INFORMATIK ▪ CONSULTING ▪ SYSTEMS AG

Vorlesung
Automotive Software Engineering

Teil 6 SW-Entwicklung (3)

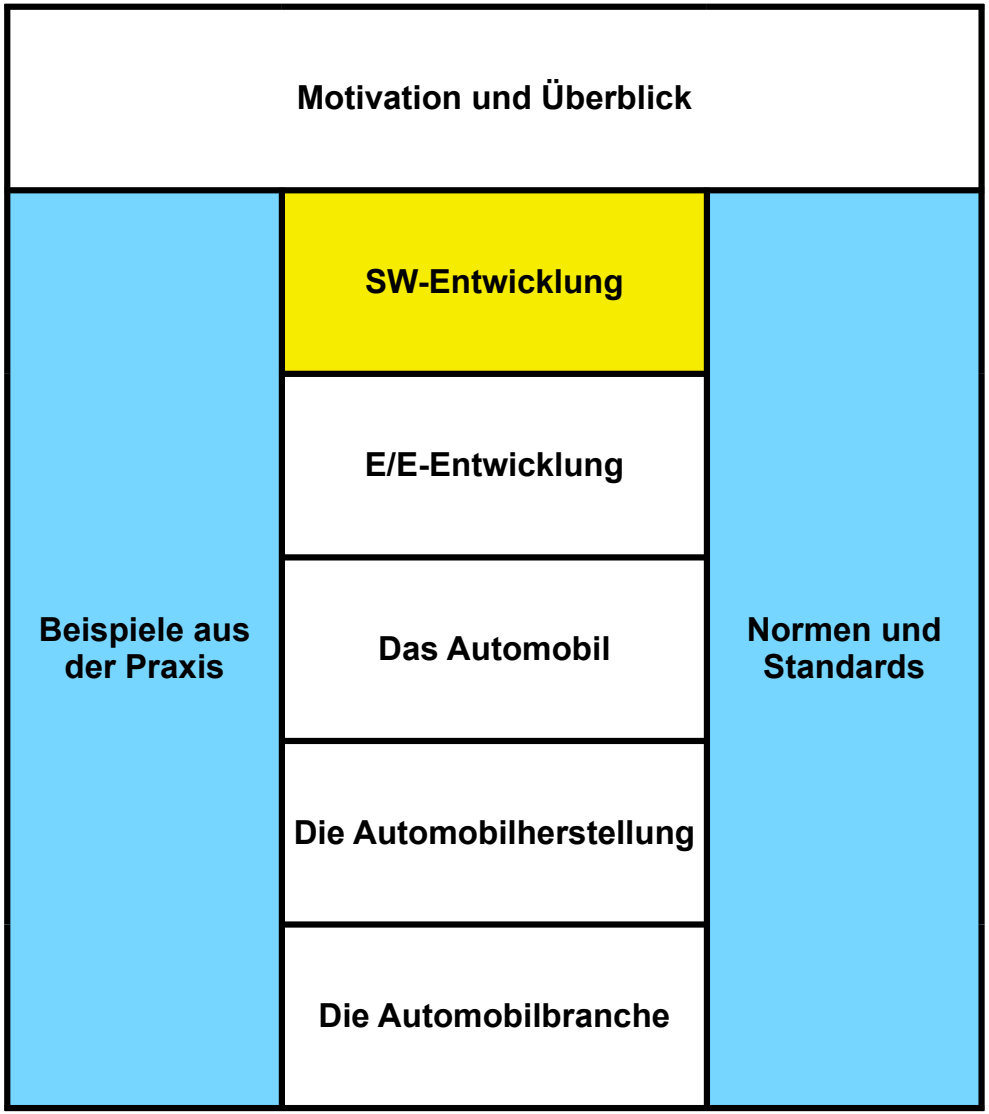
Sommersemester 2014

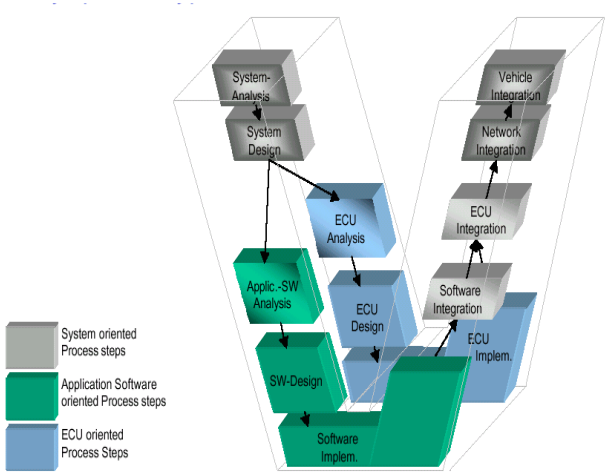
Prof. Dr. rer. nat. Bernhard Hohlfeld

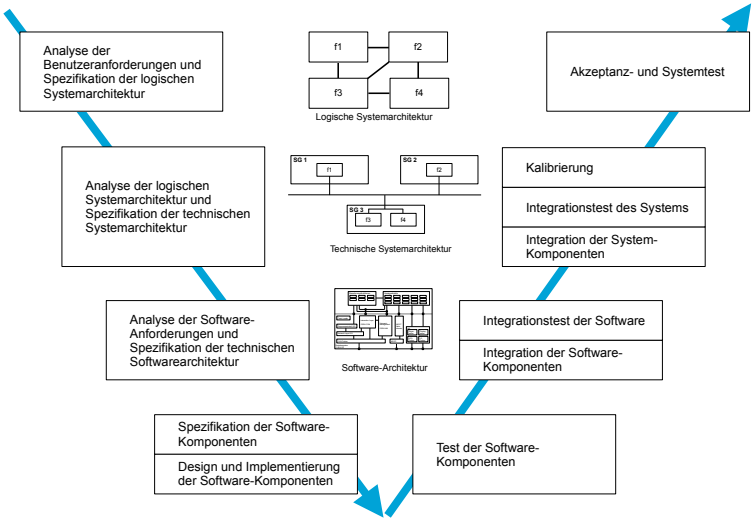
Bernhard.Hohlfeld@mailbox.tu-dresden.de

Technische Universität Dresden, Fakultät Informatik
Honorarprofessur Automotive Software Engineering

Vorlesung Automotive Software Engineering







Lernziele
SW-Entwicklung

- 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**

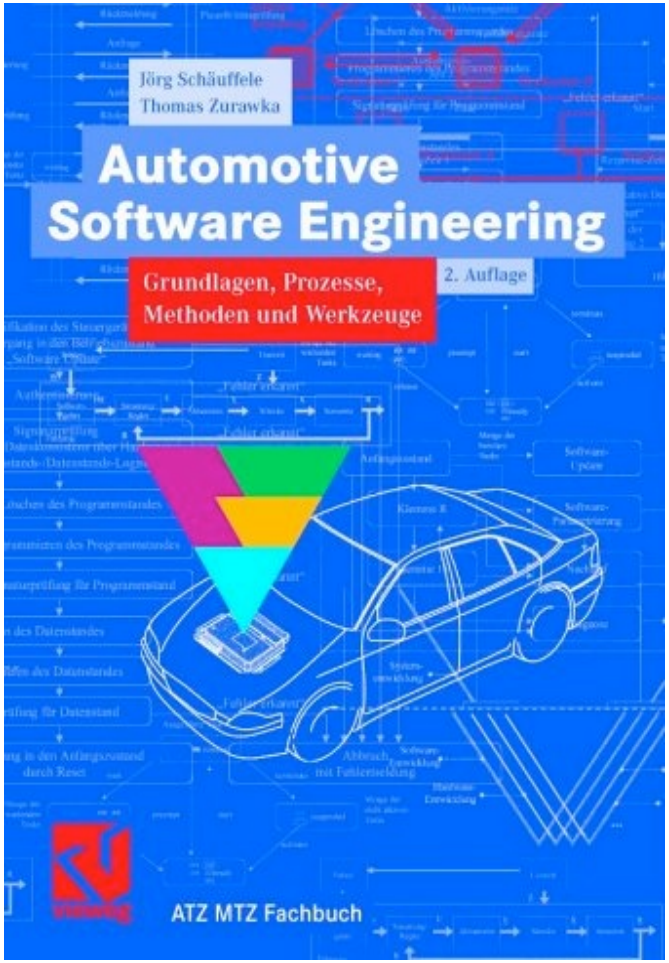
6. SW-Entwicklung / 3. Unterstützungsprozesse
Unterstützungsprozesse für die Embedded Software Entwicklung

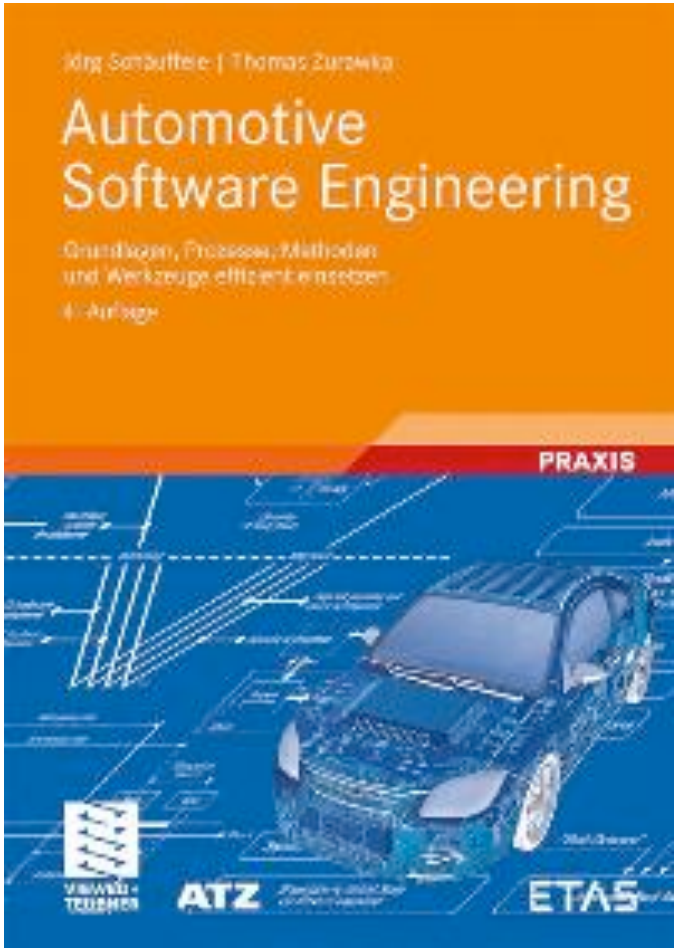
1. Vorgehensmodelle und Standards
2. Konfigurationsmanagement
3. Projektmanagement
4. Lieferantenmanagement
5. Anforderungsmanagement
6. Qualitätssicherung

Quelle

Kapitel 3

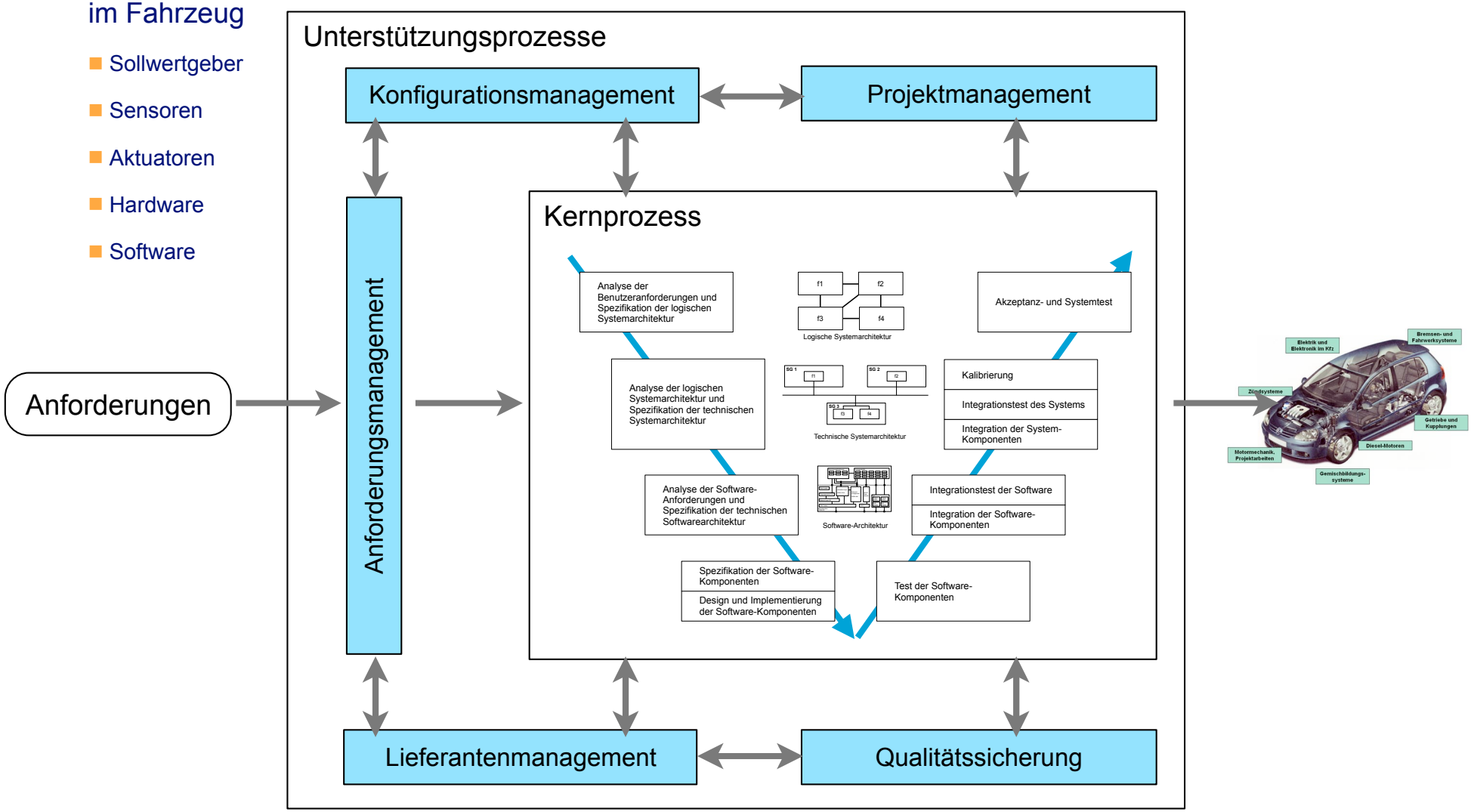
Die Abbildungen werden mit freundlicher Genehmigung der Autoren verwendet.





Unterstützungsprozesse für die Entwicklung von elektronischen Systemen und Software

- Weitgehend unabhängig von Software
- Anwendbar auf allen Systemebenen im Fahrzeug
  - Sollwertgeber
  - Sensoren
  - Aktuatoren
  - Hardware
  - Software



6. SW-Entwicklung / 3. Unterstützungsprozesse
Unterstützungsprozesse für die Embedded Software Entwicklung

1. Vorgehensmodelle und Standards

2. Konfigurationsmanagement

3. Projektmanagement

4. Lieferantenmanagement

5. Anforderungsmanagement

6. Qualitätssicherung

Vorgehensmodelle und Standards

- CMMI Capability Maturity Model Integration
  - http://www.sei.cmu.edu/cmmi
- SPICE Software Process Improvement and Capability Determination (ISO/IEC 15504-1)
  - http://www.sqi.gu.edu.au/spice/
- V-Modell
  - V-Model 1997
    - http://www.v-modell.iabg.de
  - V-Modell XT 2005
    - http://www.CIO.bund.de/DE/IT-Methoden/V-Modell_XT/v-modell_xt_node.html
  - flyXT
    - Unternehmensspezifische Anpassung des V-Modells XT bei EADS DE
    - Nicht öffentlich

Unterstützungsprozesse

Vorlesung Automotive SWE (Buch Schäuffele/Zurawka)	V-Modell: Tätigkeitsbereiche	CMMI Reifegradstufe 2: Key Process Areas
Kernprozess	Systemerstellung	
Unterstützungsprozesse		
Konfigurationsmanagement	Konfigurationsmanagement	Konfigurationsmanagement
Projektmanagement	Projektmanagement	Projektplanung Projektverfolgung
Lieferantenmanagement	Teil des Projektmanagements	Lieferantenmanagement
Anforderungsmanagement	Teil des Projektmanagements	Anforderungsmanagement
Qualitätssicherung	Qualitätssicherung	Qualitätssicherung

Unterstützungsprozesse

Vorlesung Automotive SWE (Buch Schäuffele/Zurawka)	**BMW Group Standard embedded Software (GSeSW)**
Kernprozess	Development processes
Teil des Kernprozesses	Software Development
Teil des Kernprozesses	Software Modification
Teil des Kernprozesses	Design Verification process
Teil des Kernprozesses	Testprocess

Unterstützungsprozesse

Vorlesung Automotive SWE (Buch Schäuffele/Zurawka)	**BMW Group Standard embedded Software (GSeSW)**
Unterstützungsprozesse	Organizational processes
Projektmanagement	Project Management
Teil des Projektmanagements	Risk Management
Teil des Projektmanagements	Role Assignment
Teilweise im Kernprozess (Erstellung Produktdokumentation)	Documentation
Teilweise im Kernprozess (Erstellung Produktdokumentation)	Training and Qualification
	Planning Training and Qualification

Unterstützungsprozesse

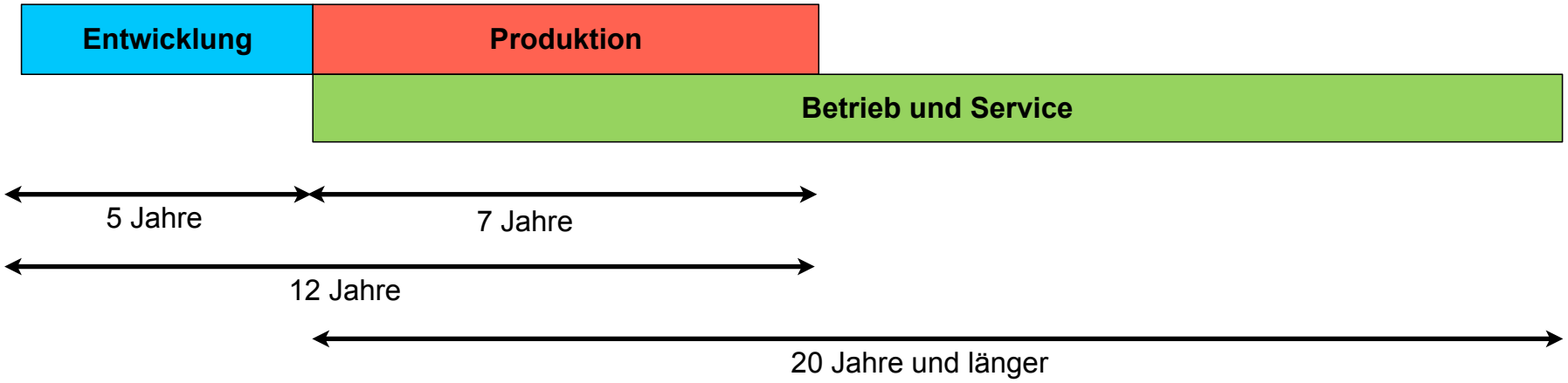
Vorlesung Automotive SWE (Buch Schäuffele/Zurawka)	**BMW Group Standard embedded Software (GSeSW)**
Unterstützungsprozesse	Supporting processes
Konfigurationsmanagement	Configuration Management
Qualitätssicherung	Quality Assurance
Teil des Projektmanagements	Problem-solving and Change Management

6. SW-Entwicklung / 3. Unterstützungsprozesse
Unterstützungsprozesse für die Embedded Software Entwicklung

1. Vorgehensmodelle und Standards
2. **Konfigurationsmanagement**
    1. **Produkt und Lebenszyklus**
    2. Varianten und Skalierbarkeit
    3. Versionen und Konfigurationen
3. Projektmanagement
4. Lieferantenmanagement
5. Anforderungsmanagement
6. Qualitätssicherung

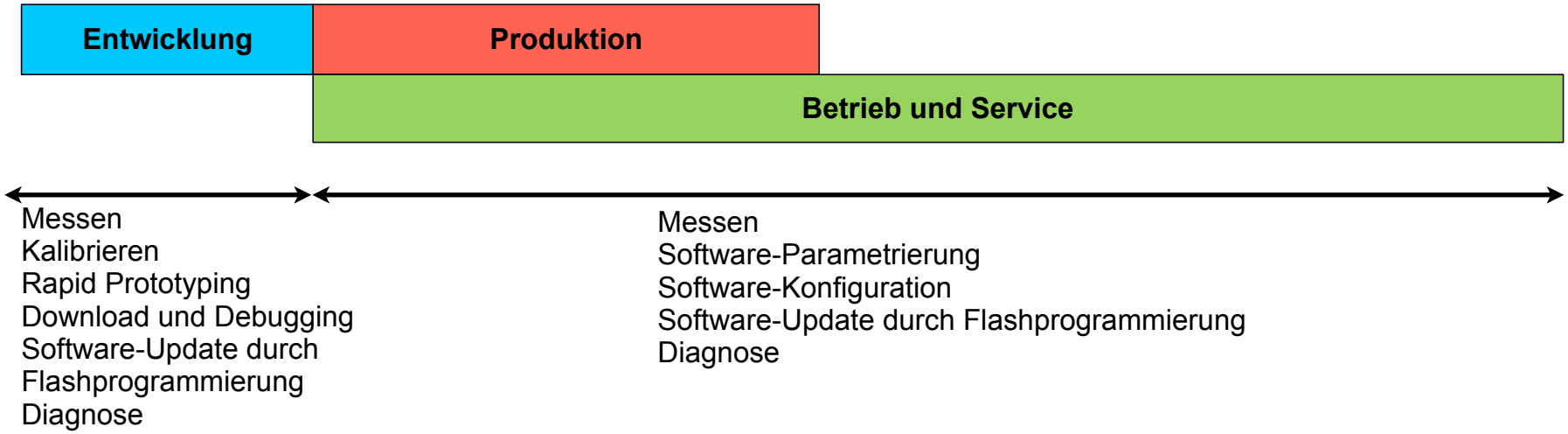
Typischer Produktlebenszyklus eines Fahrzeugs (Nach Schäuffele, Zurawka)

- Phasen
  - Entwicklung
  - Produktion
  - Betrieb und Service



Unterschiedliche Anforderungen an Schnittstellen des Steuergerätes (Nach Schäuffele, Zurawka)

- Unterschiedliche Lebenszyklen der Komponenten
  - Fahrzeug 10 - 20 Jahre
  - Steuergeräte 2 Jahre
  - Unterhaltungselektronik 6 Monate
- Unterschiedliche Anforderungen an Schnittstellen des Steuergerätes in Entwicklung, Produktion, Betrieb und Service

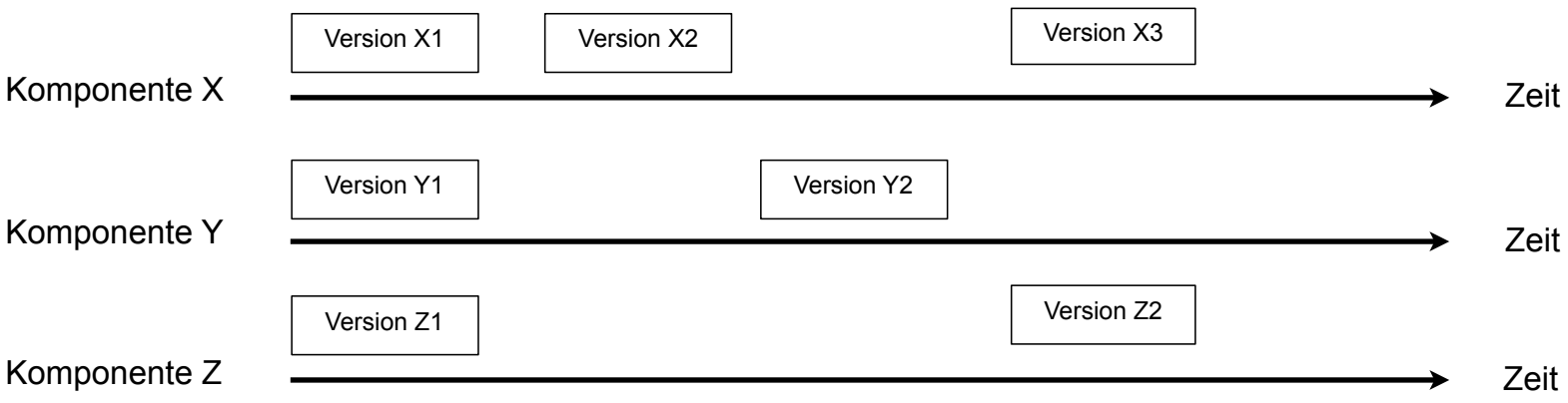


6. SW-Entwicklung / 3. Unterstützungsprozesse
Unterstützungsprozesse für die Embedded Software Entwicklung

1. Vorgehensmodelle und Standards
2. **Konfigurationsmanagement**
   1. Produkt und Lebenszyklus
   2. **Varianten und Skalierbarkeit**
   3. Versionen und Konfigurationen
3. Projektmanagement
4. Lieferantenmanagement
5. Anforderungsmanagement
6. Qualitätssicherung

Versionen
(Nach Schäuffele, Zurawka)

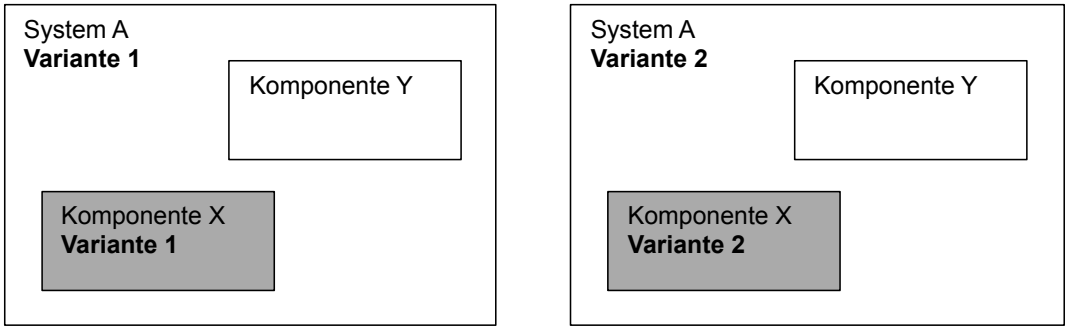
- Bestehende elektronische Systeme werden weiterentwickelt
- Zusätzliche elektronische Systeme werden eingeführt
- Komponentenebene
  - Zu bestimmten Zeitpunkten verschiedene Versionen
- Systemebene
  - Zusätzlich Verwaltung der Beziehungen (Referenzen) zu den enthaltenen Komponenten
- Version:
  - Verwaltung der Komponente an sich
  - Änderung der Komponente (Zeit)



Verschiedene Versionen auf Komponentenebene

Varianten und Skalierbarkeit
Variantenbildung für Komponenten (Nach Schäuffele, Zurawka)

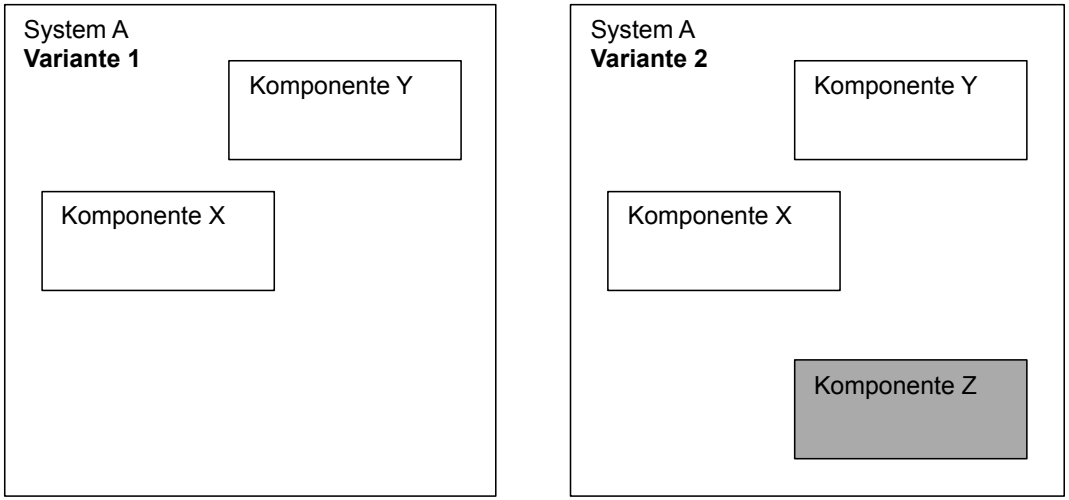
- Zunehmende Anzahl von Fahrzeugvarianten
- Gestiegene Kundenerwartungen
  - **Individualisierung**
  - Erweiterbarkeit
- Bespiel
  - Variante 1: Automatikgetriebe
  - Variante 2: Schaltgetriebe



Bildung von Systemvarianten durch Komponentenvarianten

Varianten und Skalierbarkeit
Skalierbare Systemarchitekturen (Nach Schäuffele, Zurawka)

- Zunehmende Anzahl von Fahrzeugvarianten
- Gestiegene Kundenerwartungen
  - Individualisierung
  - **Erweiterbarkeit**
- Beispiel
  - Komponente Z: Schiebedach



Bildung von Systemvarianten durch Skalierung

Baureihen, Fahrzeugvarianten, Karosserievarianten

- 1 Baureihe: 3er MBW
- 4 Fahrzeugvarianten: Limousine, Coupé, Kombi, Cabrio
- 7 Karosserievarianten: Mit und ohne Schiebedach

Karosserie: Varianten und Gleichteile

	Front	Fahrgastzelle mit Schiebedach	Fahrgastzelle ohne Schiebedach	Heck
Limousine mit Schiebedach	Variante LmS	Variante LmS	entfällt	Variante LmS
Limousine ohne Schiebedach	Variante LmS	entfällt	Variante LoS	Variante LmS
Coupé mit Schiebedach	Variante LmS	Variante CmS	entfällt	Variante LmS
Coupé ohne Schiebedach	Variante LmS	entfällt	Variante CoS	Variante LmS
Kombi mit Schiebedach	Variante LmS	Variante LmS	entfällt	Variante KmS
Kombi ohne Schiebedach	Variante LmS	entfällt	Variante LoS	Variante KmS
Cabrio	Variante LmS	entfällt	Variante Cabrio	Variante Cabrio
7 Karosserie-varianten	1 Frontvariante	2 Fahrgast-zellenvarianten	3 Fahrgast-zellenvarianten	3 Heckvarianten

Karosserie: Varianten und Gleichteile

- Unterschiedliche Karosserievarianten benötigen unterschiedliche Karosseriefunktionen
- Realisierung der Karosseriefunktionen durch
  - Bedienelemente / Sollwertgeber
  - Aktoren
  - Sensoren
  - Steuergeräte
  - Software
- Folge: Varianten und Gleichteile auf den Ebenen
  - Bedienelemente / Sollwertgeber
  - Aktoren
  - Sensoren
  - Steuergeräte
  - Software

Karosseriefunktionen: Varianten und Gleichteile

Steuerung von 12 Funktionen	Limousine mit Schiebedach	Limousine ohne Schiebedach	Coupé mit Schiebedach	Coupé ohne Schiebedach	Kombi mit Schiebedach	Kombi ohne Schiebedach	Cabrio	7
Türschliessen vorne F	Variante L	Variante L	Variante L	Variante L	Variante L	Variante L	Variante L	1
Türschliessen vorne BF	Variante L	Variante L	Variante L	Variante L	Variante L	Variante L	Variante L	1
Türschliessen hinten F	Variante L	Variante L	entfällt	entfällt	Variante L	Variante L	entfällt	1
Türschliessen hinten BF	Variante L	Variante L	entfällt	entfällt	Variante L	Variante L	entfällt	1
Fensterheber vorne F	Variante L	Variante L	Variante Coupé	Variante Coupé	Variante L	Variante L	Variante Cabrio	3
Fensterheber vorne BF	Variante L	Variante L	Variante Coupé	Variante Coupé	Variante L	Variante L	Variante Cabrio	3
Fensterheber hinten F	Variante L	Variante L	Variante Coupé	Variante Coupé	Variante L	Variante L	Variante Cabrio	3
Fensterheber hinten BF	Variante L	Variante L	Variante Coupé	Variante Coupé	Variante L	Variante L	Variante Cabrio	3
Schiebedach	Variante L	entfällt	Variante L	entfällt	Variante L	entfällt	entfällt	1
Beleuchtung Innenraum	Variante L	Variante L	Variante Coupé	Variante Coupé	Variante K	Variante K	Variante Cabrio	4
Heckklappe	Variante L	Variante L	Variante L	Variante L	Variante K	Variante K	Variante Cabrio	3
Verdeck	entfällt	entfällt	entfällt	entfällt	entfällt	entfällt	Variante Cabrio	1

Motorvarianten C-Klasse Limousine

Modell	Zylinder-anordnung/ -anzahl	Hubraum (cm3)	Nenn-leistung (kW bei 1/min)[1]	Höchst-geschwind -igkeit (km/h)	Kraftstoff-verbrauch kombiniert (l/100 km)[2]	CO2-Emissionen kombiniert (g/km)[2]
C 180 CDI BlueEFFICIENCY	R4	2.143	88/2.800–4.600 (88/3.000–4.600)	208 (206)	5,3–4,8 (5,3–4,9)	139–125 (140–129)
C 200 CDI BlueEFFICIENCY	R4	2.143	100/2.800–4.600 (100/2.800–4.600)	218 (215)	5,3–4,8 (5,3–4,9)	139–125 (140–129)
C 220 CDI BlueEFFICIENCY	R4	2.143	125/3.000–4.200 (125/3.000–4.200)	232 (231)	5,1–4,4 (5,2–4,8)	133–117 (136–125)
C 220 CDI BlueEFFICIENCY Edition	R4	2.143	125/3.000–4.200 (125/3.000–4.200)	232 (231)	4,1 (4,4)	109 (116)
C 250 CDI BlueEFFICIENCY	R4	2.143	150/4.200 (150/4.200)	240 (240)	5,3–4,8 (5,2–4,8)	140–125 (136–125)
C 250 CDI 4MATIC BlueEFFICIENCY	R4	2.143	– (150/4.200)	– (240)	– (5,7–5,4)	– (152–144)
C 300 CDI 4MATIC BlueEFFICIENCY	V6	2.987	– (170/3.800)	– (250)[3]	– (7,2–7,0)	– (189–185)
C 350 CDI BlueEFFICIENCY	V6	2.987	– (195/3.800)	– (250)[3]	– (6,0–5,9)	– (157–154)

http://www.mercedes-benz.de/content/germany/mpc/mpc_germany_website/de/home_mpc/passengercars/home/new_cars/models/c-class/_w204/facts_/drivetrain/dieselengines.html

Motorvarianten C-Klasse Limousine

Modell	Zylinder-anordnung/ -anzahl	Hubraum (cm3)	Nenn-leistung (kW bei 1/min)[1]	Höchst-geschwind -igkeit (km/h)	Kraftstoff-verbrauch kombiniert (l/100 km)[2]	CO2-Emissionen kombiniert (g/km)[2]
C 180 CDI BlueEFFICIENCY	R4	2.143	88/2.800–4.600 (88/3.000–4.600)	208 (206)	5,3–4,8 (5,3–4,9)	139–125 (140–129)
C 200 CDI BlueEFFICIENCY	R4	2.143	100/2.800–4.600 (100/2.800–4.600)	218 (215)	5,3–4,8 (5,3–4,9)	139–125 (140–129)
C 220 CDI BlueEFFICIENCY	R4	2.143	125/3.000–4.200 (125/3.000–4.200)	232 (231)	5,1–4,4 (5,2–4,8)	133–117 (136–125)
C 220 CDI BlueEFFICIENCY Edition	R4	2.143	125/3.000–4.200 (125/3.000–4.200)	232 (231)	4,1 (4,4)	109 (116)
C 250 CDI BlueEFFICIENCY	R4	2.143	150/4.200 (150/4.200)	240 (240)	5,3–4,8 (5,2–4,8)	140–125 (136–125)
C 250 CDI 4MATIC BlueEFFICIENCY	R4	2.143	– (150/4.200)	– (240)	– (5,7–5,4)	– (152–144)
C 300 CDI 4MATIC BlueEFFICIENCY	V6	2.987	– (170/3.800)	– (250)[3]	– (7,2–7,0)	– (189–185)
C 350 CDI BlueEFFICIENCY	V6	2.987	– (195/3.800)	– (250)[3]	– (6,0–5,9)	– (157–154)

http://www.mercedes-benz.de/content/germany/mpc/mpc_germany_website/de/home_mpc/passengercars/home/new_cars/models/c-class/_w204/facts_/drivetrain/dieselengines.html

Softwarevarianten und Fahrzeugvarianten

- Karosserieformen und Karosseriefunktionen Softwarevarianten folgen Fahrzeugvarianten



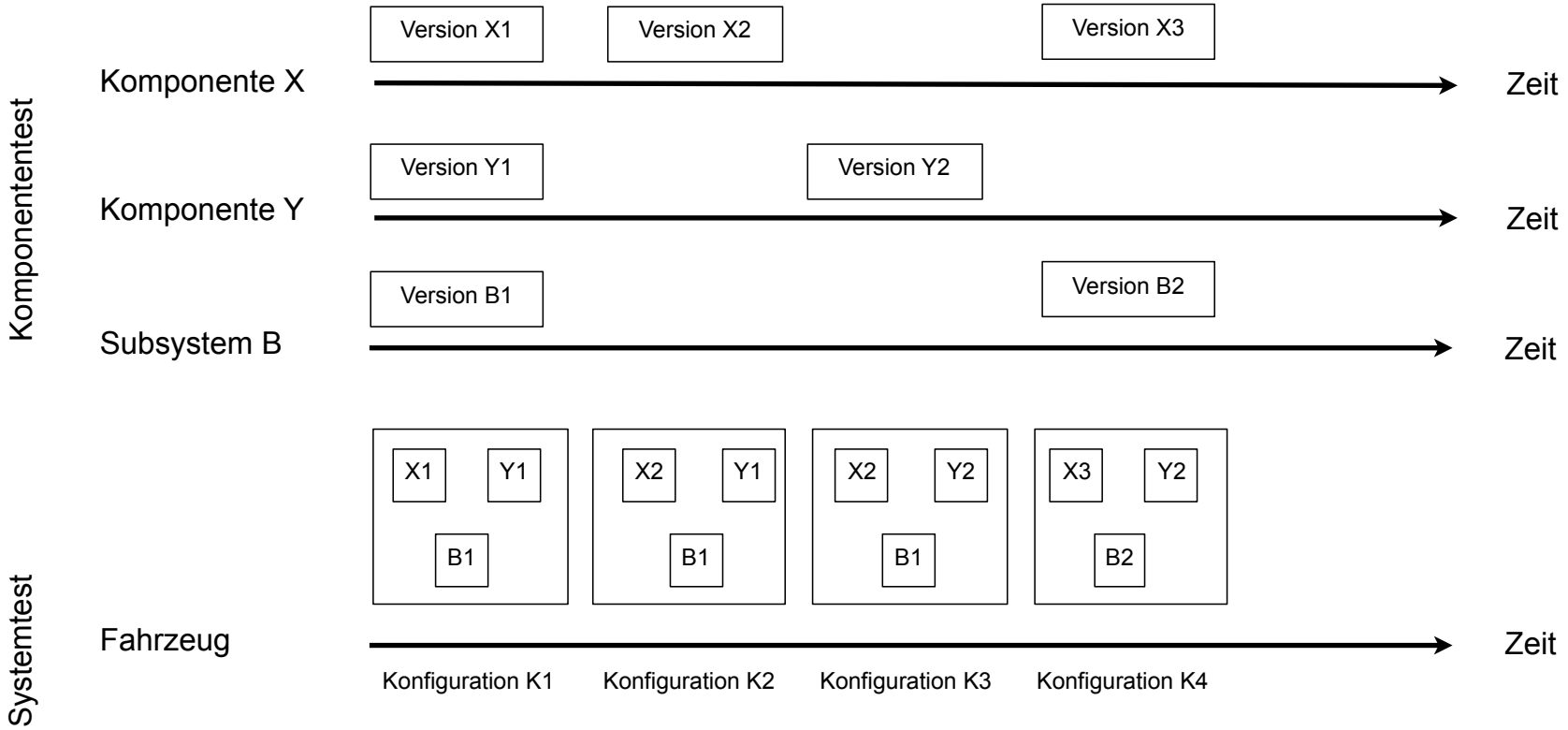

- Motorvarianten C-Klasse Softwarevarianten ermöglichen Fahrzeugvarianten

6. SW-Entwicklung / 3. Unterstützungsprozesse
Unterstützungsprozesse für die Embedded Software Entwicklung

1. Vorgehensmodelle und Standards
2. **Konfigurationsmanagement**
    1. Produkt und Lebenszyklus
    2. Varianten und Skalierbarkeit
    3. **Versionen und Konfigurationen**
3. Projektmanagement
4. Lieferantenmanagement
5. Anforderungsmanagement
6. Qualitätssicherung

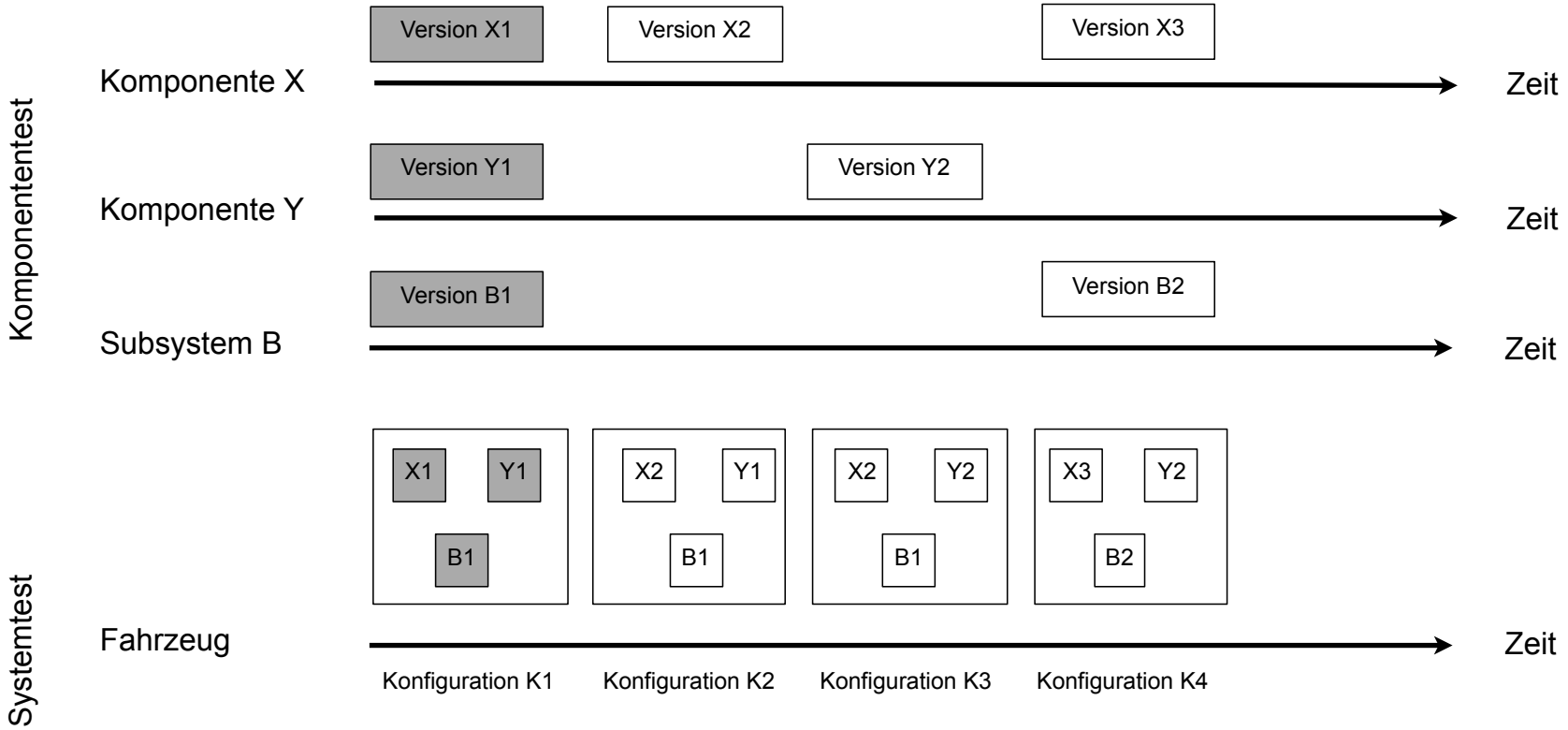
Integration der System-Komponenten (Nach Schäuffele, Zurawka)

- Abhängigkeiten zwischen Komponenten- und Systemtest



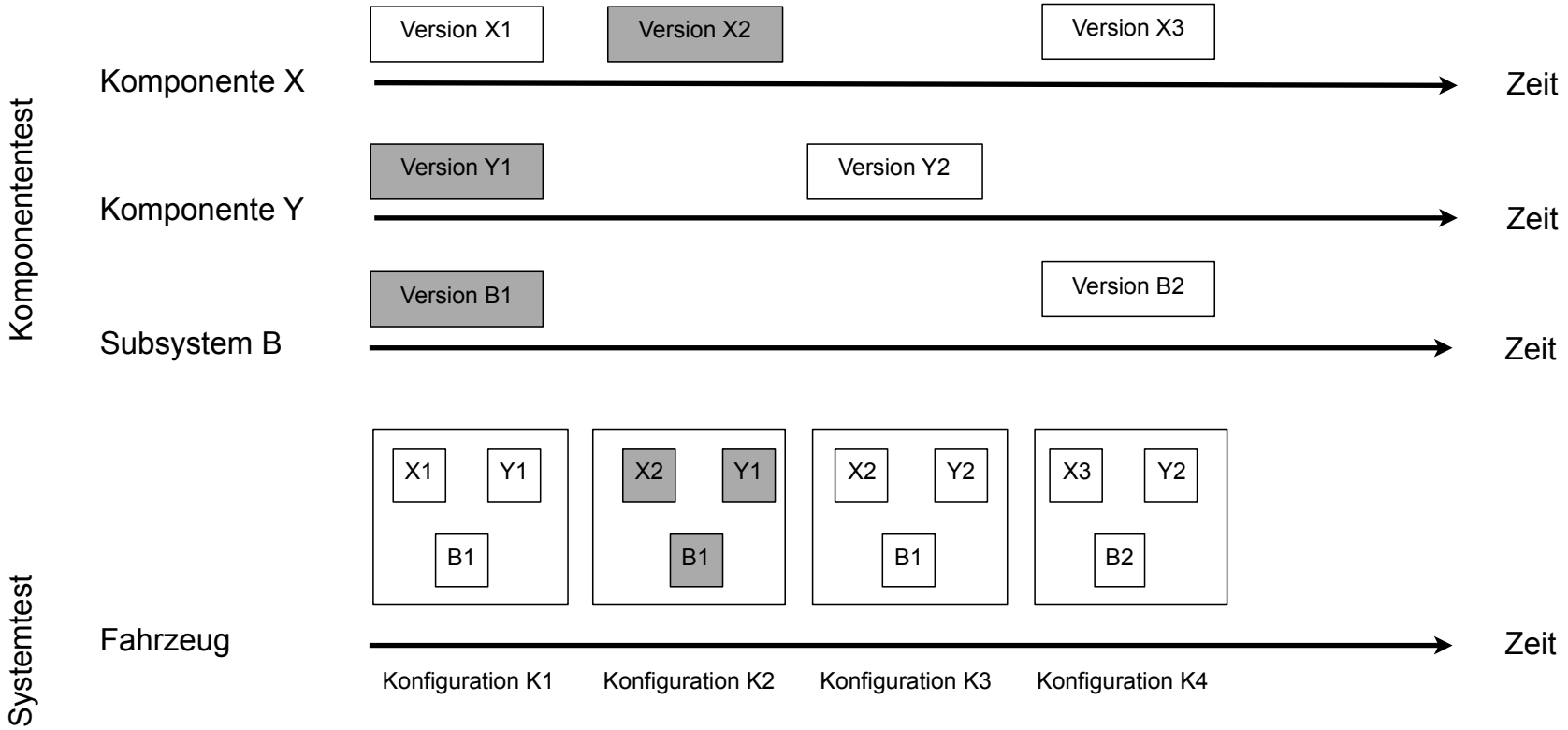
Integration der System-Komponenten (Nach Schäuffele, Zurawka)

- Abhängigkeiten zwischen Komponenten- und Systemtest



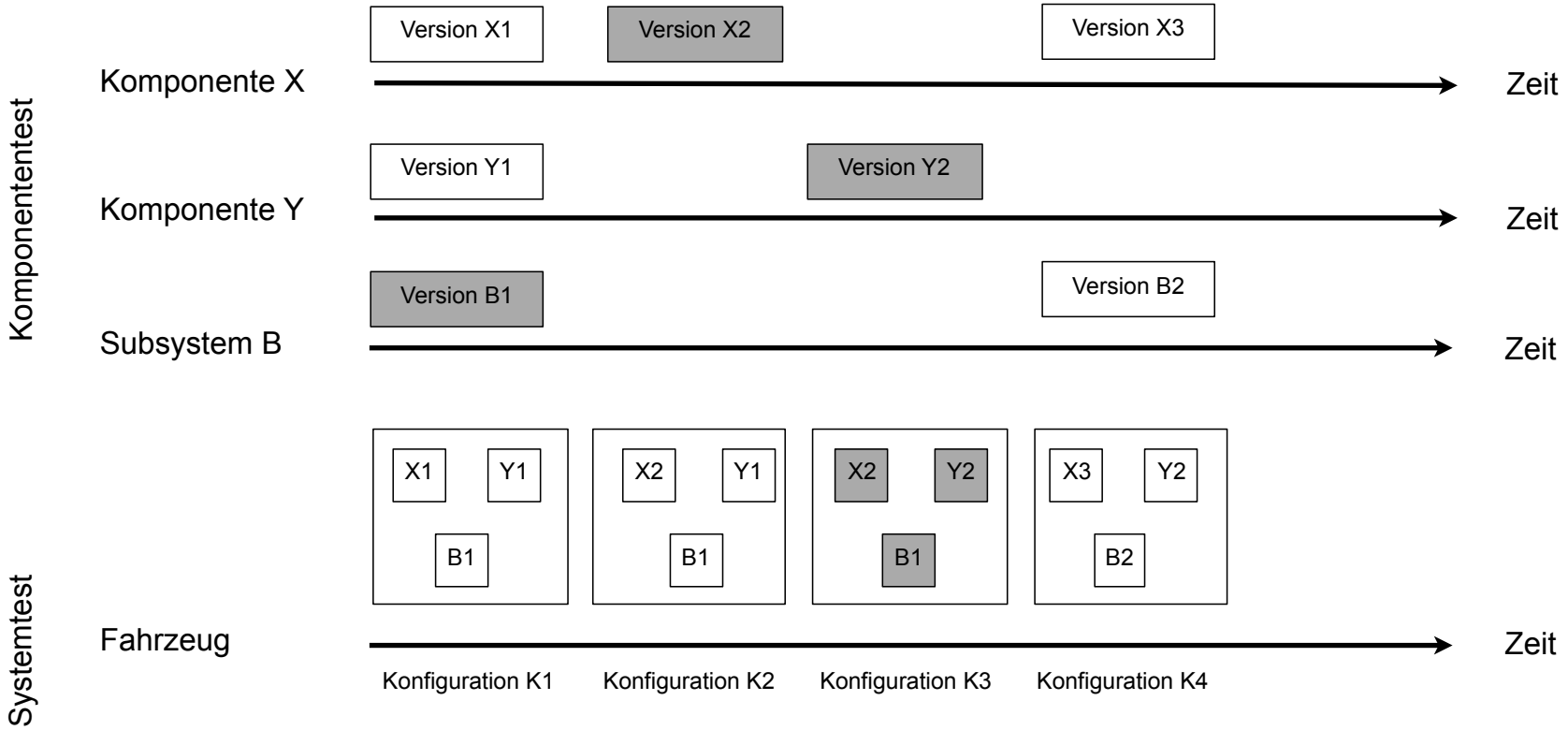
Integration der System-Komponenten (Nach Schäuffele, Zurawka)

- Abhängigkeiten zwischen Komponenten- und Systemtest



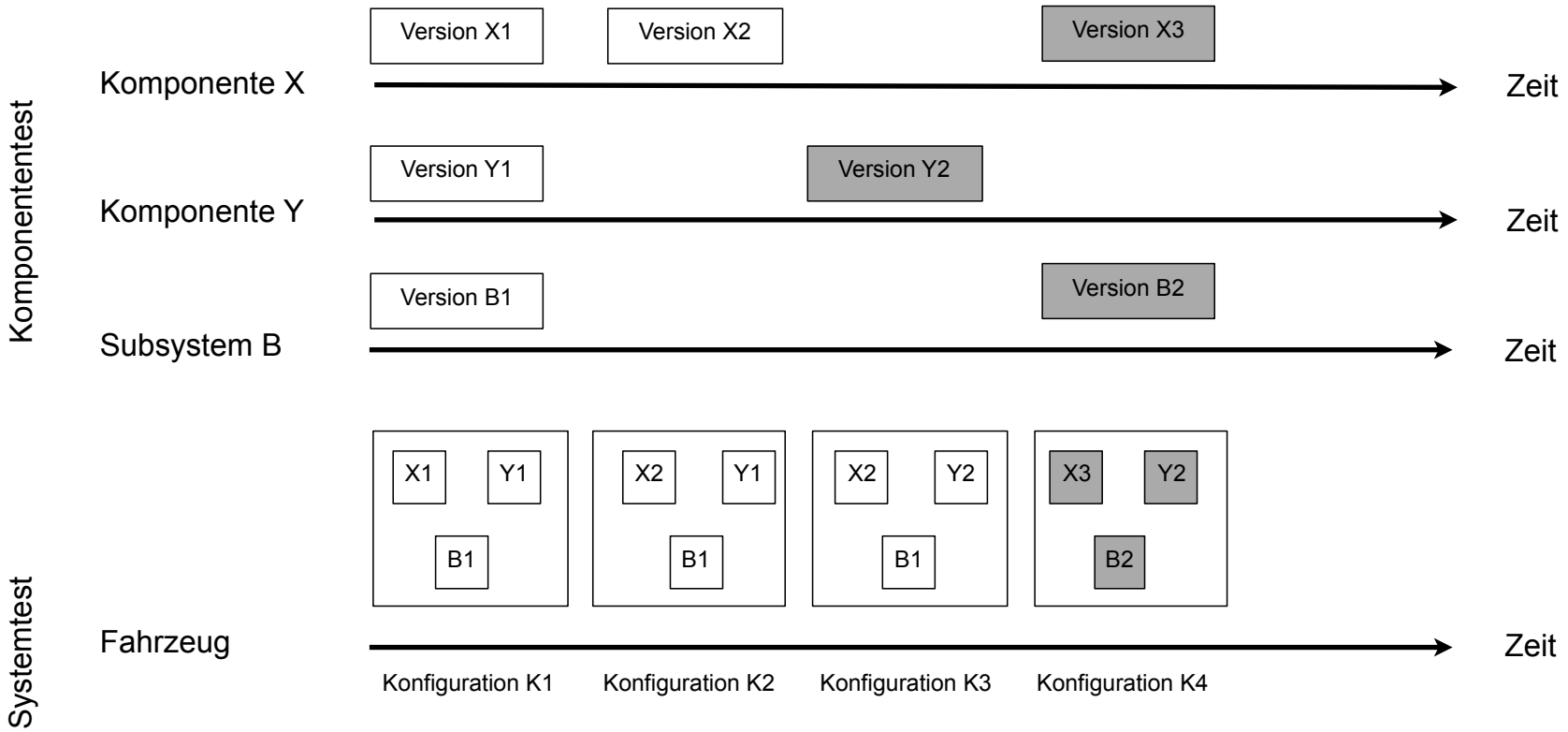
Integration der System-Komponenten (Nach Schäuffele, Zurawka)

- Abhängigkeiten zwischen Komponenten- und Systemtest



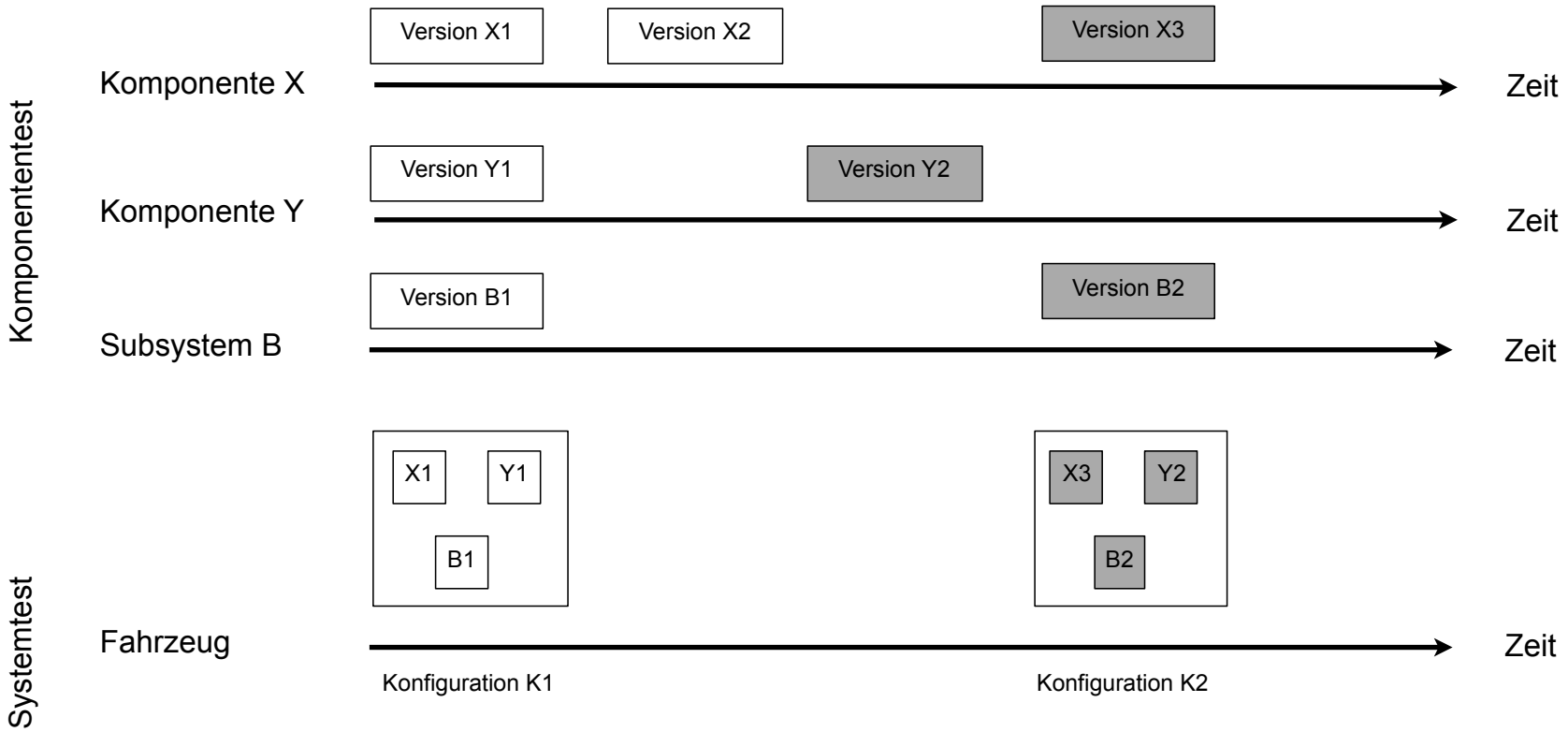
Integration der System-Komponenten (Nach Schäuffele, Zurawka)

- Abhängigkeiten zwischen Komponenten- und Systemtest



Integration der System-Komponenten (Nach Schäuffele, Zurawka)

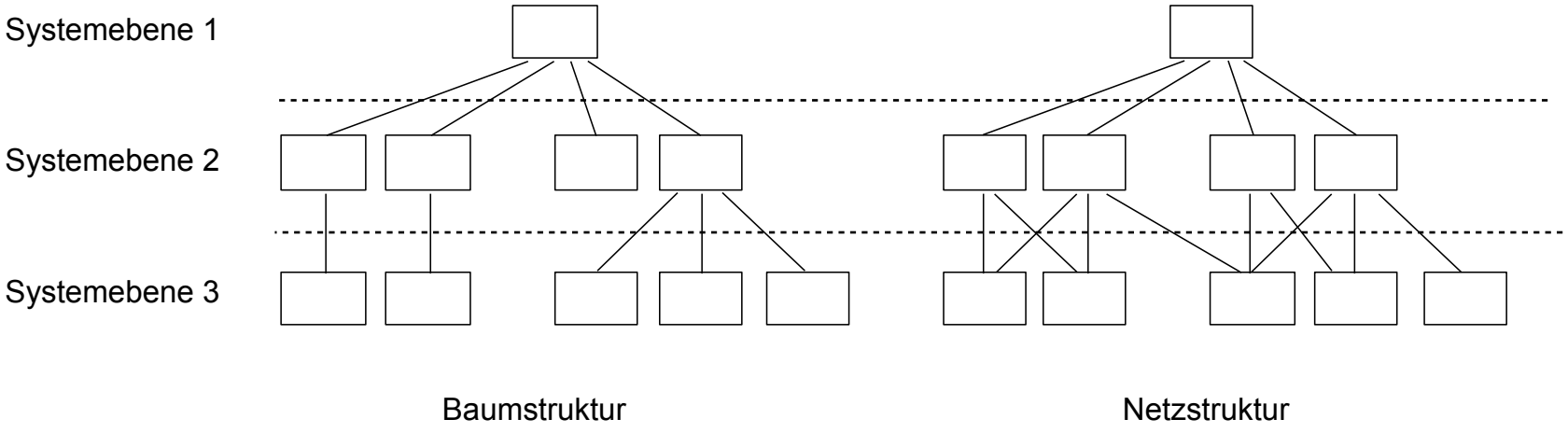
- Abhängigkeiten zwischen Komponenten- und Systemtest



- In der Praxis meist Bündelung: Weniger Konfigurationen

Versionen und Konfigurationen (Nach Schäuffele, Zurawka)

- Systemvarianten können auf allen Systemebenen auftreten
- Hierarchiebeziehungen
  - Baumstrukturen
    - Jede Komponente gehört zu genau einem System
  - Netzstrukturen
    - Eine Komponente kann zu mehreren Systemen gehören
- Das Versions- und Konfigurationsmanagement (Verwaltung von Versionen und Konfigurationen) geht von Netzstrukturen aus
  - Warum?

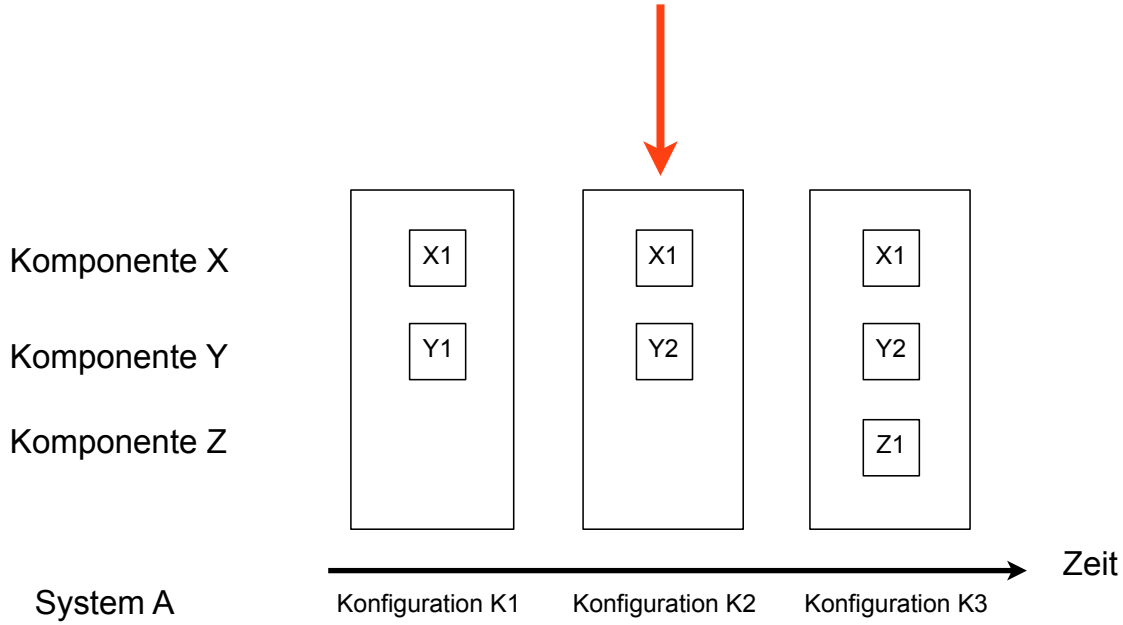


Konfiguration

- Eine Konfiguration ist eine versionierbare Komponente, die eine Menge anderer versionierbarer Komponenten referenziert.
- Verwaltung der Beziehungen zu den enthaltenen Komponentenversionen
- Referenzierte Versionen dürfen nicht mehr geändert werden :-)
  - ... müssen über die Lebenszeit der Konfiguration verfügbar sein

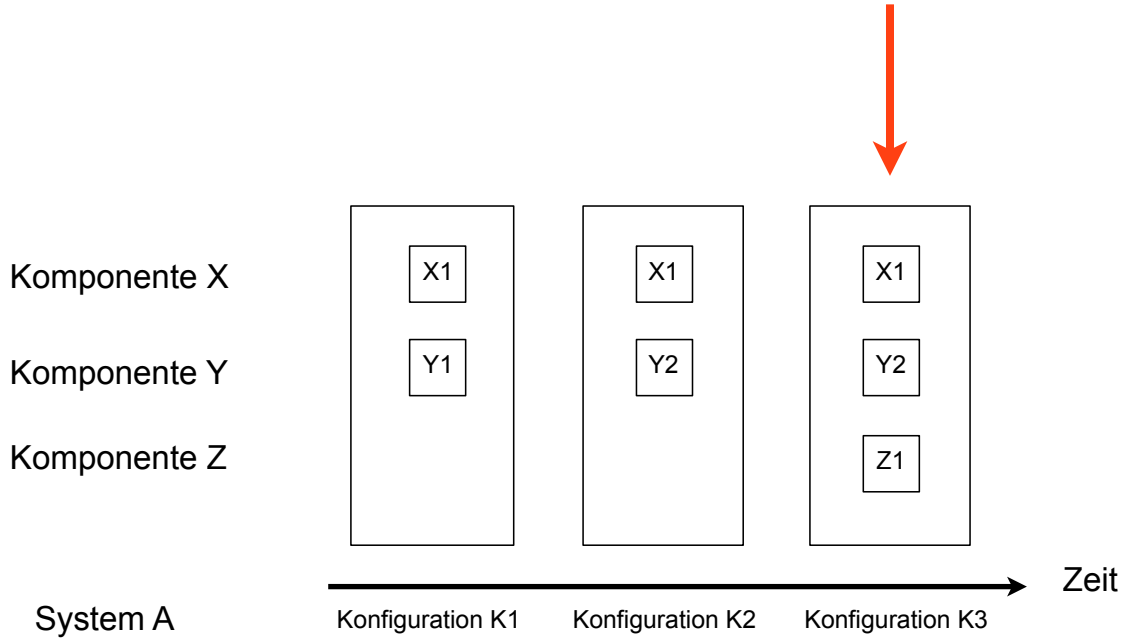
Konfiguration
(Nach Schäuffele, Zurawka)

- Versionen einer Konfiguration
  - **Versionen der Komponenten**
  - Änderung der Hierarchiebeziehungen



Konfiguration
(Nach Schäuffele, Zurawka)

- Versionen einer Konfiguration
  - Versionen der Komponenten
  - **Änderung der Hierarchiebeziehungen**



Konfigurationsmanagement

- Eigentlich:
  - Versions- und Konfigurationsmanagement
- Verwaltet die Beziehungen zwischen Systemen und Komponenten und deren Änderungen
  - Weiterentwicklungen von Komponenten
  - Änderung der Hierarchien von Systemen
- Wichtiger Bestandteil von
  - Entwicklungsprozessen
  - Produktionsprozessen
  - Serviceprozessen
- Das Konfigurationsmanagement verwaltet
  - Anforderungen
  - Spezifikationen
  - Implementierungen (Programmstände, Datenstände)
  - Beschreibungsdateien für Diagnose, Software-Update, Software-Parametrierung
  - Dokumentation

6. SW-Entwicklung / 3. Unterstützungsprozesse
Unterstützungsprozesse für die Embedded Software Entwicklung

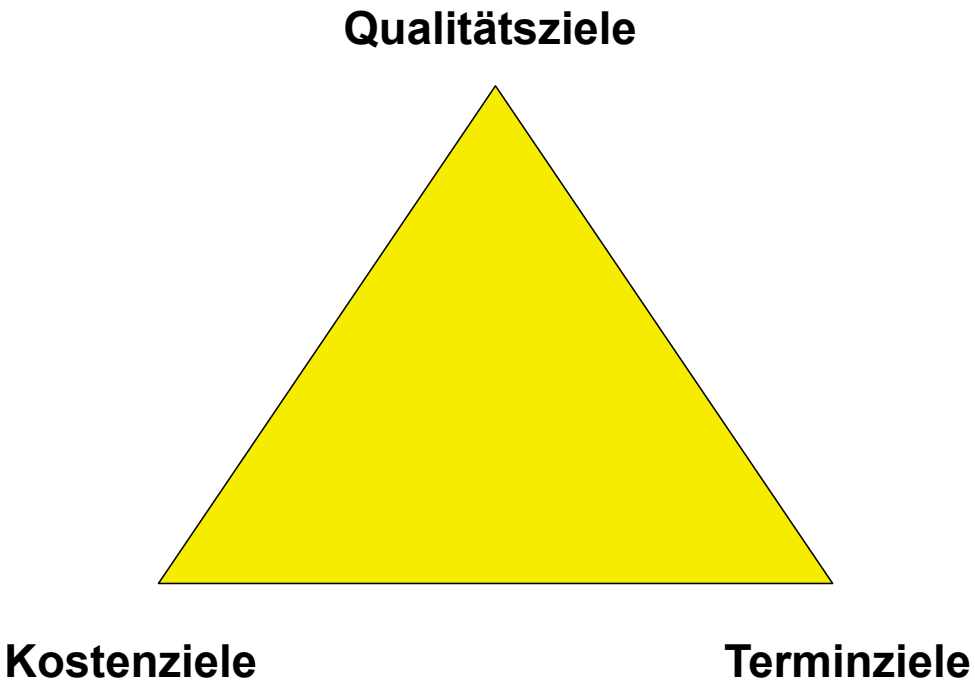
1. Vorgehensmodelle und Standards
2. Konfigurationsmanagement
3. **Projektmanagement**
    1. **Projektplanung**
    2. Projektverfolgung und Risikomanagement
4. Lieferantenmanagement
5. Anforderungsmanagement
6. Qualitätssicherung

Projekte

- Als Projekte werden Aufgabenstellungen bezeichnet, die durch folgende Merkmale gekennzeichnet sind
  - Aufgabenstellung mit Risiko und einer gewissen Einmaligkeit - keine Routineaufgaben
  - Eindeutige Aufgabenstellung
  - Verantwortung und Zielsetzung für ein zu lieferndes Gesamtergebnis
  - Zeitliche Befristung mit klarem Anfangs- und Endtermin
  - Begrenzter Ressoureneinsatz
  - Besondere auf das Projekt abgestimmte Organisation
  - Häufig: Teilaufgaben und Organisationseinheiten
    - Verschiedenartig
    - Untereinander verbunden
    - Wechselseitig voneinander abhängig

Projektziele

- Qualitätsziele
  - Welche Anforderungen sollen vom Gesamtergebnis erfüllt werden?
- Kostenziele
  - Wie viel darf die Erarbeitung des Gesamtergebnisses kosten?
- Terminziele
  - Wann soll das Gesamtergebnis vorliegen?



Projektmanagement
Projektplanung

- Planung der Umsetzung der Projektziele
  - Qualitätsplanung
  - Kostenplanung
  - Terminplanung
- Organisationsplanung
- Einsatzplanung
- Risikoanalyse

Unterstützungsprozesse für die Embedded Software Entwicklung

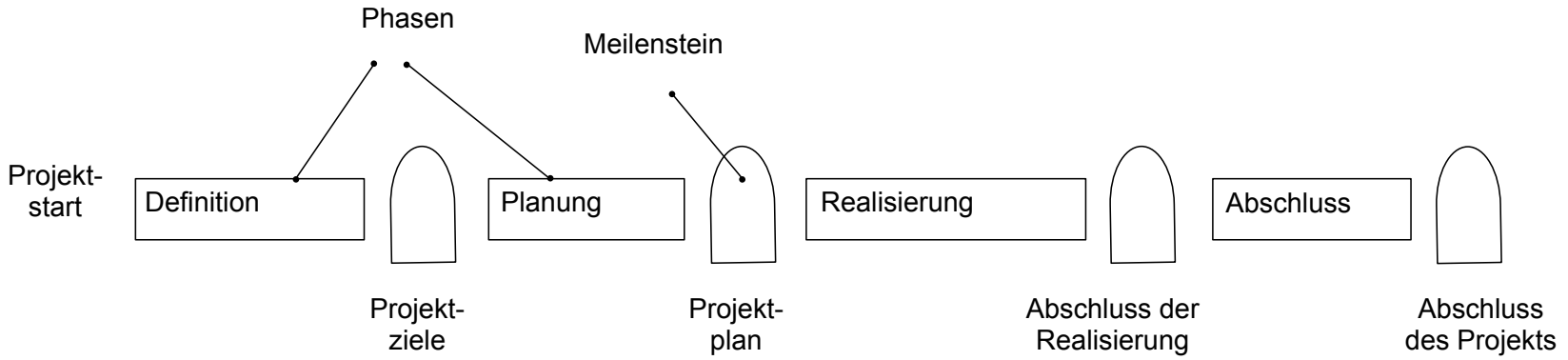
1. Vorgehensmodelle und Standards
2. Konfigurationsmanagement
3. Projektmanagement
    1. Projektplanung
    2. Projektverfolgung und Risikomanagement
4. Lieferantenmanagement
5. Anforderungsmanagement
6. Qualitätssicherung

Projektmanagement
Projektverfolgung / Projektsteuerung

- Verfolgung und Überwachung von
  - Qualität
  - Kosten
  - Terminen
- Risikomanagement
  - Beobachtung von auftretenden Risiken und Gegensteuerung

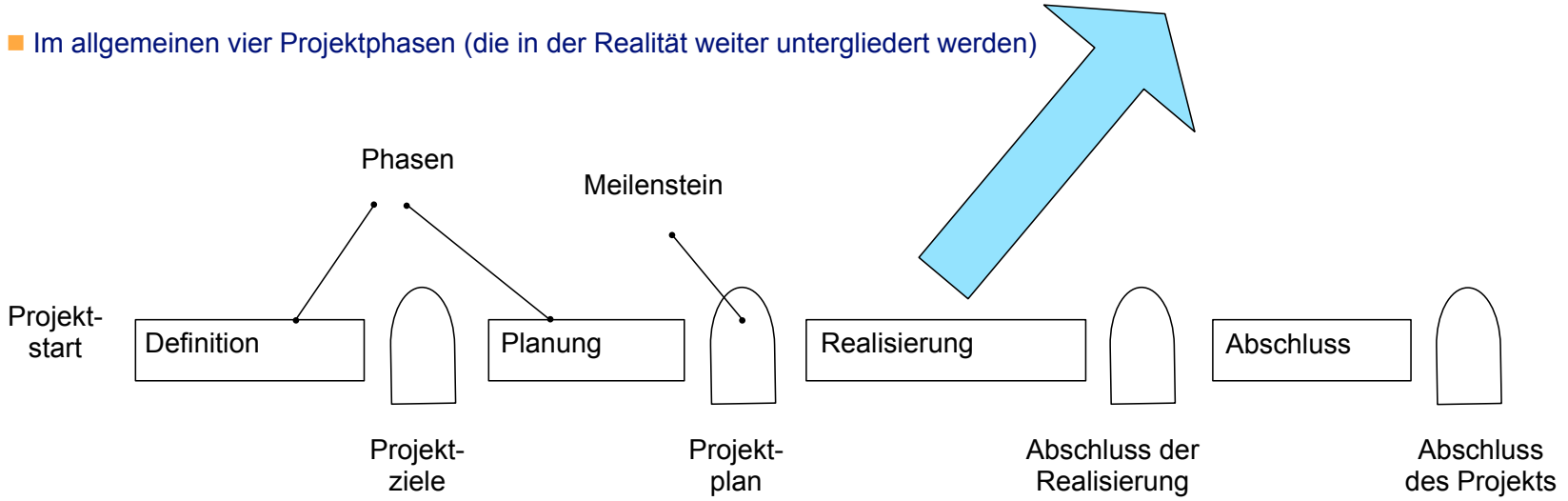
Projektplanung

- Definition der Teilaufgaben eines Projektes
- Meilenstein
  - Abschluss einer Teilaufgabe
  - Termin für
    - Teillieferungen
    - Tests
    - Teilzahlungen
- Projektphase
  - Zeitraum, in dem eine Teilaufgabe bearbeitet wird
  - Im allgemeinen vier Projektphasen (die in der Realität weiter untergliedert werden)



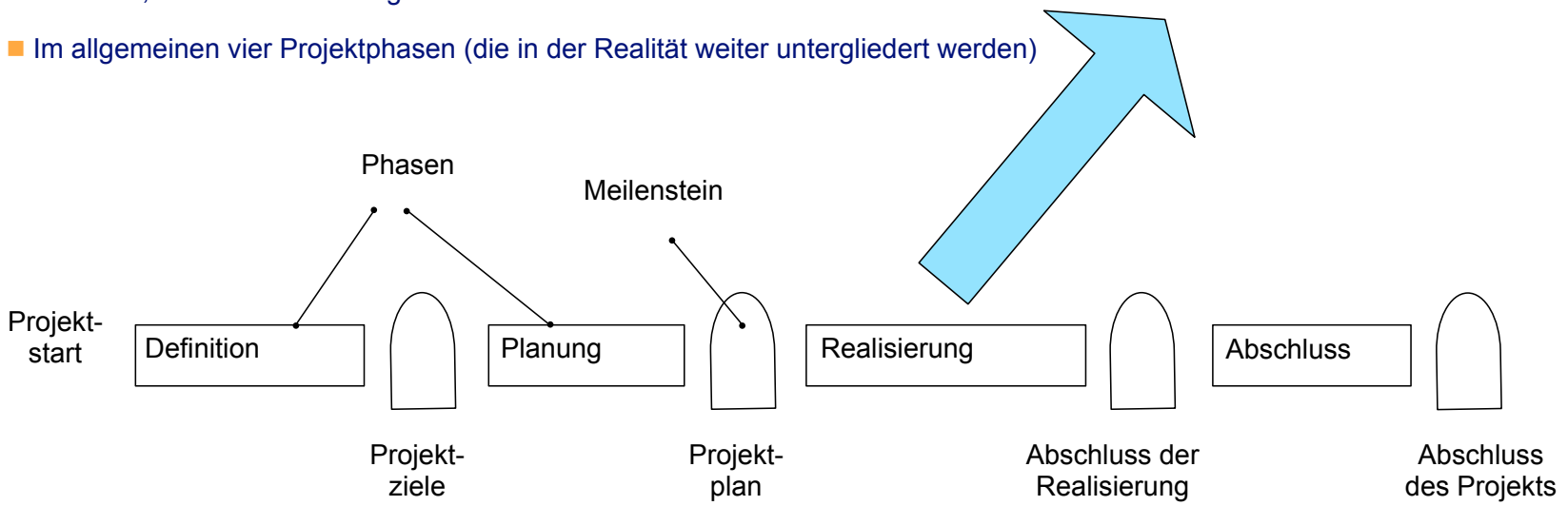
Projektplanung

- Definition der Teilaufgaben eines Projektes
- Meilenstein
  - Abschluss einer Teilaufgabe
  - Termin für
    - Teillieferungen
    - Tests
    - Teilzahlungen
- Projektphase
  - Zeitraum, in dem eine Teilaufgabe bearbeitet wird
  - Im allgemeinen vier Projektphasen (die in der Realität weiter untergliedert werden)



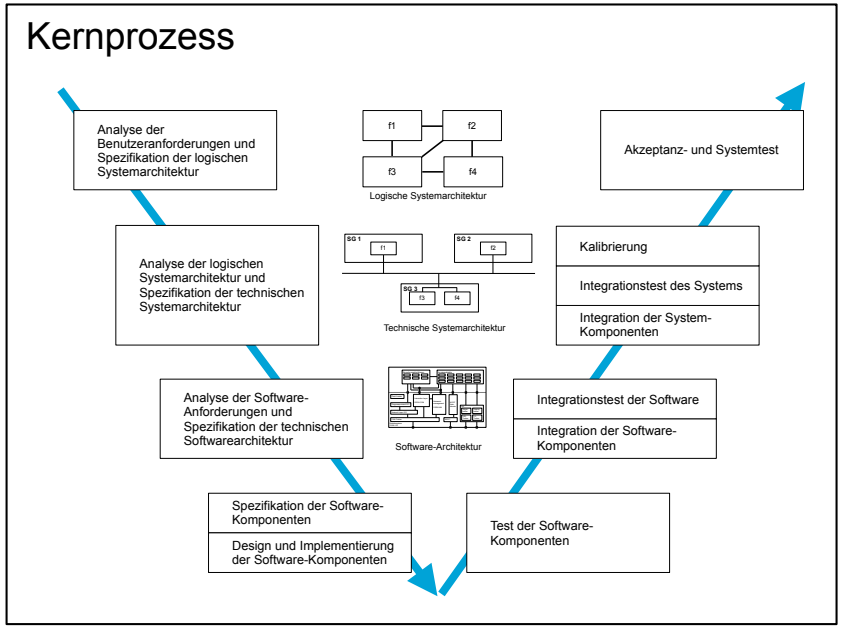
Projektplanung

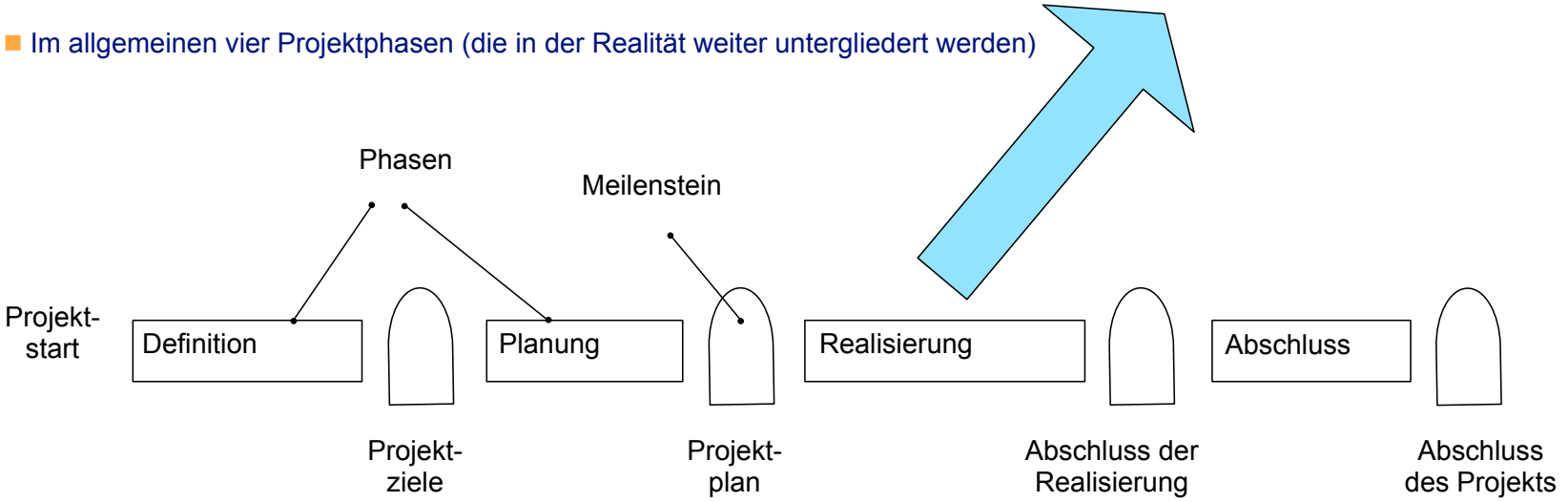
- Definition der Teilaufgaben eines Projektes
- Meilenstein
  - Abschluss einer Teilaufgabe
  - Termin für
    - Teillieferungen
    - Tests
    - Teilzahlungen
- Projektphase
  - Zeitraum, in dem eine Teilaufgabe bearbeitet wird
  - Im allgemeinen vier Projektphasen (die in der Realität weiter untergliedert werden)



Projektplanung

- Definition der Teilaufgaben eines Projektes
- Meilenstein
  - Abschluss einer Teilaufgabe
  - Termin für
    - Teillieferungen
    - Tests
    - Teilzahlungen
- Projektphase
  - Zeitraum, in dem eine Teilaufgabe bearbeitet wird
  - Im allgemeinen vier Projektphasen (die in der Realität weiter untergliedert werden)





Projektplanung: Qualität und Kosten

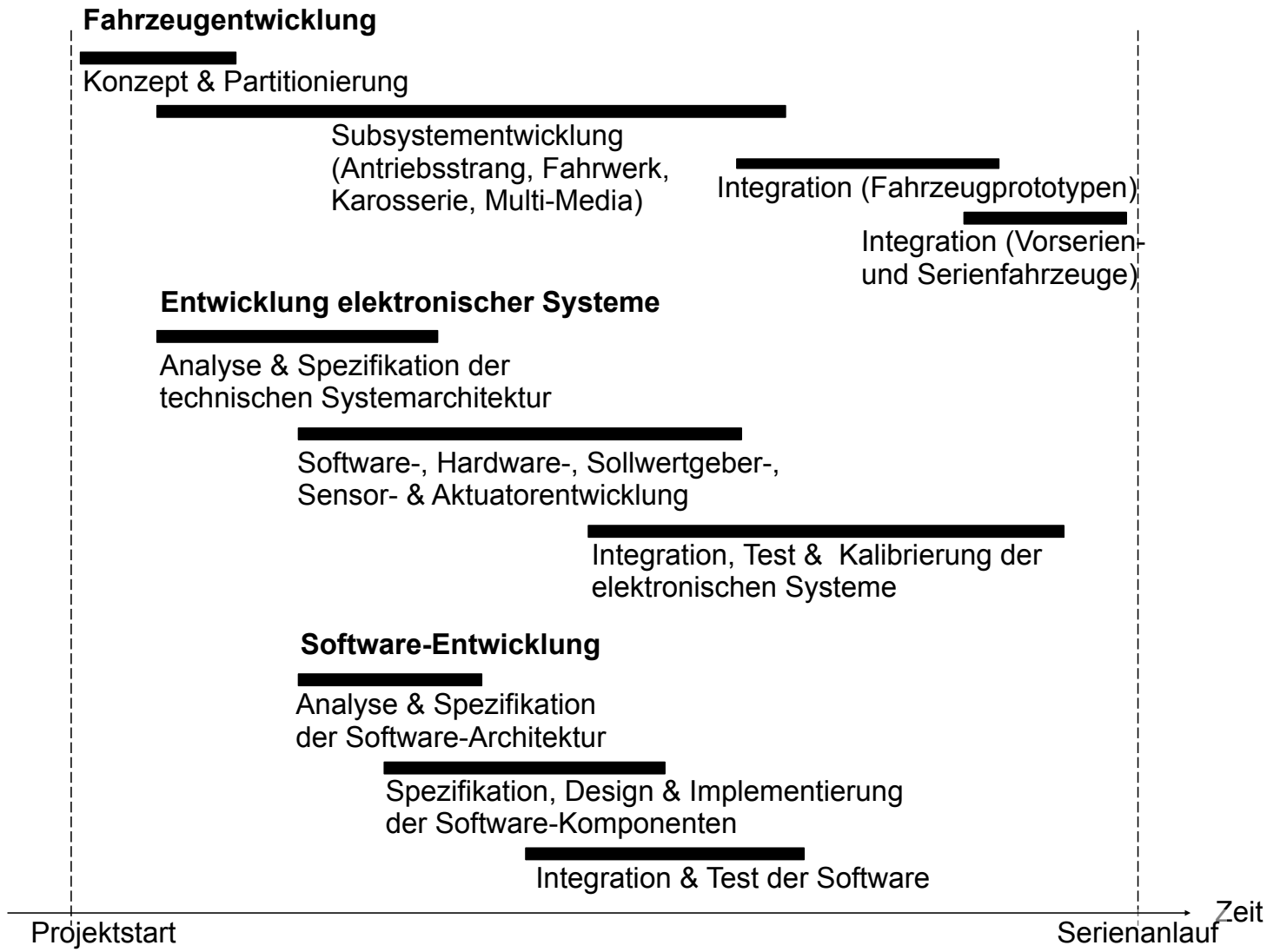
- Qualitätsplanung
  - Festlegung der Massnahmen zur Erreichung des Gesamtergebnisses
  - Für alle Projektphasen
    - Richtlinien zur Qualitätssicherung
    - Massnahmen zur Qualitätsprüfung
  - Phasenübergreifende Zusammenfassung in einem Qualitätsplan
- Kostenplanung
  - Personalkosten
    - Einsatzpläne für Mitarbeiter
  - Sachkosten
    - Materialkosten
    - Raumkosten
    - Reisekosten
  - Massnahmen
    - Z.B. Wiederverwendung von Ergebnissen aus anderen Projekten

Projektplanung: Termine

- Terminplanung
  - Festlegung des Zeitraums für die Durchführung der Projektphasen
    - Anfangstermin
    - Endtermin / Meilenstein
  - Herausforderungen
    - Einsatz von Mitarbeitern in verschiedenen Projekten
    - Gleichzeitige Durchführung mehrerer Projekte
    - Abhängigkeiten zwischen den Projektphasen
  - Abarbeitung
    - Sequentiell bei abhängigen Aufgaben
    - Parallel bei unabhängigen Aufgaben
      - Verkürzung der Entwicklungszeit
      - Simultaneous Engineering
        - Planung und und Synchronisation zeitlich paraller Entwicklungsschritte

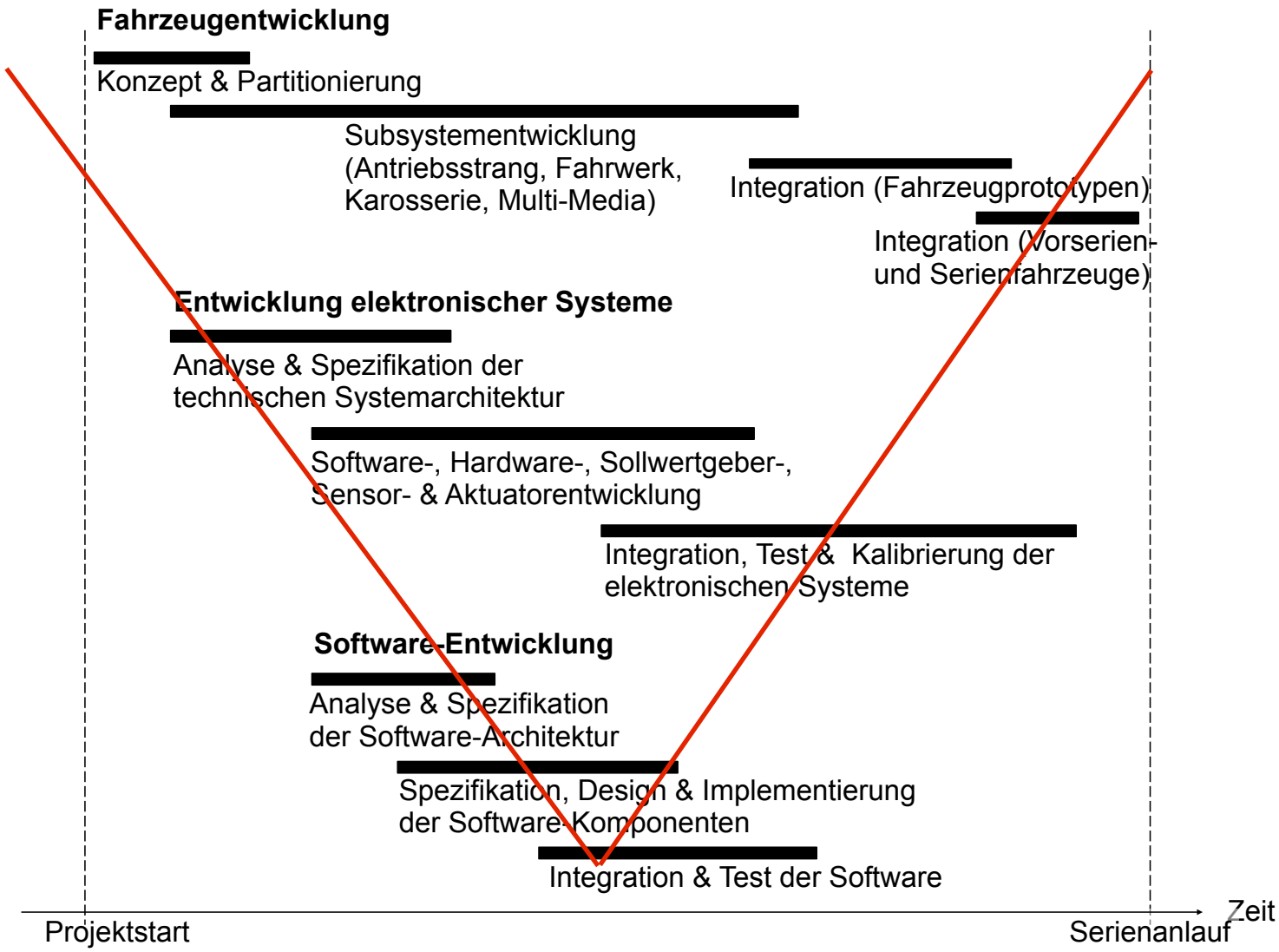
Beispiel: Terminplan für Fahrzeugentwicklung

Abstimmung von Fahrzeug-, Elektronik- und Software-Entwicklung



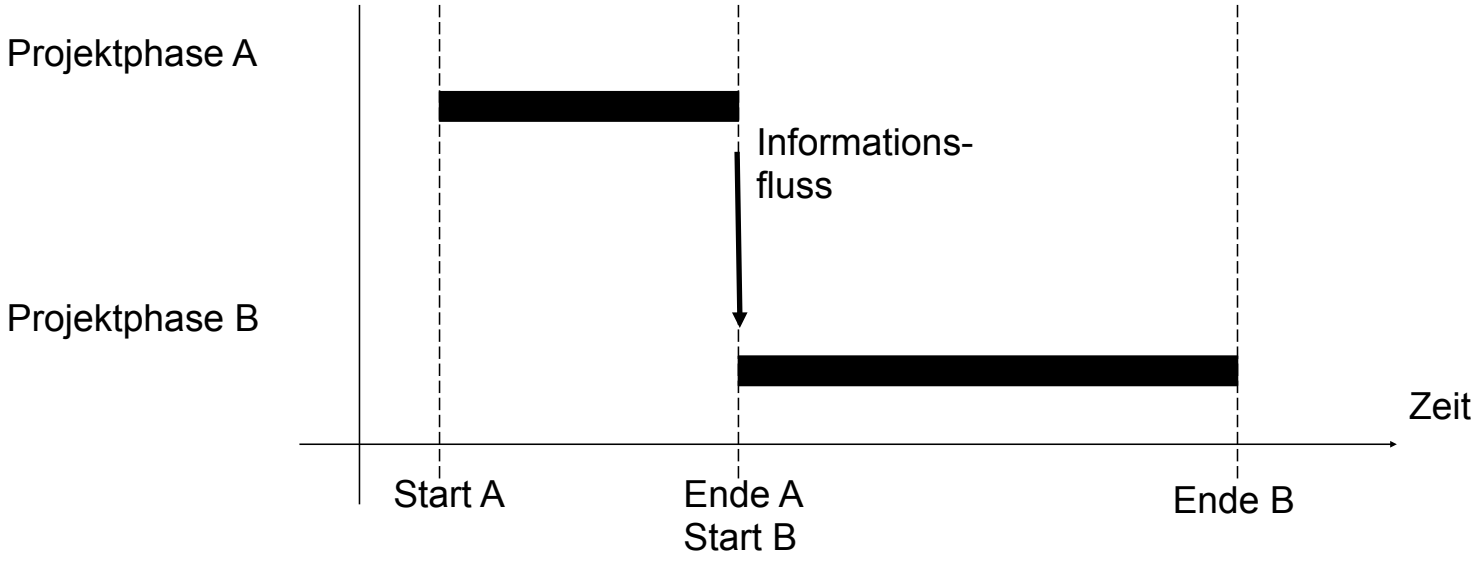
Beispiel: Terminplan für Fahrzeugentwicklung

Abstimmung von Fahrzeug-, Elektronik- und Software-Entwicklung



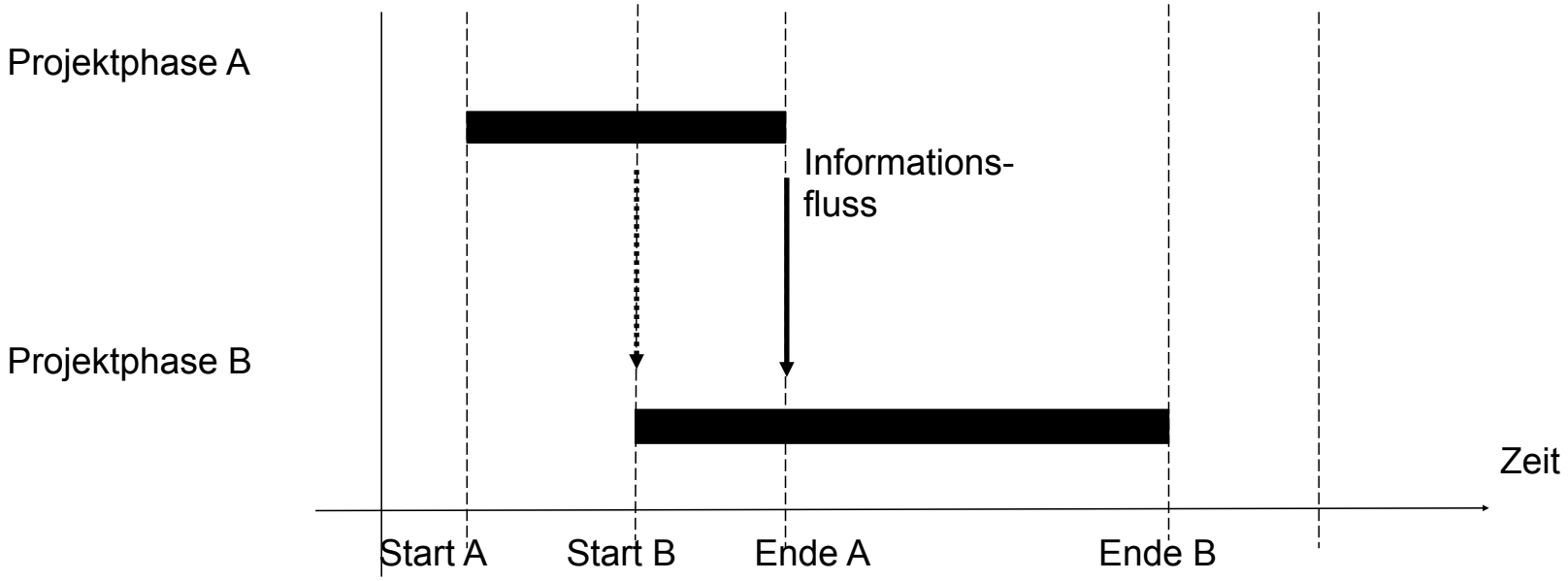
Sequentielle Planung von Projektphasen

- Start von Phase B nach Ende von Phase A
- Vorteile: Sequentieller Informationsfluss ohne Risiko
- Nachteile: Lange Bearbeitungszeit
- Beispiel: Integrationstest (Phase B) nach Integration (Phase A)



Parallele Planung ohne Einfrieren

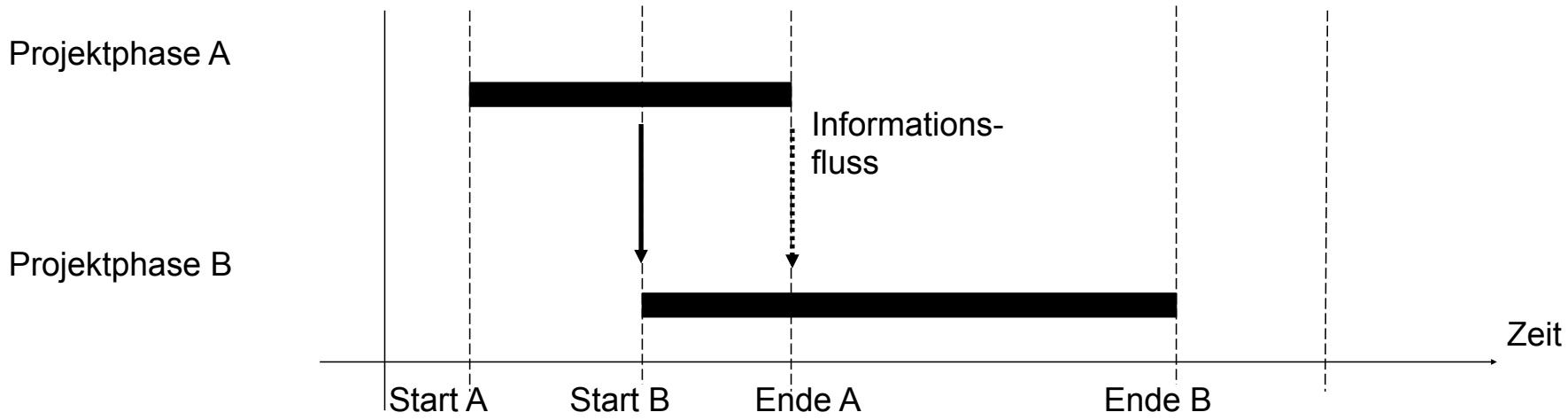
- Start von Phase B mit Teilinformationen von Phase A ohne frühes Einfrieren von Entscheidungen in Phase A
- Vorteile: Kürzere Bearbeitungsdauer
- Nachteile: Risiko von Verzögerungen durch Iterationen in der Phase B
- Beispiel: Anwendungssoftware (Phase B) wird auf Rapid Prototyping System vor Fertigstellung der Plattformsoftware (Phase A) entwickelt
  Risiko: Plattformsoftware verhält sich anders wie Rapid Prototyping System



Parallele Planung mit Einfrieren

- Start von Phase B mit Teilinformationen von Phase A mit frühem Einfrieren einiger Entscheidungen in Phase A

- Vorteile: Kürzere Bearbeitungsdauer
- Nachteile: Risiko von Qualitätseinbussen durch frühe Einschränkungen in der Phase A
- Beispiel: Kalibrierung von implementierten Funktionen der Anwendungssoftware (Phase B) vor Fertigstellung aller Funktionen (Phase A)

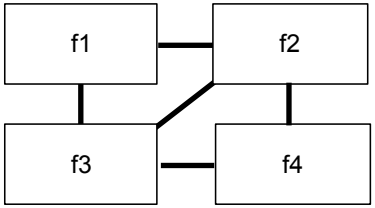


Rollen und Aufgabengebiete im Entwicklungsprozess
siehe 6. SW-Entwicklung / 1. Kernprozess

Rolle	Aufgabengebiet
Funktionsentwicklung	Analyse der Benutzeranforderungen und Spezifikation der logischen Systemarchitektur
Systementwicklung	Analyse der logischen Systemarchitektur und Spezifikation der technischen Systemarchitektur
Software-Entwicklung	Analyse der Software-Anforderungen, Spezifikation, Design, Implementierung, Integration und Test der Software
Hardware-Entwicklung	Analyse der Hardware-Anforderungen, Spezifikation, Design, Realisierung, Integration und Test der Hardware
Sensor-/Sollwertgeber-/ Aktuator-Entwicklung	Analyse der Anforderungen, Spezifikation, Design, Realisierung, Integration und Test von Sensoren, Sollwertgebern und Aktuatoren
Integration, Erprobung und Kalibrierung	Integration, Test und Kalibrierung von Systemen des Fahrzeugs und deren Funktionen

Schnittstellen für Spezifikation und Integration (Nach Schäuffele, Zurawka)

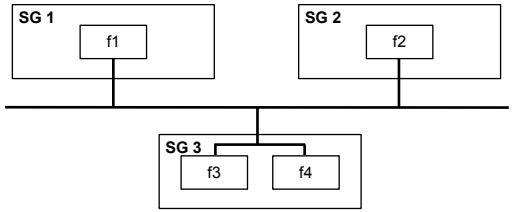
Analyse der Benutzeranforderungen und Spezifikation der logischen Systemarchitektur



Logische Systemarchitektur

Akzeptanz- und Systemtest

Analyse der logischen Systemarchitektur und Spezifikation der technischen Systemarchitektur



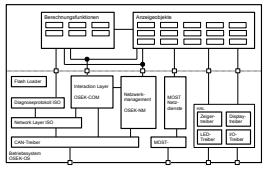
Technische Systemarchitektur

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

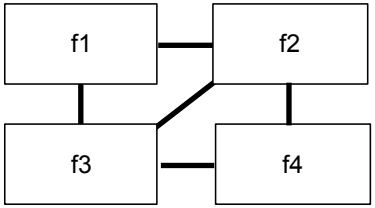
Spezifikation der Software-Komponenten

Design und Implementierung der Software-Komponenten

Test der Software-Komponenten

Funktionsentwicklung

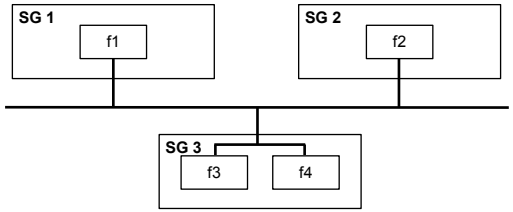
Analyse der Benutzeranforderungen und Spezifikation der logischen Systemarchitektur



Logische Systemarchitektur

Akzeptanz- und Systemtest

Analyse der logischen Systemarchitektur und Spezifikation der technischen Systemarchitektur



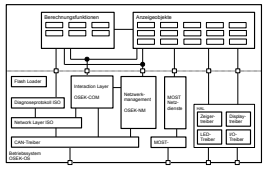
Technische Systemarchitektur

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

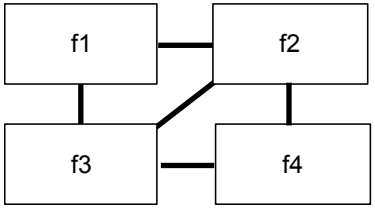
Spezifikation der Software-Komponenten

Design und Implementierung der Software-Komponenten

Test der Software-Komponenten

Systementwicklung

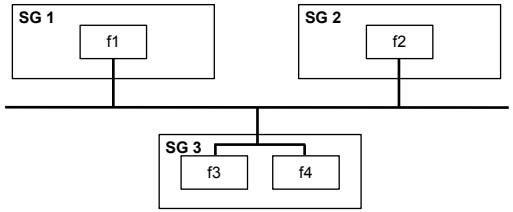
Analyse der Benutzeranforderungen und Spezifikation der logischen Systemarchitektur



Logische Systemarchitektur

Akzeptanz- und Systemtest

Analyse der logischen Systemarchitektur und Spezifikation der technischen Systemarchitektur



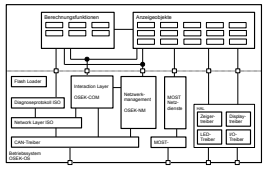
Technische Systemarchitektur

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

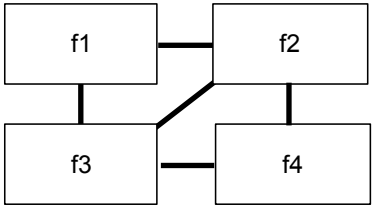
Spezifikation der Software-Komponenten

Design und Implementierung der Software-Komponenten

Test der Software-Komponenten

Software-Entwicklung (Hardware-Entwicklung, Sensor-/Sollwertgeber-/Aktuator-Entwicklung)

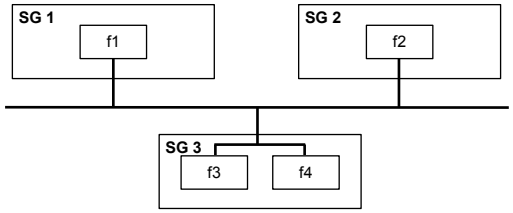
Analyse der Benutzeranforderungen und Spezifikation der logischen Systemarchitektur



Logische Systemarchitektur

Akzeptanz- und Systemtest

Analyse der logischen Systemarchitektur und Spezifikation der technischen Systemarchitektur



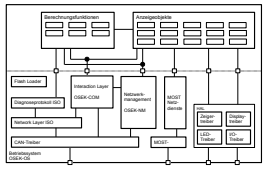
Technische Systemarchitektur

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

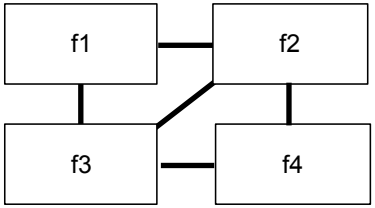
Spezifikation der Software-Komponenten

Design und Implementierung der Software-Komponenten

Test der Software-Komponenten

Integration, Erprobung und Kalibrierung

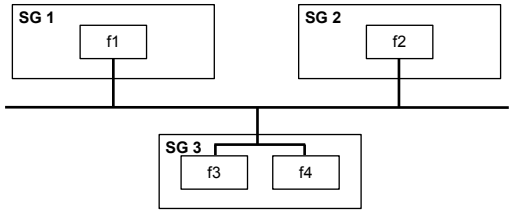
Analyse der Benutzeranforderungen und Spezifikation der logischen Systemarchitektur



Logische Systemarchitektur

Akzeptanz- und Systemtest

Analyse der logischen Systemarchitektur und Spezifikation der technischen Systemarchitektur



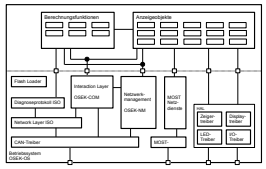
Technische Systemarchitektur

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

6. SW-Entwicklung / 3. Unterstützungsprozesse
Unterstützungsprozesse für die Embedded Software Entwicklung

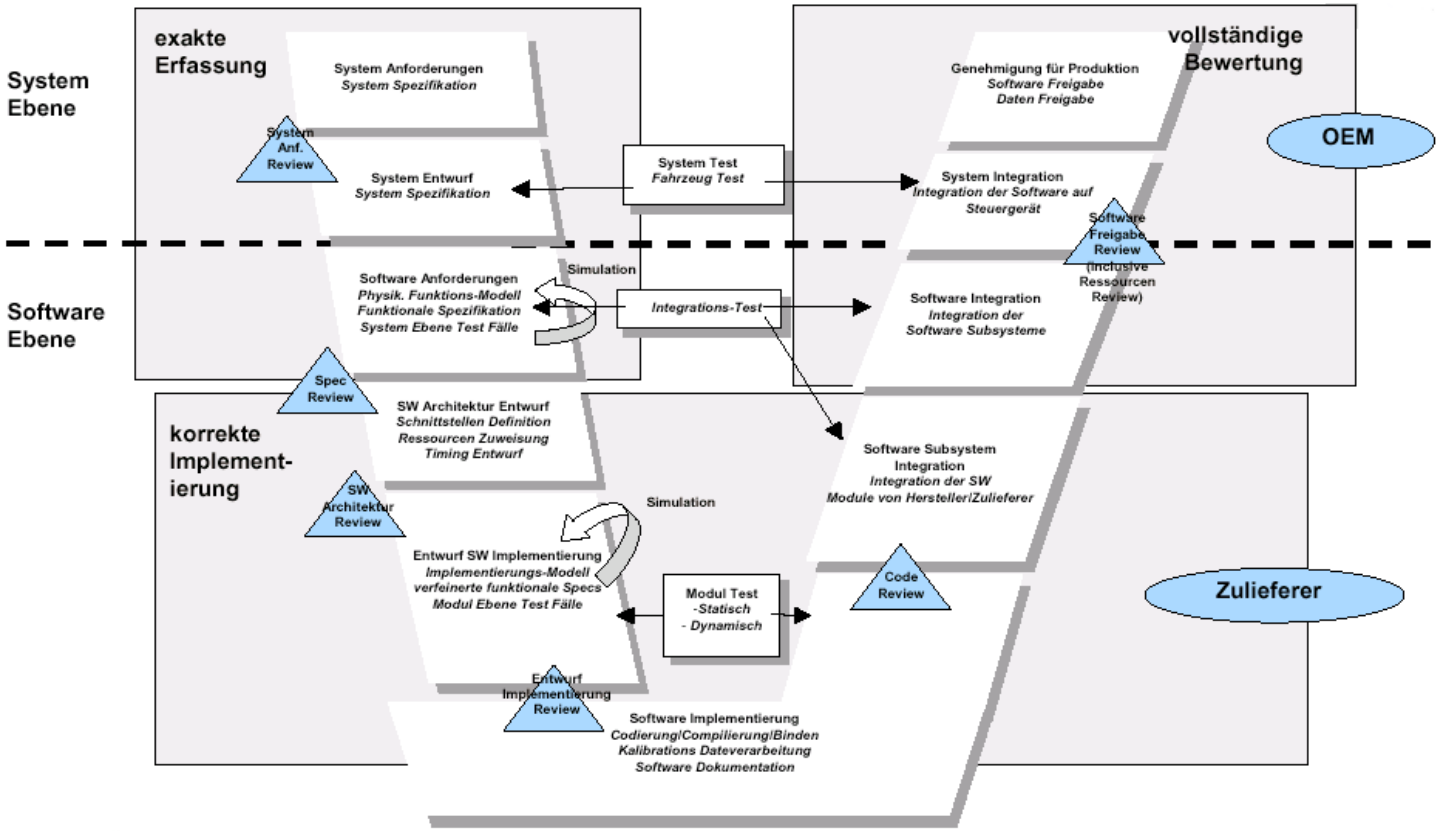
1. Vorgehensmodelle und Standards
2. Konfigurationsmanagement
3. Projektmanagement
4. **Lieferantenmanagement**
    1. **System- und Komponentenverantwortung**
    2. Schnittstellen für Spezifikation und Integration
    3. Festlegung des firmenübergreifenden Entwicklungsprozesses
5. Anforderungsmanagement
6. Qualitätssicherung

Entwicklung elektronischer Systeme im Automobil

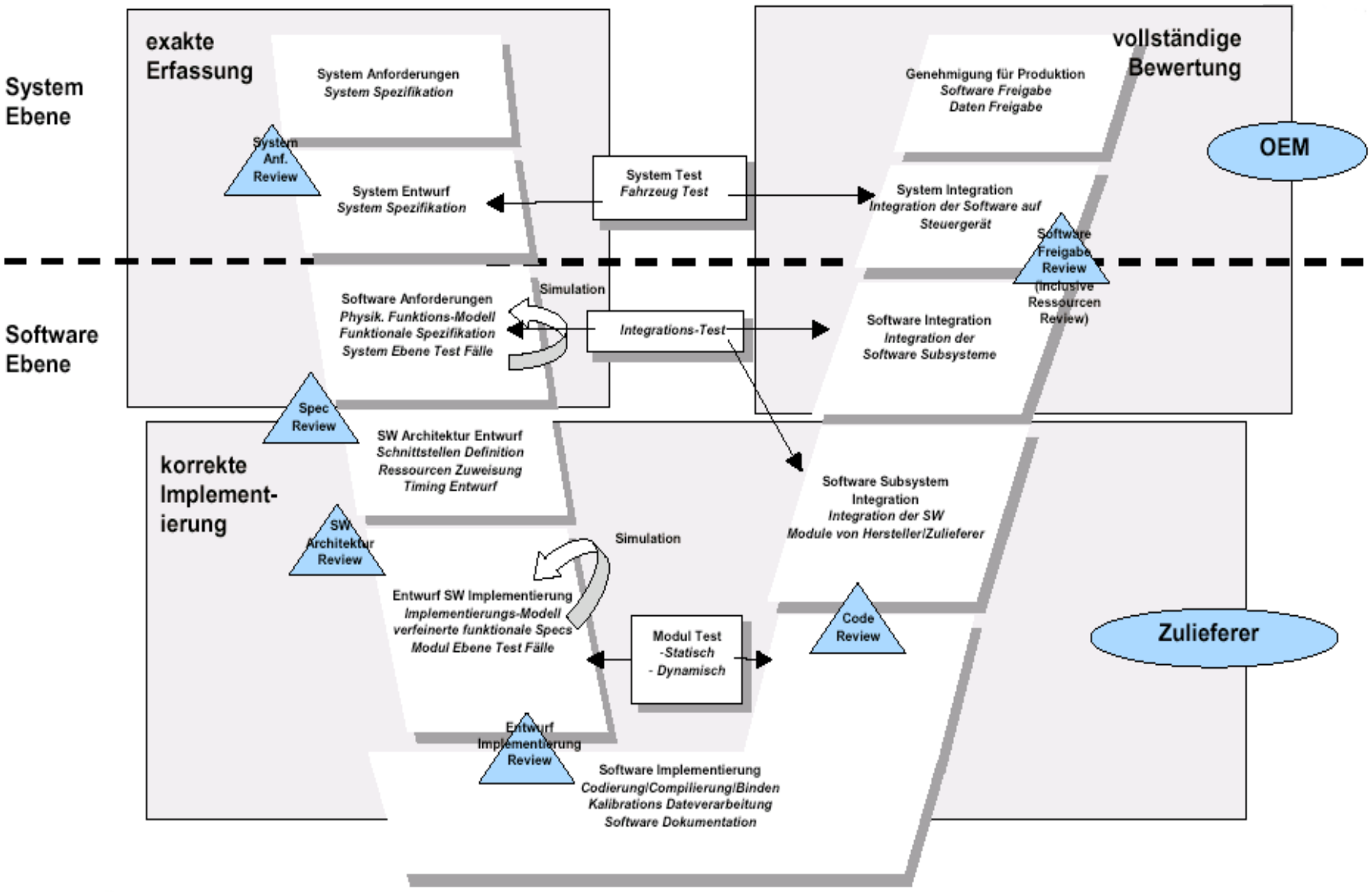
- Starke Arbeitsteilung zwischen Fahrzeugherstellern und Zulieferern
- Festlegung der Anforderungen an eine zu entwickelnde Funktion
  - Fahrzeughersteller
- Realisierung der Funktion durch elektronische Systeme
  - Zulieferer
- Abstimmung und Abnahme der Funktion im Fahrzeug
  - Fahrzeughersteller
- Firmenübergreifende Zusammenarbeit
  - Technische Aspekte
  - Organisatorische Aspekte
  - Rechtliche Aspekte
- Lieferantenmanagement
  - Alle Aufgaben, die im Rahmen der Systementwicklung an den Schnittstellen zwischen Fahrzeughersteller und Zulieferern beachtet werden müssen

System- und Komponentenverantwortung

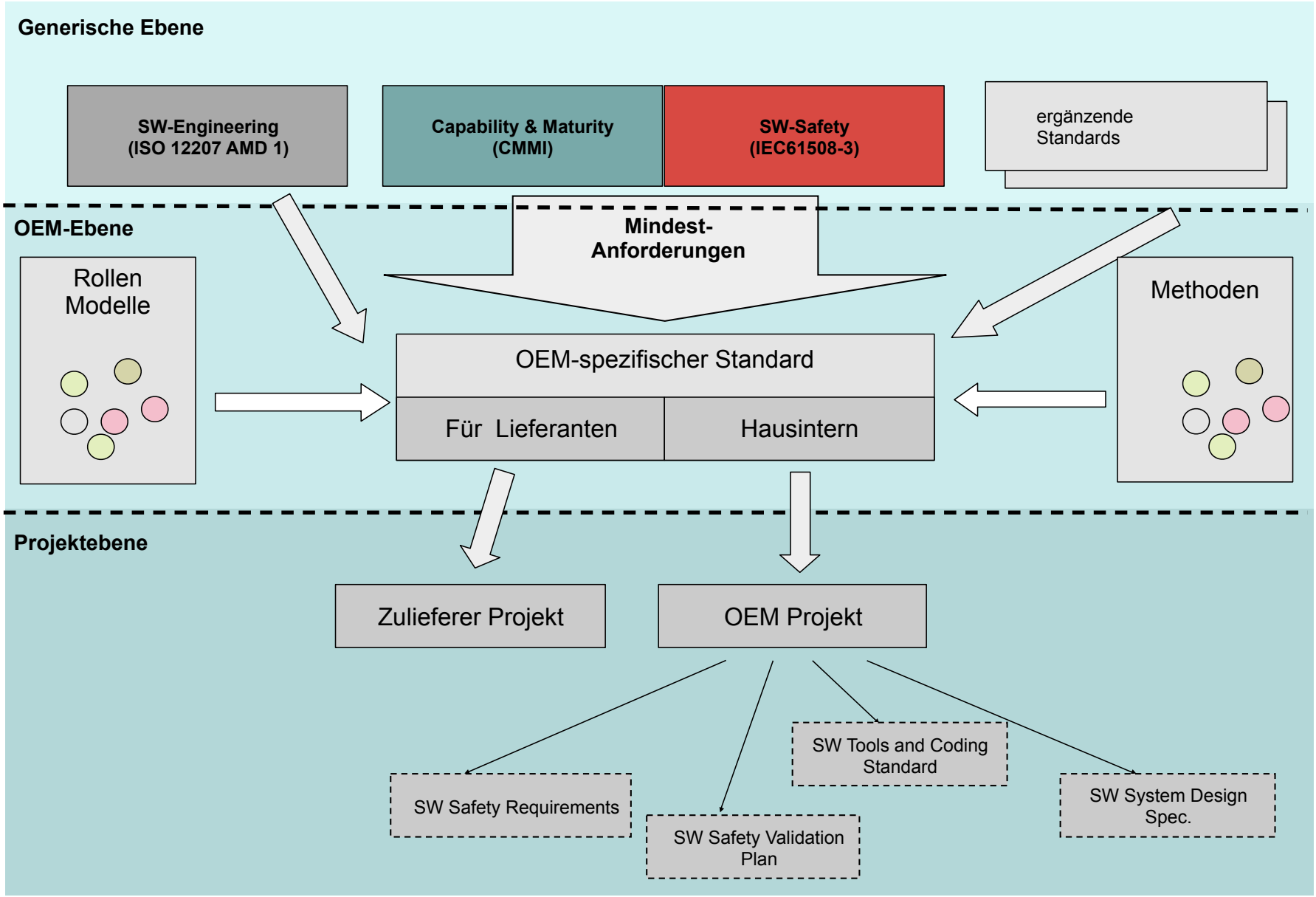
- Präzise Definition der Schnittstellen zwischen Fahrzeughersteller und Zulieferern
- Orientierung am V-Modell
- Fahrzeughersteller
  - Gesamtfahrzeug
- Zulieferer
  - Komponenten



System- und Komponentenverantwortung



BMW Group Standard embedded Software (GSeSW)



6. SW-Entwicklung / 3. Unterstützungsprozesse
Unterstützungsprozesse für die Embedded Software Entwicklung

1. Vorgehensmodelle und Standards
2. Konfigurationsmanagement
3. Projektmanagement
4. **Lieferantenmanagement**
    1. System- und Komponentenverantwortung
    2. **Schnittstellen für Spezifikation und Integration**
    3. Festlegung des firmenübergreifenden Entwicklungsprozesses
5. Anforderungsmanagement
6. Qualitätssicherung

Schnittstellen für Spezifikation und Integration

- Zwei Arten von Schnittstellen
  - Spezifikationsschnittstelle im linken Ast des V-Modellls
  - Integrationsschnittstelle im rechten Ast des V-Modellls
- Grosse Komplexität
  - Hersteller
    - n Komponenten von n verschiedenen Zulieferern
    - 1:n-Beziehung an der Spezifikationsschnittstelle
    - 1:n-Beziehung an der Integrationsschnittstelle
  - Zulieferer
    - 1 Komponente an m verschiedene Hersteller
    - 1:m-Beziehung an der Spezifikationsschnittstelle
    - 1:m-Beziehung an der Integrationsschnittstelle

Herstellersicht
Baugruppenverantwortlicher Türe

- Ansprechpartner
  - Baugruppenverantwortlicher Karosserie
  - Baugruppenverantwortlicher Sitze
  - Baugruppenverantwortlicher Kombi-Instrument
  - Baugruppenverantwortlicher Blinker
  - Baugruppenverantwortlicher Mittelkonsole
  - Baugruppenverantwortlicher Soundsystem
  - Baugruppenverantwortlicher Seitenairbag
  - Verantwortlicher Passive Sicherheit
  - Verantwortlicher EMV
  - Verantwortlicher Verkabelung
  - Verantwortlicher Vernetzung
  - Verantwortlicher Telematik

- **Zulieferer**
  - **Schliesssystem**
  - **Scheiben**
  - **Fensterheber**
  - **Aussenspiegel**
  - **Türsteuergerät**
  - **Schalter**

- Schnittstellen
  - Mechanik
  - Energie
  - Information

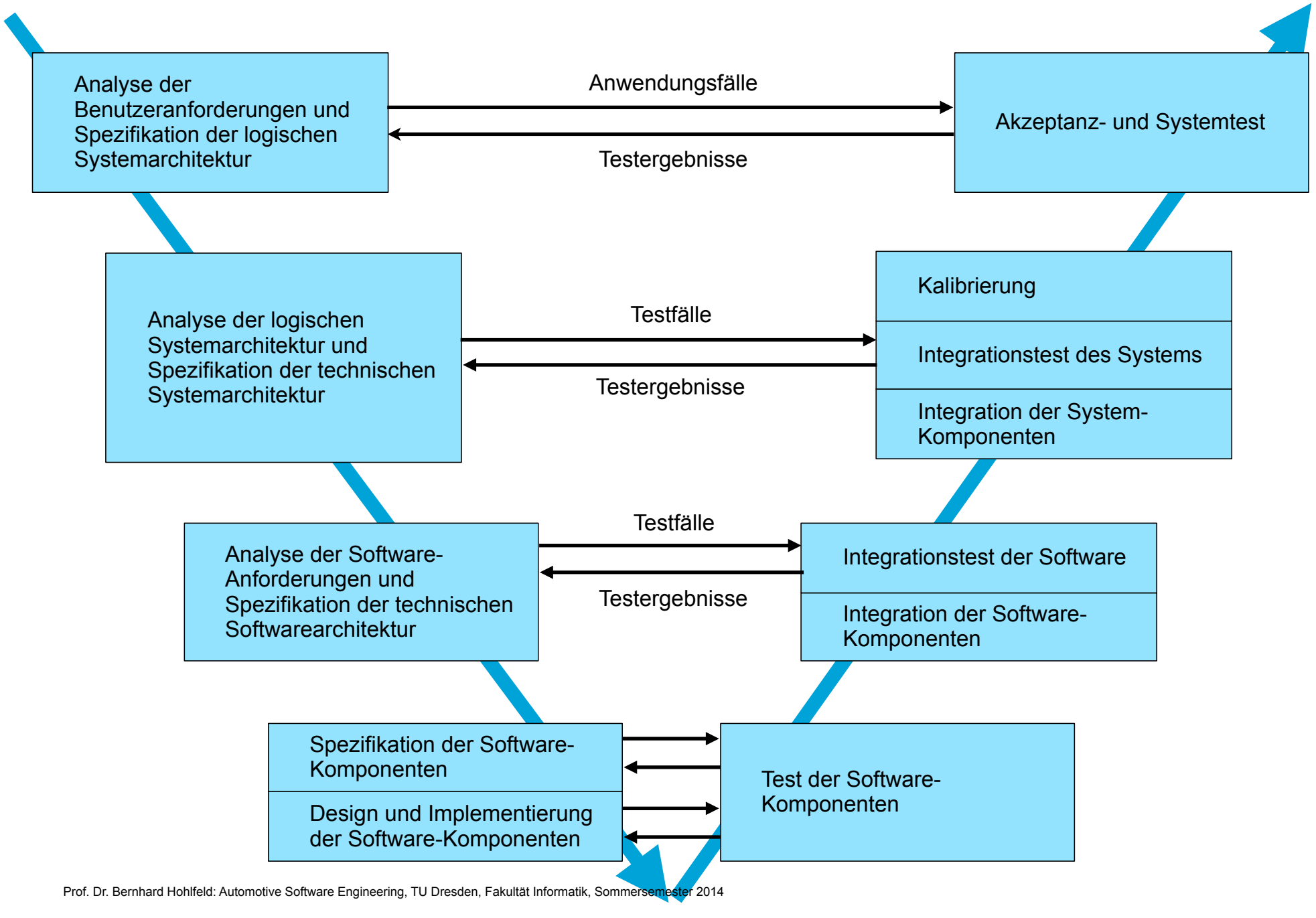
Herstellersicht (Prinzipdarstellung)

	Bosch	Conti	Kostal	Dräxl-meier	Magna	Delphi	TRW
Schliesssystem		X	X				
Scheiben			X				
Fensterheber			X	X	X		
Aussenspiegel					X		X
Türsteuergerät	X	X					
Schalter						X	X

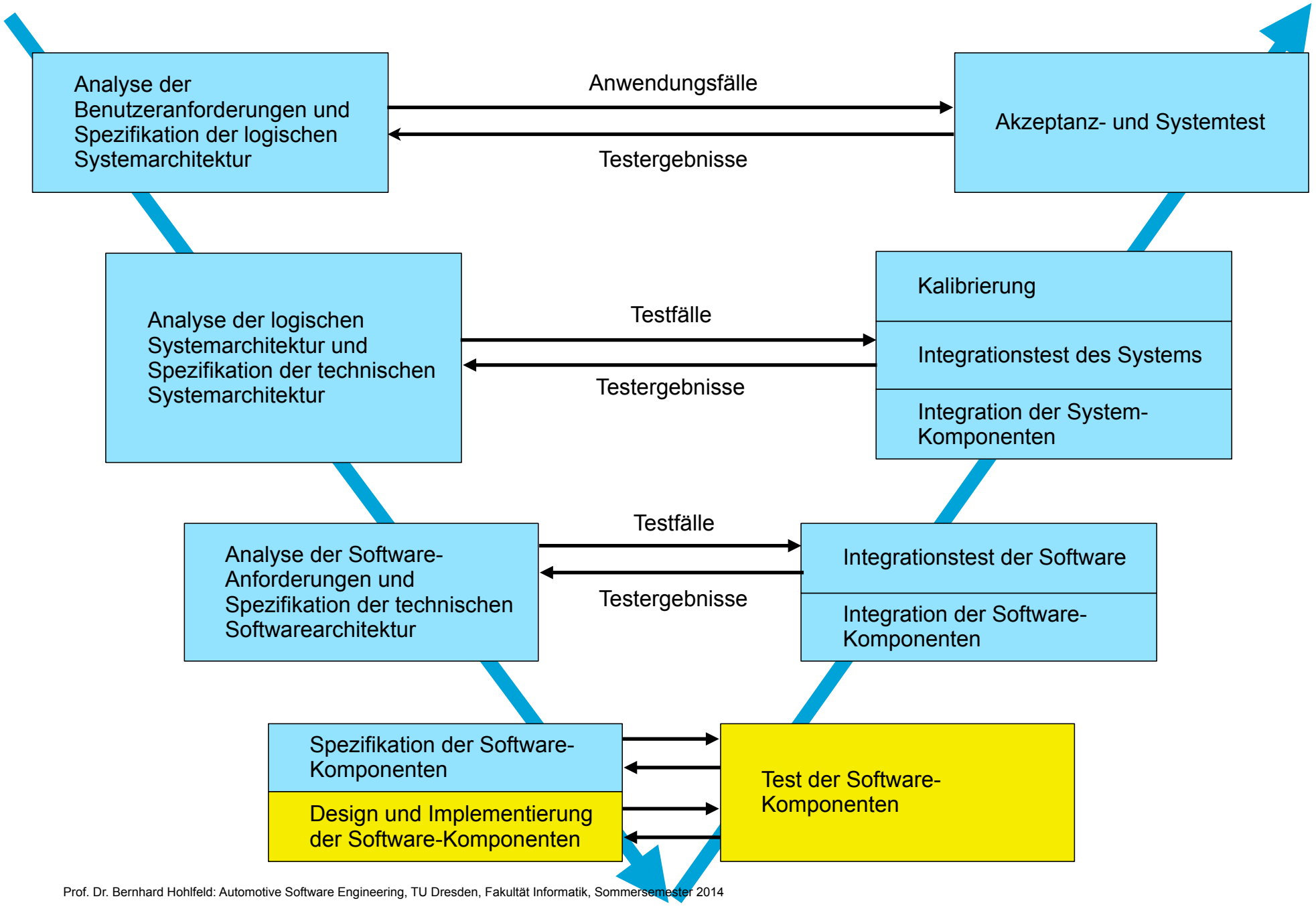
Zulieferersicht (Prinzipdarstellung)

	BMW	Mer-cedes	Audi	Porsche	VW	Ford	Opel
Schliesssystem	X	X	X				
Scheiben			X				
Fensterheber			X	X	X		
Aussenspiegel	X	X	X	X	X	X	X
Türsteuergerät		X					
Schalter						X	X

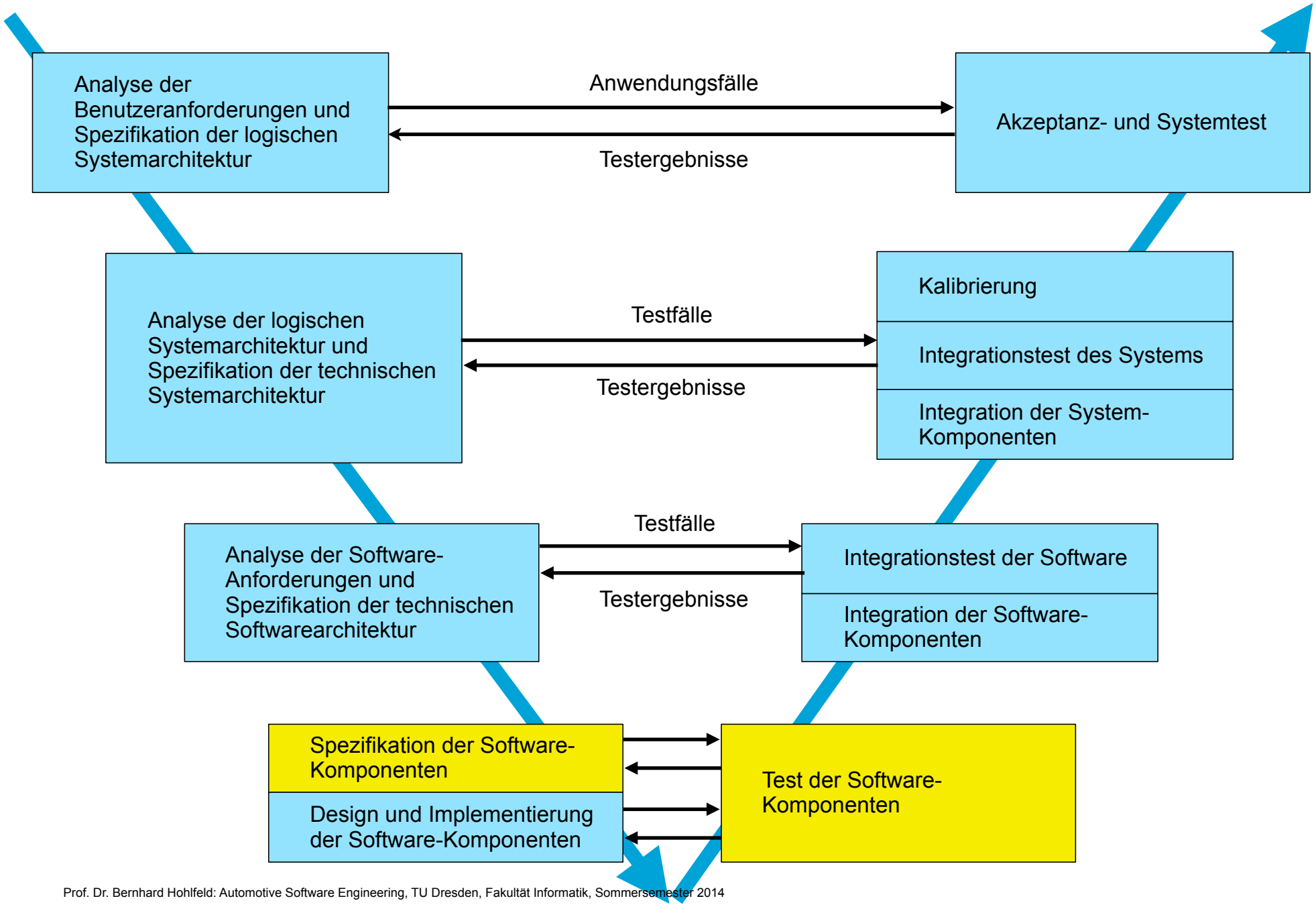
Kernprozess zur Entwicklung von elektronischen Systemen und Software (Nach Schäuffele, Zurawka)



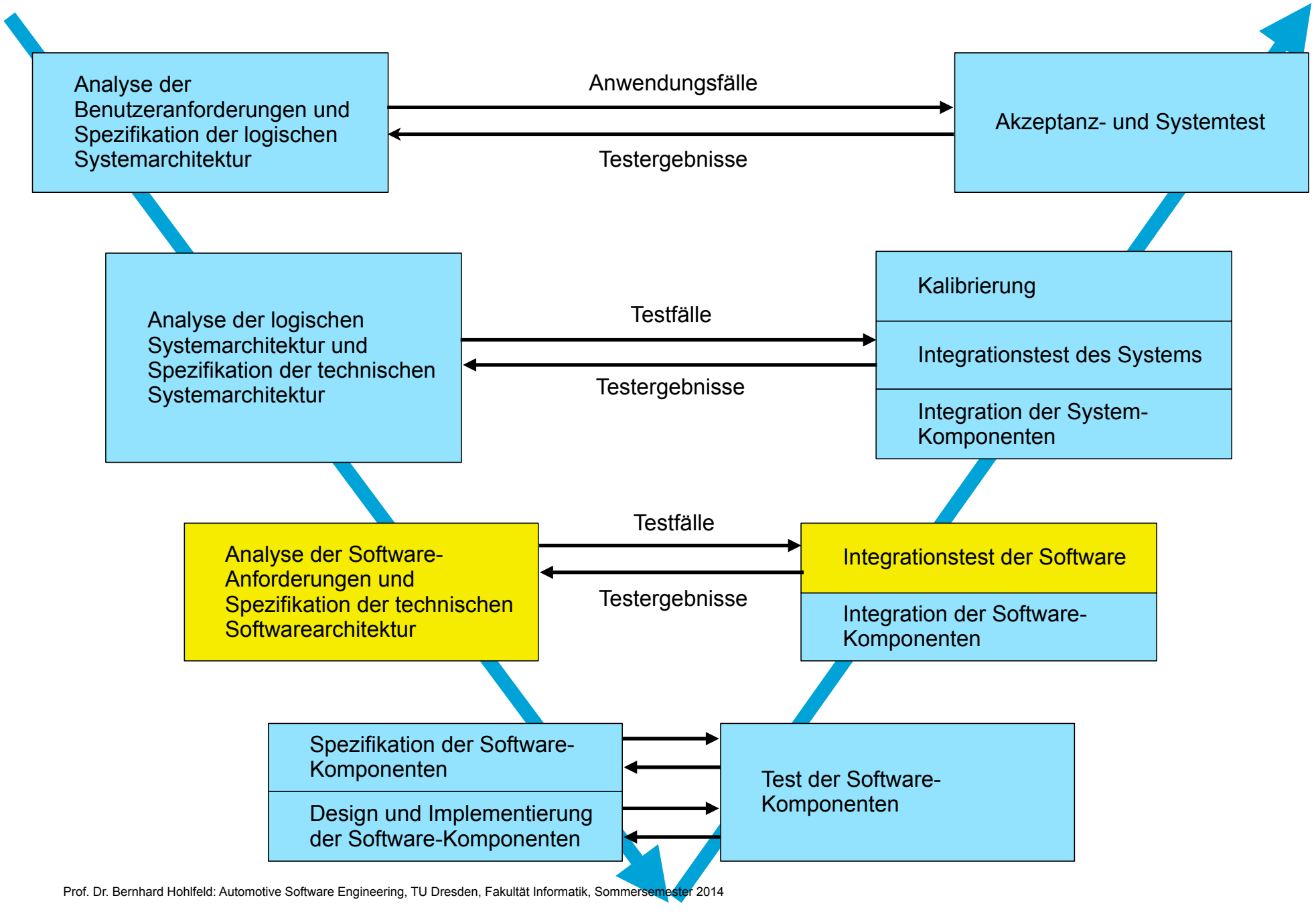
Kernprozess zur Entwicklung von elektronischen Systemen und Software (Nach Schäuffele, Zurawka)



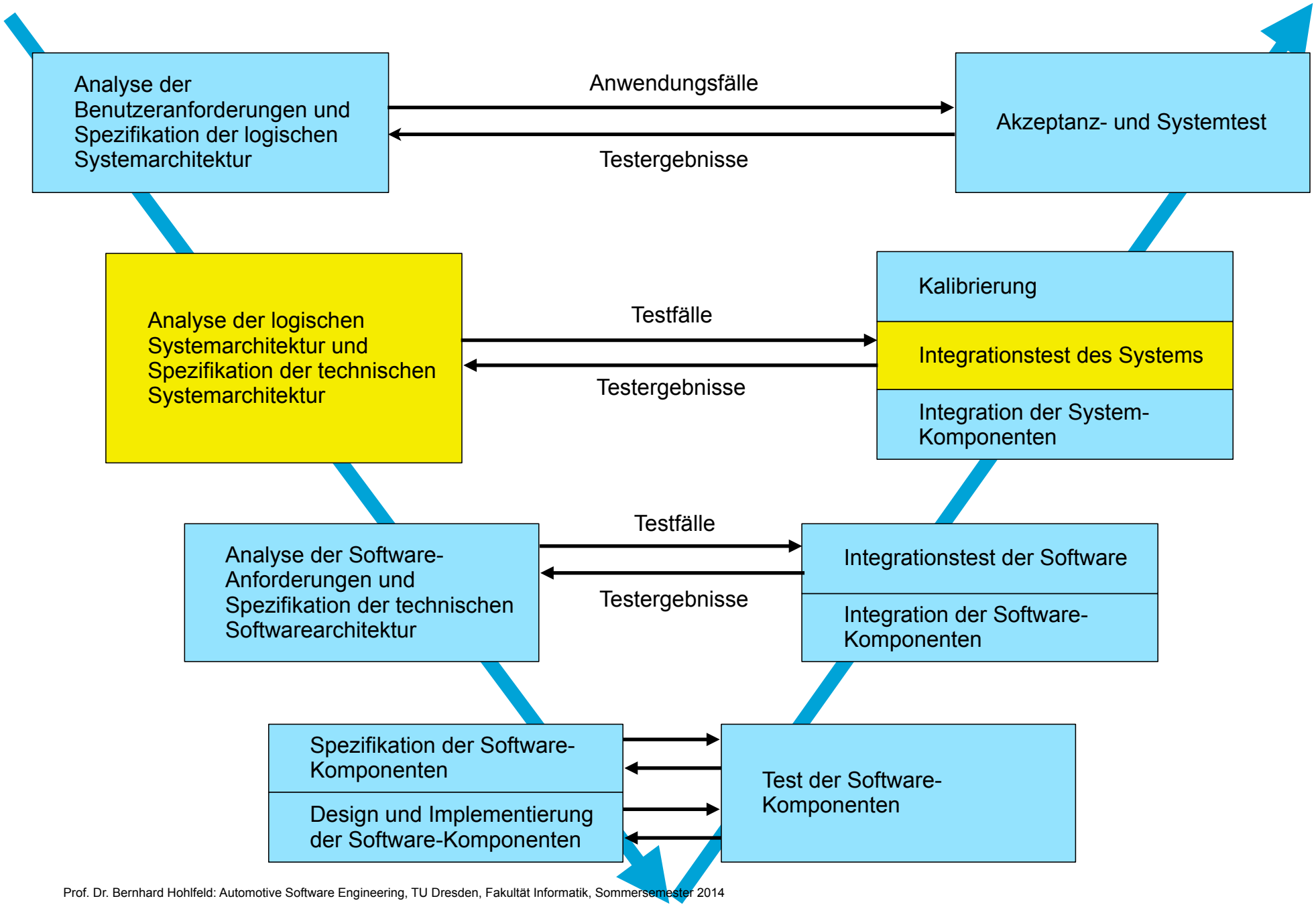
Kernprozess zur Entwicklung von elektronischen Systemen und Software (Nach Schäuffele, Zurawka)



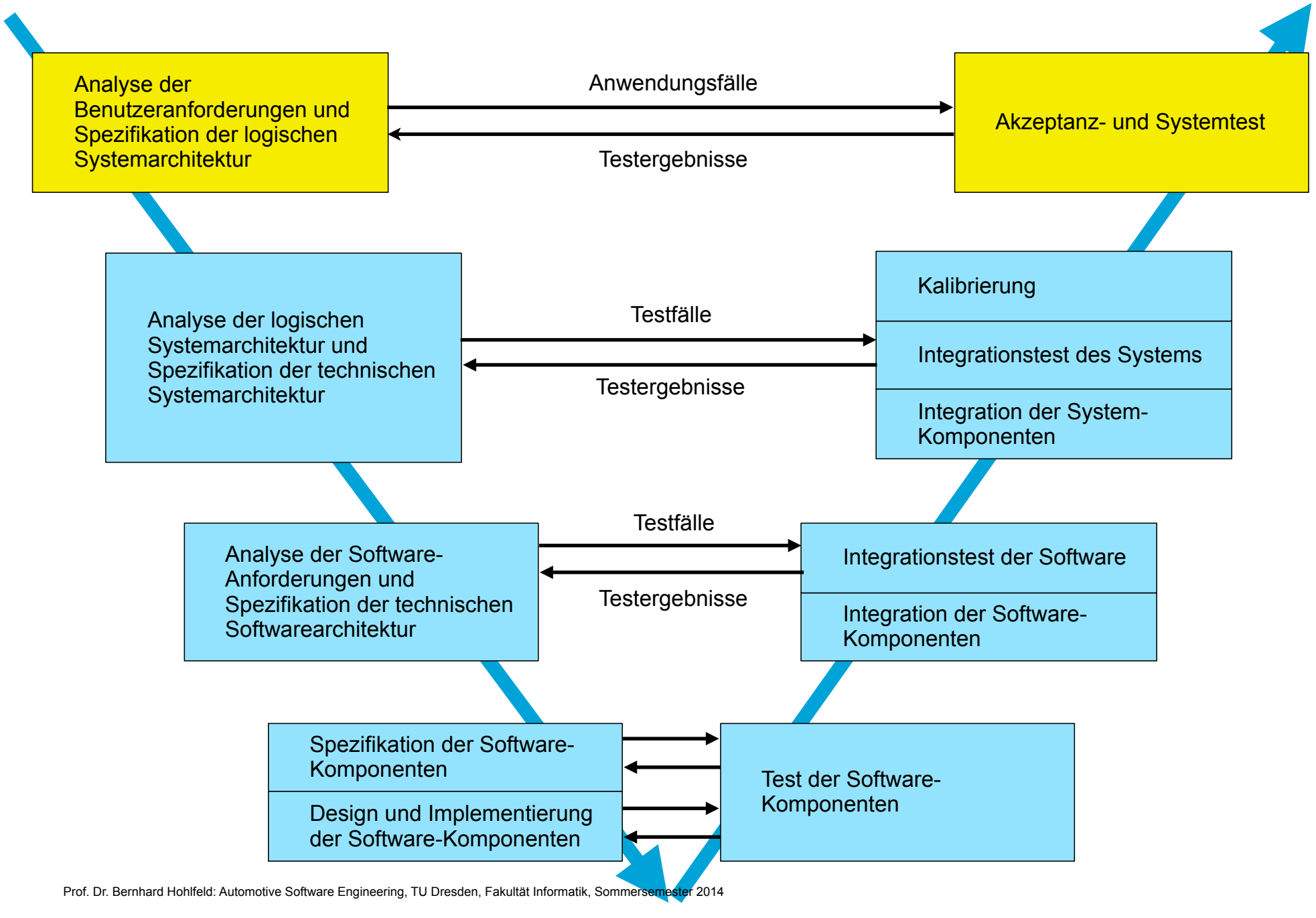
Kernprozess zur Entwicklung von elektronischen Systemen und Software (Nach Schäuffele, Zurawka)



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



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



Unterstützungsprozesse für die Embedded Software Entwicklung

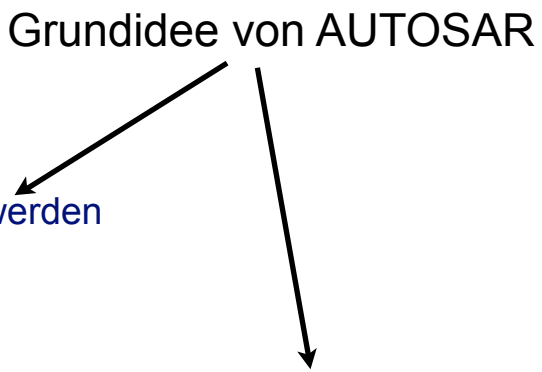
1. Vorgehensmodelle und Standards
2. Konfigurationsmanagement
3. Projektmanagement
4. Lieferantenmanagement
    1. System- und Komponentenverantwortung
    2. Schnittstellen für Spezifikation und Integration
    3. Festlegung des firmenübergreifenden Entwicklungsprozesses
5. Anforderungsmanagement
6. Qualitätssicherung

Festlegung des firmenübergreifenden Entwicklungsprozesses

- Elektronische Steuergeräte sind eingebettete Systeme, die für den Benutzer nicht unmittelbar in Erscheinung treten
  - Beispiel: Türsteuergerät
- Für den Benutzer sind die Funktionen sichtbar
  - Beispiele:
    - Fensterheber
    - Spiegelverstellung
    - Sitzverstellung
- Grundfunktionen können herstellerübergreifend entwickelt werden
  - Beispiel:
    - Steuerung von Fensterheber, Spiegelverstellung, Sitzverstellung
- Wettbewerbsdifferenzierende Zusatzfunktionen werden herstellerspezifisch entwickelt
  - Beispiel:
    - Automatische Spiegelverstellung und Sitzverstellung (Memory, Ergonomie)

Festlegung des firmenübergreifenden Entwicklungsprozesses

- Elektronische Steuergeräte sind eingebettete Systeme, die für den Benutzer nicht unmittelbar in Erscheinung treten
  - Beispiel: Türsteuergerät
- Für den Benutzer sind die Funktionen sichtbar
  - Beispiele:
    - Fensterheber
    - Spiegelverstellung
    - Sitzverstellung
- Grundfunktionen können herstellerübergreifend entwickelt werden
  - Beispiel:
    - Steuerung von Fensterheber, Spiegelverstellung, Sitzverstellung
- Wettbewerbsdifferenzierende Zusatzfunktionen werden herstellerspezifisch entwickelt
  - Beispiel:
    - Automatische Spiegelverstellung und Sitzverstellung (Memory, Ergonomie)



LOV-Diagramme

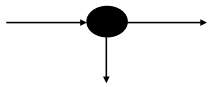
- LOV: Line of Visibility
- Grafische Beschreibung der komplexen Beziehungen zwischen Herstellern und Zulieferern

In der **linken Spalte des LOV-Diagramms**:
Für Prozessschritte verantwortliche Organisationseinheit
In der **oberen Zeile des LOV-Diagramms**:
Prozessschritte des Kunden

Organisationseinheit des Kunden bzw. des Lieferanten

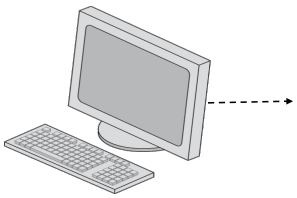
Prozessschritt

Verbindung zwischen zwei Prozessschritten:
Der Pfeil sagt aus: „wird angestoßen vom Vorgänger“



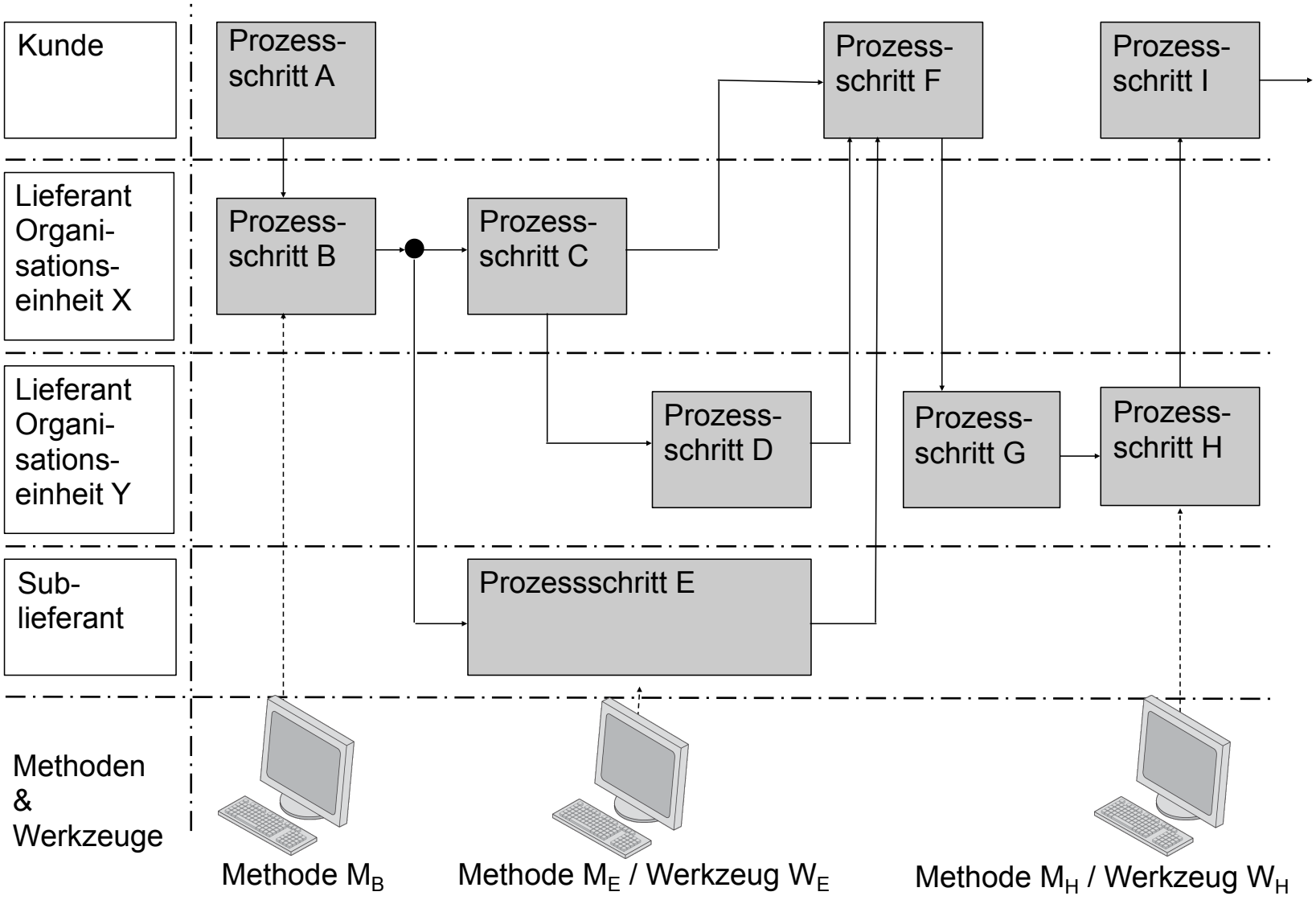
Fallunterscheidung, die zum Ausdruck bringt, dass im Weiteren unterschiedliche Prozessverläufe eingeschlagen werden können

Trennlinie zwischen zwei Organisationseinheiten

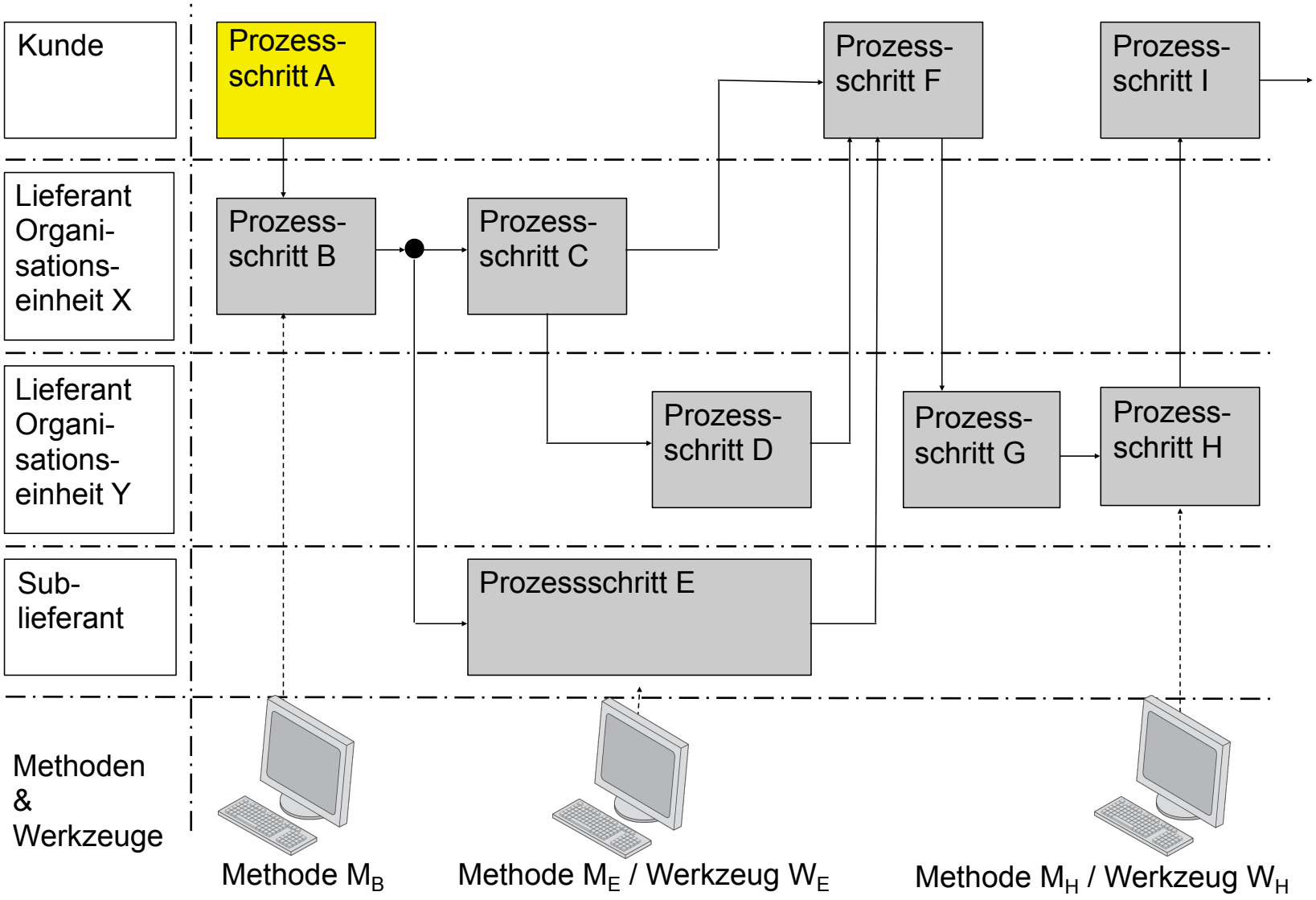


Methode oder Werkzeug für Einsatz bei einem Prozessschritt

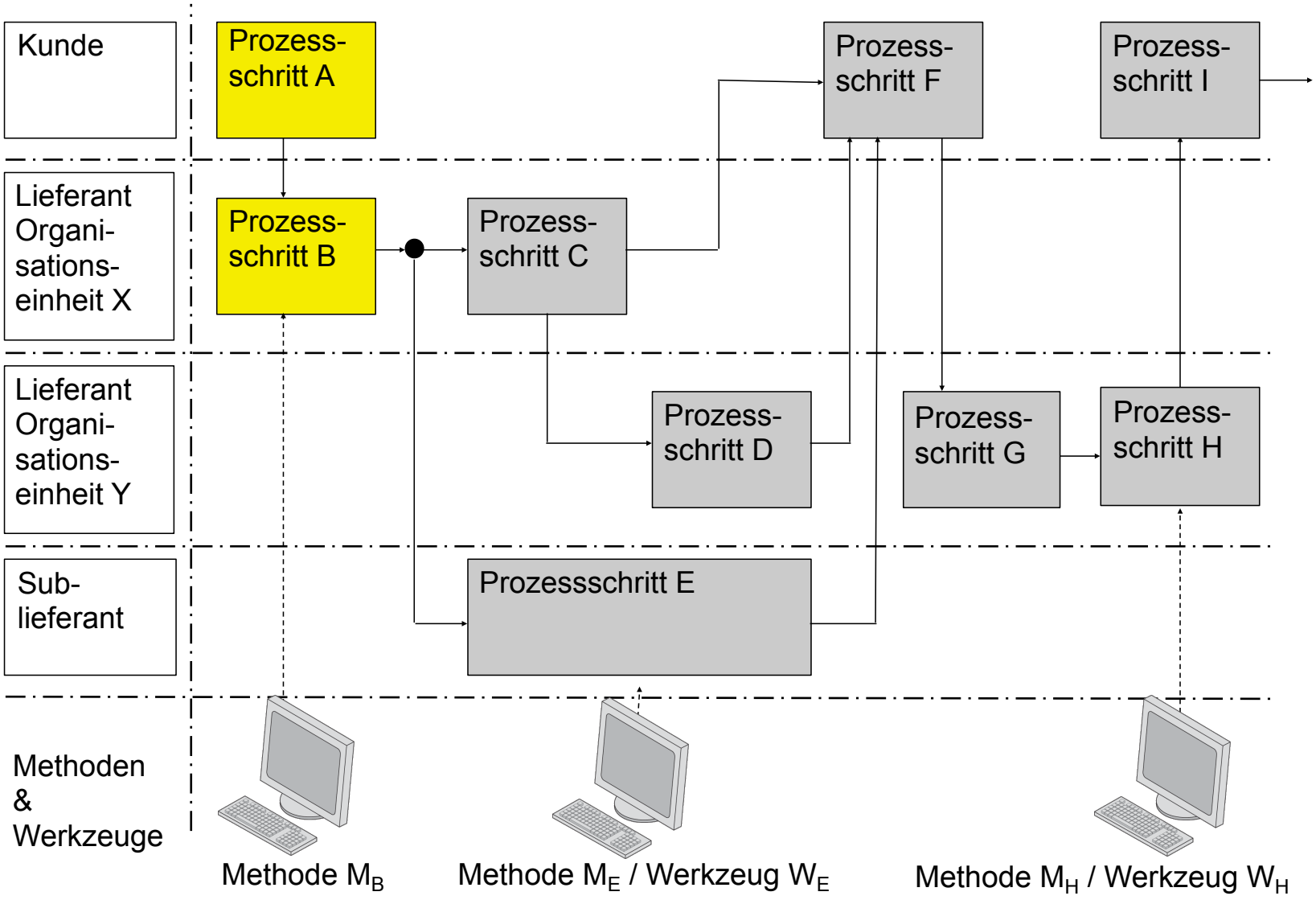
LOV-Diagramme: Beispiel



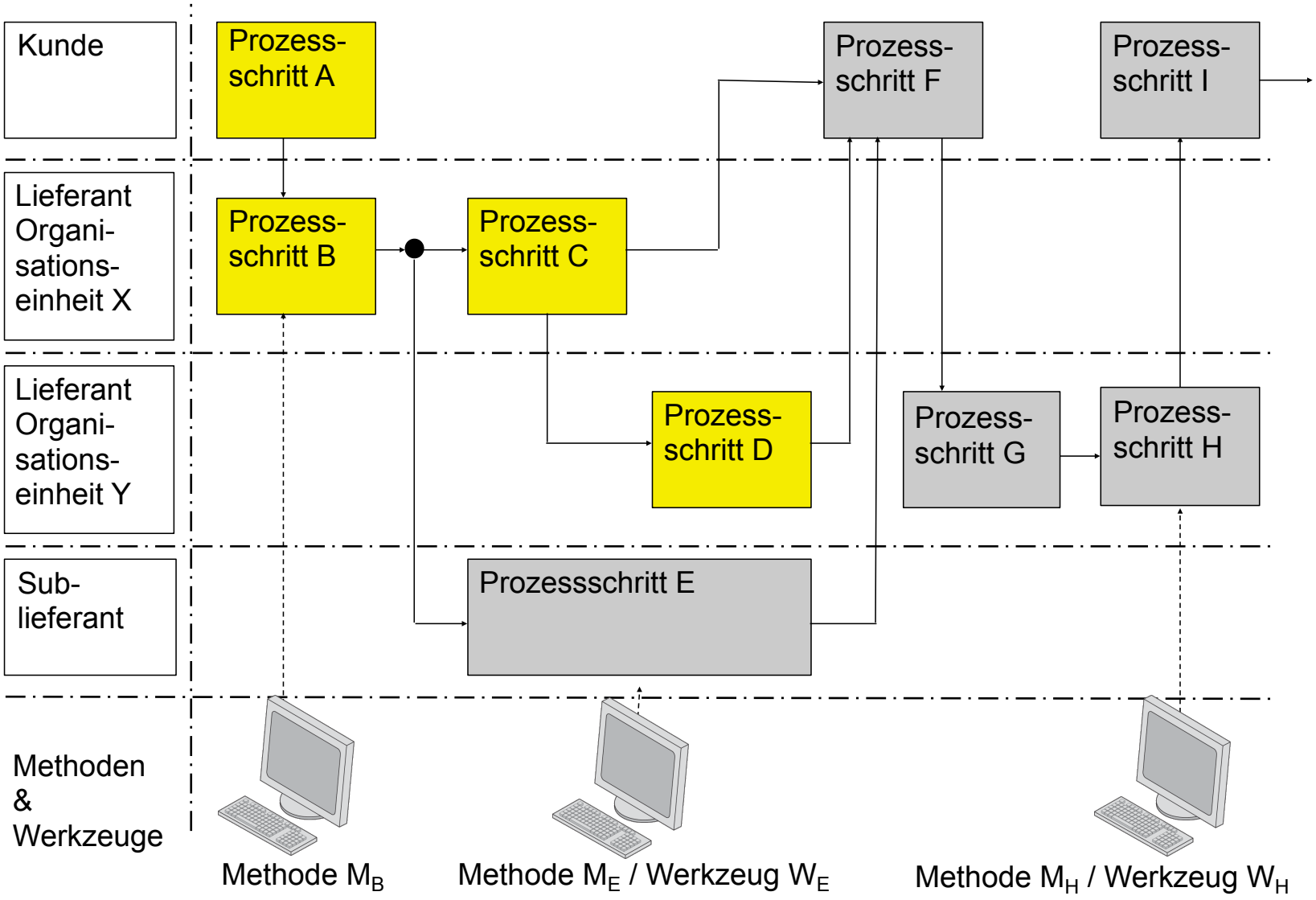
Prozessschritt A:
Spezifikation der logischen Systemarchitektur



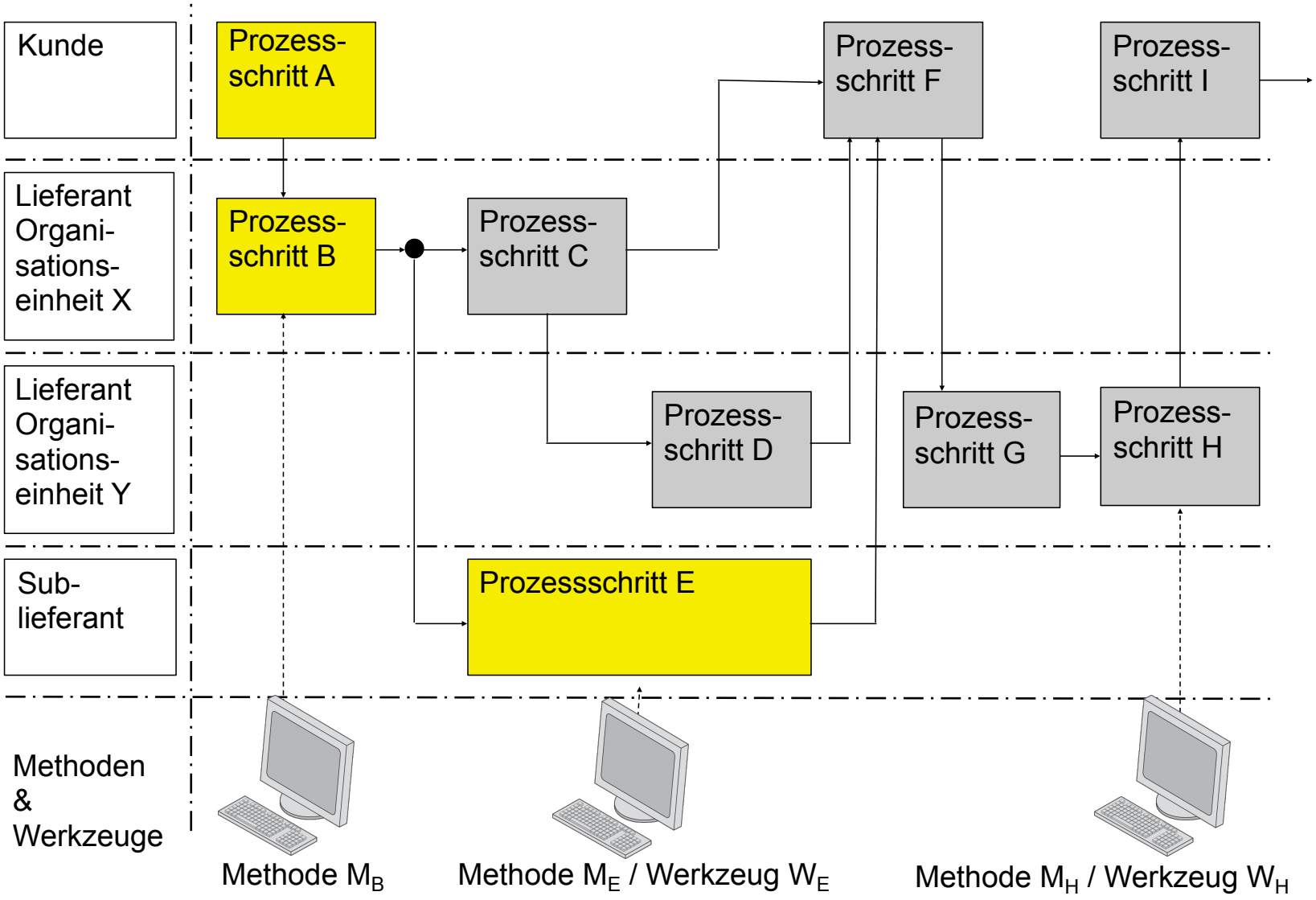
Prozessschritt B:
Spezifikation der technischen Systemarchitektur



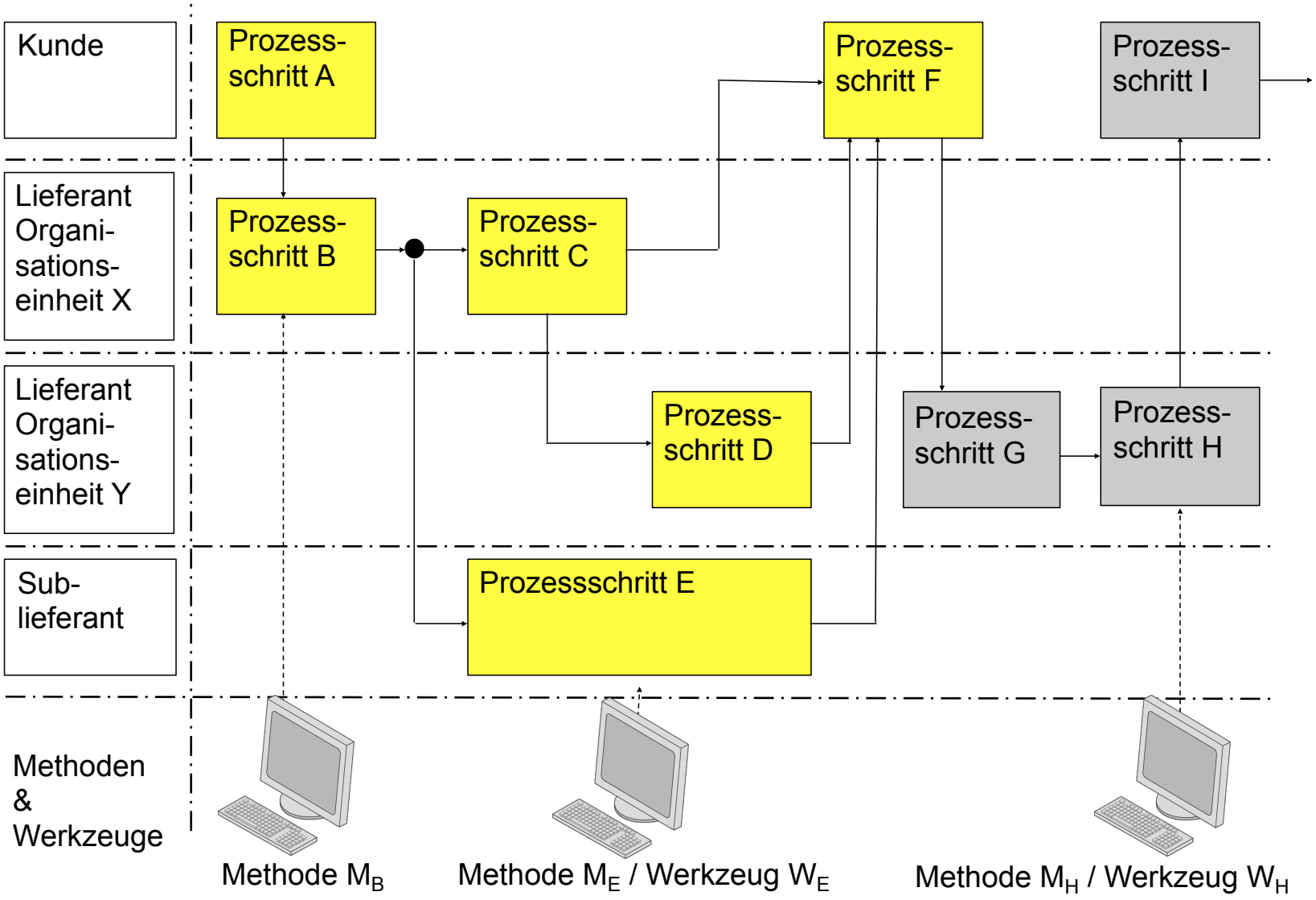
Prozessschritte C und D:
SW-Realisierung (D) auf neu entwickeltem Steuergerät (C)



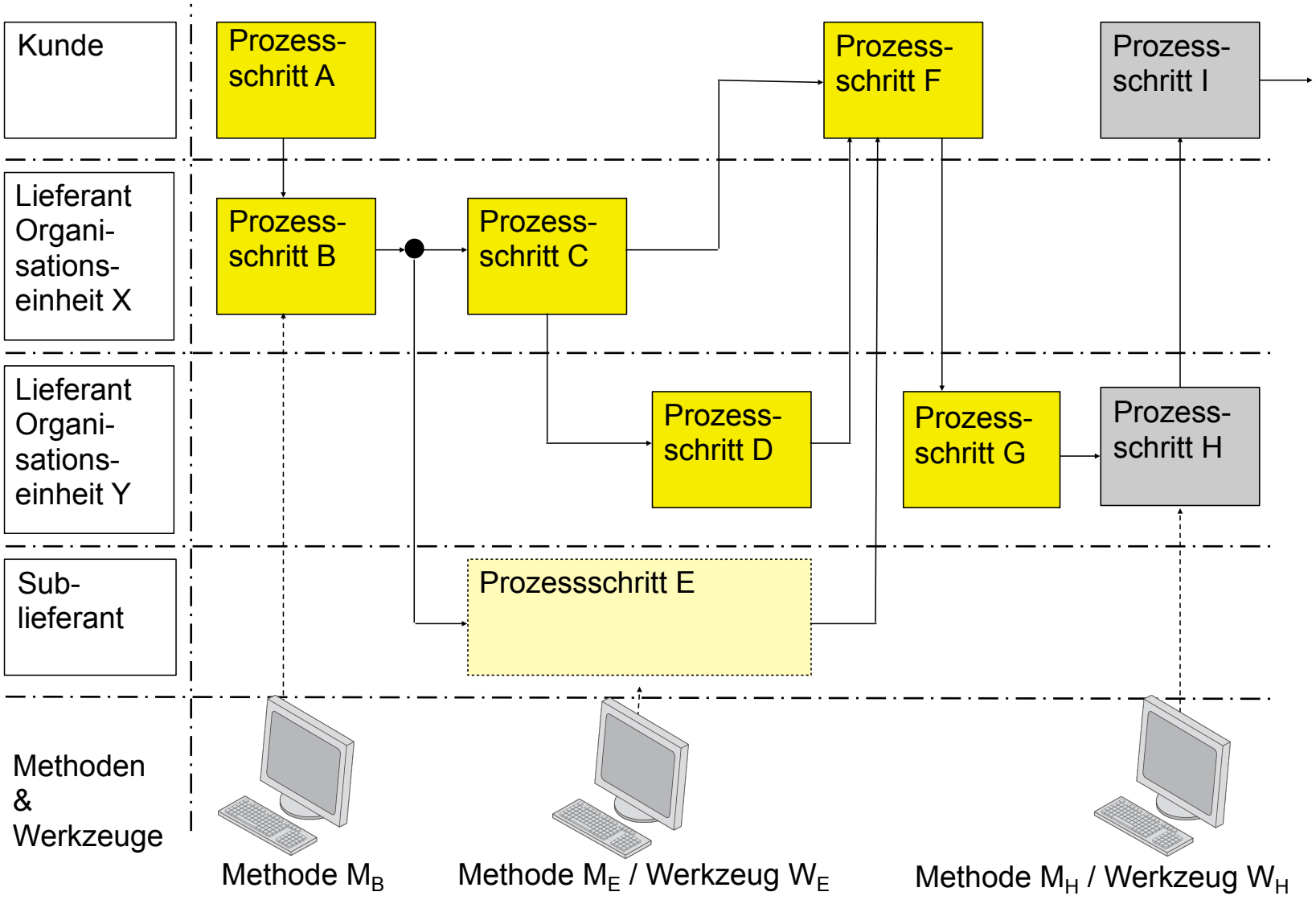
Alternativ Prozessschritt E:
SW-Realisierung auf vorhandenem Steuergerät



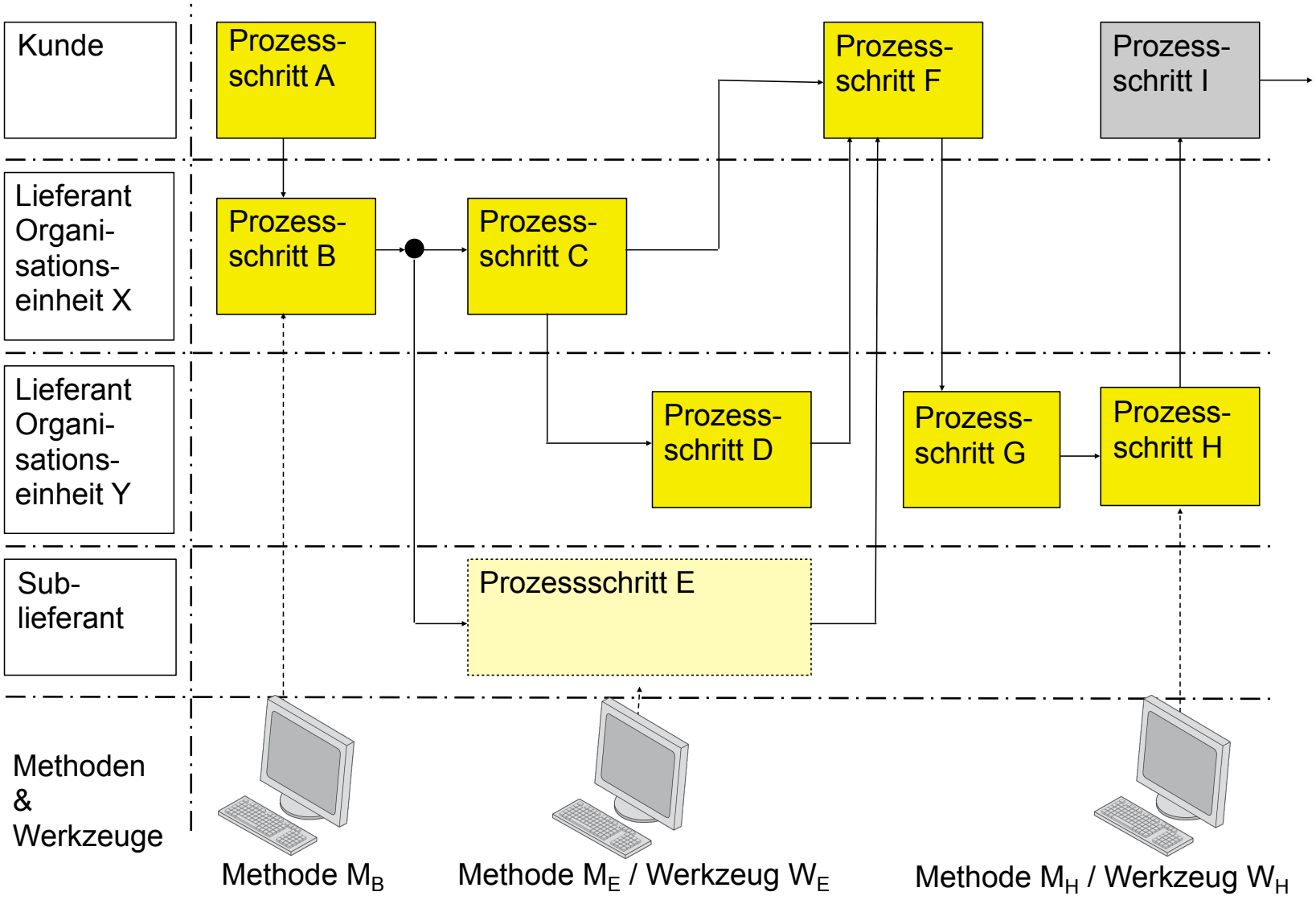
Prozessschritt F:
Vergleich der beiden Alternativen (C,D) und (E)



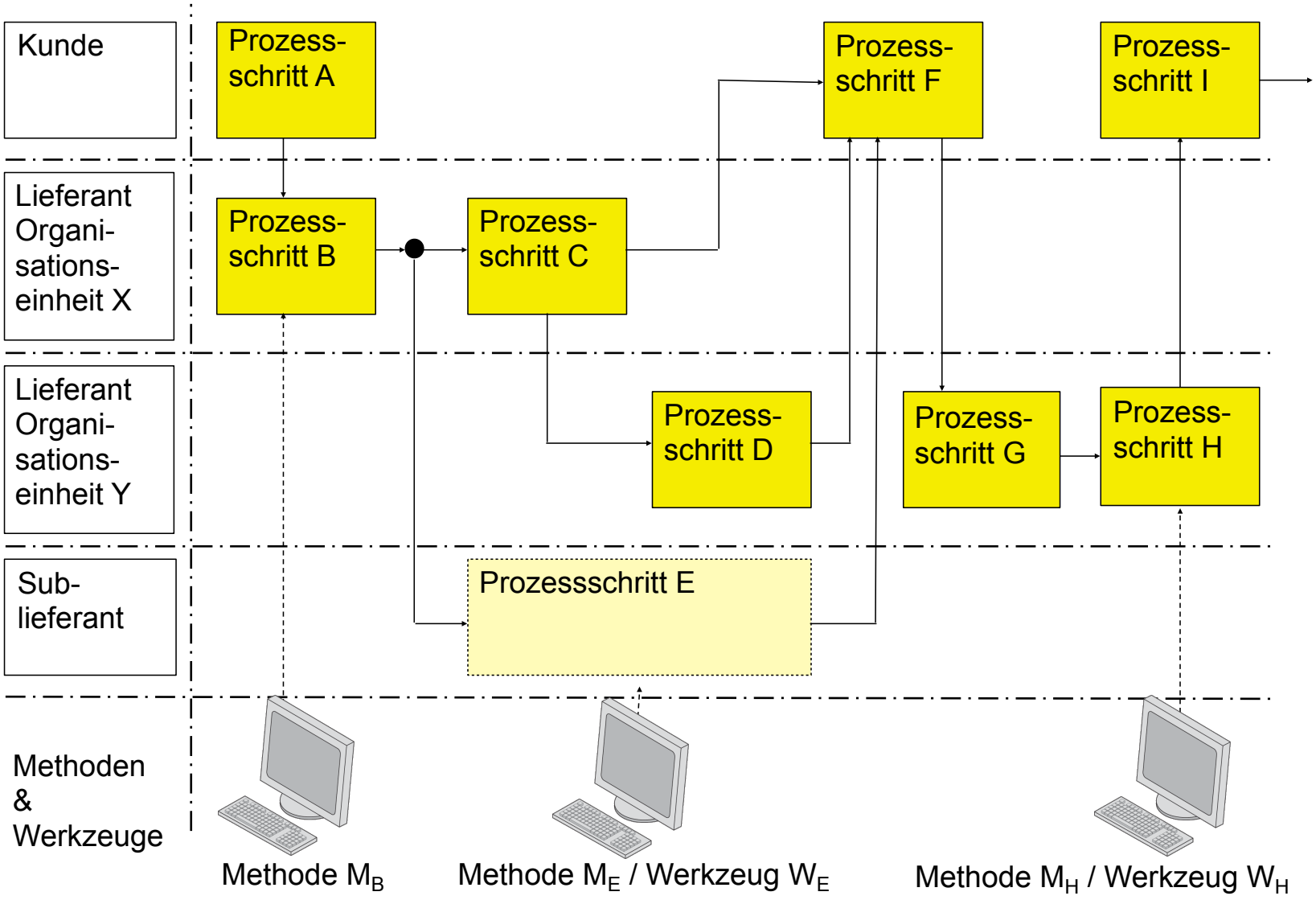
Prozessschritt G:
Weiterentwicklung der SW aus Prozessschritt D



Prozessschritt H:
Integration der SW aus Prozessschritt G auf Steuergerät aus Prozessschritt C



Prozessschritt I: Fahrzeugintegration



Unterstützungsprozesse für die Embedded Software Entwicklung

1. Vorgehensmodelle und Standards
2. Konfigurationsmanagement
3. Projektmanagement
4. Lieferantenmanagement
5. **Anforderungsmanagement**
   1. **Erfassen von Benutzeranforderungen**
   2. Verfolgen von Anforderungen
6. Qualitätssicherung

Anforderungsmanagement

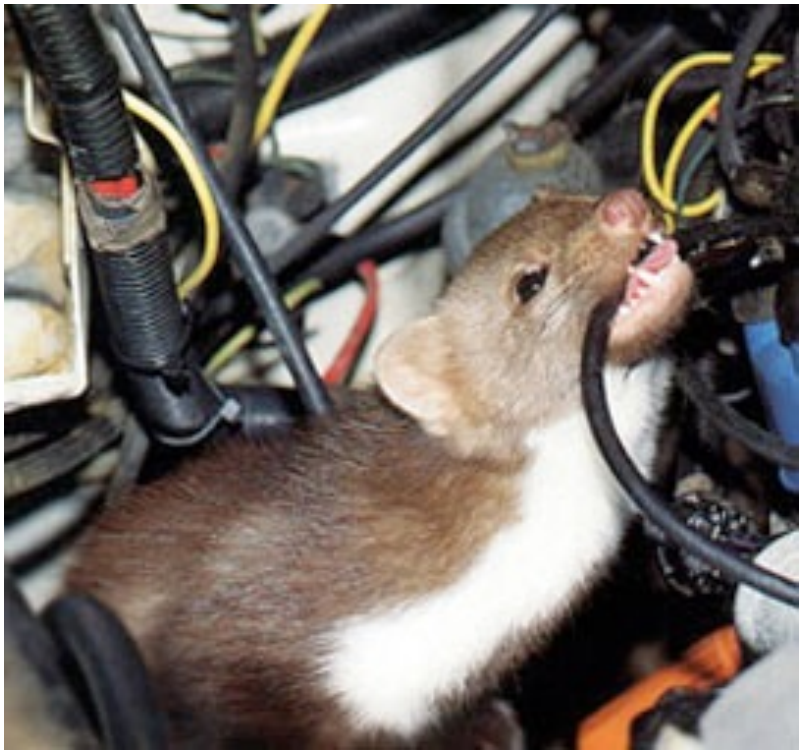
- Anforderungsmanagement ist - wie Konfigurationsmanagement - nicht grundsätzlich automobilspezifisch
  - Einsatz von Standardwerkzeugen wie Telelogic DOORS (Jetzt IBM)
  - http://www-01.ibm.com/software/awdtools/doors/
- Aber: Berücksichtigung von besonderen Anforderungen in Fahrzeugprojekten
  - Unterstützung der unternehmens- und standortübergreifenden Zusammenarbeit im Anforderungsmanagement
  - Versionierung von Anforderungen
    - Zusammenspiel von Anforderungsmanagement und Konfigurationsmanagement
  - Lange Produktlebenszyklen
    - Änderung der Anforderungen
- Unterstützungsprozess Anforderungsmanagement
  - Erfassen von Benutzeranforderungen
  - Verfolgen von Anforderungen
- Kernprozess der System- und Softwareentwicklung
  - Analyse der Anforderungen
  - Spezifikation der logischen und technischen Systemarchitektur

Erfassen von Benutzeranforderungen

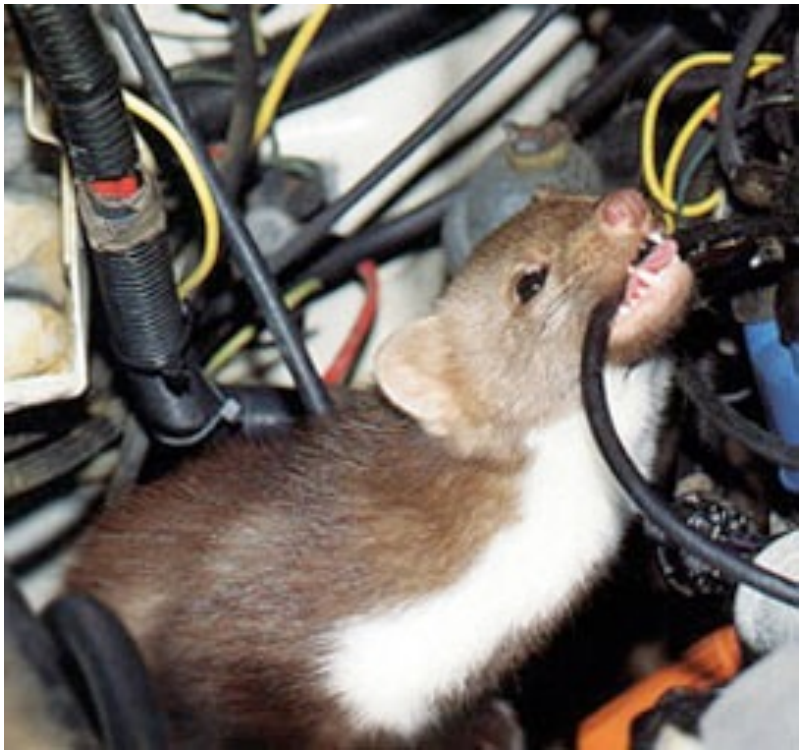
- Benutzer sind alle Personen, deren Wünsche Einfluss auf das Fahrzeug haben können
- Benutzergruppen eines Fahrzeugs
  - Fahrer
  - Insassen
  - Andere Verkehrsteilnehmer
    - Fahrzeuge
    - Fahrer
    - Radfahrer
    - Reiter
    - Fussgänger
  - ...
  - ...
  - ...
  - Mitarbeiter im Service
  - Gesetzgeber
    - Gesetzliche Vorgaben werden auch als Randbedingungen bezeichnet

Weitere Benutzergruppen („Stakeholder“)

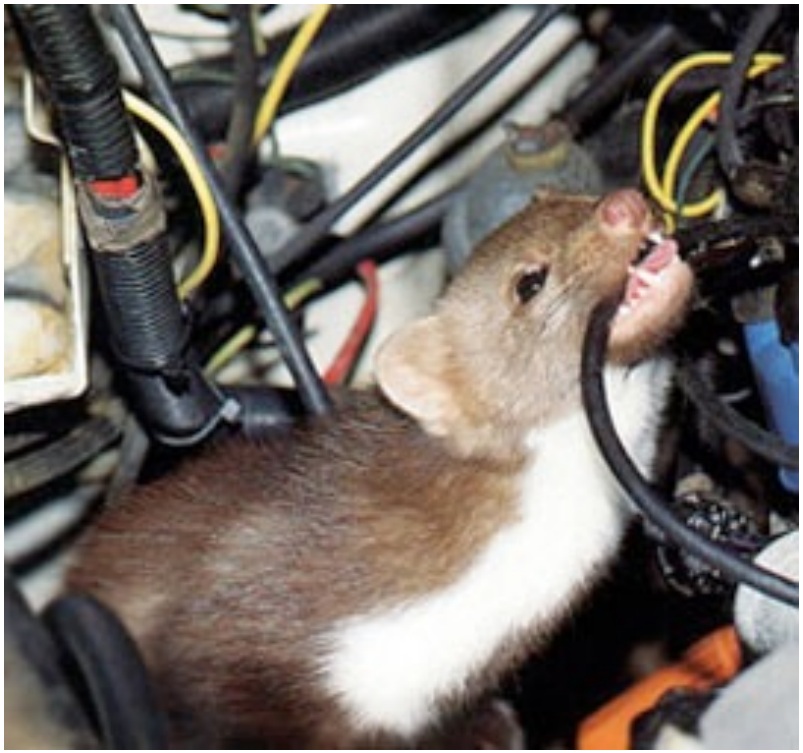
Weitere Benutzergruppen („Stakeholder“)



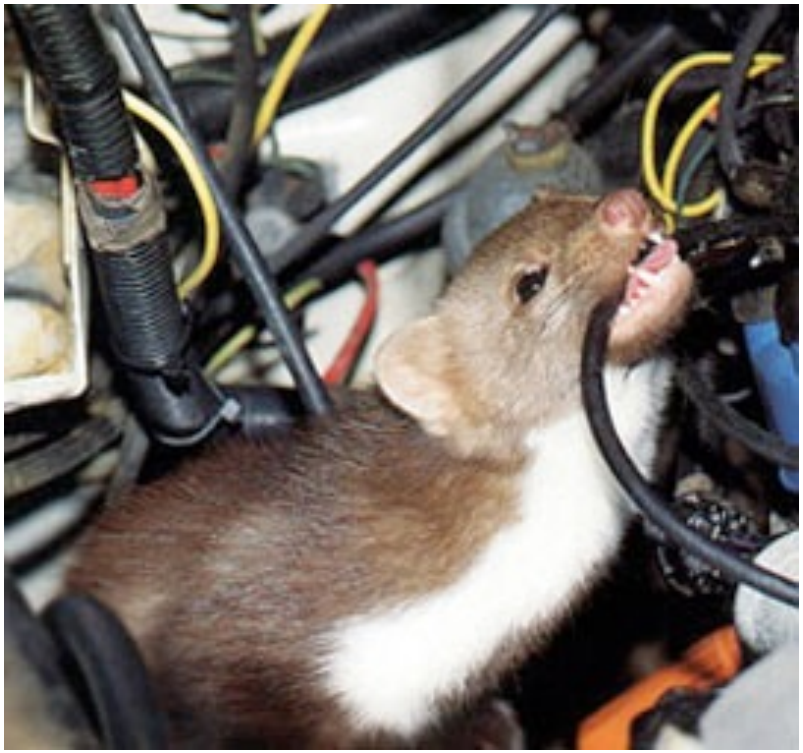
Weitere Benutzergruppen („Stakeholder“)



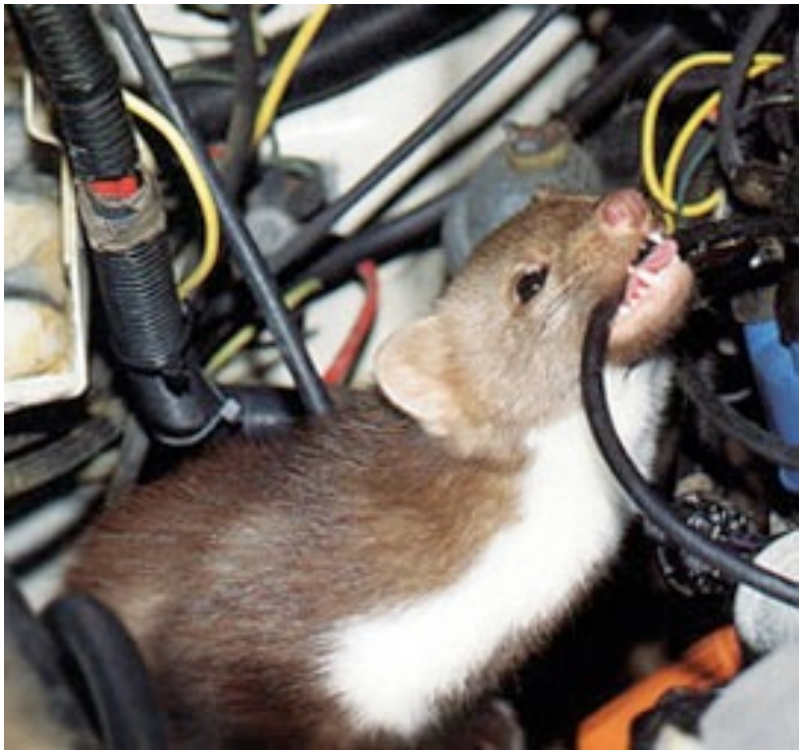
Weitere Benutzergruppen („Stakeholder“)



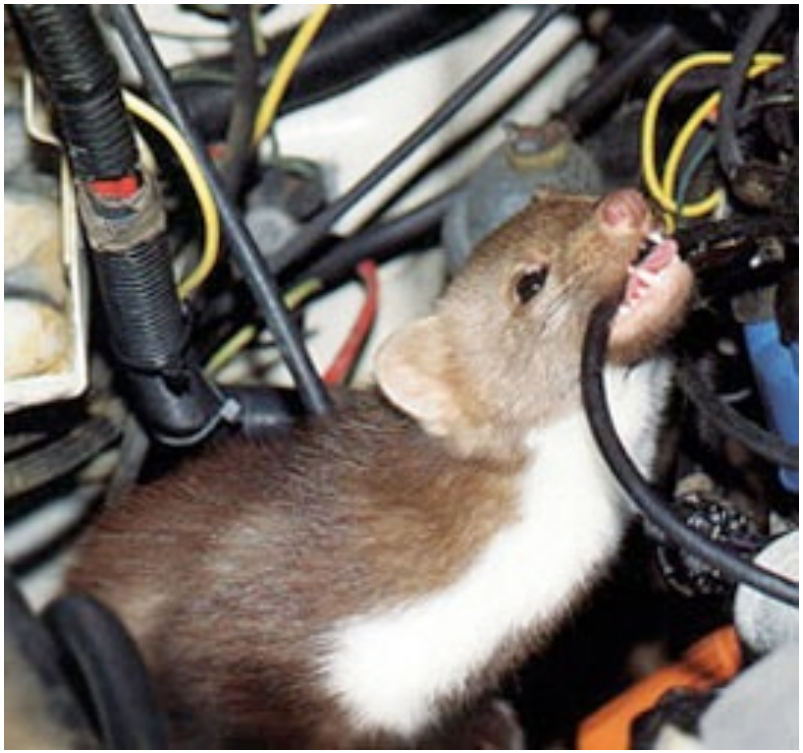
Weitere Benutzergruppen („Stakeholder“)



Weitere Benutzergruppen („Stakeholder“)



Weitere Benutzergruppen („Stakeholder“)



Erfassen von Benutzeranforderungen

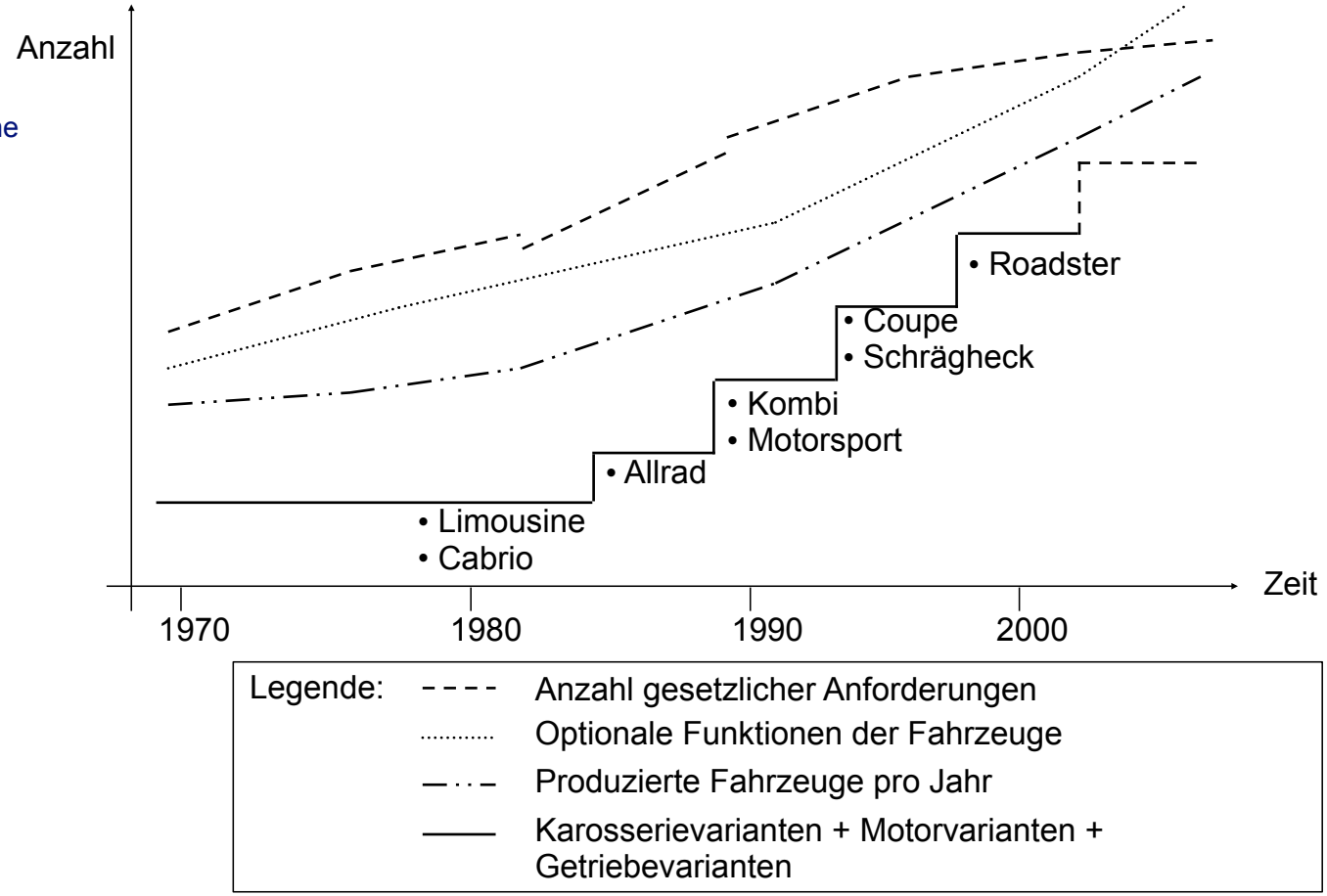
- Benutzer sind alle Personen, deren Wünsche Einfluss auf das Fahrzeug haben können
- Benutzergruppen eines Fahrzeugs
  - Fahrer
  - Insassen
  - Andere Verkehrsteilnehmer
    - Fahrzeuge
    - Fahrer
    - Radfahrer
    - Reiter
    - Fussgänger
  - Diebe, Vandalen
  - Marder
  - Insekten, Wild(unfälle)
  - Mitfahrende Haustiere
  - ...
  - Mitarbeiter im Service
  - Gesetzgeber
    - Gesetzliche Vorgaben werden auch als Randbedingungen bezeichnet

Drei Arten von Benutzeranforderungen

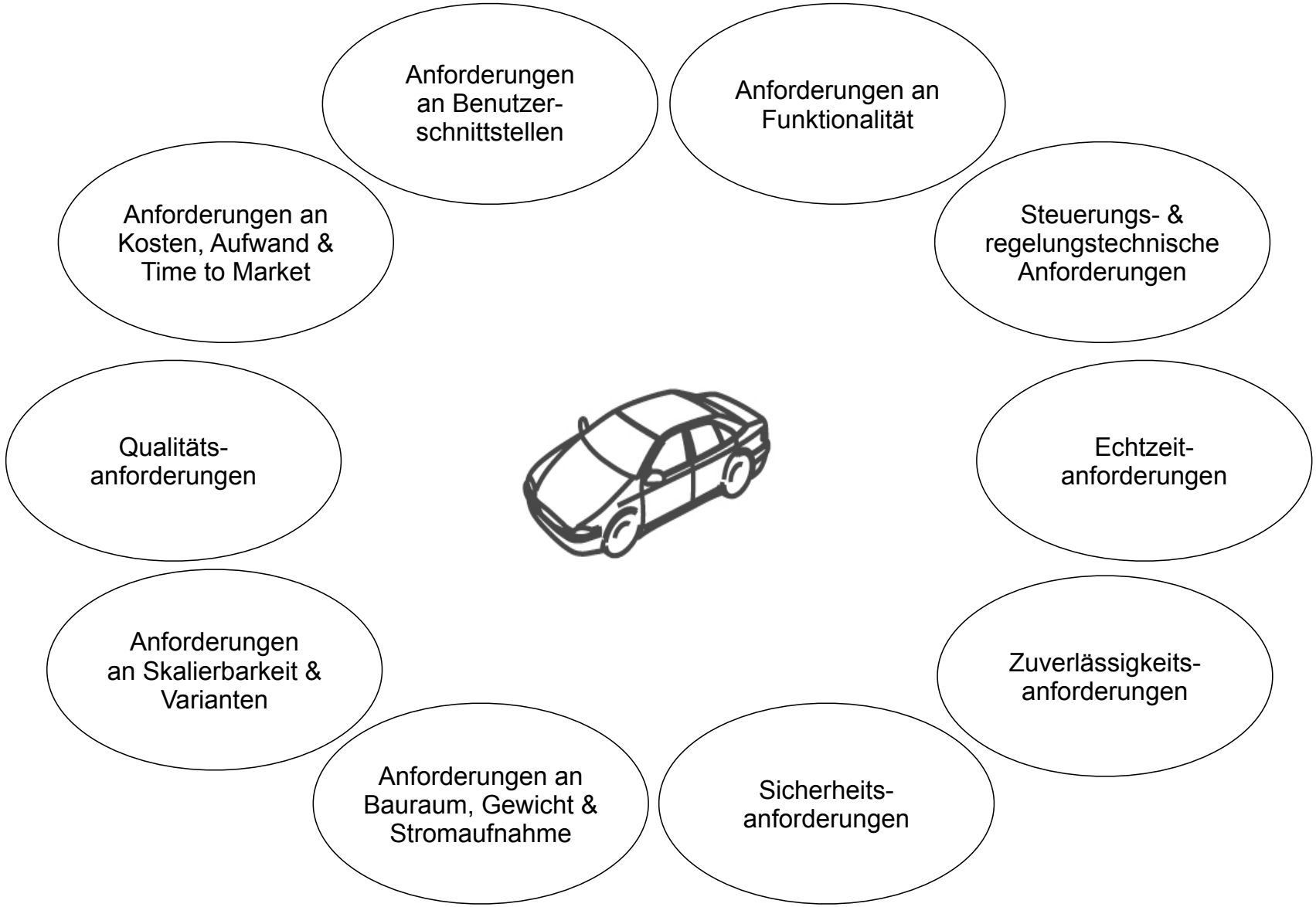
- Unterscheidung nach Zeitpunkt der Erfassung
  - Anforderungen, die zu Beginn des Projektes gestellt werden
  - Anforderungen, die im Laufe des Projektes gestellt werden
    - Änderungswünsche
    - Zusätzliche Anforderungen
  - Anforderungen, die nach Übergabe des Produktes gestellt werden
    - Neue Anforderungen
    - Fehlermeldungen
    - Verbesserungsvorschläge
- Erfassung von Benutzeranforderungen durch
  - Ableitung aus existierenden Systemen
  - Technische Entwicklungen
  - Rückmeldungen
  - Interviews
  - Workshops
  - ...

Anzahl der Benutzeranforderungen

- Zunahme mit jeder Fahrzeuggeneration
  - Gestiegene Anzahl der Fahrzeugfunktionen
  - Zunahme der Fahrzeugvarianten
  - Gestiegene Kundenerwartungen
  - Gestiegene Gesetzliche Anforderungen



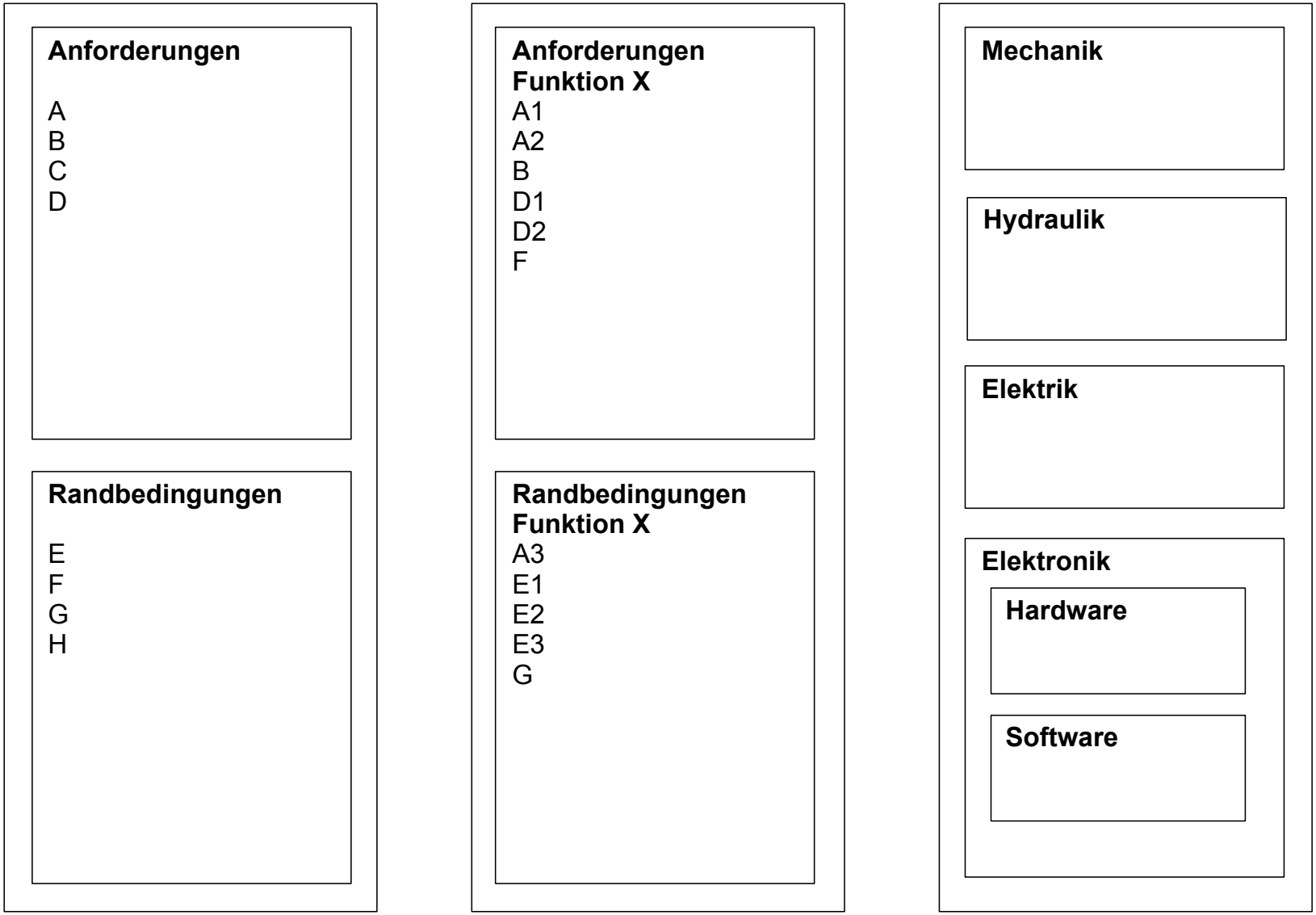
Verschiedene Anforderungsklassen bei elektreonischen Systemen im Automobil



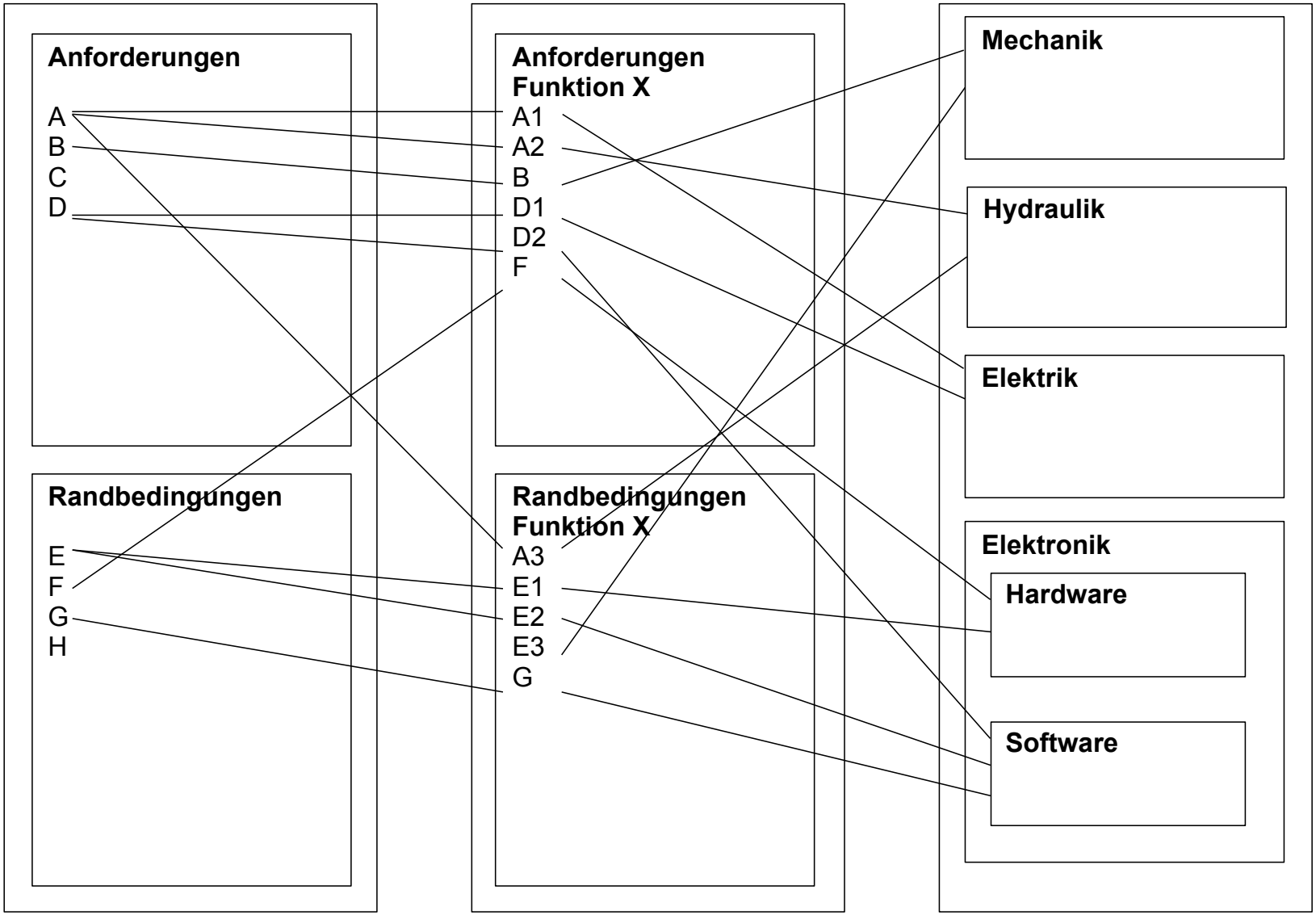
Unterstützungsprozesse für die Embedded Software Entwicklung

1. Vorgehensmodelle und Standards
2. Konfigurationsmanagement
3. Projektmanagement
4. Lieferantenmanagement
5. **Anforderungsmanagement**
    1. Erfassen von Benutzeranforderungen
    2. **Verfolgen von Anforderungen**
6. Qualitätssicherung

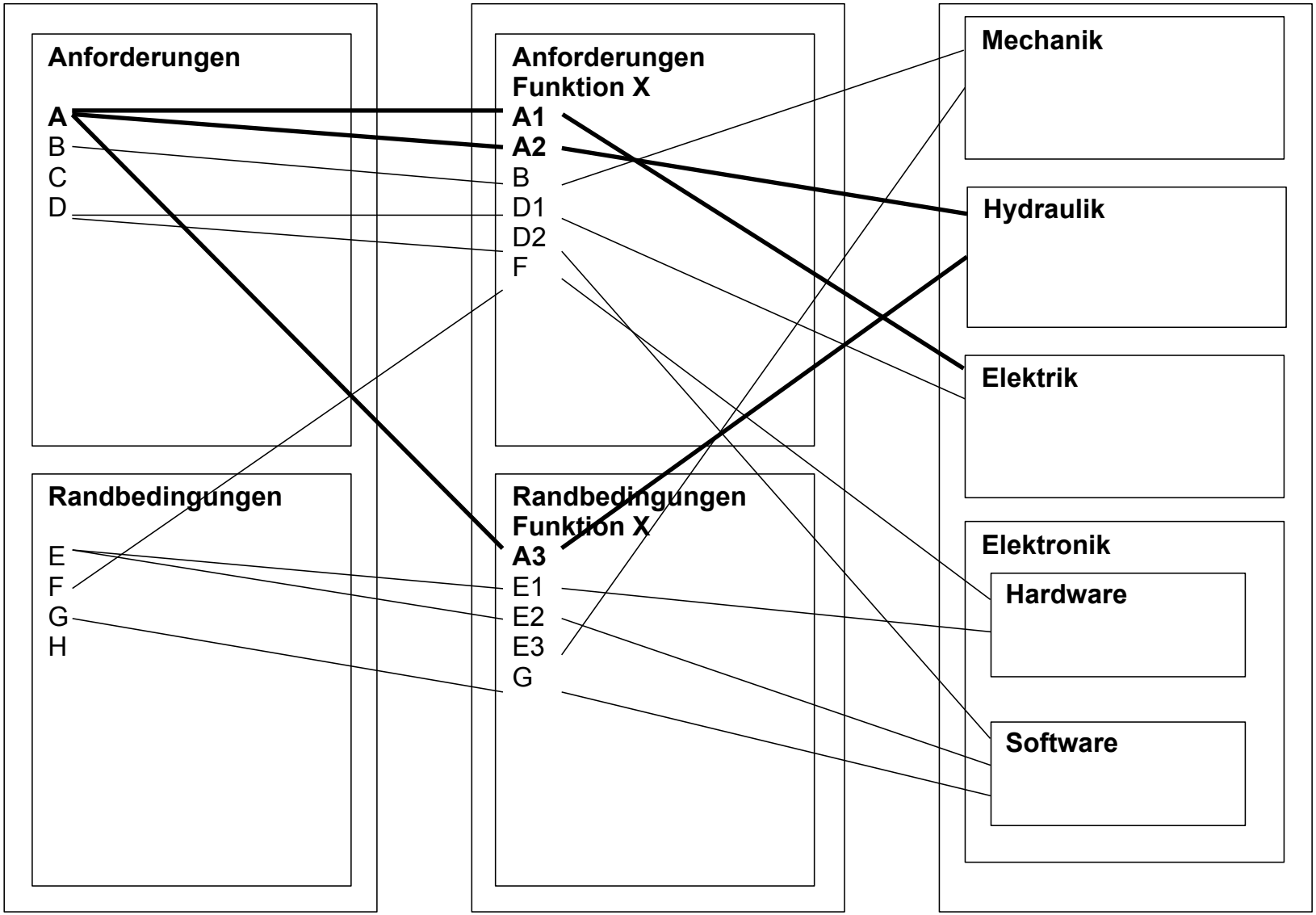
Benutzeranforderungen, logische und technische Systemarchitektur (Nach Schäuffele, Zurawka)



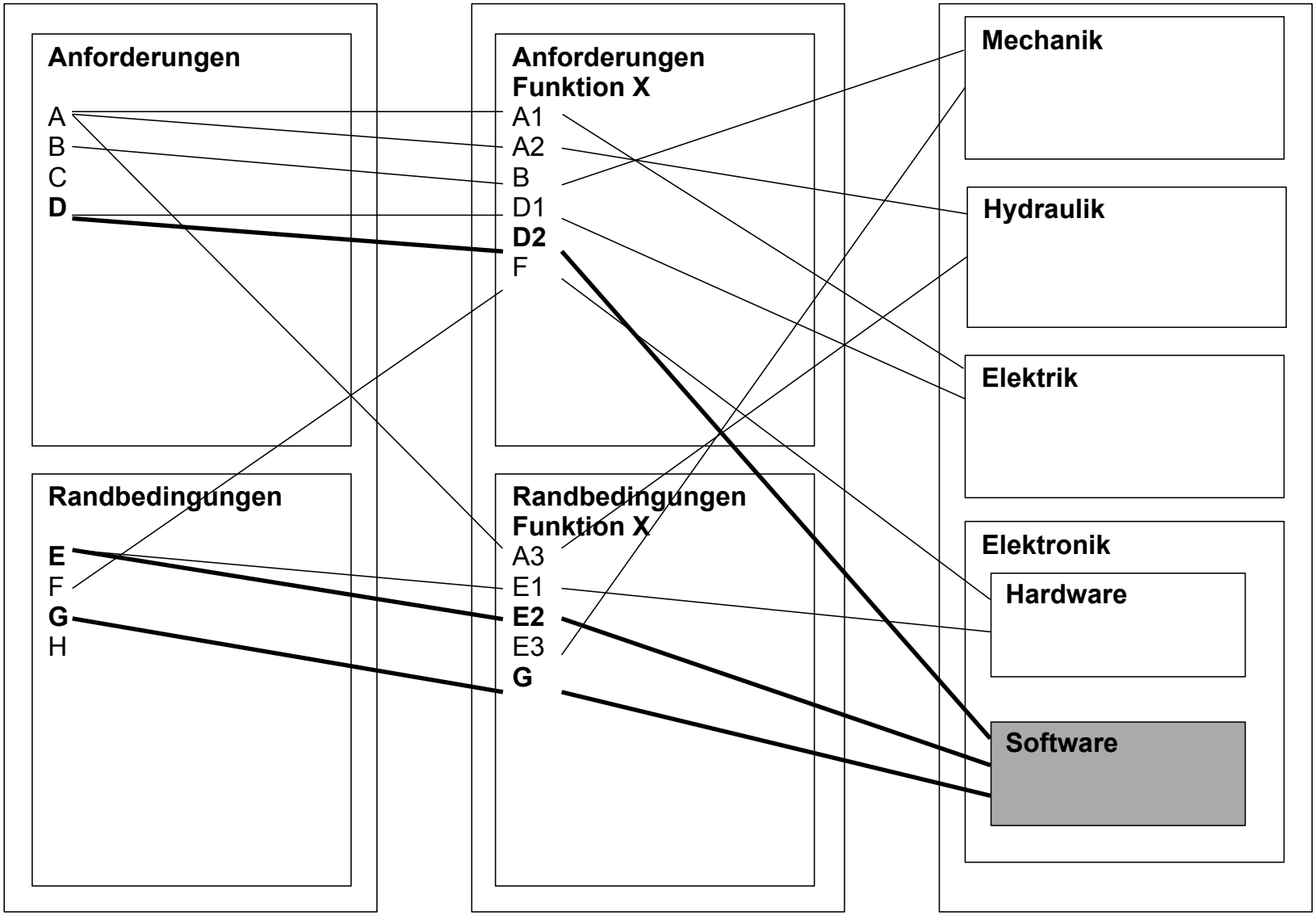
Verfolgen von Anforderungen:
Welche Anforderungen werden wie umgesetzt?



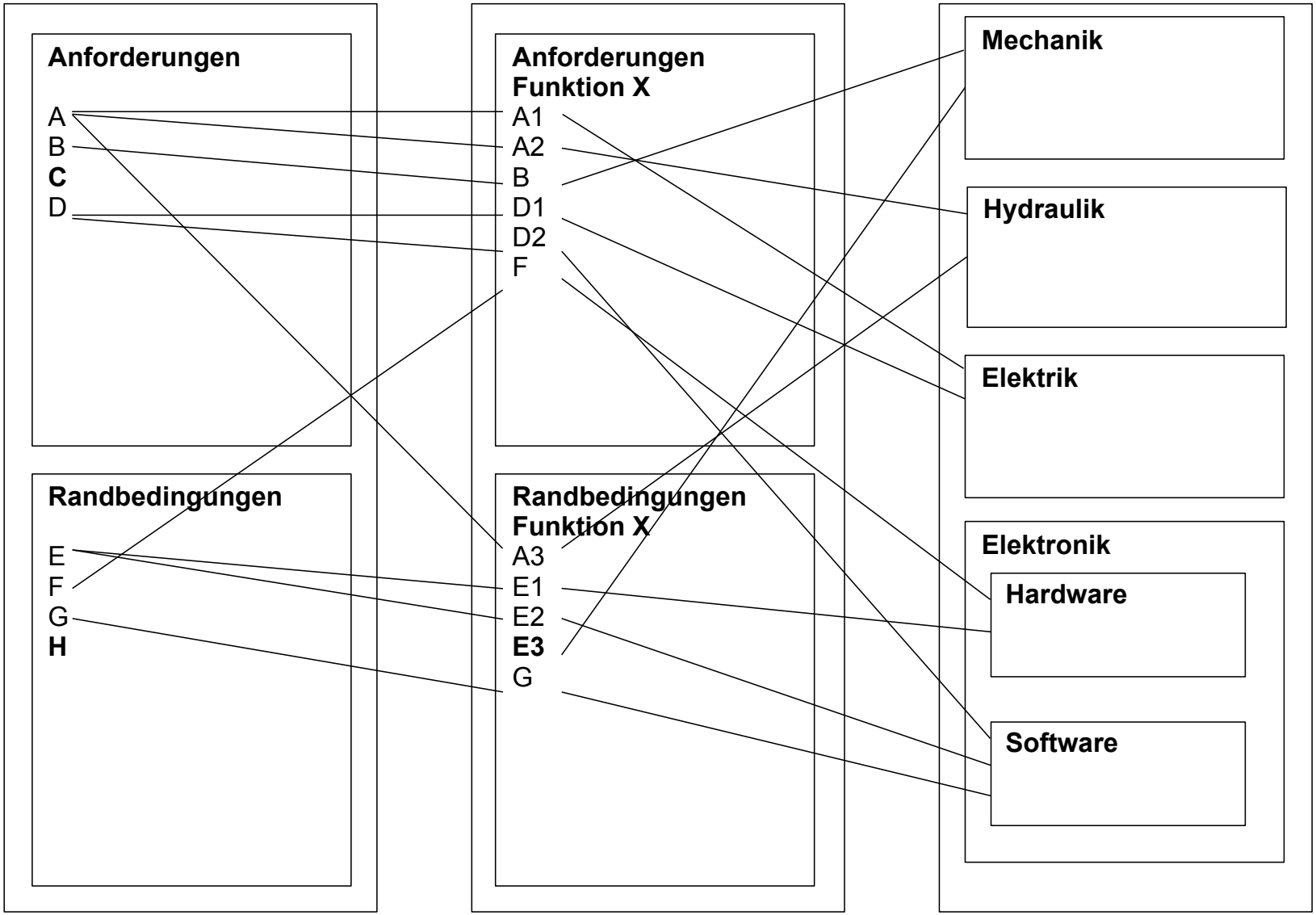
Verfolgen von Anforderungen:
Welche Anforderungen werden wie umgesetzt?



Verfolgen von Anforderungen:
Welche Anforderungen werden wie umgesetzt?



Verfolgen von Anforderungen:
Welche Anforderungen werden wie umgesetzt?



Unterstützungsprozesse für die Embedded Software Entwicklung

1. Vorgehensmodelle und Standards
2. Konfigurationsmanagement
3. Projektmanagement
4. Lieferantenmanagement
5. Anforderungsmanagement
6. **Qualitätssicherung**
    1. **Integrations- und Testschritte**
    2. Maßnahmen zur Qualitätssicherung von Software

Qualitätssicherung

- Qualitätssicherung
  - Alle Maßnahmen, die sicherstellen, dass das Produkt die geforderten Anforderungen erfüllt.
- Richtlinien zur Qualitätssicherung: Vorbeugende Maßnahmen
  - Einsatz von erfahrenen und geschulten Mitarbeitern
  - Geeigneter Entwicklungsprozess mit definierten Testschritten
  - Richtlinien, Maßnahmen und Standards zur Unterstützung des Prozesses
  - Werkzeugumgebung zur Unterstützung des Prozesses
  - Automatisierung manueller und fehlerträchtiger Arbeitsschritte
- Maßnahmen zur Qualitätssicherung: Finden von Fehlern
  - Nach jedem Schritt des Entwicklungsprozesses
  - Ins V-Modell integriert, siehe „Kernprozess“
- Software-Produkte
  - Spezifikationsfehler
  - Implementierungsfehler
  - Ergebnis von zahlreichen Untersuchungen:
    - In den meisten Projekten überwiegen die Spezifikationsfehler

Validation und Verifikation

- Validation: Prüfen auf Spezifikationsfehler
- Validation ist der Prozess zur Beurteilung eines Systems oder einer Komponente mit dem Ziel festzustellen, ob der Einsatzzweck oder die Benutzererwartungen erfüllt werden. Funktionsvalidation ist demnach die Prüfung, ob die Spezifikation die Benutzeranforderungen erfüllt, ob überhaupt die Benutzerakzeptanz durch eine Funktion erreicht wird.
- Verifikation: Prüfen auf Implementierungsfehler
- Verifikation ist der Prozess zur Beurteilung eines Systems oder einer Komponente mit dem Ziel festzustellen, ob die Resultate einer gegebenen Entwicklungsphase den Vorgaben für diese Phase entsprechen. Software-Verifikation ist demnach die Prüfung, ob eine Implementierung der für den betreffenden Entwicklungsschritt vorgegebenen Spezifikation genügt.

Verifikation und Validation

- Verifikation:
  - „Does the system things right?“ / „Erfüllt das System seine Aufgabe richtig?“
  - Prüfung z.B. gegen Technische Anforderungen
  - Beispiel: Analyse des Zeitverhaltens durch Untersuchung der Worst Case Execution Time (WCET)
- Validation:
  - „Does the system the right things?“ / „Erfüllt das System die richtige Aufgabe?“
  - Prüfung z.B. gegen Benutzeranforderungen
  - Beispiel: Simulation der Benutzeroberfläche
- Ähnlich:
  - Effektiv: Die richtigen Dinge tun.
  - Effizient: Die Dinge richtig tun.

Integrations- und Testschritte

- Das V-Modell unterscheidet vier verschiedene Testschritte:
  - Beim Komponententest wird eine Komponente gegen die Spezifikation der Komponente getestet.
  - Beim Integrationstest wird das System gegen die Spezifikation der technischen Systemarchitektur getestet.
  - Beim Systemtest wird das System gegen die Spezifikation der logischen Systemarchitektur getestet.
  - Beim Akzeptanztest wird das System gegen die Benutzeranforderungen getestet.
- Validation und Verifikation

Integrations- und Testschritte

- Das V-Modell unterscheidet vier verschiedene Testschritte:
  - Beim Komponententest wird eine Komponente gegen die Spezifikation der Komponente getestet.
  - Beim Integrationstest wird das System gegen die Spezifikation der technischen Systemarchitektur getestet.
  - Beim Systemtest wird das System gegen die Spezifikation der logischen Systemarchitektur getestet.
  - Beim Akzeptanztest wird das System gegen die Benutzeranforderungen getestet.
- Validation und Verifikation

Integrations- und Testschritte

- Das V-Modell unterscheidet vier verschiedene Testschritte:
  - Beim Komponententest wird eine Komponente gegen die Spezifikation der Komponente getestet.
  - Beim Integrationstest wird das System gegen die Spezifikation der technischen Systemarchitektur getestet.
  - Beim Systemtest wird das System gegen die Spezifikation der logischen Systemarchitektur getestet.
  - Beim Akzeptanztest wird das System gegen die Benutzeranforderungen getestet.
- Validation und Verifikation

Integrations- und Testschritte

- Das V-Modell unterscheidet vier verschiedene Testschritte:
  - Beim Komponententest wird eine Komponente gegen die Spezifikation der Komponente getestet.
  - Beim Integrationstest wird das System gegen die Spezifikation der technischen Systemarchitektur getestet.
  - Beim Systemtest wird das System gegen die Spezifikation der logischen Systemarchitektur getestet.
  - Beim Akzeptanztest wird das System gegen die Benutzeranforderungen getestet.
- Validation und Verifikation

Integrations- und Testschritte

- Das V-Modell unterscheidet vier verschiedene Testschritte:
  - Beim Komponententest wird eine Komponente gegen die Spezifikation der Komponente getestet.
  - Beim Integrationstest wird das System gegen die Spezifikation der technischen Systemarchitektur getestet.
  - Beim Systemtest wird das System gegen die Spezifikation der logischen Systemarchitektur getestet.
  - Beim Akzeptanztest wird das System gegen die Benutzeranforderungen getestet.
- Validation und Verifikation

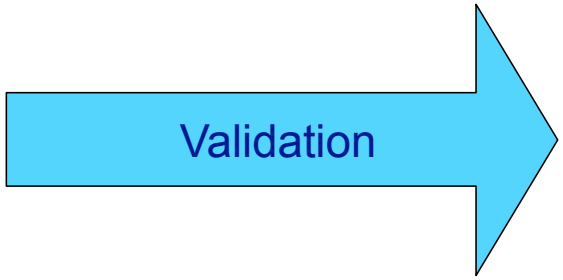
Integrations- und Testschritte

- Das V-Modell unterscheidet vier verschiedene Testschritte:
  - Beim Komponententest wird eine Komponente gegen die Spezifikation der Komponente getestet.
  - Beim Integrationstest wird das System gegen die Spezifikation der technischen Systemarchitektur getestet.
  - Beim Systemtest wird das System gegen die Spezifikation der logischen Systemarchitektur getestet.
  - Beim Akzeptanztest wird das System gegen die Benutzeranforderungen getestet.
- Validation und Verifikation

Integrations- und Testschritte

- Das V-Modell unterscheidet vier verschiedene Testschritte:
  - Beim Komponententest wird eine Komponente gegen die Spezifikation der Komponente getestet.
  - Beim Integrationstest wird das System gegen die Spezifikation der technischen Systemarchitektur getestet.
  - Beim Systemtest wird das System gegen die Spezifikation der logischen Systemarchitektur getestet.
  - Beim Akzeptanztest wird das System gegen die Benutzeranforderungen getestet.
- Validation und Verifikation

Integrations- und Testschritte



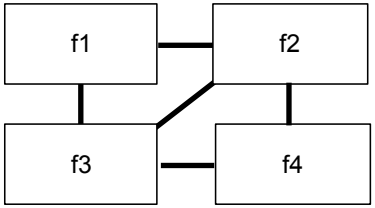
- Das V-Modell unterscheidet vier verschiedene Testschritte:
  - Beim Komponententest wird eine Komponente gegen die Spezifikation der Komponente getestet.
  - Beim Integrationstest wird das System gegen die Spezifikation der technischen Systemarchitektur getestet.
  - Beim Systemtest wird das System gegen die Spezifikation der logischen Systemarchitektur getestet.
  - Beim Akzeptanztest wird das System gegen die Benutzeranforderungen getestet.
- Validation und Verifikation

Integrations- und Testschritte

- Das V-Modell unterscheidet vier verschiedene Testschritte:
  - Beim Komponententest wird eine Komponente gegen die Spezifikation der Komponente getestet.
  - Beim Integrationstest wird das System gegen die Spezifikation der technischen Systemarchitektur getestet.
  - Beim Systemtest wird das System gegen die Spezifikation der logischen Systemarchitektur getestet.
  - Beim Akzeptanztest wird das System gegen die Benutzeranforderungen getestet.
- Validation und Verifikation

Integrations- und Testschritte (Nach Schäuffele, Zurawka)

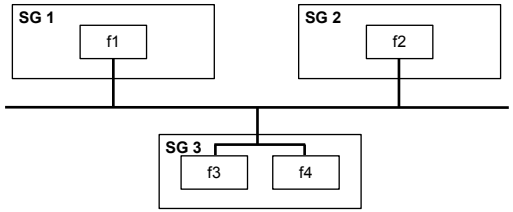
Analyse der Benutzeranforderungen und Spezifikation der logischen Systemarchitektur



Logische Systemarchitektur

Akzeptanz- und Systemtest

Analyse der logischen Systemarchitektur und Spezifikation der technischen Systemarchitektur



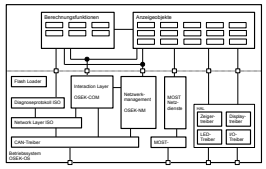
Technische Systemarchitektur

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

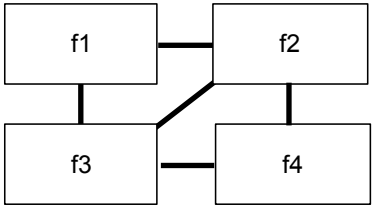
Spezifikation der Software-Komponenten

Design und Implementierung der Software-Komponenten

Test der Software-Komponenten

Integrations- und Testschritte (Nach Schäuffele, Zurawka)

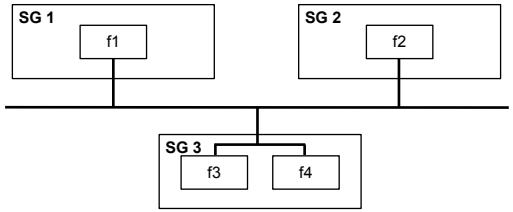
Analyse der Benutzeranforderungen und Spezifikation der logischen Systemarchitektur



Logische Systemarchitektur

Akzeptanz- und Systemtest

Analyse der logischen Systemarchitektur und Spezifikation der technischen Systemarchitektur



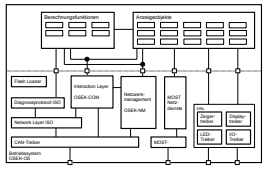
Technische Systemarchitektur

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

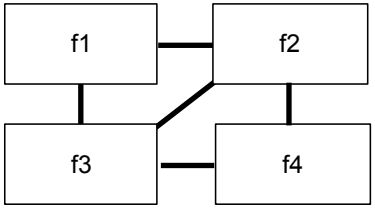
Spezifikation der Software-Komponenten

Design und Implementierung der Software-Komponenten

Test der Software-Komponenten

Integrations- und Testschritte (Nach Schäuffele, Zurawka)

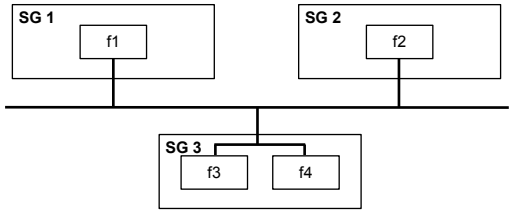
Analyse der Benutzeranforderungen und Spezifikation der logischen Systemarchitektur



Logische Systemarchitektur

Akzeptanz- und Systemtest

Analyse der logischen Systemarchitektur und Spezifikation der technischen Systemarchitektur



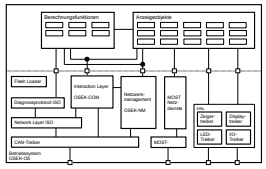
Technische Systemarchitektur

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

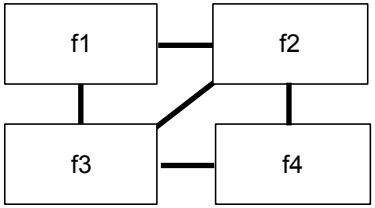
Spezifikation der Software-Komponenten

Design und Implementierung der Software-Komponenten

Test der Software-Komponenten

Integrations- und Testschritte (Nach Schäuffele, Zurawka)

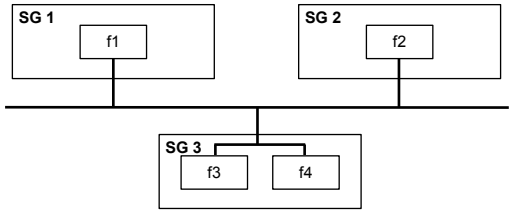
Analyse der Benutzeranforderungen und Spezifikation der logischen Systemarchitektur



Logische Systemarchitektur

Akzeptanz- und Systemtest

Analyse der logischen Systemarchitektur und Spezifikation der technischen Systemarchitektur



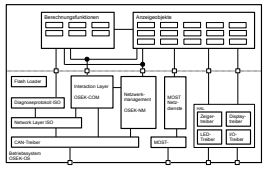
Technische Systemarchitektur

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

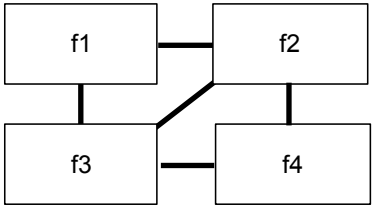
Spezifikation der Software-Komponenten

Design und Implementierung der Software-Komponenten

Test der Software-Komponenten

Integrations- und Testschritte (Nach Schäuffele, Zurawka)

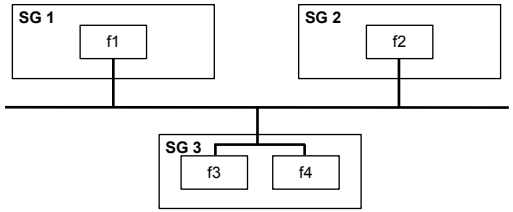
Analyse der Benutzeranforderungen und Spezifikation der logischen Systemarchitektur



Logische Systemarchitektur

Akzeptanz- und Systemtest

Analyse der logischen Systemarchitektur und Spezifikation der technischen Systemarchitektur



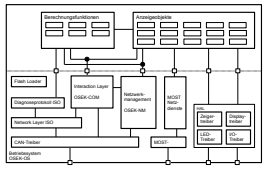
Technische Systemarchitektur

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

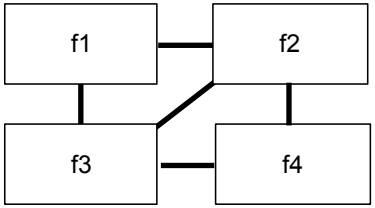
Spezifikation der Software-Komponenten

Design und Implementierung der Software-Komponenten

Test der Software-Komponenten

Integrations- und Testschritte (Nach Schäuffele, Zurawka)

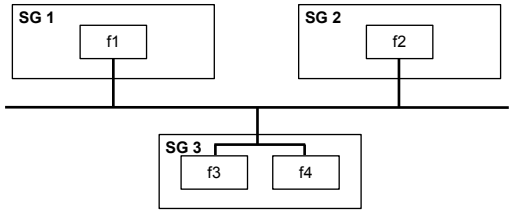
Analyse der Benutzeranforderungen und Spezifikation der logischen Systemarchitektur



Logische Systemarchitektur

Akzeptanz- und Systemtest

Analyse der logischen Systemarchitektur und Spezifikation der technischen Systemarchitektur



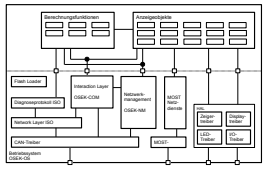
Technische Systemarchitektur

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

Unterstützungsprozesse für die Embedded Software Entwicklung

1. Vorgehensmodelle und Standards
2. Konfigurationsmanagement
3. Projektmanagement
4. Lieferantenmanagement
5. Anforderungsmanagement
6. **Qualitätssicherung**
   1. Integrations- und Testschritte
   2. **Maßnahmen zur Qualitätssicherung von Software**

Motivation

Tests können die Anwesenheit von Fehlern zeigen, nie aber deren Abwesenheit.

Edsger W. Dijkstra

Ein fehlender Programmzweig ... wird auch durch Austesten aller vorhandenen Programmzweige nicht gefunden.

Bernhard Hohlfeld

Software wird hauptsächlich getestet

- Testverfahren können nie alle Systemzustände erreichen
  - Wurde ausreichend getestet?
- Software muss für Tests vorhanden und ausführbar sein
  - Ist die Spezifikation korrekt?

Interne Qualitätseigenschaften werden selten geprüft

- Ist die Software zuverlässig, wartbar, änderbar etc. ?
- Ist die Software effizient?
- Wie viel Speicher/Zeit verbraucht das System maximal?

Übersicht über Maßnahmen zur Qualitätssicherung von Software

Verifikation

Statische Techniken

Review

Walkthrough, Fagan-Inspektion, Code-Inspektion, Peer-Review, ...

Analyse

Statische Analyse, Formale Prüfung, Kontroll- und Datenfluss, ...

Dynamischer Test
Komponenten-/Integrationstest

Black-Box-Test
Funktionale Leistungsfähigkeit, Stress, Grenzwert, Fehlererwartung, ...

White-Box-Test
Struktur, Pfad, Zweig, Bedingung, Abdeckung, ...

Validation

Animation

Formale Spezifikation
Modellierung
Simulation
Rapid Prototyping, ...

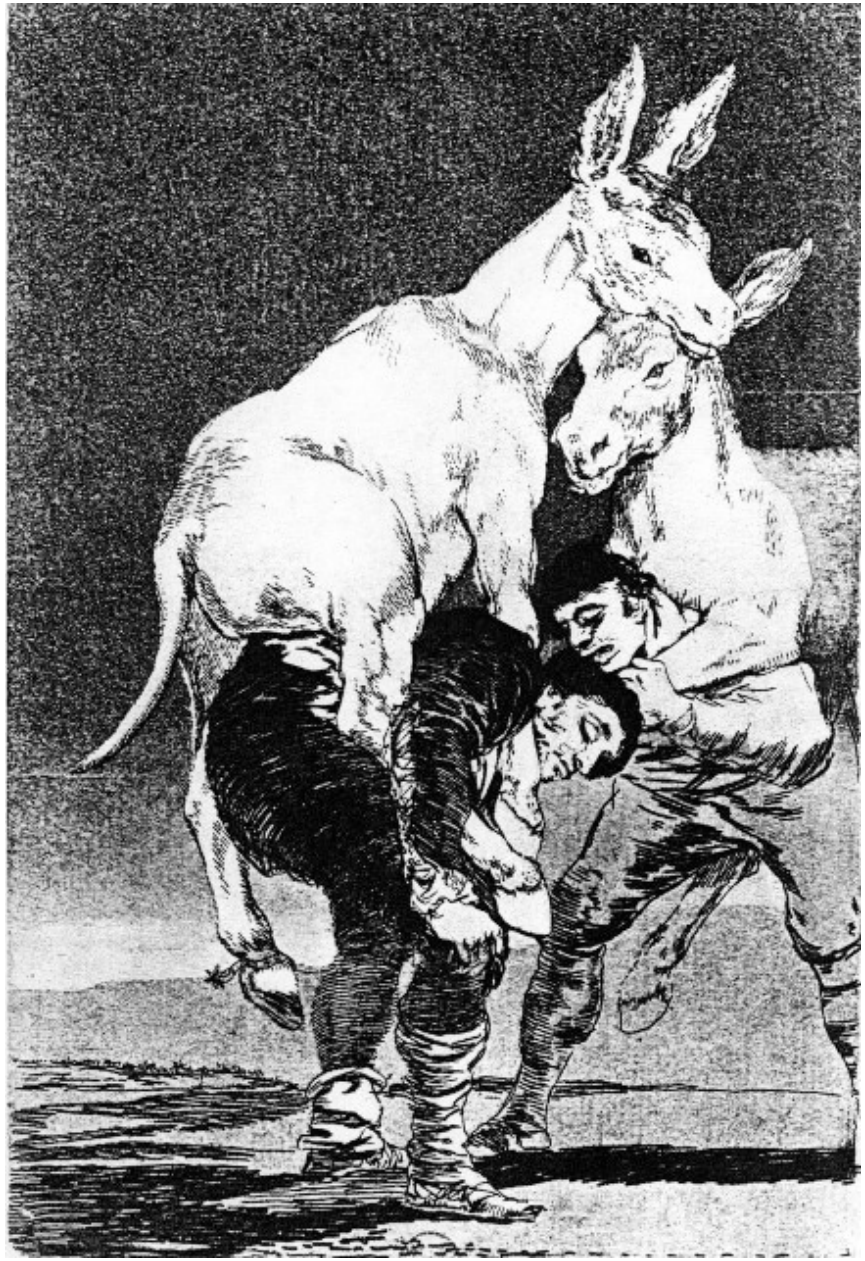
Systemtest/Akzeptanztest

Funktionale Leistungsfähigkeit
Stresstests, Grenzwerttests, Fehlererwartungstests, Ursache-Wirkungs-Graph, Äquivalenz-klassentests, ...

siehe auch http://campar.in.tum.de/twiki/pub/Chair/TeachingWs04MedInform/Softwaresicherheit.pdf

Unterstützungsprozesse aus Sicht des Entwicklers (1)

Unterstützungsprozesse aus Sicht des Entwicklers (2)



Entwickler aus Sicht der Unterstützungsprozesse (1)

Entwickler aus Sicht der Unterstützungsprozesse (2)