Verlässliche Echtzeitsysteme – Können wir unseren Autos noch vertrauen?

Bernhard Sechser

Method Park Consulting GmbH, Erlangen

23.06.2015

Contents

Contents

- **Who is Method Park?**
- Why do we need Safety Standards?
- Process and Safety demands in Automotive
- Hazard Analysis and Risk Assessment
- Functional and Technical Development
- Software Process in detail
- Tool Qualification
- Summary

Method Park - Facts and Figures

methodpark

Facts

- Founded in 2001
- Locations:
  Germany: Erlangen, Munich, Stuttgart
  USA: Detroit, Miami

Awards

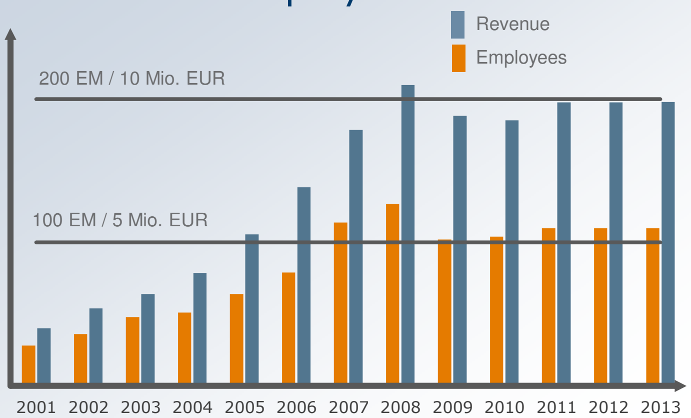
TOP JOB 2004 2008 2011

IHK Gründerpreis 2005

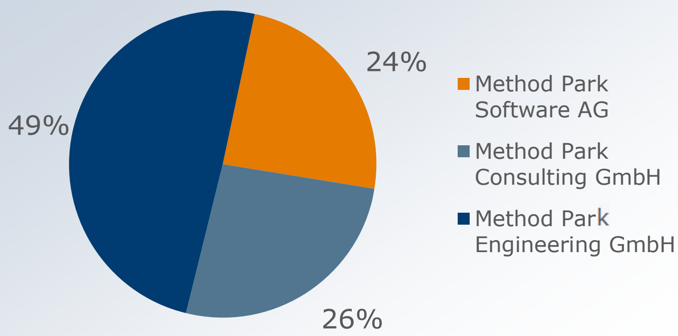
Bayerns Best 50 2006, 2007, 2009

Great Place to Work – Deutschlands Beste Arbeitgeber 2009 – Handelsblatt 2009

Revenue & employees



Business unit revenue



Portfolio

Product

stages

Solution for integrated process management

Engineering

Areas:

- Project Coaching
- Software Development & Support
- On Site Support
- Off Site Projects
- Fixed Price Projects

Consulting/Coaching

Topics:

- CMMI®, SPICE, Automotive SPICE®
- Project Management & Agile Development
- Process Improvement & Quality Management
- Functional Safety (ISO 26262)
- Variant & Complexity Management
- Product Line Management (PLM)
- Application Lifecycle Management (ALM)
- Requirements Management
- System & Software Architecture & Design
- AUTOSAR
- System & Software Testing

Training

Wide range of seminars in the division systems and software engineering

Accredited by the following organizations: SEI, ISTQB, iSQI, iNTACS, IREB, iSAQB, ECQA

Our Customers

Automotive
- Audi
- Automotive Lighting
- Blaupunkt
- BMW
- Bosch
- Brose
- Continental
- Daimler
- Delphi
- ETAS
- HE System Elektronik
- Helbako
- Hella
- IAV
- Johnson Controls
- Knorr-Brakes
- Kostal
- Marquardt
- Peiker Acustic
- Preh
- Renesas
- Thales
- TRW
- Volkswagen
- Webasto
- Witte Automotive
- ZF
- Zollner

Engineering/ Automation
- 7 layers
- ABB
- BDT
- Carl Schenk
- EBM Papst
- Heidelberger Druckmaschinen
- Insta
- Kratzer Automation
- Magirus
- Mettler Toledo
- Mühlbauer Group
- Rohde&Schwarz
- Siemens Industries
- Wago

Government/Public
- Bundesagentur für Arbeit
- Curiavant
- Kassenärztliche Vereinigung Baden-Württemberg

Healthcare
- Carl Zeiss
- Siemens
- Fresenius
- Agfa
- Ziehm Imaging
- NewTec
- Innovations Software
- Technology

IT/ Telecommunications
- GFT
- Intersoft
- Nash Technologies
- NEC
- Micronas
- Siemens
- Teleca

Defense
- Airbus Deutschland
- Diehl
- EADS
- Elbit
- Orbital
- Raytheon Anschütz
- KID

Further
- Bosch und Siemens Hausgeräte
- Deutsche Post
- GMC Software Technologies
- Kodak
- Landesbank Kiel
- Raab Karcher
- Giesecke & Devrient
- Thales Rail Signaling

Contents

Example – Ariane 5 (July 4th, 1996)

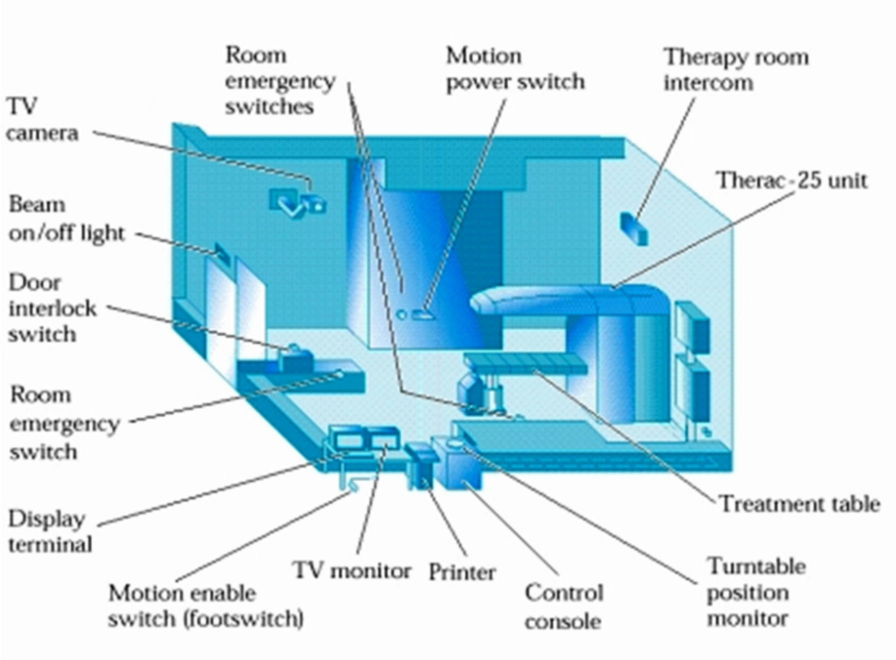
Source: ESA

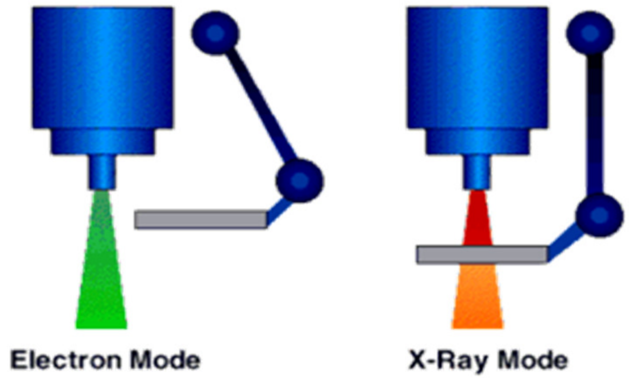
Detonation shortly after takeoff because of an error in the control software

Root cause:
Insufficient tests of a reused "proven in use" software component

Source: YouTube

Example – Therac-25





Irradiation of patients with a lethal dose

Root cause:
Insufficient safety functions

PATIENT NAME : TEST				
TREATMENT MODE : FIX		BEAM TYPE: X	ENERGY (MeV): 25	
		ACTUAL	PRESCRIBED	
UNIT RATE/MINUTE		0	200	
MONITOR UNITS		50 50	200	
TIME (MIN)		0.27	1.00	
GANTRY ROTATION (DEG)		0.0	0	VERIFIED
COLLIMATOR ROTATION (DEG)		359.2	359	VERIFIED
COLLIMATOR X (CM)		14.2	14.3	VERIFIED
COLLIMATOR Y (CM)		27.2	27.3	VERIFIED
WEDGE NUMBER		1	1	VERIFIED
ACCESSORY NUMBER		0	0	VERIFIED
DATE : 84-OCT-26	SYSTEM : BEAM READY		OP. MODE : TREAT	AUTO
TIME : 12:55: 8	TREAT : TREAT PAUSE		X-RAY	173777
OPR ID : T25V02-R03	REASON : OPERATOR		COMMAND:	

Examples

Application that can cause harm (a risk):

- Airbag exploding when infant is sitting in front seat

Need to assess the risk

- Infant getting injured – “not good at all”

Find a mitigation strategy, e.g. a safety function:

- Detecting infant in front seat and disabling airbag
  - a) sensor delivers signal to
  - b) software/hardware controlling an
  - c) actuator (disabler)

Functional Safety is then:

- An infant in front seat is not exposed to an unacceptable (unreasonable) risk

Question:
How to measure and agree on the measures?

Examples

Warning

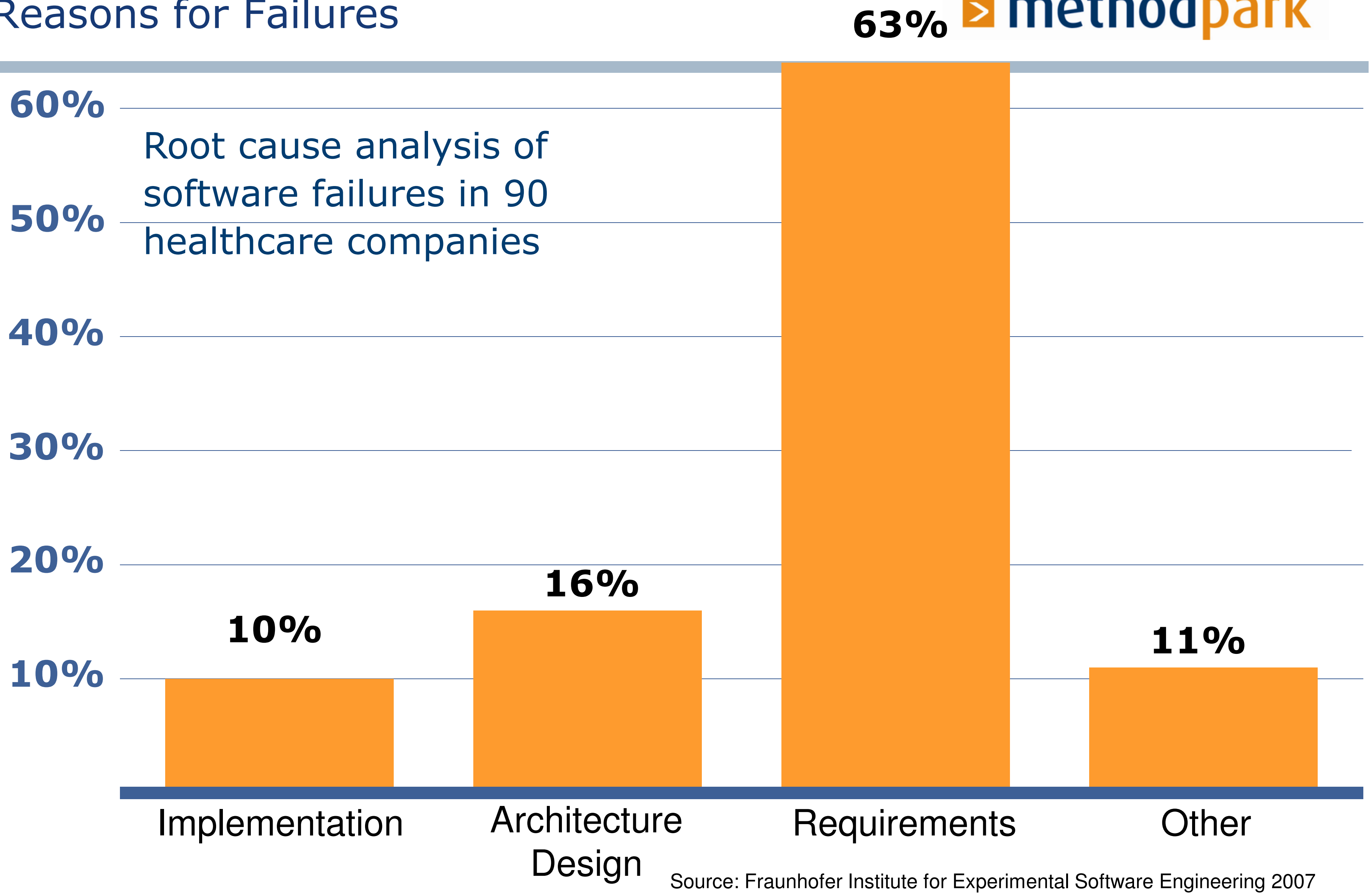
Your Brake Function is temporarily unavailable, please STOP the Car immediately!

OK

CANCEL

Question:
Do we dare putting software in direct control of people’s life?

Reasons for Failures



Source: Fraunhofer Institute for Experimental Software Engineering 2007

Complexity

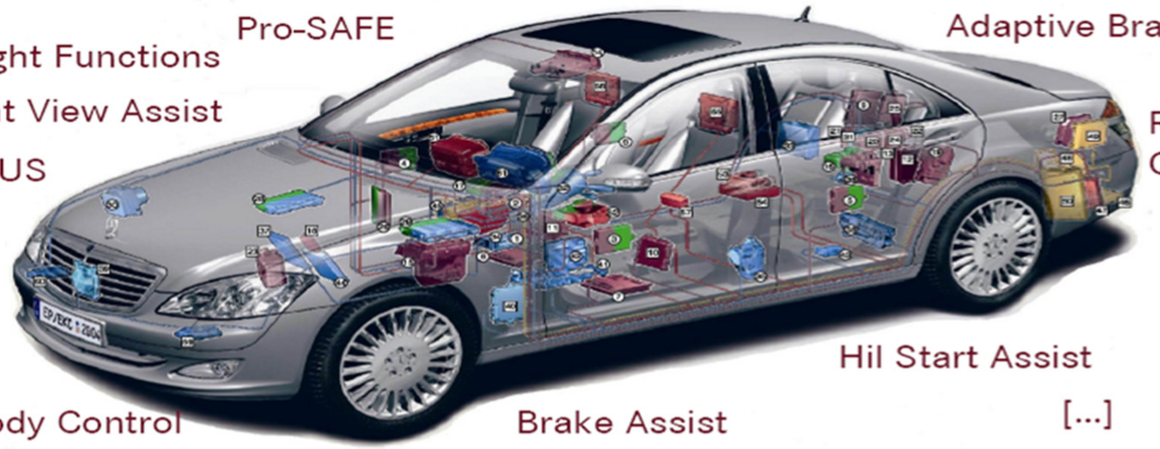
DAIMLER

Functional Safety

Current Situation

Trends in Automotive Electric/Electronics (E/E)

- Increasing functionality and complexity of software-based car functions
- Increasing risks from systematic faults and random hardware faults
- Most of the new car functions are safety-related



Source: © Courtesy of Daimler; Presentation given at Automotive Electronics and Electrical Systems Forum 2008, May 6, 2008, Stuttgart, Germany

Extract from German law

§ 823 Abs. 1 BGB:

„Anyone who injures intentionally or negligently the life, body, health, liberty, property or any other right of another person, is obliged to compensate for the resulting damages.“

§ 1 Abs. 1 ProdhaftG:

„If someone is killed, his body or health injured or an item damaged by a defect in a product, the manufacturer of the product is obliged to replace the resulting damages.“

Contents

Definitions

Safety

… is the absence of unacceptable (unreasonable) risks that can cause harm achieved through a planned strategy

Functional Safety

… is part of the overall safety that depends on a system or equipment operating correctly in response to its inputs.

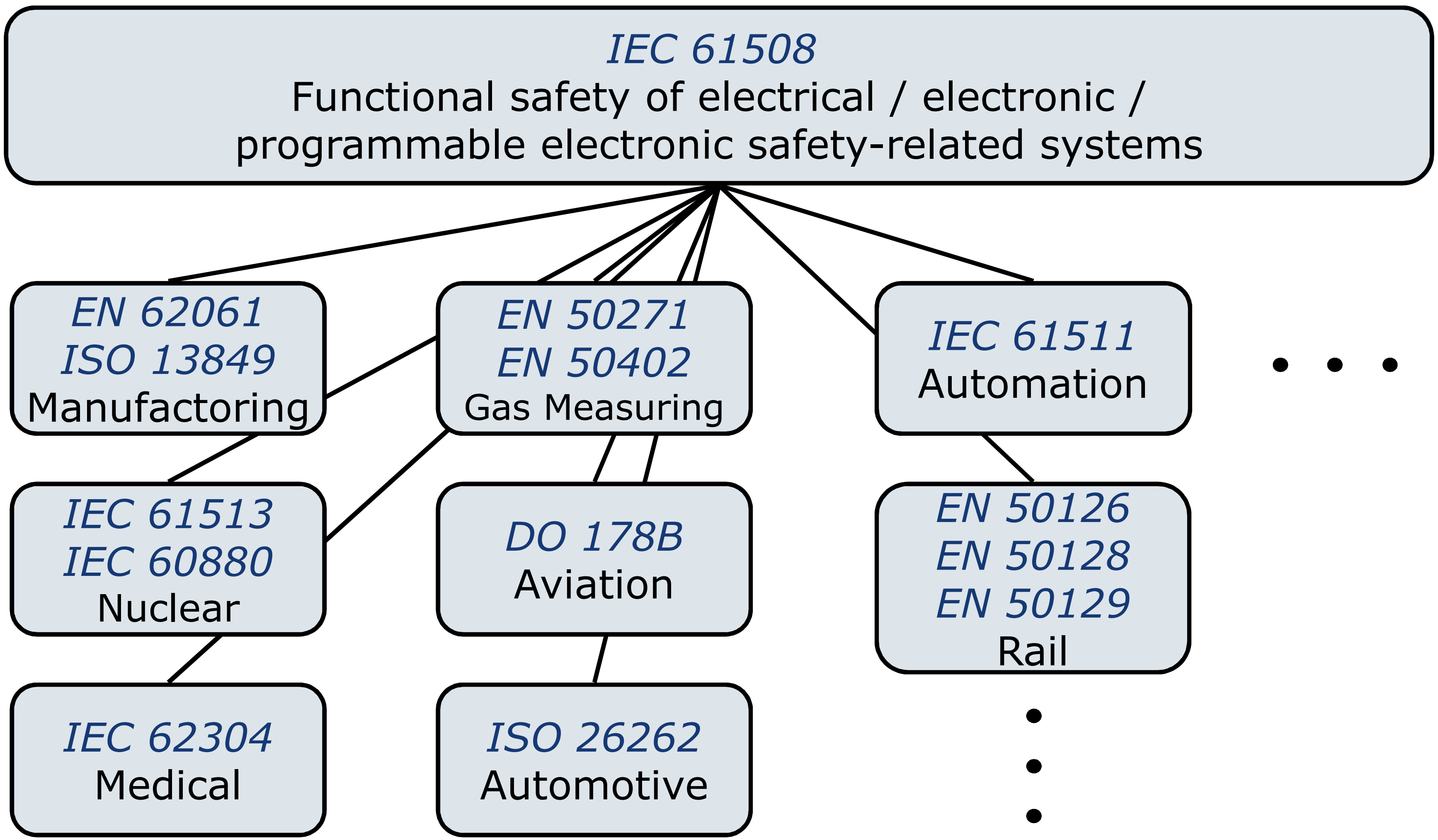
… is achieved when every specified safety function is carried out and the level of performance required of each safety function is met

… is **not** to provide the perfect car, but a safe car.

Functional Safety Management

… is the management (plan, do, act, check) of all activities necessary to reach functional safety.

Existing Standards



Scope of ISO 26262

Why not using IEC 61508?

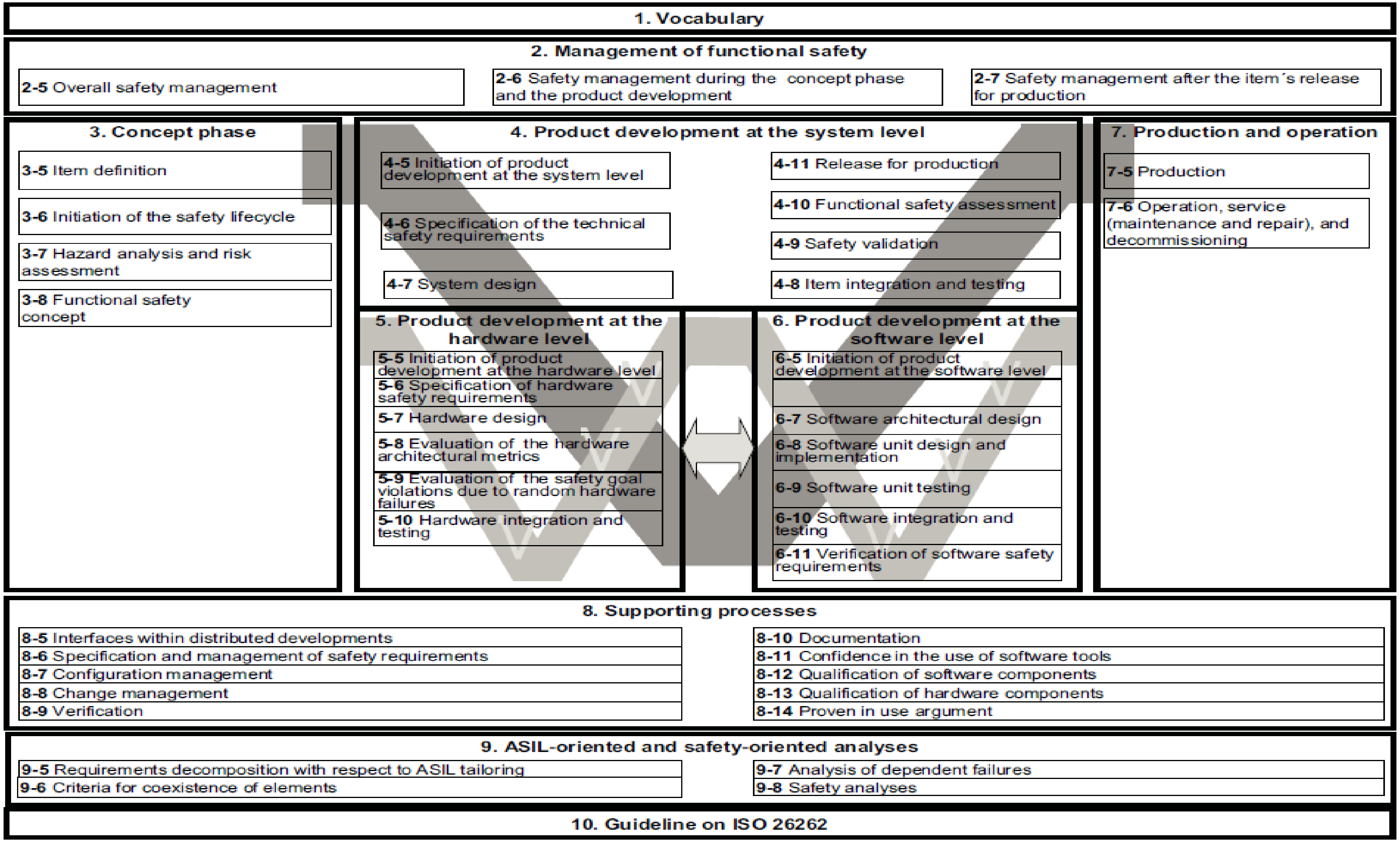
Lessons learnt from application of IEC 61508 in automotive industry:

- Not adapted to real-time and integrated embedded systems
- Not adapted to automotive development and life cycles
- No requirements for manufacturer / supplier relationship
- No ‘consumer-goods’ orientation
- …

Companies had to solve these issues themselves until introduction of

ISO 26262

Structure of ISO 26262



Source: ISO 26262:2011

ISO 15504 & Automotive SPICE®

methodpark

Process Category
Process Group
Process

A = Automotive SPICE®
H = HIS-Scope

Primary Life Cycle Processes

Acquisition Process Group (ACQ)

		ACQ.1 Acquisition preparation
		ACQ.2 Supplier selection
A		ACQ.3 Contract agreement
A	H	ACQ.4 Supplier monitoring
		ACQ.5 Customer acceptance
A		ACQ.11 Technical requirements
A		ACQ.12 Legal and administrative requirements
A		ACQ.13 Project requirements
A		ACQ.14 Request for proposals
A		ACQ.15 Supplier qualification

Supply Process Group (SPL)

A		SPL.1 Supplier tendering
A		SPL.2 Product release
		SPL.3 Product acceptance support

Engineering Process Group (ENG)

A		ENG.1 Requirements elicitation
A	H	ENG.2 System requirements analysis
A	H	ENG.3 System architectural design
A	H	ENG.4 Software requirements analysis
A	H	ENG.5 Software design
A	H	ENG.6 Software construction
A	H	ENG.7 Software integration
A	H	ENG.8 Software testing
A	H	ENG.9 System integration
A	H	ENG.10 System testing
		ENG.11 Software installation
		ENG.12 Software and system maintenance

Operation Process Group (OPE)

		OPE.1 Operational use
		OPE.2 Customer support

Supporting Life Cycle Processes

Support Process Group (SUP)

A	H	SUP.1 Quality assurance
A		SUP.2 Verification
		SUP.3 Validation
A		SUP.4 Joint review
		SUP.5 Audit
		SUP.6 Product evaluation
A		SUP.7 Documentation
A	H	SUP.8 Configuration management
A	H	SUP.9 Problem resolution management
A	H	SUP.10 Change request management

Organizational Life Cycle Processes

Management Process Group (MAN)

		MAN.1 Organizational alignment
		MAN.2 Organizational management
A	H	MAN.3 Project management
		MAN.4 Quality management
A		MAN.5 Risk management
A		MAN.6 Measurement

Process Improvement Process Group (PIM)

		PIM.1 Process establishment
		PIM.2 Process assessment
A		PIM.3 Process improvement

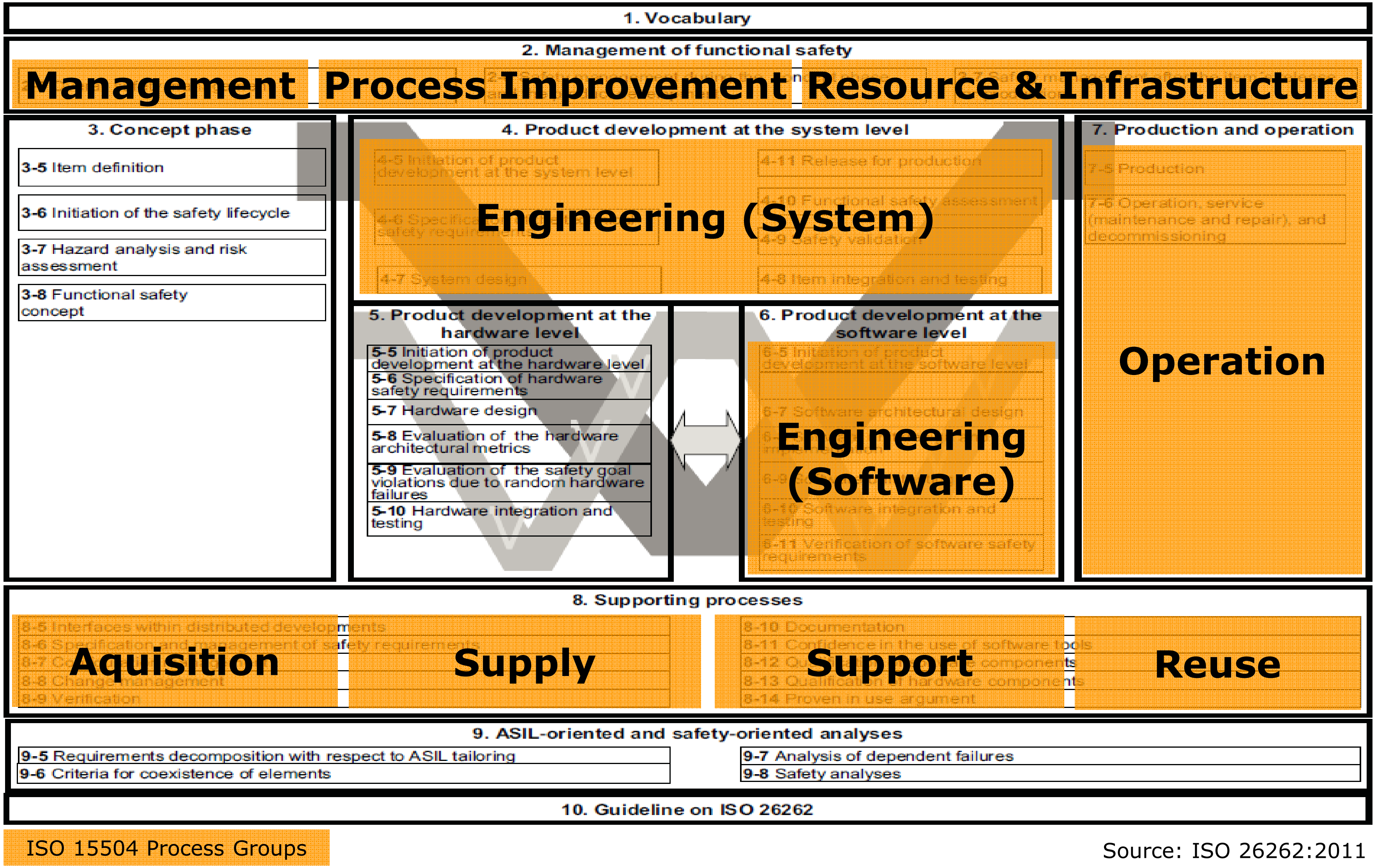
Resource & Infrastructure Process Group (RIN)

		RIN.1 Human resource management
		RIN.2 Training
		RIN.3 Knowledge management
		RIN.4 Infrastructure

Reuse Process Group (REU)

		REU.1 Asset management
A		REU.2 Reuse program management
		REU.3 Domain engineering

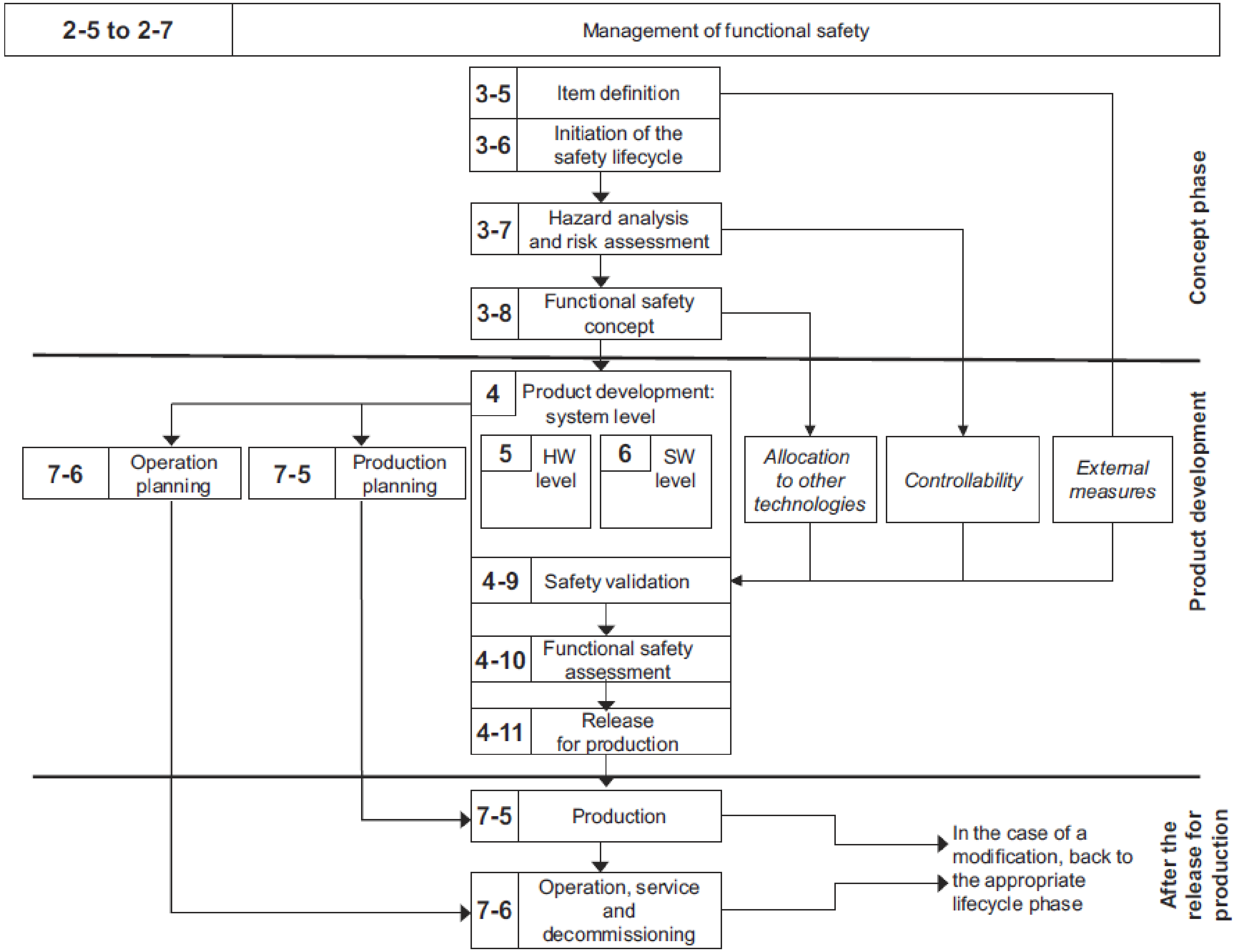
Structure of ISO 26262



Source: ISO 26262:2011

Safety Lifecycle Overview

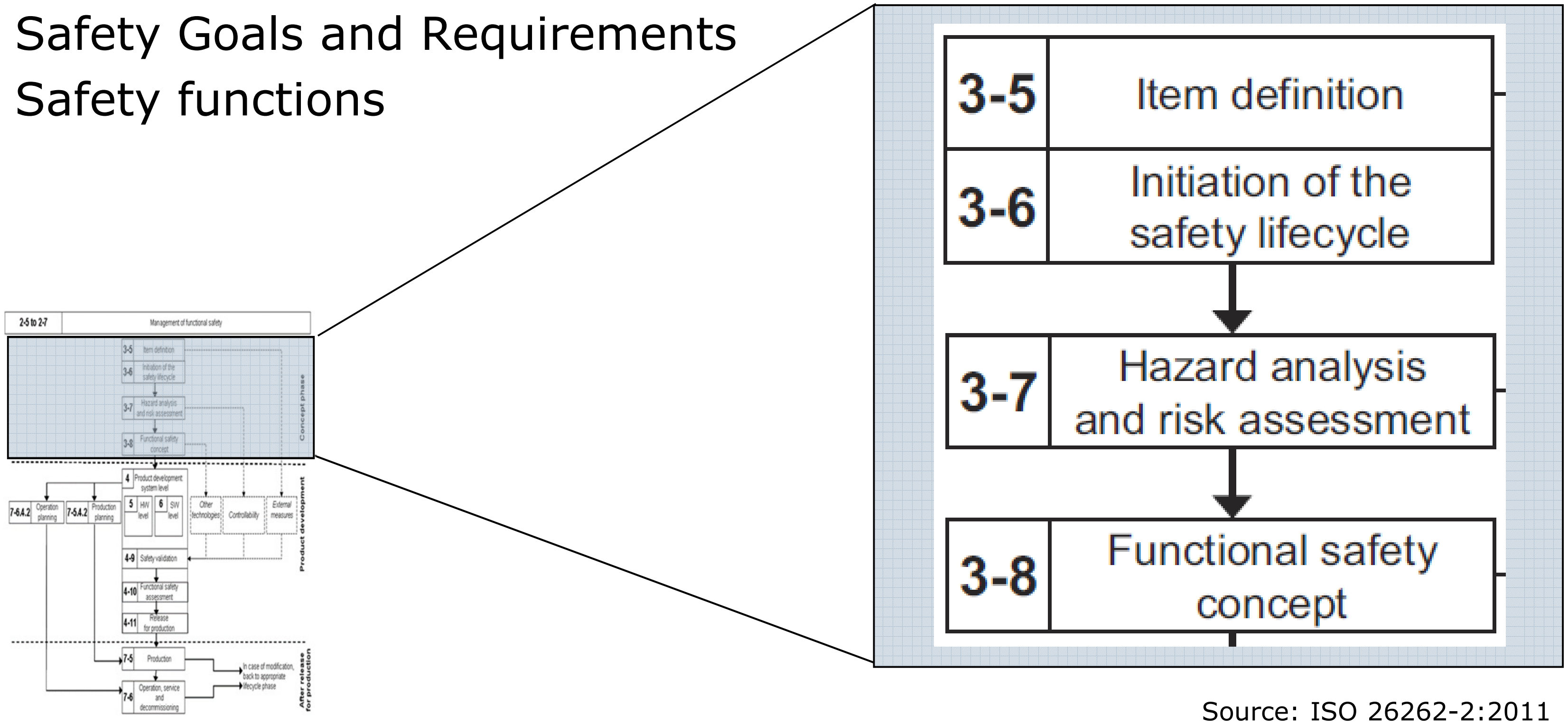
- Concept
- Development
- Production



Source: ISO 26262-2:2011

Concept Phase

- Focus on entire system
- Risks
- Safety Goals and Requirements
- Safety functions

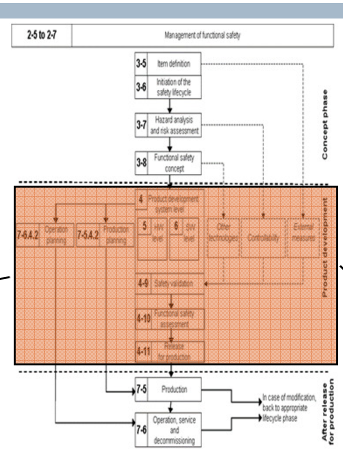


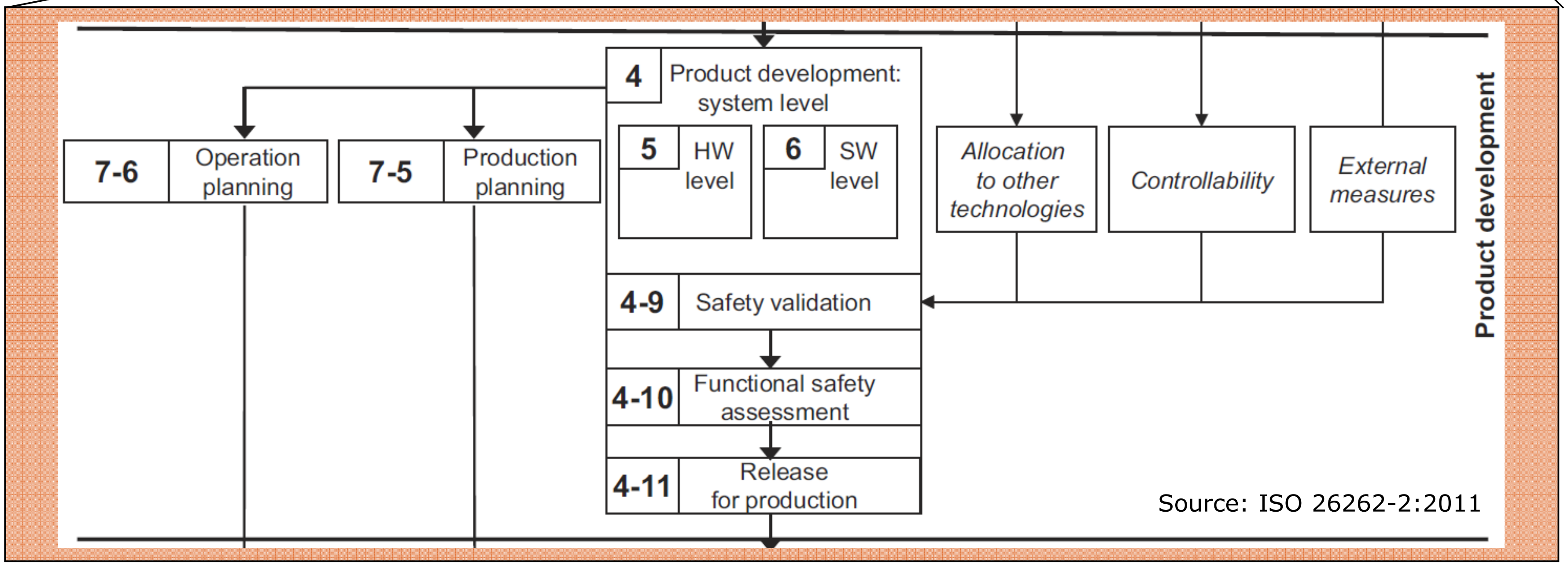
Source: ISO 26262-2:2011

methodpark

Product Development

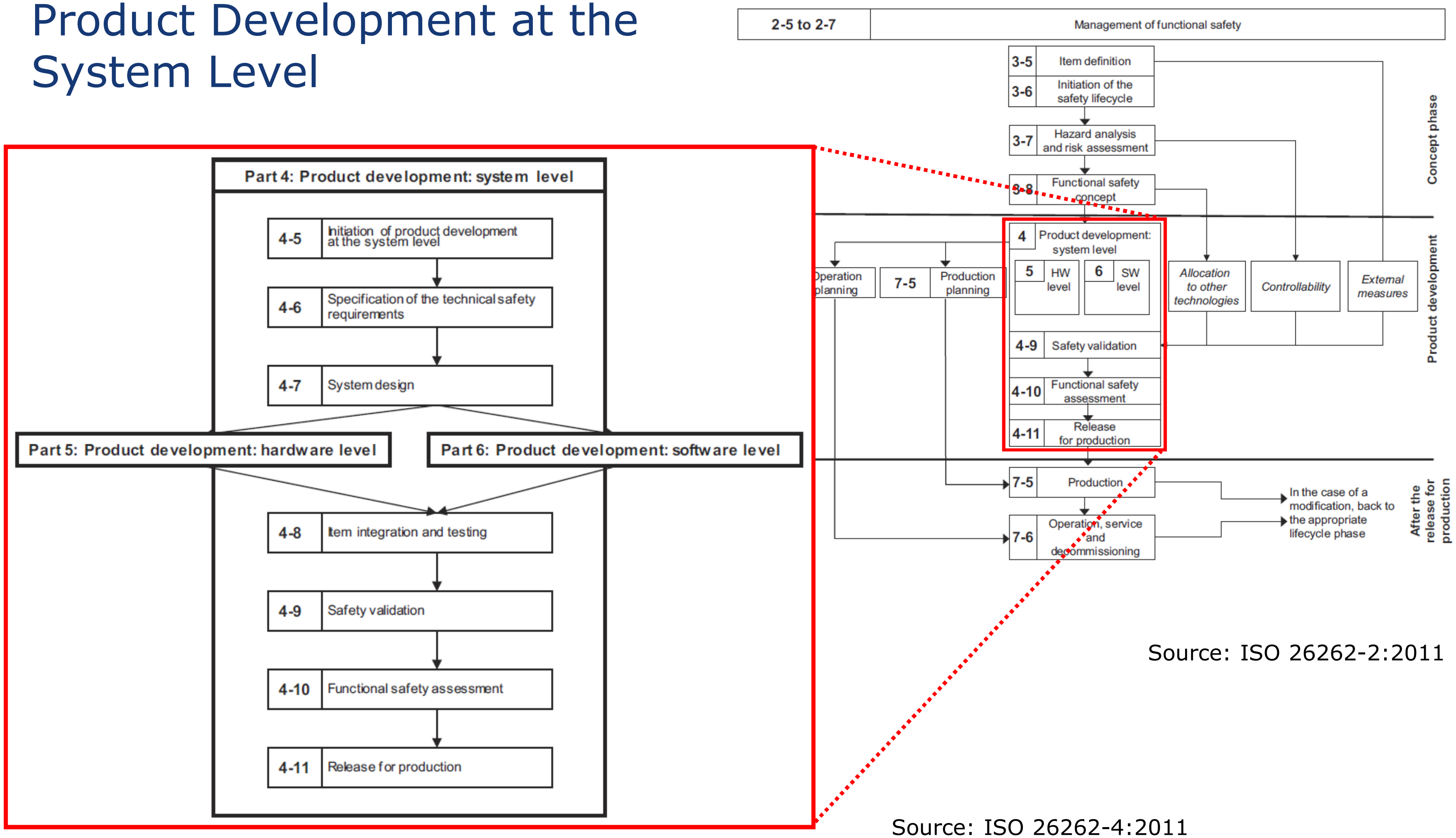
- System, Hardware and Software
- Safety validation and assessment
- Production and Operation (Planning)





Source: ISO 26262-2:2011

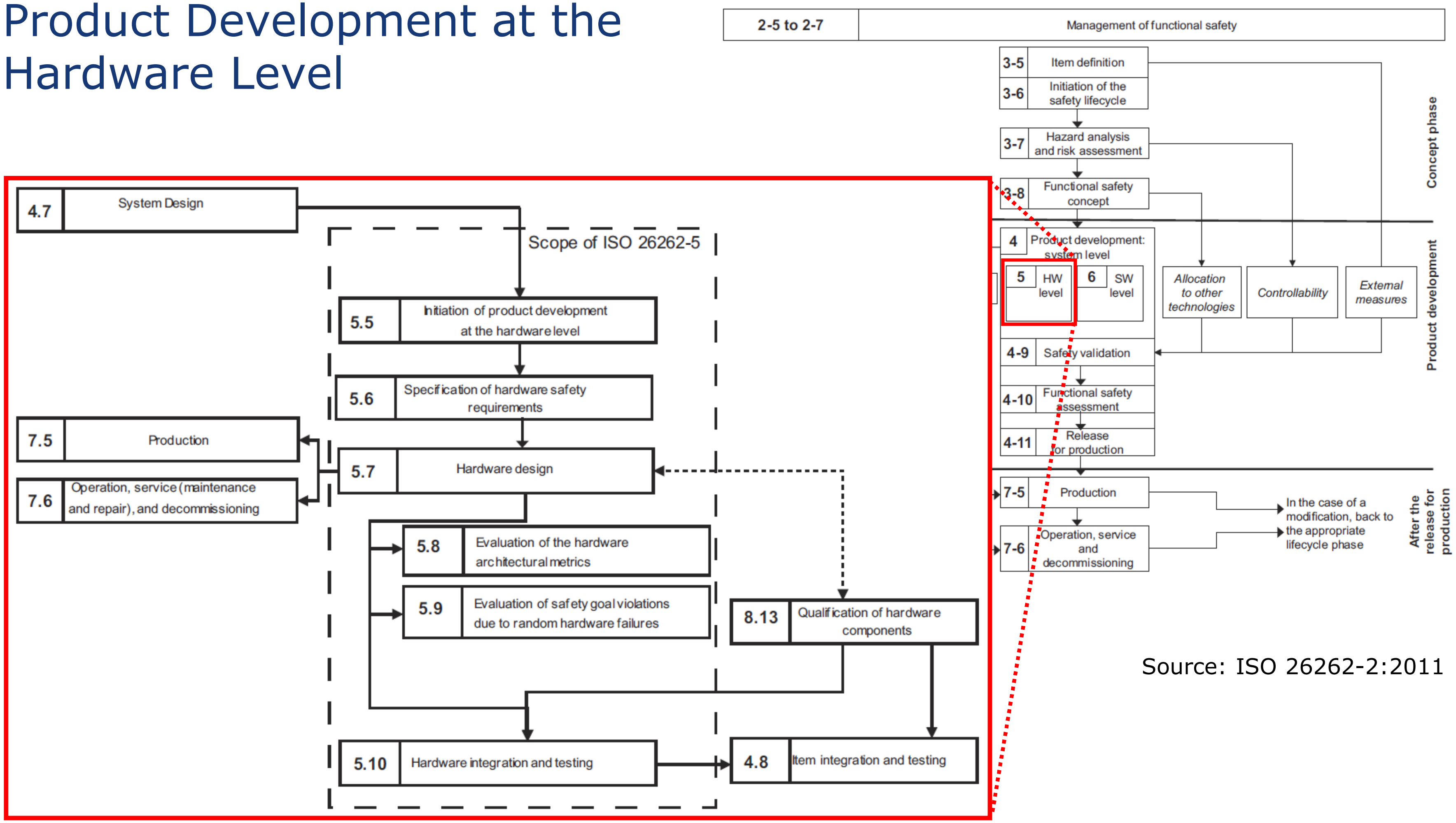
Product Development at the System Level



Source: ISO 26262-2:2011

Source: ISO 26262-4:2011

Product Development at the Hardware Level

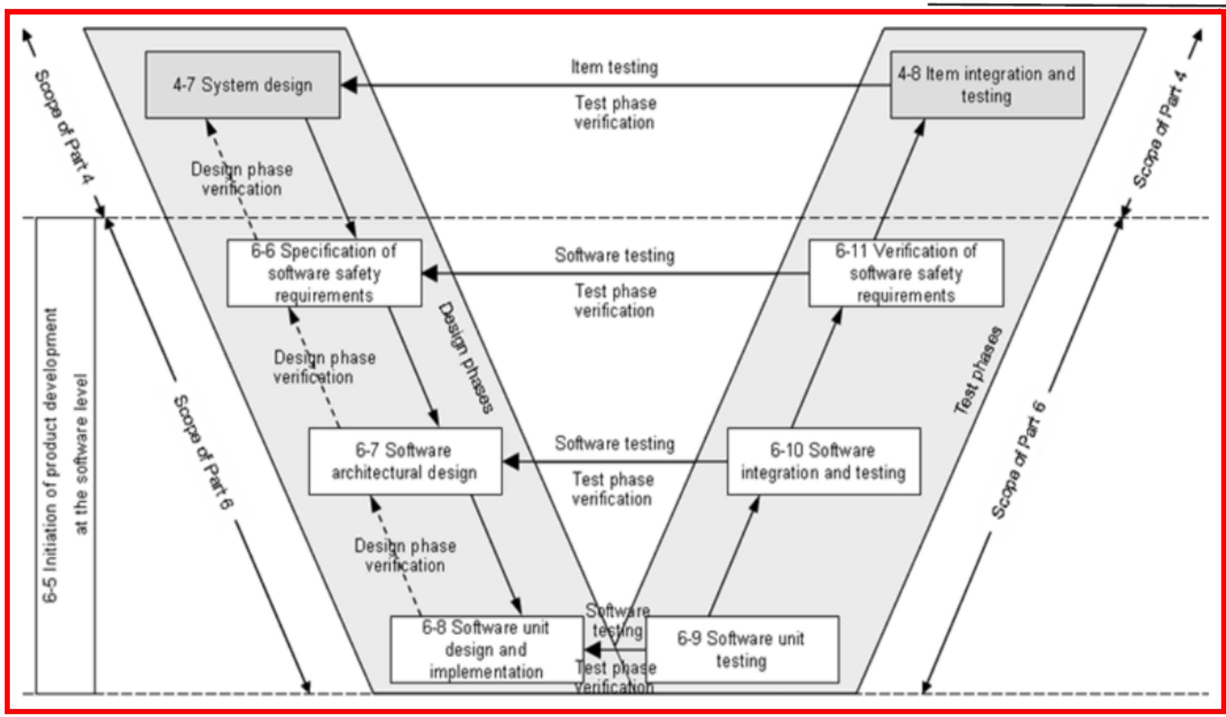


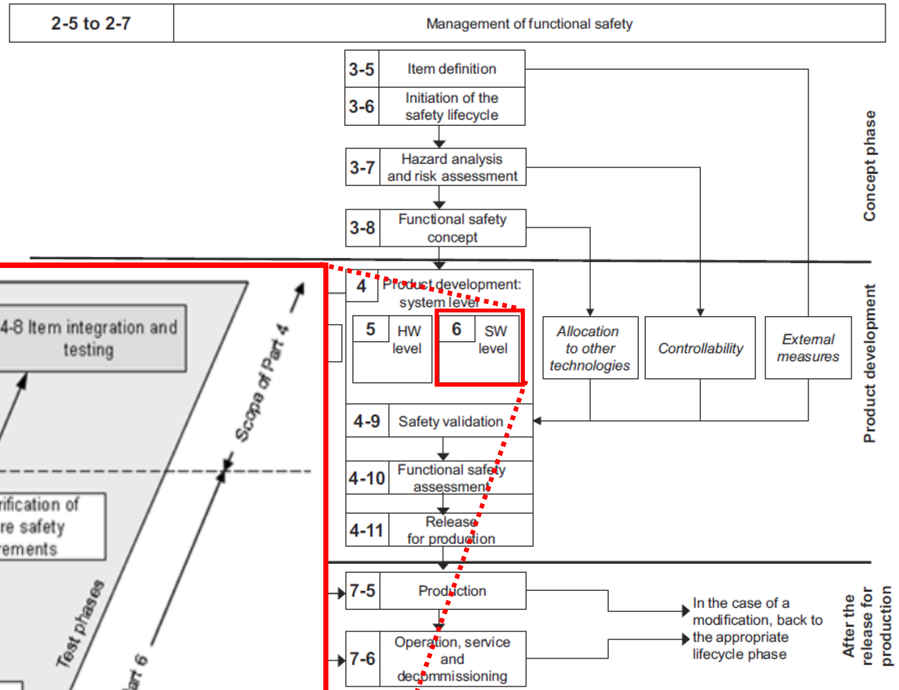
Source: ISO 26262-2:2011

Source: ISO 26262-5:2011

Product Development

Product Development at the Software Level





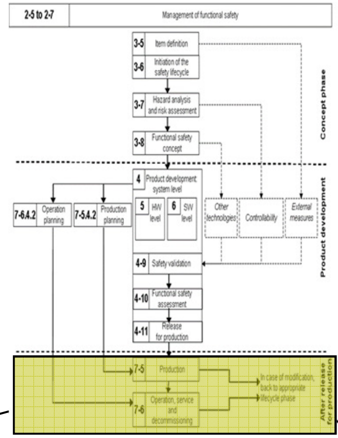
Source: ISO 26262-2:2011

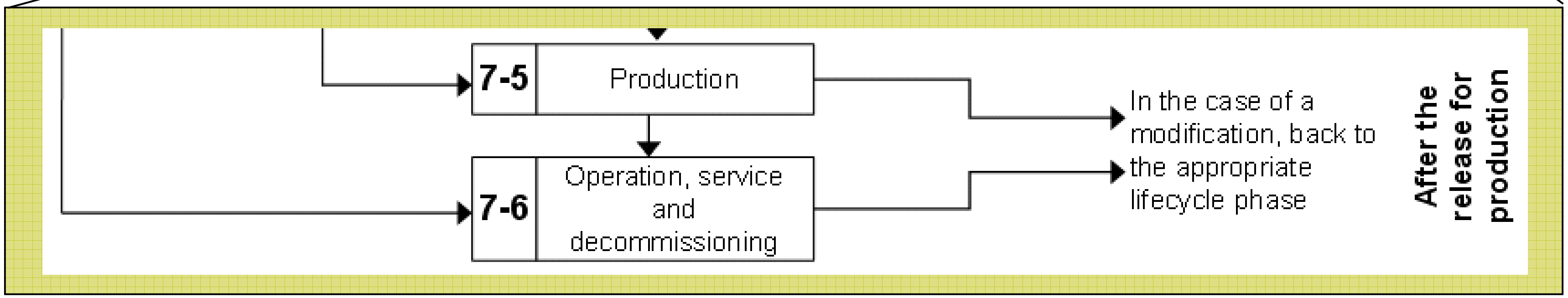
Source: ISO 26262-6:2011

Safety Lifecycle Overview

After Release for Production

- Production
- Installation
- Operation
- Maintenance and reparation
- Disassembly



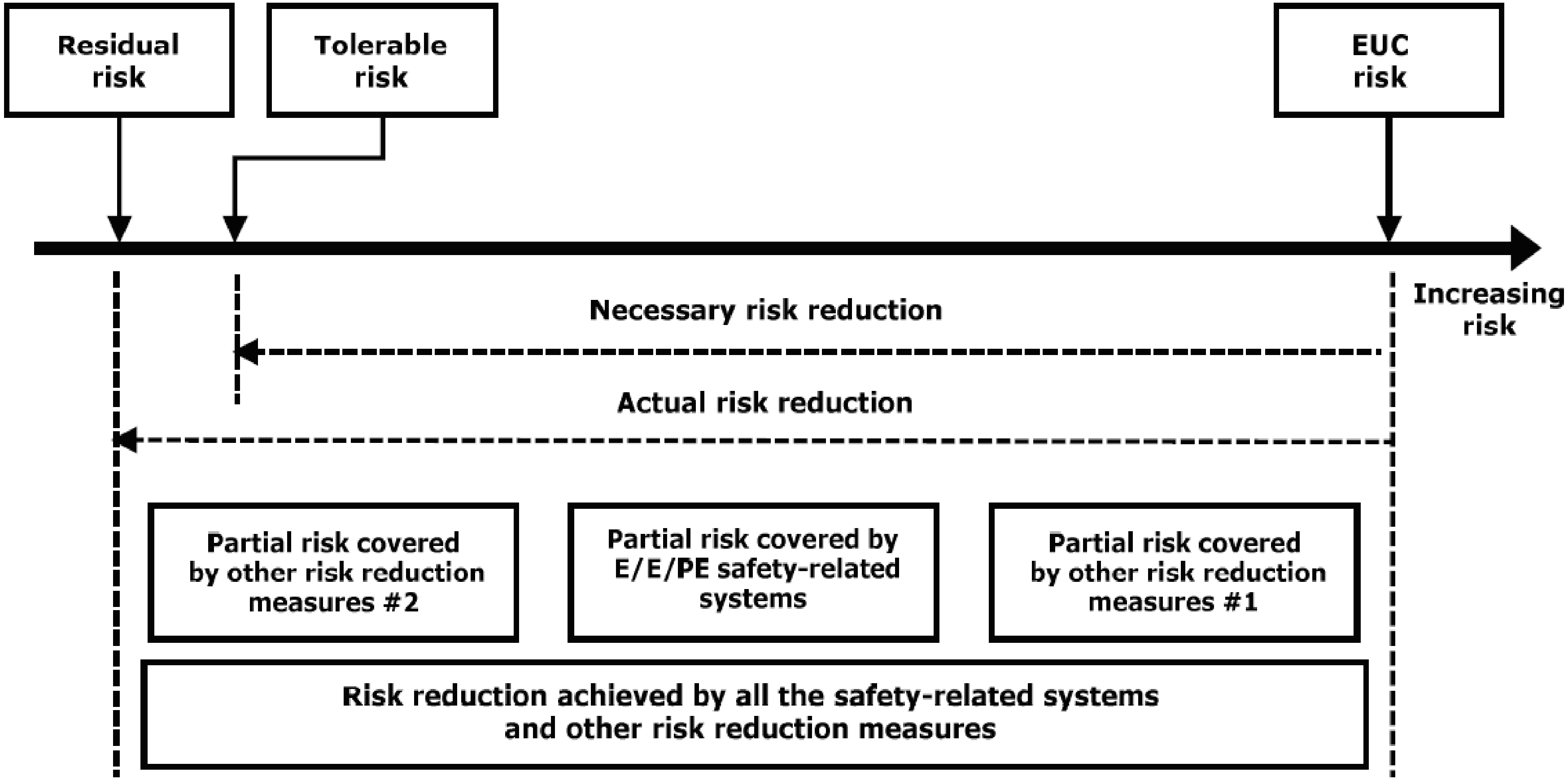


Source: ISO 26262-2:2011

Contents

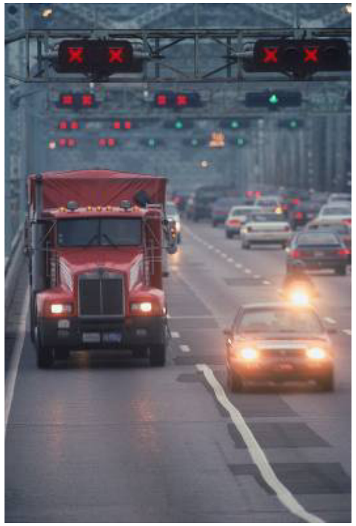
Hazard Analysis and Risk Assessment

Risk reduction to an acceptable level



Source: IEC 61508-5:2010

Hazard Analysis and Risk Assessment



Situation analysis and hazard identification

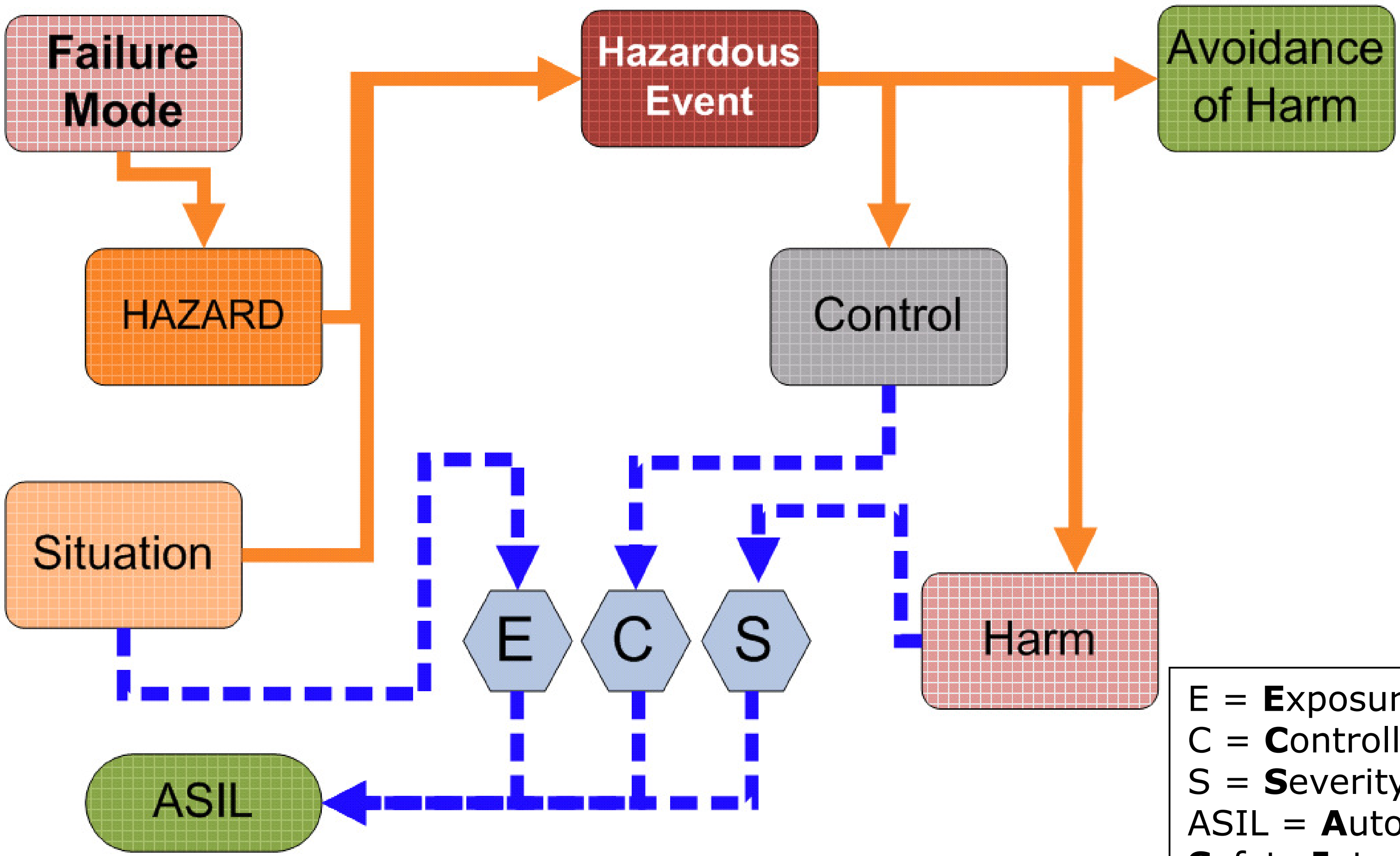
- List of driving and operating situations
  → Estimation of the probability of **E**xposure
- Detailing failure modes leading to hazards in specific situations
  → Estimation of **C**ontrollability
- Evaluating consequences of the hazards
  → Estimation of potential **S**everity

→ Respect only the plain item (do not take risk-reducing measures into account!)

→ Involve persons with good knowledge and domain experience

Hazard Analysis and Risk Assessment

Associations of the central concepts



E = **E**xposure
C = **C**ontrollability
S = **S**everity
ASIL = **A**utomotive **S**afety **I**ntegrity **L**evel

Exposure

State of being in an operational situation that can be hazardous if coincident with the failure mode under analysis

Class	E0	E1	E2	E3	E4
Description	Incre-dible	Very low probability	Low probability	Medium probability	High probability
Time		Not specified	Less than 1% of average operating time	1% - 10% of average operating time	> 10% of average operating time
Event		Situations that occur less often than once a year for the great majority of drivers	Situations that occur a few times a year for the great majority of drivers	Situations that occur once a month or more often for an average driver	All situations that occur during almost every drive on average

Source: ISO 26262-3:2011

Controllability

Avoidance of the specified harm or damage through the timely reactions of the persons involved

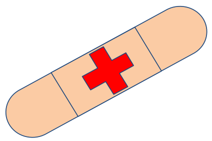
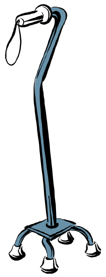
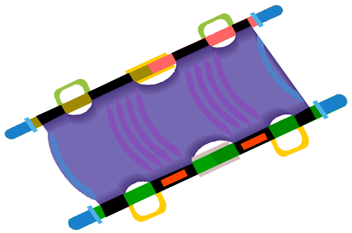
Class	C0	C1	C2	C3
Description	Controllable in general	Simply controllable	Normally controllable	Difficult to control or uncontrollable
Definition	Controllable in general	99% or more of all drivers or other traffic participants are usually able to avoid a specific harm.	90% or more of all drivers or other traffic participants are usually able to avoid a specific harm.	Less than 90% of all drivers or other traffic participants are usually able, or barely able, to avoid a specific harm.

Source: ISO 26262-3:2011

Hazard Analysis and Risk Assessment

Severity

Measure of the extent of harm to an individual in a specific situation

Class	S0	S1	S2	S3
Description	No injuries	Light and moderate injuries	Severe and life-threatening injuries (survival probable)	Life-threatening injuries (survival uncertain), fatal injuries

Source: ISO 26262-3:2011

Hazard Analysis and Risk Assessment

Combinations of Severity, Exposure and Controllability result in the applicable ASIL.

The ASIL's influence the development process of the items.

QM = Quality Management
No specific ISO 26262 requirement has to be observed

If S0 or E0 or C0 is set, no ASIL is required (QM).

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

Source: ISO 26262-3:2011

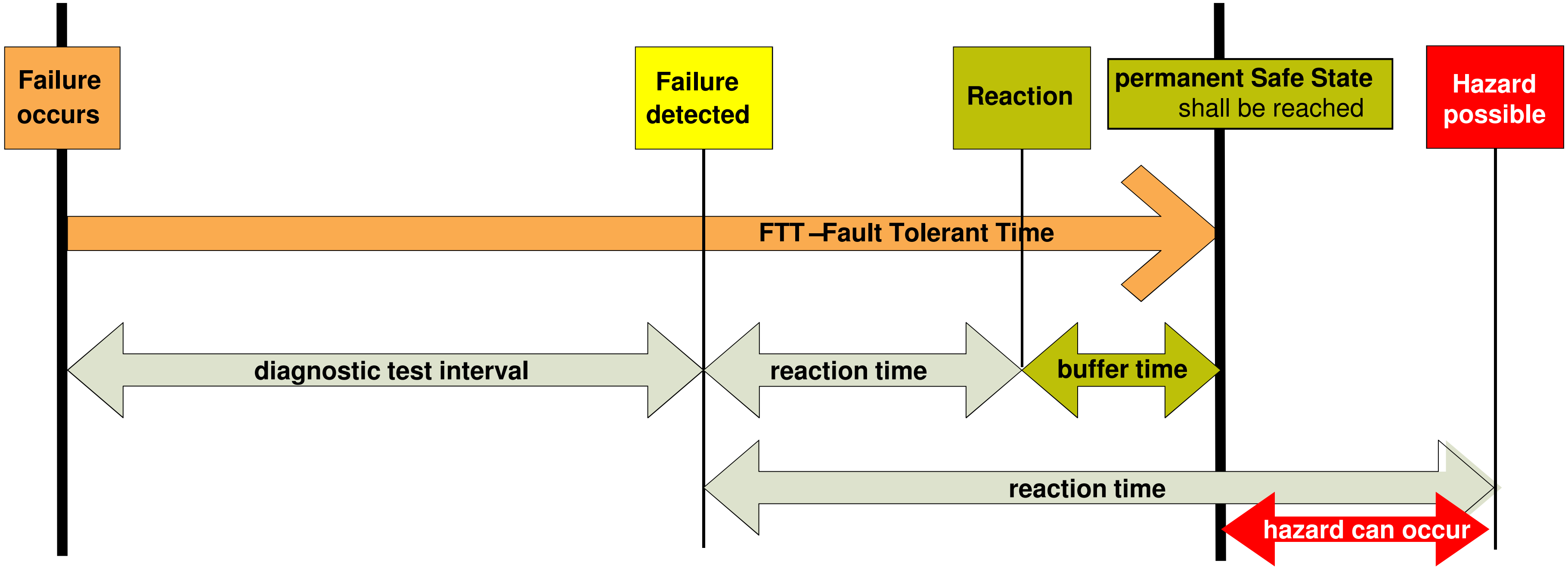
Safety Goals

- top-level safety requirements as a result of the hazard analysis and risk assessment
- assigned to each identified hazard rated with an ASIL A-D
- lead to item characteristics needed to avert hazards or to reduce risks associated with the hazards to an acceptable level
- are assigned to a safe state that must be reached in case of appearance
- indicate the maximum fault tolerance time within the safe state must be reached

fault tolerance time = fault recognition time + fault reaction time

Safe State – Operating mode of an item without an unreasonable level of risk

- Example: intended operating mode, degraded operating mode or switched-off mode



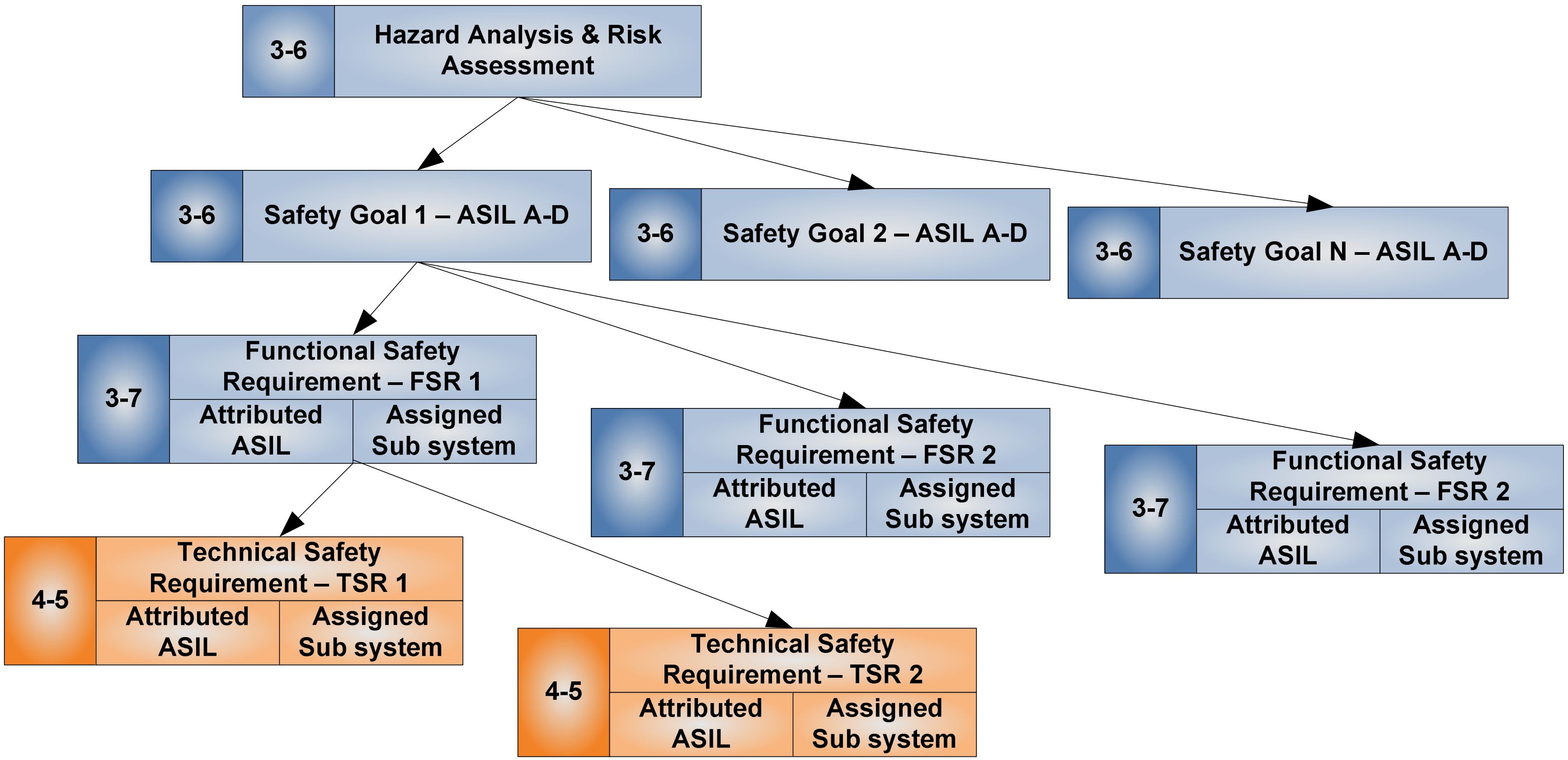
Example for Safety Goals: Park Brake System

ID	Safety Goal	ASIL	Safe State	FTT
G1	Avoidance of unintended maximum brake force build up at one or several wheels during drive and in all environmental conditions	D	Brake released	50 ms
G2	Guarantee the specified parking brake function in use case situation "parking on slope" in all environmental conditions	A	Brake closed	500 ms
G3	Avoidance of unintended release of the parking brake in use case situation "parking on slope" in all environmental conditions	C	Brake closed	500 ms

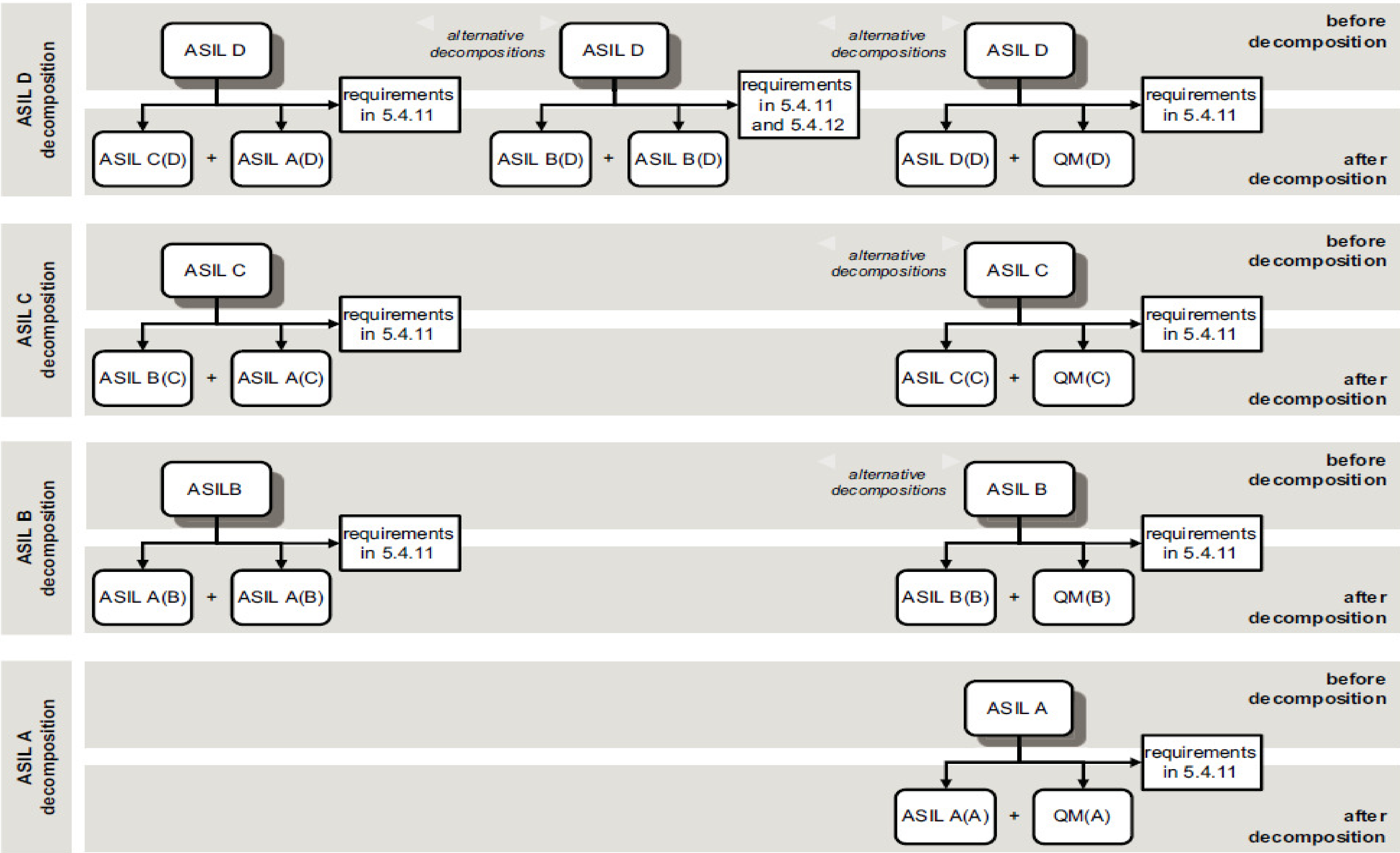
Contents

Functional Safety Concept

Safety Goals and Functional Safety Requirements

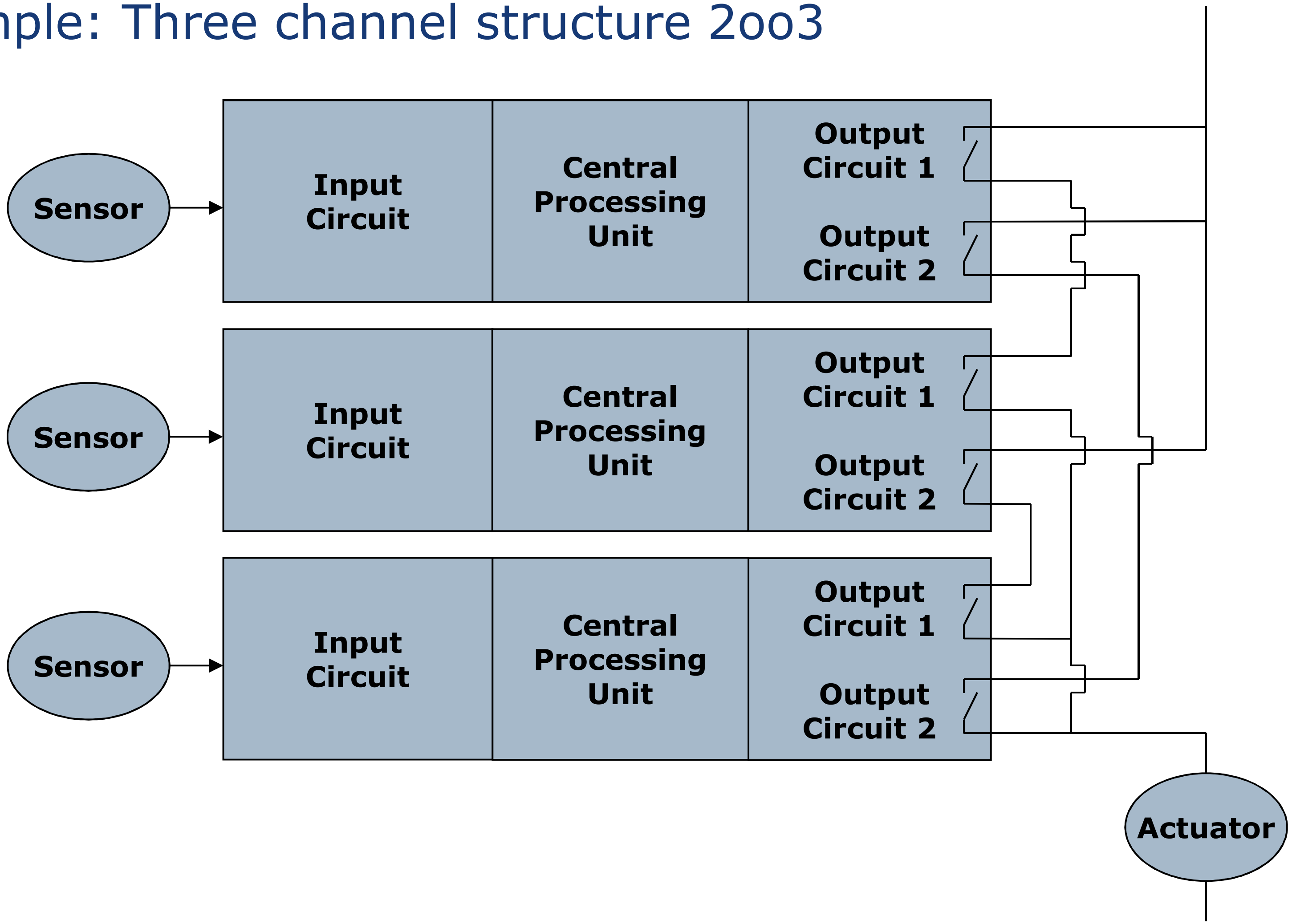


ASIL Decomposition



Source: ISO 26262-9:2011

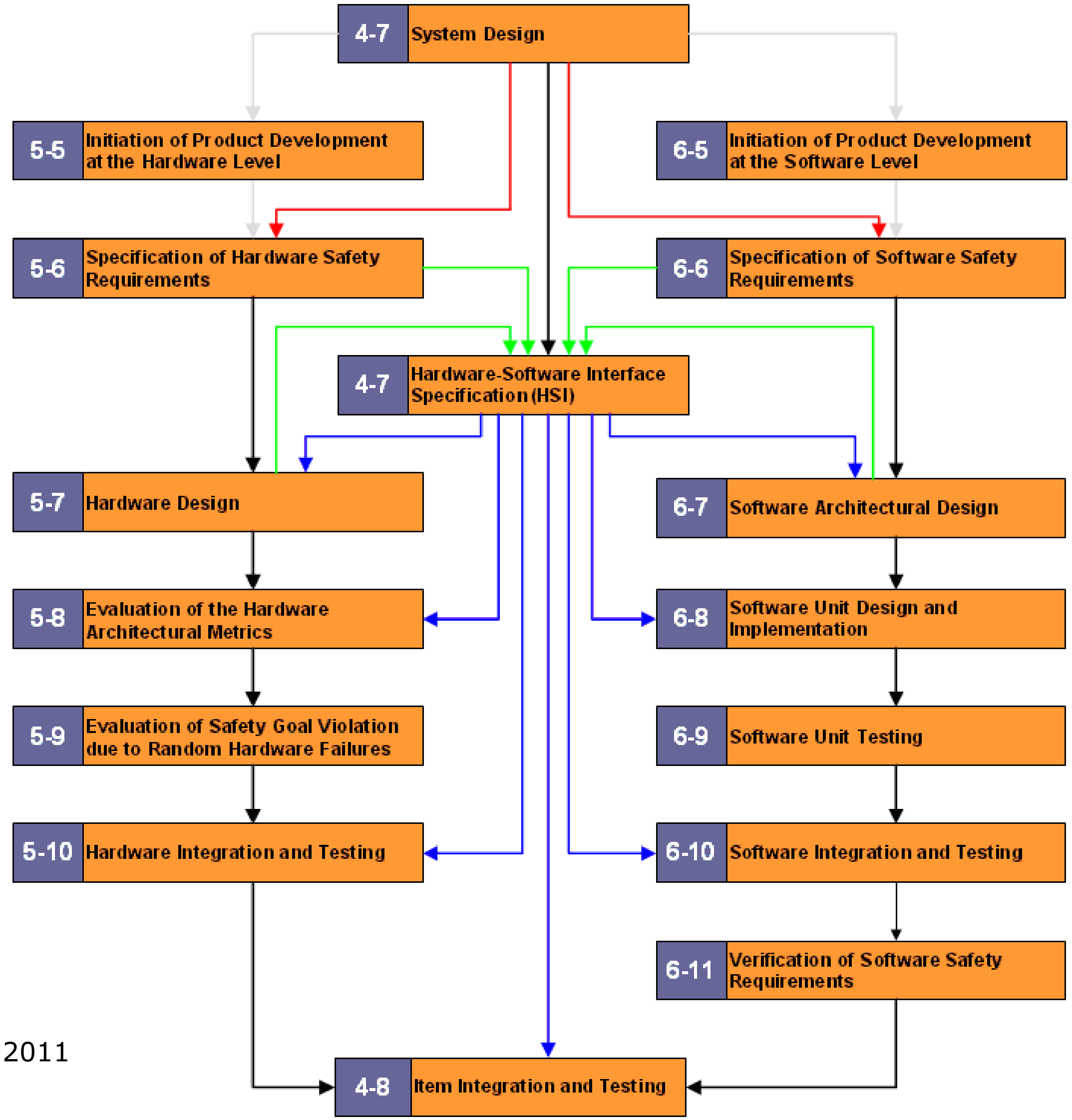
Example: Three channel structure 2oo3



Contents

Product Development at Hardware & Software Level

Important part:
Hardware-Software Interface Specification (HSI)

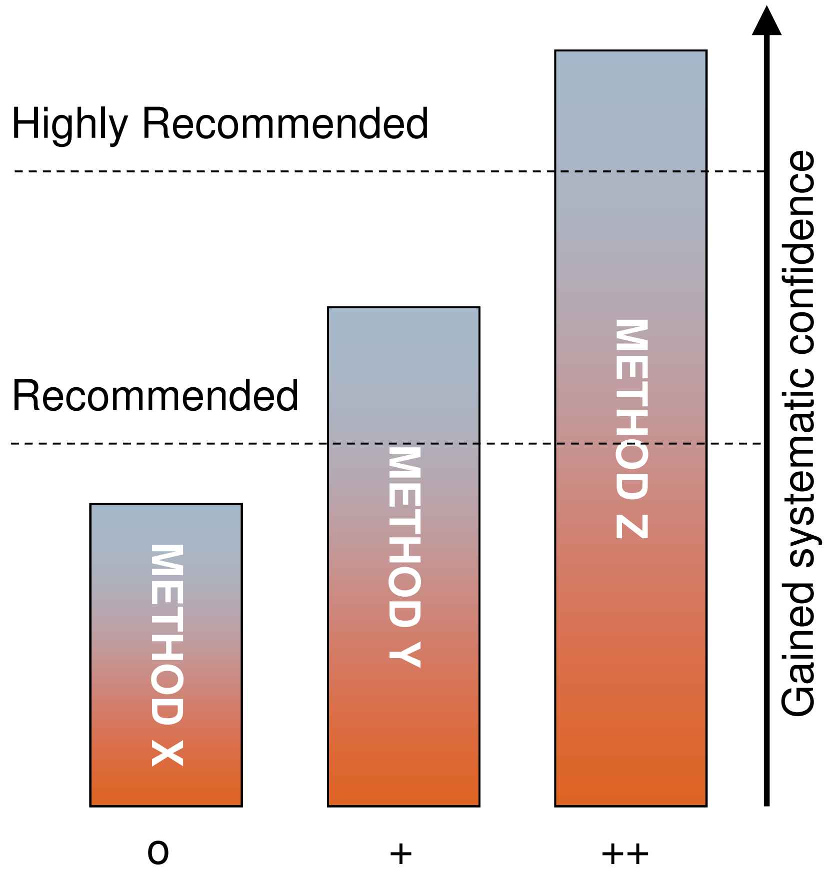


Source: ISO 26262-4:2011

How to understand the standard tables

For each method, the degree of recommendation to use corresponding methods depends on the ASIL and is categorized as follows:

"++"	The method is highly recommended for this ASIL
"+"	The method is recommended for this ASIL
"o"	The method has no recommendation for or against its usage for this ASIL



Initiation of Product Development at the Software Level

Topics to be covered by modeling and coding guidelines

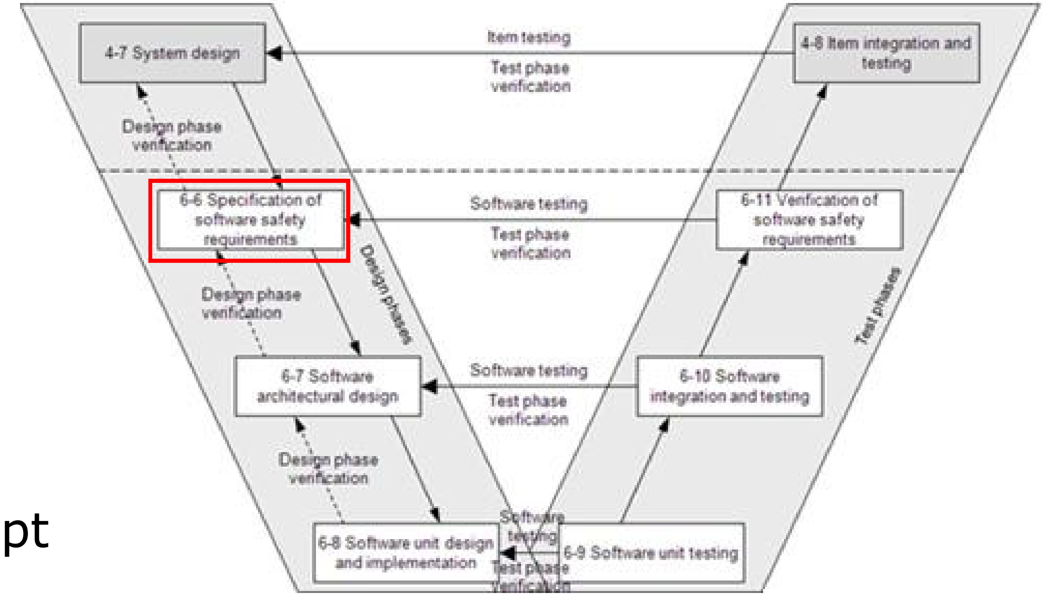
	Topics	ASIL A	ASIL B	ASIL C	ASIL D
1a	Enforcement of low complexity	++	++	++	++
1b	Use of language subsets	++	++	++	++
1c	Enforcement of strong typing	++	++	++	++
1d	Use of defensive implementation techniques	o	+	++	++
1e	Use of established design principles	+	+	+	++
1f	Use of unambiguous graphical representation	+	++	++	++
1g	Use of style guides	+	++	++	++
1h	Use of naming conventions	++	++	++	++

Source: ISO 26262-6:2011

Specification of Software Safety Requirements

Goals

- Derive Software Safety Requirements from and ensure consistency with
  - System Design
  - Technical Safety Concept
- Detail the hardware-software interface requirements



Source: ISO 26262-6:2011

Specification of Software Safety Requirements

Methods for specifying Safety Requirements

- Safety requirements shall be specified by an appropriate combination of natural language and methods listed in the table
- For higher level safety requirements (e.g. functional and technical safety requirements) natural language is more appropriate while for lower level safety requirements (e.g. software and hardware safety requirements) notations listed in the table are more appropriate

Methods		ASIL			
		A	B	C	D
1a	Informal notations for requirements specification	++	++	+	+
1b	Semi-formal notations for requirements specification	+	+	++	++
1c	Formal notations for requirements specification	+	+	+	+

Source: ISO 26262-8:2011

Specification of Software Safety Requirements

Methods for the verification of Safety Requirements

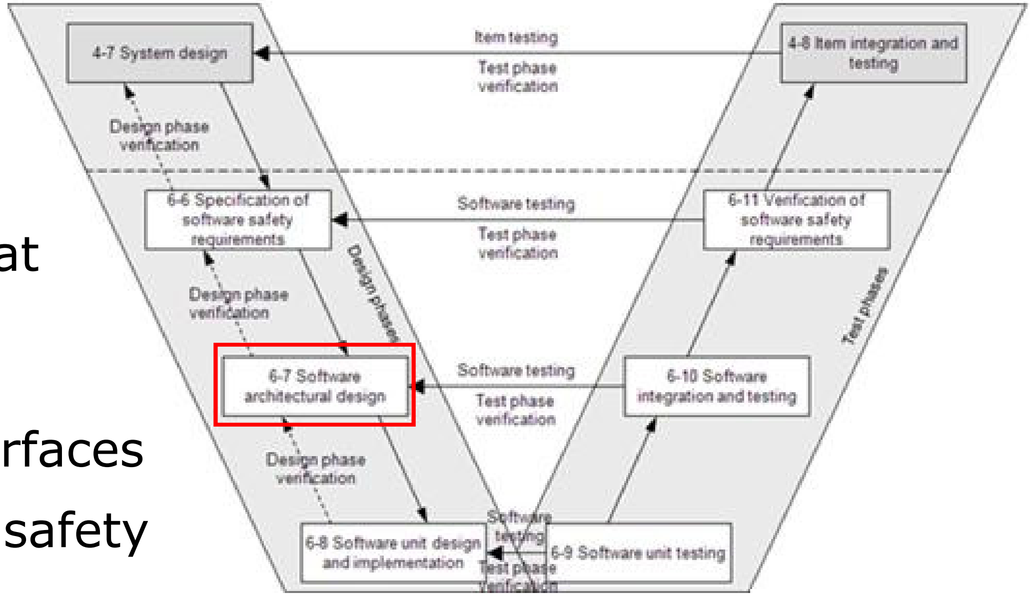
	Methods	ASIL A	ASIL B	ASIL C	ASIL D
1a	Verification by walk-through	++	+	o	o
1b	Verification by inspection	+	++	++	++
1c	Semi-formal verification (e.g. executable models)	+	+	++	++
1d	Formal verification	o	+	+	+

Source: ISO 26262-8:2011

Software Architectural Design

Goals

- Develop an Architecture that implements the Software Safety Requirements
  - Static and dynamic interfaces
  - Safety-related and non safety related requirements



Source: ISO 26262-6:2011

- Verify the Software Architecture
  - Compliance with the requirements
  - Compatibility with hardware
  - Respect of design principles and standards

Principles for software architectural design

Methods		ASIL			
		A	B	C	D
1a	Hierarchical structure of software components	++	++	++	++
1b	Restricted size of software components	++	++	++	++
1c	Restricted size of interfaces	+	+	+	+
1d	High cohesion within each software component	+	++	++	++
1e	Restricted coupling between software components	+	++	++	++
1f	Appropriate scheduling properties	++	++	++	++
1g	Restricted use of interrupts	+	+	+	++

Source: ISO 26262-6:2011

Software Architectural Design

methodpark

Based on the results of the safety analysis the mechanisms for error detection and error handling shall be applied

	Methods	ASIL A	ASIL B	ASIL C	ASIL D
1a	Range checks of input and output data	++	++	++	++
1b	Plausibility check	+	+	+	++
1c	Detection of data errors	+	+	+	+
1d	External monitoring facility	o	+	+	++
1e	Control flow monitoring	o	+	++	++
1f	Diverse software design	o	o	+	++

Error detection

	Methods	ASIL A	ASIL B	ASIL C	ASIL D
1a	Static recovery mechanism	+	+	+	+
1b	Graceful degradation	+	+	++	++
1c	Independent parallel redundancy	o	o	+	++
1d	Correcting codes for data	+	+	+	+

Error handling

Source: ISO 26262-6:2011

Methods for the verification of the software architectural design

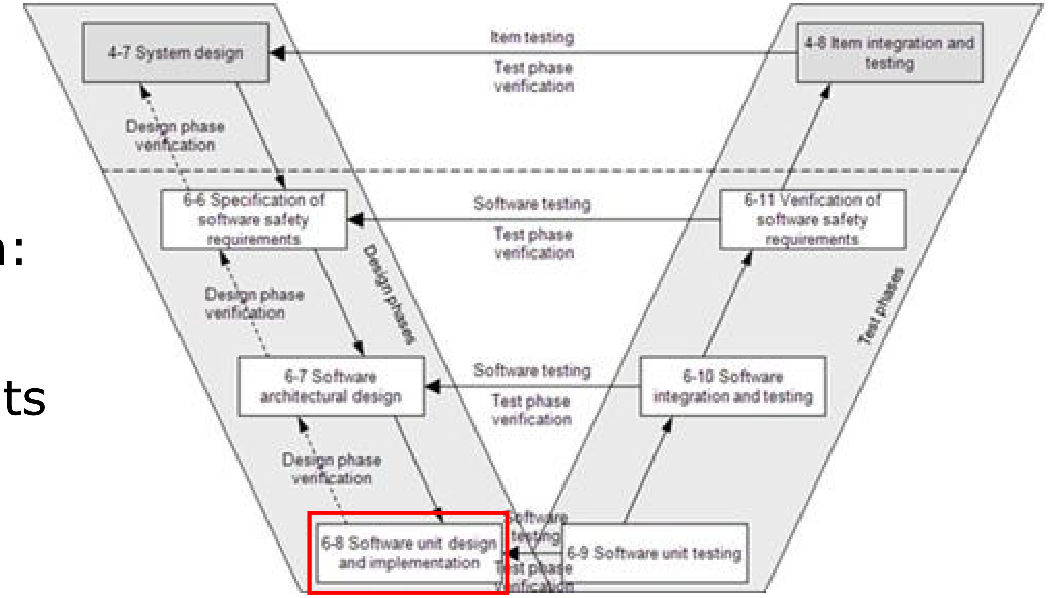
Methods		ASIL A	ASIL B	ASIL C	ASIL D
1a	Walk-through of the design	++	+	o	o
1b	Inspection of the design	+	++	++	++
1c	Simulation of dynamic parts of the design	+	+	+	++
1d	Prototype generation	o	o	+	++
1e	Formal verification	o	o	+	+
1f	Control flow analysis	+	+	++	++
1g	Data flow analysis	+	+	++	++

Source: ISO 26262-6:2011

Software Unit Design and Implementation

Goals

- Specify SW Units based on:
  - SW Architecture
  - SW Safety Requirements
- Implement the SW Units
- Verify SW Units
  - Code reviews / inspections



Source: ISO 26262-6:2011

Design principles for software unit design and implementation

	Methods	ASIL			
		A	B	C	D
1a	One entry and one exit point in subprograms and functions	++	++	++	++
1b	No dynamic objects or variables, or else online test during their creation	+	++	++	++
1c	Initialization of variables	++	++	++	++
1d	No multiple use of variable names	+	++	++	++
1e	Avoid global variables or else justify their usage	+	+	++	++
1f	Limited use of pointers	o	+	+	++
1g	No implicit type conversions	+	++	++	++
1h	No hidden data flow or control flow	+	++	++	++
1i	No unconditional jumps	++	++	++	++
1j	No recursions	+	+	++	++

Source: ISO 26262-6:2011

Example: MISRA C

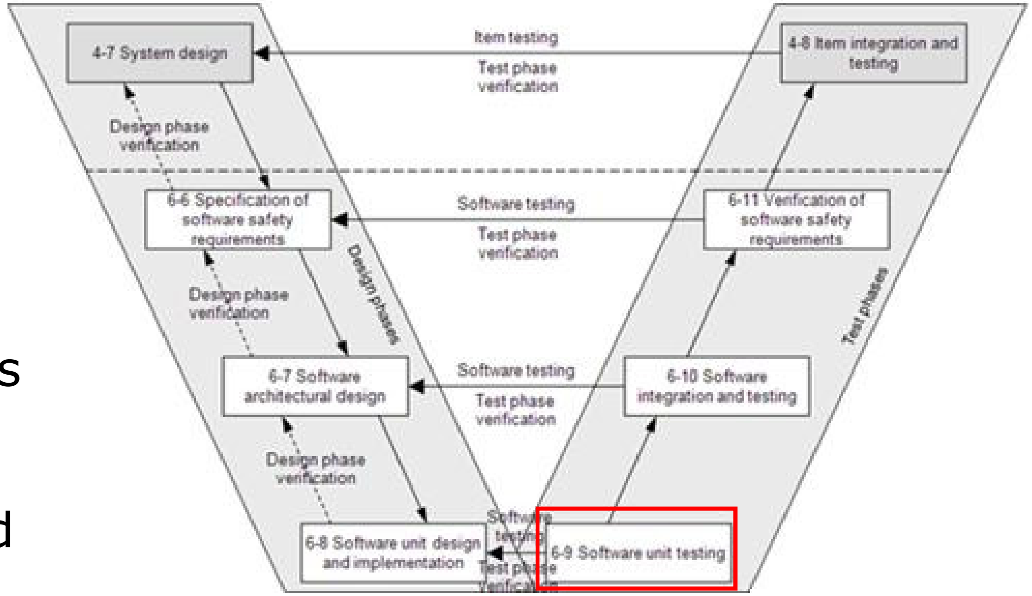
- Programming standard developed by Motor Industry Software Reliability Association
- Avoidance of runtime errors due to unsafe C constructs
- The respect of MISRA C shall be demonstrated ➔ static code analysis

Infos: www.**misra**.org

Software Unit Testing

Goals

- Demonstrate that the software units fulfil the Software Unit Specifications
- Verify absence of undesired functionalities



Source: ISO 26262-6:2011

Software Unit Testing

The software unit testing methods shall be applied to demonstrate that the software units achieve:

- Compliance with the software unit design specification
- Compliance with the specification of the hardware-software interface
- Correct implementation of the functionality
- Absence of unintended functionality
- Robustness
- Sufficiency of the resources to support the functionality

Methods		ASIL			
		A	B	C	D
1a	Requirements-based test	++	++	++	++
1b	Interface test	++	++	++	++
1c	Fault injection test	+	+	+	++
1d	Resource usage test	+	+	+	++
1e	Back-to-back comparison test between model and code, if applicable	+	+	++	++

Source: ISO 26262-6:2011

Software Unit Testing

Methods for deriving test cases for software unit testing

Methods		ASIL			
		A	B	C	D
1a	Analysis of requirements	++	++	++	++
1b	Generation and analysis of equivalence classes	+	++	++	++
1c	Analysis of boundary values	+	++	++	++
1d	Error guessing	+	+	+	+

Source: ISO 26262-6:2011

Structural coverage metrics at the software unit level

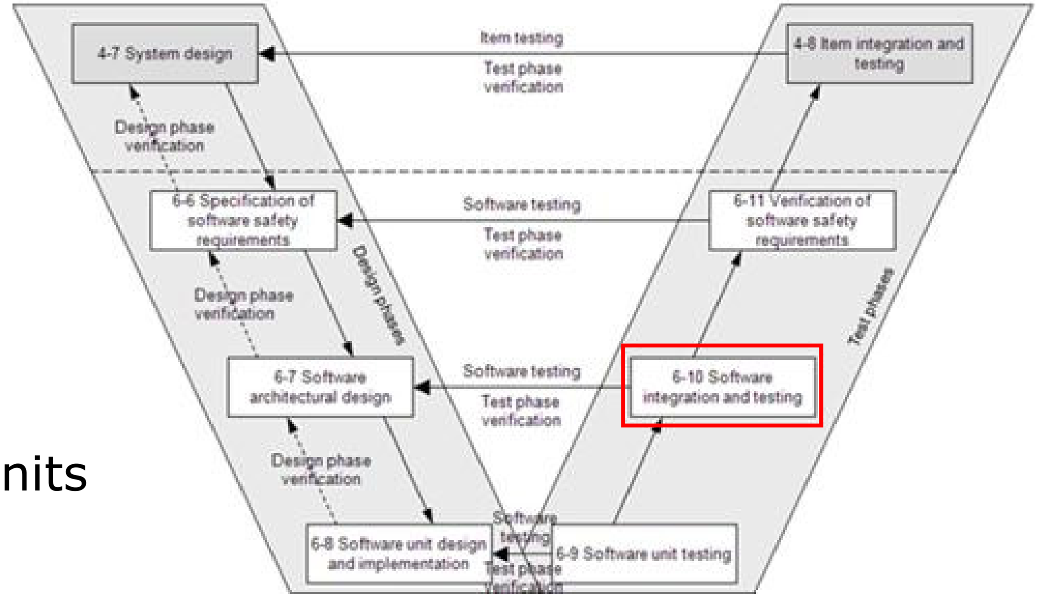
Methods		ASIL			
		A	B	C	D
1a	Statement coverage	++	++	+	+
1b	Branch coverage	+	++	++	++
1c	MC/DC (Modified Condition/Decision Coverage)	+	+	+	++

Source: ISO 26262-6:2011

Software Integration and Testing

Goals

- Integrate SW components
  - Integration sequence
  - Testing of interfaces between components/units
- Verify correct implementation of the SW Architecture



Source: ISO 26262-6:2011

Software Integration and Testing

The software integration test methods shall be applied to demonstrate that both the software components and the embedded software achieve:

- Compliance with the software architectural design
- Compliance with the specification of the hardware-software interface
- Correct implementation of the functionality
- Robustness and sufficiency of the resources to support the functionality

Methods		ASIL			
		A	B	C	D
1a	Requirements-based test	++	++	++	++
1b	Interface test	++	++	++	++
1c	Fault injection test	+	+	++	++
1d	Resource usage test	+	+	+	++
1e	Back-to-back comparison test between model and code, if applicable	+	+	++	++

Source: ISO 26262-6:2011

Structural coverage metrics at the software architectural level

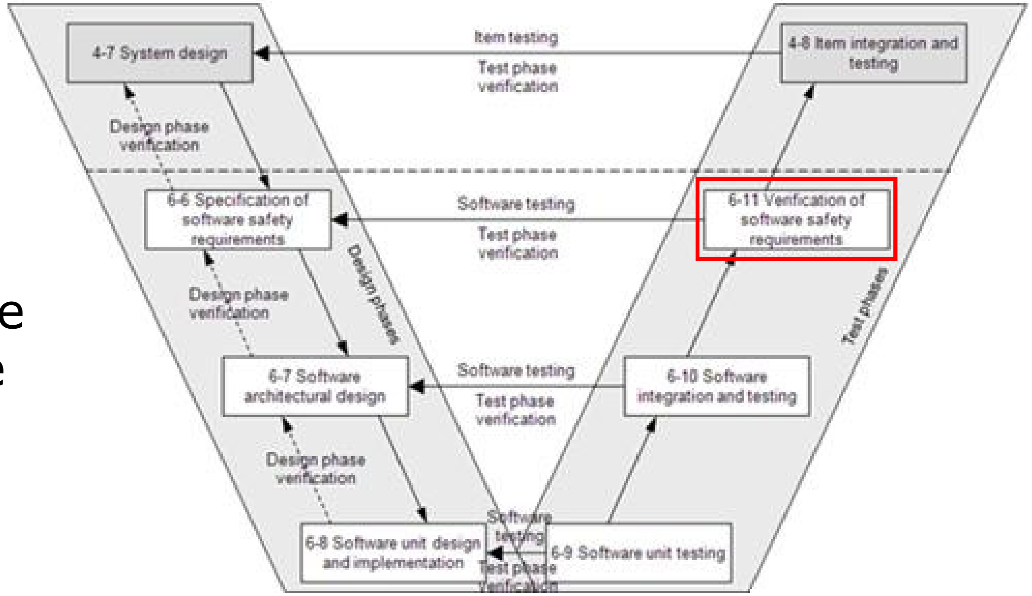
Methods		ASIL			
		A	B	C	D
1a	Function coverage	+	+	++	++
1b	Call coverage	+	+	++	++

Source: ISO 26262-6:2011

Verification of Software Safety Requirements

Goals

- Verify that the embedded software fulfils the Software Safety Requirements in the target environment



Source: ISO 26262-6:2011

Verification of Software Safety Requirements

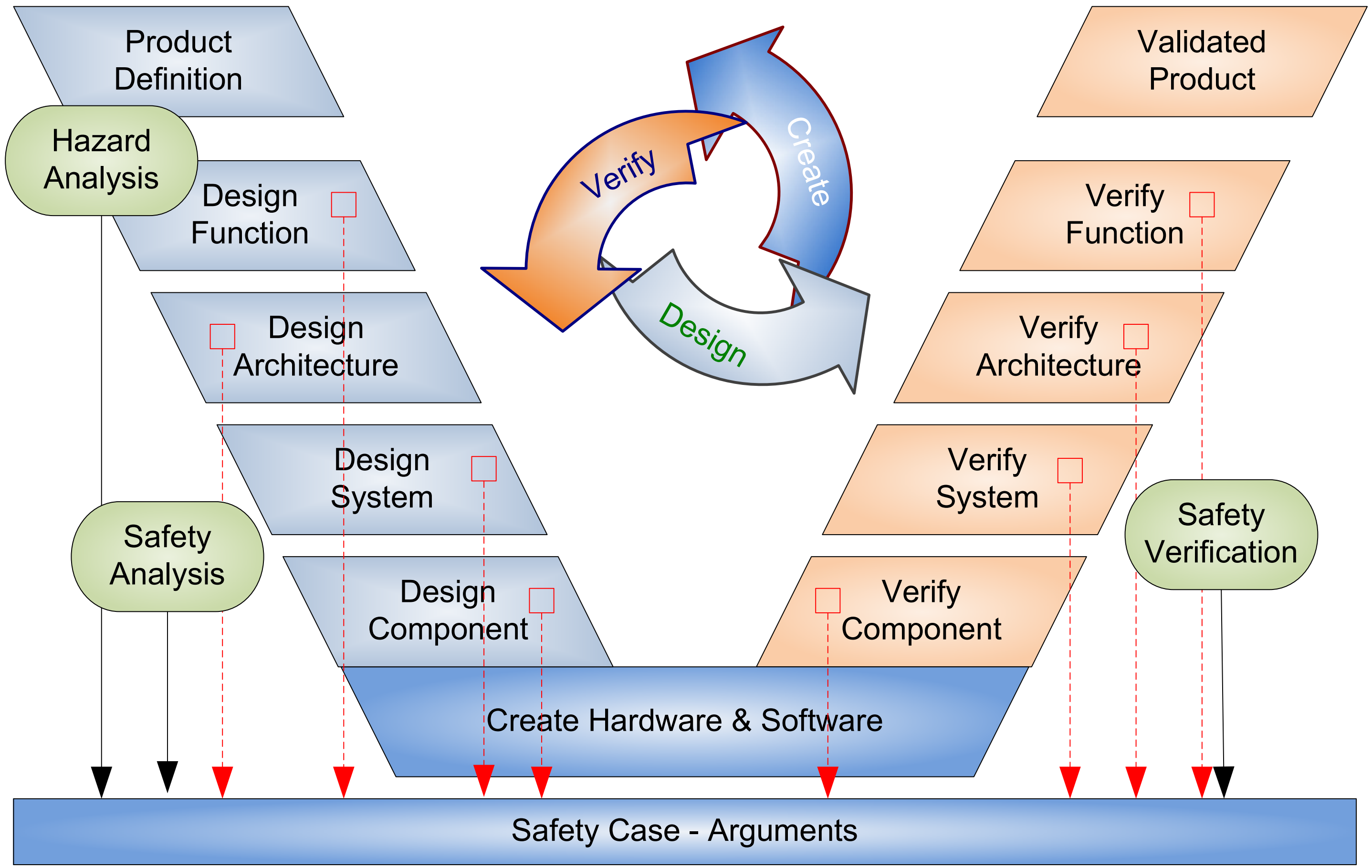
- Verify that the embedded software fulfils the software safety requirements
- Verification of the software safety requirements shall be executed on the target hardware
- The results of the verification of the software safety requirements shall be evaluated in accordance with:
  - Compliance with the expected results
  - Coverage of the software safety requirements
  - A pass or fail criteria

Methods		ASIL			
		A	B	C	D
1a	Hardware-in-the-loop	+	+	++	++
1b	Electronic control unit network environments	++	++	++	++
1c	Vehicles	++	++	++	++

Source: ISO 26262-6:2011

Functional Safety Assessment

What shall be provided to support the Safety Case?



Contents

Qualification of Software Tools

To determine the required level of confidence in a software tool, perform a use case analysis:

- Evaluate if a malfunctioning software tool and its erroneous output can lead to the violation of any safety requirement allocated to the safety-related item or element to be developed
- Establish probability of preventing or detecting such errors in its output
  - Considers measures internal to the software tool (e.g. monitoring)
  - Measures external to the software tool implemented in the development process for the safety-related item or element (e.g. guidelines, tests, reviews)

Tool Impact (TI)

Possibility that a safety requirement, allocated to the safety-related item or element, is violated if the software tool is malfunctioning or producing erroneous output

TI1 – no such possibility

TI2 – all other cases

Tool error Detection (TD)

Probability of preventing or detecting that the software tool is malfunctioning or producing erroneous output

TD1 – high degree of confidence for prevention or detection

TD2 – medium degree of confidence for prevention or detection

TD3 – all other cases

Tool Confidence Level (TCL)

Based on the values determined for the classes of TI and TD

	TD1	TD2	TD3
TI1	TCL1	TCL1	TCL1
TI2	TCL1	TCL2	TCL3

Source: ISO 26262-8:2011

Qualification methods:

Qualification methods of software tools classified TCL3		ASIL			
		A	B	C	D
1a	Increased confidence from use	++	++	+	+
1b	Evaluation of the tool development process	++	++	+	+
1c	Validation of the software tool	+	+	++	++
1d	Development in accordance with a safety standard	+	+	++	++

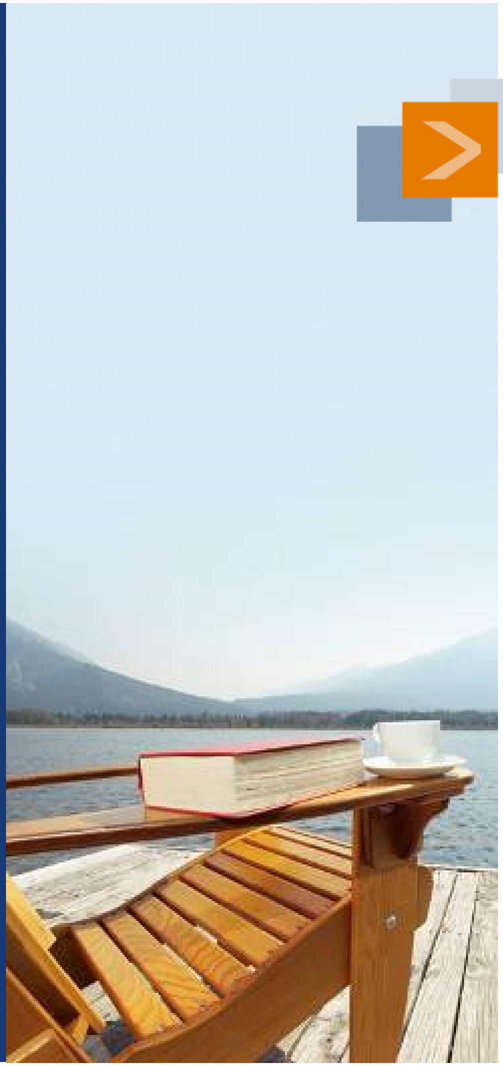
Qualification methods of software tools classified TCL2		ASIL			
		A	B	C	D
1a	Increased confidence from use	++	++	++	+
1b	Evaluation of the tool development process	++	++	++	+
1c	Validation of the software tool	+	+	+	++
1d	Development in accordance with a safety standard	+	+	+	++

Source: ISO 26262-8:2011

Contents

Summary

- Today's electronic systems are too complex to understand all potential hazards
- An approach for Functional Safety is needed to avoid severe injuries and damages in human lives and property
- A standardized way to show that your product is safe is needed – best practice yet not fully established – guidance needed



Thank you !

Bernhard Sechser

Principal Consultant SPICE & Safety

Method Park Consulting GmbH
Wetterkreuz 19a
91058 Erlangen
Germany

Phone: +49 9131 97206-427
Mobile: +49 173 3882055

Bernhard.Sechser@methodpark.com
http://www.xing.com/profile/Bernhard_Sechser
http://www.methodpark.com