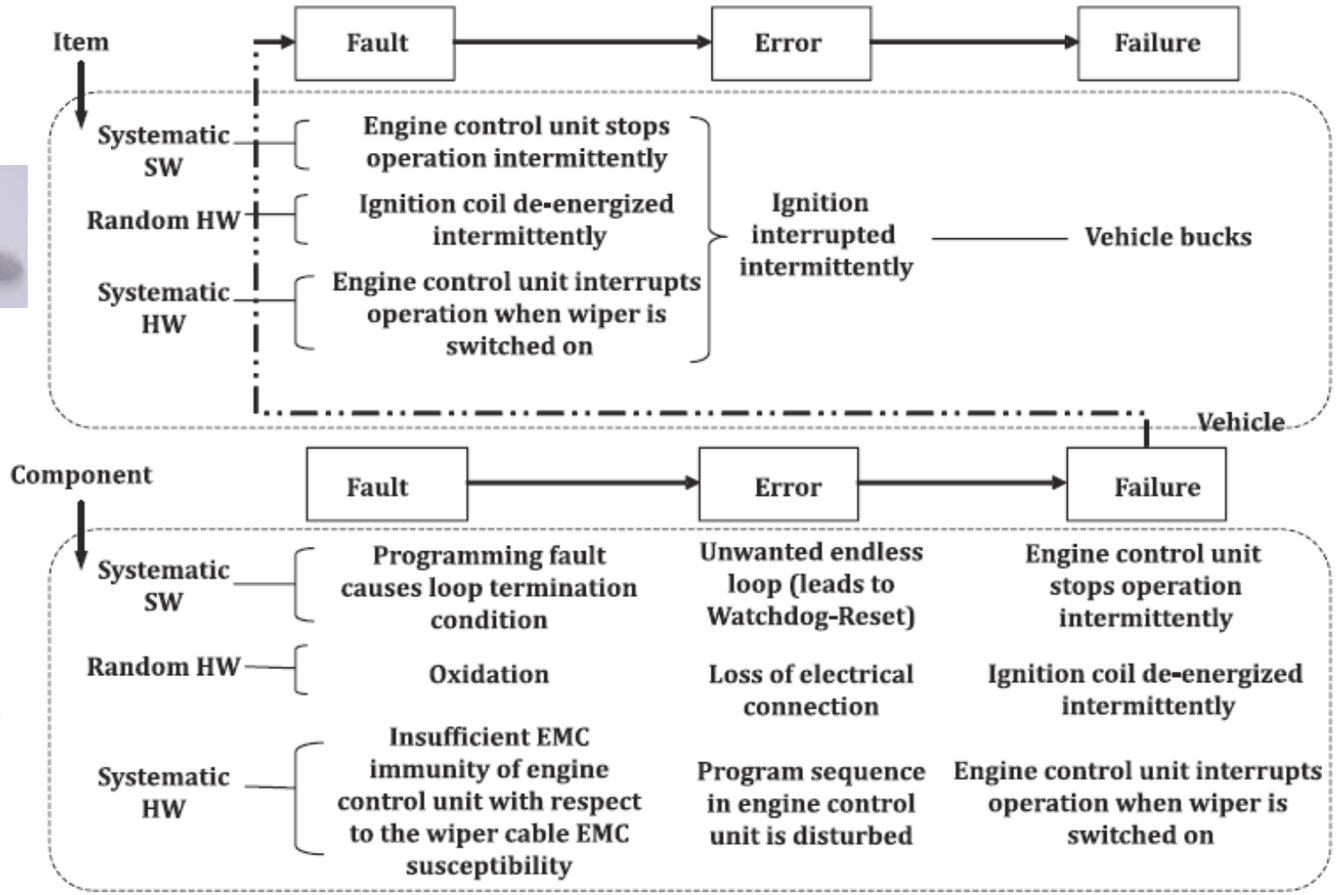
to reduce as low as possible the residual risk introduced by the malfunction of safety relevant functionalities.

Keep the malfunction under control to preserve the safeness of the people in the surrounding environment.

Failures connection



Faults leading to failures – From system to item



ISO 26262-10 : Example item dissolution

The terms fault, error and failure are defined in ISO 26262-1:2018. The figure depicts the progression of faults to errors to failures from three different types of causes:

- systematic software issues,
- random hardware issues
- and systematic hardware issues

At the component level, each different type of fault can lead to different failures. However, failures at the component level are faults at the item level.

Note that in this example, at the vehicle level, faults from different causes can lead to the same failure.

Countermeasures to Faults

Avoidance of Faults

- Avoid systematic faults/failures during design, development and manufacturing

Control of faults/failures during operation

- Control of systematic faults/failures
- Control of random HW failures

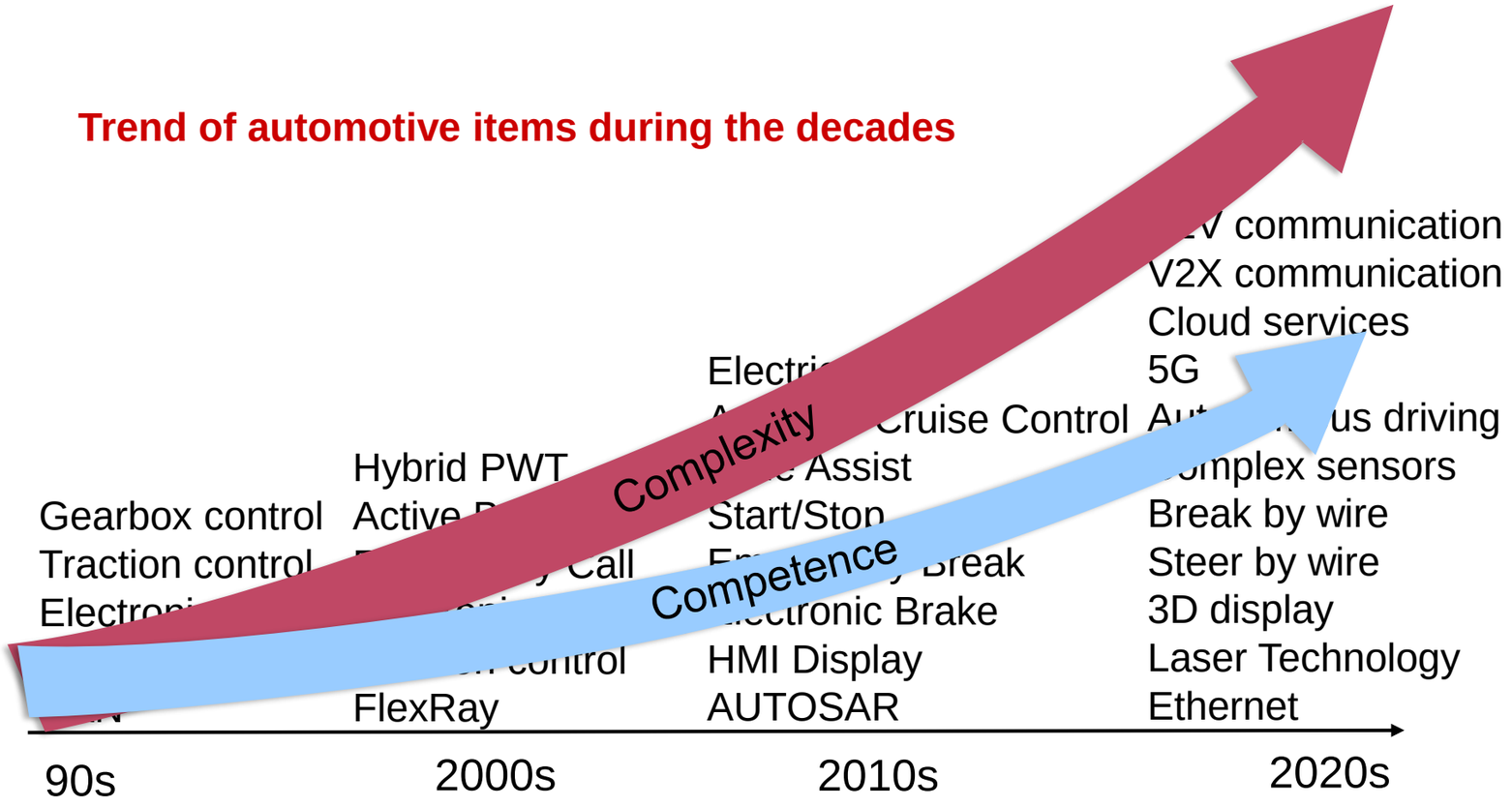
Before Delivery

In Operation

WHAT IS ISO 26262?

The current trend of automotive functionalities

Trend of automotive items during the decades



- Increasing number and complexity of items
- More and more distributed developments
- Growth of safety, security and network requirements

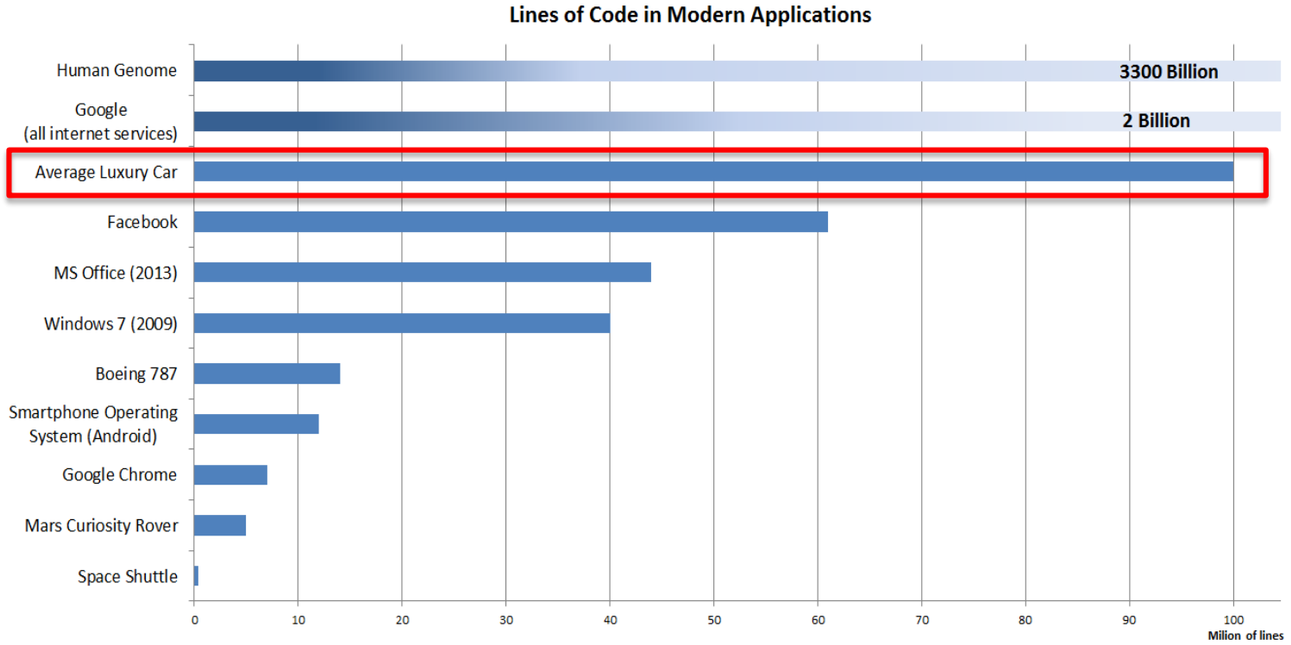
- Quantity: Boost of the number of systems
- Maturity: Young processes and tools
- Quality: Lack of experts of new technologies in automotive field

Why Functional Safety

Trend of the automotive field:

- introduction of new **complex functionalities** in the area of *autonomous driving, in vehicle dynamics control, active & passive safety systems and electrical propulsion.*

- Creation of a network of so-called **Integrated Safety Systems**.



Lines of Code in Modern Applications. Extract from: "Information is beautiful" by David McCandless
http://www.informationisbeautiful.net/visualizations/million-lines-of-code/

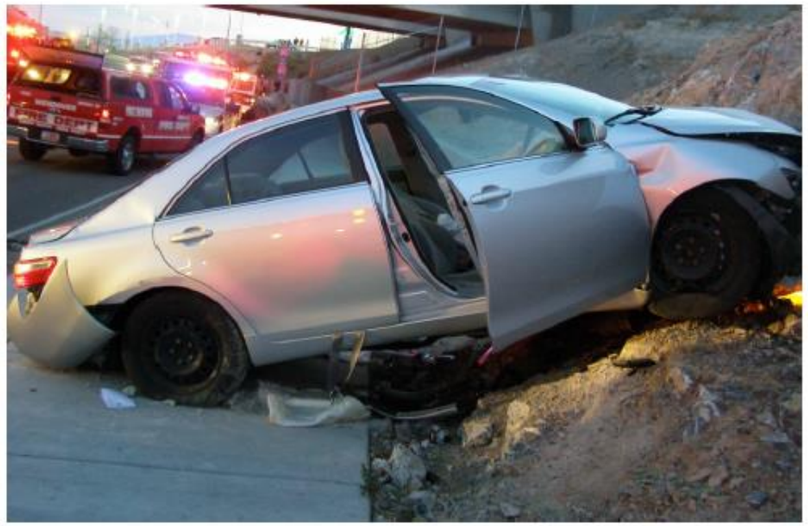
CONSEQUENCES

New functionalities become **potentially *safety critical***

In case of relevant failure, potential **liability impact for OEM**

Functional Safety in the Automotive industry

Oklahoma court ruled against Japanese OEM in a case of unintended acceleration that led to the death of one of the occupants.



HOME SECTIONS SEARCH NEW YORK POST

NEWS

Toyota to pay $1.2B settlement in vehicle acceleration laws uit

By Bob Fredericks and Post Wires March 19, 2014 | 9:19am

A Toyota Camry that crashed as it exited Interstate 80 in Wendover, Utah, in 2010.

Photo: AP

And many other cases…

Brake lights either don't light up or light up continuously

Gear is unintentionally switch to neutral with automatic gear control

Airbags and seat belt pre-tensioner are not or too late activated

ISO 26262 and ISO 21448: Functional Safety Overview

Voluntary regulation, not imposed by the law

ISO 26262

Considered the international state of the art for the Functional Safety

ISO 26262 covers functional safety in the event of system failures. It doesn't cover safety hazards that result without system failure. That's why ISO 21448 (SOTIF) is necessary (but not yet formalized)

What have we learned so far?

- Functional Safety Definition
- Fault vs Failure relationship
- Functional Safety Objectives
- ISO 26262 importance and history

Don’t hesitate to ask questions

ISO 26262 OVERVIEW

ISO 26262 application

ISO 26262 is applied to

- **E/E safety-related systems installed in series production road vehicles, excluding mopeds (in 2018 revision included bus, trucks, trailer, semi trailer and motorcycles)**
- **possible hazards caused by malfunctioning behavior of E/E safety-related systems,** including interaction of these systems
- possible hazards related by malfunctioning behavior of E/E safety-related systems, related to **electric shock, fire, smoke, heat, radiation, toxicity, flammability, reactivity, corrosion, release of energy and similar hazards, unless directly caused by malfunctioning behavior of safety-related E/E systems**
- E/E systems in special vehicles such as E/E systems designed for drivers with disabilities

ISO 26262, major changes in 2018

New Topics in 2nd release

Trucks and Buses

Cybersecurity

Motorcycles

Semiconductors

1. Vocabulary

2. Management of functional safety

- **2-5** Overall safety management
- **2-6** Safety management during the concept phase and the product development phases
- **2-7** Safety management during production, operation, service and decommissioning

3. Concept phase

- **3-5** Item definition
- **3-6** Hazard analysis and risk assessment
- **3-7** Functional safety concept

4. Product development at the system level

- **4-5** General topics for the product development at the system level
- **4-6** Technical safety concept
- **4-7** System architectural design
- **4-8** System and Item integration and verification
- **4-9** Safety validation

7. Production, operation, service and decommissioning

- **7-5** Planning for production, operation, service and decommissioning
- **7-6** Production
- **7-7** Operation, service and decommissioning

12. Adaptation of ISO 26262 for motorcycles

- **12-5** Safety culture
- **12-6** Confirmation measures
- **12-7** Hazard analysis and risk assessment
- **12-8** vehicle integration and testing
- **12-9** Safety validation

5. Product development at the hardware level

- **5-5** General topics for the product development at the hardware level
- **5-6** Specification of hardware safety requirements
- **5-7** Hardware design
- **5-8** Evaluation of the hardware architectural metrics
- **5-9** Evaluation of the safety goal violations due to random hardware failures
- **5-10** Hardware integration and verification

6. Product development at the software level

- **6-5** General topics for the product development at the software level
- **6-6** Specification of software safety requirements
- **6-7** Software architectural design
- **6-8** Software unit design and implementation
- **6-9** Software unit verification
- **6-10** Software integration and verification
- **6-11** Testing of the embedded software

8. Supporting processes

- **8-5** Interfaces within distributed developments
- **8-6** Specification and management of safety requirements
- **8-7** Configuration management
- **8-8** Change management
- **8-9** Verification
- **8-10** Documentation management
- **8-11** Confidence in the use of software tools
- **8-12** Qualification of software components
- **8-13** Evaluation of hardware elements
- **8-14** Proven in use argument
- **8-15** Interfacing a base vehicle or item in an application out of scope of ISO 26262
- **8-16** Integration of safety-related systems not developed according to ISO 26262

9. ASIL-oriented and safety-oriented analyses

- **9-5** Requirements decomposition with respect to ASIL tailoring
- **9-6** Criteria for coexistence of elements
- **9-7** Analysis of dependent failures
- **9-8** Safety analyses

10. Guideline on ISO 26262

11. Guideline on application of ISO 26262 to semiconductors

Main concepts: ASIL

ASIL – Automotive Safety Integrity Level

is a standardized risk classification

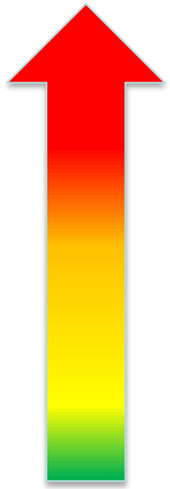
ASIL D

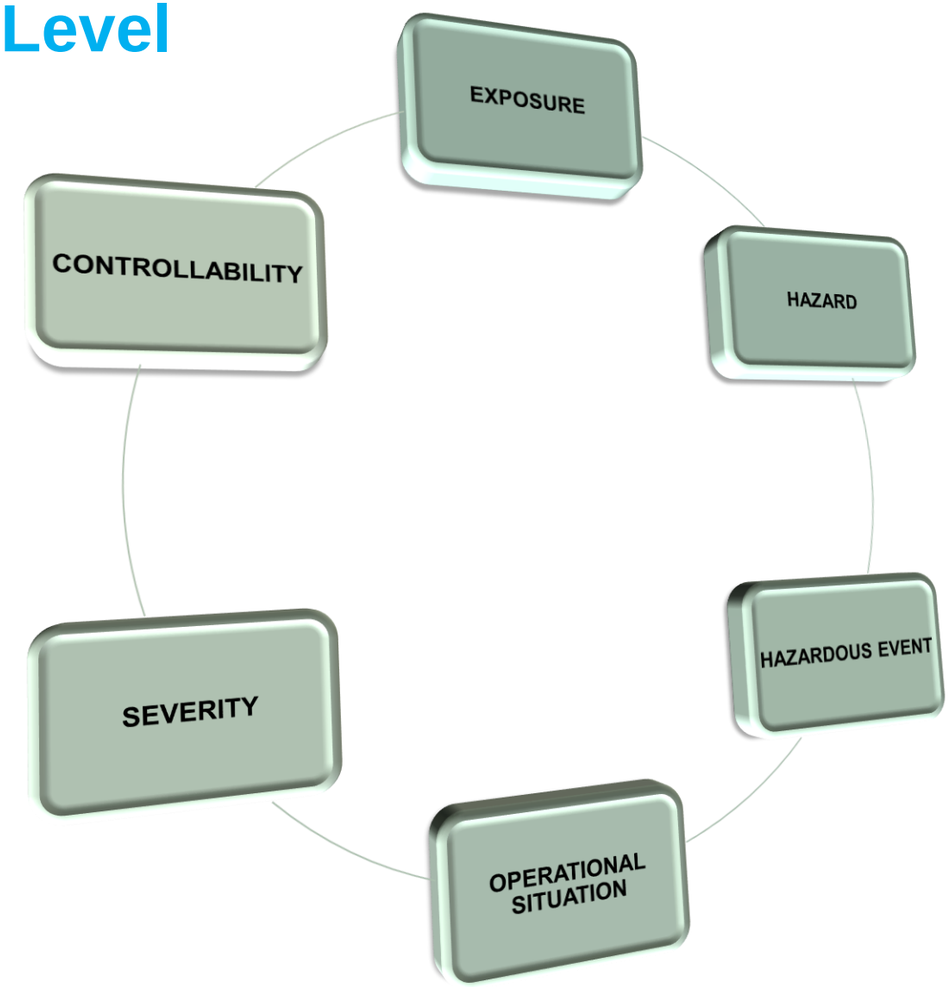
ASIL C

ASIL B

ASIL A

QM (not safety related)





Risk Analysis takes into account 3 factors:

- Exposure - Severity - Controllability

Main concepts: ASIL

ASIL is the combination of:

- Potential Risks (malfunction)
- Driving situations and environmental conditions
- Who & how might be harmed (S)
- Potential Risk frequency (E)
- Potential of avoidance (C)

		C1	C2	C3
S1	E1	QM	QM	QM
	E2	QM	QM	QM
	E3	QM	QM	ASIL A
	E4	QM	ASIL A	ASIL B
S2	E1	QM	QM	QM
	E2	QM	QM	ASIL A
	E3	QM	ASIL A	ASIL B
	E4	ASIL A	ASIL B	ASIL C
S3	E1	QM	QM	ASIL A
	E2	QM	ASIL A	B
	E3	ASIL A	ASIL B	ASIL C
	E4	ASIL B	ASIL C	ASIL D

Main concepts: Safety Goal

Safety Goal

A Safety Goal is a top-level requirement
aimed at reducing Hazardous events

Safety Goals are identified in HARA:

- All hazardous events with an ASIL shall have **at least one** associated safety goal;
- One safety goal can be related to several **hazards**, and several safety goals can be related to a single hazard;

From the Safety Goals:

- high-level **Functional Safety Requirements** (**FSR**); and
- low-level **Technical Safety Requirements** (**TSR**)

Main concepts: Safe State

Safe State

A Safe State is a mode of the **item**
without an unreasonable level of **risk**

Any item has different working conditions (operating modes) – examples:

- **Switched-off** mode
- **Intended operating** mode
- **Startup phase** mode
- **Emergency** mode

Typically, a safe state is reached by the system after **detecting a fault**. Two or more systems can cooperate in reaching the safe state, but even if the transition is driven by a single system, the **whole item** is always affected by the state transition.

Safety Goals Examples

ITEM: Electronic Parking Brake (EPB)

Failure mode	Hazard	Specific situation	Hazardous event	Possible consequences	ASIL	Safety goal	Safe state
Unintended parking brake activation	Unexpected deceleration	High speed **OR** taking a bend **OR** low friction surface	Unexpected deceleration at high speed **OR** taking a bend **OR** low friction surface	Loss of vehicle stability	*Higher ASIL*	Avoid activating the parking function without the driver's request when the vehicle is moving	EPB disabled
Unintended parking brake activation	Unexpected deceleration	Medium-low speed **AND** high friction surface	Unexpected deceleration at medium-low speed **AND** high friction surface	Rear end collision with the following vehicle	*Lower ASIL*	Avoid activating the parking function without the driver's request when the vehicle is moving	EPB disabled

From ISO 26262-3

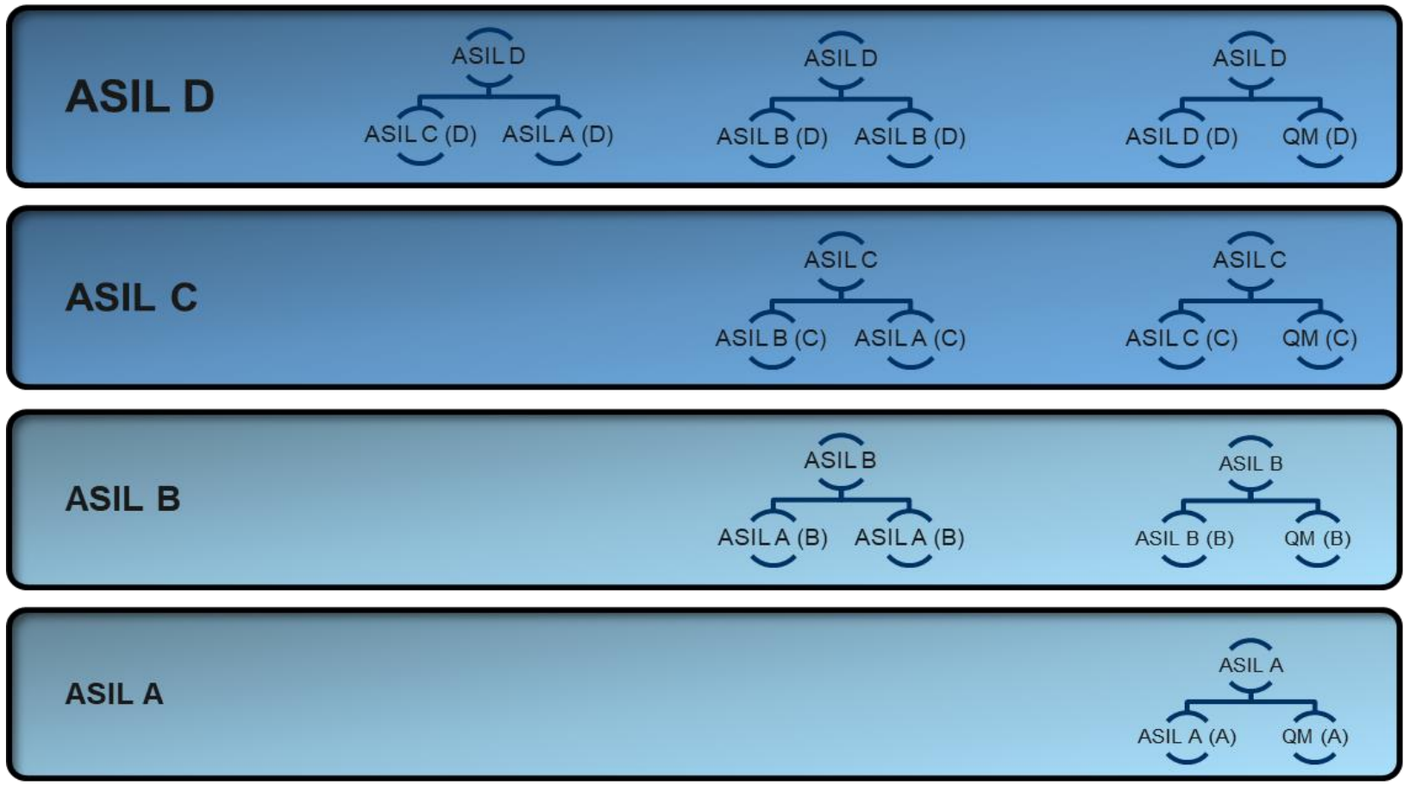
What have we learned so far?

- ISO 26262 structure
- Functional safety of the item
  - ASIL, Safety Goal, Safe State

Don’t hesitate to ask questions

FUNCTIONAL SAFETY PILLARS

Main concepts: ASIL Decomposition



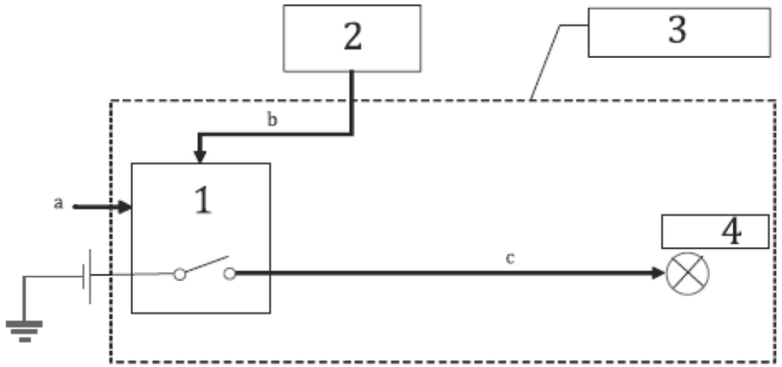
Additional requirements for Independence!

The objective of ASIL decomposition is to comply with the safety goal by using multiple sufficiently independent elements with respect to systematic faults. In such a decomposed architecture, the safety requirement before decomposition is only violated if both elements simultaneously violate their safety requirements resulting from the decomposition.

EXAMPLE: The main processor of an ECU can be monitored by a redundant monitoring processor, both of which are independently capable of initiating a defined safe state, even if the monitoring processor is not able to fulfil the functional requirements allocated to the ECU.

ASIL Decomposition Example

ITEM DEFINITION: a system with an actuator that is triggered on demand by the driver using a dashboard switch. For the purpose of this example, the actuator provides a comfort function if the vehicle is at zero speed, but can cause hazards if activated above 15 km/h



Key

1	AC ECU	a	Driver's request.
2	VS ECU	b	Vehicle speed.
3	item boundary	c	Command to the actuator.
4	actuator		

- The dashboard switch input is read by a dedicated ECU (referred to as "Actuator Control ECU (AC ECU)"), which powers the actuator through a dedicated power line.
- VS ECU can provide the information that the vehicle speed is greater than 15 km/h and it is assumed to be compliant with ASIL C requirements.

HAZARDOUS EVENT: activation of the actuator while driving at a speed above 15 km/h, with or without a driver request.

SAFETY GOAL: Avoid activating the actuator while the vehicle speed is greater than 15 km/h: ASIL C

INITIAL SAFETY CONCEPT

Requirement A 1: The VS ECU sends the accurate vehicle speed information to the AC ECU. → ASIL C

Requirement A 2: The AC ECU does not power the actuator if the vehicle speed is greater than 15 km/h. → ASIL C

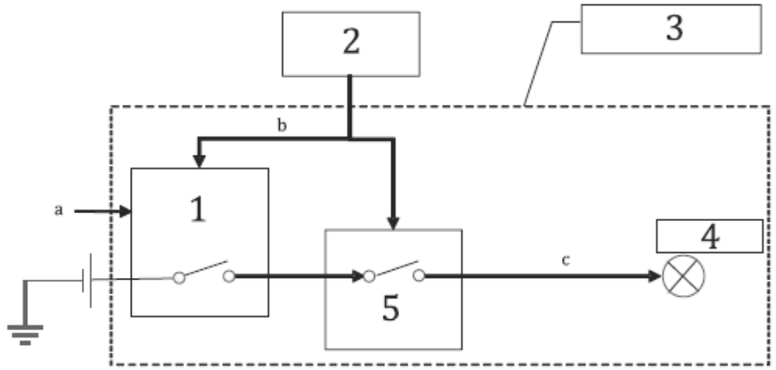
Requirement A 3: The actuator is activated only when powered by the AC ECU. → ASIL C

ISO 26262-10 Clause 11

ASIL Decomposition Example

EVOLVED SAFETY CONCEPT

The developers can choose to introduce a redundant element, here a Redundant Switch. By introducing this redundant element, the AC ECU is developed with an ASIL that is equal to or lower than ASIL C, in accordance with the results of an ASIL decomposition.



Key

1	AC ECU	5	redundant switch
2	VS ECU	a	Driver's request.
3	item boundary	b	Vehicle speed.
4	actuator	c	Command to the actuator.

- Requirement B 1: the VS ECU sends accurate vehicle speed information to the AC ECU. → ASIL C
- Requirement B 2: the AC ECU does not power the actuator if the vehicle speed is greater than 15 km/h. → ASIL X(C)
- Requirement B 3: the VS ECU sends accurate vehicle speed information to the Redundant Switch. → ASIL C
- Requirement B 4: The Redundant Switch is in an open state if the vehicle speed is greater than 15 km/h. → ASIL Y(C) (see Table 5)
- Requirement B 5: The actuator operates only when powered by the AC ECU and the Redundant Switch is closed. → ASIL C

To permit an ASIL decomposition, the developers add an independency requirement if deemed necessary:

- Requirement B 6: Sufficient independence of the AC ECU and the Redundant Switch is shown. → ASIL C

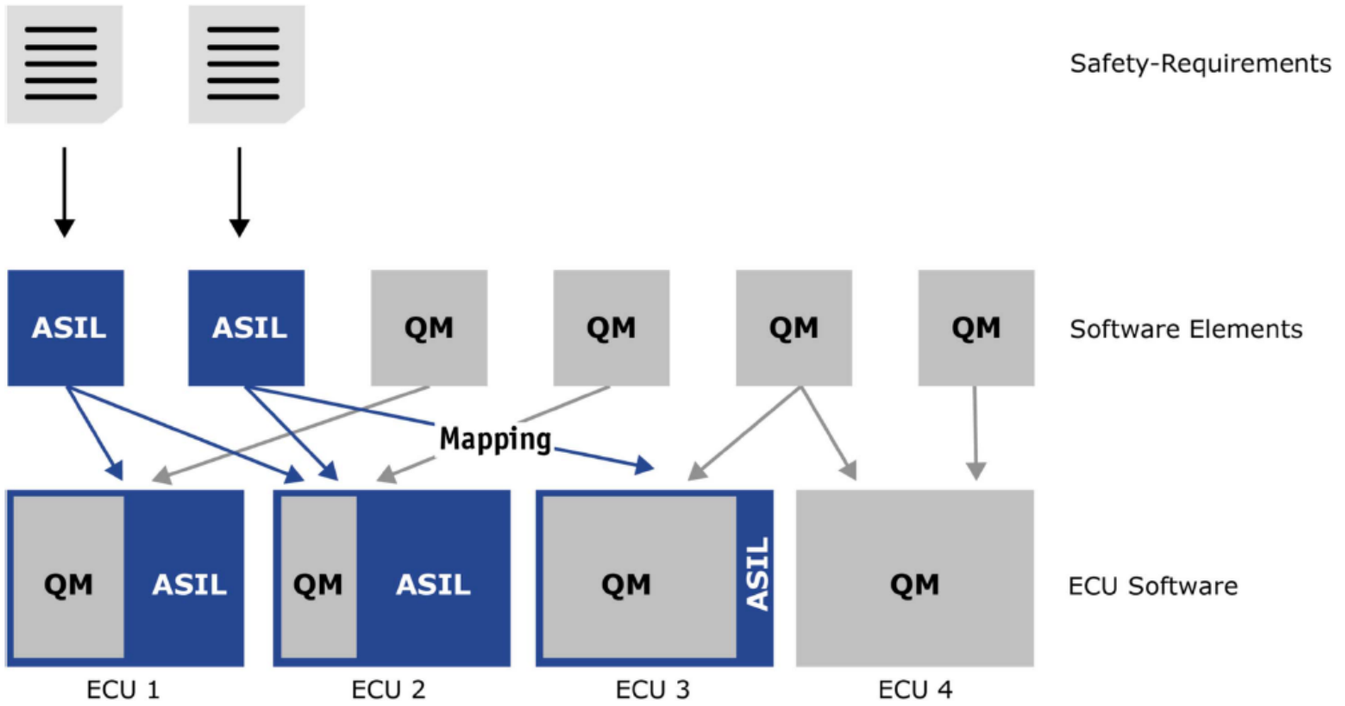
The original requirement A 2 has been replaced by the redundant requirements B 2 and B 4, both of which comply with the safety goal, and therefore ASIL decomposition can be applied.

ISO 26262-10 Clause 11

Mixed ASIL ECUs

MIXED ASIL ECUs

Safety Functions can be developed in different SW components and mapped on different ECUs



ASIL coexistence

The safety mechanisms detect and handle interference faults

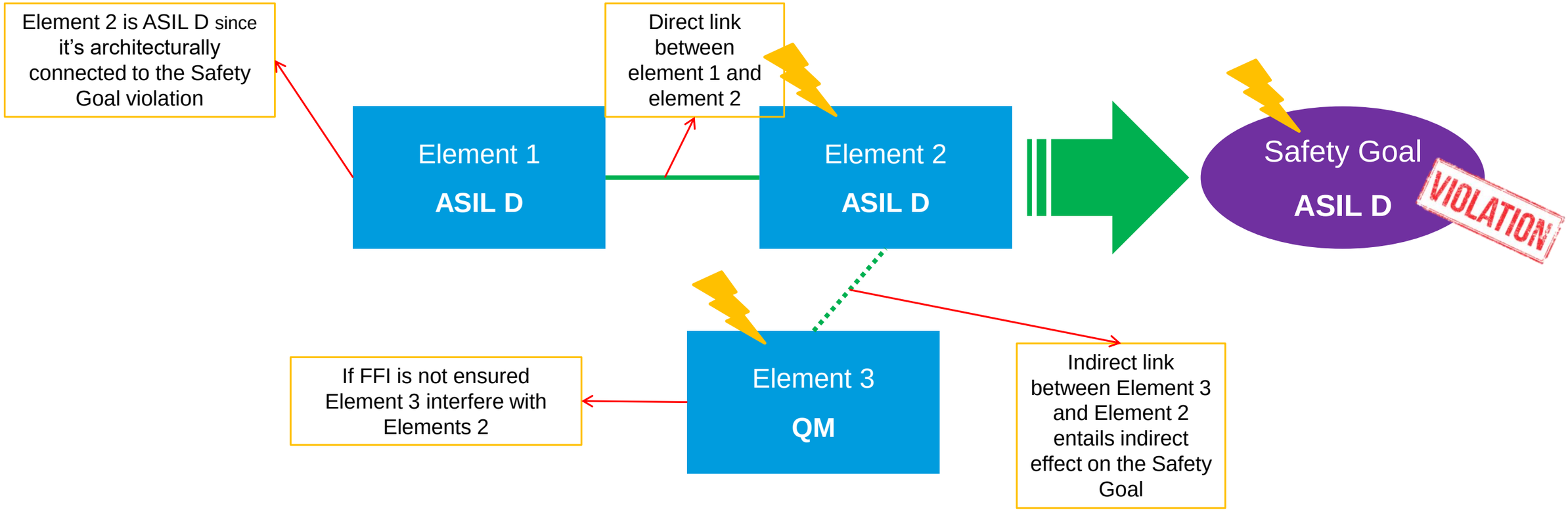
- in the basic SW
- in the application SW
- in the hardware (partly)

In this way the coexistence of software with different integrity levels is allowed!

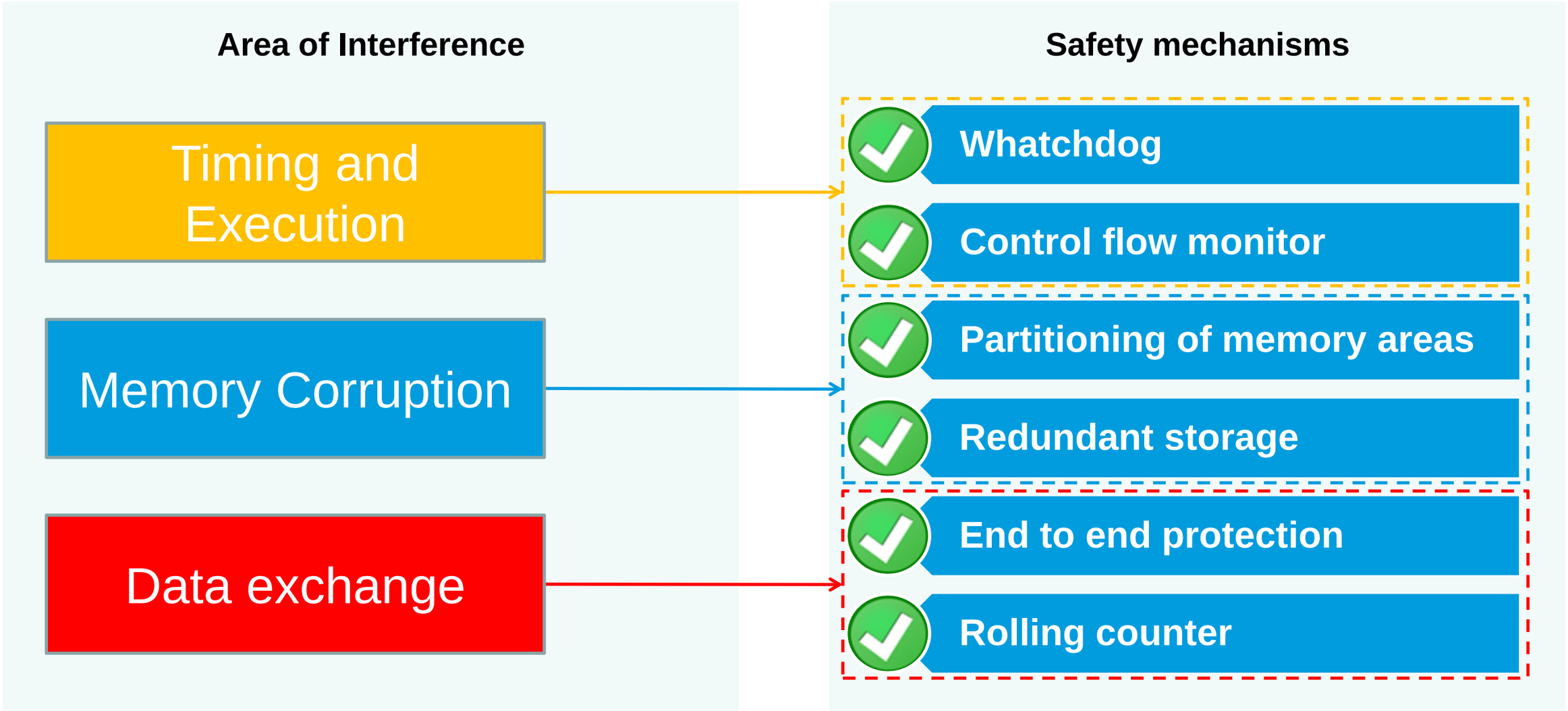
Freedom From Interference (FFI)

FREEDOM FROM INTERFERENCE

Absence of cascading failures between two or more elements that can lead to the violation of a safety requirement

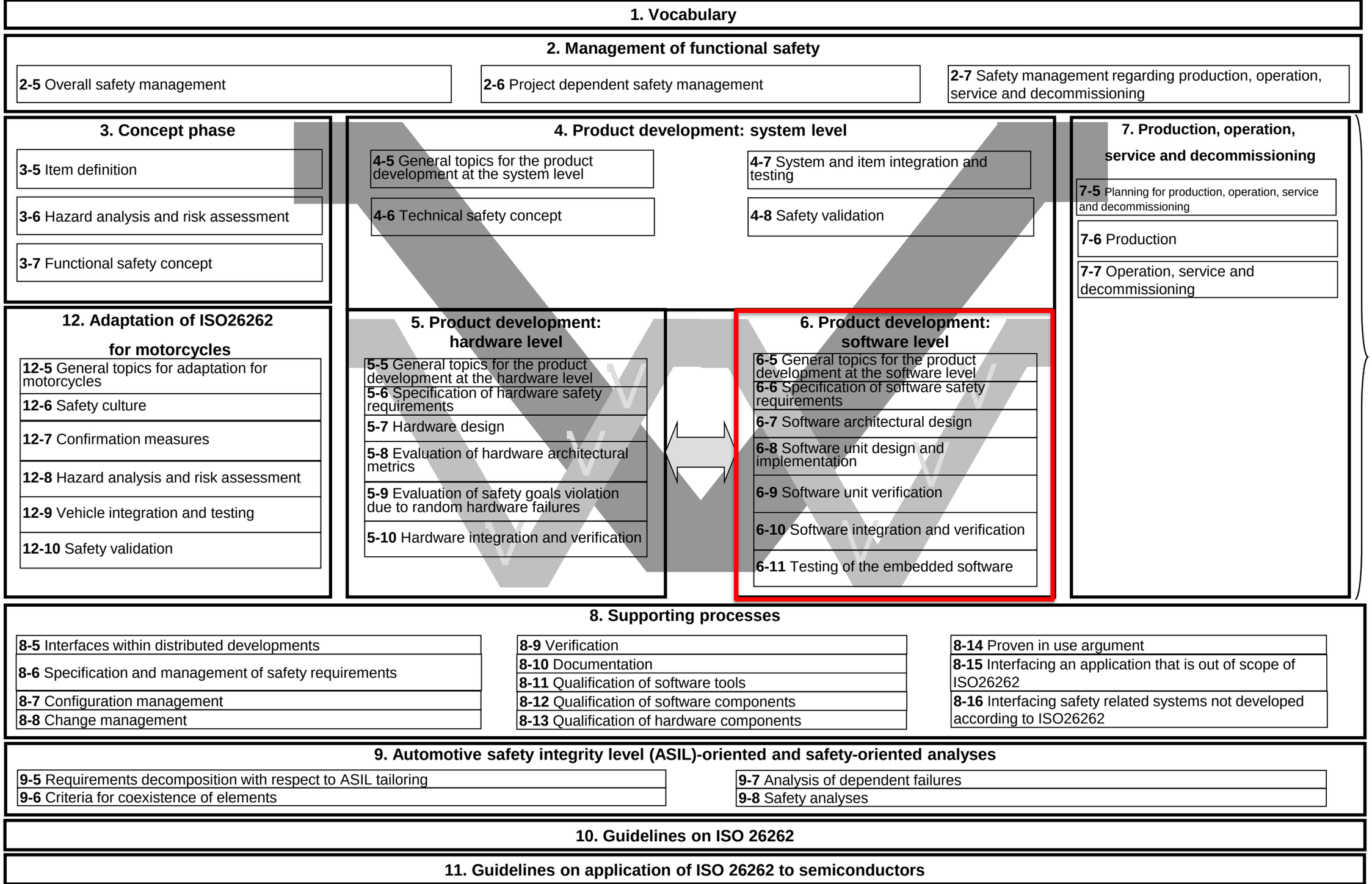


Freedom from Interference – Possible Solutions



Examples in the following slides

PART 6: PRODUCT DEVELOPMENT AT SW LEVEL


1. Vocabulary
2. Management of functional safety
2-5 Overall safety management
2-6 Project dependent safety management
2-7 Safety management regarding production, operation, service and decommissioning
3. Concept phase
3-5 Item definition
3-6 Hazard analysis and risk assessment
3-7 Functional safety concept
4. Product development: system level
4-5 General topics for the product development at the system level
4-6 Technical safety concept
4-7 System and item integration and testing
4-8 Safety validation
7. Production, operation, service and decommissioning
7-5 Planning for production, operation, service and decommissioning
7-6 Production
7-7 Operation, service and decommissioning
Core processes
12. Adaptation of ISO26262 for motorcycles
12-5 General topics for adaptation for motorcycles
12-6 Safety culture
12-7 Confirmation measures
12-8 Hazard analysis and risk assessment
12-9 Vehicle integration and testing
12-10 Safety validation
5. Product development: hardware level
5-5 General topics for the product development at the hardware level
5-6 Specification of hardware safety requirements
5-7 Hardware design
5-8 Evaluation of hardware architectural metrics
5-9 Evaluation of safety goals violation due to random hardware failures
5-10 Hardware integration and verification
6. Product development: software level
6-5 General topics for the product development at the software level
6-6 Specification of software safety requirements
6-7 Software architectural design
6-8 Software unit design and implementation
6-9 Software unit verification
6-10 Software integration and verification
6-11 Testing of the embedded software
8. Supporting processes
8-5 Interfaces within distributed developments
8-6 Specification and management of safety requirements
8-7 Configuration management
8-8 Change management
8-9 Verification
8-10 Documentation
8-11 Qualification of software tools
8-12 Qualification of software components
8-13 Qualification of hardware components
8-14 Proven in use argument
8-15 Interfacing an application that is out of scope of ISO26262
8-16 Interfacing safety related systems not developed according to ISO26262
9. Automotive safety integrity level (ASIL)-oriented and safety-oriented analyses
9-5 Requirements decomposition with respect to ASIL tailoring
9-6 Criteria for coexistence of elements
9-7 Analysis of dependent failures
9-8 Safety analyses
10. Guidelines on ISO 26262
11. Guidelines on application of ISO 26262 to semiconductors

Product development at SW level

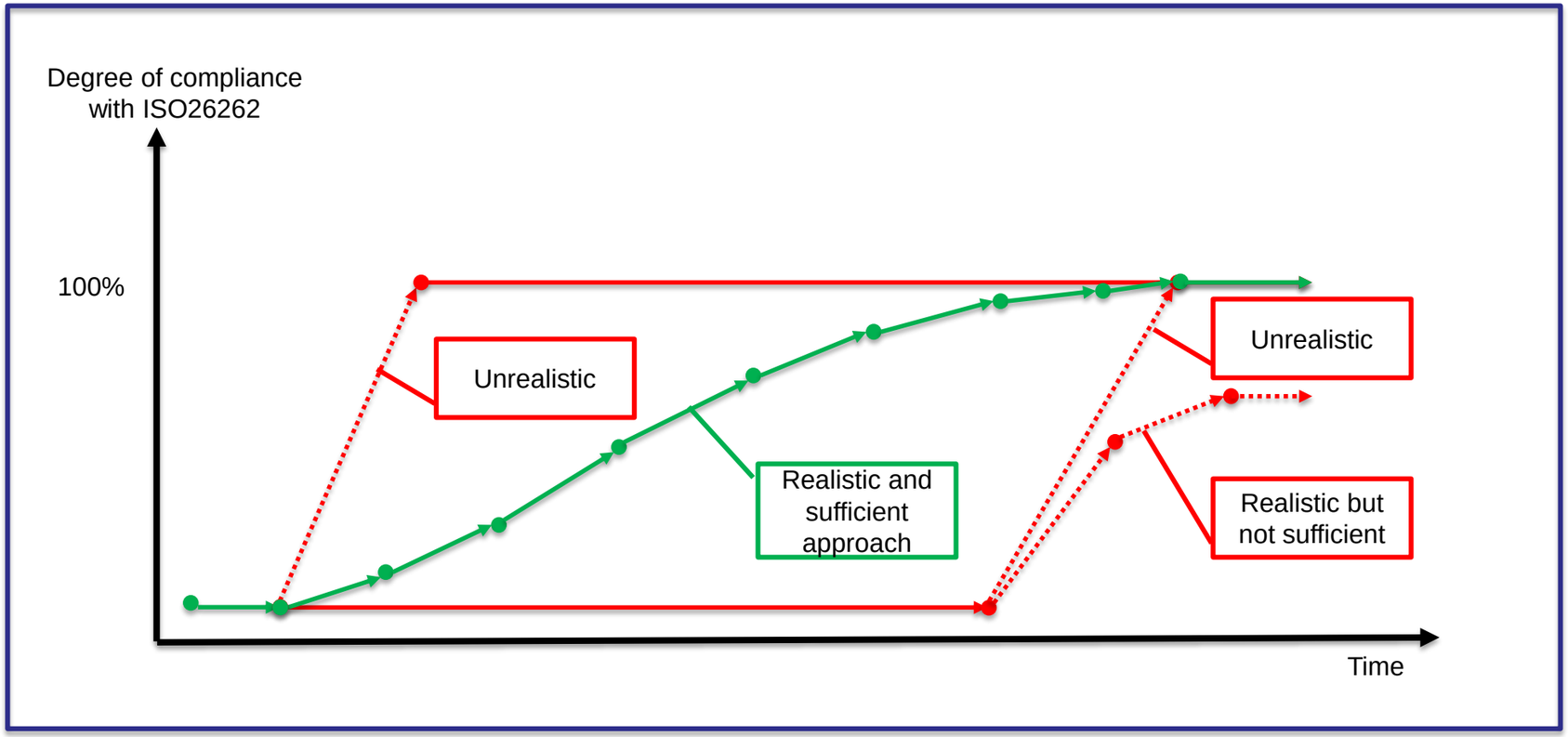
OBJECTIVES

- **Minimization of systematic faults during the software development**

Methodology:

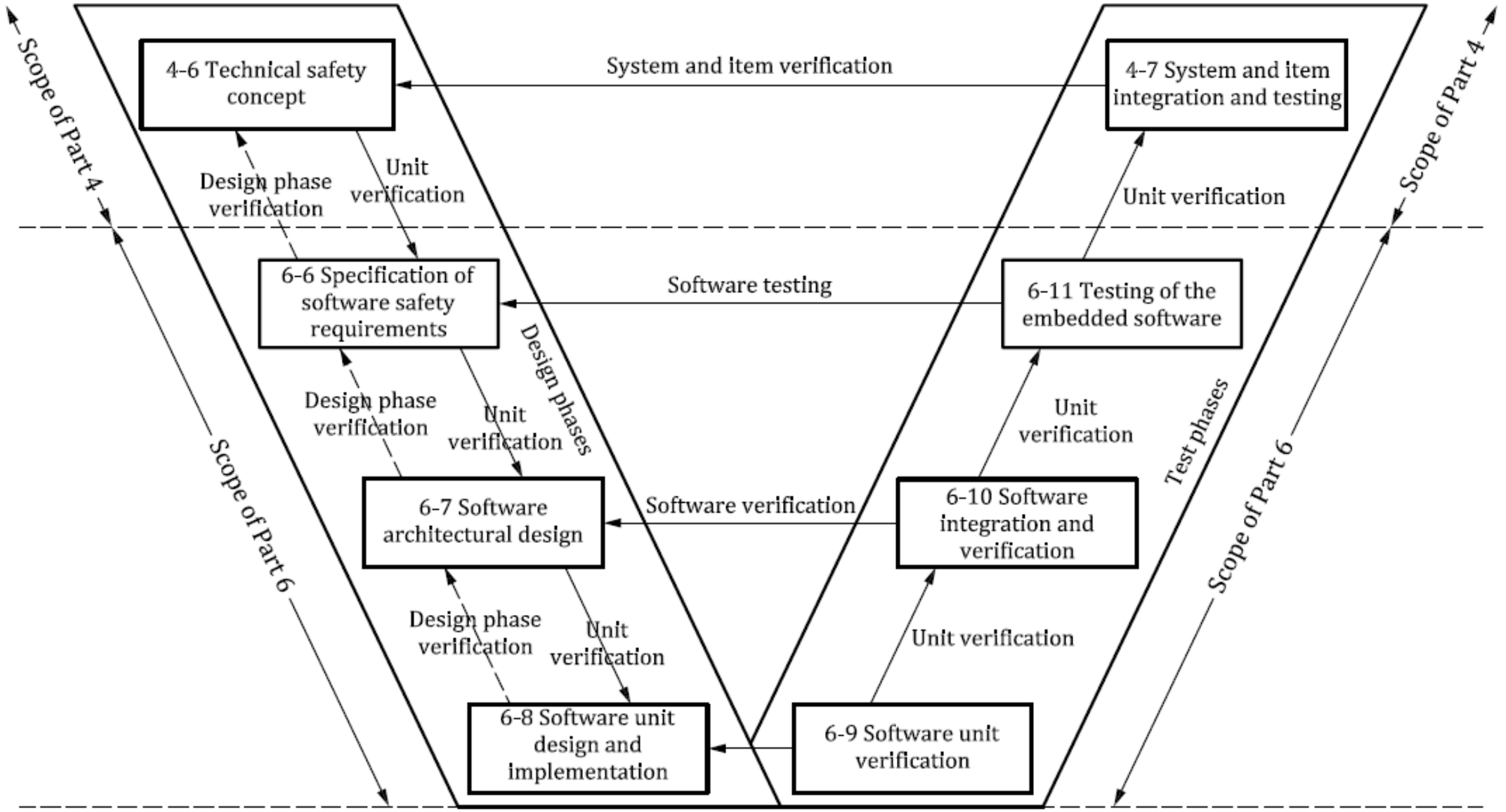
- Instantiate appropriate processes for the project (verification methods, modelling and coding guidelines)
- Determine and select the appropriate methods to comply with the requirements and their respective ASIL
- Usage of state of the art solutions (e.g. AUTOSAR, eGAS)
- Reduction of SW modules complexity

Different paths to ISO26262 SW compliance

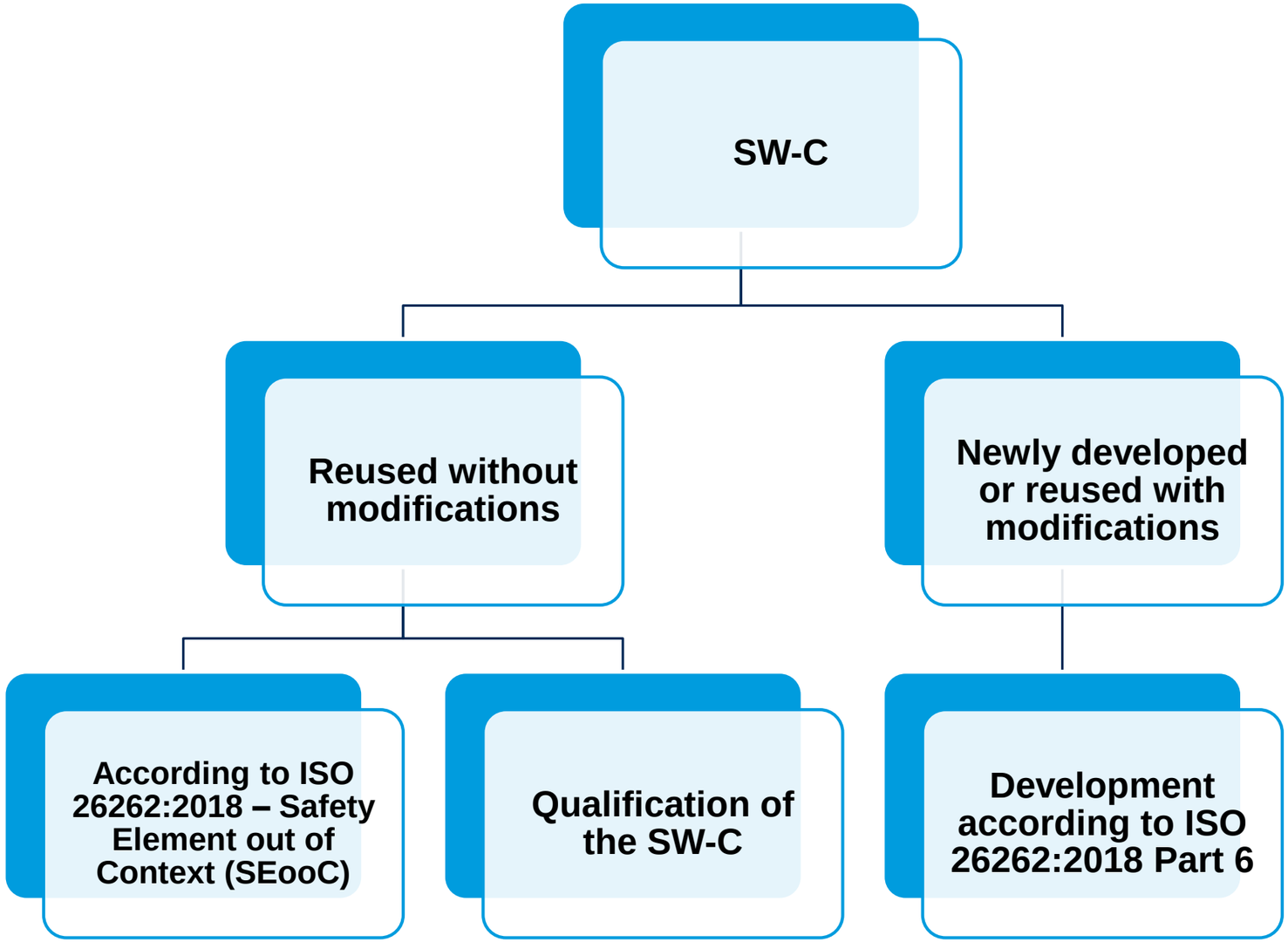


Software/Firmware Release Flow

Overall view

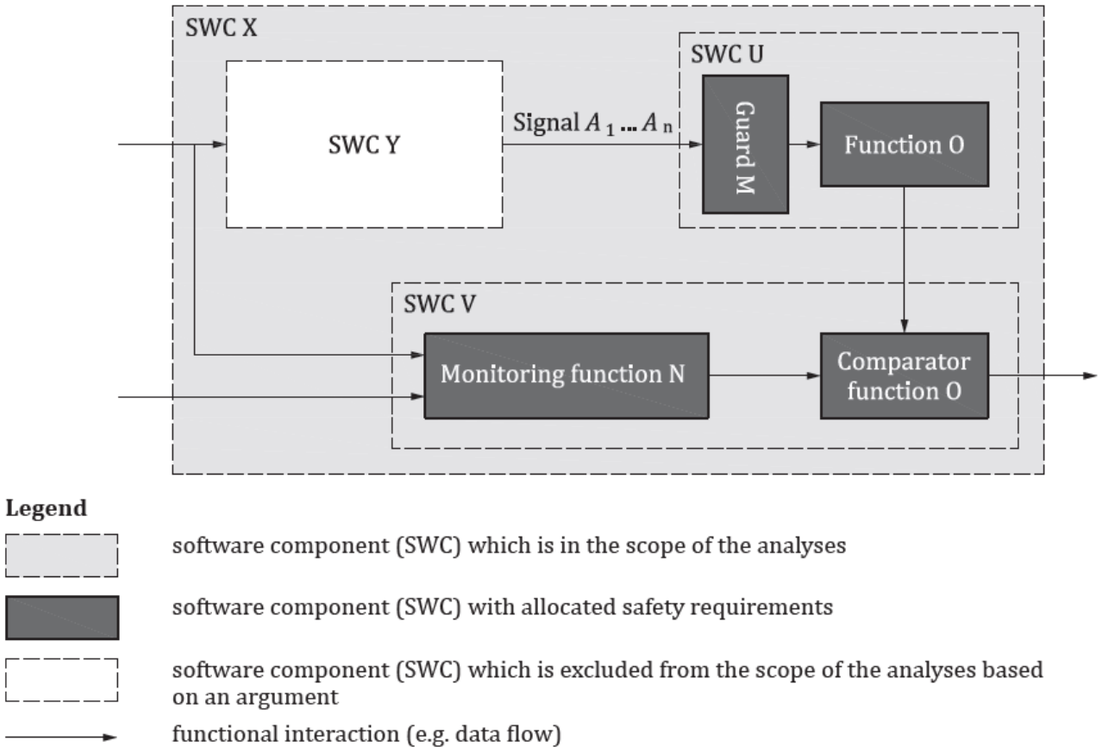


Software component classification



FMEA – ISO26262 Example

- The safety-related functionalities are implemented only in SWC U and controlled by a monitoring function implemented in SWC V.
- In this example, SWC Y can be exempted from a further detailed analysis only if the analyses of SWC X, SWC V and SWC U are able to confirm the assumed freedom from interference (e.g. no weaknesses in the Guard M regarding cascading failures with respect to erroneous signals A1 to An).



FMEA – ISO26262 Example

Examined function-ality or property	Examples for guide words	Interpretation
signal flow A_1 to A_n	After / Late	Signal too late or out of se-quence.
	Before / Early	Signal too early or out of sequence.
	No	No signal
Signal value A_1 to A_n	More	Signal value exceeds permit-ted range
	Less	Signal value falls bellows the permitted range

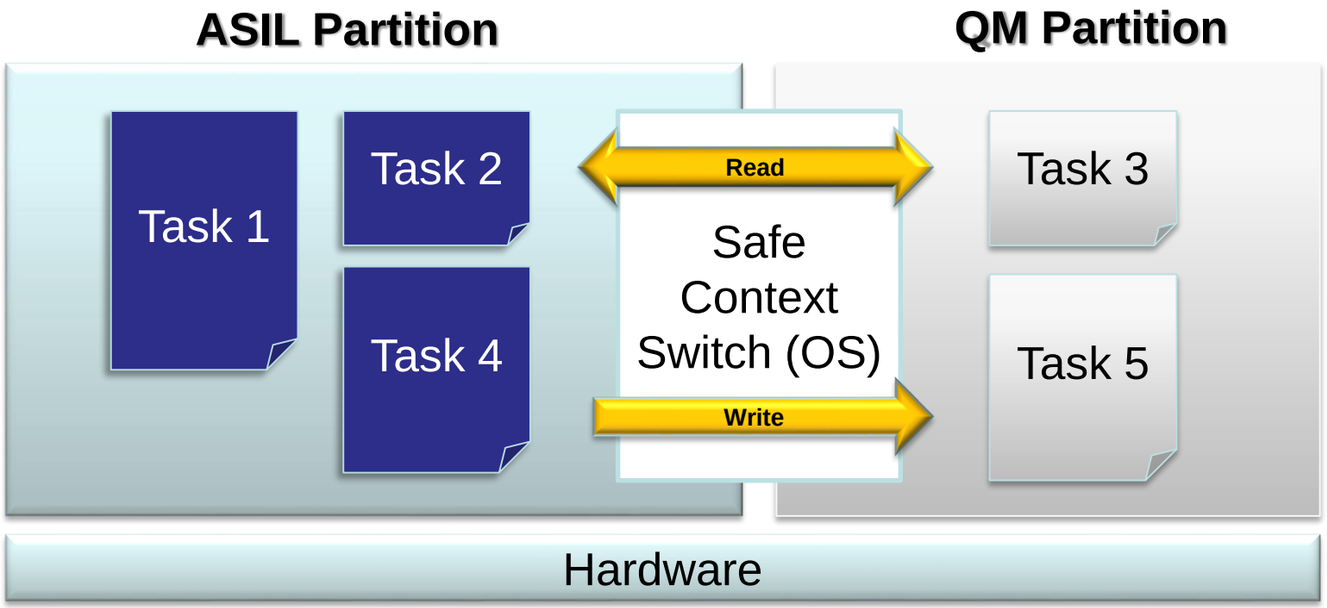
During the analysis, guide words can be applied using a deductive or inductive approach. During the analyses, a further refinement of the guide words is possible if required (e.g. to better match specific design aspects)

Example for guide word	Interpretation	Consequence	Safety measure	Required activity
More	Signal value exceeds permitted range	Function O will gener-ate erroneous output	Guard M limits sig-nal A_1 to A_n to the per-mitted maximum range	Implement Guard M
Less	Signal value falls below the permitted range	Function O will gener-ate erroneous output	Guard M limits sig-nal A_1 to A_n to the per-mitted minimum range	Implement Guard M
Other than	Values of sig-nals A_1 to A_n are inconsistent	Function O will gener-ate erroneous output	Guard M checks signal A_1 to A_n for con-sistency using physical dependencies	Implement Guard M

ISO 26262-6 Annex D

REALISTIC USE CASES

FFI – Memory Protection Unit



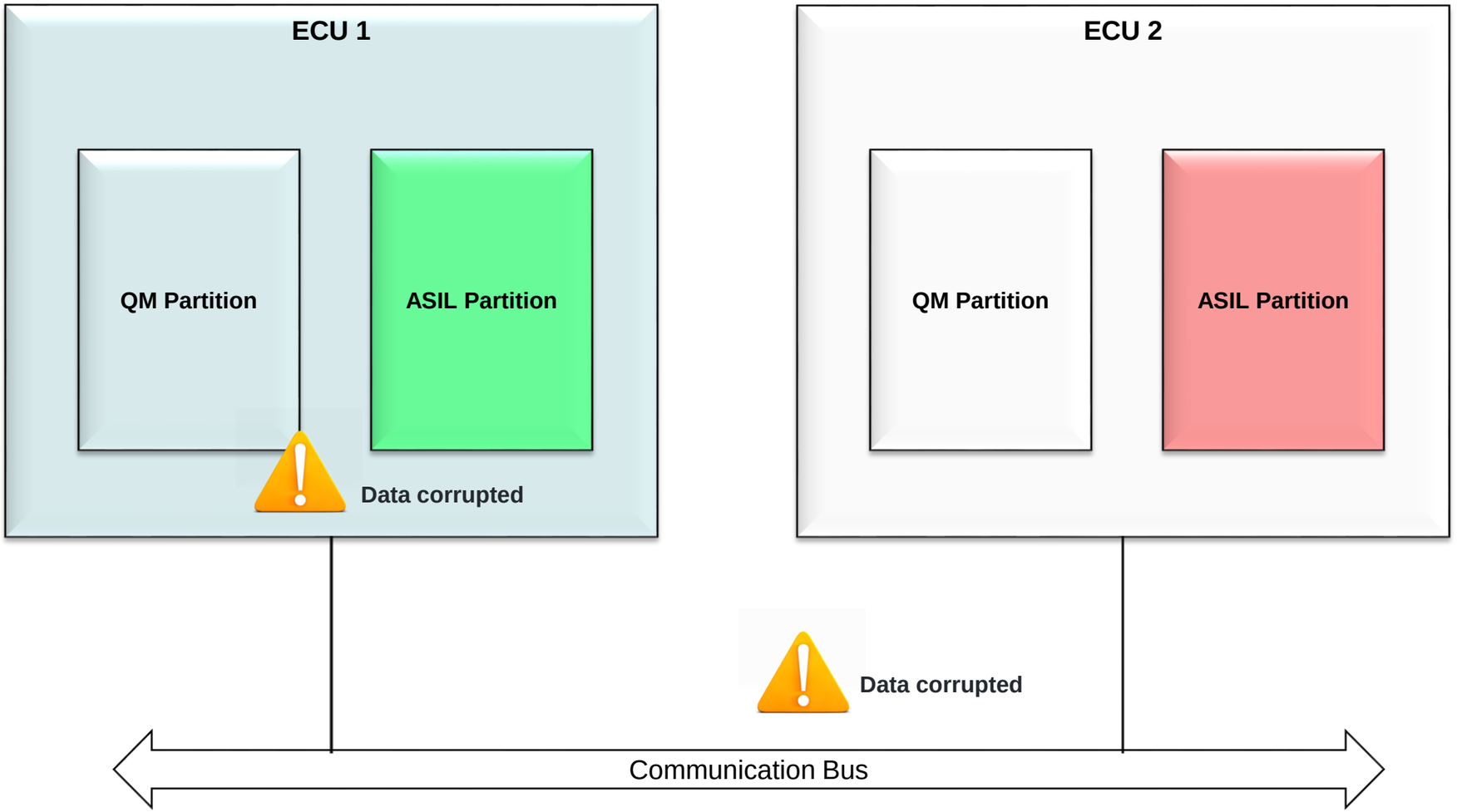
Pay attention to mixed criticality systems:

- Evaluate the allocation of those SW elements used by functionalities having different integrity
- Evaluate those cases where peripherals used for QM functionalities shall be accessed in privileged mode because of HW constraints

Other safety mechanisms against memory interfence:

- Double inverted storage: Dual storage of relevant data with a 2's complement
- Redundancy checks such as CRC to make sure data is not changed unintentionally

Source of Communication Interference



Communication

To grant the detection of failures several End2End mechanisms included in the used communication protocols can be applied

Communication errors	Safety Mechanisms							
	Sequence number	Time Stamp	Time expectation/ Time Out[c]	Connection Authentication	Feedback message[d]	Redundancy with cross checkimg	Prioritisation of messages	Different data integrity assurance systems
Corruption					X	X		
Unintended repetition	X	X				X		
Incorrect Sequence	X	X				X		
Loss	X				X	X	X	
Unacceptable delay		X	X					
Insertion	X			X	X	X		
Masquerade[a]				X	X			X
Addressing[b]				X				

NOTE:

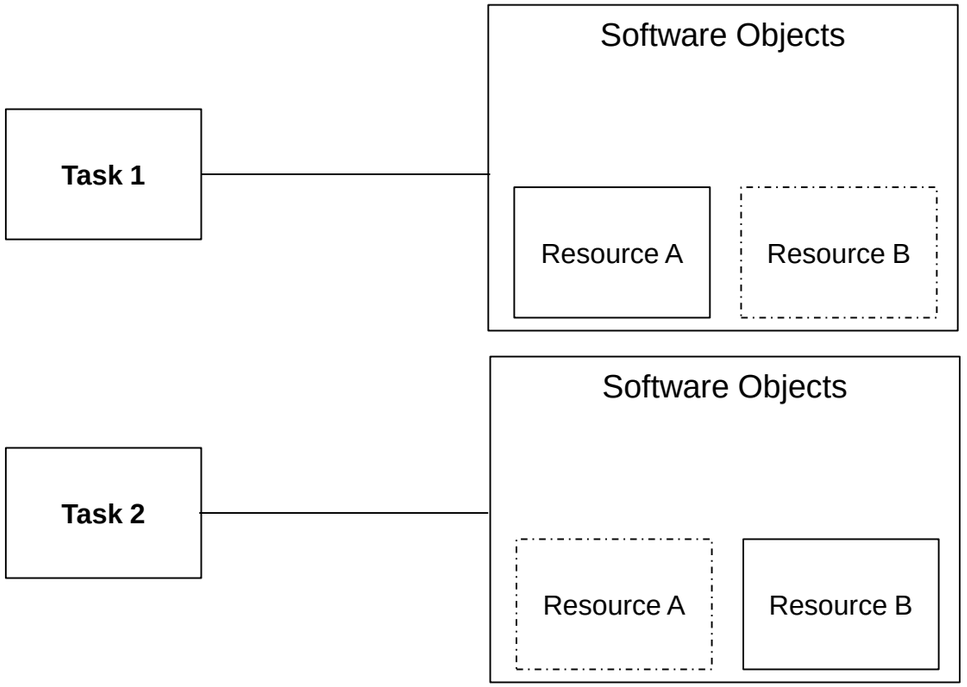
[a] A type of communication fault, were additional information is inserted into a stream of transmitted information.

[b] A type of communication fault, were non-authentic information is accepted as authentic information by a receiver.

[c] In most cases the content of a message is only valid at a particular point in time.

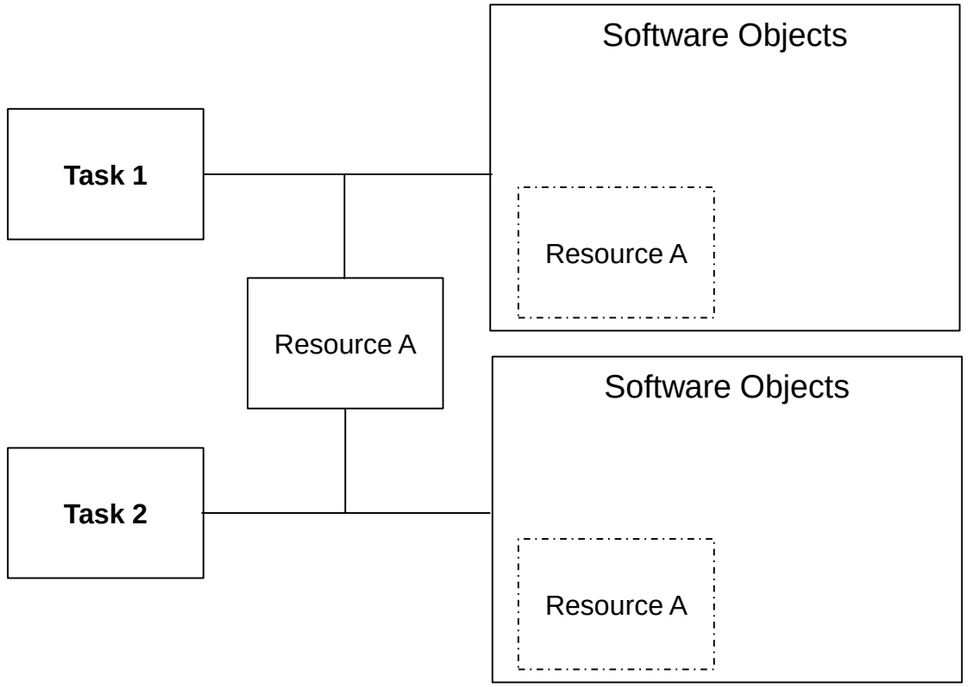
[d] The feedback has to be processed by a safety communication layer and in many case it will include the original message or a reference to it, in order to allow an integrity checking by the source.

Source of Timing Interference: Deadlock vs Livelock



DEADLOCK

- Task 1 needs Resource B and have Resource A.
- Same way Thread 2 needs Resource A and have Resource B



LIVELOCK

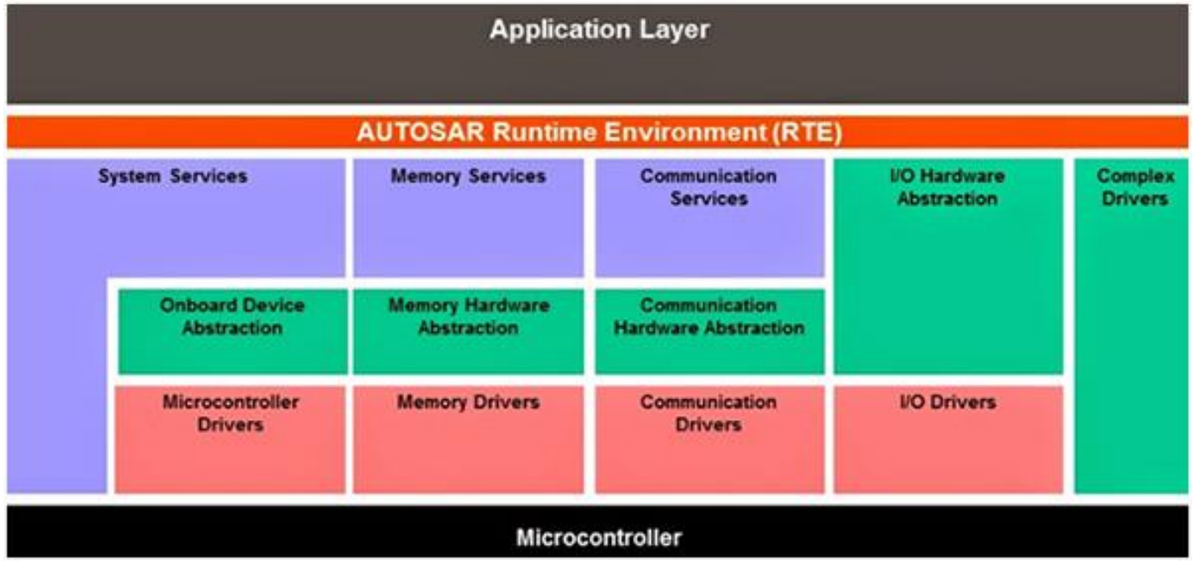
- Situation where a request for an exclusive lock is denied repeatedly, as many overlapping shared locks keep on interfering each other.

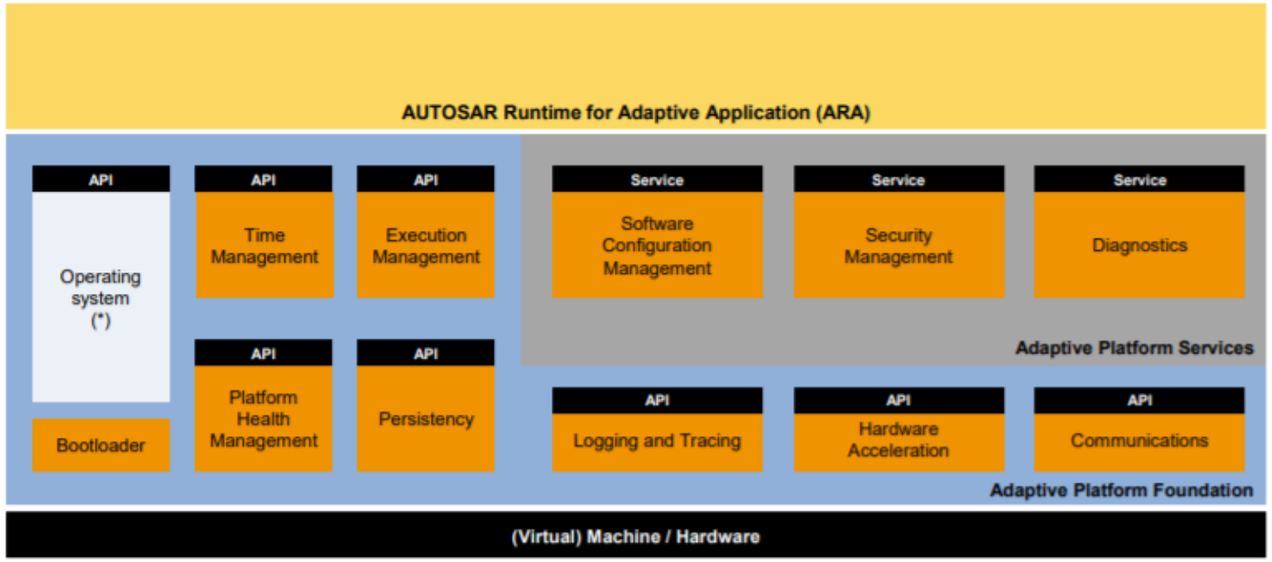
Timing and Execution Monitoring

LOGICAL SUPERVISION	▪ Supervision of sequence of checkpoints • Checkpoint stack • Entry exit signature at each checkpoint
DEADLINE SUPERVISION	▪ Supervision of time between two subsequent checkpoints
ALIVE SUPERVISION	▪ Supervision of periodicity of a checkpoint

Logical/deadline/alive supervision shall be supported by “independent” hardware for time supervision

Functional Safety and AUTOSAR





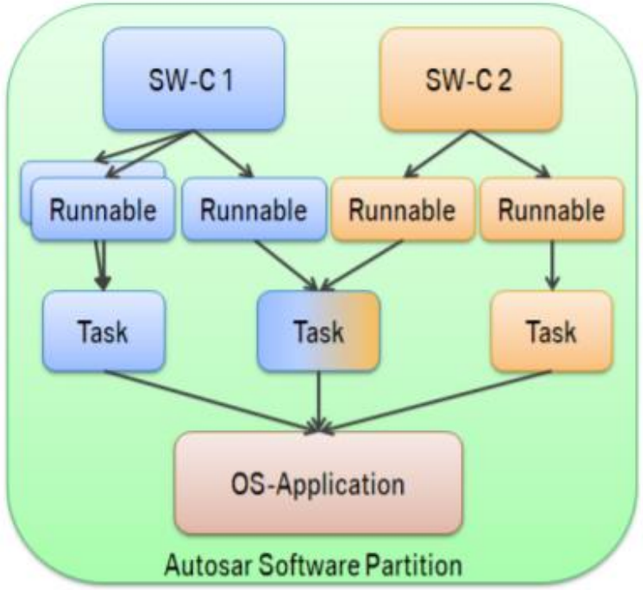
AUTOSAR CLASSIC PLATFORM

- Decouple Application from the infrastructure
- Functional Safety Concept quite mature
- Based on OSEK OS (standardized)

AUTOSAR ADAPTIVE PLATFORM

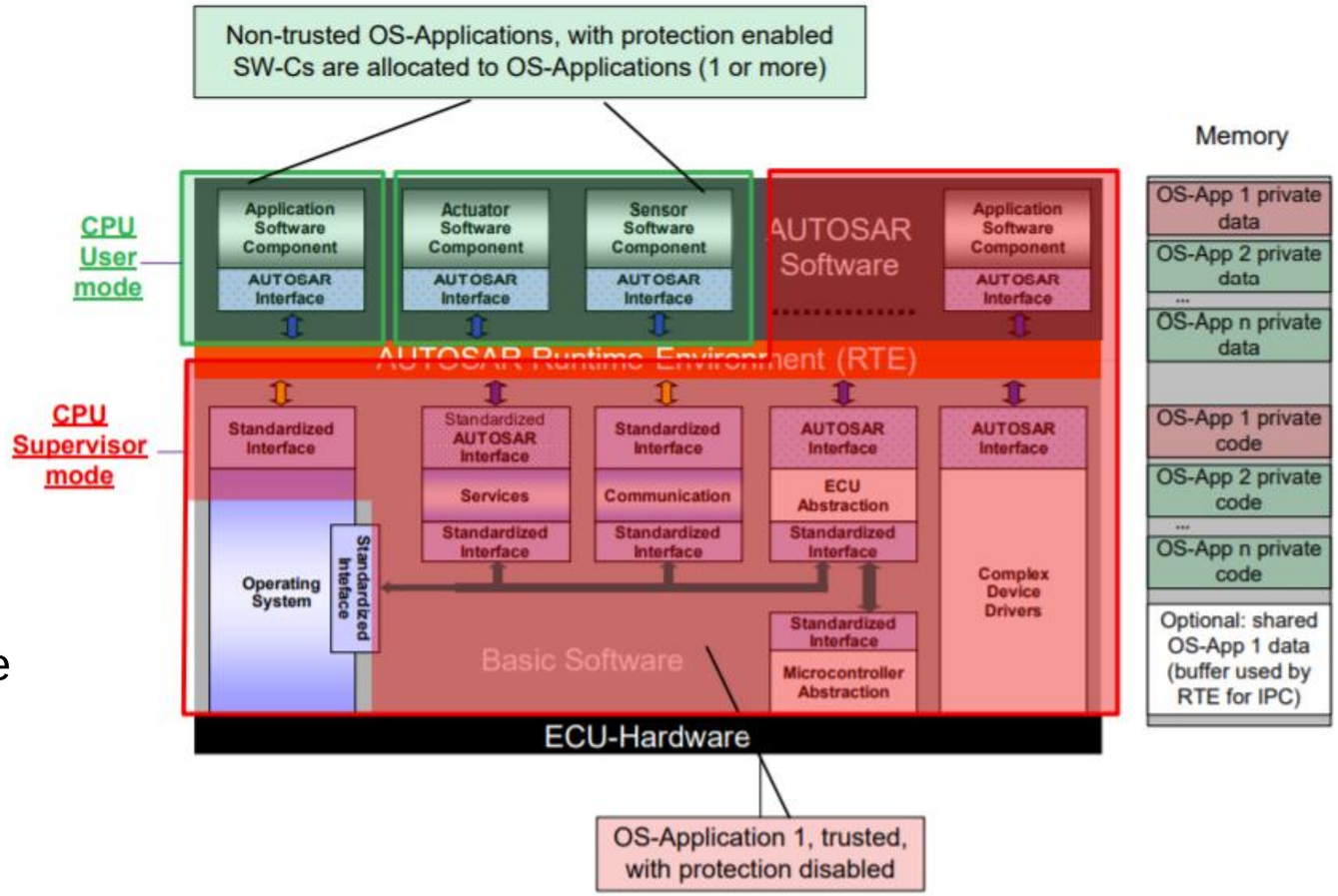
- Service Oriented Architecture
- Improvement about Functional Safety Concept in progress
- Based on Posix OS (not standardized)

Example of Safety mechanism in AUTOSAR CP



- Some SW-Cs are logically grouped and put in separate non- trusted/user mode memory partitions (highlighted in green).
- Selected SW-Cs belong to the same trusted/supervisor-mode memory partition as the Basic Software Modules.
- There may be several non-trusted/user-mode13 partitions, each containing one or more SW-C

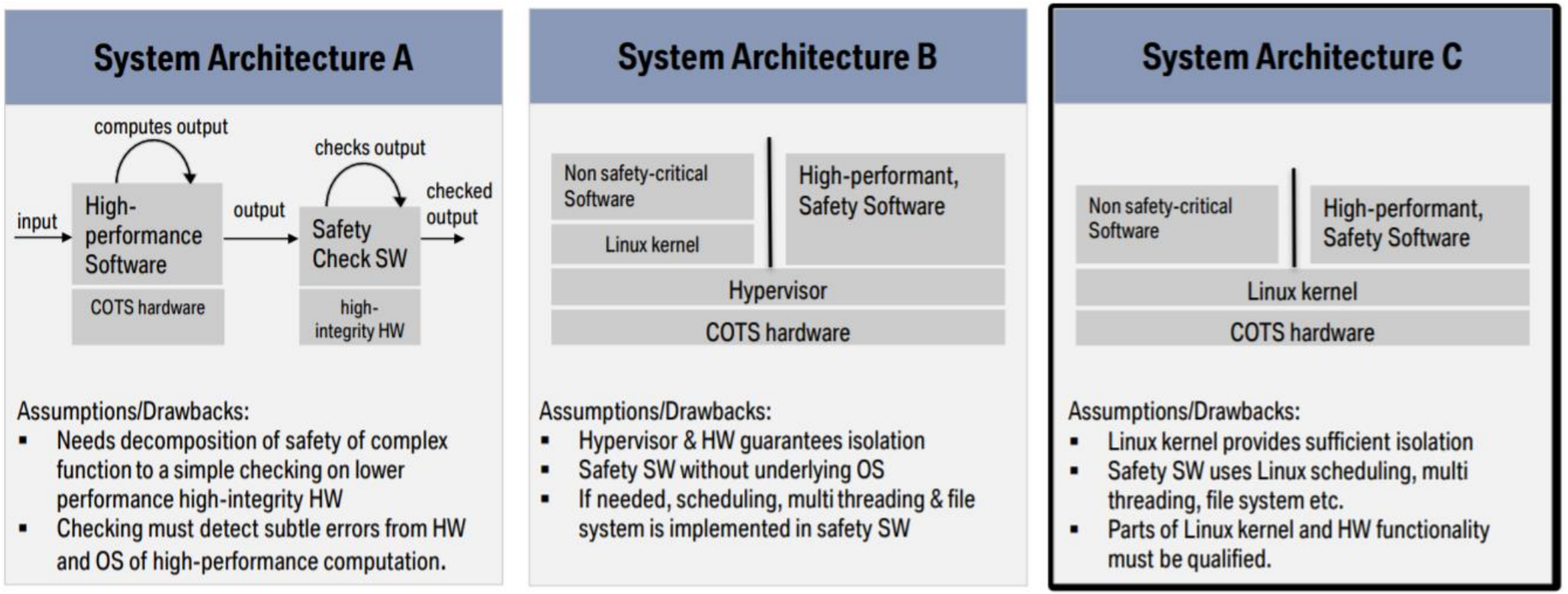
- A possible implementation where all Basic Software Modules are executed in one trusted/supervisor-mode memory partition.



How might Linux be used in safety-related systems

The challenge:

- 20 million lines of code & 2 million lines of documentation
- 14,000 commits in every release/every 70 days
- - 17,000 developers in total history, 1,700 developers in each release



Many safety concepts and the activities of ELISA project are currently focusing on alternative C

Thanks to: Lukas Bulwahn, Embedded Linux Conference

ELISA challenge: Demonstrate Linux FFI

Freedom from Interference

Absence of cascading failures between two or more elements that could lead to the violation of a safety requirement (ISO26262-1)

What shall be guaranteed according to ISO 26262?

Basic concepts of MCS

A **Mixed-Criticality System** (MCS) is a real-time system that comprises tasks having two or more distinct **Safety Integrity Levels** (ASIL). When these systems are implemented on the same embedded platform (*consolidation*) they may share computation and communication resources.

There are two conflicting trends in the design of such systems:

- **Safety requirements are** being increasingly **emphasized**
- **More functionalities** are being **implemented on integrated platforms** due to size, weight and power (SWaP) constraints.

Objectives

- Reduced Weight
- Reduced Space
- Reduced Power Consumption

Benchmarks

- Performance
- Functional Safety

Possible solutions

- MCSs are usually divided in two criticality levels:
  - ***high-level*** for safety-critical partition (e.g., ASIL-D or ASIL-B)

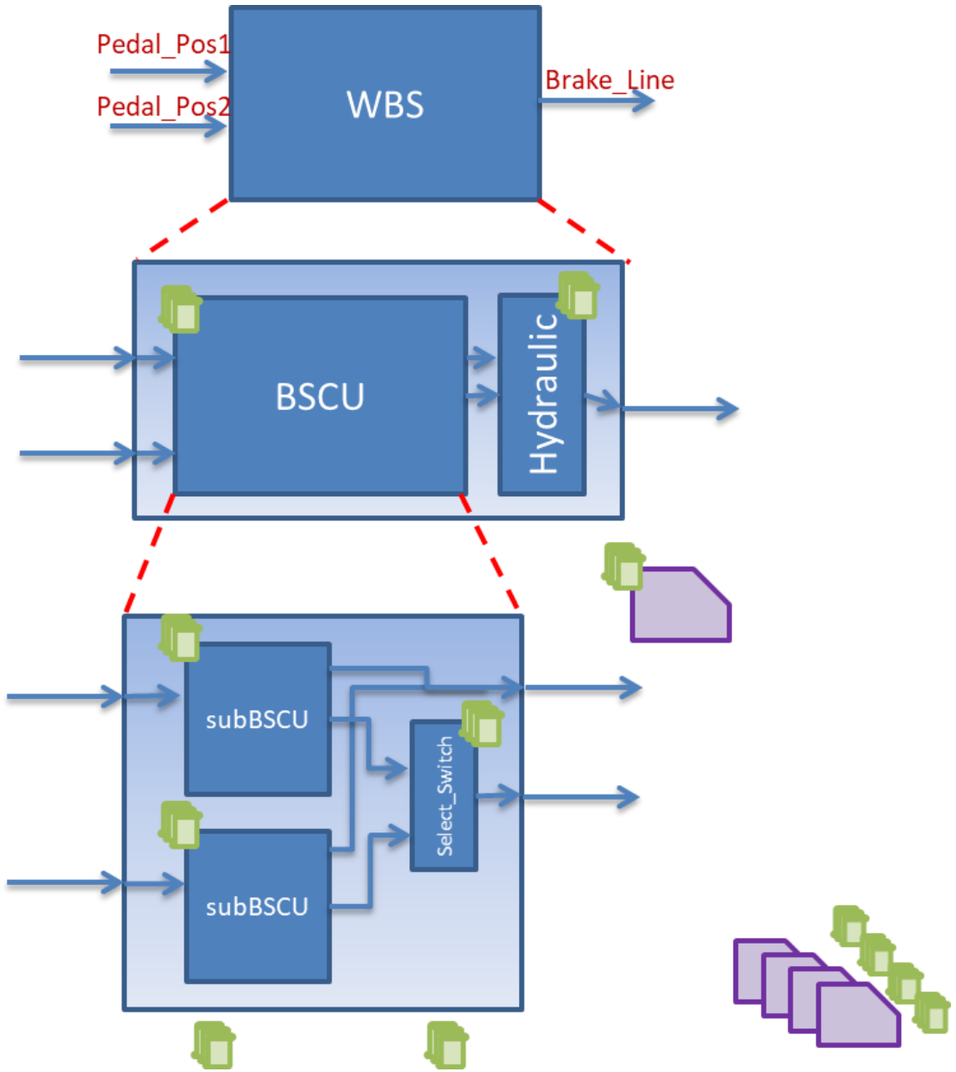
  and
  - ***low-level*** for non-critical partitions (i.e., QM)

- The strategy of the current mitigation mechanisms is to **allow the nominal functionality of the VM** (or partition) **with the highest ASIL** by reducing the interference due to VMs with lower criticality by:
  - *suspending the execution* of lower-critical VMs
  - reducing the interference due to lower-critical VMs through *frequency scaling*

- It is worthwhile to notice that in proposed solutions, typically, there is **only a safety-related VM running in the system at a given time**.

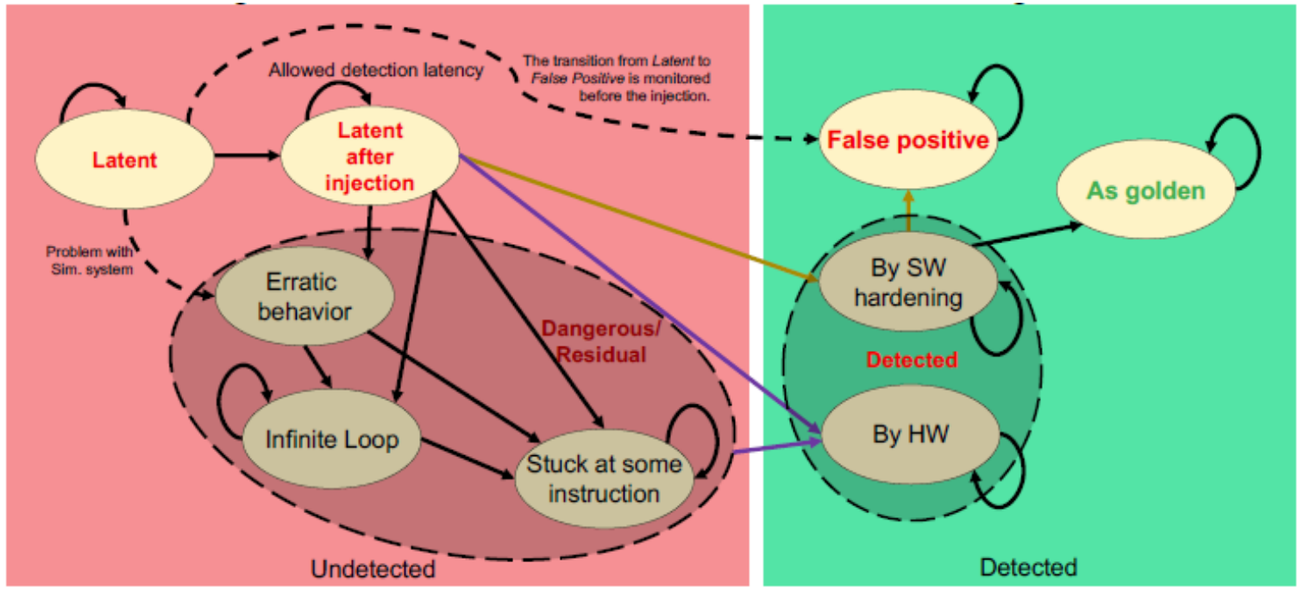
Functional Safety and Model Based Approach

- Contracts used to specify assumptions and guarantees
  - First conceived for software, now popular also for system architectural design
- Assumptions and guarantees are properties respectively of the environment and of component
- Can be seen as assertions for component interfaces.
- Contracts used for:
  - Early validation of refinement
  - Composition verification
  - Ensuring correct reuse



Thanks to: Alessandro Cimatti, Huawei Automotive Software Innovation workshop

Hardware Faults Classifier



Citation: *Sini, J.; Violante, M.; Tronci,F. A Novel ISO 26262-Compliant Test Bench to Assess the Diagnostic Coverage of Software Hardening Techniques against Digital Components Random Hardware Failures. Electronics* ***2022****, 11, 901. https://doi.org/10.3390/electronics11060901*

Assessment based on comparisons between the golden (fault-free) runs and the fault-affected ones, with the following descriptions:

• Safe: if the detection mechanism is triggered, and the output is the same as the one obtained from the golden run (regardless of whether it is due to the type of fault or the algorithm by itself, or thanks to a software-based mitigation strategy). Of course, to apply such a hypothesis, it is required that the software component is developed in compliance with the prescriptions of ISO 26262 part 6.

• Latent: if the detection is not triggered, but the output is the same as the one obtained from the golden run.

• Dangerous: the set of failure modes for which the detection mechanism is not triggered and for which the output is different from the golden run; so, it also includes multiple-point perceived faults.

• Residual: based on the frequency of latent after injection (false negatives) obtained during the fault injections for a given fault

False positive: when the detection mechanism is triggered, but no fault has been yet injected.

What have we learned so far?

- Key functional safety terminologies
- Realistic use cases at software level

Don’t hesitate to ask questions

Challenges and ideas for cooperation

- Improve the safety integrity of AI systems
- Application of Neural Network to basic software (hypervisor, scheduling decision..)
- Usage of Linux (or POSIX based OS) in safety related systems
- Usage of formal methods for safety related design
- Automatize the realization of safety analysis on complex software systems
- Hardware/Software co-design on innovative RISC V solutions
- Smart safety-secure mechanisms for embedded software
- Optimized testing techniques for safe basic-software products
- Usage of RUST for safety related applications

Thank you

fabrizio.tronci@huawei.com
www.huawei.com