

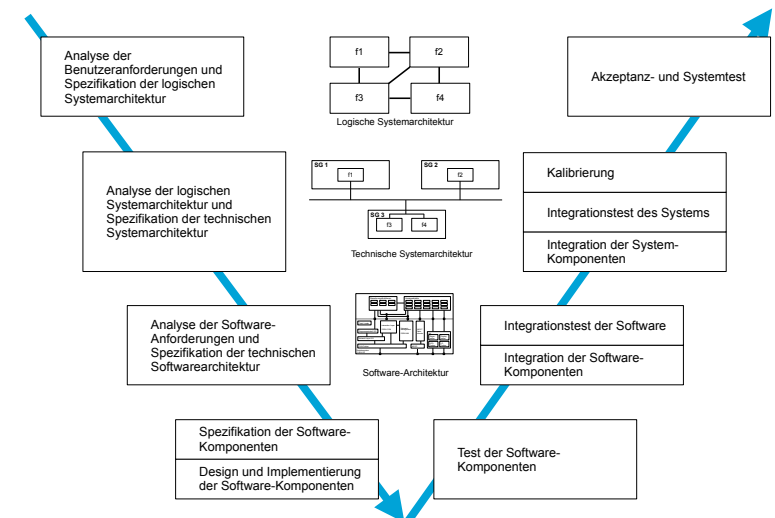
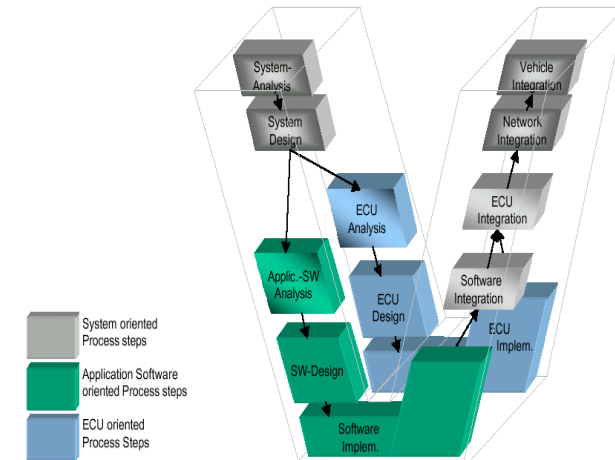
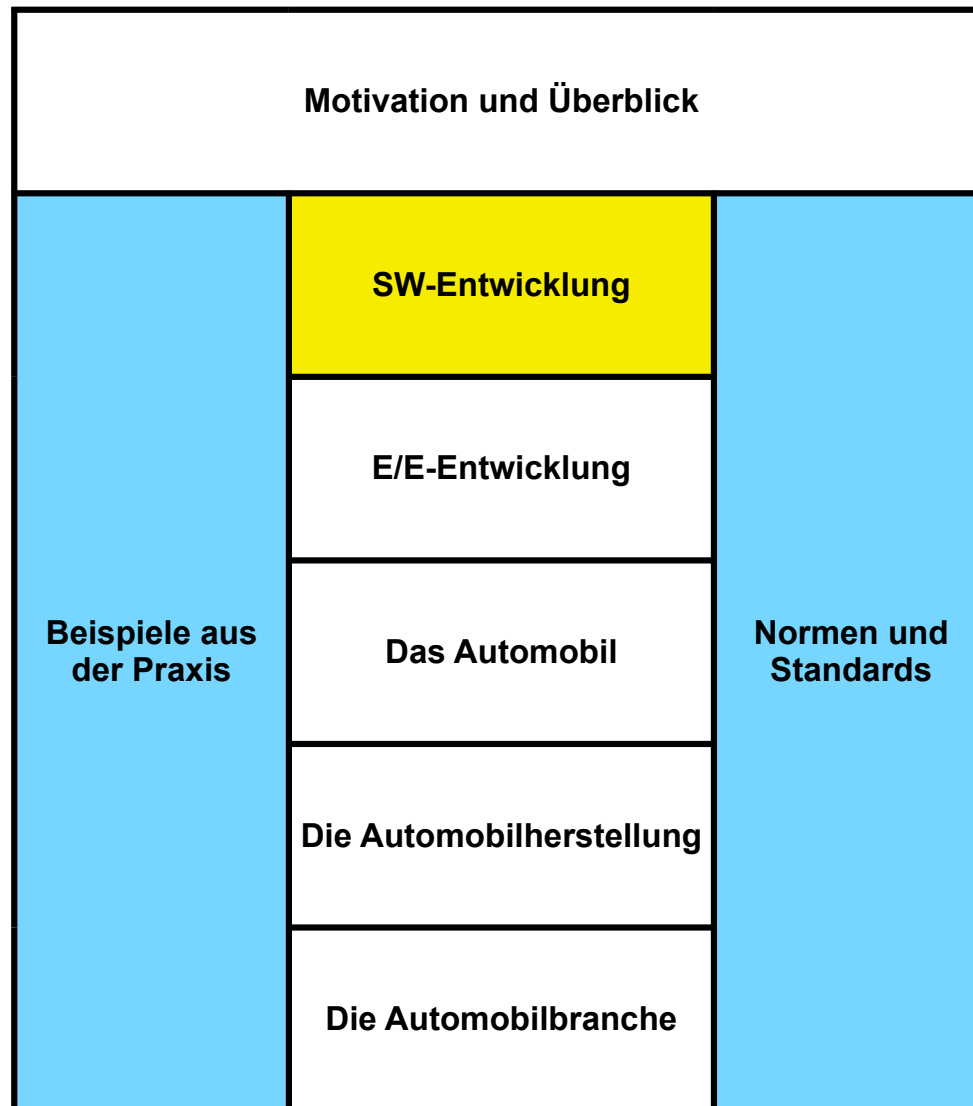
Vorlesung Automotive Software Engineering Teil 6 SW-Entwicklung (1)

Sommersemester 2014

Prof. Dr. rer. nat. Bernhard Hohlfeld

Bernhard.Hohlfeld@mailbox.tu-dresden.de

Technische Universität Dresden, Fakultät Informatik
Honorarprofessur Automotive Software Engineering



- Vorgehensmodelle kennen
- Grundbegriffe des Systems Engineering kennen
- Den Kernprozess zur Entwicklung von elektronischen Systemen und Software im Fahrzeug verstehen
- Das V-Modell zu Software-Entwicklung in einer speziellen Ausprägung mit zahlreichen Beispielen kennen
- Die Unterstützungsprozesse zur Entwicklung von elektronischen Systemen und Software im Fahrzeug kennen

6. SW-Entwicklung



1. Vorgehensmodelle

2. Kernprozess

3. Unterstützungsprozesse

Das V-Modell ist schon ziemlich alt ...



Das V-Modell ist schon ziemlich alt ...



Das V-Modell ist schon ziemlich alt ...



Das V-Modell ist schon ziemlich alt ...



Das V-Modell ist schon ziemlich alt ...



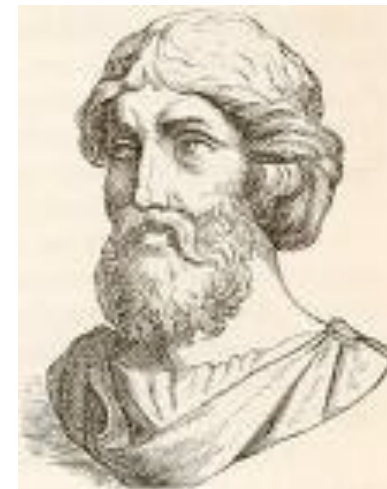
Das V-Modell ist schon ziemlich alt ...



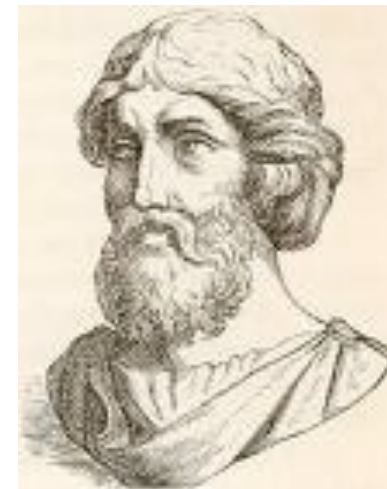
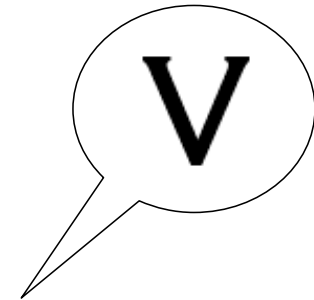
... und wurde ständig weiterentwickelt und verfeinert



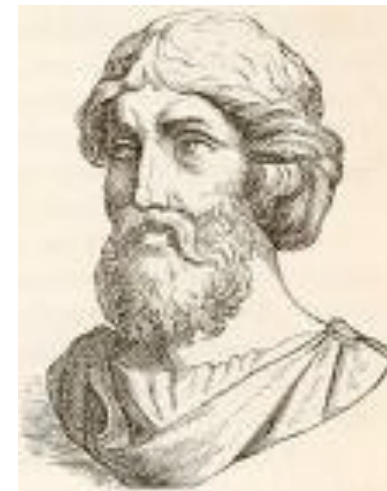
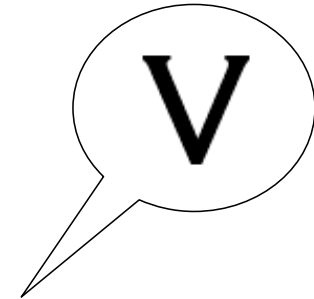
... und wurde ständig weiterentwickelt und verfeinert



... und wurde ständig weiterentwickelt und verfeinert



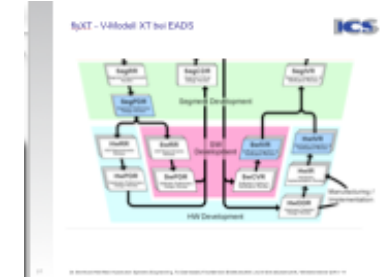
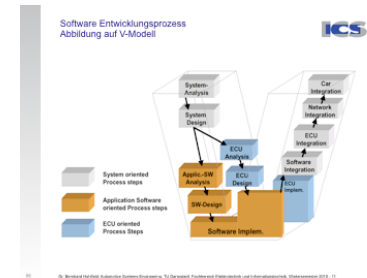
... und wurde ständig weiterentwickelt und verfeinert

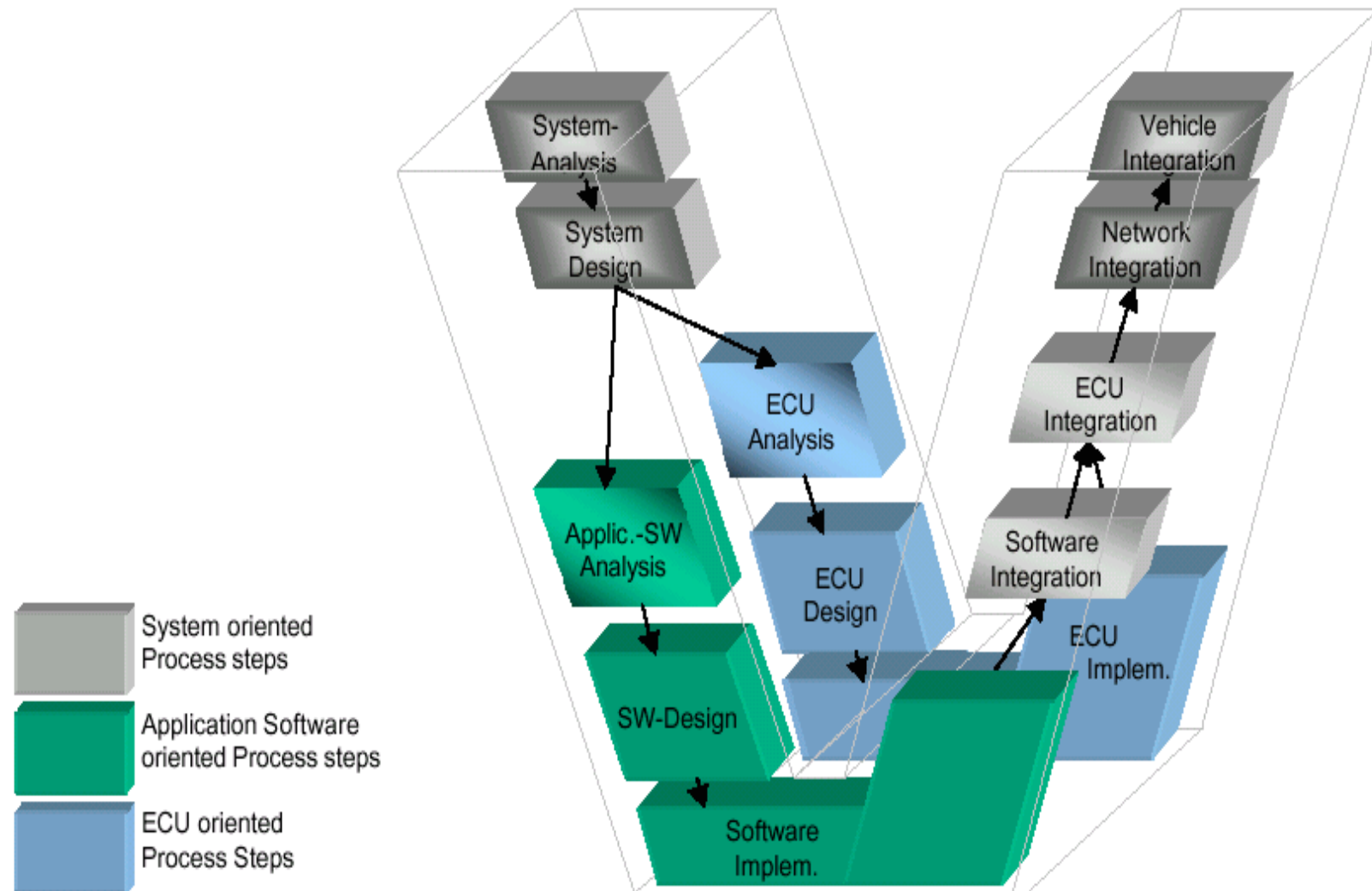


V-Modell(e)

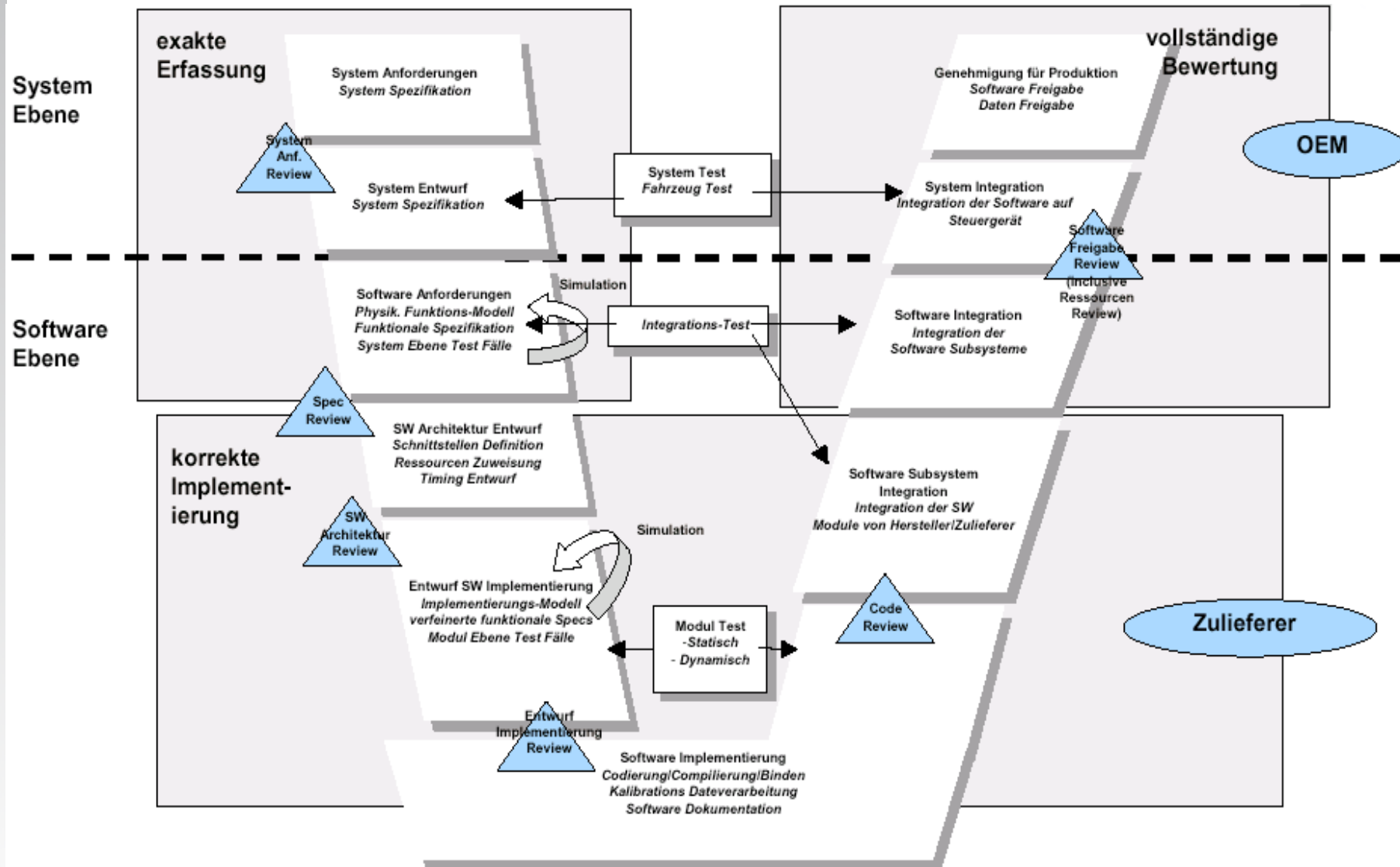


- Es gibt nicht „das“ allgemein gültige V-Modell, sondern verschiedene Ausprägungen
- Charakteristisch für die SW-Entwicklung bei eingebetteten Systemen ist die Kopplung mit der System-Entwicklung und der HW-Entwicklung

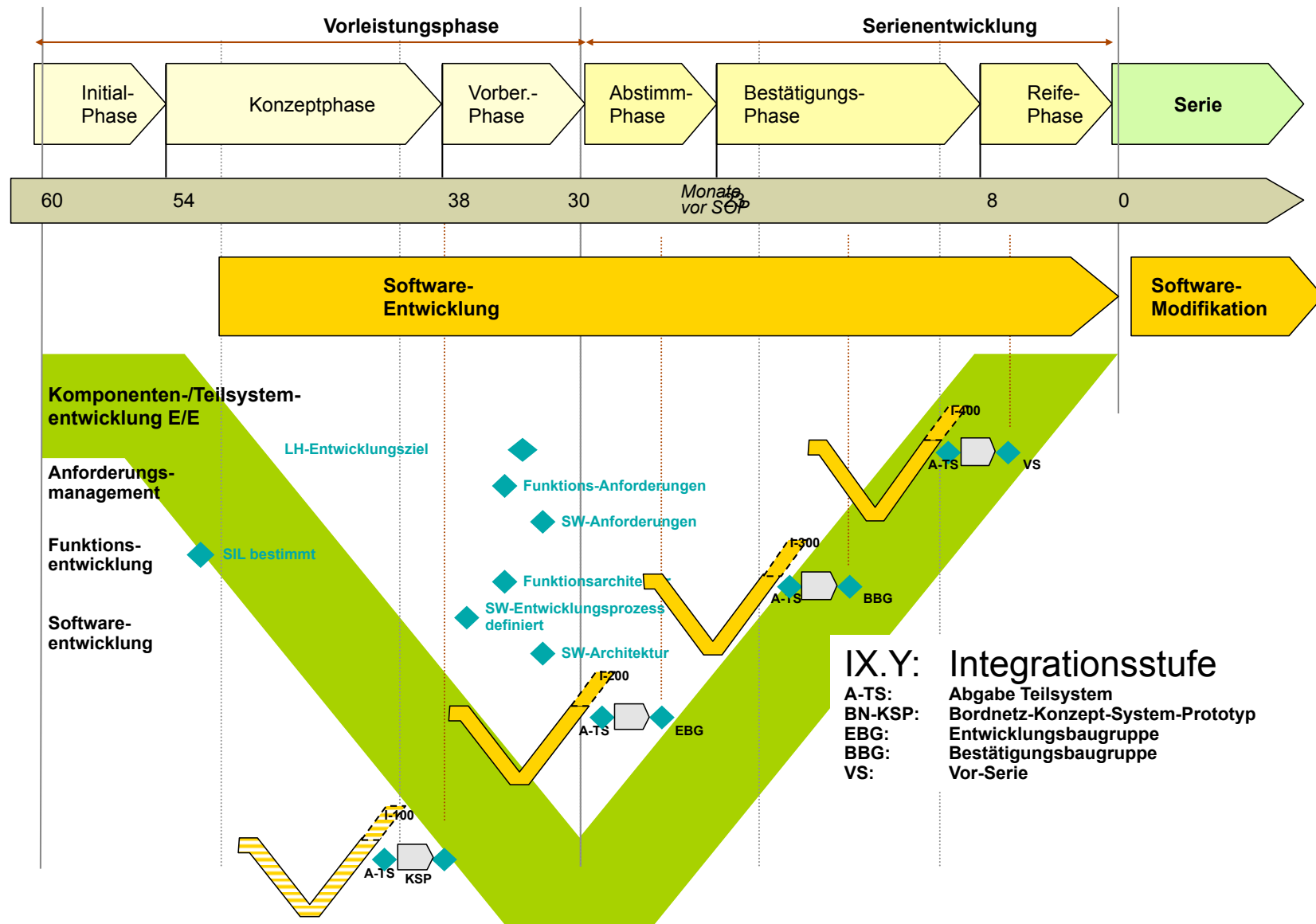




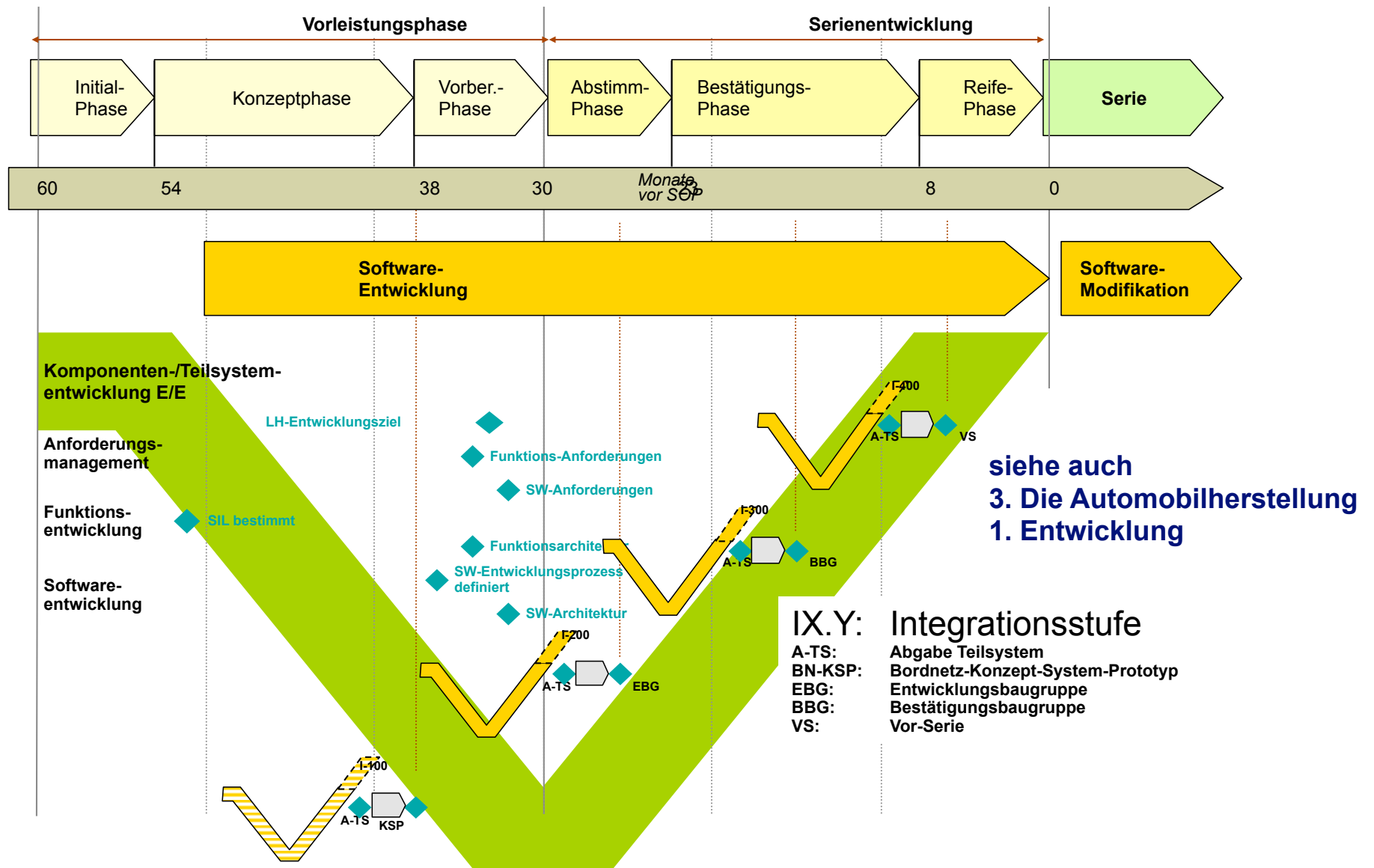
Detaillierung V-Modell



Software Entwicklungsprozess Einordnung in Produktentstehungsprozess



Software Entwicklungsprozess Einordnung in Produktentstehungsprozess



Software Entwicklungsprozess

Einordnung in Produktentstehungsprozess



■ Komponenten-/Teilsystementwicklung E/E

- LH-Entwicklungsziel Lastenheft
- Funktions-Anforderungen
- SW-Anforderungen

■ Funktionsentwicklung

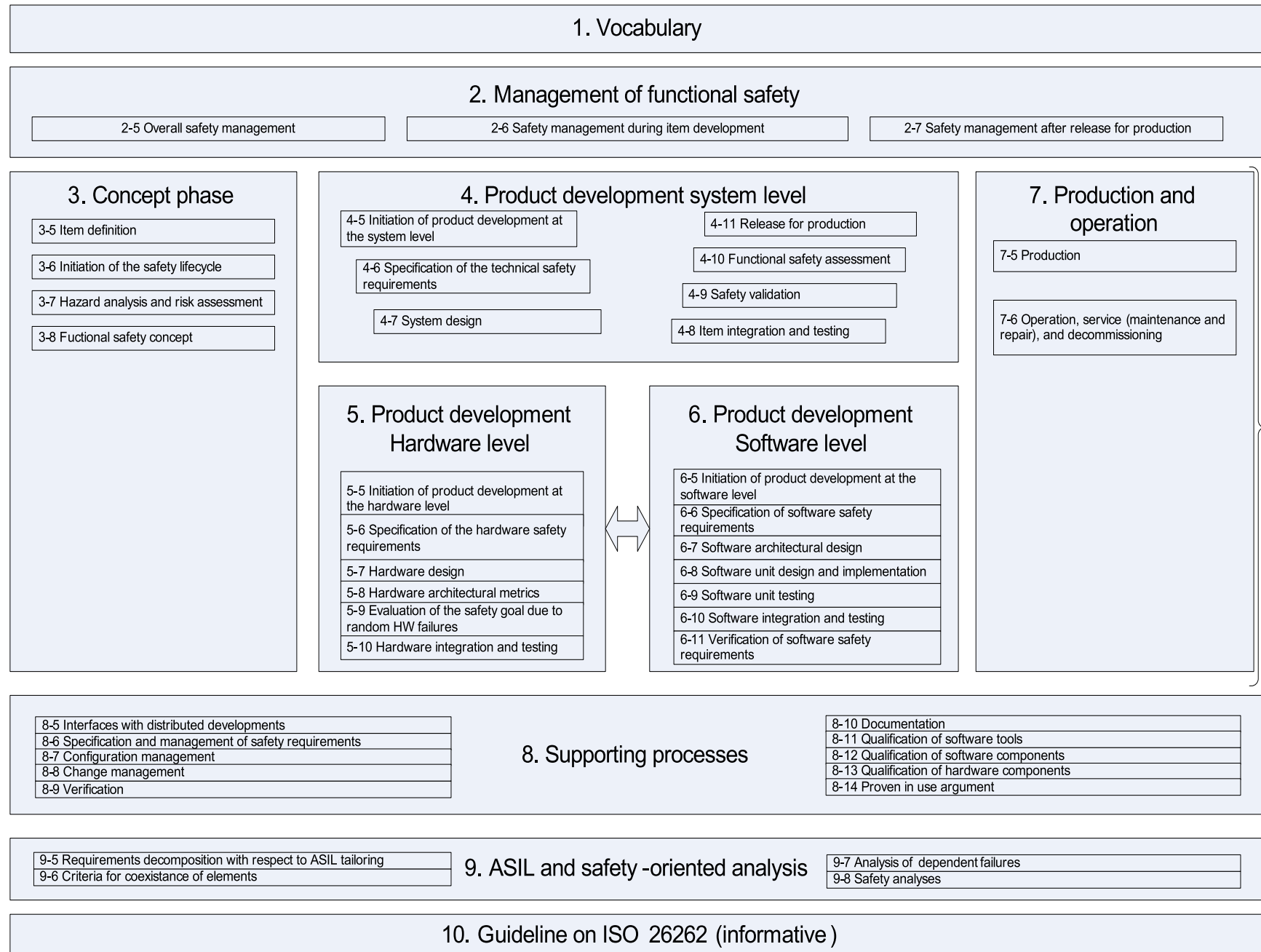
- (A)SIL bestimmt (Automotive) Safety Integrity Level, siehe auch Teil 7 Standards
- Funktionsarchitektur

■ Softwareentwicklung

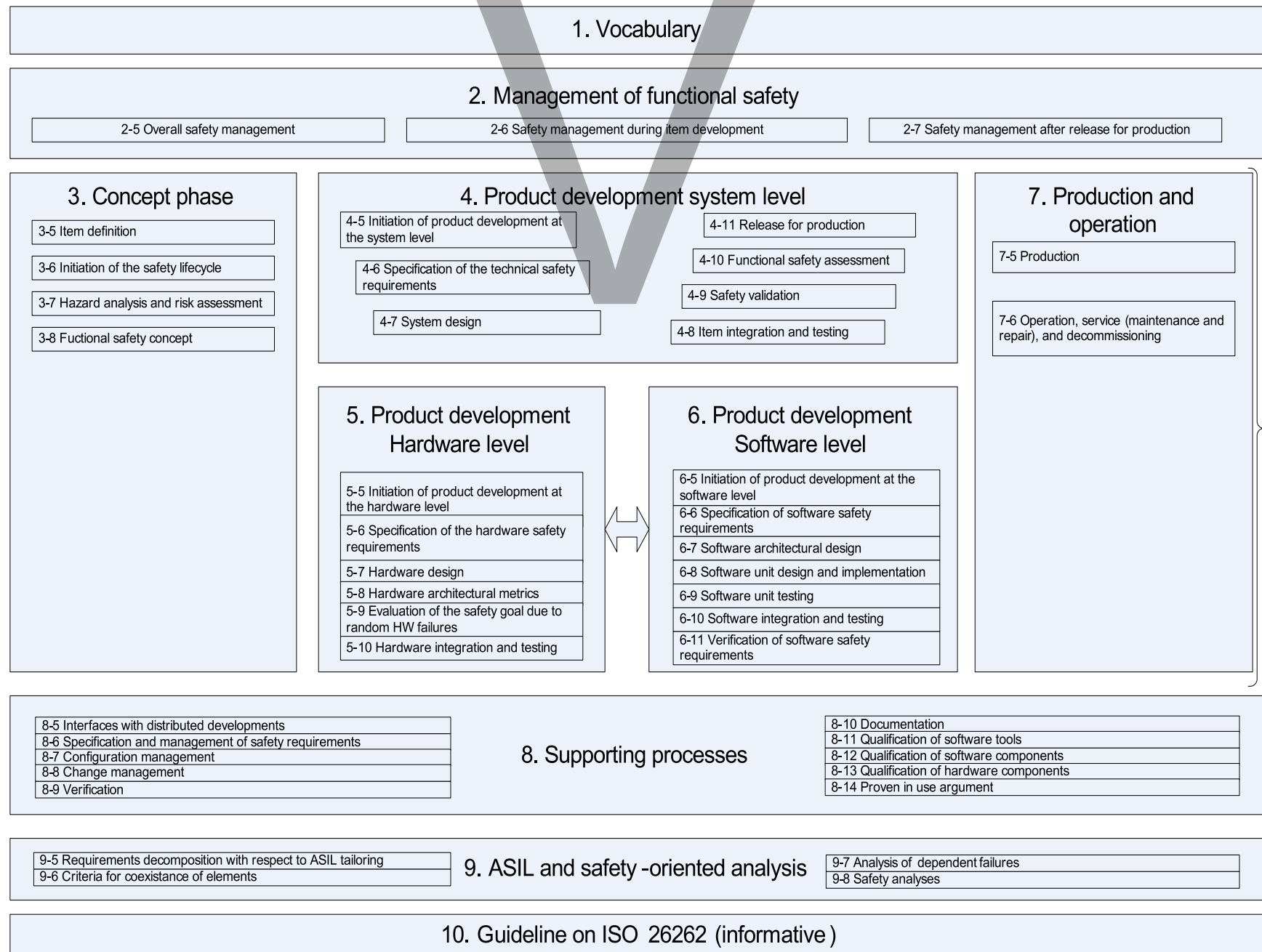
- SW-Entwicklungsprozess definiert
- SW-Architektur

■ Integrationsstufen

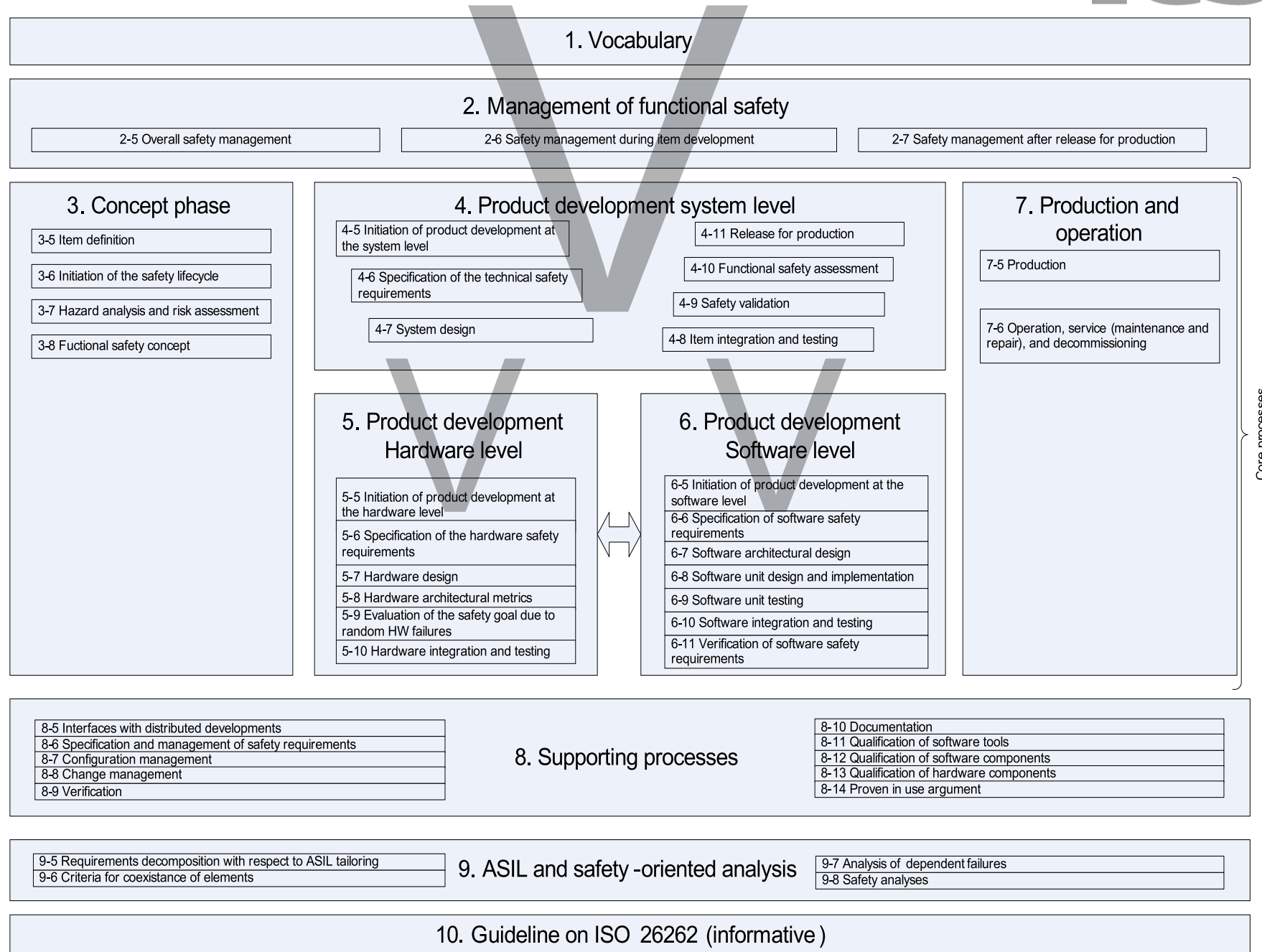
- I-100 BN-KSP: Bordnetz-Konzept-System-Prototyp
- I-200 EBG: Entwicklungsbaugruppe
- I-300 BBG: Bestätigungsbaugruppe
- I-400 VS: Vorserie



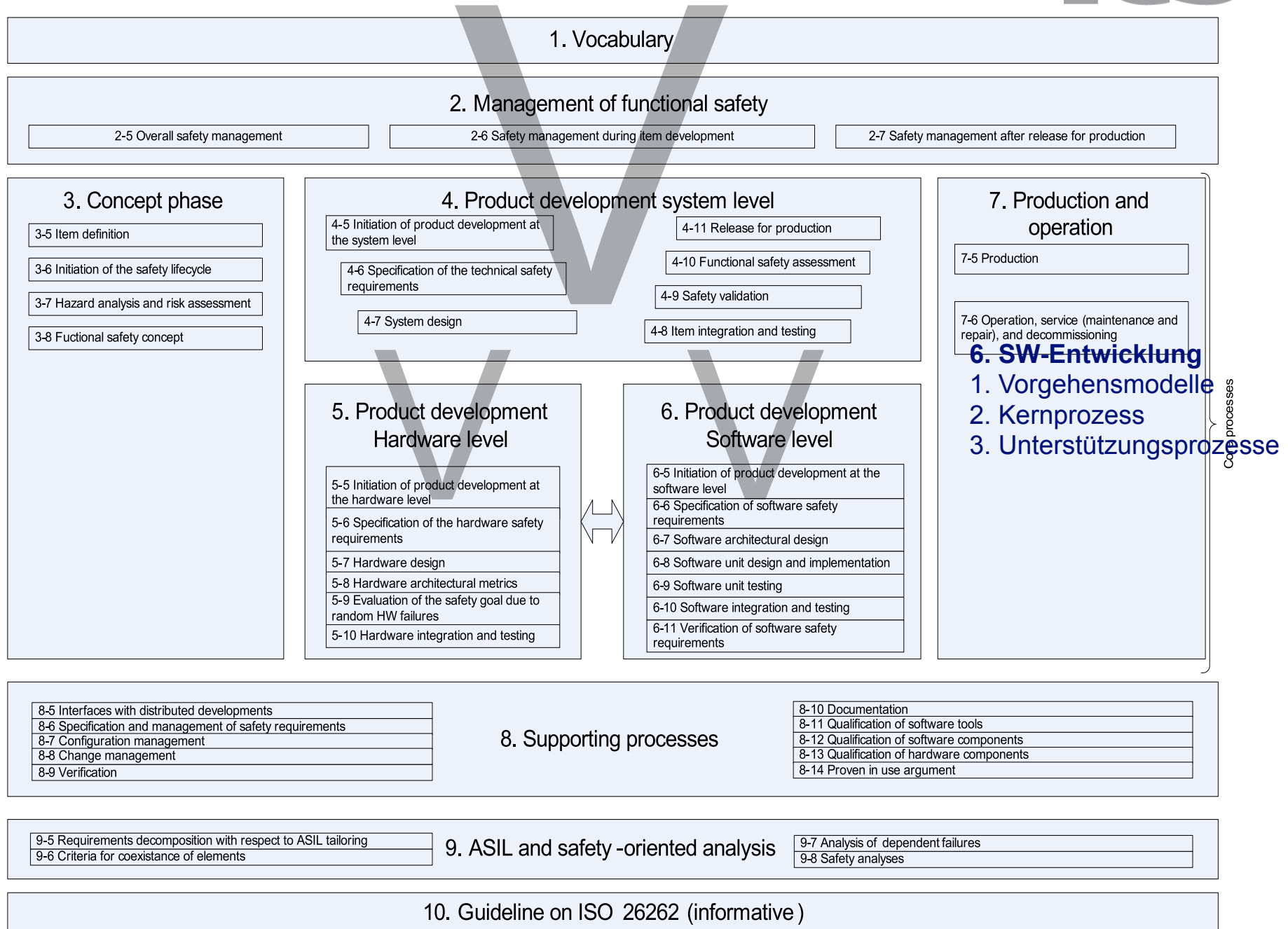
Automotive Safety Standard ISO 26262 im Detail



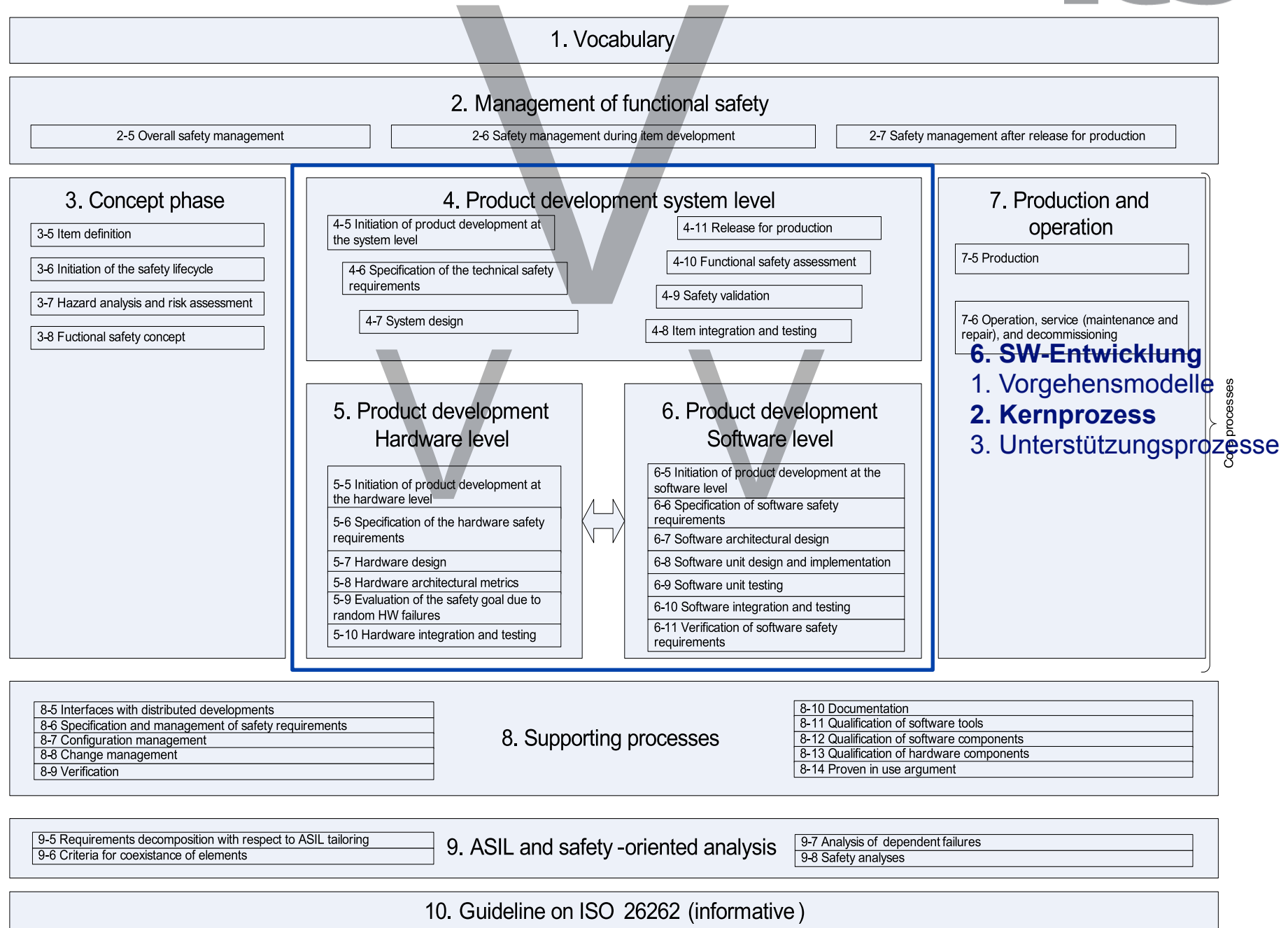
Automotive Safety Standard ISO 26262 im Detail



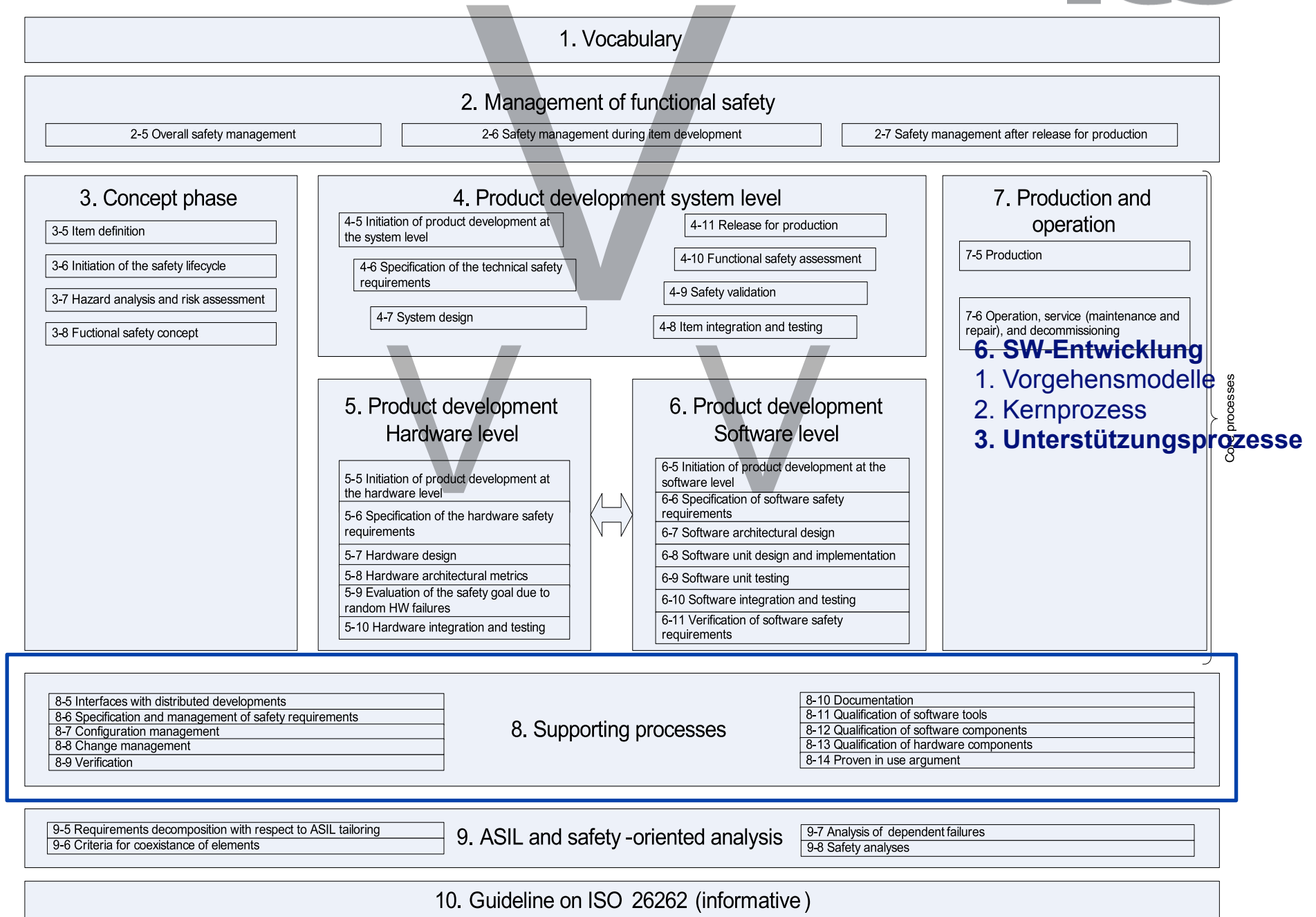
Automotive Safety Standard ISO 26262 im Detail

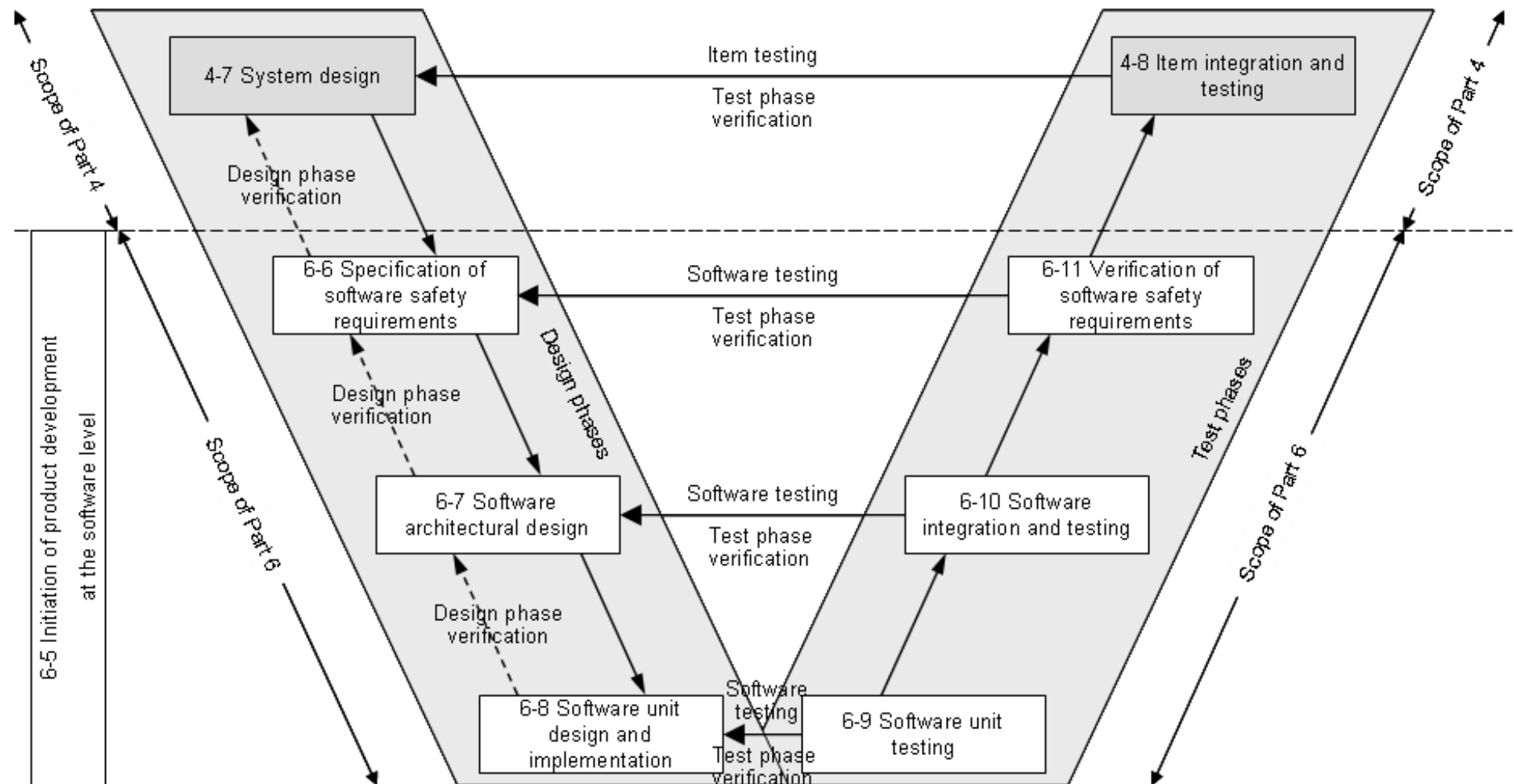


Automotive Safety Standard ISO 26262 im Detail

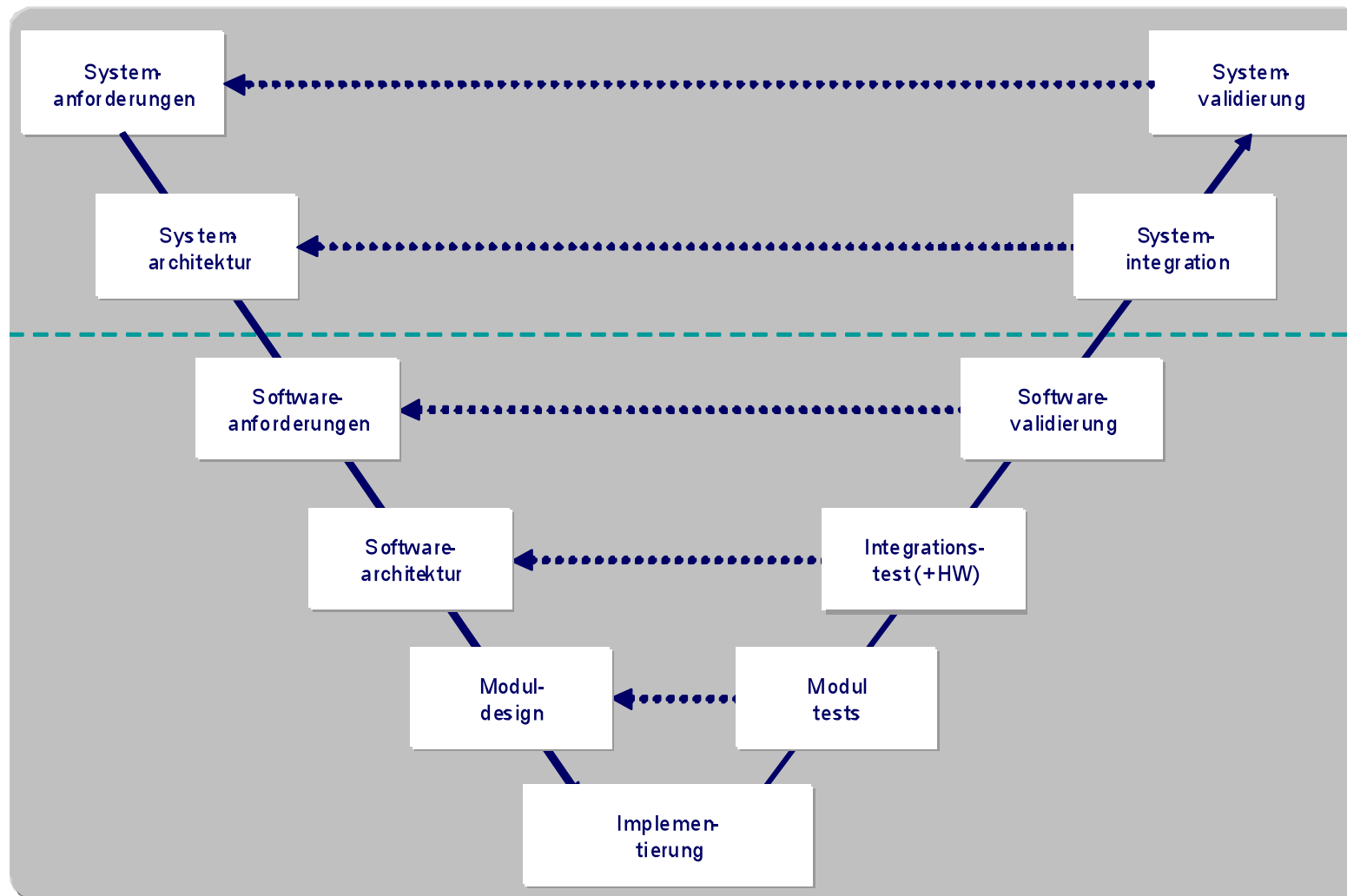


Automotive Safety Standard ISO 26262 im Detail



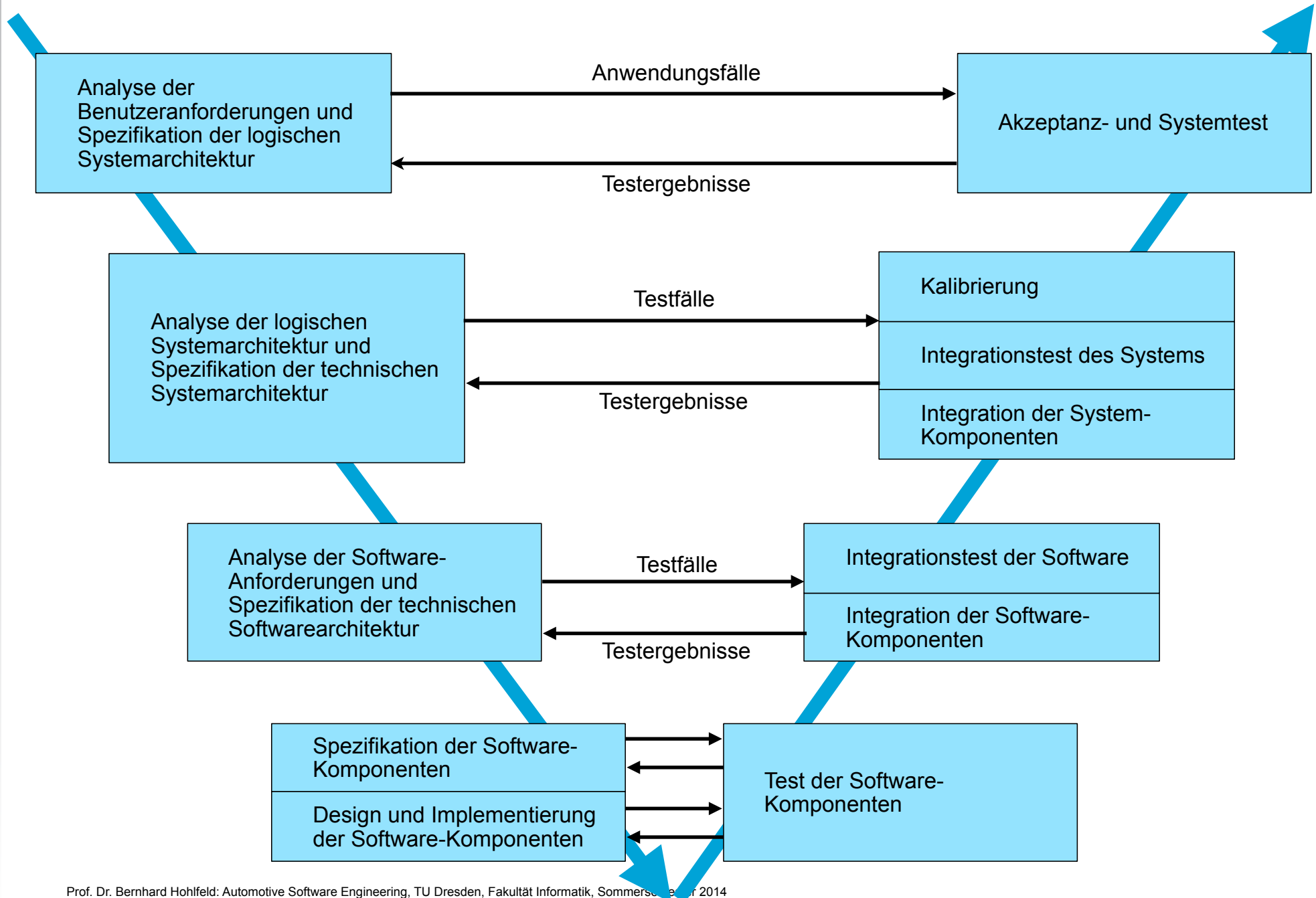


siehe auch Teil 7 Normen und Standards



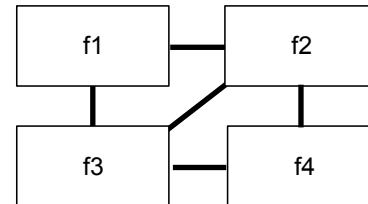
- DIN EN 50128 Bahnanwendungen -
- Telekommunikationstechnik, Signaltechnik und Datenverarbeitungssysteme Software für Eisenbahnsteuerungs- und Überwachungssysteme

Kernprozess zur Entwicklung von elektronischen Systemen und Software (Nach Schäuuffele, Zurawka)



Kernprozess zur Entwicklung von elektronischen Systemen und Software (Nach Schäuuffele, Zurawka)

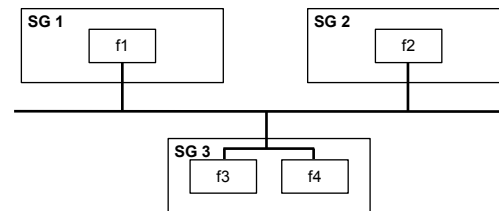
Analyse der Benutzeranforderungen und Spezifikation der logischen Systemarchitektur



Logische Systemarchitektur

Akzeptanz- und Systemtest

Analyse der logischen Systemarchitektur und Spezifikation der technischen Systemarchitektur



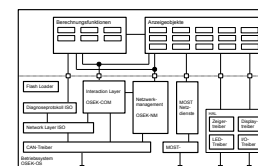
Technische Systemarchitektur

Kalibrierung

Integrationstest des Systems

Integration der System-Komponenten

Analyse der Software-Anforderungen und Spezifikation der technischen Softwarearchitektur



Software-Architektur

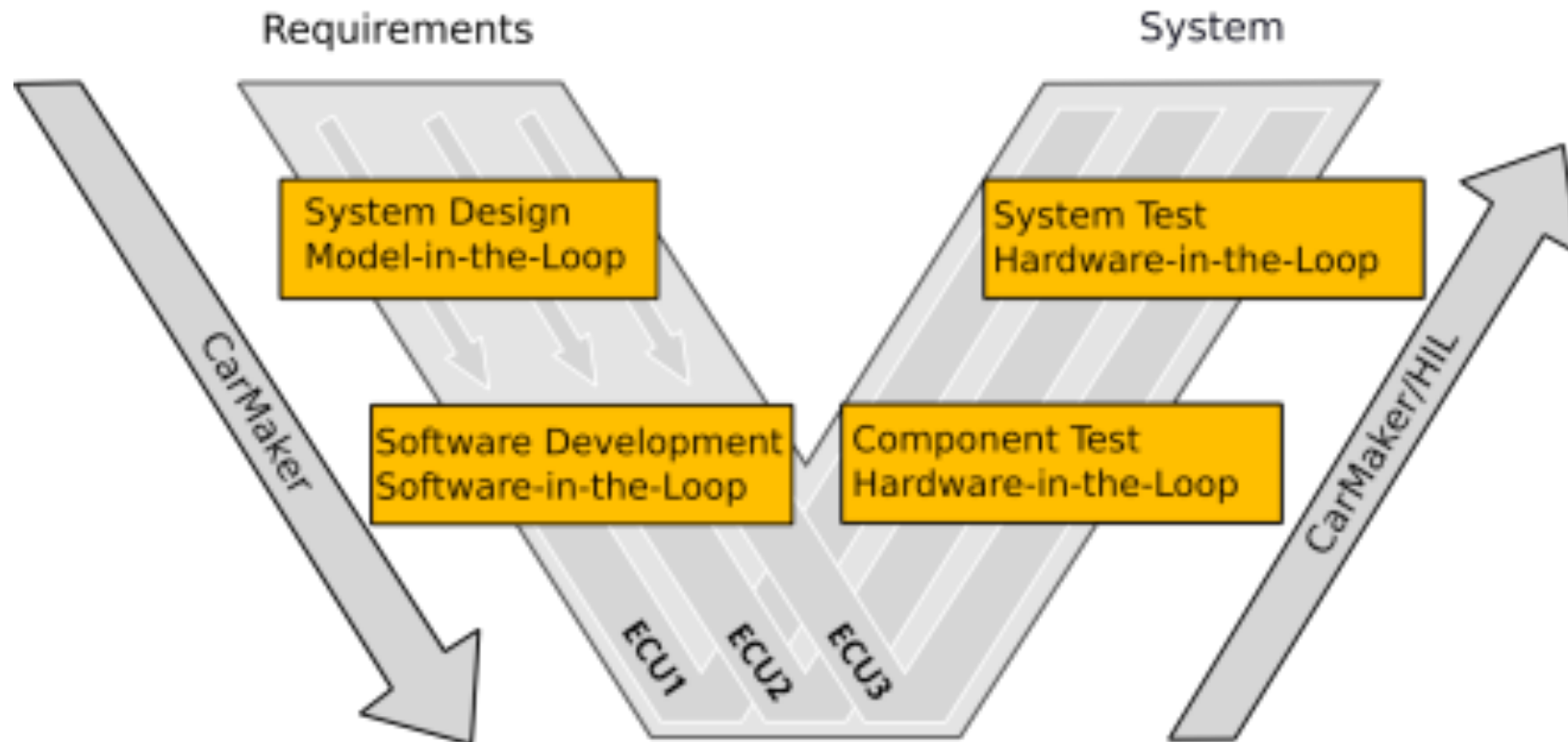
Integrationstest der Software

Integration der Software-Komponenten

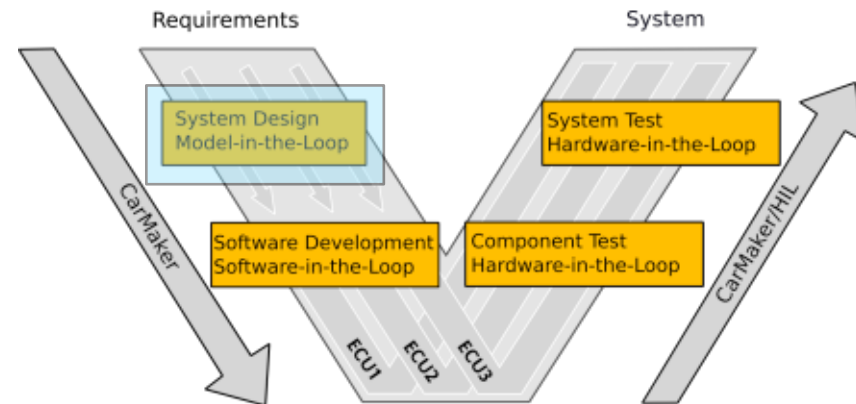
Spezifikation der Software-Komponenten

Design und Implementierung der Software-Komponenten

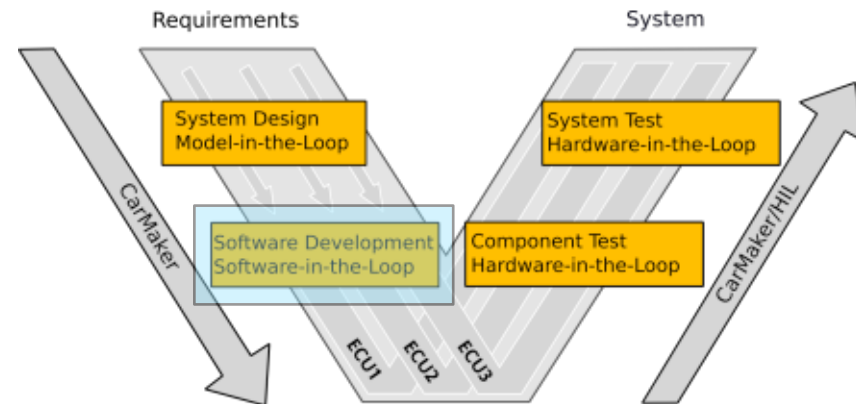
Test der Software-Komponenten



- Ausgangspunkt der Entwicklung sind die Anforderungen, die in Form einer Systemspezifikation festgehalten werden.
- Die Funktionen der Systemspezifikation werden in Software-Modellen mit grafisch orientierten Programmiersystemen wie MATLAB/Simulink entwickelt. Diese Phase kann durch die Model-in-the-Loop (MIL)-Simulation unterstützt werden. Dabei werden die Software-Modelle mit CarMaker im virtuellen Fahrzeug validiert.
- Alle Modelle von CarMaker sind vollständig in Simulink integriert, so dass der Anwender problemlos Modifikationen an den Modellen vornehmen kann, indem er z.B. bestimmte Komponenten der Modelle durch eigene Simulink-Blöcke ersetzt.

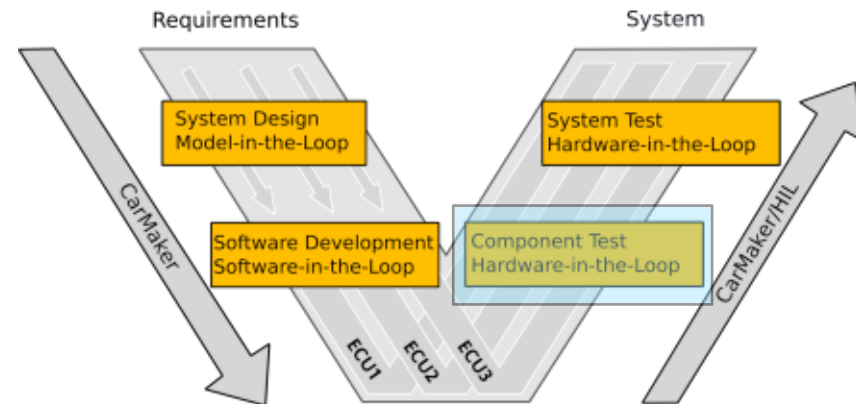


- Das in der MIL-Simulation getestete Software-Modell wird durch die Software des Reglers ersetzt und in die Simulationsumgebung von CarMaker eingebunden (Software-in-the-Loop SIL). Somit wird der spätere Seriencode in einer virtuellen Fahrzeugumgebung ausgeführt, um das Verhalten zu untersuchen und Fehler in der Implementierung aufzudecken. Die Besonderheit von CarMaker ist, dass auch die SIL-Tests auf einem Echtzeitsystem erfolgen können, um die Software unter zeitlich kritischen Randbedingung zu prüfen. Damit muss die Software nicht wie in vielen anderen Anwendungen für den Test extra angepasst werden.

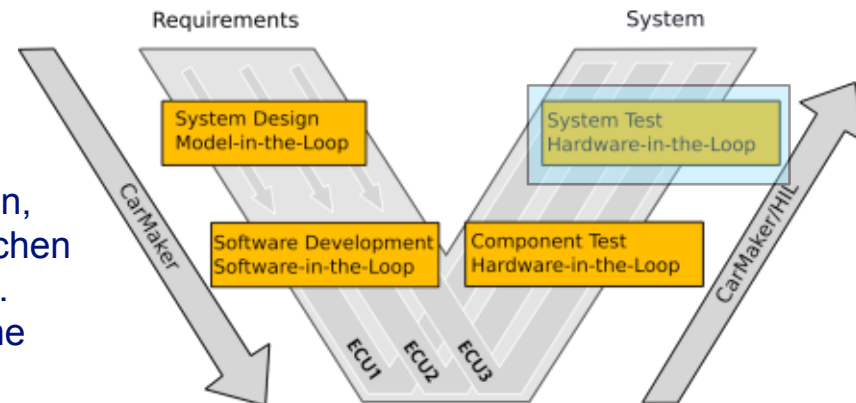


- Die Komponenten-Tests sollen sicherstellen, dass das ausführbare Programm im Zusammenspiel mit der Steuergeräte-Hardware und dem Betriebssystem funktioniert. Das reale Steuergerät wird hierzu an einen CarMaker/HIL-Prüfstand angebunden, der in Echtzeit das Fahrzeug simuliert und die Signale zur Ansteuerung des Steuergerätes generiert (Hardware-in-the-Loop).

Die Bandbreite der Tests am HIL-Prüfstand reicht von Performance-Tests bis hin zu Sicherheitstest, bei denen das Verhalten des Fahrzeugs in Folge von elektrischen Fehlern geprüft wird. Für diese Tests bietet IPG zusätzlich den FailSafeTester an, mit dem elektrische Fehler am HIL-Prüfstand automatisch generiert werden können. Dank der Unterstützung von CCP (CAN Calibration Protocol) und XCP (Universal Measurement and Calibration Protocol) können mit CarMaker auch Steuergeräte-interne Messgrößen erfasst und analysiert werden.



- Der Automobilhersteller hat die Verantwortung für das Fahrzeug als Gesamtsystem. Er muss sicherstellen, dass die einzelnen Steuergeräte nach der Integration im Fahrzeug optimal miteinander interagieren. Um dies zu gewährleisten, müssen die Systeme, die häufig von unterschiedlichen Zulieferern stammen, im Verbund getestet werden. Die Soft- und Hardware der CarMaker/HIL-Systeme ist daher modular aufgebaut, so dass einzelne Steuergeräte flexibel zu- oder abgeschaltet werden können.



- Die Funktionalität des Gesamtverbundes kann damit in allen Fahrsituationen (z.B. Kurvenfahrten, Vollbremsung etc.) untersucht werden. Eine wichtige Rolle spielt bei den Verbundtests die Überwachung der Kommunikation zwischen den Steuergeräten (via CAN, FlexRay etc.). Neben dem Normalbetrieb muss auch hier das Verhalten im Fehlerfall geprüft werden. Je nach Ziel der Untersuchung kann der Prüfling neben den Steuergeräten auch mechanische Komponenten (z.B. Dämpfer, Lenksystem, Motor) enthalten. Die Anzahl der realen Komponenten, die in den Prüfstand integriert werden, kann je nach Anwendung stark variieren.

Entwicklungsprozesse “Software/Hardware in the Loop”

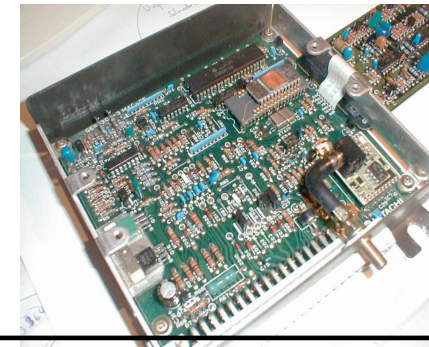
■ Begriffsdefinitionen:

■ HIL Hardware in the Loop

- Reale SG-Hard- und Software wird in simulierter Fahrzeugumgebung gespeist, d.h. mit rechnergenerierten Sensor- und Bussignalen angesteuert.

■ SIL Software in the Loop

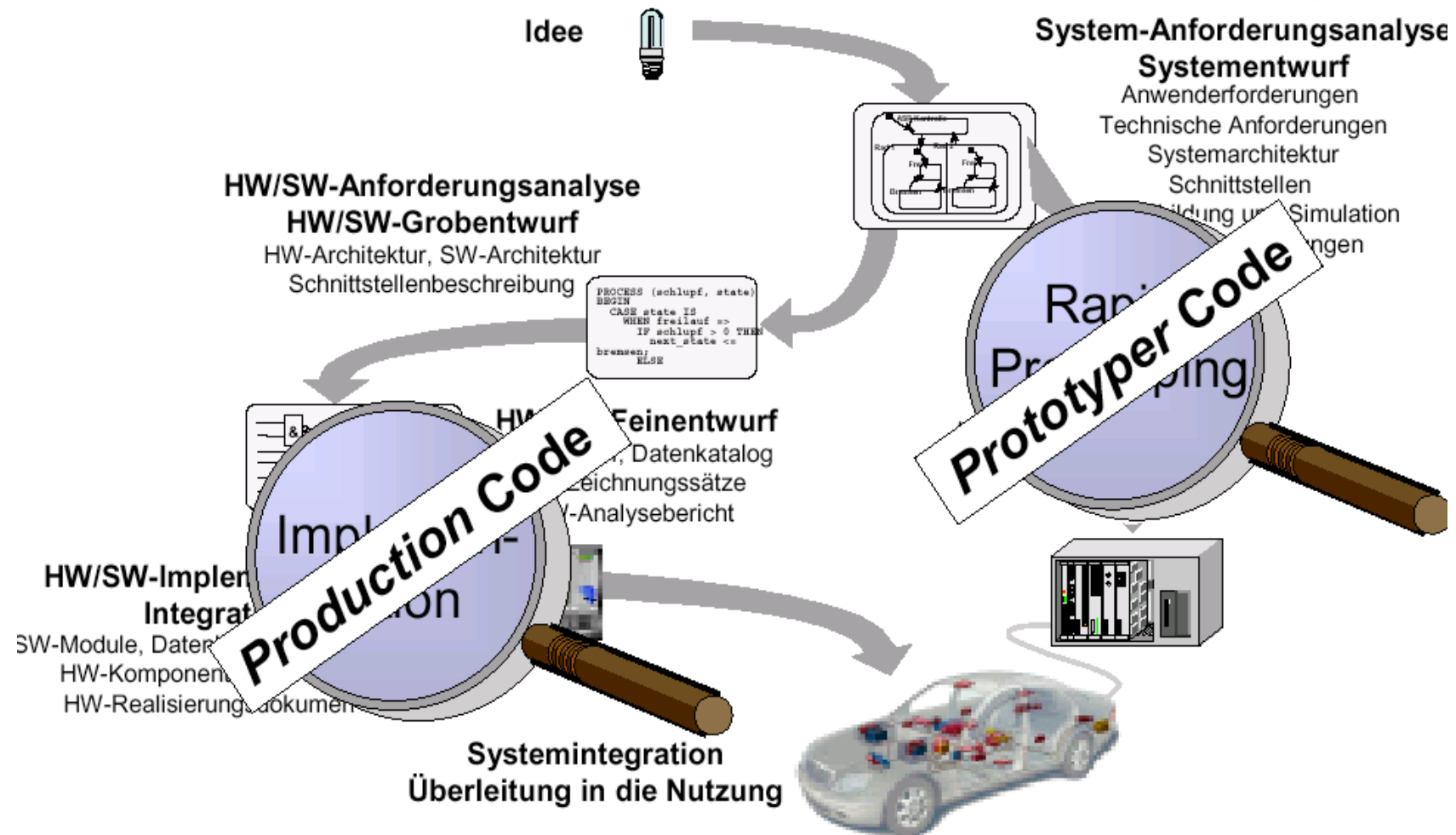
- SG-Software „läuft“ auf simuliertem SG, das von simulierter Fahrzeugumgebung gespeist wird.

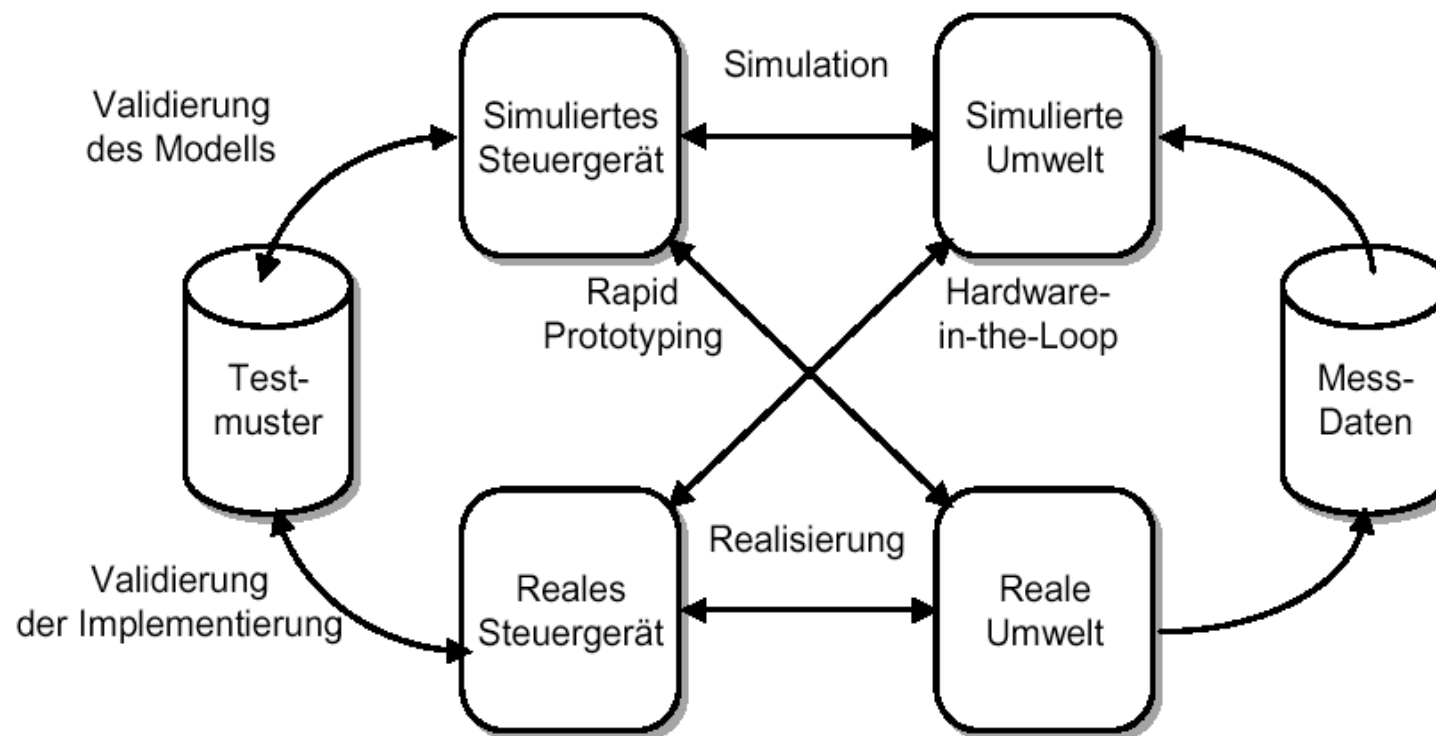


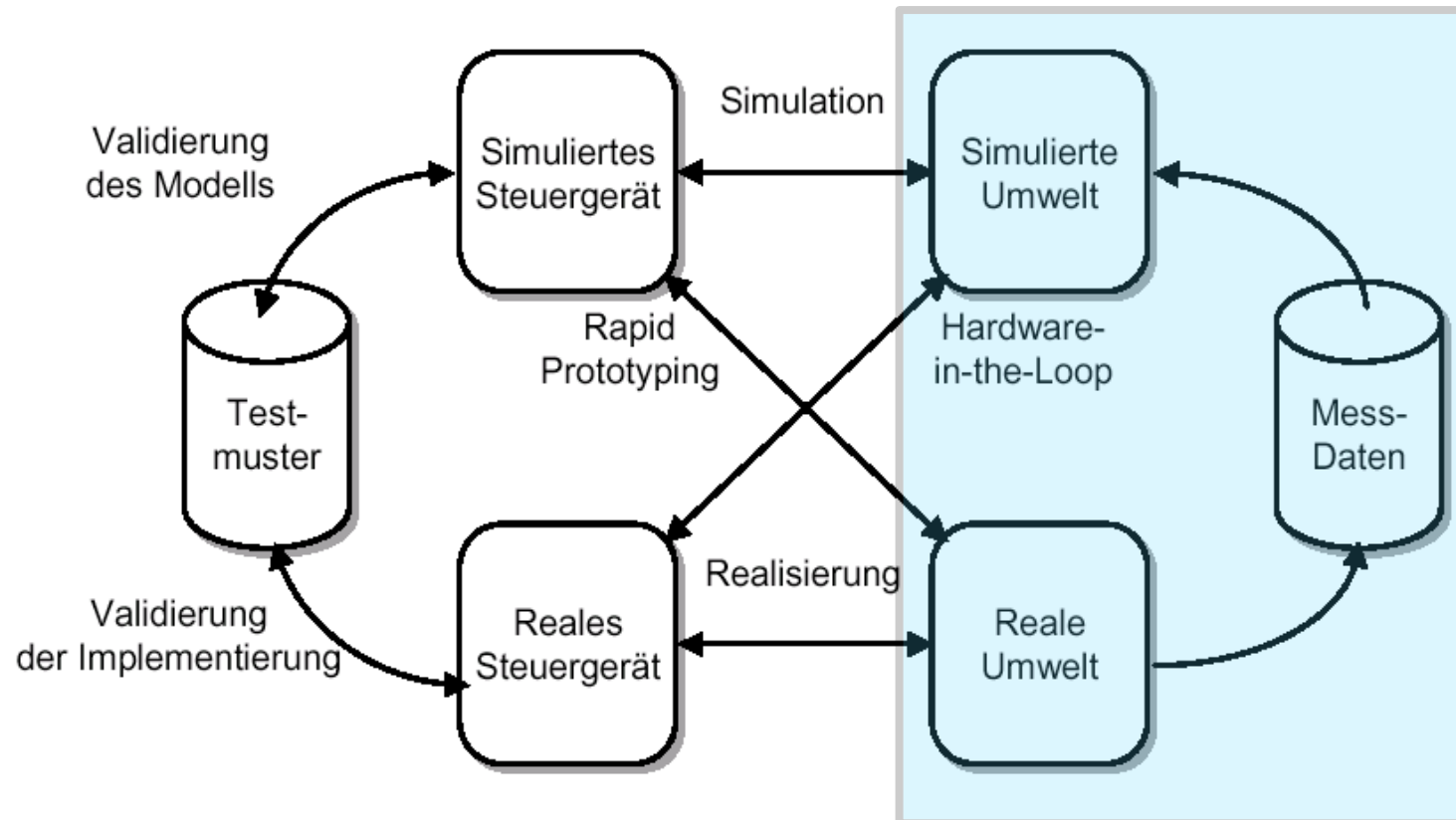
Fahrzeug / Umwelt

rd.

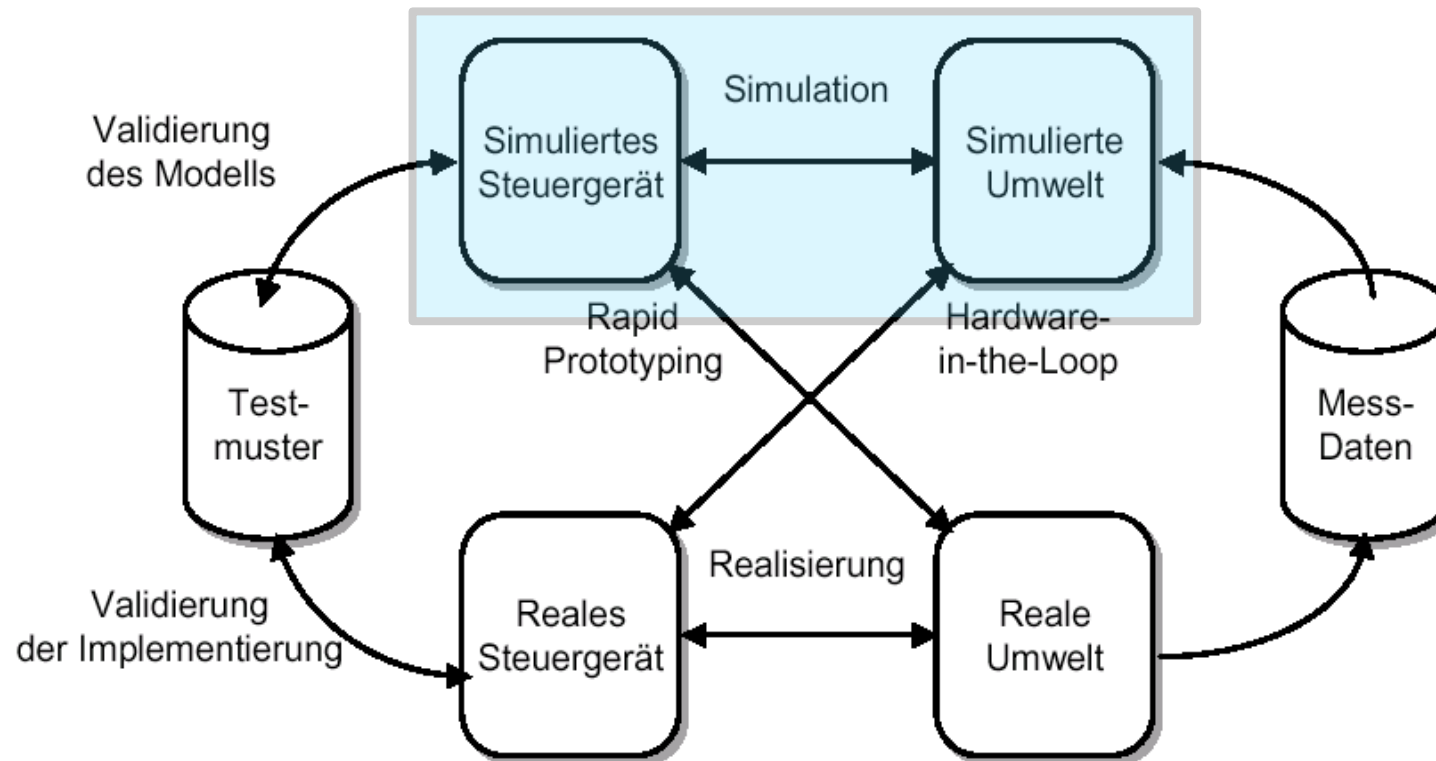
		Steuergerät	
		simuliert	real
simuliert		SIL	HIL
real		Prototyp	Fahrversuch



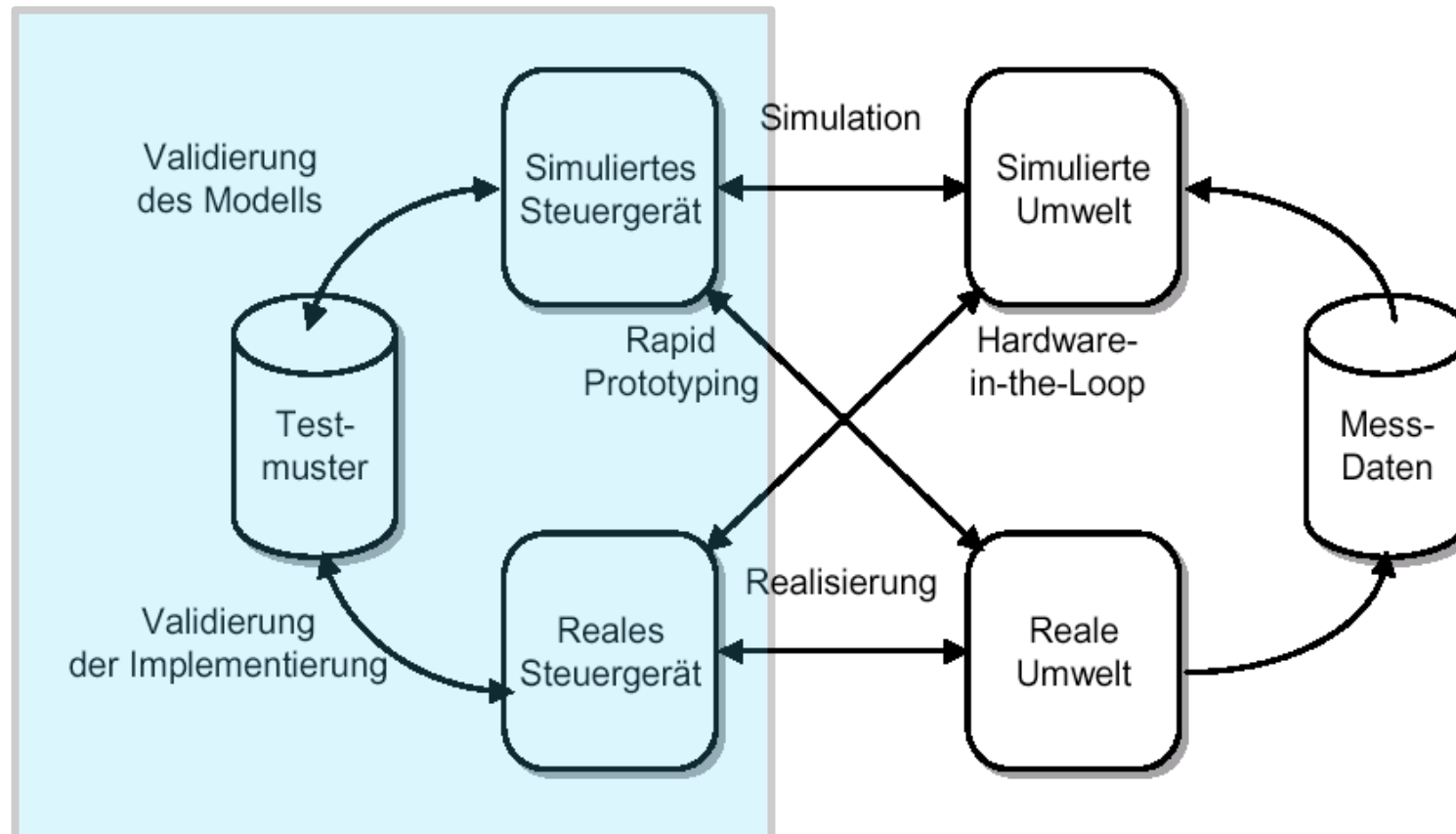




Erfassung von Messdaten zur Simulation der Umwelt

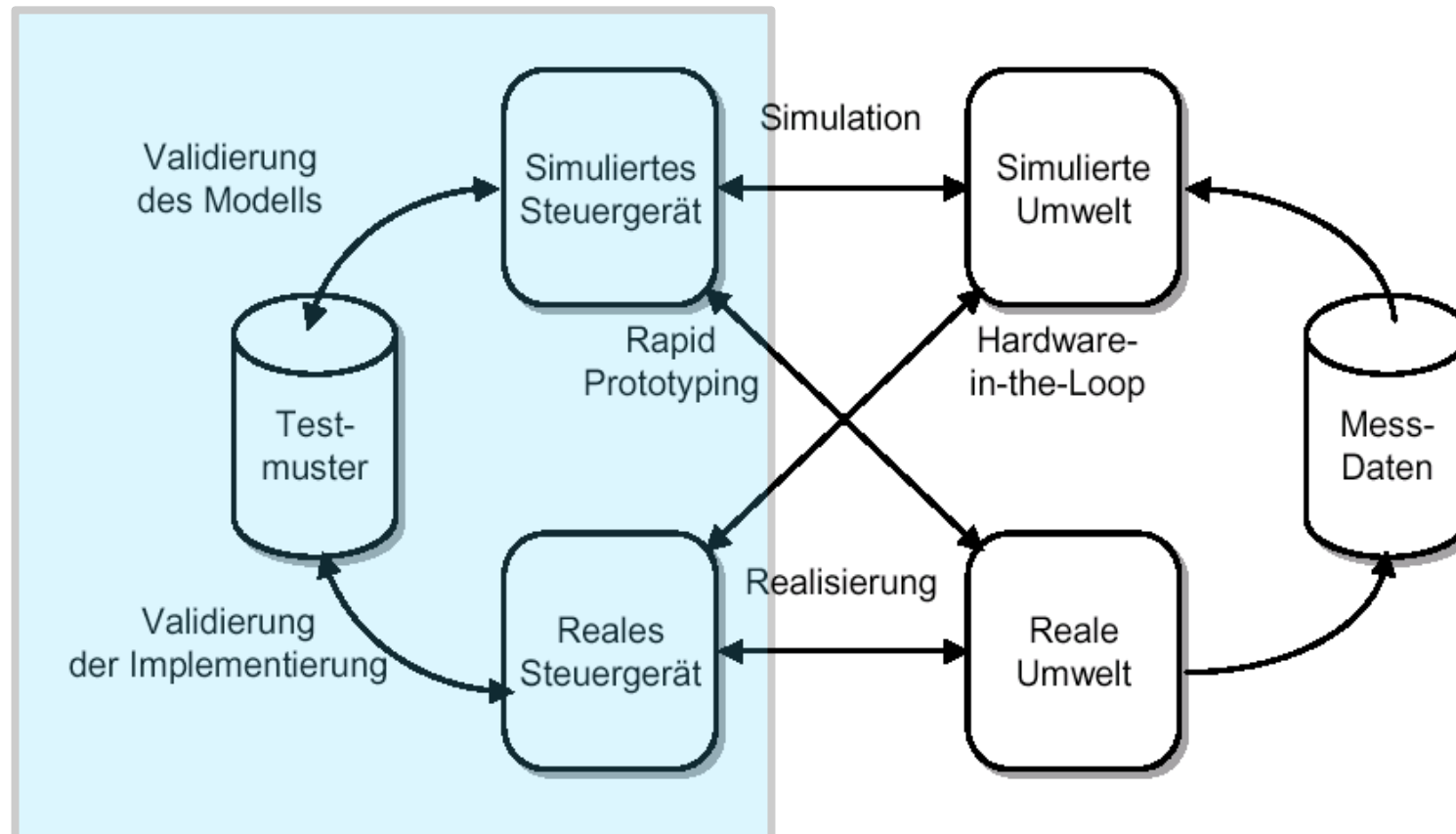


Simulation von Steuergerät und Umwelt: Model in the Loop (MIL), Software in the Loop (SIL)



Validierung des Modells: Test des Simulierten Steuergerätes

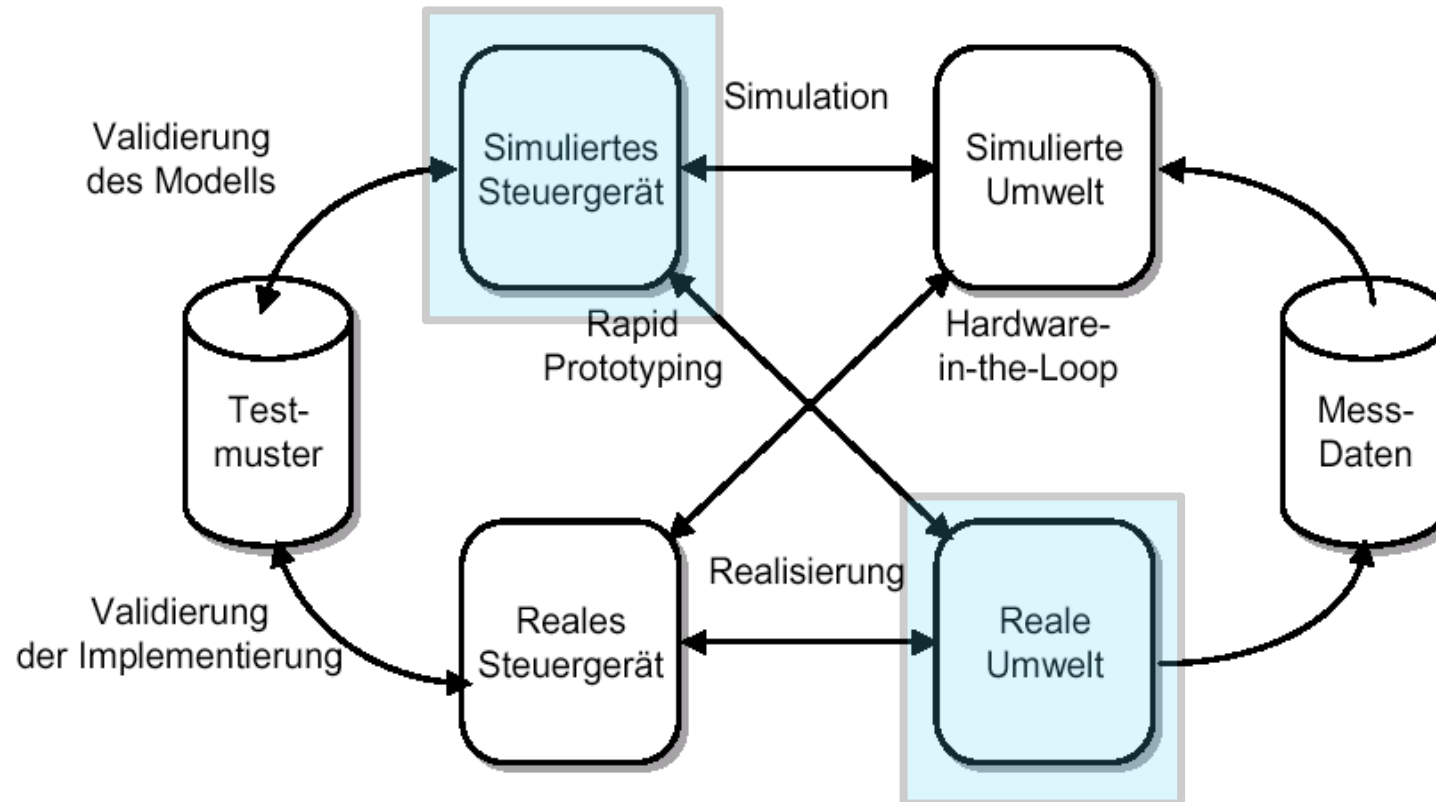
Validierung der Implementierung: Test des Realen Steuergerätes mit den



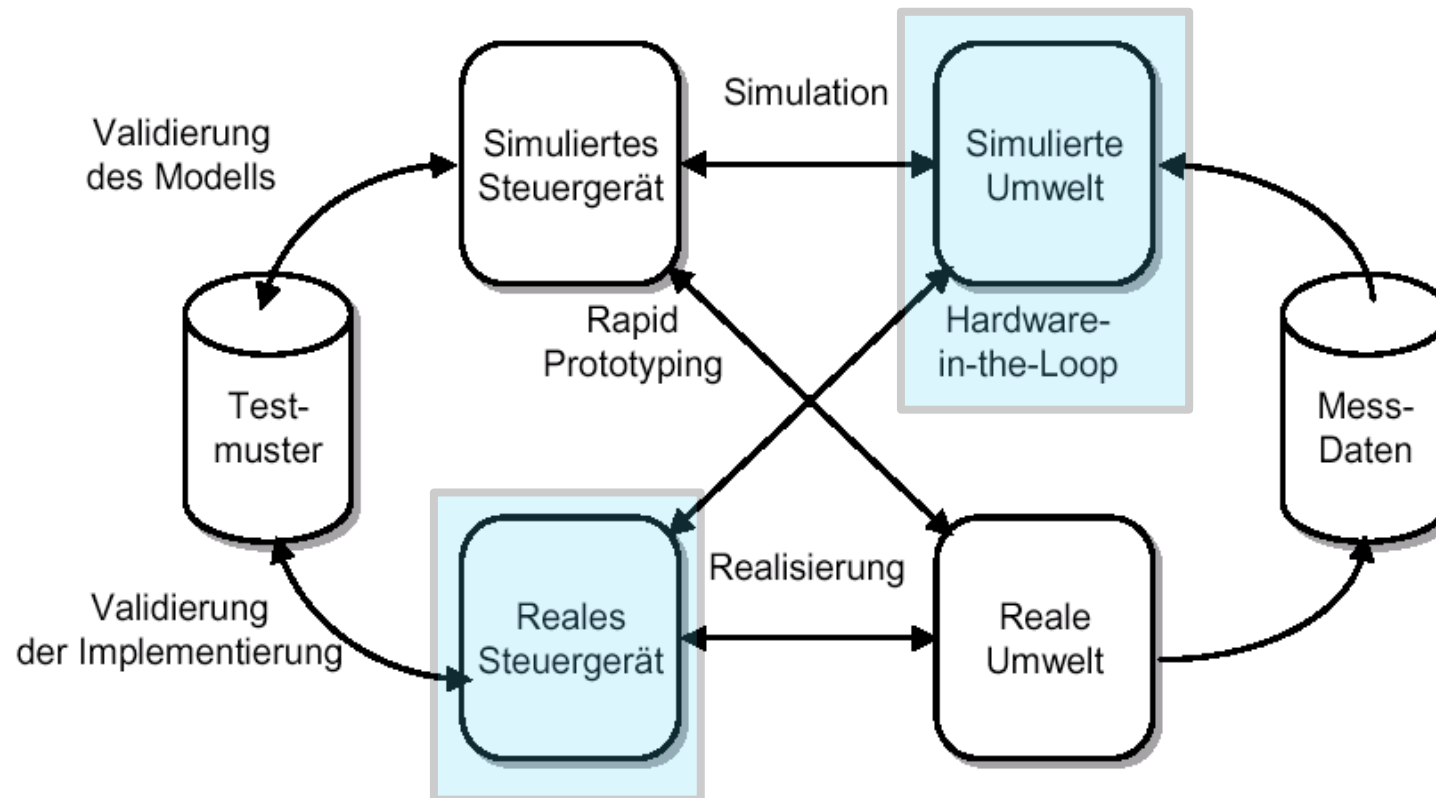
Validierung des Modells: Test des Simulierten Steuergerätes

Validierung der Implementierung: Test des Realen Steuergerätes mit den

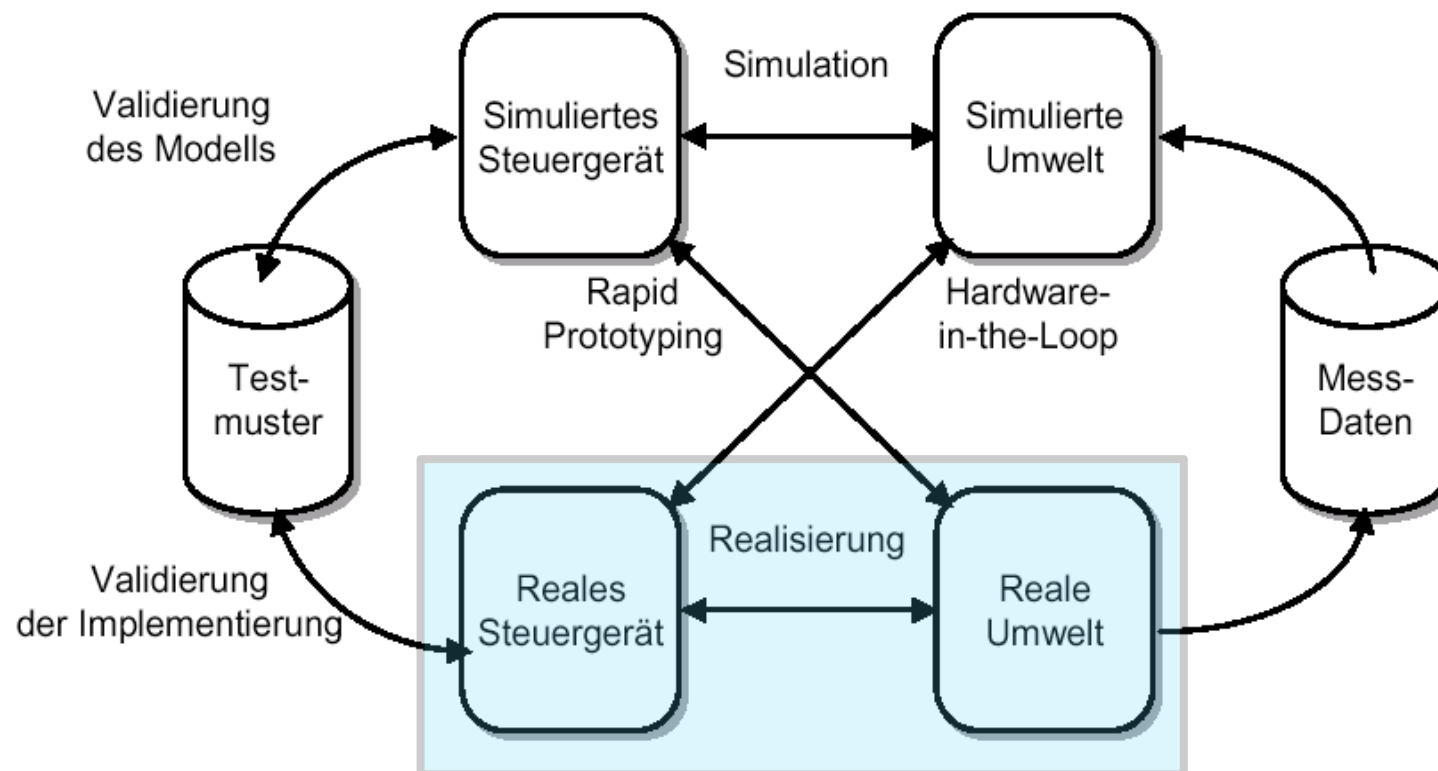
← Gleiche Testmuster (Testdaten)



Rapid Prototyping: Frühe Tests in realer Umgebung



Umfangreiche Tests des Steuergerätes: Hardware in the Loop (HIL)



Realisierung: Fahrversuch

Testvarianten im Integrationsprozess

- MIL Model in the Loop
- SIL Software in the Loop
- PIL Prozessor in the Loop
- HIL Hardware in the Loop
 - K-HIL Komponente
 - S-HIL System
- VIL Vehicle in the Loop
- EFP Erweiterte Funktionsprüfung

Entwicklungsprozesse “Software/Hardware in the Loop”

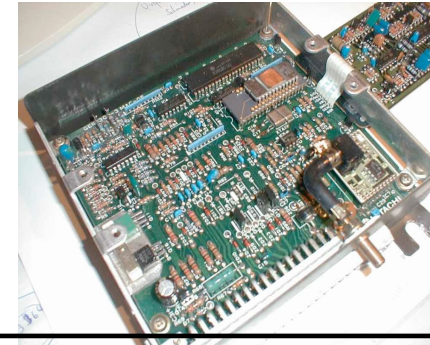
■ Begriffsdefinitionen:

■ HIL Hardware in the Loop

- Reale SG-Hard- und Software wird in simulierter Fahrzeugumgebung gespeist, d.h. mit rechnergenerierten Sensor- und Bussignalen angesteuert.

■ SIL Software in the Loop

- SG-Software „läuft“ auf simuliertem SG, das von simulierter Fahrzeugumgebung gespeist wird.

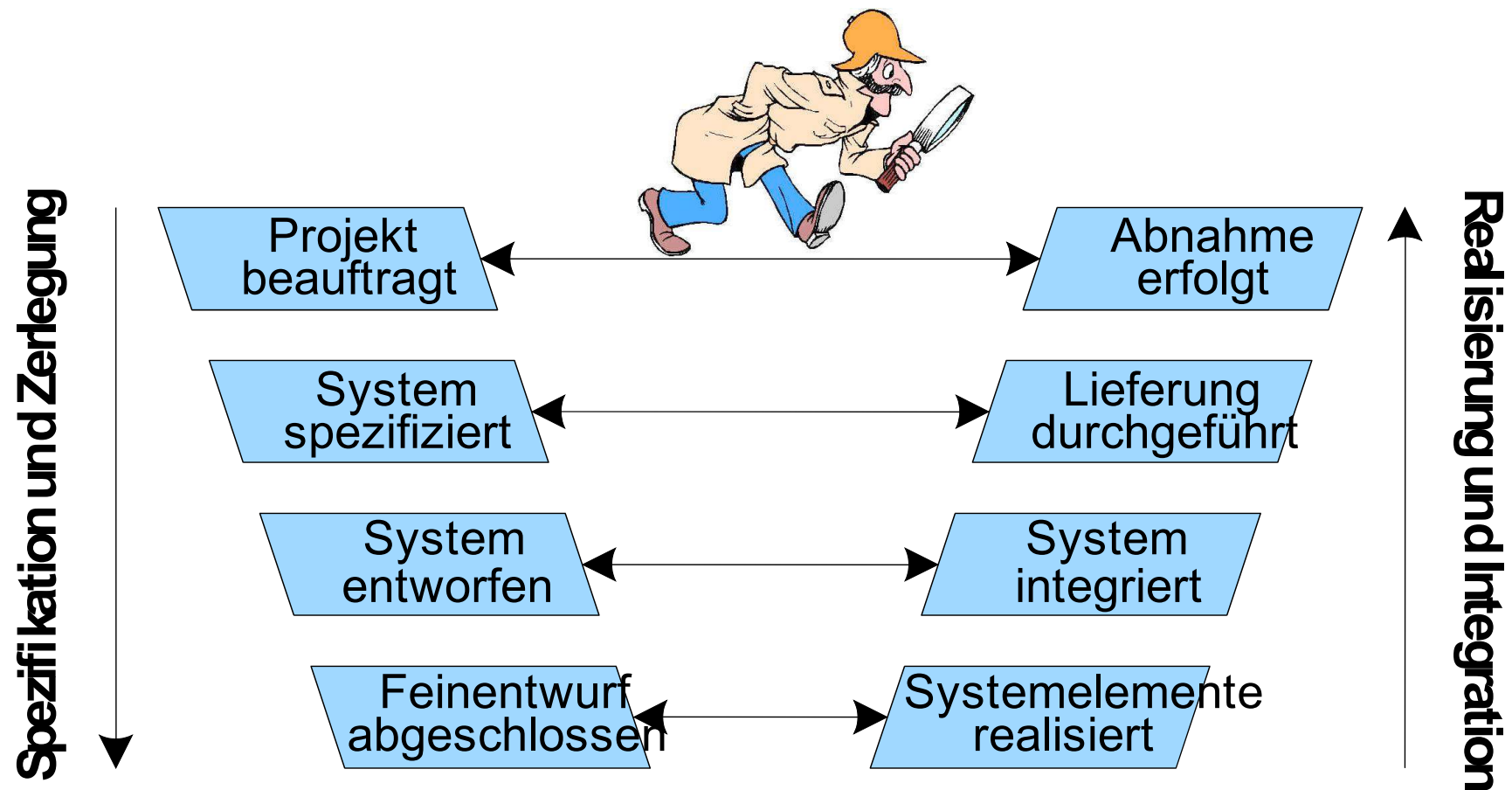


Fahrzeug / Umwelt

rd.

		Steuergerät	
		simuliert	real
	simuliert	MIL, SIL	PIL, HIL, VIL
	real	Prototyp	Fahrversuch

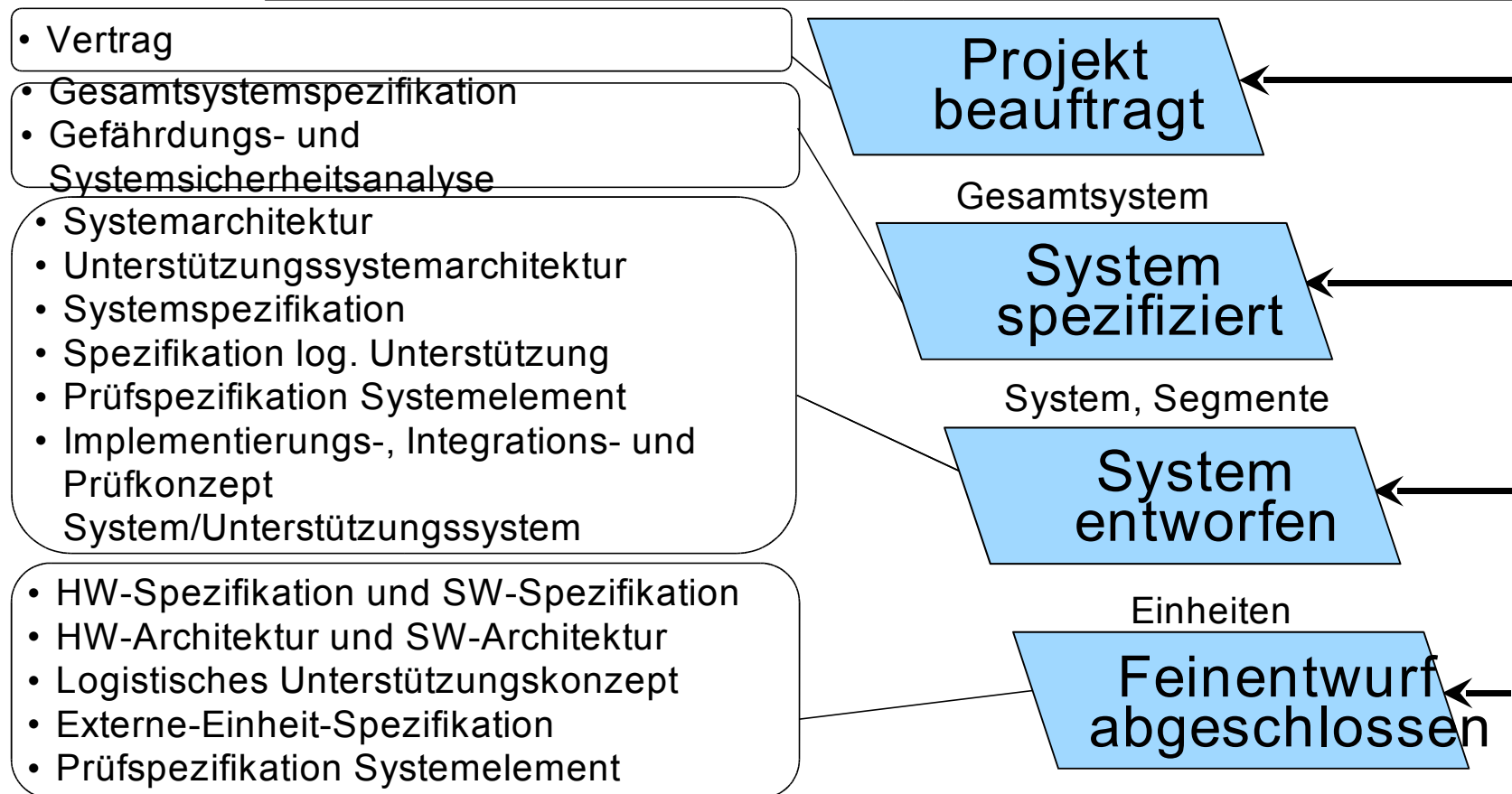
Verifizierung und Validierung



© Wolfgang Kranz: Systementwicklung Auftragnehmer

12

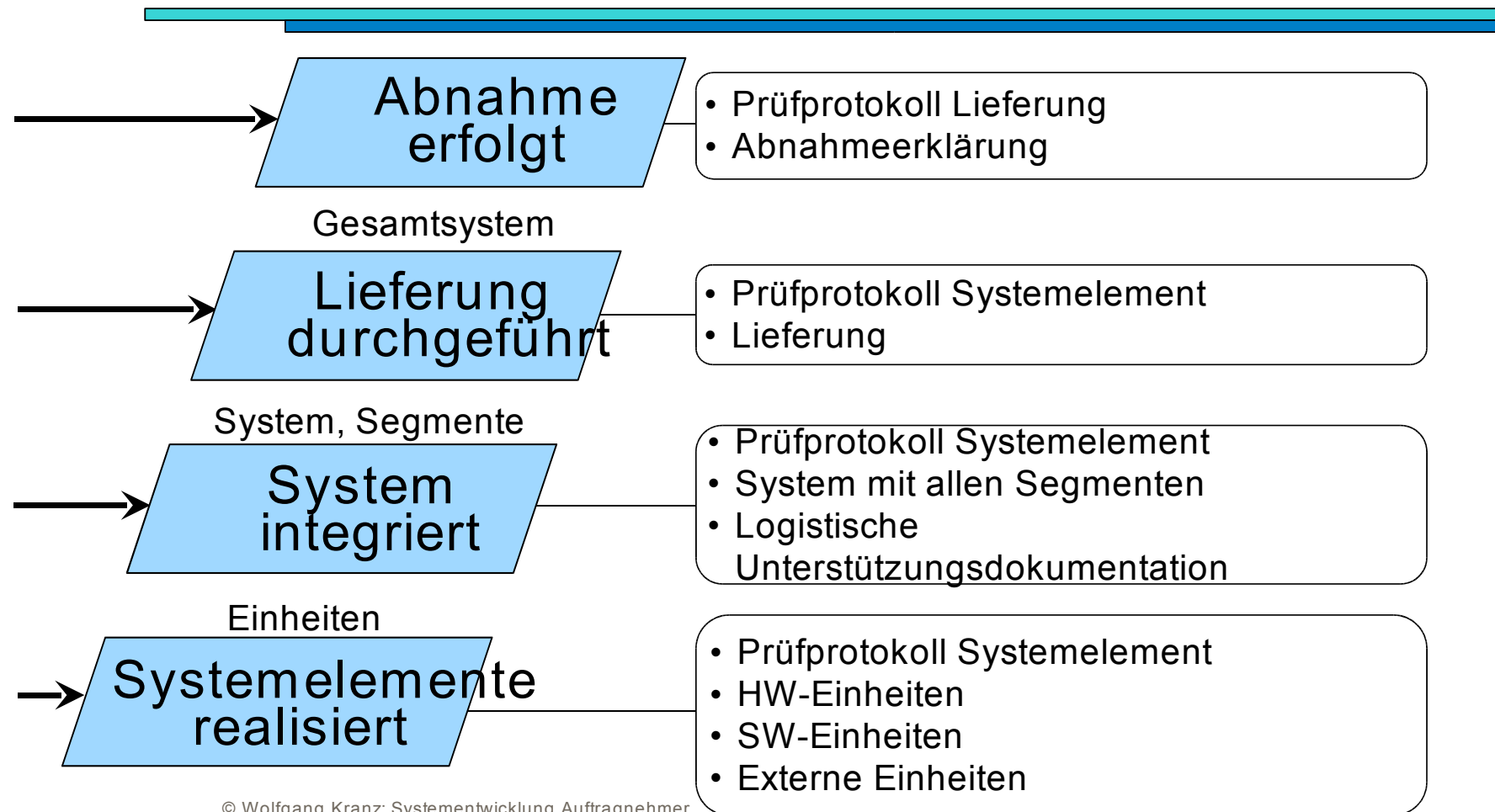
Entscheidungspunkte - Produkte Systementwicklung



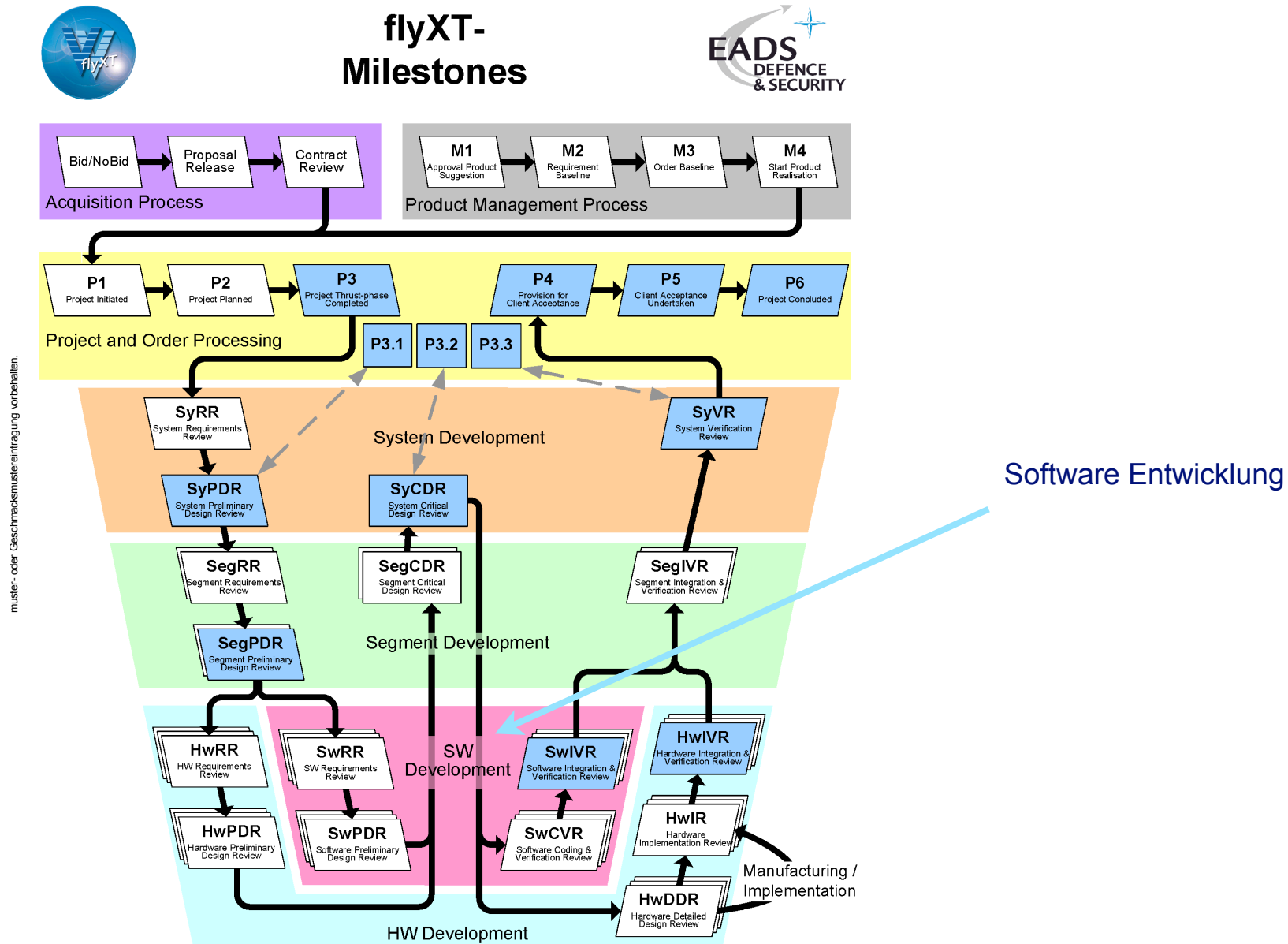
© Wolfgang Kranz: Systementwicklung Auftragnehmer

10

Entscheidungspunkte - Produkte Systementwicklung









V-Modell (5/7)

Softwaretechnik
Inst. PM

Aktueller Stand: V-Modell®XT (seit 2005)

Bereits eingesetzt: Adressverwaltung des Bundestags, Inpol-neu, Bordradar Eurofighter, ...

- Produktzentrierter Ansatz („Integriertes Produktmodell“)
 - **Produkte** (= Ergebnisse oder Zwischenergebnisse eines Projekts)
 - stehen im Mittelpunkt (> 100 Produkte in Produktgruppen geordnet)
 - haben definierte Abhängigkeiten
 - **Aktivitäten**
 - dienen dazu, Produkte zu erzeugen oder zu bearbeiten
(90 Aktivitäten in 13 Aktivitätsgruppen, die entsprechenden Produktgruppen zugeordnet sind)
 - **Rollen**
 - beschreiben zusammengehörige Aufgaben, Verantwortlichkeit + benötigte Fähigkeiten
- Submodelle, Abstraktionsebenen und Erzeugnisstruktur i.w. wie bei V-Modell 97
(zusätzliches Submodell: Problem- und Änderungsmanagement)

Früher: Teil des Konfigurationsmanagements

- 1. Einführung in die Softwaretechnik (Zitat aus Skript)
 - Einführung und Klärung von Grundbegriffen
 - den Unterschied von Softwareprodukten zu anderen Produkten verstehen und erklären können
 - den Softwarelebenszyklus kennen
 - Phasen, Aktivitäten, Produkte
 - Methoden und Konzepte
- 2. Vorgehensmodelle (Zitat aus Skript)
 - die prinzipiellen Phasen und Aktivitäten der Softwareentwicklung und ihre Produkte verstehen und erklären können
 - die Unterschiede, Vor- und Nachteile verschiedener Vorgehensmodelle kennen
 - die Schritte zur Vorgabe eines Prozessmodells kennen

Vorgehensmodell

Allgemeiner Vorgehensrahmen

- benennt und beschreibt allgemein die Art und Anordnung von Aktivitäten im Rahmen der SW-Entwicklung
- man unterscheidet
 - sequenzielle, streng phasenorientierte Modelle
 - nichtlineare evolutionäre Modelle
- wird in der Praxis für konkrete Projekte und Unternehmen näher spezifiziert

→ konkretes Prozessmodell festlegen

Prozessmodell

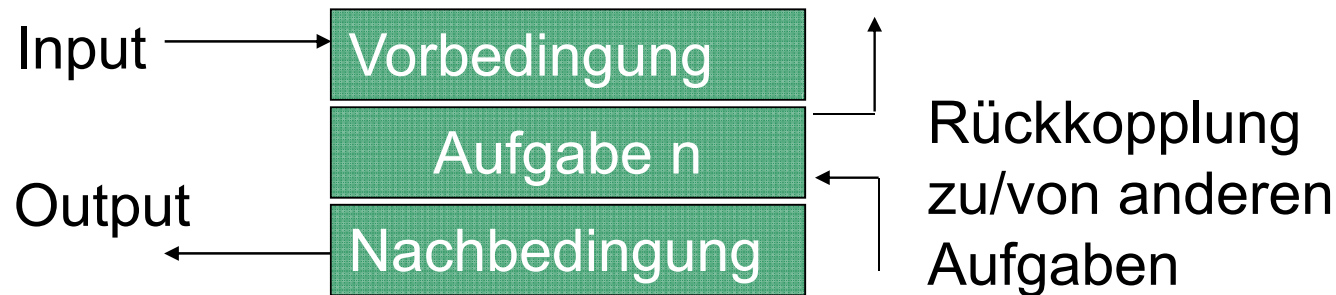
Konkrete Instanz eines Vorgehensmodells

- definiert die konkreten Aktivitäten, ihre Reihenfolge und ihre Produkte für die Durchführung eines Projektes
- Definition ist abhängig von
 - Projekttyp und
 - Unternehmenspolitik und -kompetenz
- die Aktivitäten müssen durch Methoden und Werkzeuge unterstützt werden

→ abhängig vom Vorgehensparadigma

Festlegung von Prozessmodellen

1. Definition der Aktivitäten (wer, was, wie, wann)



2. Auswahl von Methoden und Werkzeugen

→ erzeugen Produktmodelle

- strukturierte Methoden und Konzepte
- objektorientierte Methoden und Konzepte

- TU München - Fakultät für Informatik Lehrstuhl IV: Software & Systems Engineering (Prof. Manfred Broy)
- Vorlesung | Wintersemester 2010/2011 (2+1, 5 ETCS-Punkte)
- <http://www4.in.tum.de/lehre/vorlesungen/vgmse/ws1011/>
- Inhalt

Die Vorlesung vermittelt die grundlegenden Techniken und Methoden der Projektorganisation und des Projektmanagements für die Entwicklung großer Softwaresysteme.

■ Grundlagen Was ist ein Vorgehensmodell?

- Einordnung von Vorgehensmodellen in Prozesslandschaften

■ Übersicht Vorgehensmodelle

- Klassische Ansätze
- Agile Modelle/Methoden
- Strukturierte Modelle
- Weitere Modelle
- Werkzeugunterstützung

■ Lebenszyklus von Vorgehensmodellen Phasenübersicht (im Folgenden vertieft)

■ Analyse

- Analyse und Optimierung von Prozessen
- Stakeholder und Rollen
- Vorgänge/Abläufe
- Artefakte/Produkte
- Strukturierung von Prozessen

■ Konstruktion und Anpassung

- Konstruktionsoptionen
- Vorgehensmetamodelle
- Umsetzung und Bereitstellung

■ Einführung

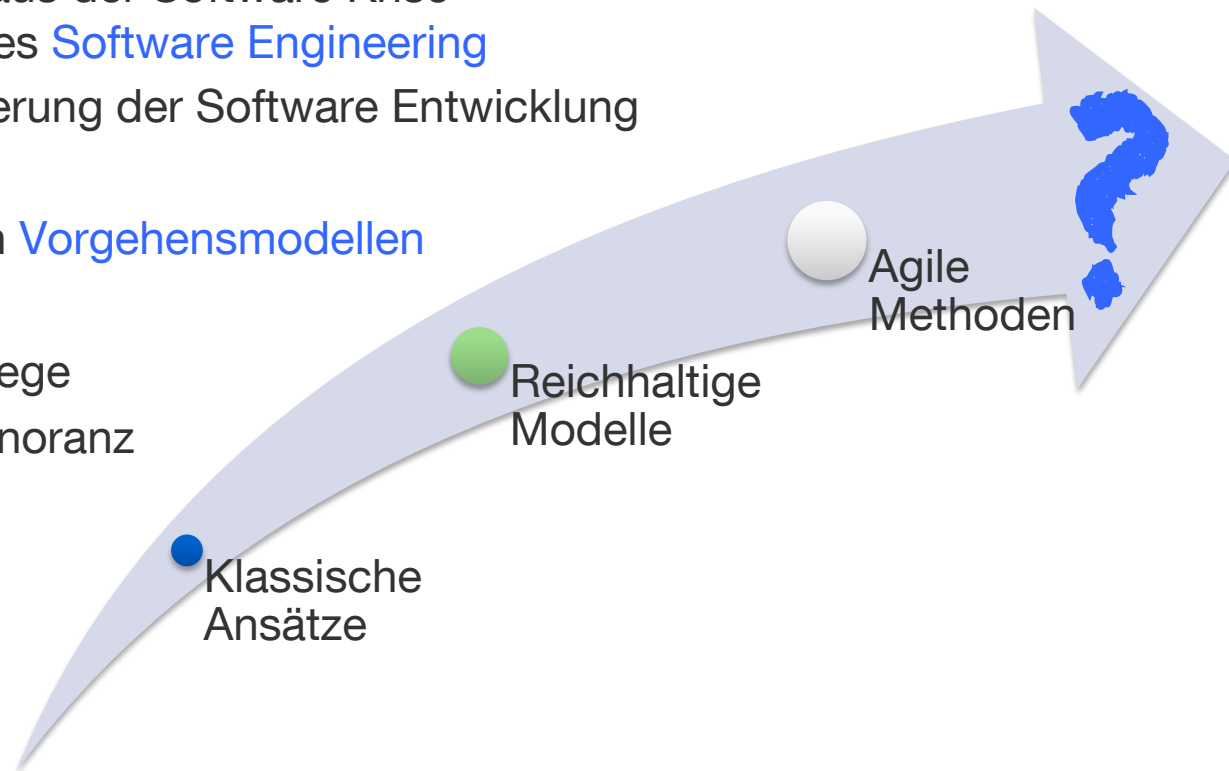
- Einführungsstrategien
- Planung und Maßnahmen
- Schulungen
- Etablieren von Feedbackschleifen

■ Weiterentwicklung und Pflege

- Prozessreife
- Audits, Assessments und Zertifizierung
- Planung von Prozessverbesserungen
- Ausgewählte Modelle und Methoden

Inhalte und Ziele der Vorlesung (1)

- Vorgehensmodelle existieren seit den frühen 70'ern
 - Ziel: „Raus aus der Software Krise“
→ Geburt des **Software Engineering**
 - Systematisierung der Software Entwicklung
- Entwicklung von **Vorgehensmodellen**
 - Inflation
 - Glaubenskriege
 - Trend zur Ignoranz



Inhalte und Ziele der Vorlesung (2)

- Ziele
 - Organisations- und Projektprozesse verstehen
 - Vorgehensmodelle verstehen
 - Prozesse planen, abbilden, definieren und umsetzen
 - Handhabung von Werkzeugen und Infrastrukturen für Prozessanpassungen erlernen

- Praktischer Anwendungskontext
 - Prozessoptimierendes Projektmanagement
 - Projekt- und Teamorganisation
 - Effektivitätsfeststellung und Verbesserung
 - Prozessanalyse und Beratung

Inhalte und Ziele der Vorlesung (3)

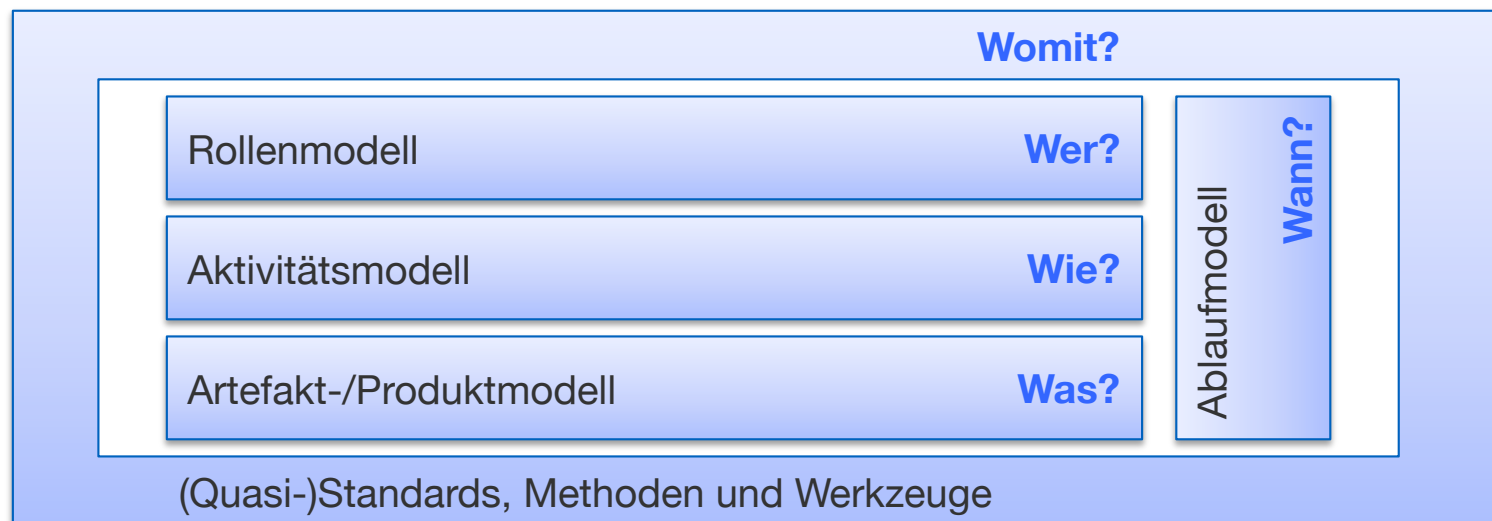
1. Grundlagen
→ *Begriffsklärung und Einordnung*
2. Übersicht Vorgehensmodelle
→ *Aufstellung/Bewertung existierender Ansätze*
3. Lebenszyklus von Vorgehensmodellen
→ *Überblick über die Phasen (Vertiefung im Anschluss)*
4. Analyse
→ *Analyse und Optimierung von Prozessen, Stakeholder, Modelle etc.*
5. Konstruktion und Anpassung
→ *Metamodellierung und Modellierung von Vorgehensmodellen*
6. Einführung
→ *Einführungsmaßnahmen, Planung, Schulung etc.*
7. Weiterentwicklung und Pflege
→ *Prozessreife, Planung, Übersicht Reifegradmodelle*

Was ist ein Vorgehensmodell?

Definition

Ein Vorgehensmodell beschreibt systematische, ingenieurmäßige und quantifizierbare Vorgehensweisen, um Aufgaben einer bestimmten Klasse wiederholbar zu lösen.

Submodelle

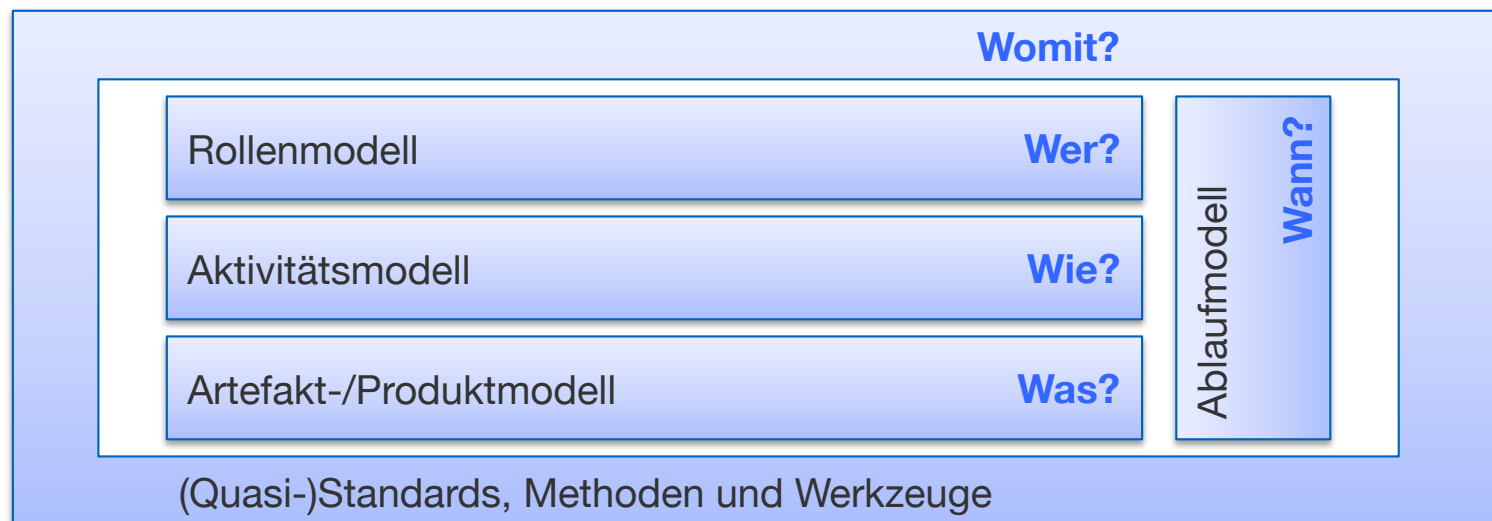


Was ist ein Vorgehensmodell?

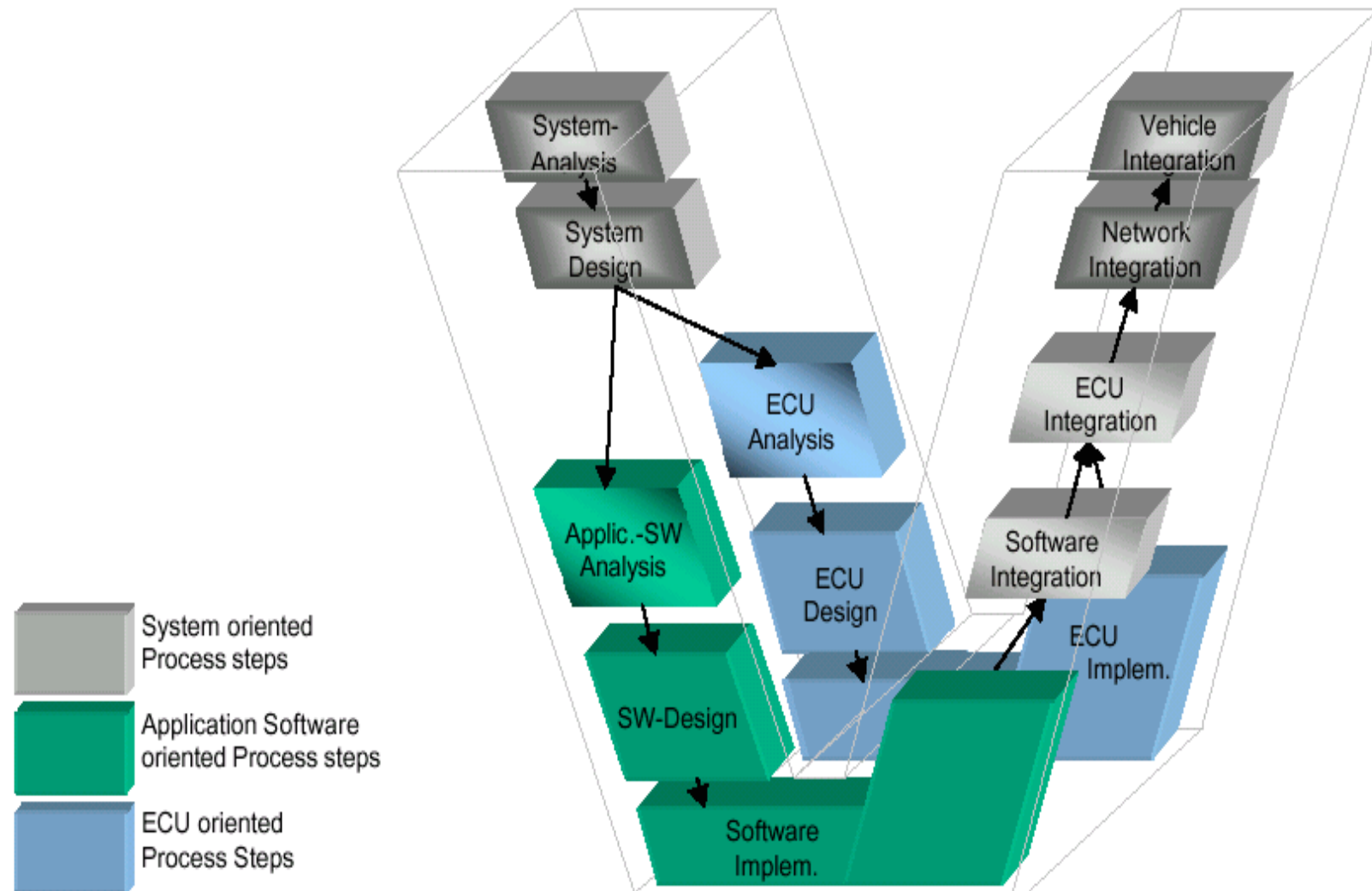
Definition

Ein Vorgehensmodell beschreibt systematische, ingenieurmäßige und quantifizierbare Vorgehensweisen, um Aufgaben einer bestimmten Klasse wiederholbar zu lösen.

Submodelle **1. Definition der Aktivitäten (wer, was, wie, wann)**



- Wer?
Rollenorientierte Sicht:
Wer ist für welche Arbeitsergebnisse (Produkte, Work Products, Artefakte) verantwortlich?
- Was?
Produktorientierte Sicht:
Was sind die zu erarbeitenden Produkte?
- Wie?
Aktivitätenorientierte Sicht:
Wie werden die Produkte erarbeitet?
- Wann?
Phasenorientierte Sicht:
Wann werden welche Arbeitsschritte durchgeführt?
Meilensteinorientierte Sicht:
Wann werden welche Produkte fertiggestellt und geprüft?
- Womit?
Methoden- und werkzeugorientierte Sicht:
Womit (Methoden und Werkzeuge) werden die Produkte erarbeitet?



- Wer?
Rollenorientierte Sicht:
Der SW-Architekt ist für die SW-Architektur verantwortlich
- Was?
Produktorientierte Sicht:
Die SW-Architektur beschreibt die Zerlegung in Komponenten, deren Schnittstellen und Wechselwirkungen
- Wie?
Aktivitätenorientierte Sicht:
Die SW-Architektur wird nach funktionalen Gesichtspunkten unter Berücksichtigung von Modularität und Kapselung entworfen
- Wann?
Phasenorientierte Sicht:
Die SW-Architektur wird in der Phase SW-Design entworfen.
Meilensteinorientierte Sicht:
Die SW-Architektur wird zum Meilenstein SW-Design Review fertiggestellt und geprüft
- Womit?
Methoden- und werkzeugorientierte Sicht:
Die SW-Architektur wird entsprechend den firmeninternen Entwurfsrichtlinien mit unterstützenden Werkzeugen erstellt.



ISO DIS 26262 Road vehicles - Functional safety
Produkte, Prozessschritte, Rollen
Prinzipielle Darstellung der Vorgehensweise
Ausarbeitung in Workproducts-26262.XLS (Eli, Hipp)

Evidence of a quality management system	I
Organization specific rules and processes for functional safety	O I
Evidence that the persons assigned	O I
Evidence of an operational E/E quality management system	O I I
Overall project plan (refined)	I/O
Dependencies on other activities, including other safety activities	I
Safety plan	O I/O I I/O I I I I
Safety case	O
Results of the confirmation measures	O
Confirmation plan	O
Functional safety assessment plan	O
Evidence of a field monitoring process	O

der

ISO DIS 26262 Road vehicles - Functional safety
 Produkte, Prozessschritte, Rollen
 Prinzipielle Darstellung der Vorgehensweise
 Ausarbeitung in Workproducts-26262.XLS (Ell, Hipp)

	Teil 2: Management of functional safety			Teil 6: Product development: software level						
	5	6	7	5	6	7	8	9	10	11
Evidence of a quality management system	—									
Organization specific rules and processes for functional safety	O	I								
Evidence that the persons assigned	O	I								
Evidence of an operational E/E quality management system	O	I	I							
Overall project plan (refined)		I/O								
Dependencies on other activities, including other safety activities		I								
Safety plan		O			I/O	I	I/O	I	I	I
Safety case		O								
Results of the confirmation measures		O								
Confirmation plan		O								
Functional safety assessment plan		O								
Evidence of a field monitoring process			O							

ISO DIS 26262 Road vehicles - Functional safety
 Produkte, Prozessschritte, Rollen
 Prinzipielle Darstellung der Vorgehensweise
 Ausarbeitung in Workproducts-26262.XLS (Eil, Hipp)

Evidence of a quality management system
 Organization specific rules and processes for functional safety
 Evidence that the persons assigned
 Evidence of an operational E/E quality management system
 Overall project plan (refined)
 Dependencies on other activities, including other safety activities
 Safety plan
 Safety case
 Results of the confirmation measures
 Confirmation plan
 Functional safety assessment plan
 Evidence of a field monitoring process

Teil 2: Management of functional safety									
5 Overall safety management									
6 Safety management during development of the item									
7 Safety management after release for production									
Teil 6: Product development: software level									
5 Initiation of product development at the software level									
6 Specification of software safety requirements									
7 Software architectural design									
8 Software unit design and implementation									
9 Software unit testing									
10 Software integration and testing									
11 Verification of software safety requirements									
Rollen									
< Qualitätsmanagement									
Safetymanagement									
Projektmanagement									
Softwareentwicklung									

- Evidence of a quality management system
- Organization specific rules and processes for functional safety
- Evidence that the persons assigned
- Evidence of an operational E/E quality management system
- Overall project plan (refined)
- Dependencies on other activities, including other safety activities
- Safety plan
- Safety case
- Results of the confirmation measures
- Confirmation plan
- Functional safety assessment plan
- Evidence of a field monitoring process

Safety plan	O	I/O	I	I/O	I	I	I	V	M
Safety case	O							V	
Results of the confirmation measures	O							V	
Confirmation plan	O							V	
Functional safety assessment plan	O							V	
Evidence of a field monitoring process	O							V	

ISO DIS 26262 Road vehicles - Functional safety
 Produkte, Prozessschritte, Rollen
 Prinzipielle Darstellung der Vorgehensweise
 Ausarbeitung in Workproducts-26262.XLS (Eil, Hipp)

Evidence of a quality management system
 Organization specific rules and processes for functional safety
 Evidence that the persons assigned
 Evidence of an operational E/E quality management system
 Overall project plan (refined)
 Dependencies on other activities, including other safety activities
 Safety plan
 Safety case
 Results of the confirmation measures
 Confirmation plan
 Functional safety assessment plan
 Evidence of a field monitoring process

Teil 2: Management of functional safety			Teil 6: Product development: software level			Rollen		
5 Overall safety management			5 Initiation of product development at the software level			< Qualitätsmanagement		
6 Safety management during development of the item			6 Specification of software safety requirements			Safetymanagement		
7 Safety management after release for production			7 Software architectural design			Projektmanagement		
			8 Software unit design and implementation			Softwareentwicklung		
			9 Software unit testing					
			10 Software integration and testing					
			11 Verification of software safety requirements					
Evidence of a quality management system	O	I				V	M	
Organization specific rules and processes for functional safety	O	I					V	
Evidence that the persons assigned	O	I					V	
Evidence of an operational E/E quality management system	O	I					V	
Overall project plan (refined)	I/O						M	
Dependencies on other activities, including other safety activities	I						M	
Safety plan	O		I/O	I	I/O	I	I	M
Safety case	O						V	
Results of the confirmation measures	O						V	
Confirmation plan	O						V	
Functional safety assessment plan	O						V	
Evidence of a field monitoring process	O						V	

ISO DIS 26262 Road vehicles - Functional safety
 Produkte, Prozessschritte, Rollen
 Prinzipielle Darstellung der Vorgehensweise
 Ausarbeitung in Workproducts-26262.XLS (Eil, Hipp)

Evidence of a quality management system
 Organization specific rules and processes for functional safety
 Evidence that the persons assigned
 Evidence of an operational E/E quality management system
 Overall project plan (refined)
 Dependencies on other activities, including other safety activities
 Safety plan
 Safety case
 Results of the confirmation measures
 Confirmation plan
 Functional safety assessment plan
 Evidence of a field monitoring process

Teil 2: Management of functional safety			Teil 6: Product development: software level			Rollen		
5 Overall safety management			5 Initiation of product development at the software level			< Qualitätsmanagement		
6 Safety management during development of the item			6 Specification of software safety requirements			Safetymanagement		
7 Safety management after release for production			7 Software architectural design			Projektmanagement		
			8 Software unit design and implementation			Softwareentwicklung		
			9 Software unit testing					
			10 Software integration and testing					
			11 Verification of software safety requirements					
Evidence of a quality management system	O	I				V	M	
Organization specific rules and processes for functional safety	O	I					M	
Evidence that the persons assigned	O	I					M	
Evidence of an operational E/E quality management system	O	I					M	
Overall project plan (refined)	I/O						V	M
Dependencies on other activities, including other safety activities	I							
Safety plan	O		I/O	I	I/O	I	I	M
Safety case	O						V	
Results of the confirmation measures	O						V	
Confirmation plan	O						V	
Functional safety assessment plan	O						V	
Evidence of a field monitoring process	O						V	