INFORMATIK ▪ CONSULTING ▪ SYSTEMS AG

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

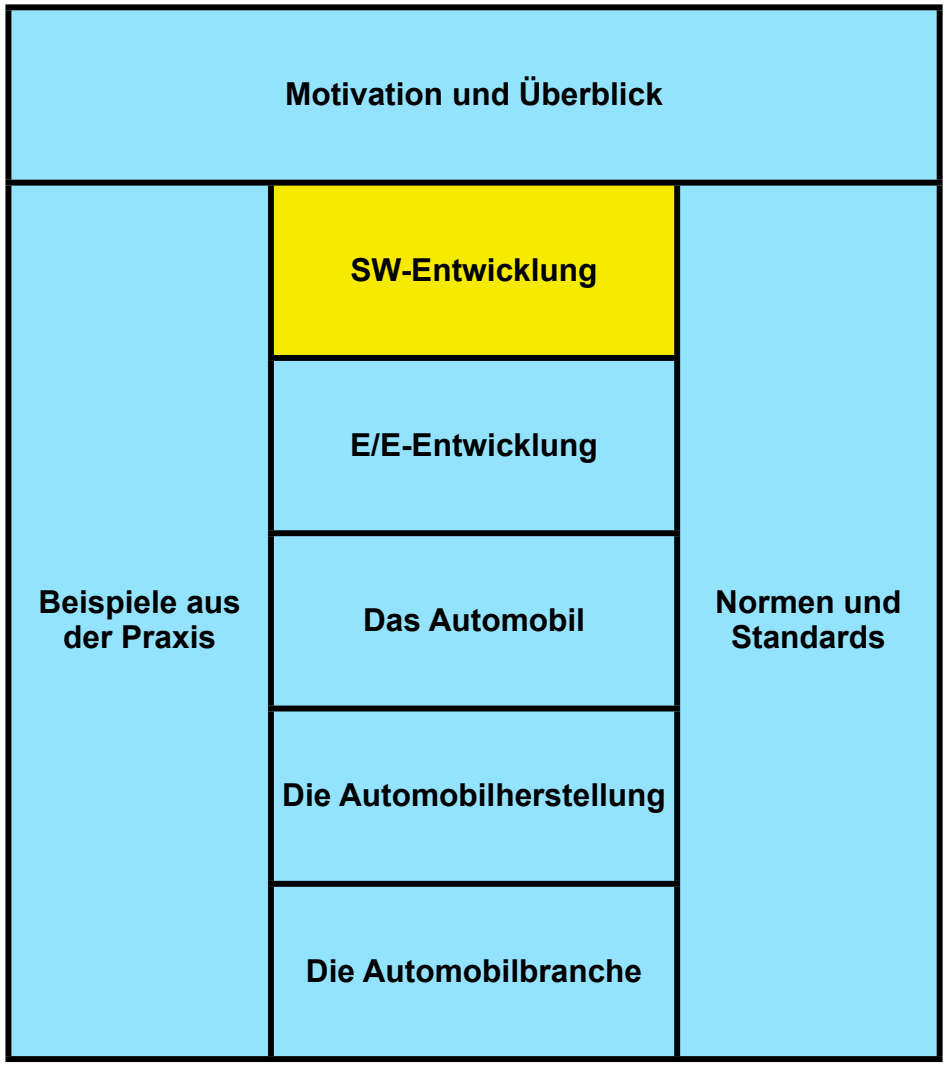
TU Dresden, Fakultät Informatik

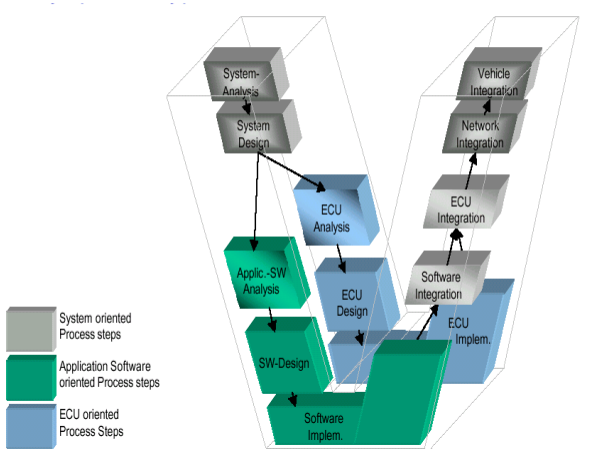
Sommersemester 2012

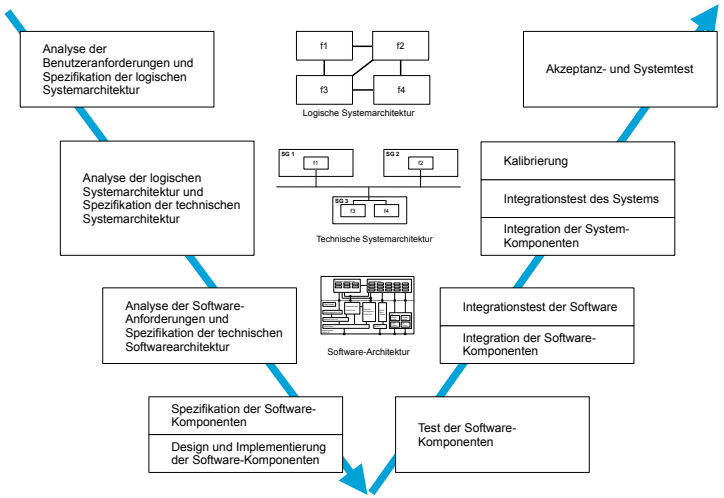
Prof. Dr. rer. nat. Bernhard Hohlfeld

bernhard.hohlfeld@daad-alumni.de

Vorlesung Automotive Software Engineering



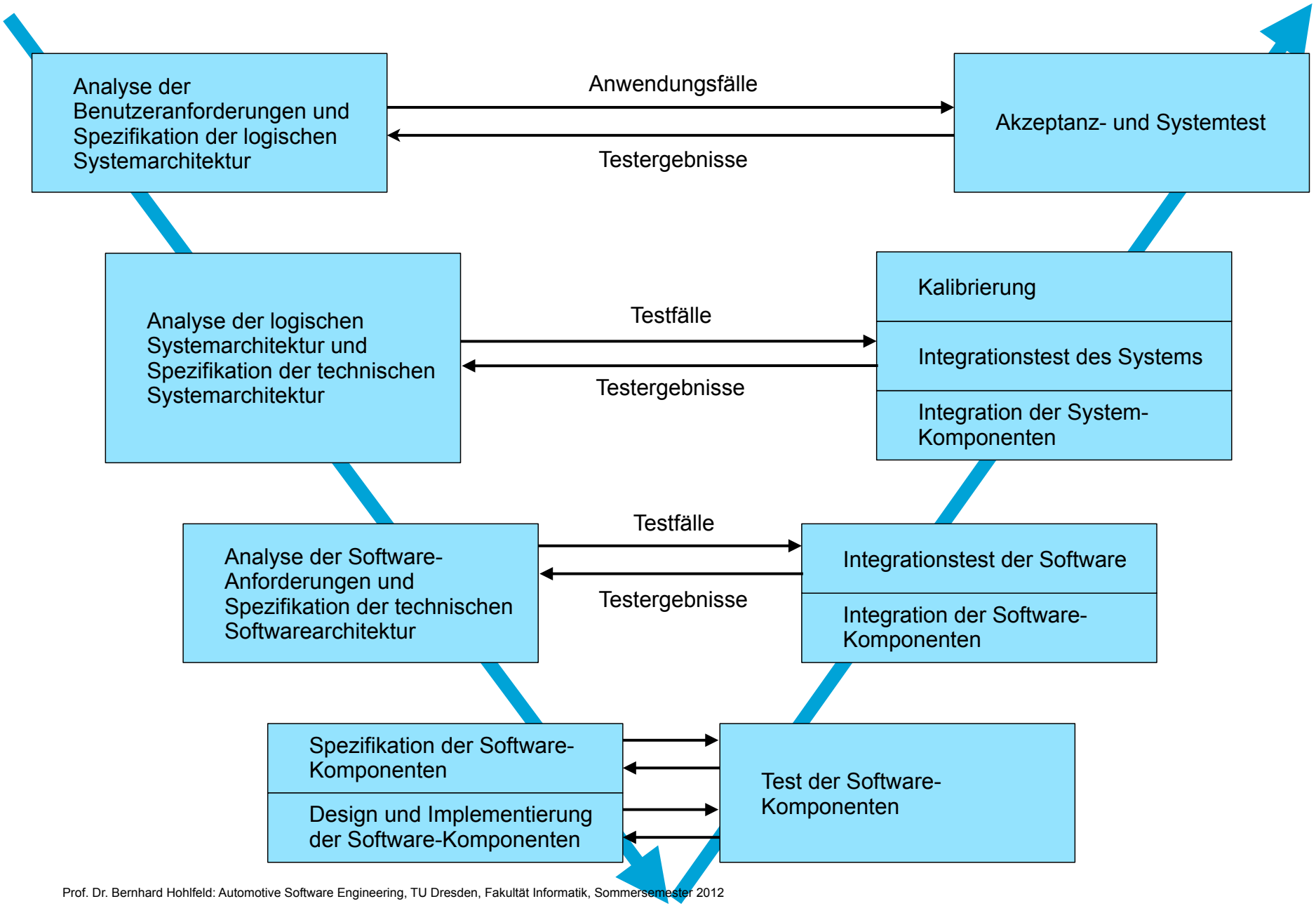




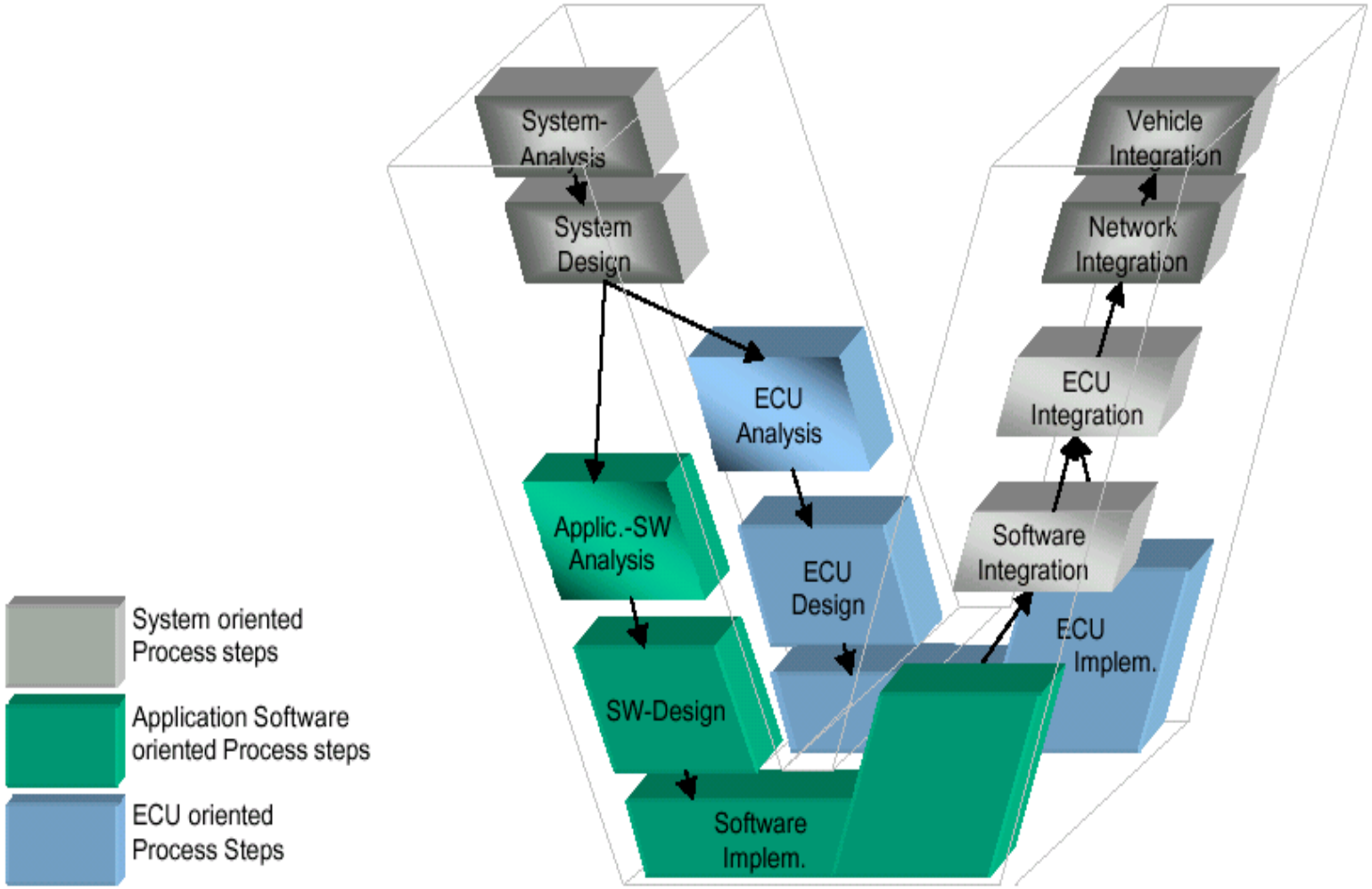
Lernziele
SW-Entwicklung

- 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

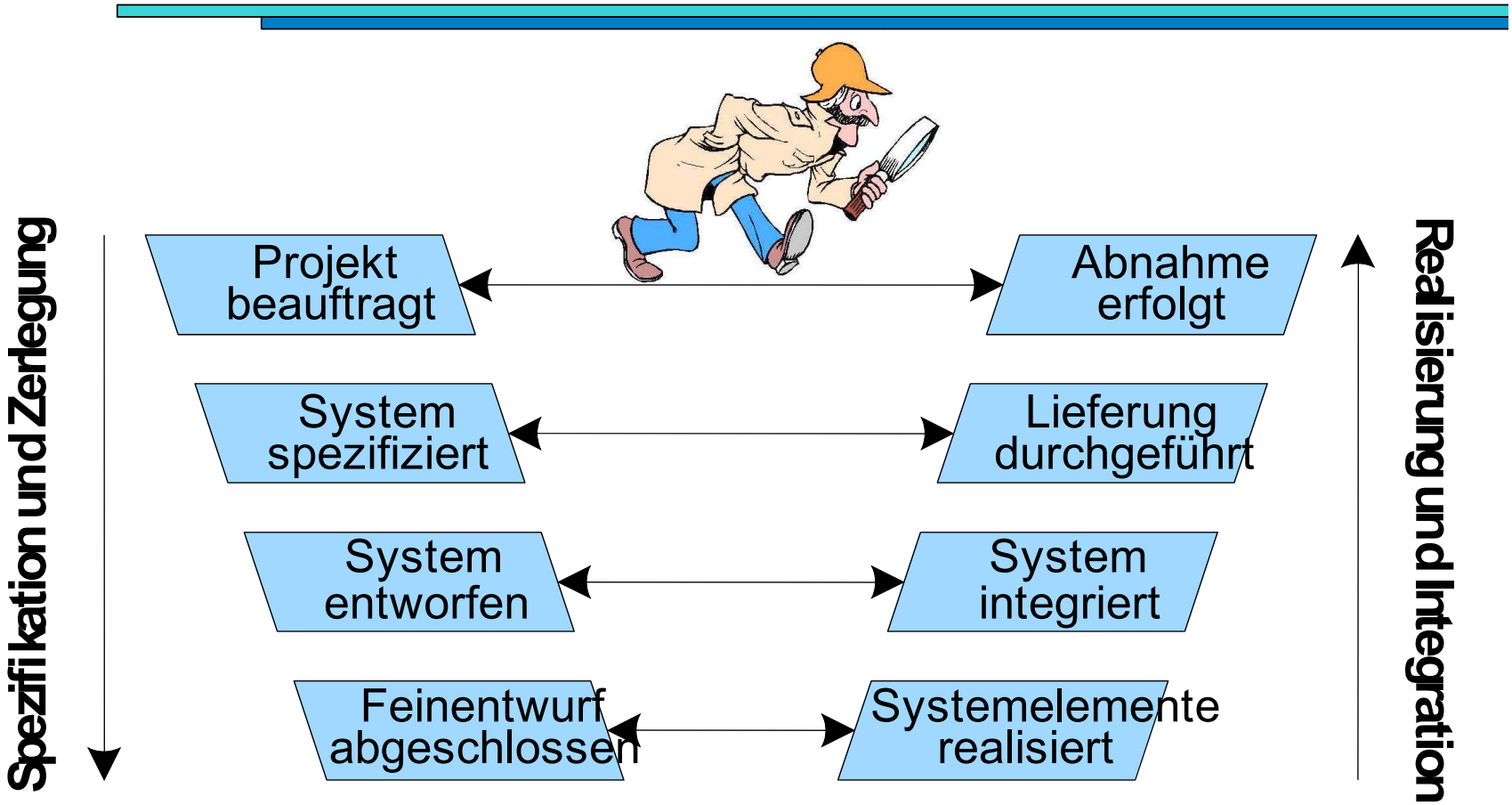
Kernprozess zur Entwicklung von elektronischen Systemen und Software (Nach Schäuffele, Zurawka)



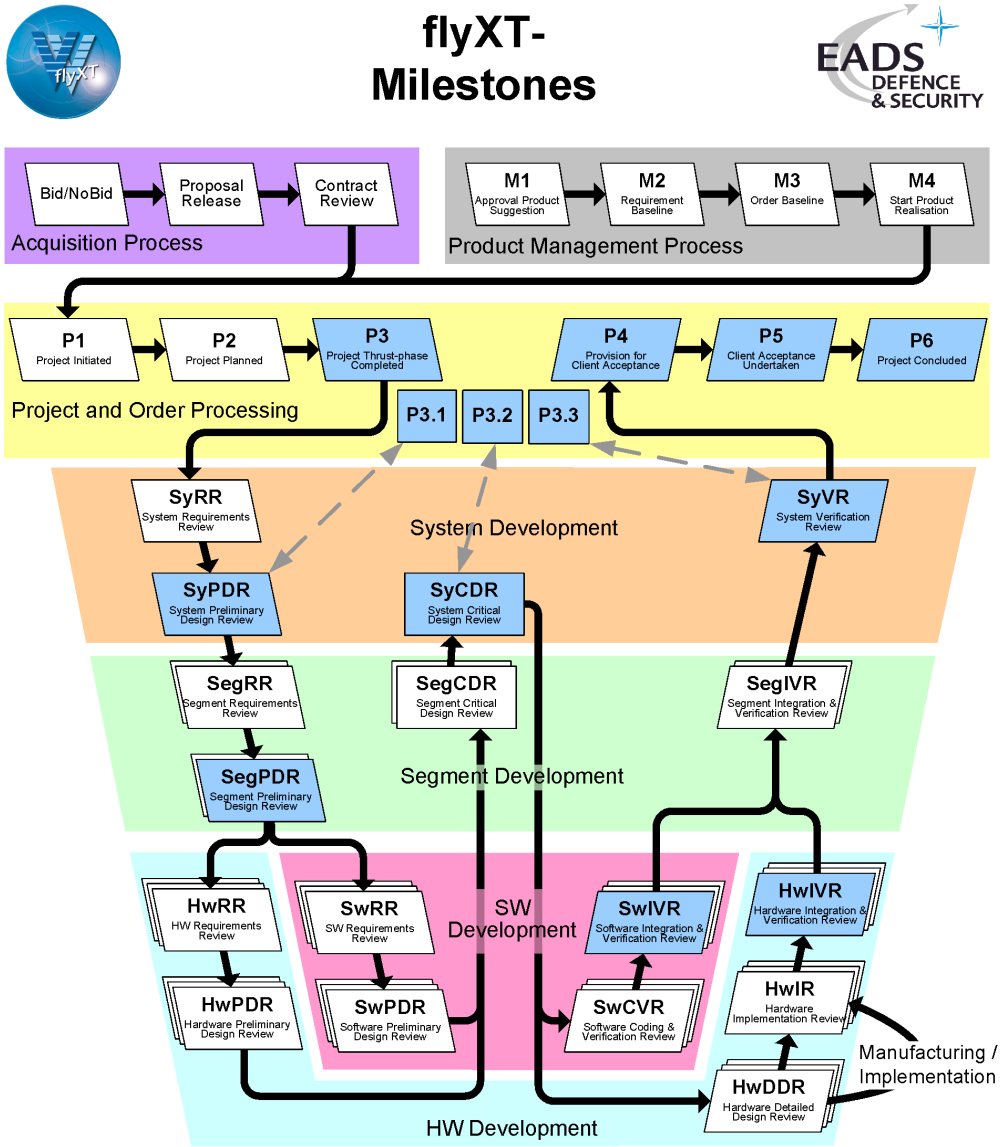
Vereinfachtes V-Modell



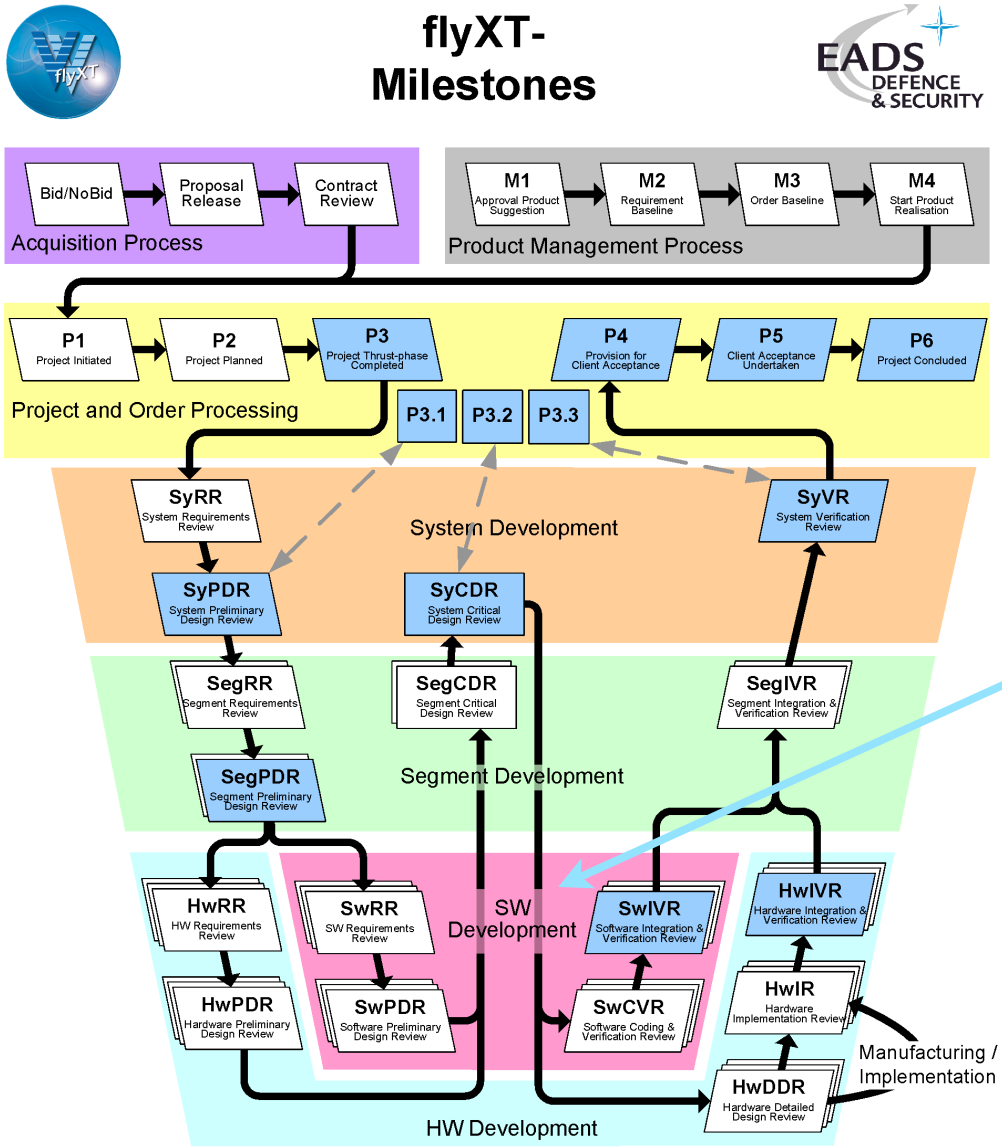
Verifizierung und Validierung



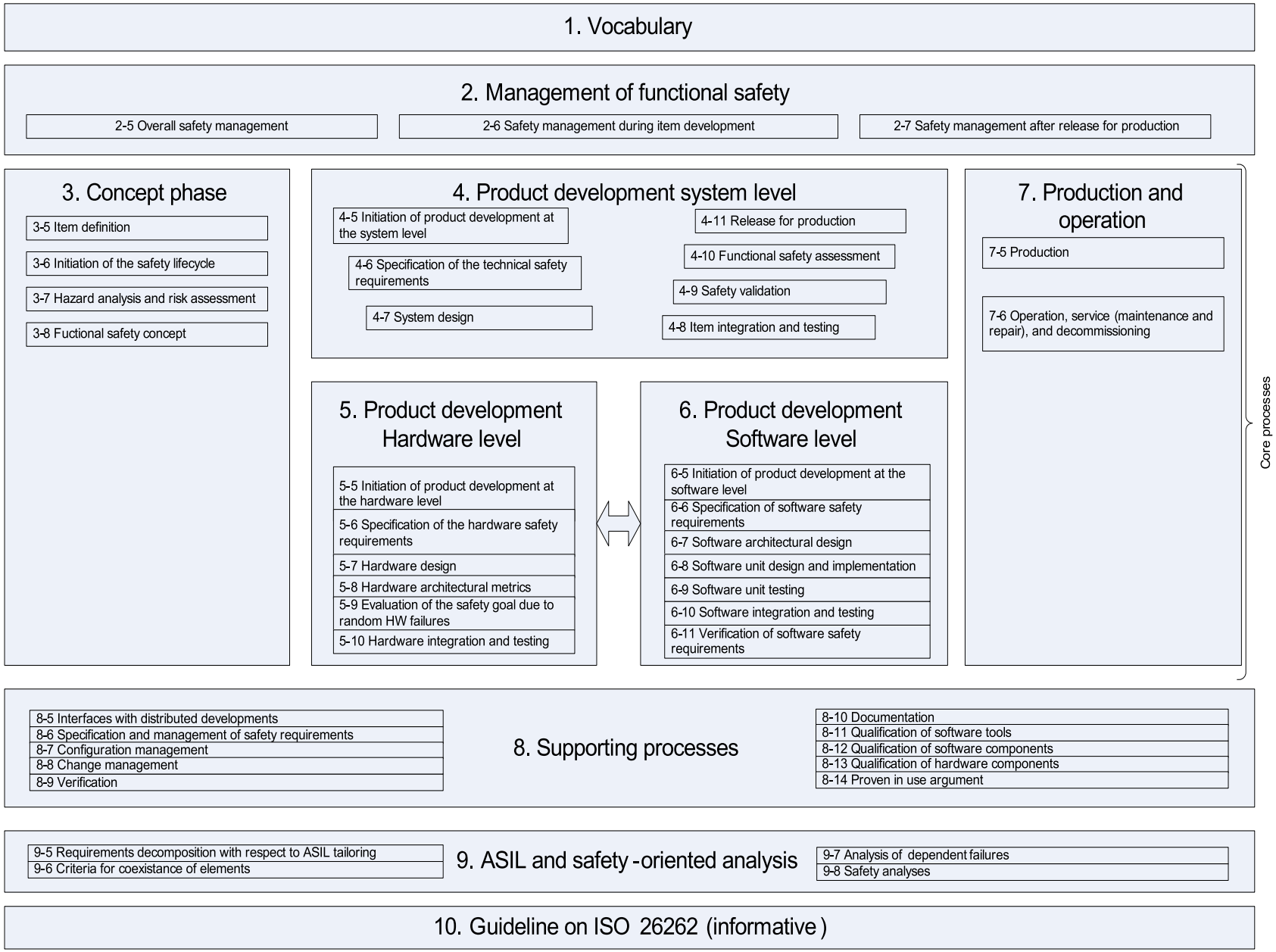
flyXT - EADS-Ausprägung des V-Modell XT



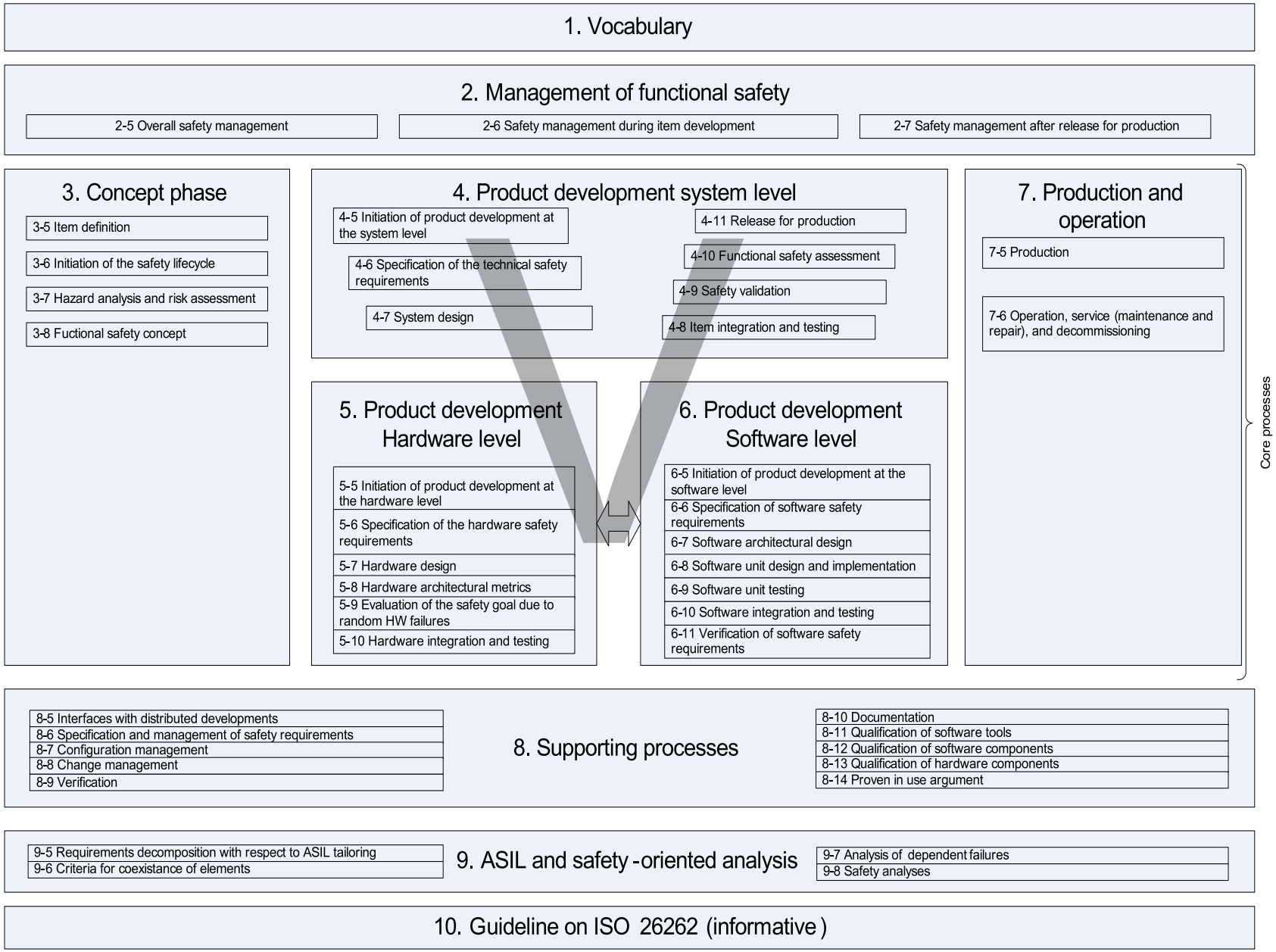
flyXT - EADS-Ausprägung des V-Modell XT



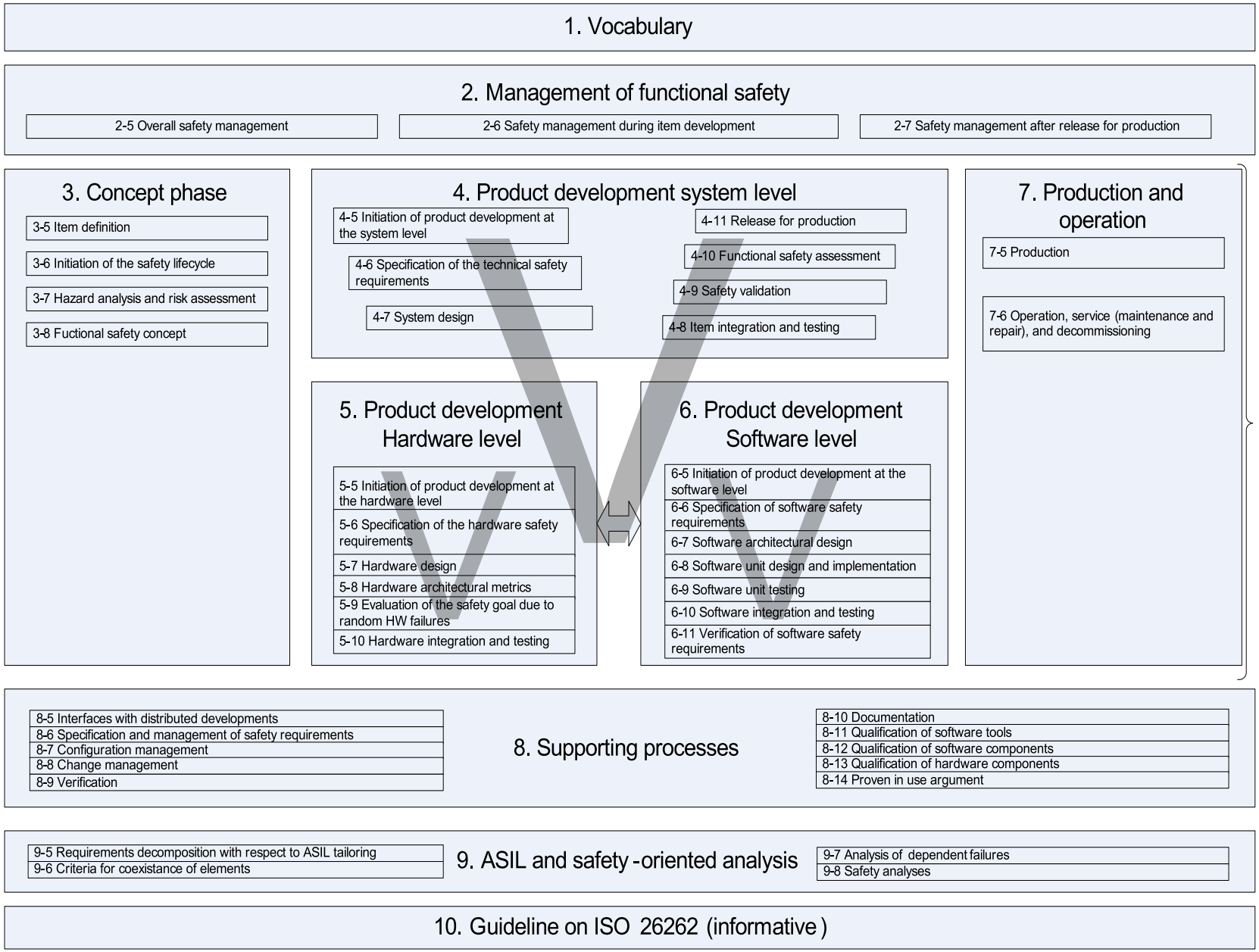
Automotive Safety Standard ISO 26262 im Detail



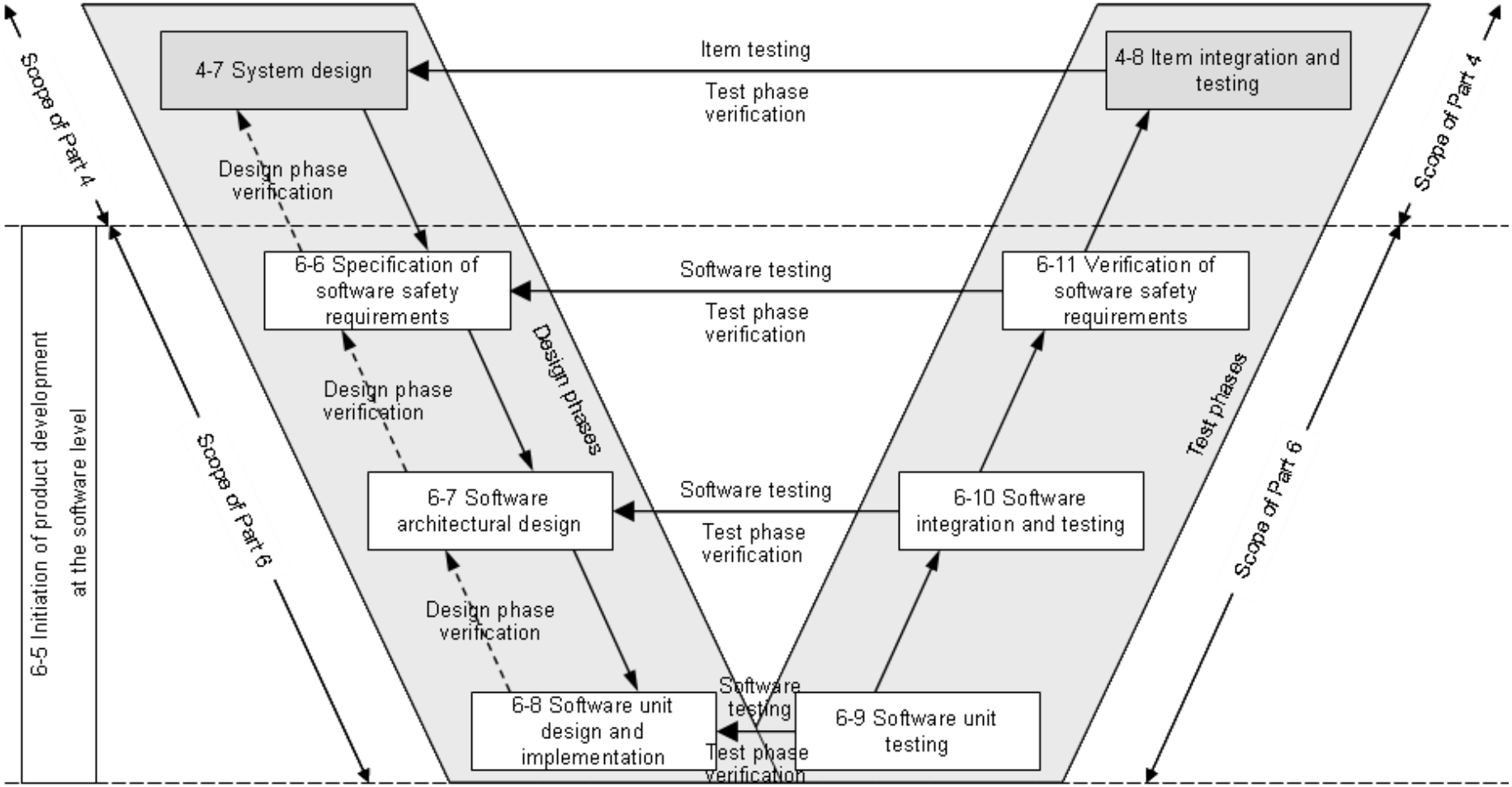
Automotive Safety Standard ISO 26262 im Detail



Automotive Safety Standard ISO 26262 im Detail

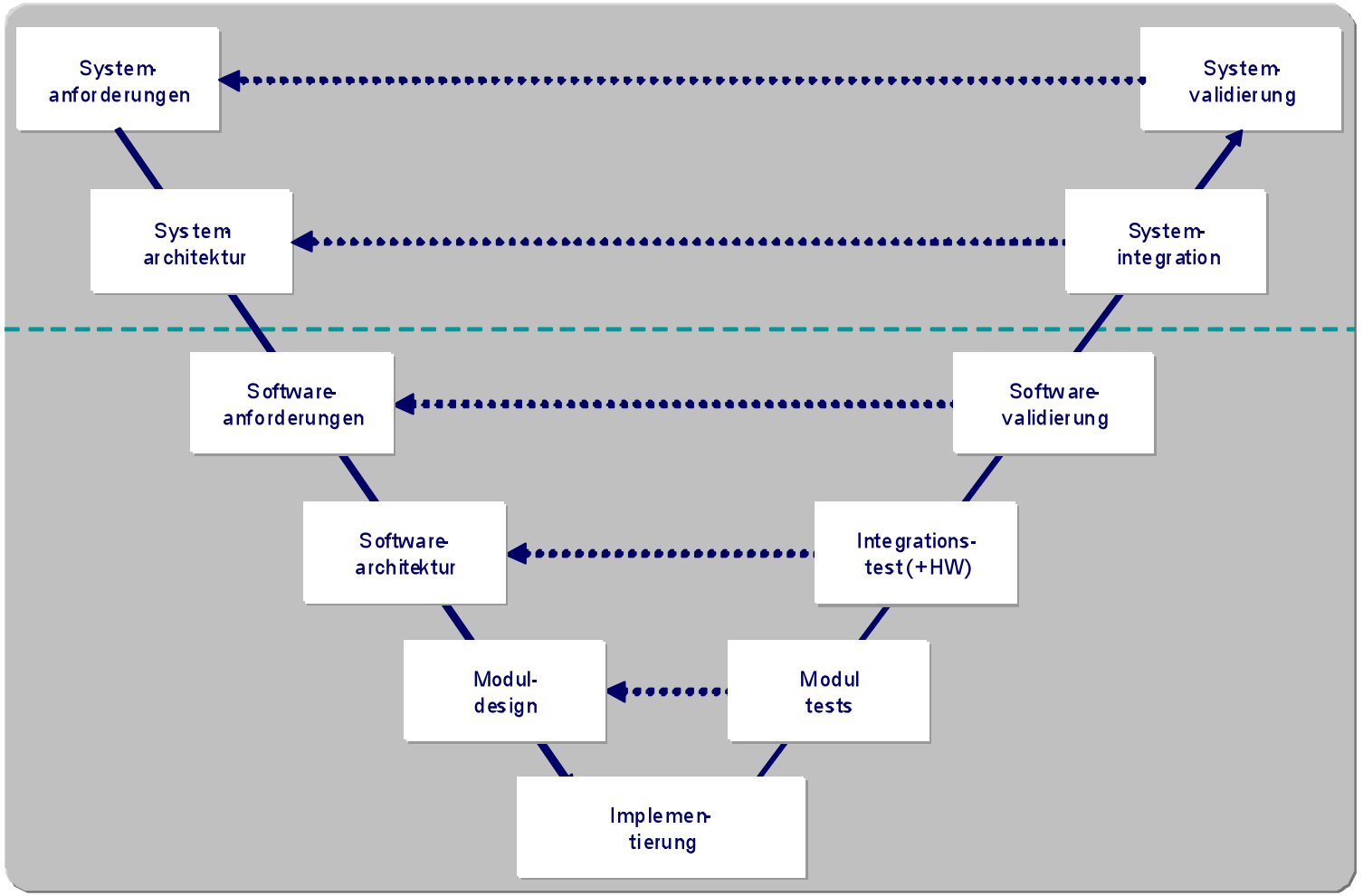


Reference phase model for the software development
ISO 26262



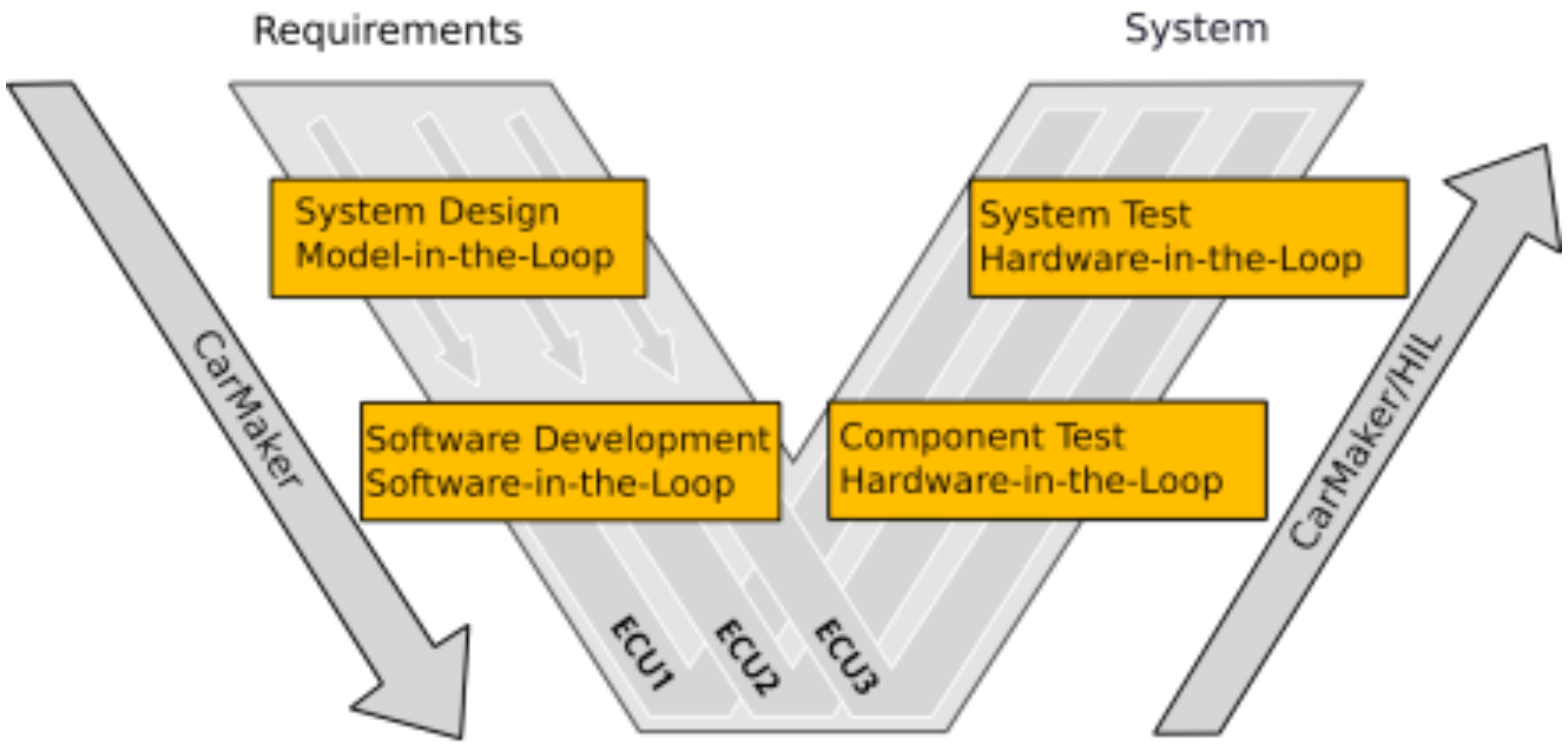
siehe auch Teil 7 Normen und Standards

V-Modell nach EN 50128



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

www.ipg.de

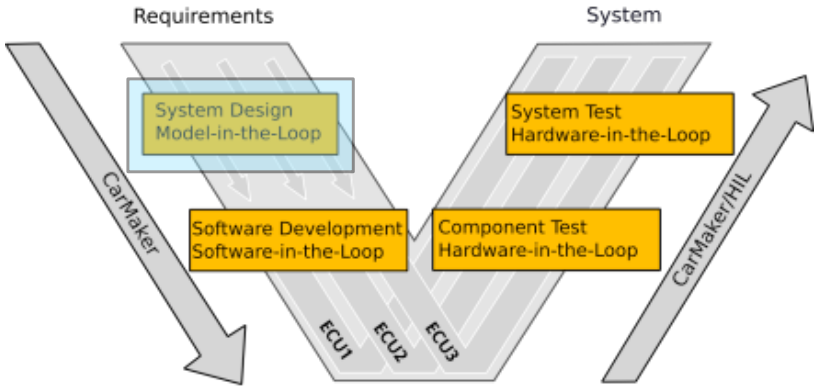

Requirements
System
System Design
Model-in-the-Loop
System Test
Hardware-in-the-Loop
Software Development
Software-in-the-Loop
Component Test
Hardware-in-the-Loop
CarMaker
CarMaker/HIL
ECU1
ECU2
ECU3

System Design (Quelle www.ipg.de)

Ausgangspunkt der Entwicklung sind die Anforderungen, die in Form einer Systemspezifikation festgehalten werden.

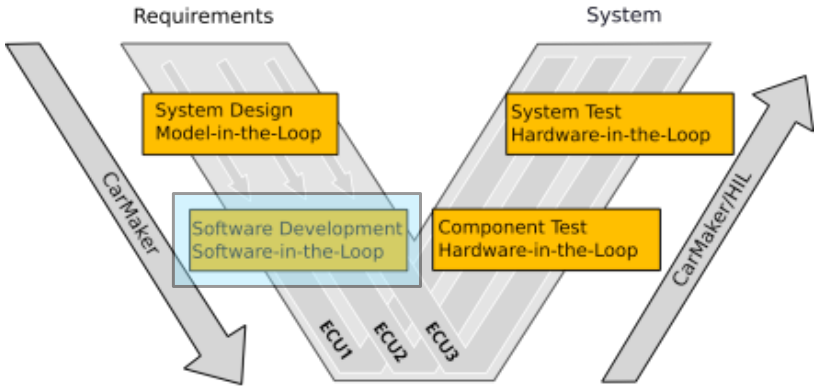
Die Funktionen der Systemspezifikation werden in Software-Modellen mit grafisch orientierten Programmier-systemen wie MATLAB/Simulink entwickelt. Diese Phase kann durch die Model-in-the-Loop (MIL)-Simulation unterstützt werden. Dabei werden die Software-Modelle mit CarMaker im virtuellen Fahrzeug validiert.

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.



Software Entwicklung (Quelle www.ipg.de)

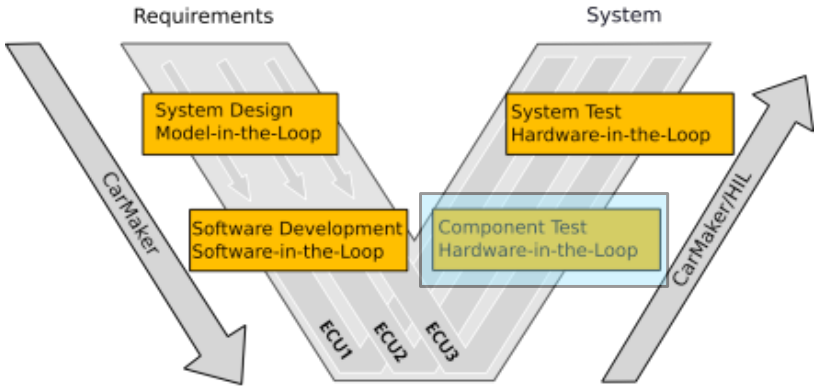
Das in der MIL-Simulation getestete Software-Modell wird durch die Software des Reglers ersetzt und in die Simulationsumgebung von CarMaker eingebunden (Software-in-the-Loop SIL). Somit wird der spätere Seriencode in einer virtuellen Fahrzeugumgebung ausgeführt, um das Verhalten zu untersuchen und Fehler 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.



Component Test (Quelle www.ipg.de)

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).

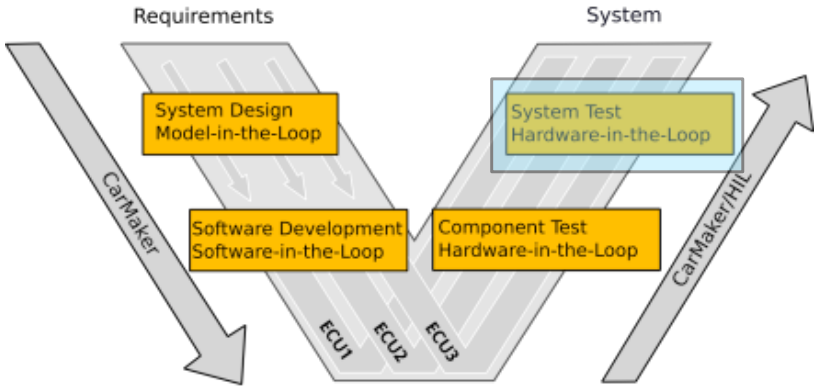
Die Bandbreite der Tests am HIL-Prüfstand reicht von Performance-Tests bis hin zu Sicherheitstest, bei denen das Verhalten des Fahrzeugs in Folge von elektrischen Fehlern geprüft wird. Für diese Tests bietet IPG zusätzlich den FailSafeTester an, mit dem elektrische Fehler am HIL-Prüfstand automatisch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



System Test (Quelle www.ipg.de)

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 Prüstand 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.

Steuergerät

Fahrzeug / Umwelt

	simuliert	real
simuliert	**SIL**	**HIL**
real	**Prototyp**	**Fahrversuch**

1. Kernprozess

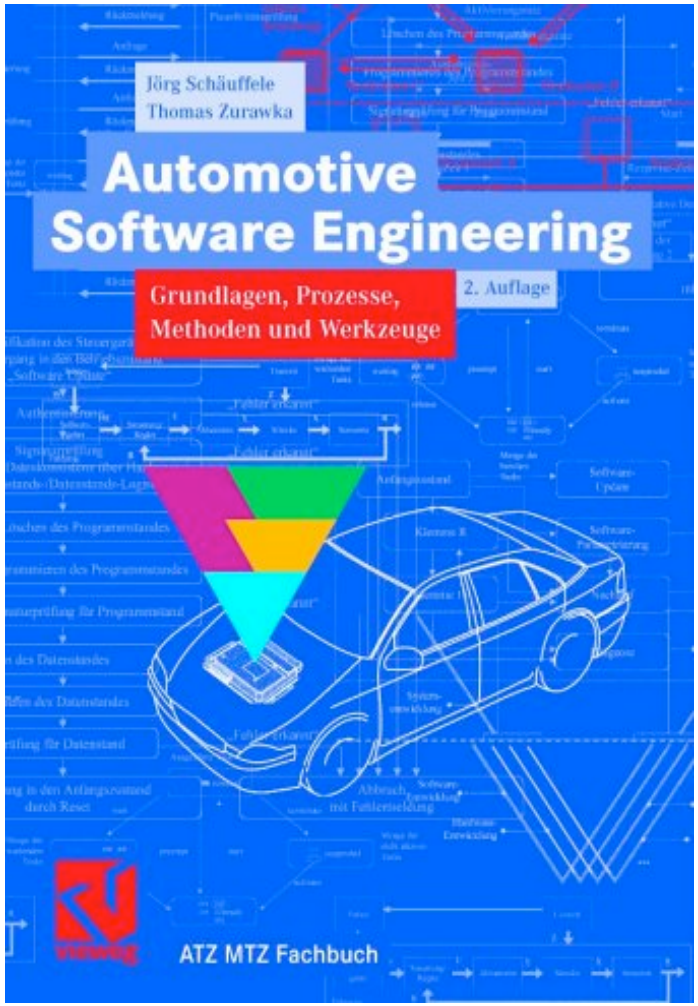
2. Unterstützungsprozesse

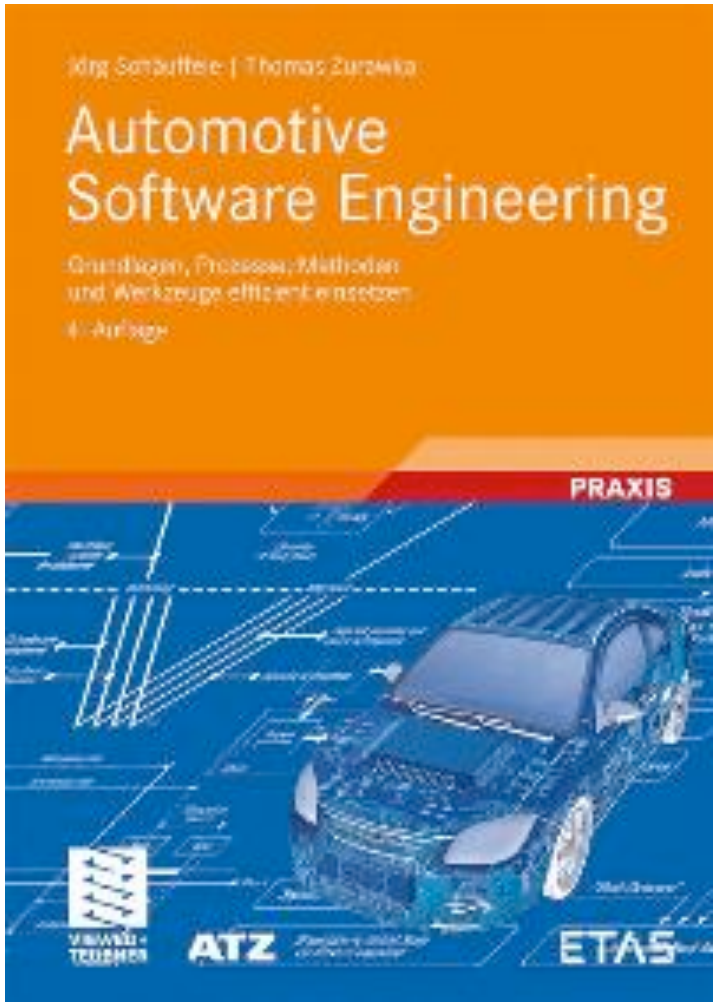
6. SW-Entwicklung / 1. Kernprozess
Kernprozess zur Entwicklung von elektronischen Systemen und Software

1. Grundbegriffe
2. Entwicklungsobjekt: Kombiinstrument
3. Analyse der Benutzeranforderungen und Spezifikation der logischen Systemarchitektur
4. Analyse der logischen Systemarchitektur und Spezifikation der technischen Systemarchitektur
5. Analyse der Software-Anforderungen und Spezifikation der technischen Softwarearchitektur
6. Spezifikation der Software-Komponenten
7. Design und Implementierung der Software-Komponenten
8. Test der Software-Komponenten
9. Integration der Software-Komponenten
10. Integrationstest der Software
11. Integration der System-Komponenten
12. Integrationstest des Systems
13. Kalibrierung
14. Akzeptanz- und Systemtest

Quelle für Inhalte und Abbildungen

Kapitel 4



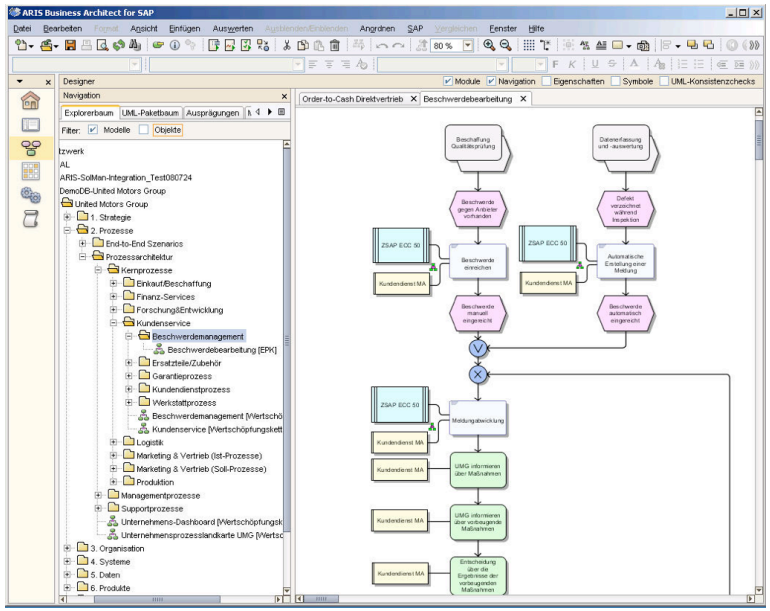


6. SW-Entwicklung / 1. Kernprozess
Kernprozess zur Entwicklung von elektronischen Systemen und Software

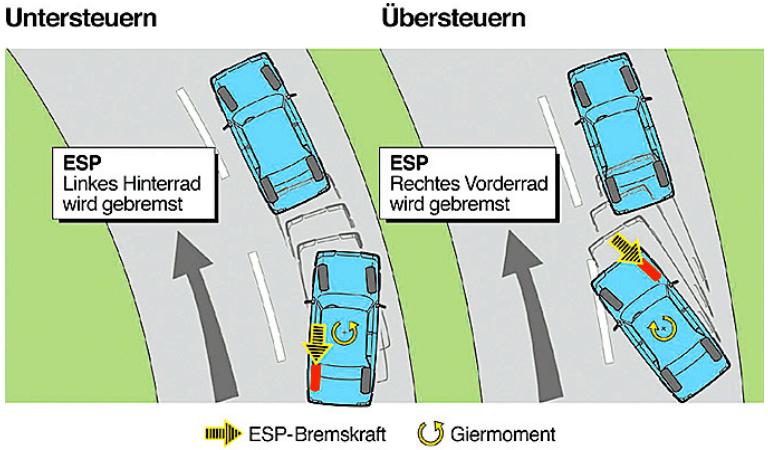
1. **Grundbegriffe**
2. Entwicklungsobjekt: Kombiinstrument
3. Analyse der Benutzeranforderungen und Spezifikation der logischen Systemarchitektur
4. Analyse der logischen Systemarchitektur und Spezifikation der technischen Systemarchitektur
5. Analyse der Software-Anforderungen und Spezifikation der technischen Softwarearchitektur
6. Spezifikation der Software-Komponenten
7. Design und Implementierung der Software-Komponenten
8. Test der Software-Komponenten
9. Integration der Software-Komponenten
10. Integrationstest der Software
11. Integration der System-Komponenten
12. Integrationstest des Systems
13. Kalibrierung
14. Akzeptanz- und Systemtest

Komponentenentwicklung und Systementwicklung

- Komponentenentwicklung
  - Analyse und Entwurf von Komponenten
  - Computerspiele
  - SAP
  - Keine oder wenig Bezug zu realer Umwelt
  - Benutzer und betriebliche Abläufe müssen sich der EDV anpassen, nicht umgekehrt



- Systementwicklung
  - Analyse und Entwurf des Systems als Ganzes
  - Liefert Vorgaben für Komponentenentwicklung
  - Embedded Systems / Eingebettete Systeme
    - Automotive
    - Luftfahrt
    - Bahnen
    - Medizin
  - Hoher Bezug zu realer Umwelt
  - Systeme wie ESP haben sich z.B. der Physik anzupassen



Embedded Systems

- „Eingebettete mikroelektronische Systeme sind die Treiber für innovative Technologien und Märkte des 21. Jahrhunderts. Fast alle volkswirtschaftlich relevanten Industriezweige, vom Automobilbau über die Automatisierungsbranche bis hin zur Medizintechnik erreichen ihre Spitzenstellung nur durch den Einsatz von Embedded Systems. Ohne sie fliegt kein Flugzeug, funktioniert kein Mobiltelefon und kein Computertomograph. Unentbehrlicher Bestandteil von Embedded Systems ist die Software. Meist werden die Programme in der verarbeitenden Industrie selbst geschrieben. Daher zählen viele deutsche Unternehmen zu den weltweit führenden Softwareproduzenten.“

  (Presseinformation Pr 173-2009 des ZVEI - Zentralverband Elektrotechnik- und Elektronikindustrie e.V.anlässlich des 4. Nationalen IT-Gipfels in Stuttgart am 8. Dezember 2009)
- 5. IT-Gipfel Anfang Dezember 2010 in Dresden
- 6. IT-Gipfel Anfang Dezember 2011 in München
- Am 8. Dezember 2009 wurde auf dem 4. IT-Gipfel in Stuttgart im Beisein von Frau Bundeskanzlerin Dr. Angela Merkel die Nationale Roadmap Embedded Systems vorgestellt.

Nationale Roadmap Embedded Systems (1)

- Ausgehend von zehn Thesen für Embedded Systems für Deutschland (siehe Kasten) werden Beiträge zur Lösung gesellschaftlicher und ökonomischer Herausforderungen (z.B. Mobilität und Sicherheit) ausgearbeitet. Die identifizierten Forschungsprioritäten werden in Technologieinnovationen und Prozessinnovationen gegliedert.
- Beispiele
- Funktionale Sicherheit Eingebetteter Systeme
- Requirements Engineering
- Architektur – Entwurf und Bewertung
- Systemanalyse
- Modellgetriebene Entwicklung
- Systematische Wiederverwendung

Nationale Roadmap Embedded Systems (2)

- „Eine weitere Herausforderung stellt die Sicherung eines ausreichenden Angebotes qualifizierter Fachkräfte dar. Es gibt nur wenige Studiengänge, die die Bandbreite der für den Bereich Eingebettete Systeme notwendigen Wissensdomänen integrieren. Darüber hinaus ist der Bereich Systems-Engineering in der klassischen Ausbildung zu wenig verankert. Hier sind verstärkt koordinierte Anstrengungen zur Sicherstellung entsprechender Ausbildungsangebote auf allen Ausbildungsebenen einschließlich der beruflichen Weiterbildung erforderlich.“
- Zu den Initiatoren der Roadmap zählen die Professoren Manfred Broy (TU München), Werner Damm (Universität Oldenburg und OFFIS e.V.) und Peter Liggesmeyer (TU Kaiserslautern und Fraunhofer IESE) sowie Prof. Heinrich Dämbkes (EADS Ulm). Herausgeber der Roapmap ist der ZVEI - Zentralverband Elektrotechnik- und Elektronikindustrie e.V.

Nationale Roadmap Embedded Systems (3)

- EMBEDDED SYSTEMS FÜR DEUTSCHLAND – ZEHN THESEN

1. Die zentralen ökonomischen und gesellschaftlichen Herausforderungen in Deutschland lassen sich ohne die Querschnittstechnologie Embedded Systems nicht lösen.
2. Arbeitsplätze und Wertschöpfung in für Deutschland wesentlichen Branchen hängen in zunehmendem Maße von Embedded Systems ab.
3. Embedded Systems sind zunehmend produktprägender Bestandteil mindestens in den drei umsatzstärksten Branchen Deutschlands.
4. Der Anteil von Embedded Systems an den Gesamtproduktentwicklungskosten wächst in allen Branchen signifikant an. Dies wird gespiegelt durch einen signifikanten Anteil von 10 % bis 20 % an den Gesamtkosten für Forschung und Entwicklung in vielen Industriezweigen.
5. Deutschland verfügt über eine exzellente Ausgangsposition, die zum Erhalt und zum Ausbau der Wettbewerbsfähigkeit jedoch einer Stärkung bedarf.
6. Es bedarf einer gemeinsamen, branchenübergreifenden Anstrengung von Industrie und Forschung mit Unterstützung durch geeignete Förderprogramme, um die zukünftigen Herausforderungen zu meistern.

Nationale Roadmap Embedded Systems (4)

7. Die wesentlichen zukünftigen Herausforderungen im Bereich Embedded Systems können mit Hilfe von sechs Forschungsschwerpunkten (FSPs) bewältigt werden.
   FSP Seamless Interaction
   FSP Autonome Systeme
   FSP Verteilte Echtzeit-Situationserkennung und Lösungsfindung
   FSP Sichere Systeme
   FSP Architekturprinzipien
   FSP Virtual Engineering
   Eine wesentliche Rolle spielen dabei offene branchenübergreifende Interoperabilitätsstandards, geeignete Referenz-Technologie-Plattformen und auf Eingebettete Systeme ausgerichtete Ausbildungsprogramme.
8. Der Gesamtbedarf an Forschungsaufwänden in diesen sechs Schwerpunkten wird für die nächsten 10 Jahre auf deutlich über 2,5 Mrd. EURO geschätzt.
9. Die Kombination von nationalen Programmen (z. B. Innovationsallianzen) und europäischen Förderinstrumenten (z. B. ARTEMIS) stellt bei entsprechender finanzieller Ausstattung einen ausgezeichneten Rahmen zur Schaffung von Spitzen-Innovationen in Deutschland und zur Mitgestaltung dafür maßgeblicher internationaler Standards dar.

Nationale Roadmap Embedded Systems (5)

10. Deutschland kann durch eine enge Zusammenarbeit zwischen Experten der Embedded-Systems-Technologien und der verschiedenen Anwendungsfelder (Gesundheit, Mobilität, Energie,…) eine Spitzenrolle bei der Lösung zentraler gesellschaftlicher und ökonomischer Herausforderungen einnehmen.

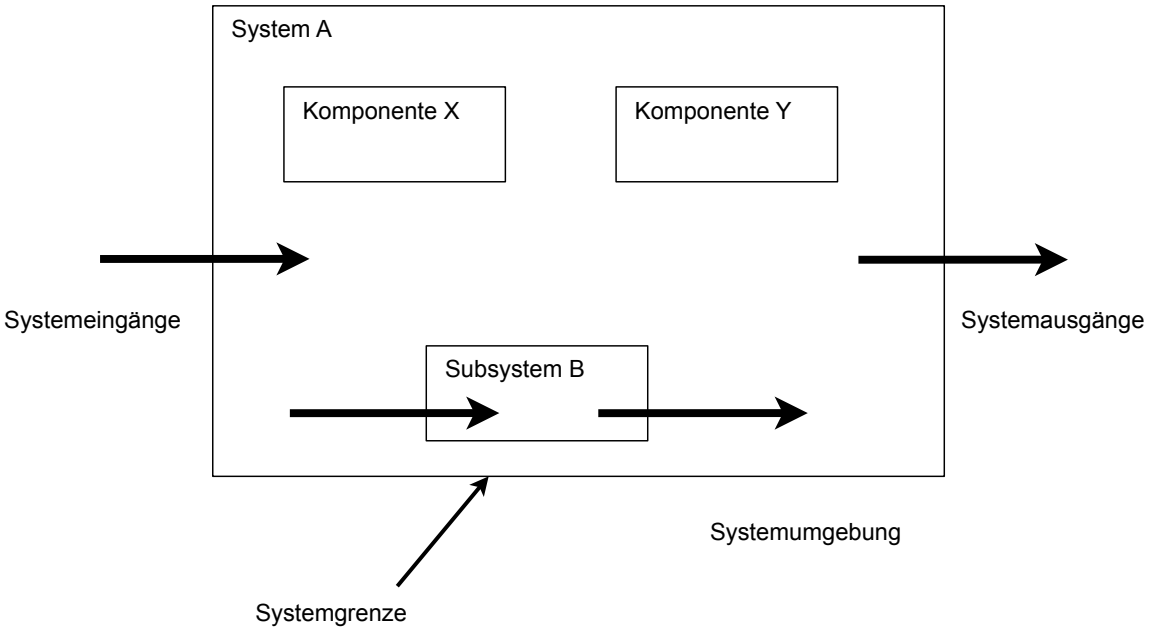
- Quelle: http://www.zvei.de/index.php?id=1829

Grundbegriffe der Systemtheorie

- Systemtheorie = Vorgehensweisen zum Umgang mit Komplexität
- Ansatz: Divide et impera / Teile und Beherrsche
- Annahmen
  - Die Aufteilung des Systems in Komponenten verfälscht nicht das betrachtete Problem.
  - Die Komponenten für sich betrachtet sind in wesentlichen Teilen identisch mit den Komponenten des Systems.
  - Die Prinzipien für den Zusammenbau der Komponenten zum System sind einfach, stabil und bekannt.
- Diese Annahmen sind bei der Entwicklung von elektronischen Systemen und Software für Fahrzeuge erfüllt bzw. erfüllbar.
- Die Eigenschaften des Systems ergeben sich aus dem Zusammenspiel der Komponenten
- Komplexität des Systems führt zu komplexen und aufwändigen Analysen der Komponenten und ihres Zusammenspiels = Kern der Systemtheorie
- Komponenten
  - Technische Bauteile
  - Menschen / Benutzer
  - Umwelt

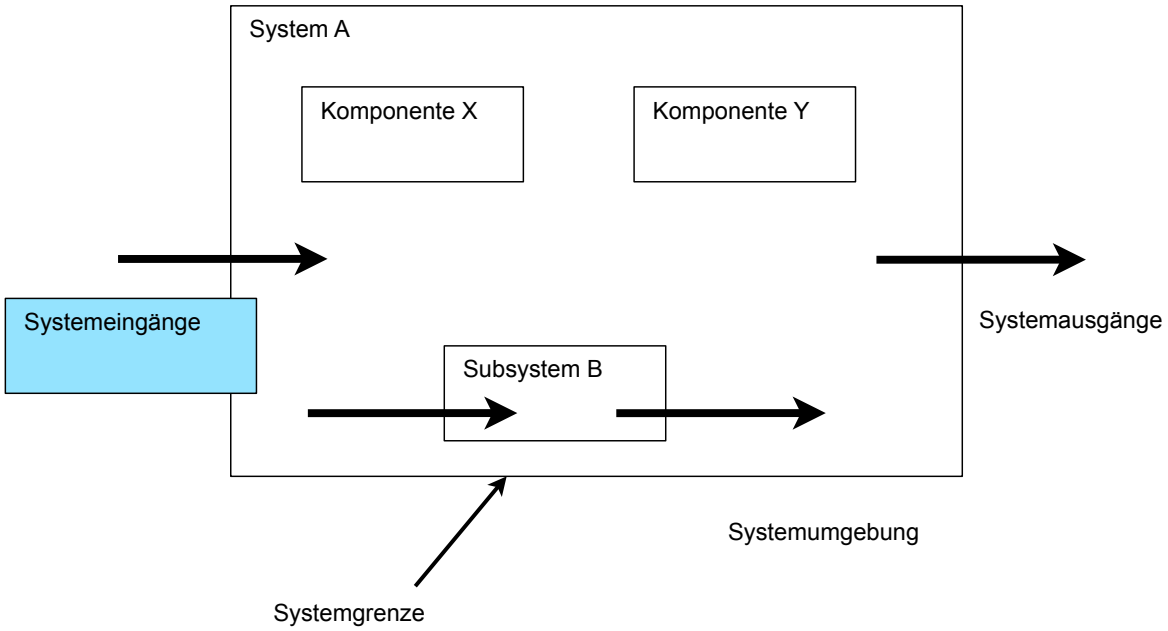
Systemdefinition
(Nach Schäuffele, Zurawka)

- Ein System ist eine von ihrer Umgebung abgegrenzte Anordnung aufeinander einwirkender Komponenten.
- Ein System interagiert mit seiner Umgebung über Systemeingänge und Systemausgänge
- Beispiel
  - Systemeingänge
    - Aktionen des Fahrers
      - Beschleunigen
      - Bremsen
      - Lenken
      - Blinken
    - Sensorsignale
      - Reibwerterkennung
  - System
    - Fahrzeug
  - Systemausgänge
    - Geschwindigkeitsvektor
    - Aktionen
      - Blinker leuchtet



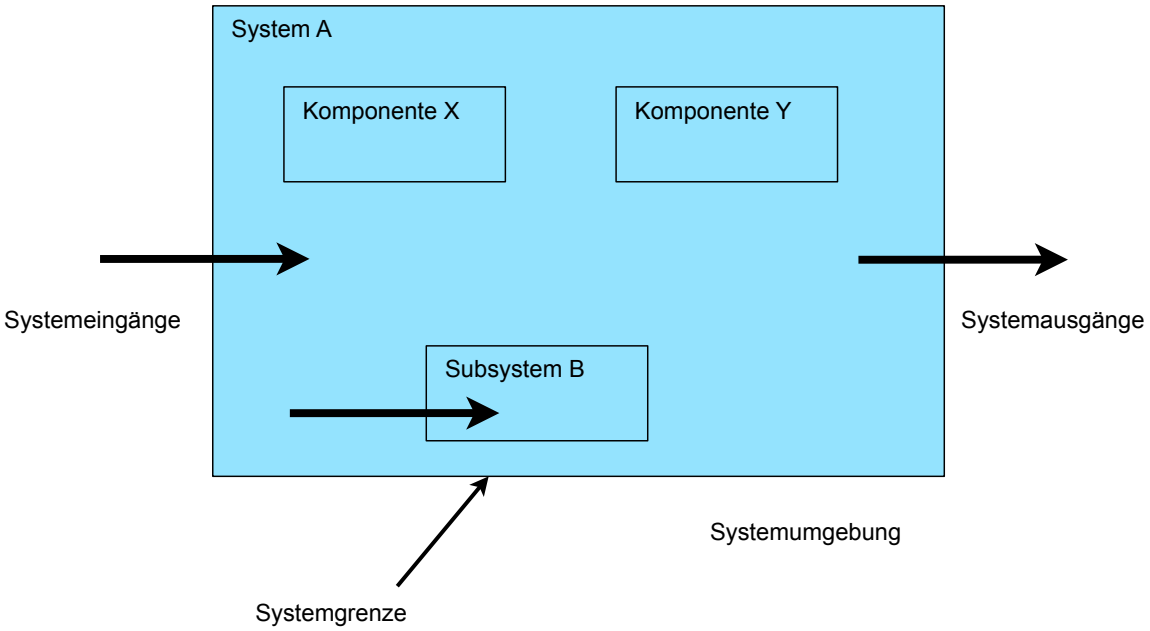
Systemdefinition

- Ein System ist eine von ihrer Umgebung abgegrenzte Anordnung aufeinander einwirkender Komponenten
- Beispiel
  - **Systemeingänge**
    - Aktionen des Fahrers: Sollwerte-Vorgaben
      - Beschleunigen
      - Bremsen
      - Lenken
      - Blinken
    - Sensorsignale
      - Reibwerterkennung
  - System
    - Fahrzeug
  - Systemausgänge
    - Geschwindigkeitsvektor
    - Aktionen
      - Blinker leuchtet



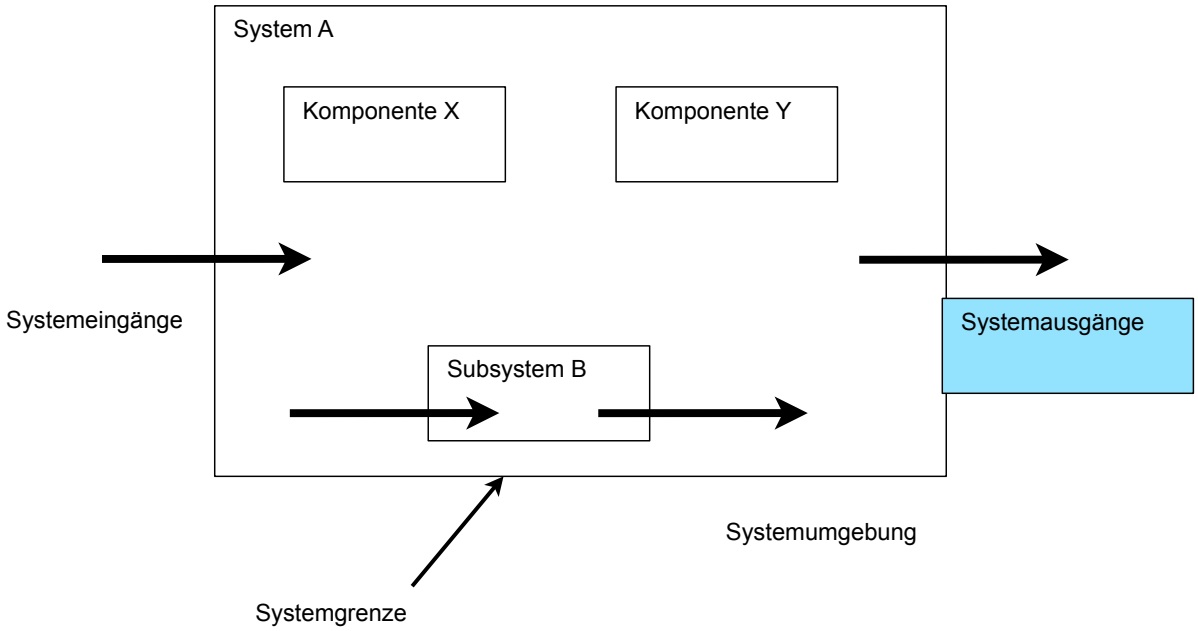
Systemdefinition

- Ein System ist eine von ihrer Umgebung abgegrenzte Anordnung aufeinander einwirkender Komponenten
- Beispiel
  - Systemeingänge
    - Aktionen des Fahrers
      - Beschleunigen
      - Bremsen
      - Lenken
      - Blinken
    - Sensorsignale
      - Reibwerterkennung
  - **System**
    - Fahrzeug
  - Systemausgänge
    - Geschwindigkeitsvektor
    - Aktionen
      - Blinker leuchtet



Systemdefinition

- Ein System ist eine von ihrer Umgebung abgegrenzte Anordnung aufeinander einwirkender Komponenten
- Beispiel
  - Systemeingänge
    - Aktionen des Fahrers
      - Beschleunigen
      - Bremsen
      - Lenken
      - Blinken
    - Sensorsignale
      - Reibwerterkennung
  - System
    - Fahrzeug
  - **Systemausgänge**
    - Geschwindigkeitsvektor
    - Aktionen
      - Blinker leuchtet



Systemzustand

- Systemzustand
  - Menge von Eigenschaften, die das System zu einem bestimmten Zeitpunkt beschreiben
  - Beispiele
    - Motor läuft mit 4862 Umdrehungen / min
    - Getriebe läuft im 5. Gang
    - Im Radio läuft MDR Jump

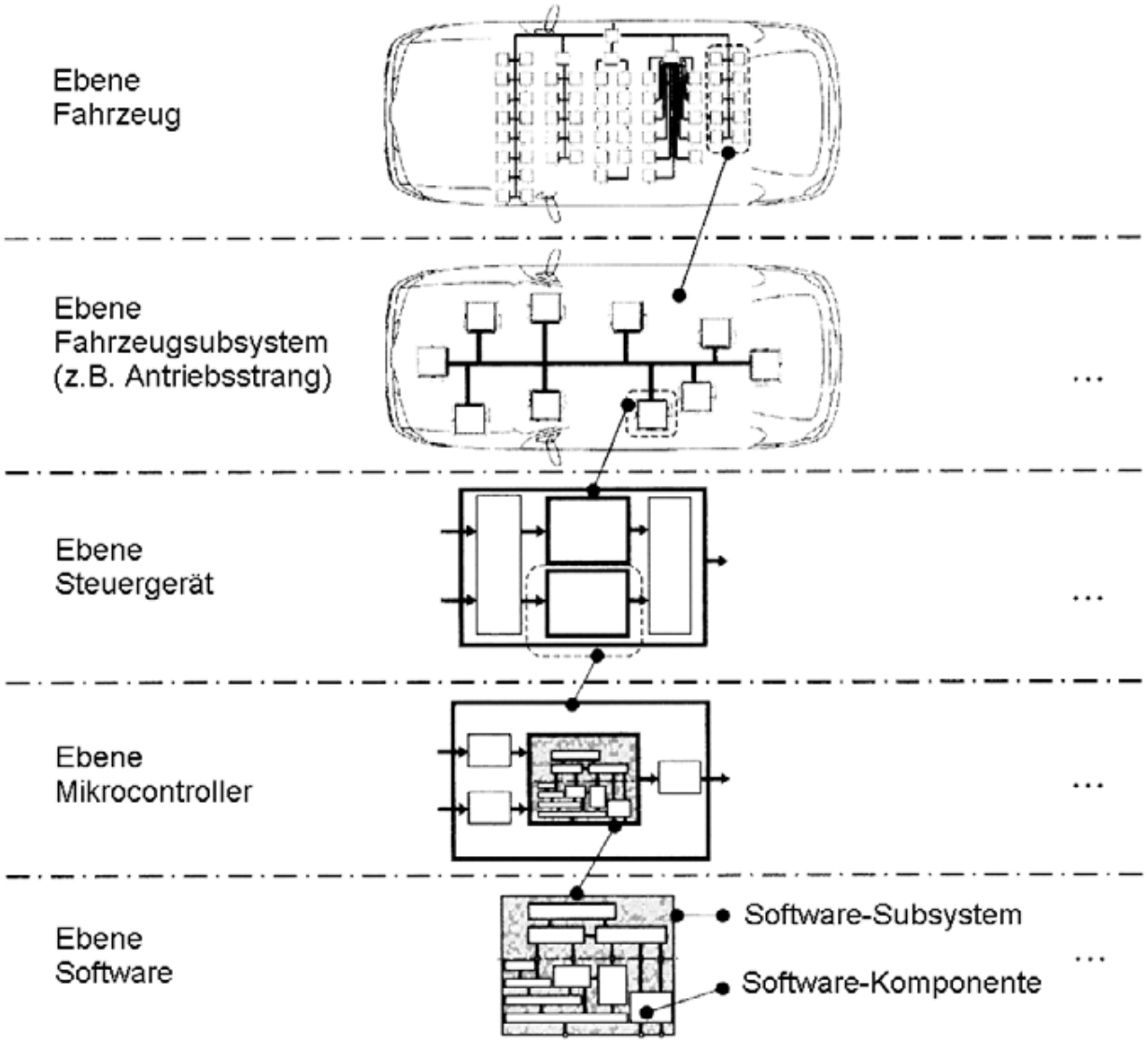
Systemumgebung

- Systemumgebung (Umgebung)
  - Menge von Komponenten, die nicht zum System gehören, aber das System beeinflussen (können)
  - Beispiel
    - System: Fahrzeug
    - Umgebung: Fahrer, Strasse
- Systemschnittstellen
    - Signale aus der Umgebung (Systemeingänge)
    - Signale an die Umgebung (Systemausgänge)
  - über die Systemgrenze
  - Komponenten eines Systems können wiederum Systeme sein (Subsysteme)
    - Mehrere Systemebenen (Betrachtungsebenen, Abstraktionsebenen)
  - Aussensicht
    - Abstraktion der Systemsicht auf Systemgrenze und Systemschnittstellen
    - Keine Unterscheidung zwischen Komponente und System

Modellierung von Systemen

- Methoden zur Modellierung von Systemen
  - Abstraktion durch
    - Hierarchiebildung
      - Zusammenführung von Komponenten zum System
      - Bezeichnung: Integration oder Komposition
    - Modularisierung
      - Zerlegung eines Systems in Komponenten
      - Bezeichnung: Partitionierung oder Dekomposition
- „7 +/- 2 - Regel“
  - Systeme mit mehr als 9 = 7 + 2 Komponenten erscheinen komplex
  - Systeme mit weniger als 5 = 7 - 2 Komponenten erscheinen trivial

Beispiel: Systemebenen in der Fahrzeugelektronik (Nach Schäuffele, Zurawka)



Systems Engineering (1)

Im Gegensatz zur Komponentenentwicklung zielt die Systementwicklung (engl. Systems Engineering) auf die Analyse und den entwurf des Systems als Ganzes und nicht auf die Analyse und den Entwurf seiner Komponenten. ...

Systems Engineering ist die gezielte Anwendung von wissenschaftlichen und technischen Ressourcen

- Zur Transformation eines operationellen Bedürfnisses in die Beschreibung einer Systemkonfiguration unter bestmöglicher Berücksichtigung aller operativen Anforderungen und nach den Maßstäben der gebotenen Effektivität.
- Zur Integration aller technischen Parameter und zur Sicherstellung der Kompatibilität aller physikalischen, funktionalen und technischen Schnittstellen in einer Art und Weise, so dass die gesamte Systemdefinition und der Systementwurf möglichst optimal werden.
- Zur Integration der Beiträge aller Fachdisziplinen in einen ganzheitlichen Entwicklungsansatz.

(Nach Schäuffele/Zurawka, Einleitung zu Kapitel 4)

Systems Engineering (2)

- Die Definition ist angelehnt an Definitionen von
- SEI/CMU CMMI - Capability Maturity Model Integration
  http://www.sei.cmu.edu/cmmi
  - Software Engineering Institute (SEI) an der Carnegie Mellon University (CMU)
- INCOSE - International Council on Systems Engineering
  http://www.incose.org
- GfSE - Gesellschaft für Systems Engineering e.V.
  German Chapter of INCOSE
  http://www.gfse.de

What is Systems Engineering?

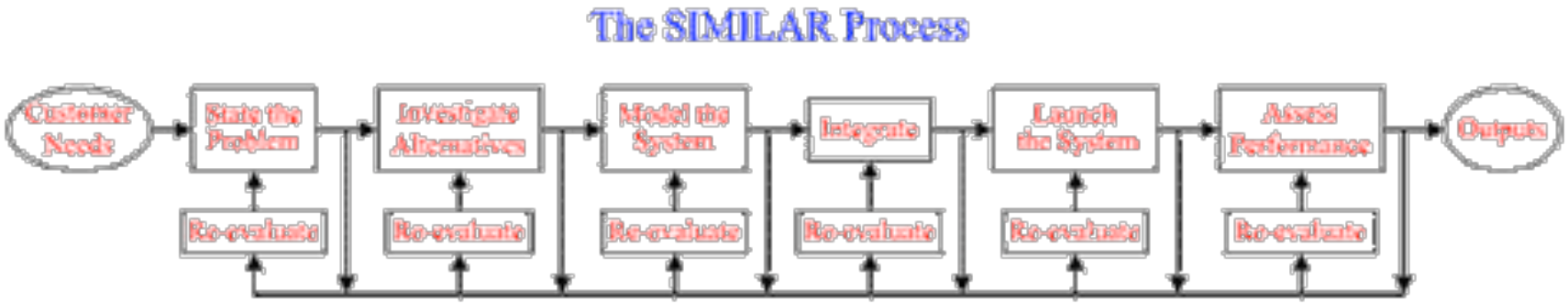
- Systems Engineering is an interdisciplinary approach and means to enable the realization of successful systems. It focuses on defining customer needs and required functionality early in the development cycle, documenting requirements, then proceeding with design synthesis and system validation while considering the complete problem:
  - Operations
  - Cost & Schedule
  - Performance
  - Training & Support
  - Test
  - Disposal
  - Manufacturing
- Systems Engineering integrates all the disciplines and specialty groups into a team effort forming a structured development process that proceeds from concept to production to operation. Systems Engineering considers both the business and the technical needs of all customers with the goal of providing a quality product that meets the user needs.
- Definition of the International Council on Systems Engineering (INCOSE)

- Näheres in den Unterlagen (in der Vorlesung ausgeblendet)

What is Systems Engineering?
A Consensus of the INCOSE Fellows (1)

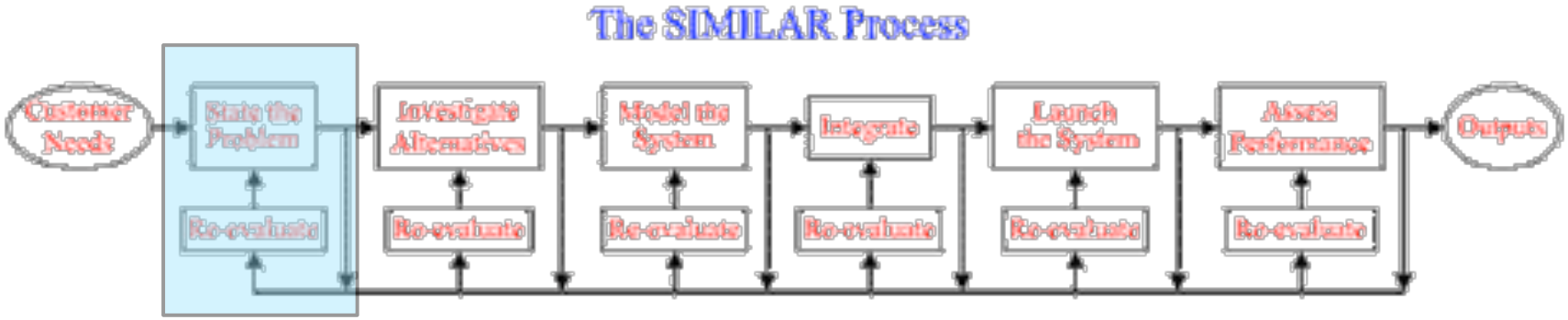
- Definition of a system
  A system is a construct or collection of different elements that together produce results not obtainable by the elements alone. The elements, or parts, can include people, hardware, software, facilities, policies, and documents; that is, all things required to produce systems-level results. The results include system level qualities, properties, characteristics, functions, behavior and performance. The value added by the system as a whole, beyond that contributed independently by the parts, is primarily created by the relationship among the parts; that is, how they are interconnected (Rechtin, 2000).
- Systems Engineering
  Systems Engineering is an engineering discipline whose responsibility is creating and executing an interdisciplinary process to ensure that the customer and stakeholder's needs are satisfied in a high quality, trustworthy, cost efficient and schedule compliant manner throughout a system's entire life cycle. This process is usually comprised of the following seven tasks: State the problem, Investigate alternatives, Model the system, Integrate, Launch the system, Assess performance, and Re-evaluate. These functions can be summarized with the acronym SIMILAR: State, Investigate, Model, Integrate, Launch, Assess and Re-evaluate. This Systems Engineering Process is shown in Figure 1. It is important to note that the Systems Engineering Process is not sequential. The functions are performed in a parallel and iterative manner.

What is Systems Engineering?
A Consensus of the INCOSE Fellows (2)



- Figure 1. The Systems Engineering Process from A. T. Bahill and B. Gissing, Re-evaluating systems engineering concepts using systems thinking, IEEE Transaction on Systems, Man and Cybernetics, Part C: Applications and Reviews, 28 (4), 516-527, 1998.

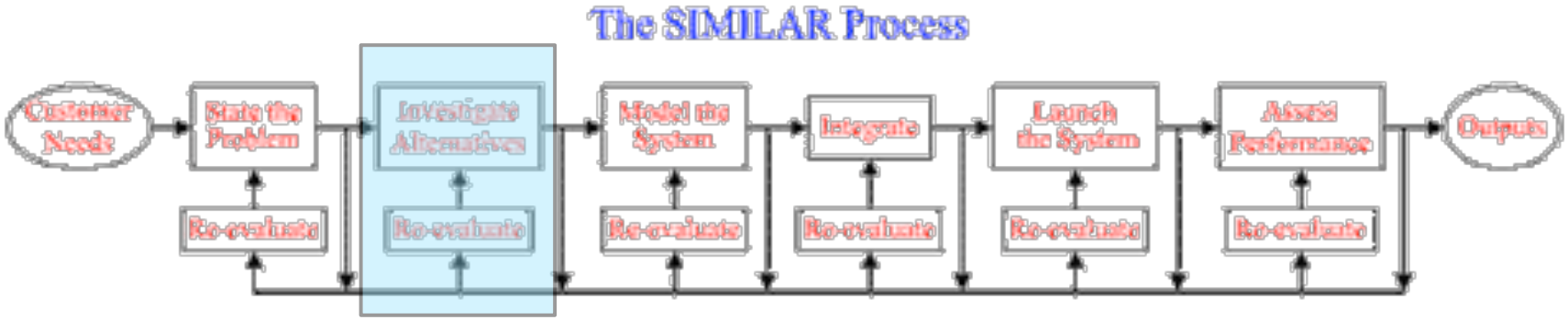
What is Systems Engineering?
A Consensus of the INCOSE Fellows (3)



- State the problem

The problem statement starts with a description of the top-level functions that the system must perform: this might be in the form of a mission statement, a concept of operations or a description of the deficiency that must be ameliorated. Most mandatory and preference requirements should be traceable to this problem statement. Acceptable systems must satisfy all the mandatory requirements. The preference requirements are traded-off to find the preferred alternatives. The problem statement should be in terms of what must be done, not how to do it. The problem statement should express the customer requirements in functional or behavioral terms. It might be composed in words or as a model. Inputs come from end users, operators, maintainers, suppliers, acquirers, owners, regulatory agencies, victims, sponsors, manufacturers and other stakeholders.

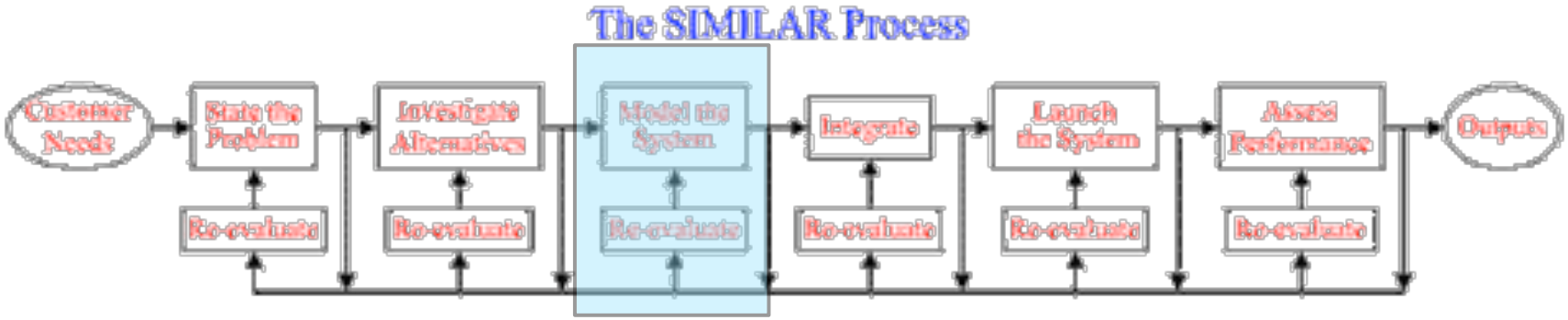
What is Systems Engineering?
A Consensus of the INCOSE Fellows (4)



- Investigate Alternatives

  Alternative designs are created and are evaluated based on performance, schedule, cost and risk figures of merit. No design is likely to be best on all figures of merit, so multicriteria decision-aiding techniques should be used to reveal the preferred alternatives. This analysis should be redone whenever more data are available. For example, figures of merit should be computed initially based on estimates by the design engineers. Then, concurrently, models should be constructed and evaluated; simulation data should be derived; and prototypes should be built and measured. Finally, tests should be run on the real system. Alternatives should be judged for compliance of capability against requirements. For the design of complex systems, alternative designs reduce project risk. Investigating innovative alternatives helps clarify the problem statement.

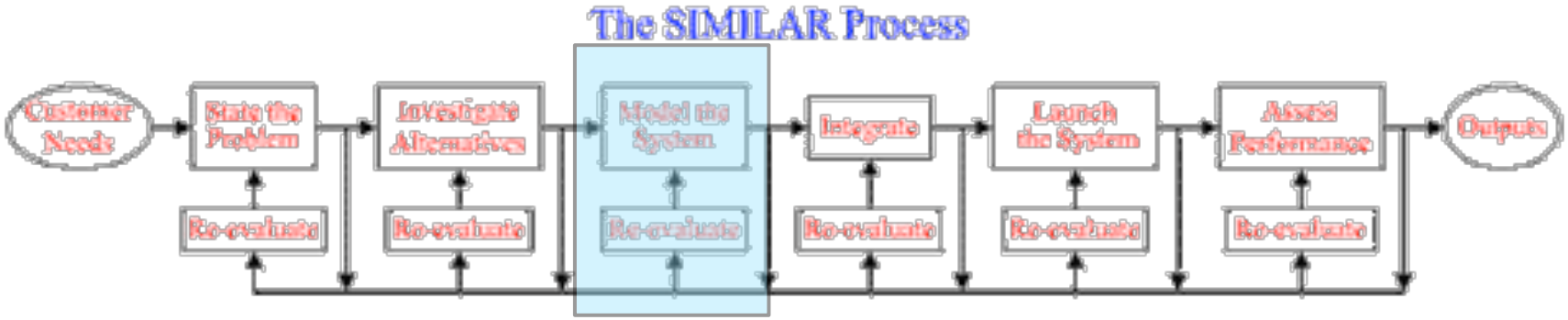
What is Systems Engineering?
A Consensus of the INCOSE Fellows (5)



- Model the system

  Models will be developed for most alternative designs. The model for the preferred alternative will be expanded and used to help manage the system throughout its entire life cycle. Many types of system models are used, such as physical analogs, analytic equations, state machines, block diagrams, functional flow diagrams, object-oriented models, computer simulations and mental models. Systems Engineering is responsible for creating a product and also a process for producing it. So, models should be constructed for both the product and the process. Process models allow us, for example, to study scheduling changes, create dynamic PERT charts and perform sensitivity analyses to show the effects of delaying or accelerating certain subprojects. ...

What is Systems Engineering?
A Consensus of the INCOSE Fellows (6)

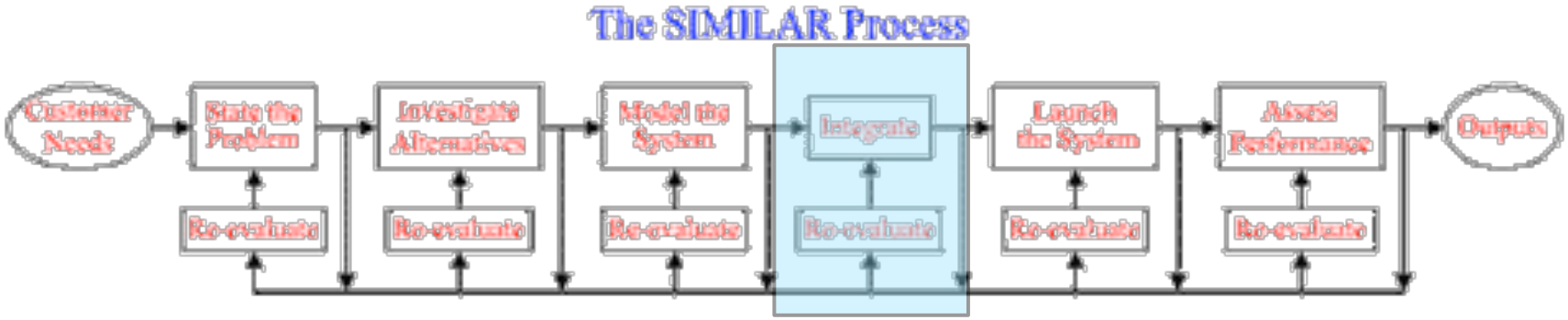


- Model the system

  ... Running the process models reveals bottlenecks and fragmented activities, reduces cost and exposes duplication of effort. Product models help explain the system. These models are also used in tradeoff studies and risk management.

  As previously stated, the Systems Engineering Process is not sequential: it is parallel and iterative. This is another example: models must be created before alternatives can be investigated.

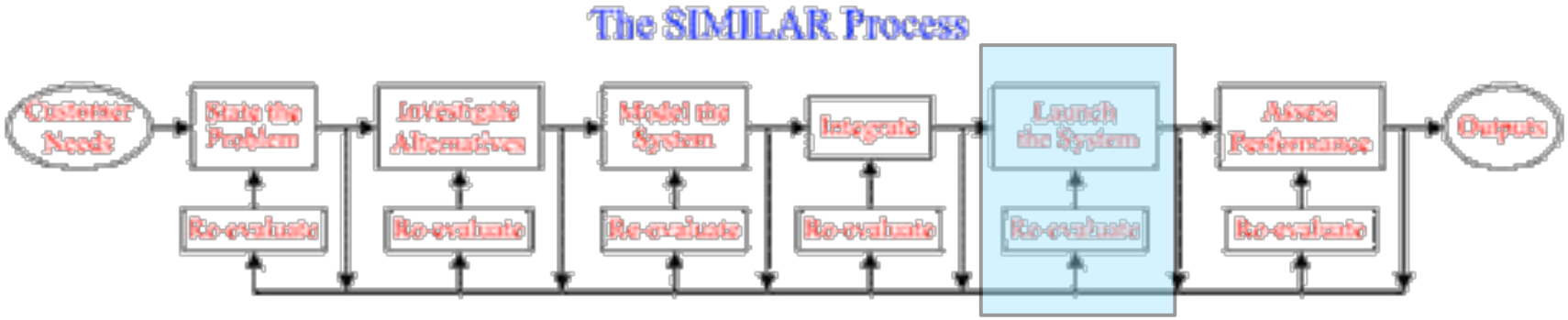
What is Systems Engineering?
A Consensus of the INCOSE Fellows (7)



- Integrate

No man is an island. Systems, businesses and people must be integrated so that they interact with one another. Integration means bringing things together so they work as a whole. Interfaces between subsystems must be designed. Subsystems should be defined along natural boundaries. Subsystems should be defined to minimize the amount of information to be exchanged between the subsystems. Well-designed subsystems send finished products to other subsystems. Feedback loops around individual subsystems are easier to manage than feedback loops around interconnected subsystems. Processes of co-evolving systems also need to be integrated. The consequence of integration is a system that is built and operated using efficient processes.

What is Systems Engineering?
A Consensus of the INCOSE Fellows (8)

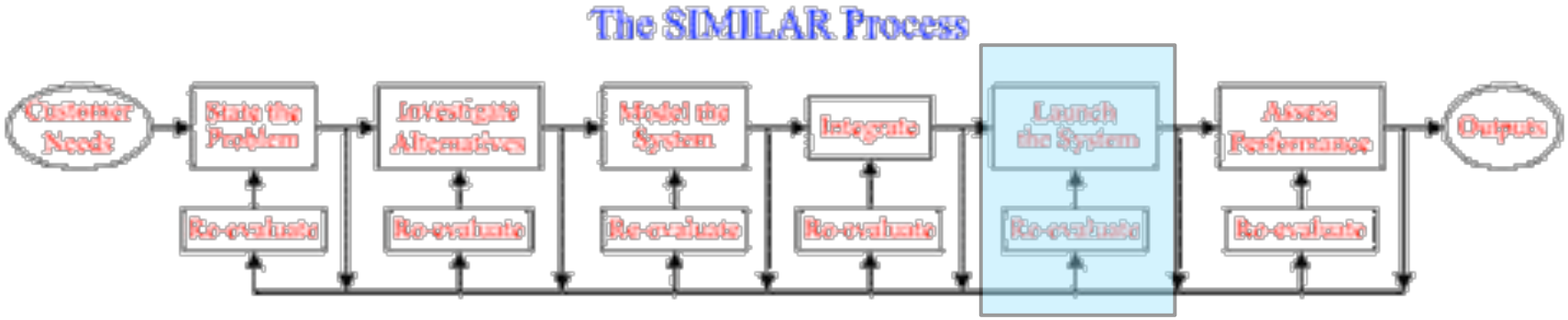


- Launch the system

  Launching the system means running the system and producing outputs. In a manufacturing environment this might mean buying commercial off the shelf hardware or software, or it might mean actually making things. Launching the system means allowing the system do what it was intended to do. This also includes the system engineering of deploying multi-site, multi-cultural systems.

  ...

What is Systems Engineering?
A Consensus of the INCOSE Fellows (9)

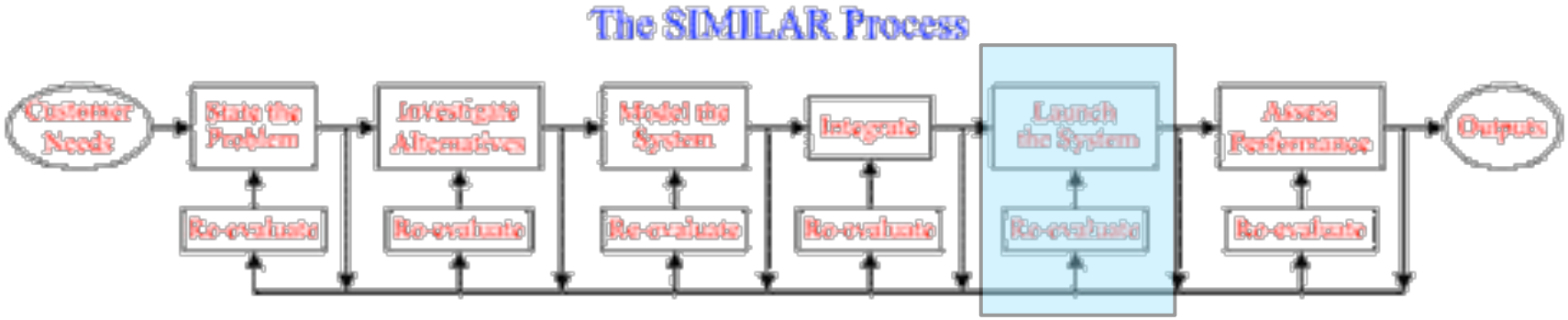


- Launch the system

...

This is the phase where the preferred alternative is designed in detail; the parts are built or bought (COTS), the parts are integrated and tested at various levels leading to the certified product. In parallel, the processes necessary for this are developed – where necessary - and applied so that the product can be produced. In designing and producing the product, due consideration is given to its interfaces with operators (humans, who will need to be trained) and other systems with which the product will interface. In some instances, this will cause interfaced systems to co-evolve. The process of designing and producing the system is iterative as new knowledge developed along the way can cause a re-consideration and modification of earlier steps.

What is Systems Engineering?
A Consensus of the INCOSE Fellows (10)

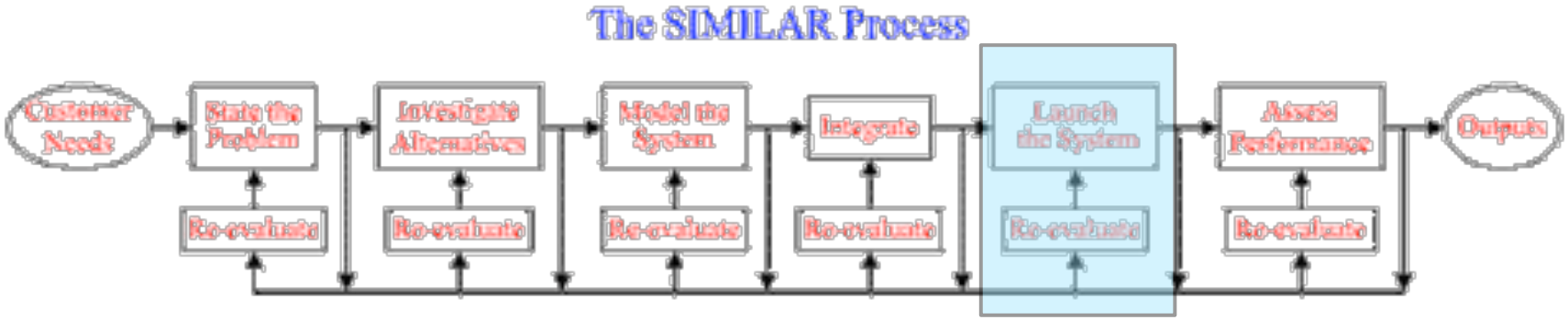


- Launch the system

  ...

  The systems engineers' products are a mission statement, a requirements document including verification and validation, a description of functions and objects, figures of merit, a test plan, a drawing of system boundaries, an interface control document, a listing of deliverables, models, a sensitivity analysis, a tradeoff study, a risk analysis, a life cycle analysis and a description of the physical architecture. The requirements should be validated (Are we building the right system?) and verified (Are we building the system right?). ...

What is Systems Engineering?
A Consensus of the INCOSE Fellows (11)

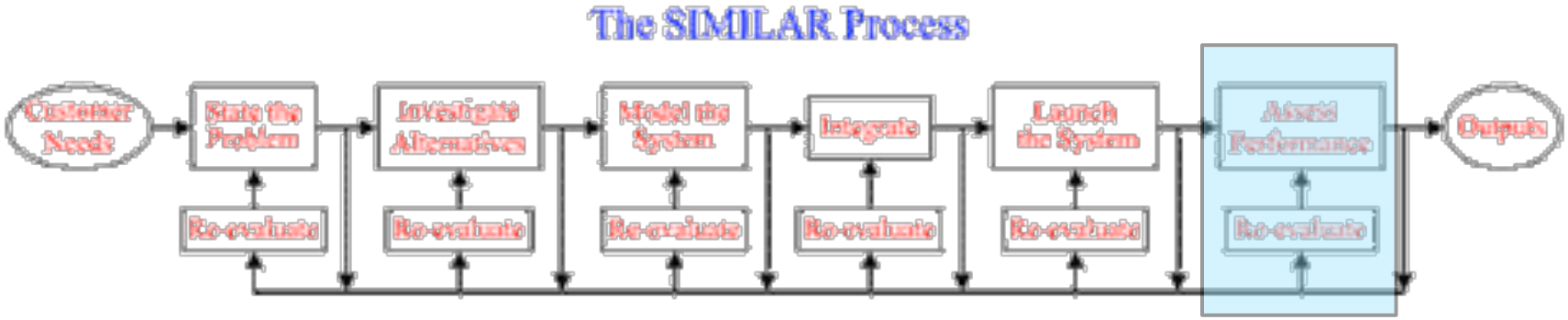


- Launch the system

  ...

  ... The system functions should be mapped to the physical components. The mapping of functions to physical components can be one to one or many to one. But if one function is assigned to two or more physical components, then a mistake might have been made and it should be investigated. One valid reason for assigning a function to more than one component would be that the function is preformed by one component in a certain mode and by another component in another mode. Another would be deliberate redundancy to enhance reliability, allowing one portion of the system to take on a function if another portion fails to do so.

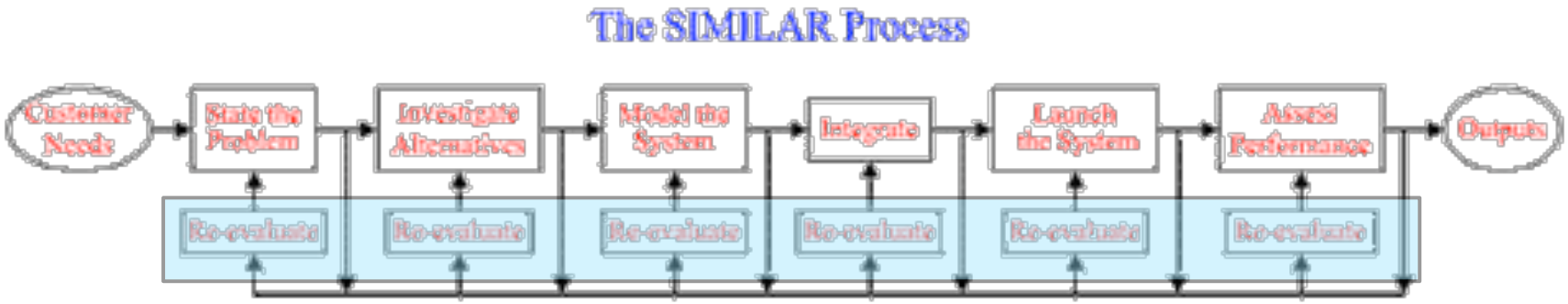
What is Systems Engineering?
A Consensus of the INCOSE Fellows (12)



- Assess performance

Figures of merit, technical performance measures and metrics are all used to assess performance. Figures of merit are used to quantify requirements in the tradeoff studies. They usually focus on the product. Technical performance measures are used to mitigate risk during design and manufacturing. Metrics (including customer satisfaction comments, productivity, number of problem reports, or whatever you feel is critical to your business) are used to help manage a company's processes. Measurement is the key. If you cannot measure it, you cannot control it. If you cannot control it, you cannot improve it. Important resources such as weight, volume, price, communications bandwidth and power consumption should be managed. Each subsystem is allocated a portion of the total budget and the project manger is allocated a reserve. These resource budgets are managed throughout the system life cycle.

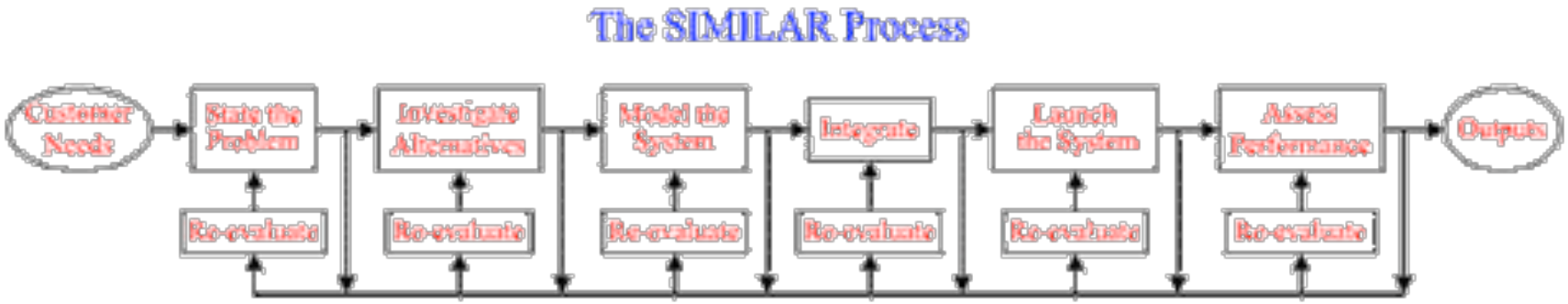
What is Systems Engineering?
A Consensus of the INCOSE Fellows (13)



- Re-evaluate

  Re-evaluate is arguably the most important of these functions. For a century, engineers have used feedback to help control systems and improve performance. It is one of the most fundamental engineering tools. Re-evaluation should be a continual process with many parallel loops. Re-evaluate means observing outputs and using this information to modify the system, the inputs, the product or the process. Figure 1 summarizes the Systems Engineering Process. This figure clearly shows the distributed nature of the Re-evaluate function in the feedback loops. However, all of these loops will not always be used. The particular loops that are used depend on the particular problem being solved.

What is Systems Engineering?
A Consensus of the INCOSE Fellows (14)



- Variations

  Like all processes, the Systems Engineering process at any company should be documented, measurable, stable, of low variability, used the same way by all, adaptive, and tailorable! This may seem like a contradiction. And perhaps it is. But one size does not fit all. The above description of the Systems Engineering process is just one of many that have been proposed. Some are bigger, some are smaller. But most are similar to this one.

What is Systems Engineering?
A Consensus of the INCOSE Fellows (15)

- This is the end of the consensus. What follows are comments and additions by individual INCOSE fellows.
- Commentary by Brian Mar
  - Most systems engineers accept the following basic core concepts:
  - Understand the whole problem before you try to solve it
  - Translate the problem into measurable requirements
  - Examine all feasible alternatives before selecting a solution
  - Make sure you consider the total system life cycle. The birth to death concept extends to maintenance, replacement and decommission. If these are not considered in the other tasks, major life cycle costs can be ignored.
  - Make sure to test the total system before delivering it.
  - Document everything.
- Commentary by George Friedman
  - The seven-task process defined above is an excellent representation of systems engineering as is presently practiced and should serve to avoid most of the problems that have plagued the development of large, complex systems in the past. Yet, in order to advance as a discipline and as a profession, systems engineering must grow from problem minimization to design optimization by the integration of these tasks into a more unified theory. Elements of this theory include quantitative risk management, decision-based design and the management of multidimensional mathematical models. As the field advances in these and similar directions, it will earn additional respect by industry, government and academia.

GfSE e.V. und Systems Engineering (1)

- Systems Engineering umfasst die wesentlichen Ingenieurtätigkeiten, die zur Entwicklung komplexer Produkte notwendig sind. Dazu gehören auch Aufgaben wie
  - Systemanalyse (System Analysis)
  - Systemarchitekturentwicklung (System Architecture Design)
  - Systementwicklung (System Design)
  - Anforderungsentwicklung (Requirements Engineering / Requirements Management)
  - Konfigurationsmanagement (Configuration Management)
  - Technologieentwicklung und -management (Technology Management, Obsolescense Management)
  - Verifikation und Validierung (V&V)

GfSE e.V. und Systems Engineering (2)

- Um eine Vielzahl von Funktionen erfolgreich zu einem beherrschbaren und kostengünstigen System zu integrieren, müssen unterschiedlichste Anforderungen über den gesamten Systemlebenszyklus hinweg berücksichtigt werden. Dazu muss der multi- und interdisziplinären Natur der Systemgestaltung zum Beispiel durch interdisziplinäre Entwickungsteams (Integrated Product Development Teams) Rechnung getragen werden.

- Projektmanagement und Systems Engineering gehen Hand-in-Hand, da es eine wesentliche Leistung des Systems Engineering ist, dem Projektmanager belastbare Planungsgrundlagen an die Hand zu geben.
- GfSE ist die Systems Engineering Organisation für die deutschsprachigen Länder.

Was ist Systems Engineering? (1)

Dies ist sicher eine der ersten Fragen, die sich jedem stellt, der anfängt sich mit Systems Engineering und der GfSE auseinander zu setzen.

Ursprünglich kommt der Begriff aus der Verteidigungstechnik und der Raumfahrt. Getrieben durch sehr vielfältige und teilweise auch schwere jedoch nötige Prozesse hat der Begriff Systems Engineering das Stigma schwerfällig und zu bürokratisch zu sein. Viele beziehen das auf immense Kosten und viel Personal. Das mag in der Vergangenheit sicherlich so gewesen sein und der Bedarf für andere Industrien nicht nachvollziehbar. Nur die Raumfahrt, die ihre Produkte nicht begleiten konnte, musste schon bei einer Produktentwicklung die Zuverlässigkeit und Funktionalität sicherstellen und bei der Verteidigungstechnik hat der Kunde auf diese Informationen bestanden. Bei einem kommerziellen Ansatz scheinen diese Argumente nicht zu greifen.

Lesen Sie kurz folgende Fragen und kommen Sie unter Umständen zu einem anderen Schluss?

1. Hat sich der Beruf des Ingenieurs in den letzten Jahrzehnten geändert?
2. Müssen Sie, als Ingenieur, nicht in einer fest vorgegebenen Zeit ein Produkt auf den Markt bringen? Muss das Produkt nicht den Kunden/Markt befriedigen?
3. Müssen Sie nicht innerhalb der abgeschätzten Kosten bleiben? Sollten Sie nicht wenigstens den Business Case verstehen?

Was ist Systems Engineering? (2)

4. Haben der Einfluss von Normen und die Umweltanforderungen an die technische Lösung nicht zugenommen? Kennen Sie noch alle Anforderungen und die Ihrer Kollegen?
5. Entwickeln und produzieren Sie noch alles in der eigenen Firma? Hat sich nicht die Anzahl der Kaufteile und Zulieferteile und damit der Schnittstellen erhöht?
6. Sind Ihre Lieferanten nur in der Nachbarschaft, oder haben Sie internationale Kontakte als Ingenieur?
7. Fordert Ihr Kunde bestimmte Prozesse und Schnittstellen zu Softwaretools?
8. Müssen Sie als Ingenieur auch ein Risikoabschätzen?
9. Versuchen Sie nicht jedem Kunden eine individuelle Lösung auf Basis von Plattformen und Modulen anzubieten? Haben Sie noch jedes ausgeliefertes Produkt mit den Bauteilen und Materialien unter Kontrolle?
10. Müssen Sie sich als Entwicklungsingenieur nicht schon am Anfang der Idee mit der Wartung oder der Entsorgung beschäftigen?
11. Sie versuchen über Modellbildung die Anzahl und die Kosten der Tests vor der Einführung zu reduzieren?

Was ist Systems Engineering? (3)

Falls Sie sich bei vielen Fragen angesprochen fühlen, so mögen Sie das in Ihrem Umfeld nicht Systems Engineering nennen – wir fassen die benötigten Prozesse und Methoden jedoch unter diesem Begriff zusammen.

Die Änderung der Randbedingungen und Umgebungseinflüsse auf ein Produkt – oder in unserer Sprache „System“ – zu entwickeln sind komplexer geworden. Der Begriff System umfasst nicht nur das Produkt selbst, sondern alle nötigen Prozesse, damit der Kunde das Produkt bedienen kann. Der Zuwachs aus anderen Industrien zeigt immer wieder den Bedarf sich mit den Prozessen und Methoden auseinanderzusetzen und sie für diese Bedürfnisse anzupassen.

Entdecken Sie selbst den gemeinsamen Nenner und spiegeln Sie die Informationen an Ihrem eigenen Erfahrungshintergrund in unserer Gemeinschaft.

Systems Engineering Definitionen INCOSE

Systems Engineering is an interdisciplinary approach and means to enable the realisation of successful systems. It focuses on defining customer needs and required functionality early in the development cycle, documenting requirements, then proceeding with design synthesis and system validation while considering the complete problem.

- Alles hängt mit allem irgendwo irgendwie zusammen

- Alles hängt mit allem irgendwo irgendwie zusammen

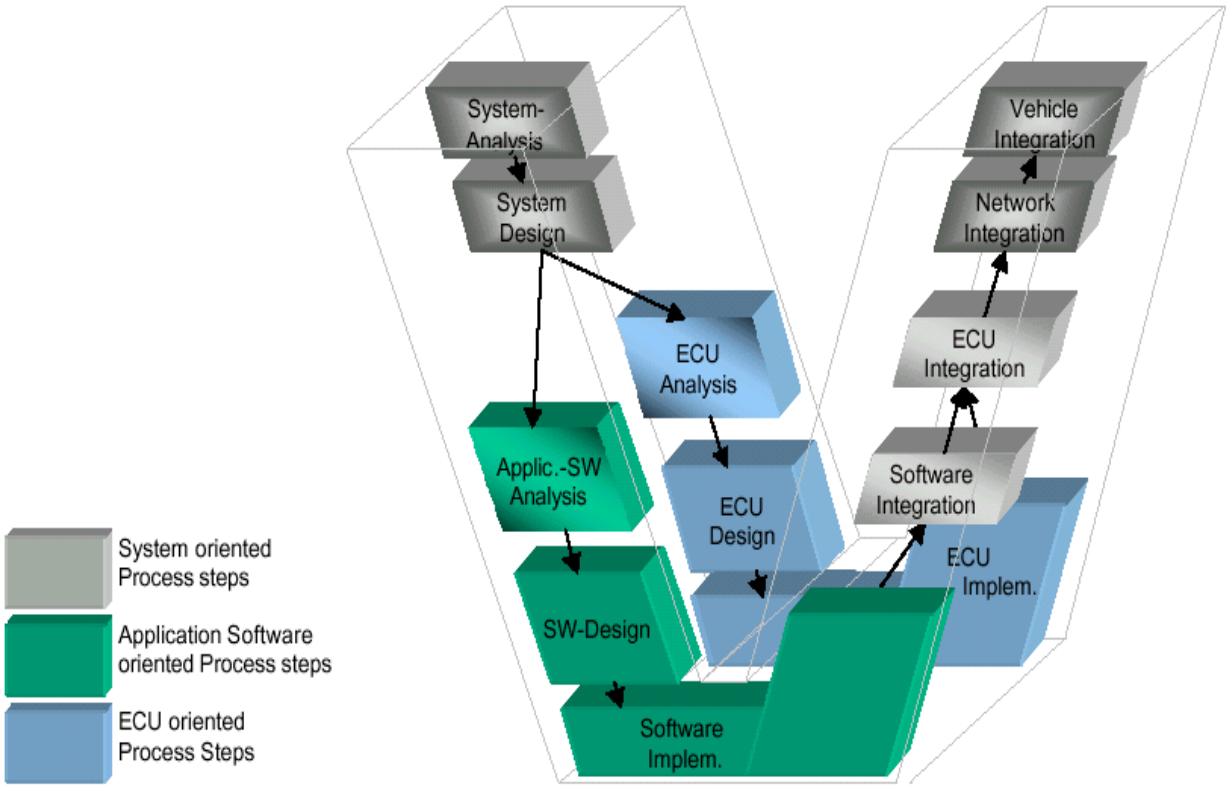
- Alles hängt mit allem irgendwo irgendwie zusammen

- „2 und 2 ist irgendwas, und 4 und 4 ist auch irgendwas, aber was anderes. Das macht alles so kompliziert.“

Systems Engineering (3)

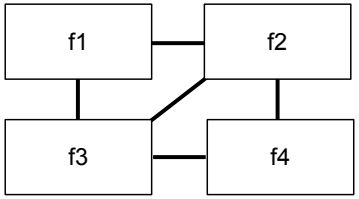
Fachdisziplinen innerhalb Systems Engineering

- Software-Entwicklung
- Hardware-Entwicklung
- Sensorik
- Aktuatorik
- ...



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

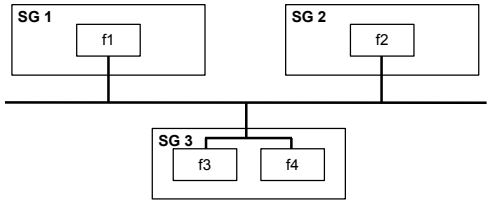
Analyse der Benutzeranforderungen und Spezifikation der logischen Systemarchitektur



Logische Systemarchitektur

Akzeptanz- und Systemtest

Analyse der logischen Systemarchitektur und Spezifikation der technischen Systemarchitektur



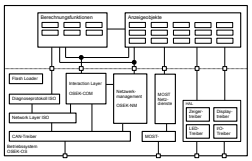
Technische Systemarchitektur

Kalibrierung

Integrationstest des Systems

Integration der System-Komponenten

Analyse der Software-Anforderungen und Spezifikation der technischen Softwarearchitektur


Software-Architektur

Integrationstest der Software

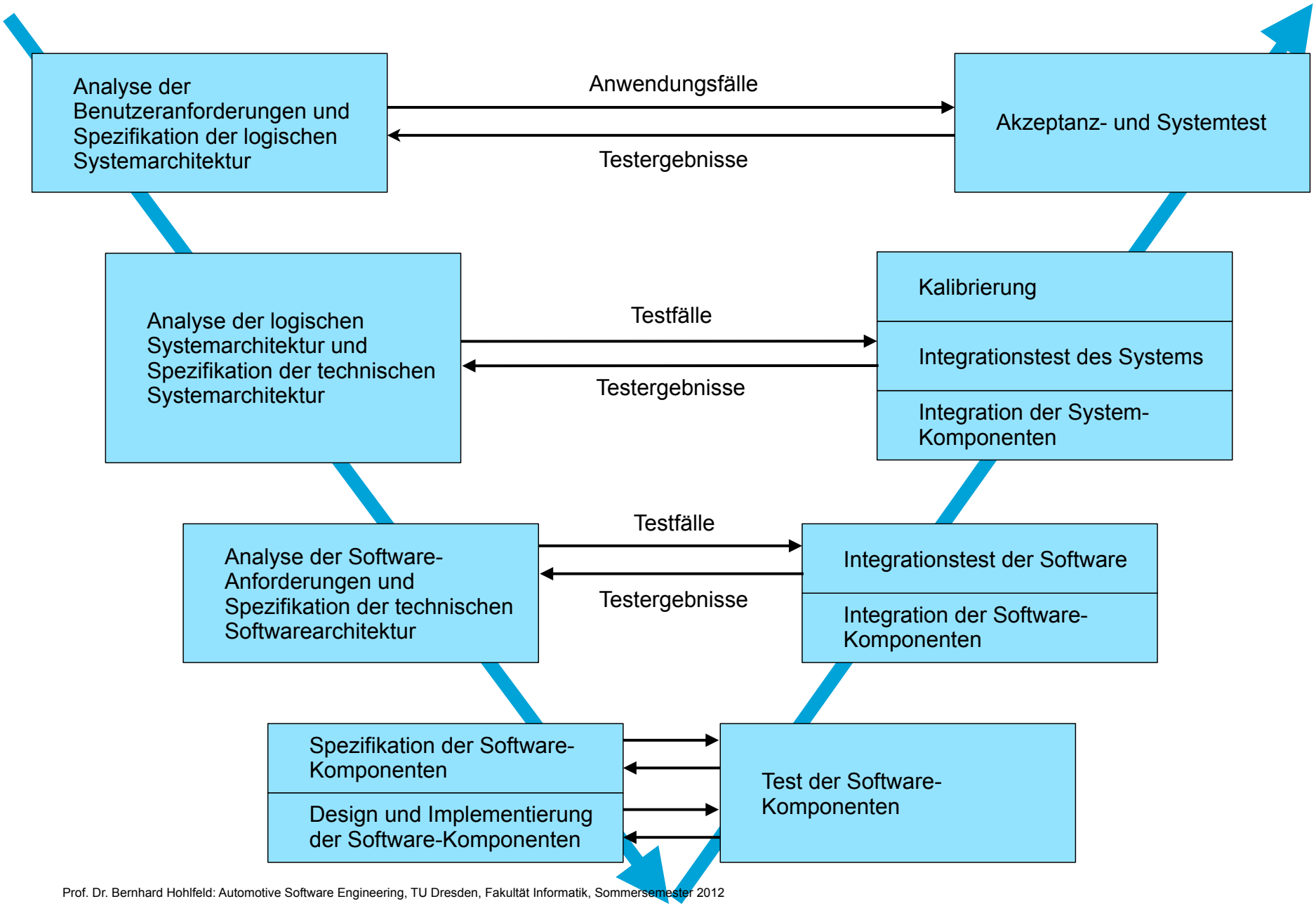
Integration der Software-Komponenten

Spezifikation der Software-Komponenten

Design und Implementierung der Software-Komponenten

Test der Software-Komponenten

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



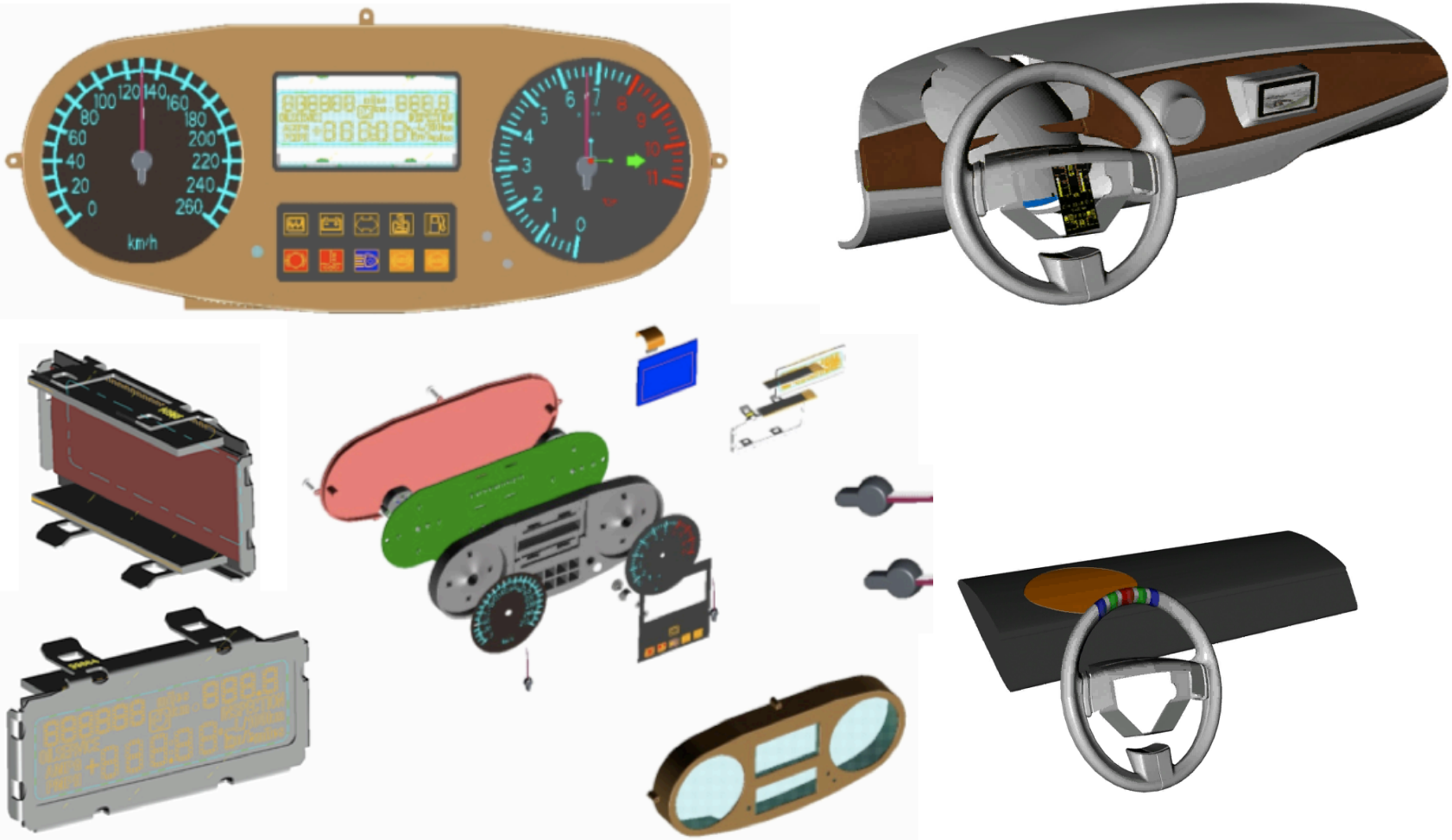
6. SW-Entwicklung / 1. Kernprozess
Kernprozess zur Entwicklung von elektronischen Systemen und Software

1. Grundbegriffe
2. **Entwicklungsobjekt: Kombiinstrument**
3. Analyse der Benutzeranforderungen und Spezifikation der logischen Systemarchitektur
4. Analyse der logischen Systemarchitektur und Spezifikation der technischen Systemarchitektur
5. Analyse der Software-Anforderungen und Spezifikation der technischen Softwarearchitektur
6. Spezifikation der Software-Komponenten
7. Design und Implementierung der Software-Komponenten
8. Test der Software-Komponenten
9. Integration der Software-Komponenten
10. Integrationstest der Software
11. Integration der System-Komponenten
12. Integrationstest des Systems
13. Kalibrierung
14. Akzeptanz- und Systemtest

Kombiinstrument

- Tachometer
- Odometer
- Drehzahlmesser
- Tankanzeige
- Kühlmitteltemperaturanzeige
- Kontrollleuchten
- ...

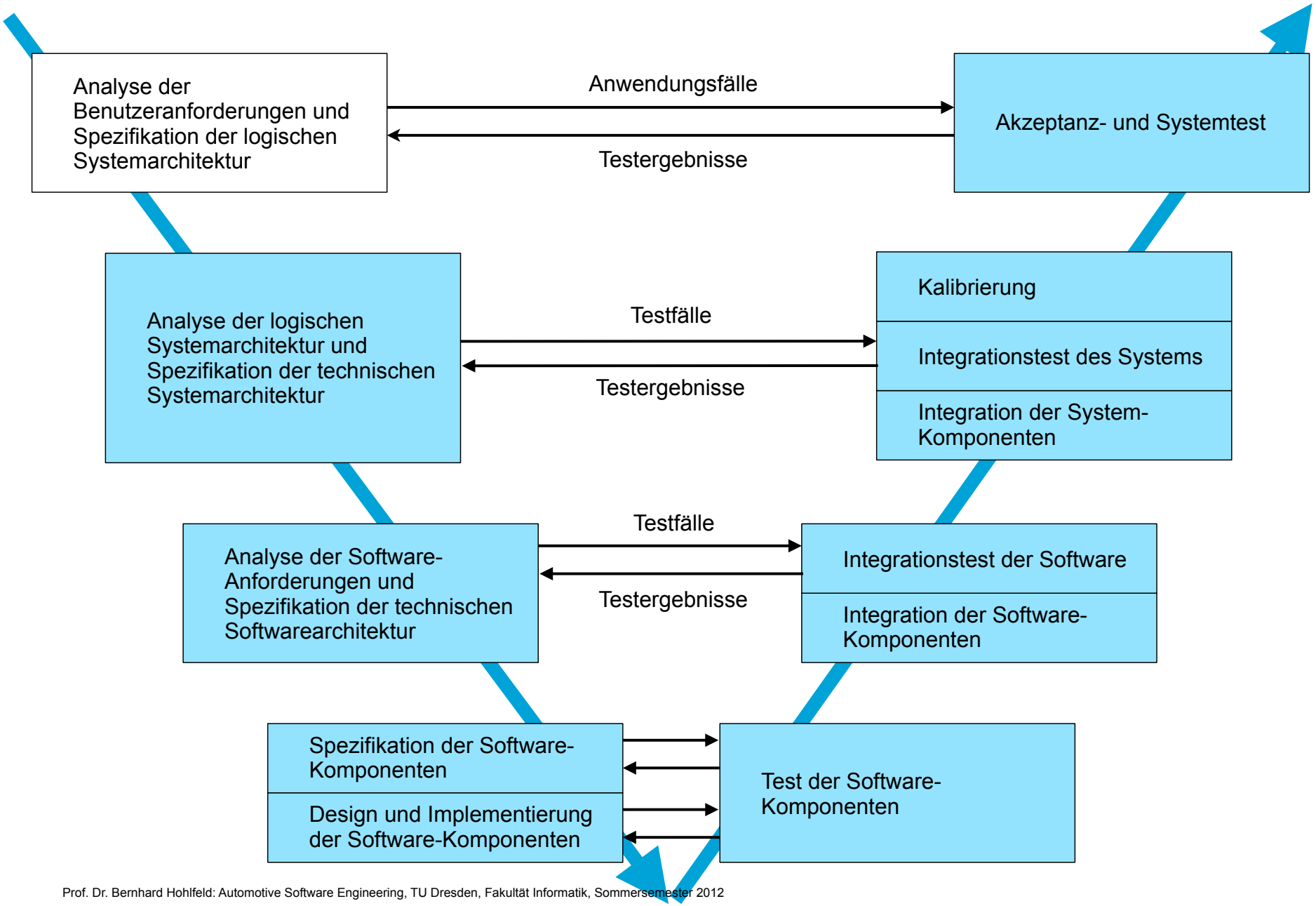
Entwicklungsobjekt: Kombiinstrument



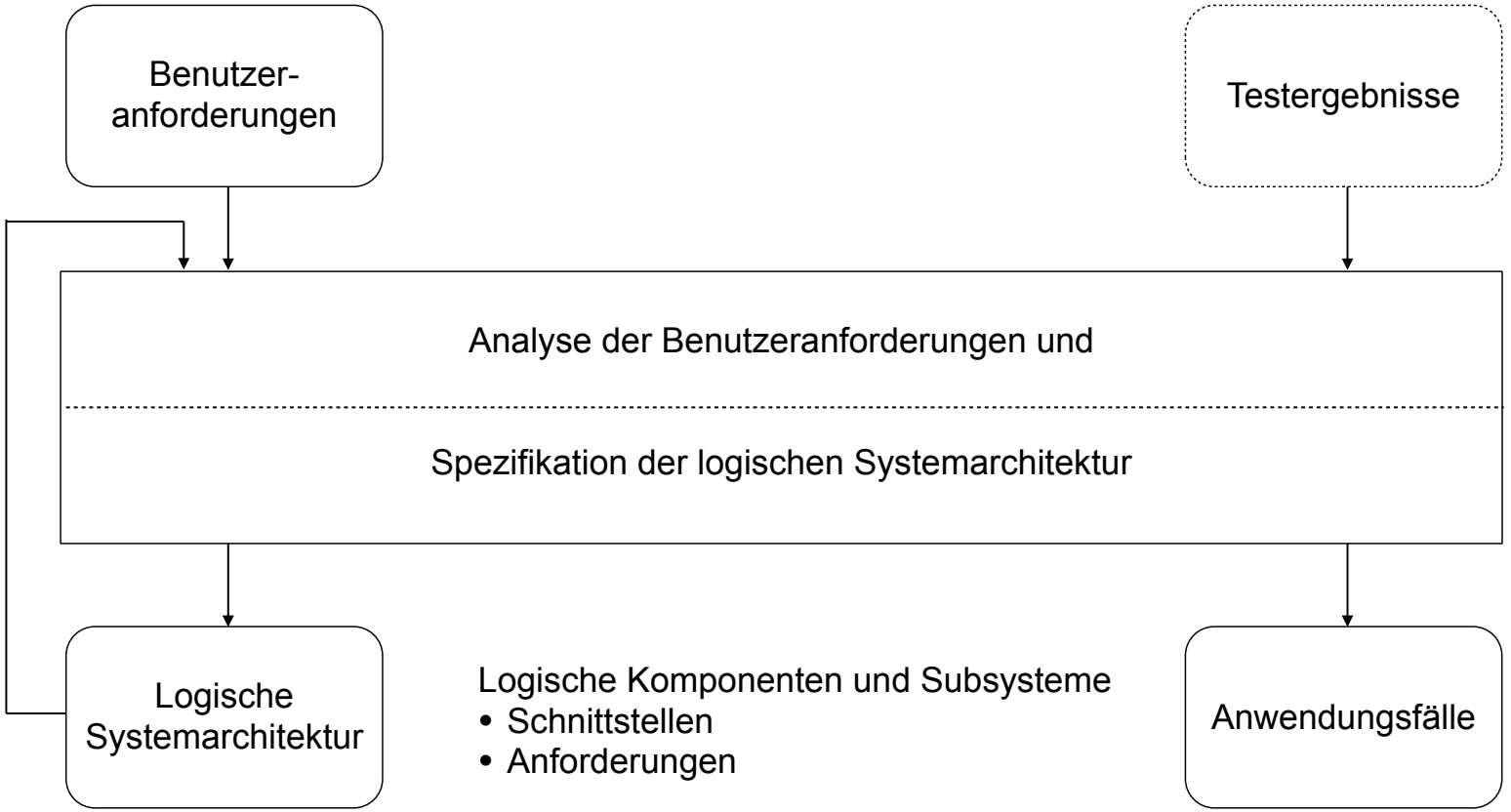
6. SW-Entwicklung / 1. Kernprozess
Kernprozess zur Entwicklung von elektronischen Systemen und Software

1. Grundbegriffe
2. Entwicklungsobjekt: Kombiinstrument
3. **Analyse der Benutzeranforderungen und Spezifikation der logischen Systemarchitektur**
4. Analyse der logischen Systemarchitektur und Spezifikation der technischen Systemarchitektur
5. Analyse der Software-Anforderungen und Spezifikation der technischen Softwarearchitektur
6. Spezifikation der Software-Komponenten
7. Design und Implementierung der Software-Komponenten
8. Test der Software-Komponenten
9. Integration der Software-Komponenten
10. Integrationstest der Software
11. Integration der System-Komponenten
12. Integrationstest des Systems
13. Kalibrierung
14. Akzeptanz- und Systemtest

Analyse der Benutzeranforderungen und Spezifikation der logischen Systemarchitektur



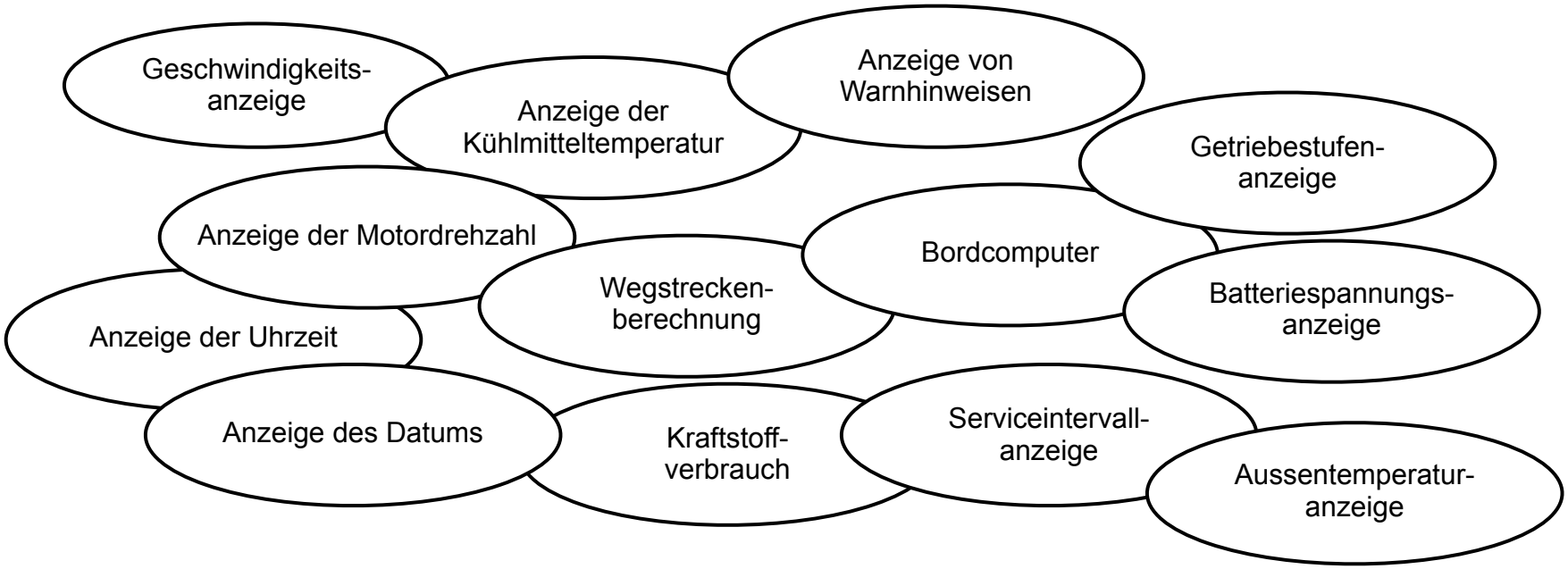
Analyse der Benutzeranforderungen und Spezifikation der logischen Systemarchitektur (Nach Schäuffele, Zurawka)

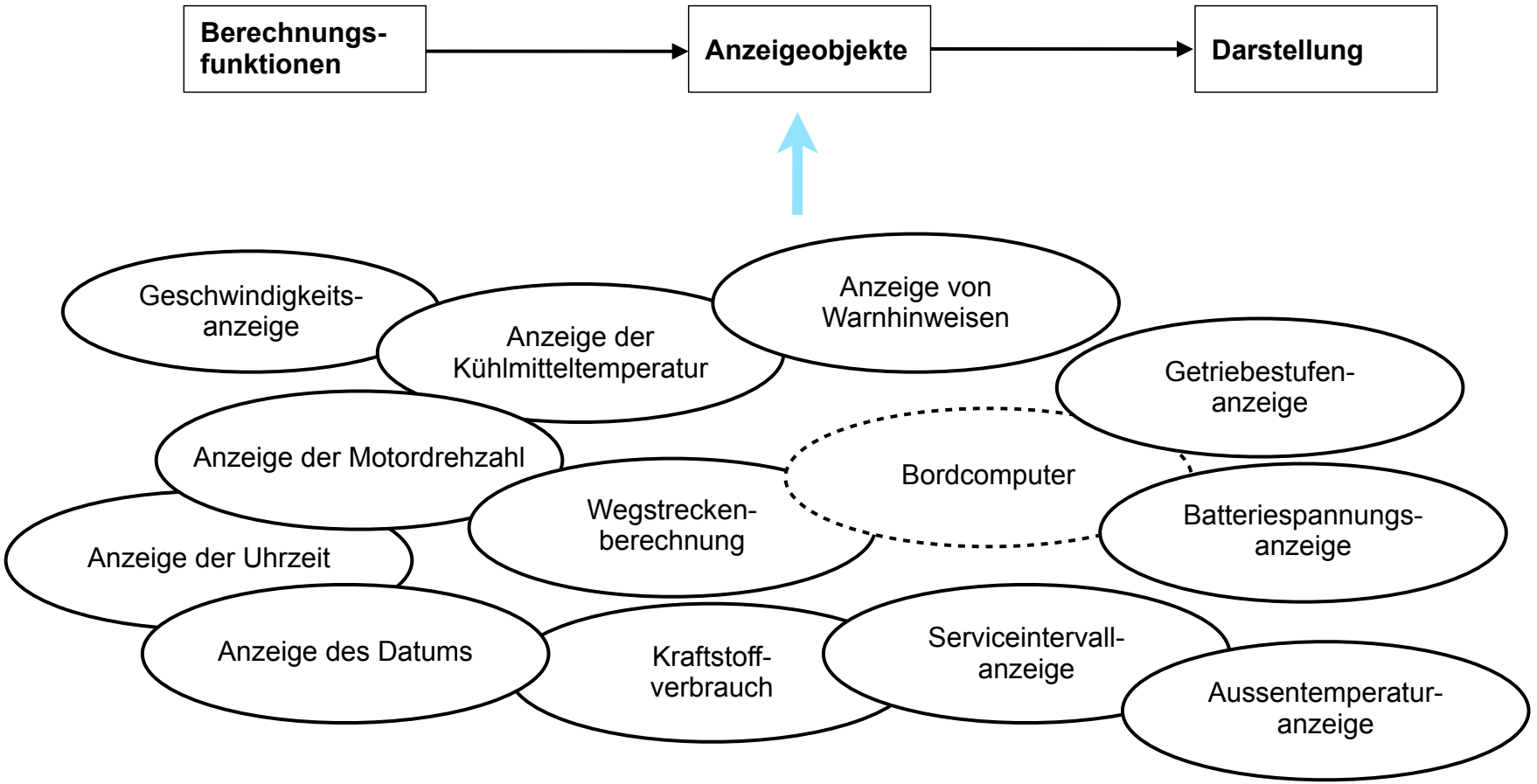


Analyse der Benutzeranforderungen und Spezifikation der logischen Systemarchitektur

- Analyse der Benutzeranforderungen
  - Strukturierungsprozess für die Anforderungen und Randbedingungen aus Sicht der Benutzer in der frühen Phase der Systementwicklung
  - Logische Komponenten und Subsysteme definieren
  - Funktionen, Anforderungen und Schnittstellen festlegen
  - Anwendungsfälle für die Funktionen als Basis für den Systemtest festlegen
- Spezifikation der logischen Systemarchitektur
  - Abstrakte Lösung
  - Bindeglied zwischen Benutzeranforderungen und technischer Systemarchitektur
  - Modellbasierte Darstellung (Blockdiagramme, Zustandsautomaten)
  - Schrittweise Zerlegung der Funktionen

Kombiinstrument: Benutzeranforderungen

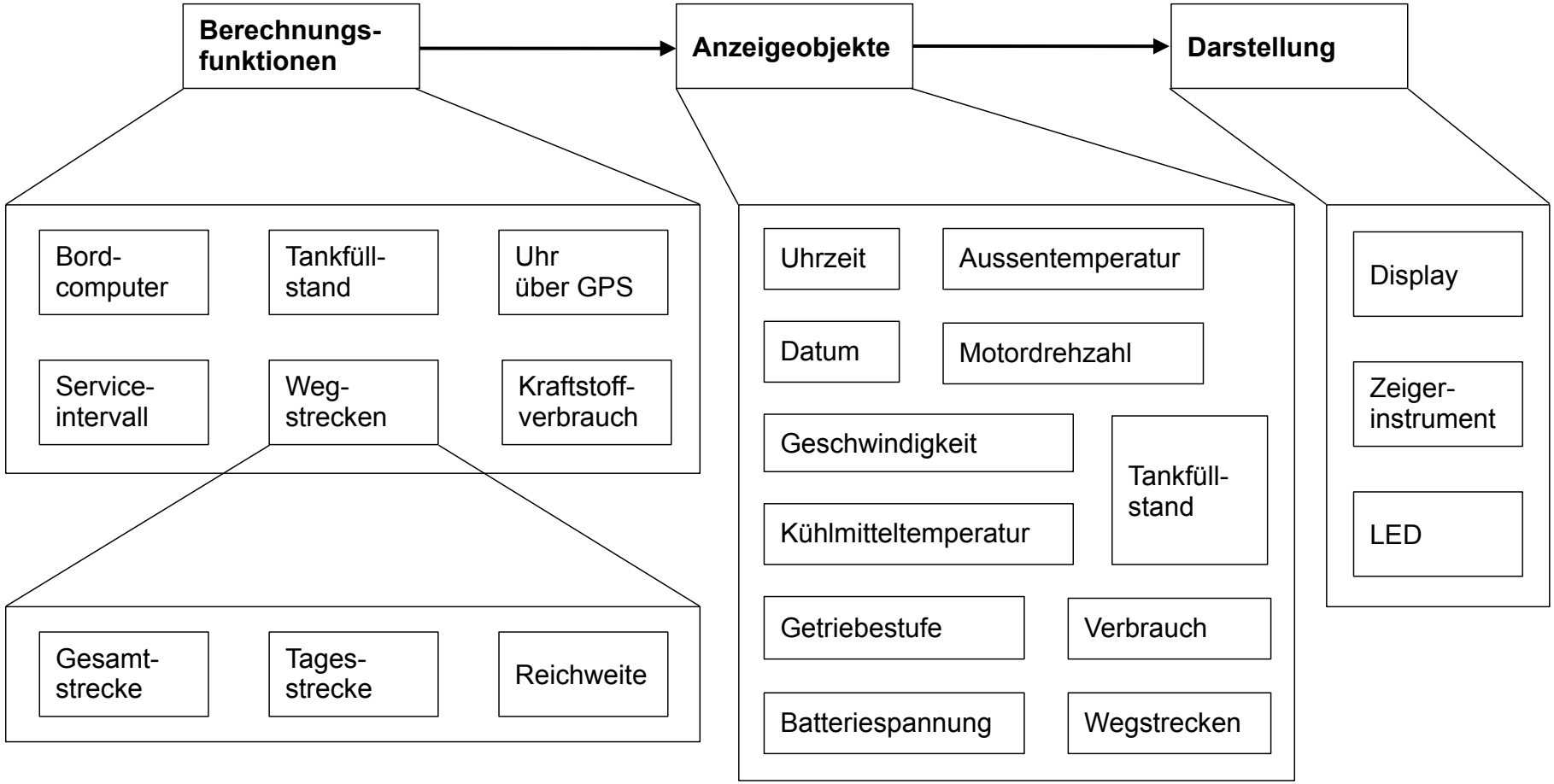




Logische Systemarchitektur des Kombiinstruments (1)

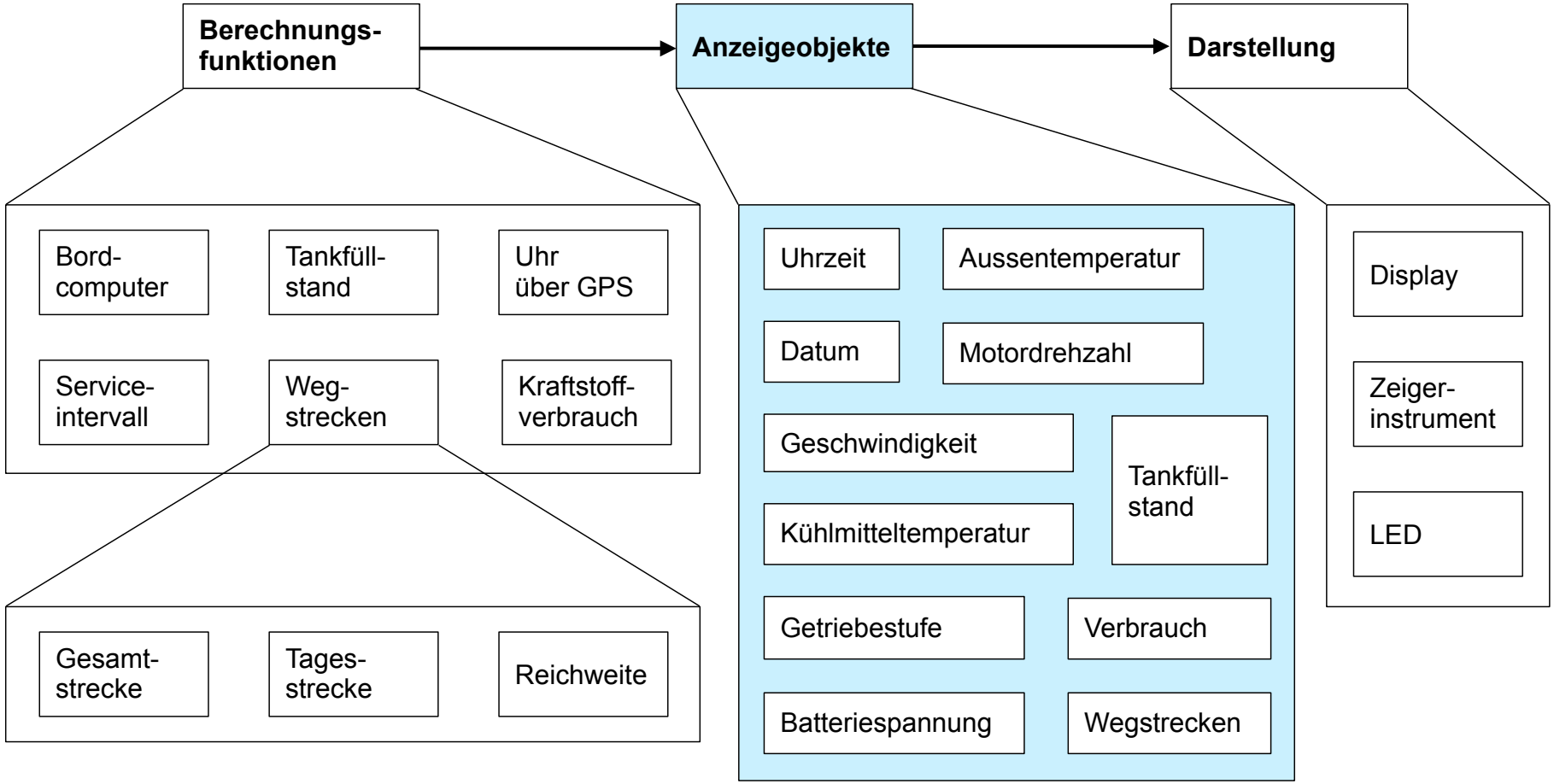
(Nach Schäuffele, Zurawka)

Analyse der Benutzeranforderungen und Spezifikation der logischen Systemarchitektur



Logische Systemarchitektur des Kombiinstruments (2)

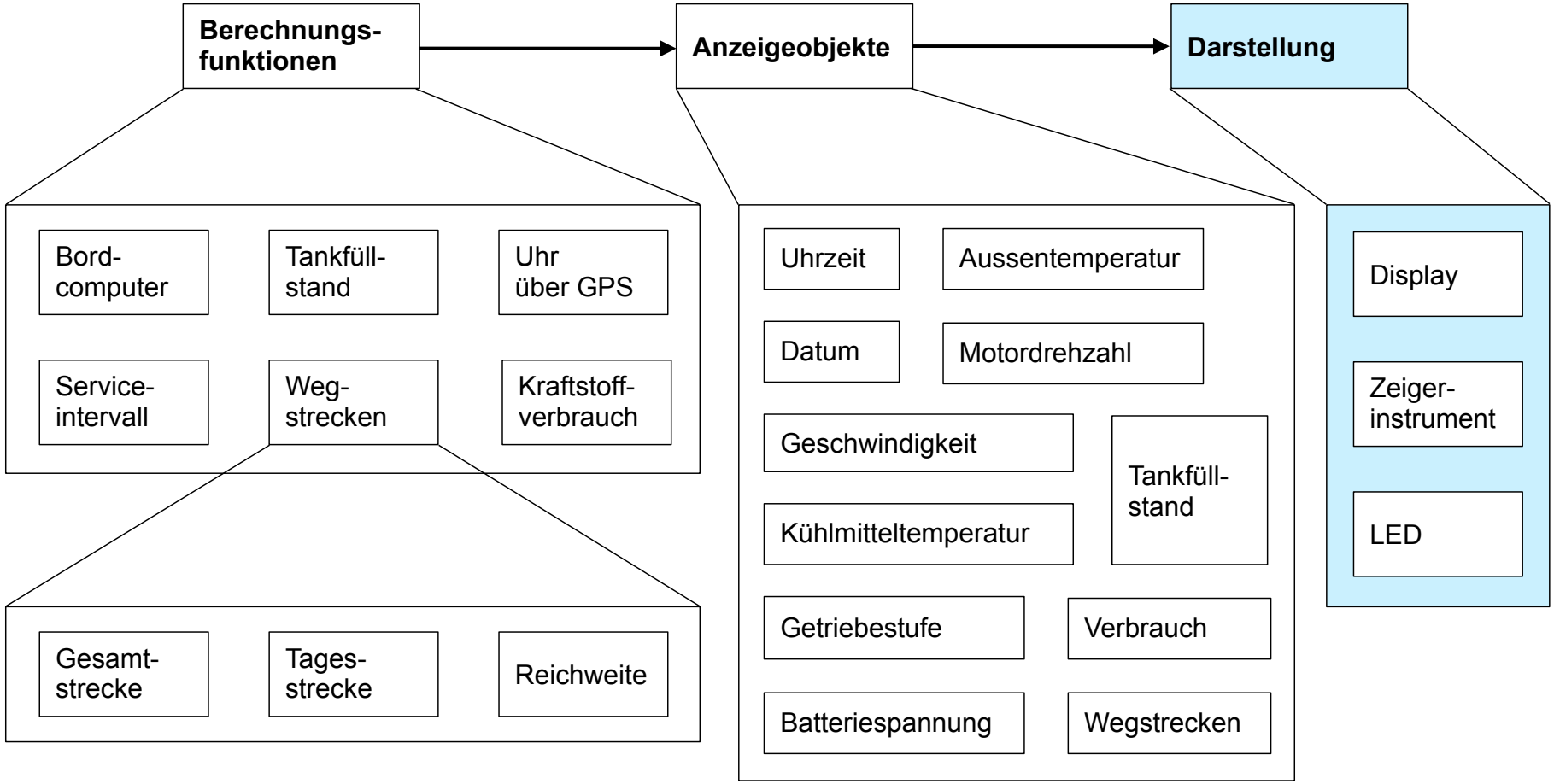
(Nach Schäuffele, Zurawka)



Logische Systemarchitektur des Kombiinstruments (3)

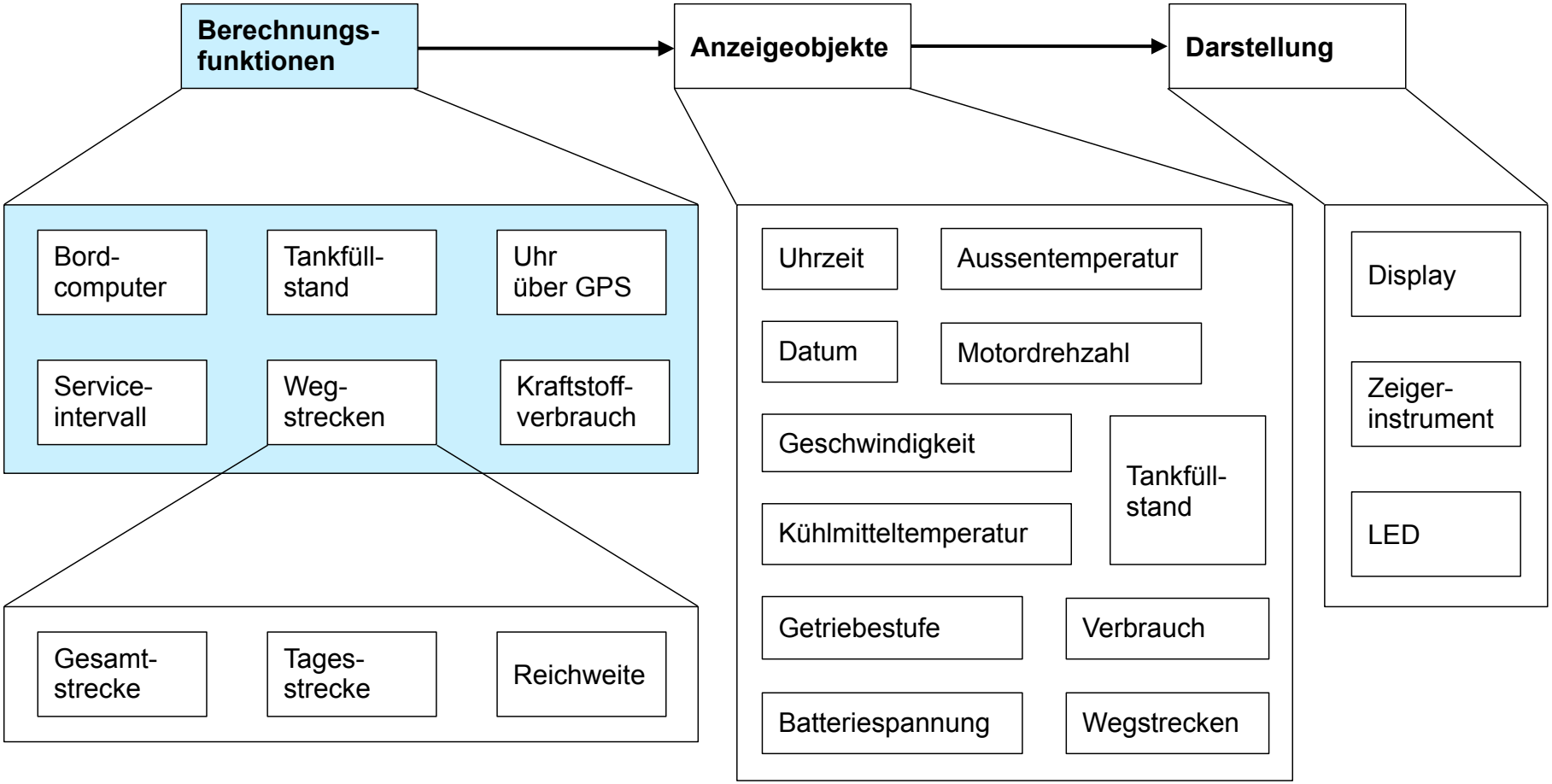
(Nach Schäuffele, Zurawka)

Analyse der Benutzeranforderungen und Spezifikation der logischen Systemarchitektur



Logische Systemarchitektur des Kombiinstruments (4)

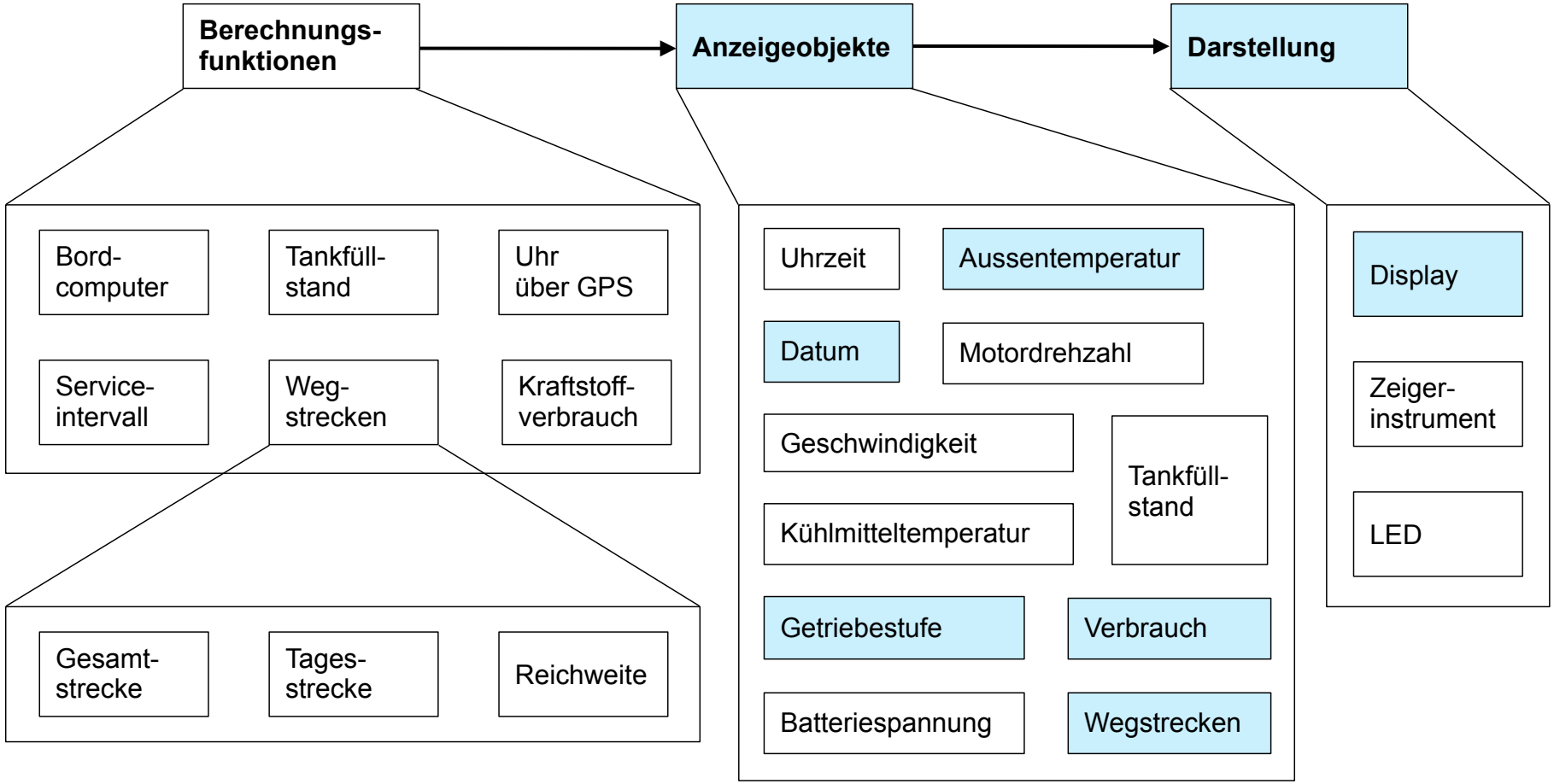
(Nach Schäuffele, Zurawka)



Logische Systemarchitektur des Kombiinstruments (5)

(Nach Schäuffele, Zurawka)

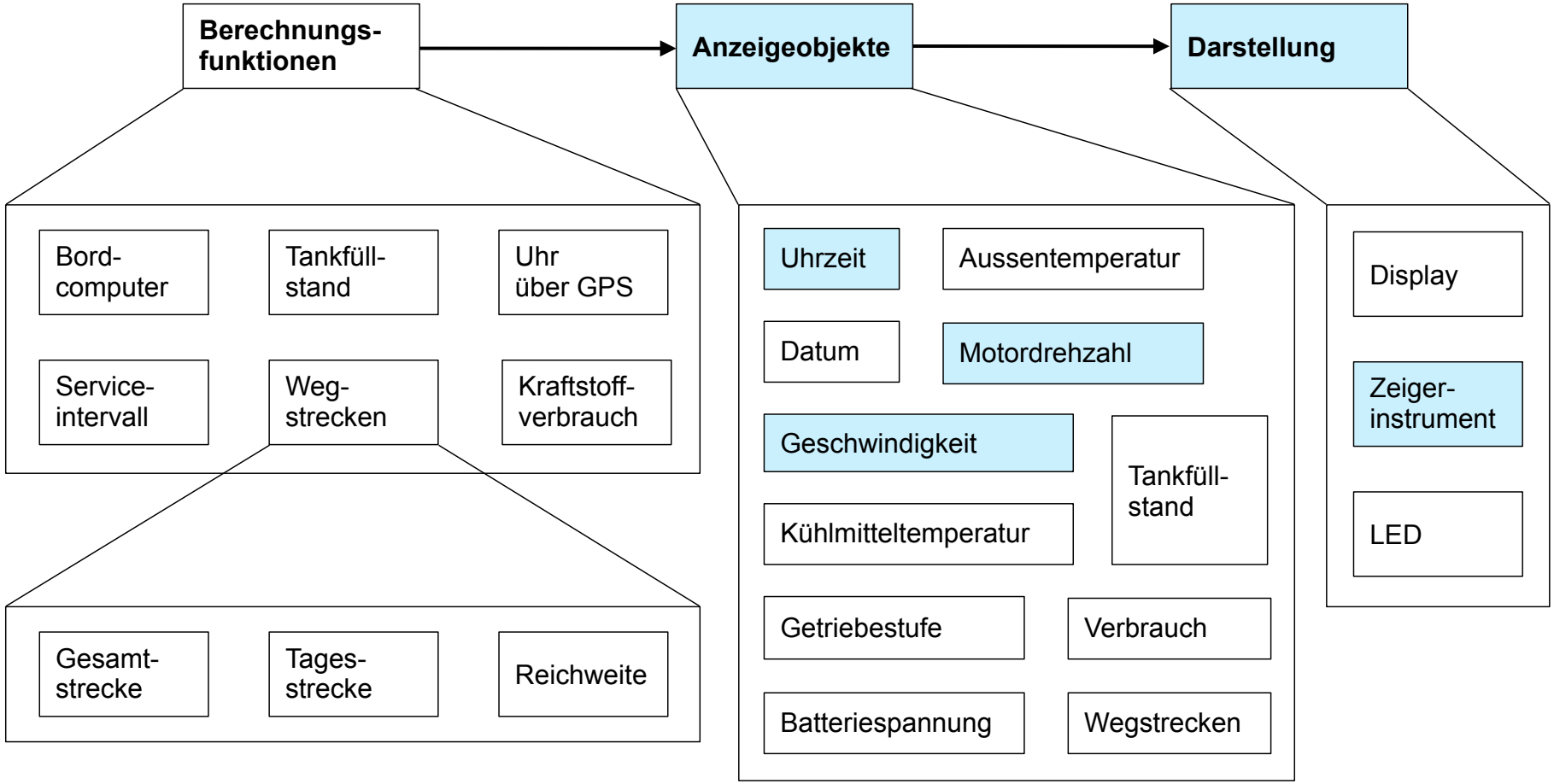
Analyse der Benutzeranforderungen und Spezifikation der logischen Systemarchitektur



Logische Systemarchitektur des Kombiinstruments (6)

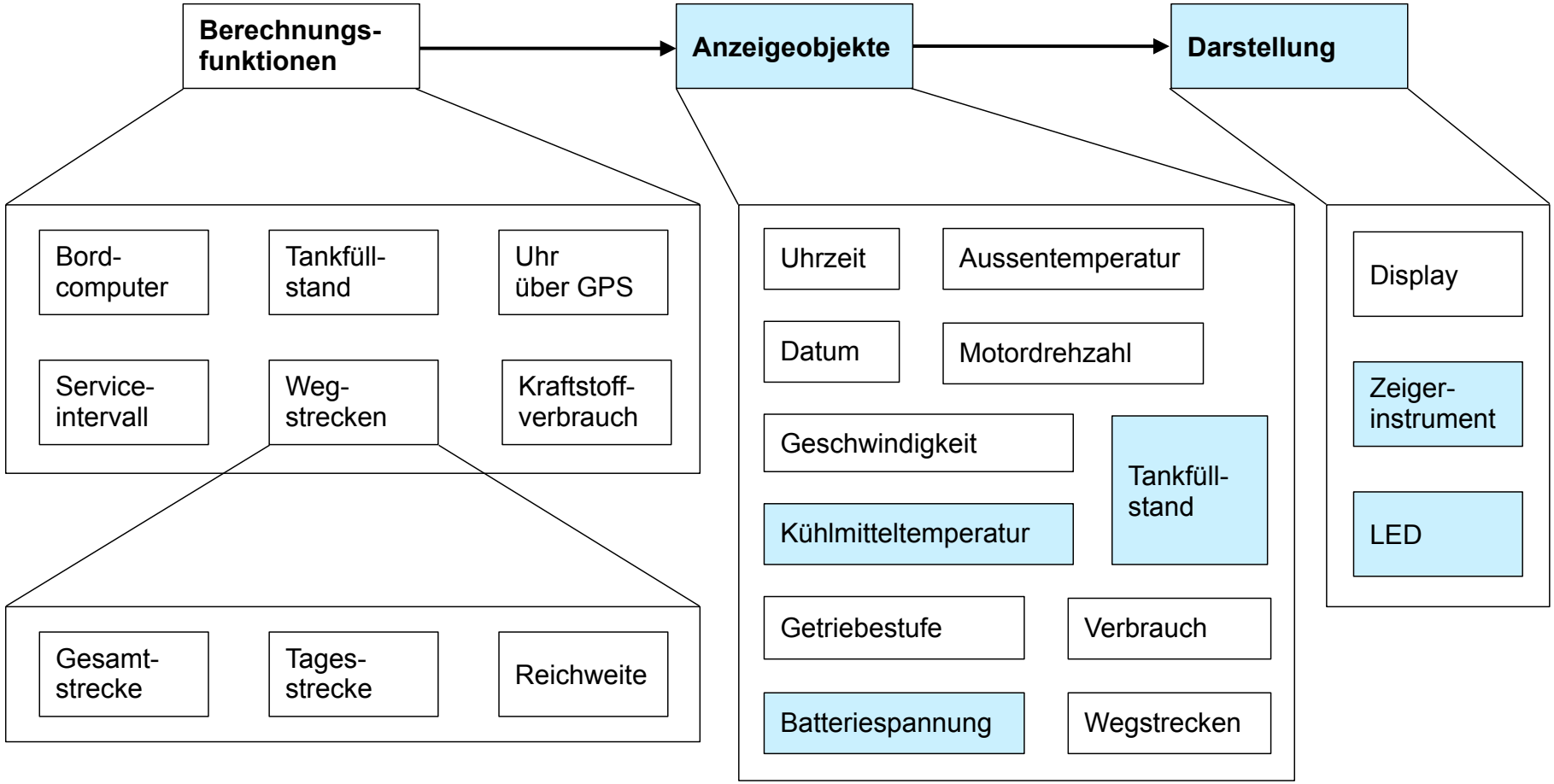
(Nach Schäuffele, Zurawka)

Analyse der Benutzeranforderungen und Spezifikation der logischen Systemarchitektur



Logische Systemarchitektur des Kombiinstruments (7)

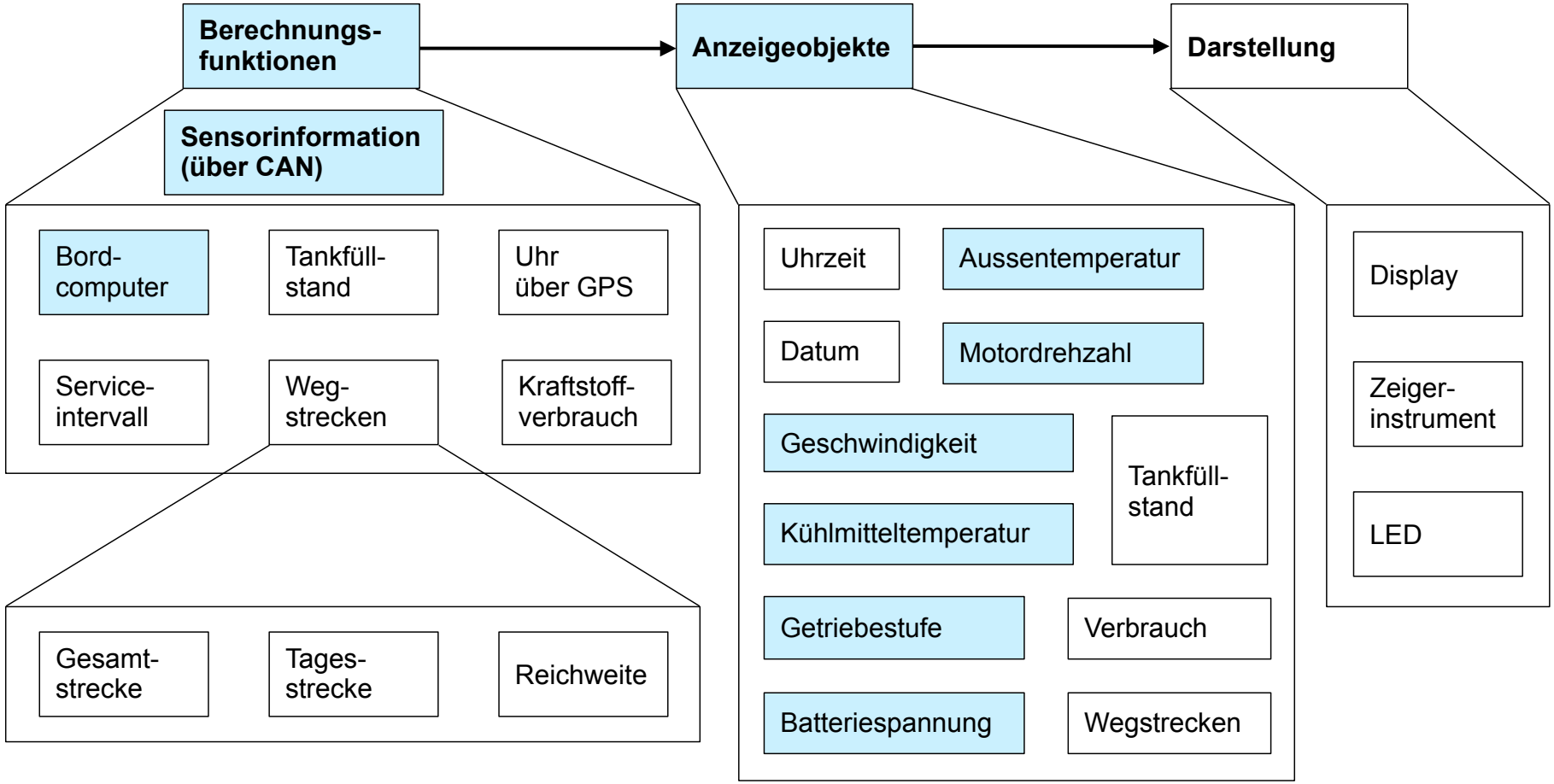
(Nach Schäuffele, Zurawka)



Logische Systemarchitektur des Kombiinstruments (8)

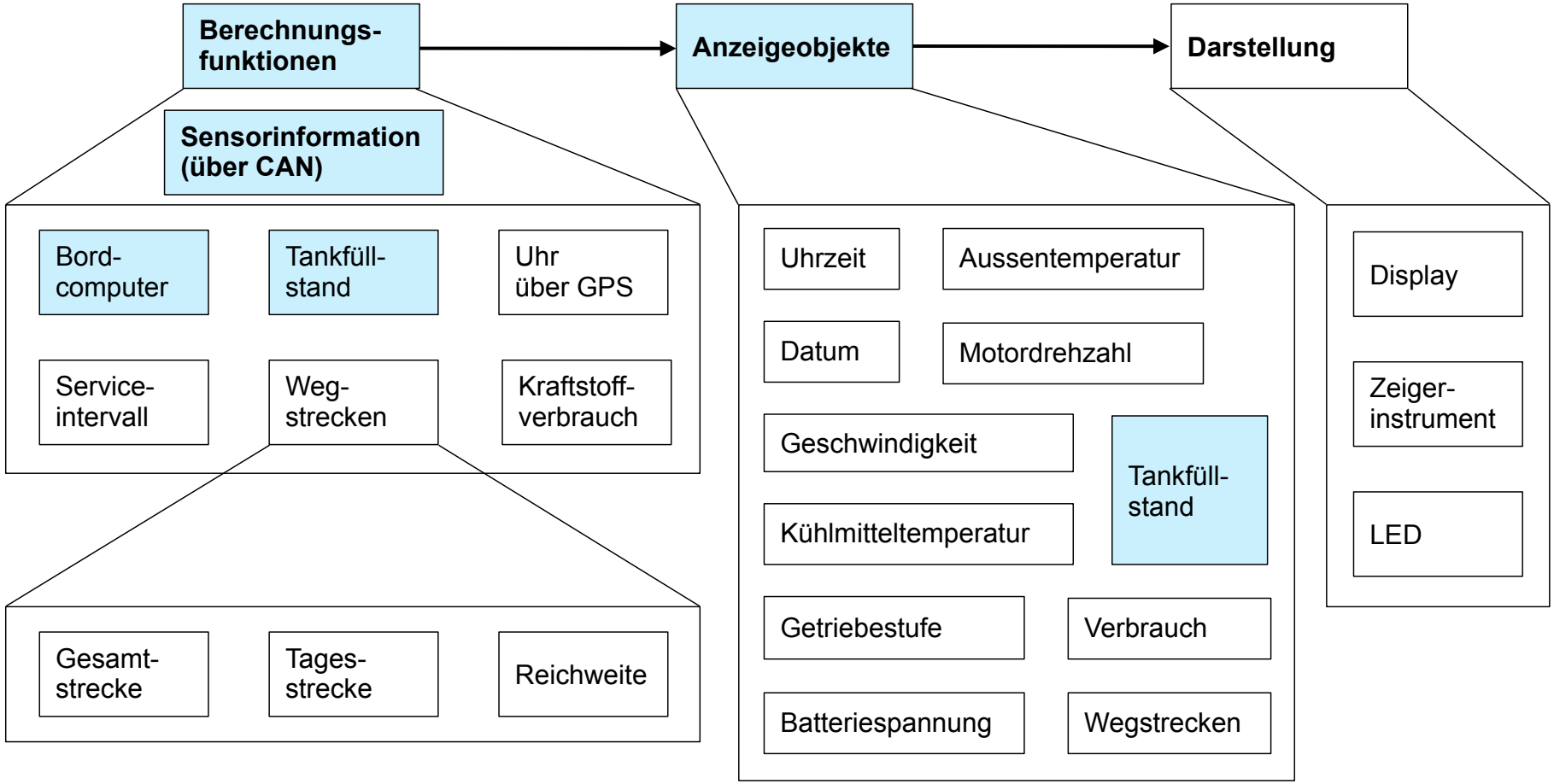
(Nach Schäuffele, Zurawka)

Analyse der Benutzeranforderungen und Spezifikation der logischen Systemarchitektur



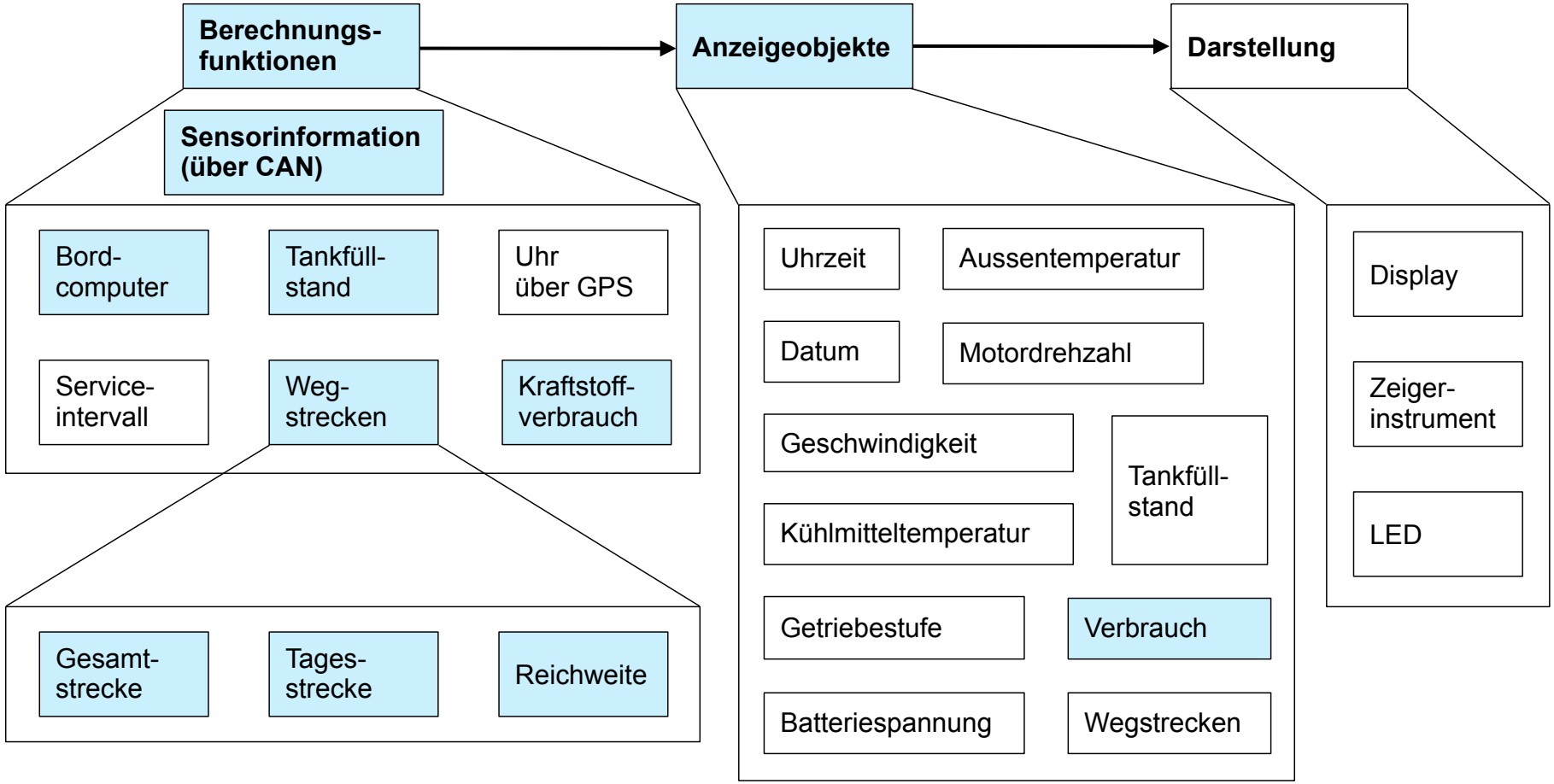
Logische Systemarchitektur des Kombiinstruments (9)

(Nach Schäuffele, Zurawka)



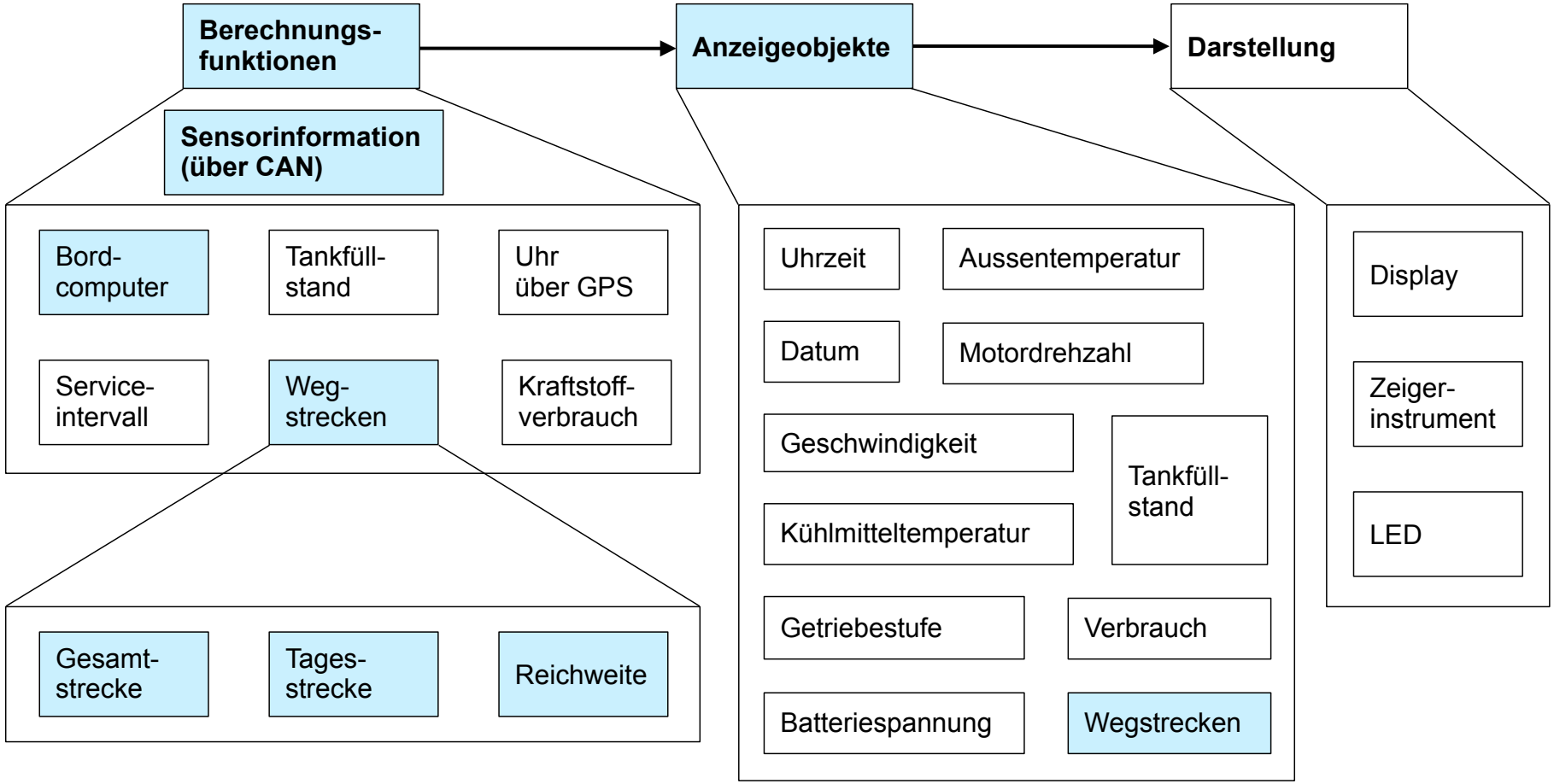
Logische Systemarchitektur des Kombiinstruments (10)

(Nach Schäuffele, Zurawka)



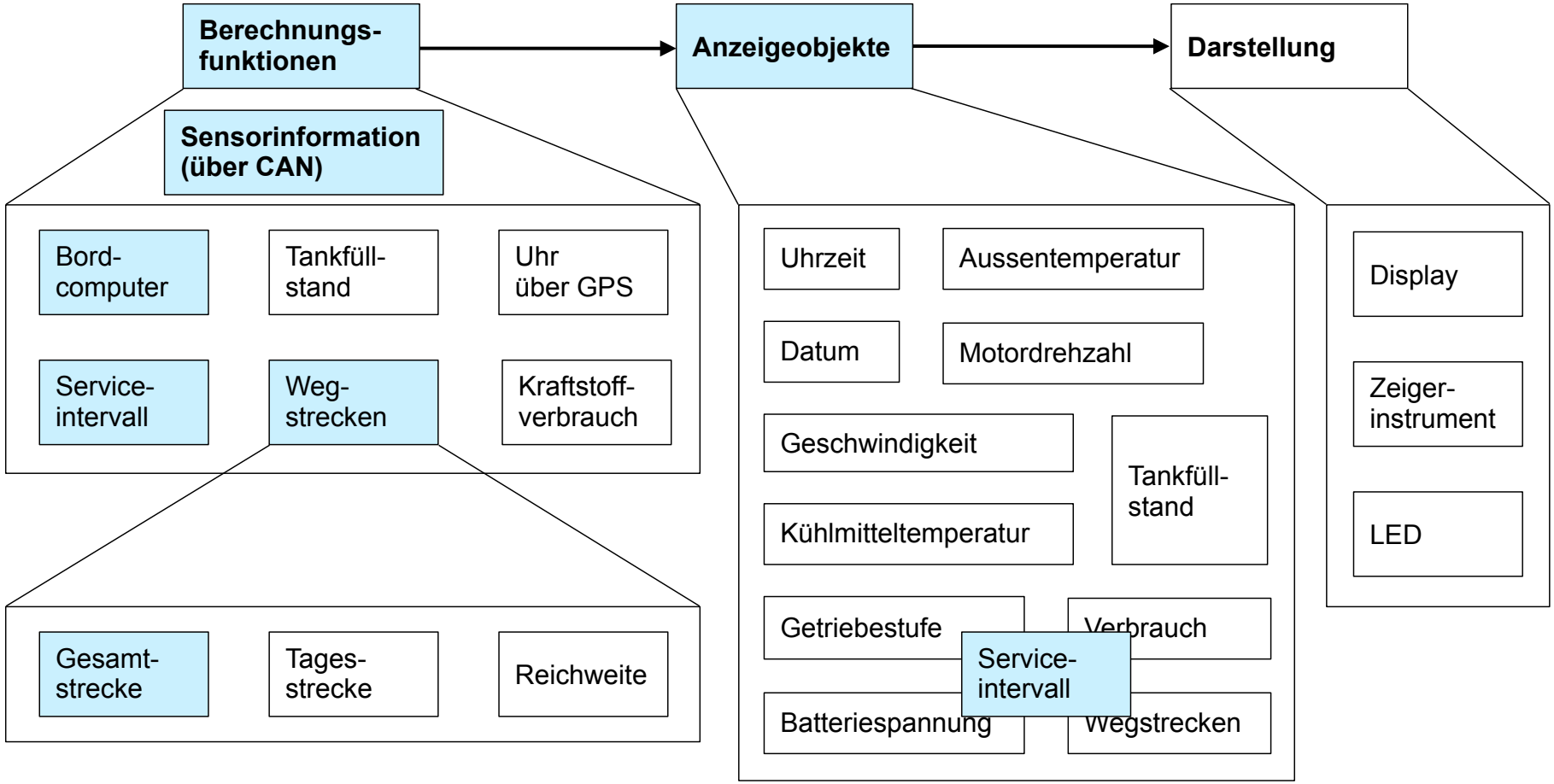
Logische Systemarchitektur des Kombiinstruments (11)

(Nach Schäuffele, Zurawka)



Logische Systemarchitektur des Kombiinstruments (12)

(Nach Schäuffele, Zurawka)



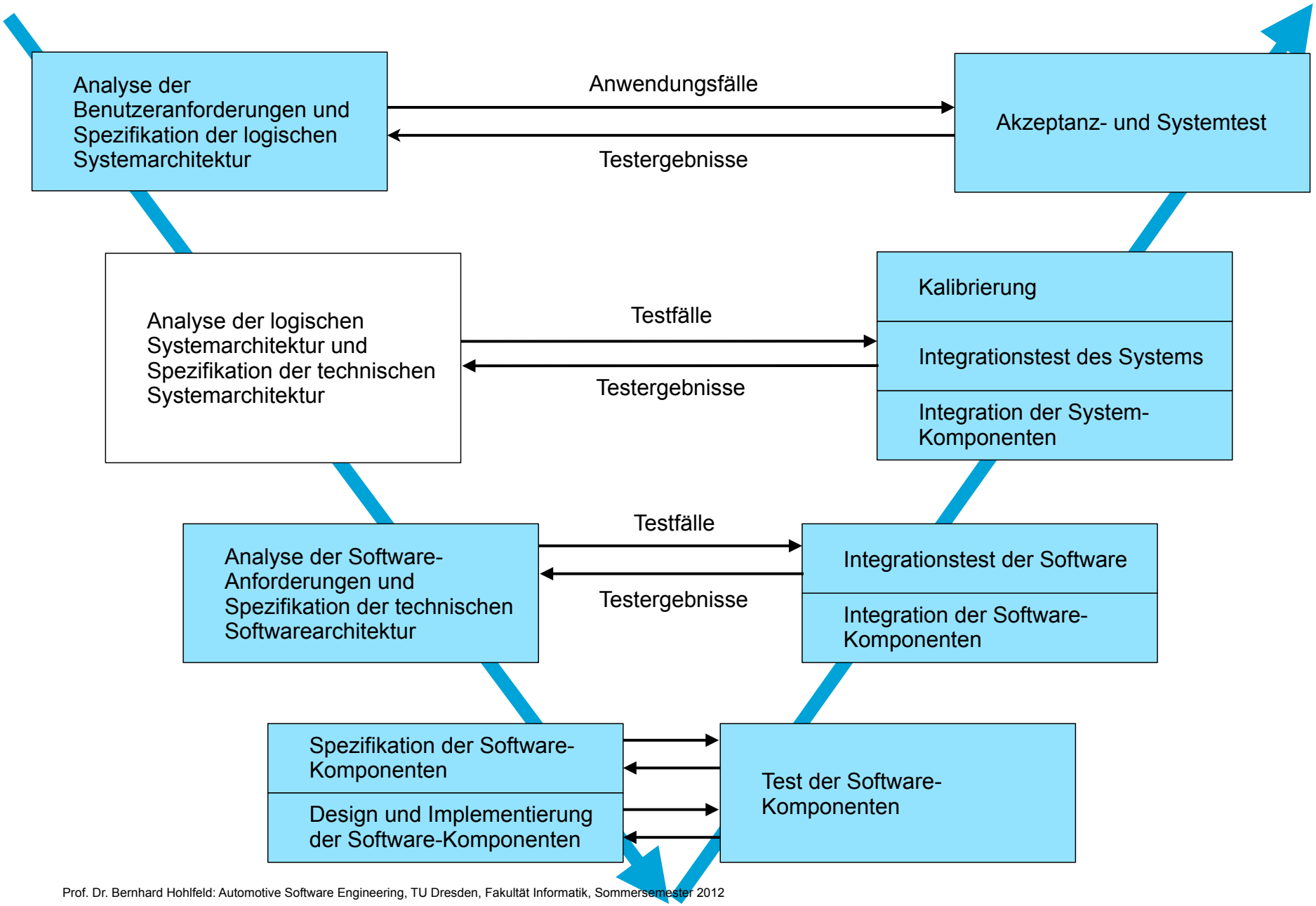
Logische Systemarchitektur des Kombiinstruments (13)

(Nach Schäuffele, Zurawka)

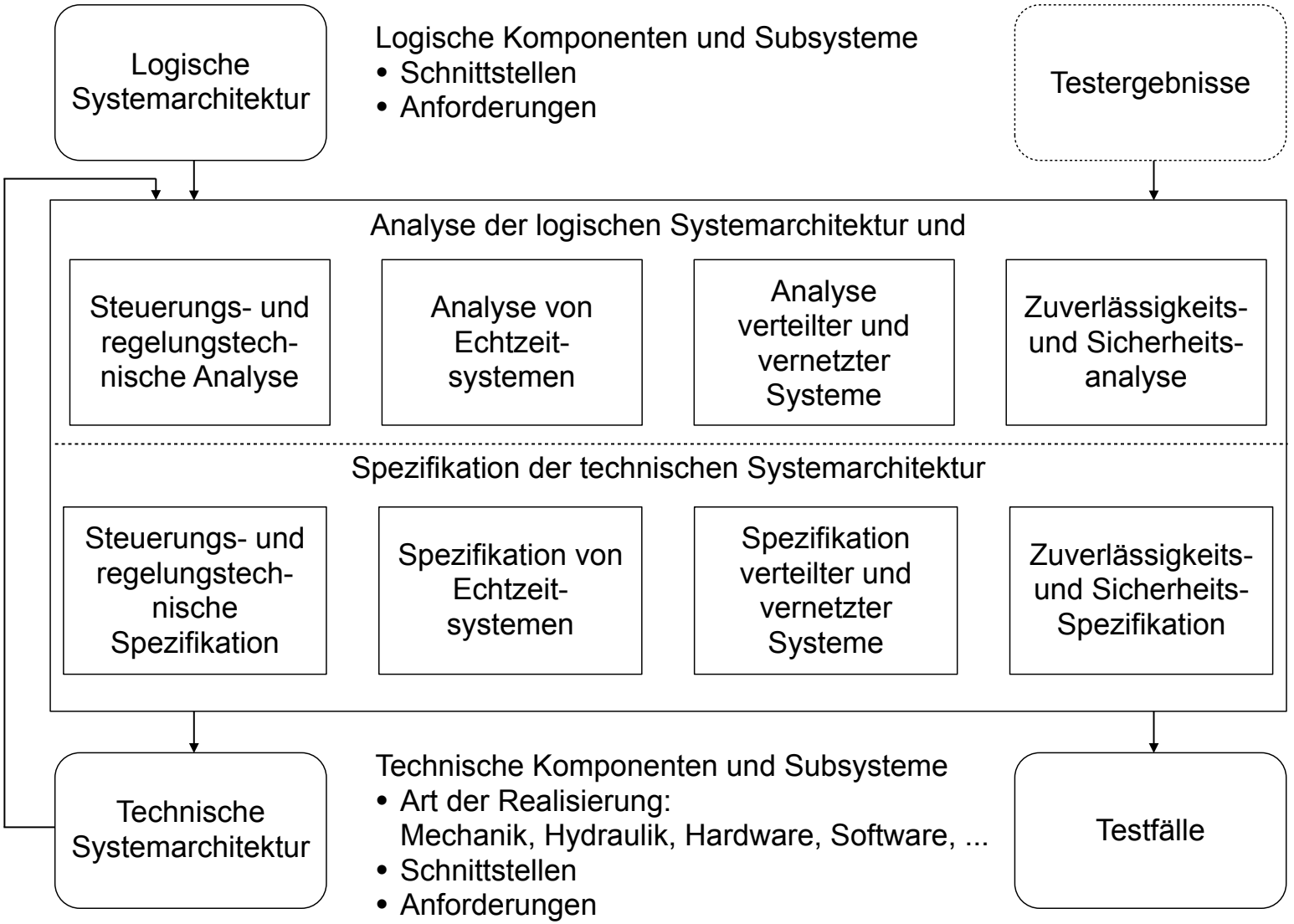
6. SW-Entwicklung / 1. Kernprozess
Kernprozess zur Entwicklung von elektronischen Systemen und Software

1. Grundbegriffe
2. Entwicklungsobjekt: Kombiinstrument
3. Analyse der Benutzeranforderungen und Spezifikation der logischen Systemarchitektur
4. **Analyse der logischen Systemarchitektur und Spezifikation der technischen Systemarchitektur**
5. Analyse der Software-Anforderungen und Spezifikation der technischen Softwarearchitektur
6. Spezifikation der Software-Komponenten
7. Design und Implementierung der Software-Komponenten
8. Test der Software-Komponenten
9. Integration der Software-Komponenten
10. Integrationstest der Software
11. Integration der System-Komponenten
12. Integrationstest des Systems
13. Kalibrierung
14. Akzeptanz- und Systemtest

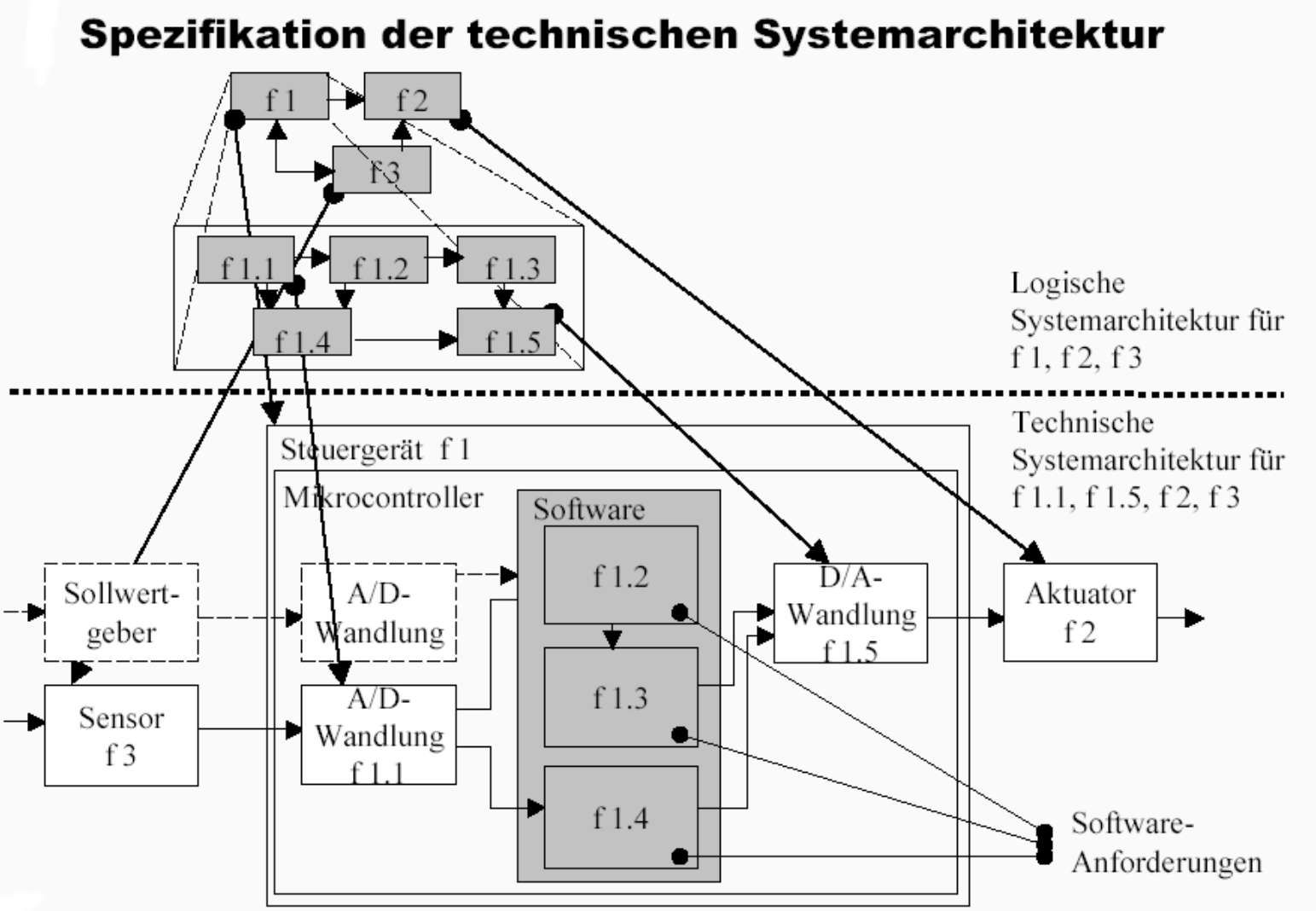
Analyse der logischen Systemarchitektur und Spezifikation der technischen Systemarchitektur



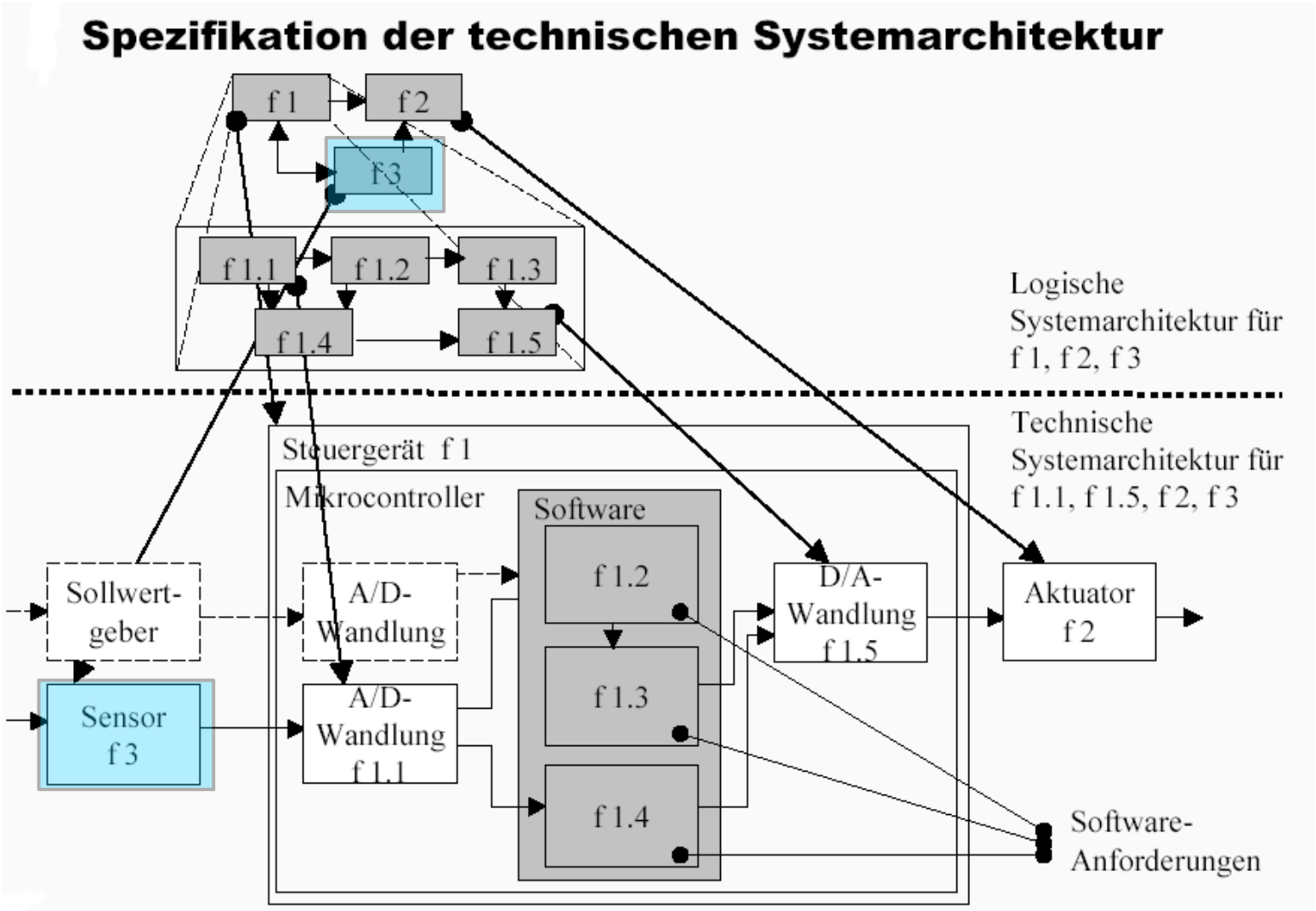
Analyse der logischen Systemarchitektur und Spezifikation der technischen Systemarchitektur (Nach Schäuffele, Zurawka)



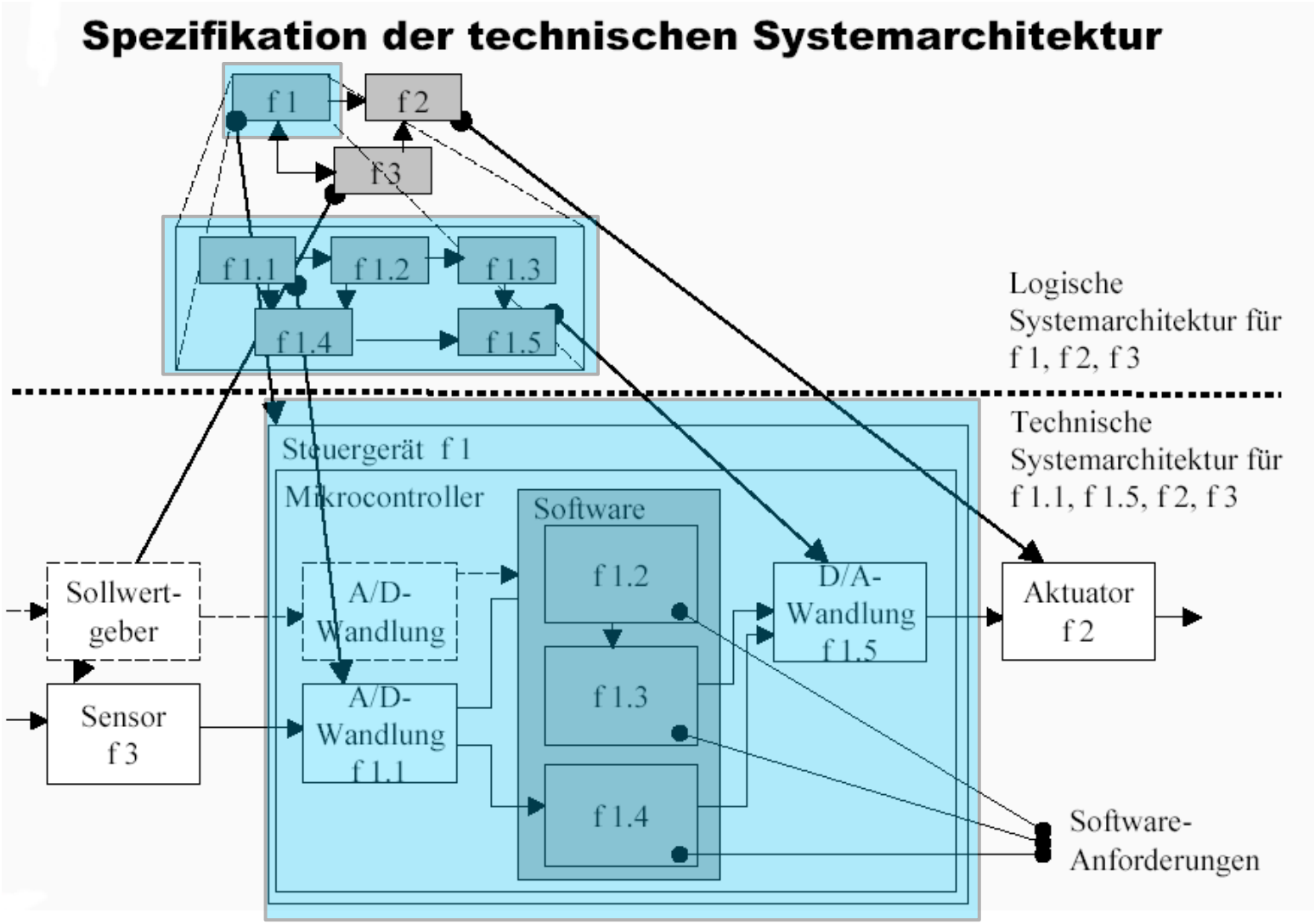
Analyse der logischen Systemarchitektur und Spezifikation der technischen Systemarchitektur (Nach Schäuffele, Zurawka)



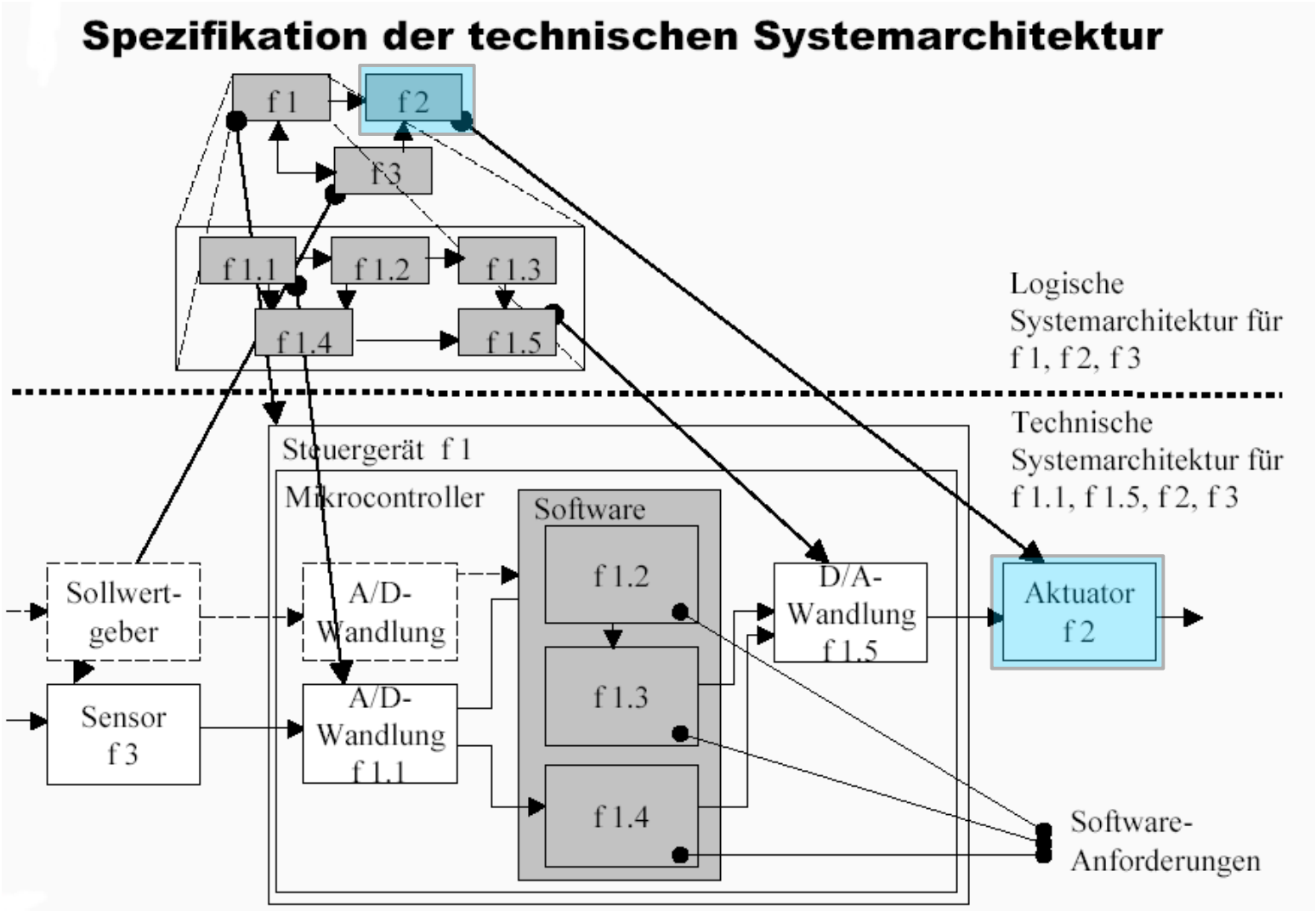
Analyse der logischen Systemarchitektur und Spezifikation der technischen Systemarchitektur (Nach Schäuffele, Zurawka)



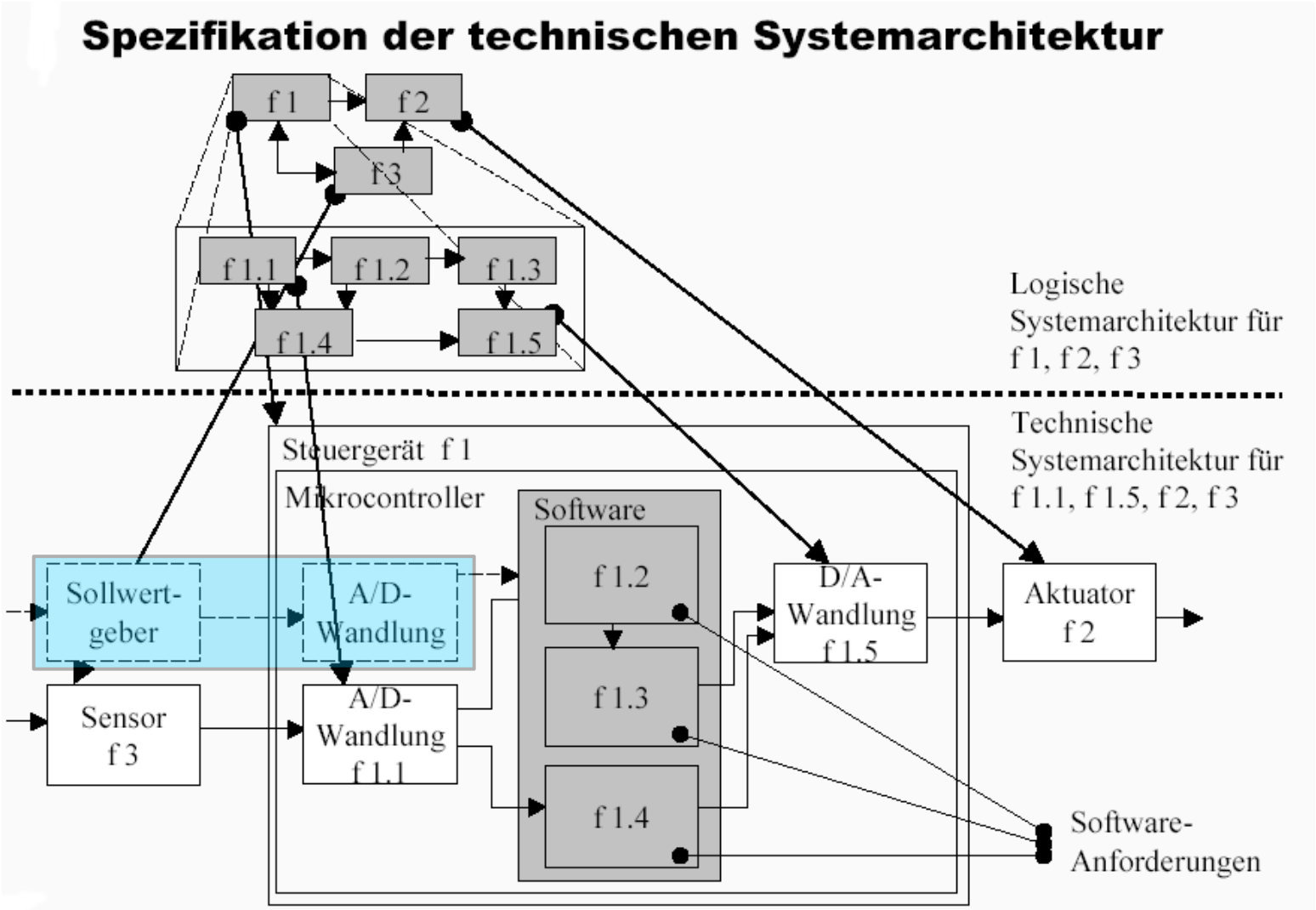
Analyse der logischen Systemarchitektur und Spezifikation der technischen Systemarchitektur (Nach Schäuffele, Zurawka)



Analyse der logischen Systemarchitektur und Spezifikation der technischen Systemarchitektur (Nach Schäuffele, Zurawka)



Analyse der logischen Systemarchitektur und Spezifikation der technischen Systemarchitektur (Nach Schäuffele, Zurawka)

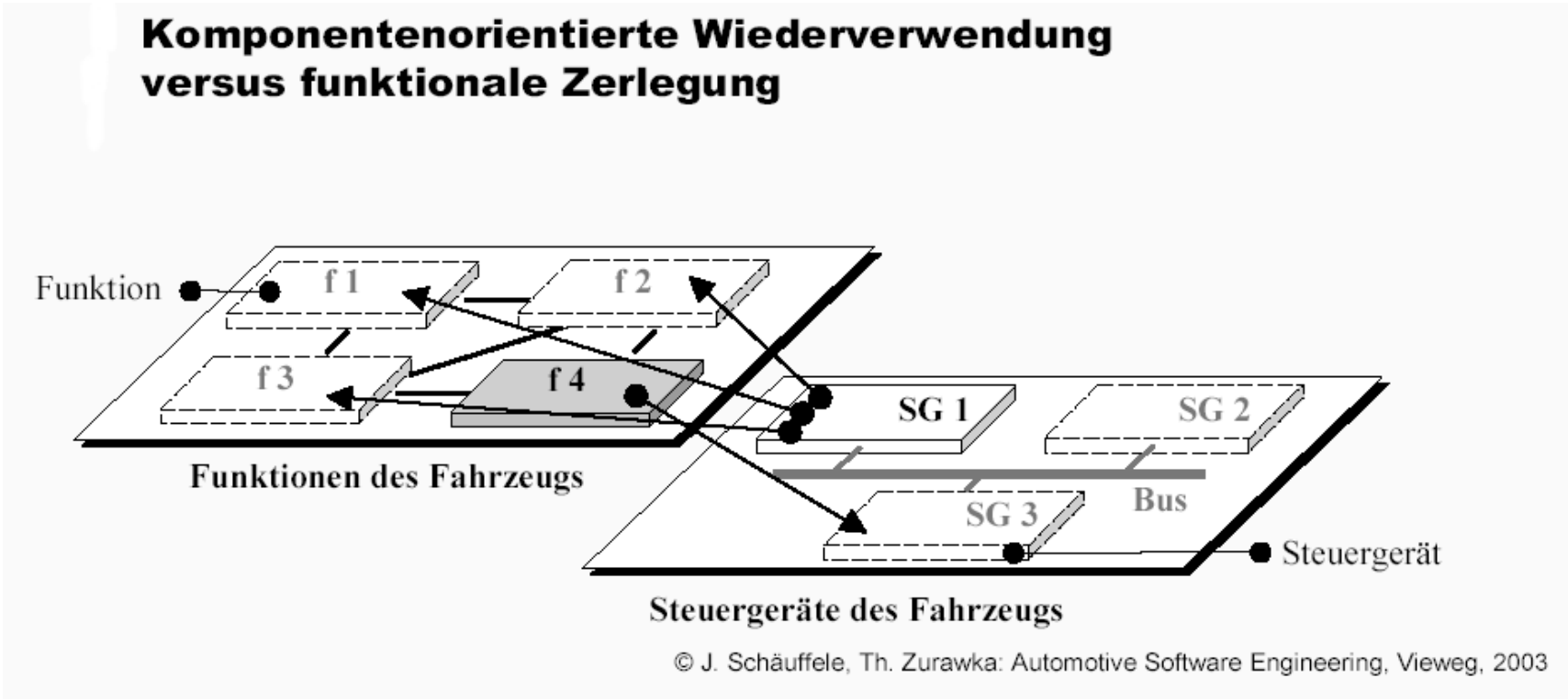


Analyse der logischen Systemarchitektur und Spezifikation der technischen Systemarchitektur (Nach Schäuffele, Zurawka)

- Randbedingungen für die technische Systemarchitektur
  - Standards und Entwurfsmuster
  - Abhängigkeiten zwischen verschiedenen Systemen und Komponenten
  - Ergebnisse von Machbarkeitsstudien
  - Produktions- und Serviceanforderungen
  - Änderbarkeits- und Testbarkeitsanforderungen
  - Aufwands-, Kosten- und Risikoabschätzungen
- Beispiele
  - Wiederverwendung von technischen Komponenten in verschiedenen Fahrzeugbaureihen
  - Verschiedene Fahrzeugvarianten innerhalb einer Fahrzeugbaureihe
  - Sonderausstattung versus Serienausstattung
  - Länderspezifische Ausstattungsvarianten
  - Komponentenorientierte Wiederverwendung

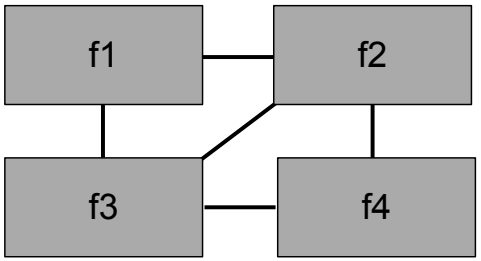
Randbedingungen und Zielkonflikte

- Wiederverwendung von technischen Komponenten in verschiedenen Baureihen
  - Motoren
  - Getriebe
  - Einheitliche Motor- und Getriebesteuergeräte mit unterschiedlichem Programm und Datenstand
- Verschiedene Varianten innerhalb einer Baureihe
  - Schaltgetriebe
  - Automatikgetriebe
  - Trennung von Motor- und Getriebesteuergerät
- Sonderausstattung und Serienausstattung
  - Serienausstattung
    - Realisierung auf einem Steuergerät
  - Sonderausstattung
    - Regensensor
    - Einparkhilfe
    - Elektrische Sitzverstellung
    - Separate Steuergeräte oder „Softwarefreischaltung“

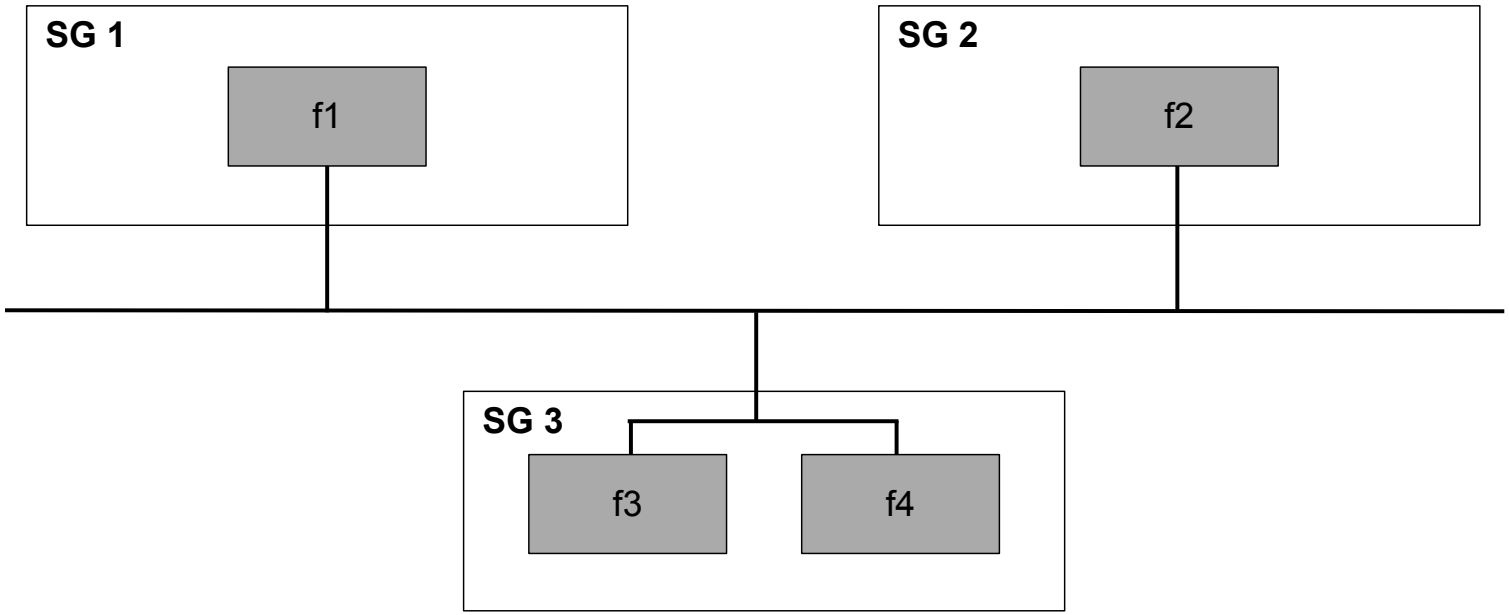


- Vorgabe:
  - Wiederverwendung des Steuergerätes SG1 mit den Funktionen f1, f2, f3
- Freiheitsgrad:
  - Zuordnung der Funktion f4 (z.B. auf SG 3)

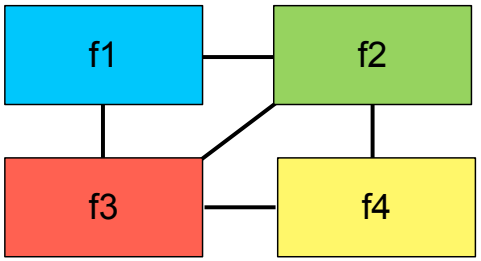
Logische Systemarchitektur und Technische Systemarchitektur (Nach Schäuffele, Zurawka)



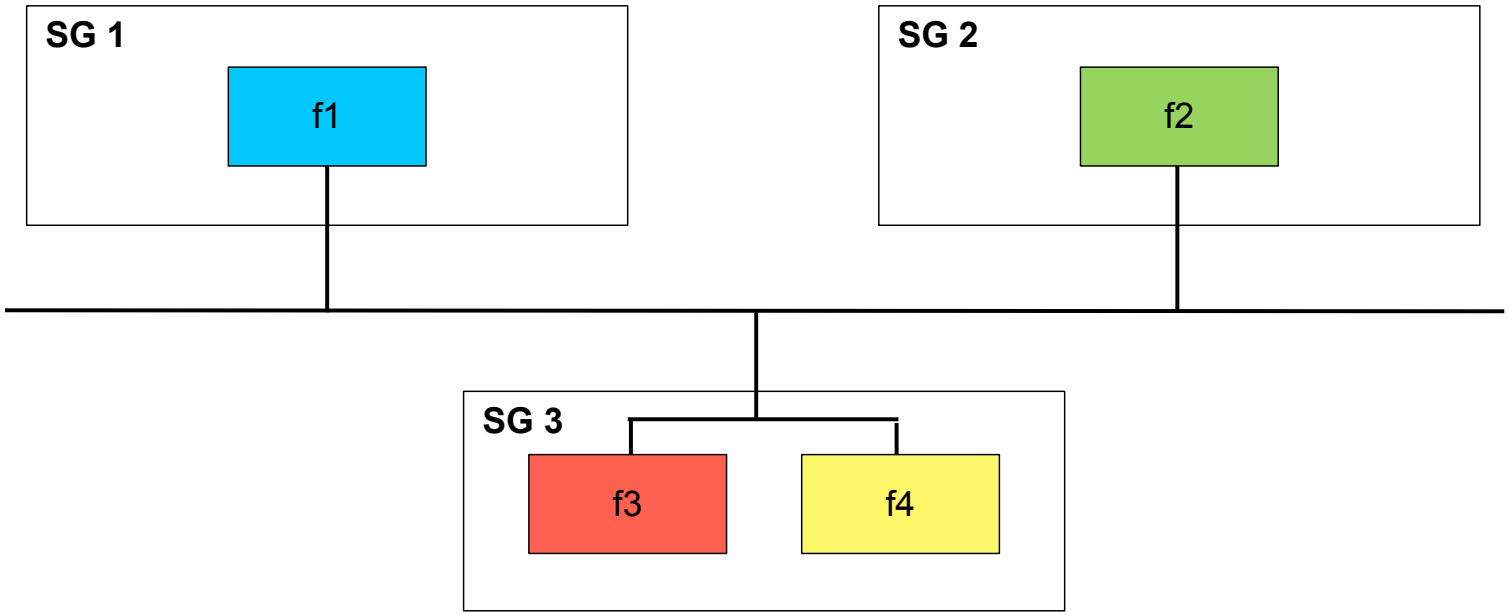
- Logische Systemarchitektur
  - Funktionen
  - Logische Kommunikation
- Technische Systemarchitektur
  - Verteilung der Funktionen auf Steuergeräte
  - Kommunikation im Steuergerät und zwischen Steuergeräten



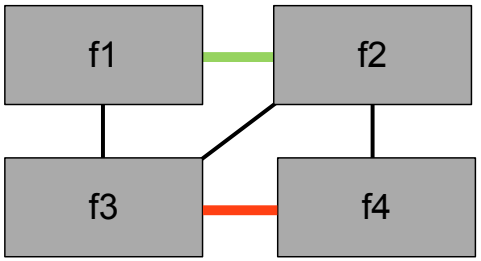
Logische Systemarchitektur und Technische Systemarchitektur (Nach Schäuffele, Zurawka)



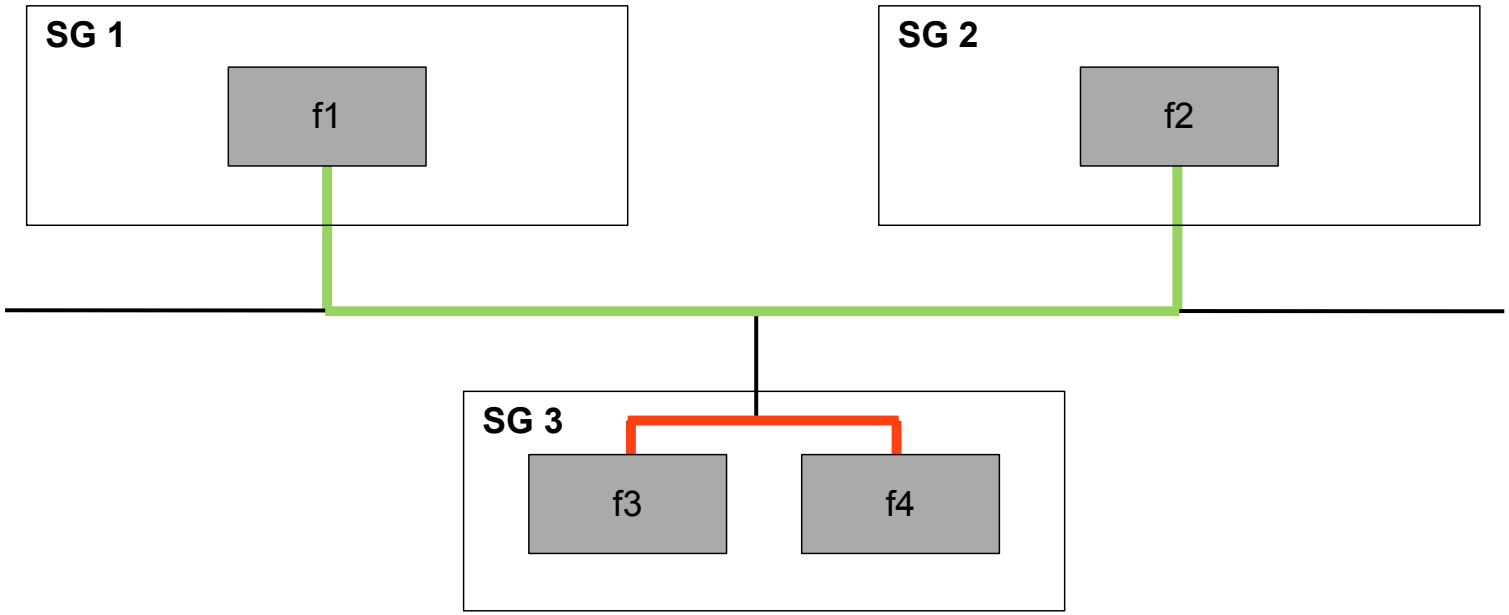
- Logische Systemarchitektur
  - **Funktionen**
  - Logische Kommunikation
- Technische Systemarchitektur
  - **Verteilung der Funktionen auf Steuergeräte**
  - Kommunikation im Steuergerät und zwischen Steuergeräten



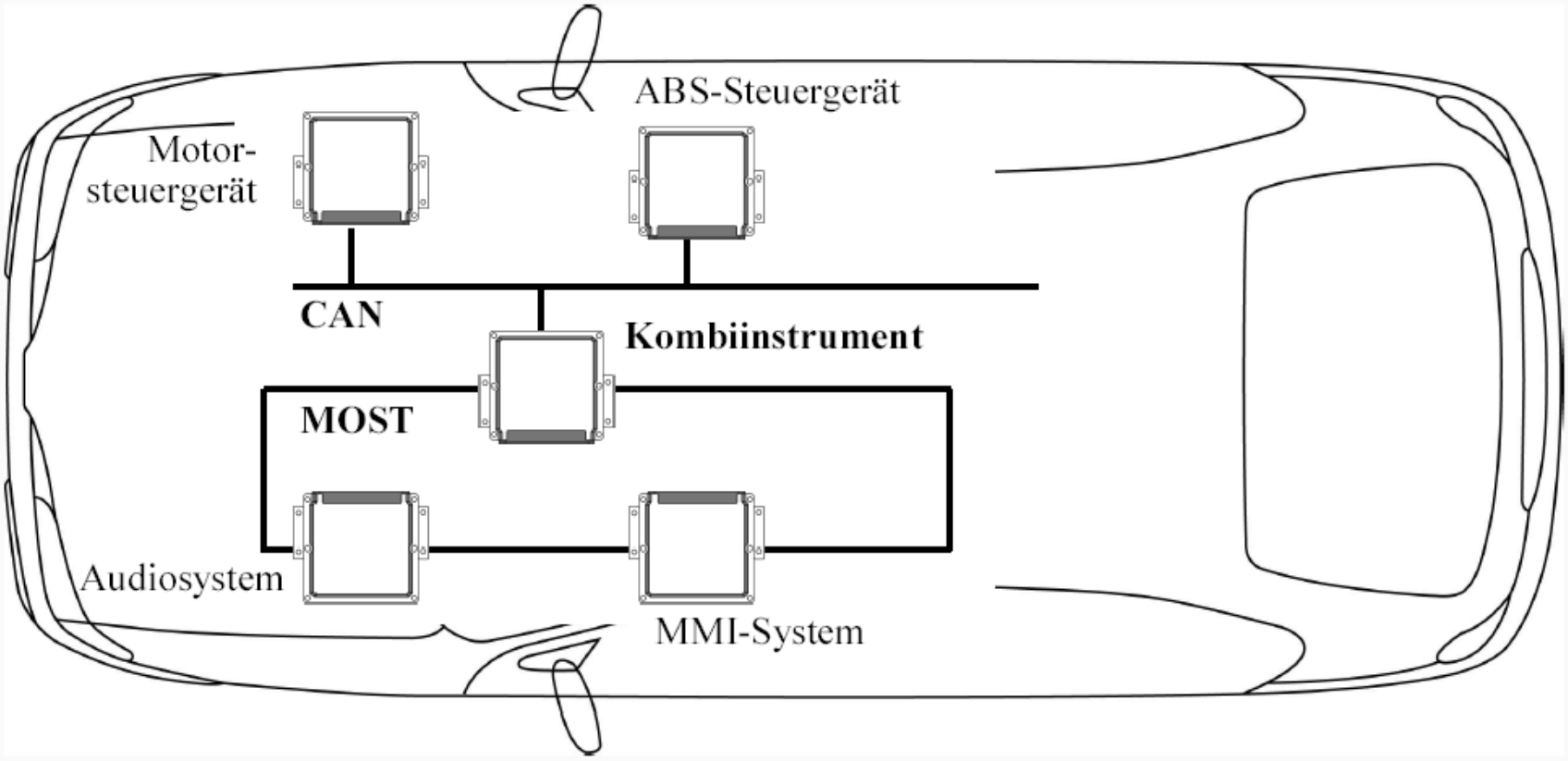
Logische Systemarchitektur und Technische Systemarchitektur (Nach Schäuffele, Zurawka)



- Logische Systemarchitektur
  - Funktionen
  - **Logische Kommunikation**
- Technische Systemarchitektur
  - Verteilung der Funktionen auf Steuergeräte
  - **Kommunikation im Steuergerät und zwischen Steuergeräten**

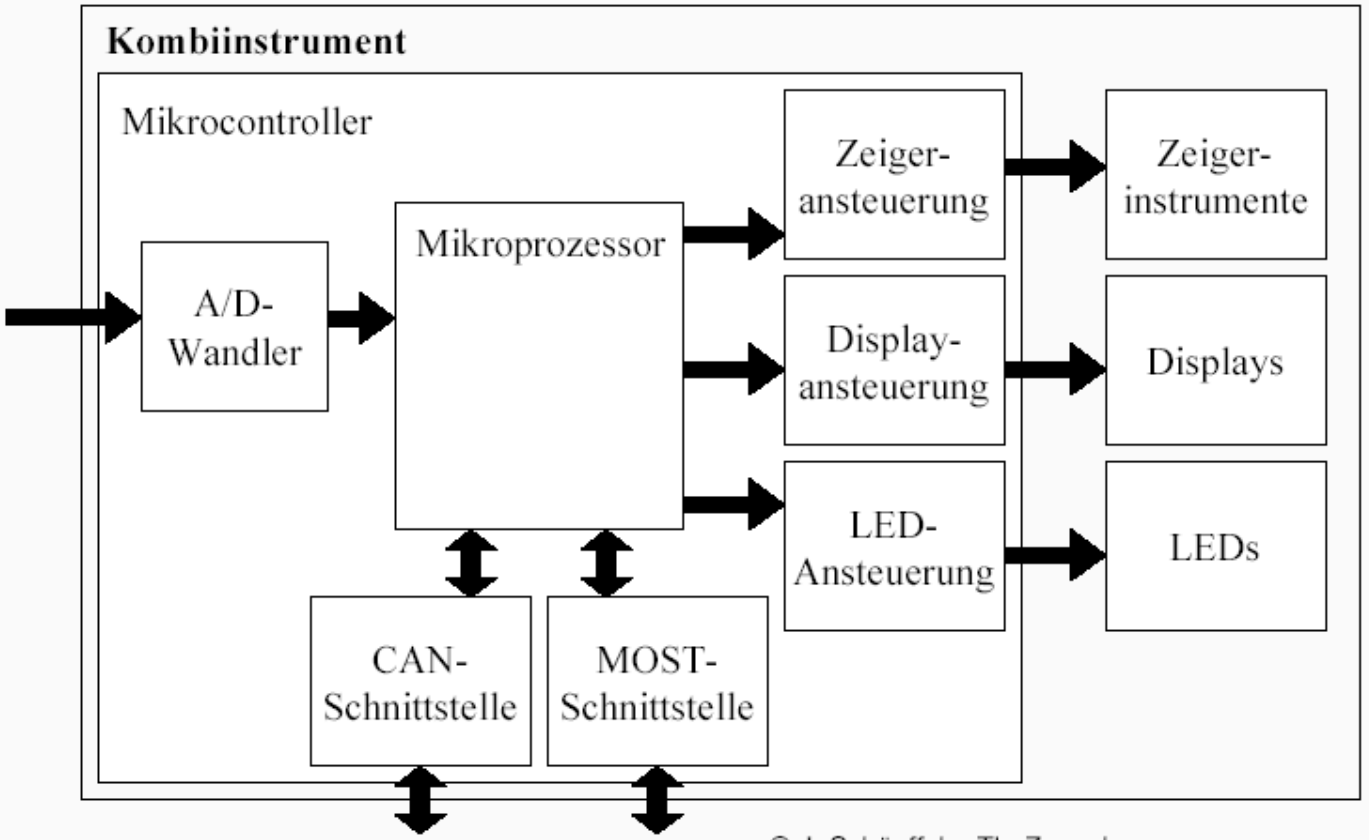


Kombiinstrument als Komponente im Steuergerätenetzwerk

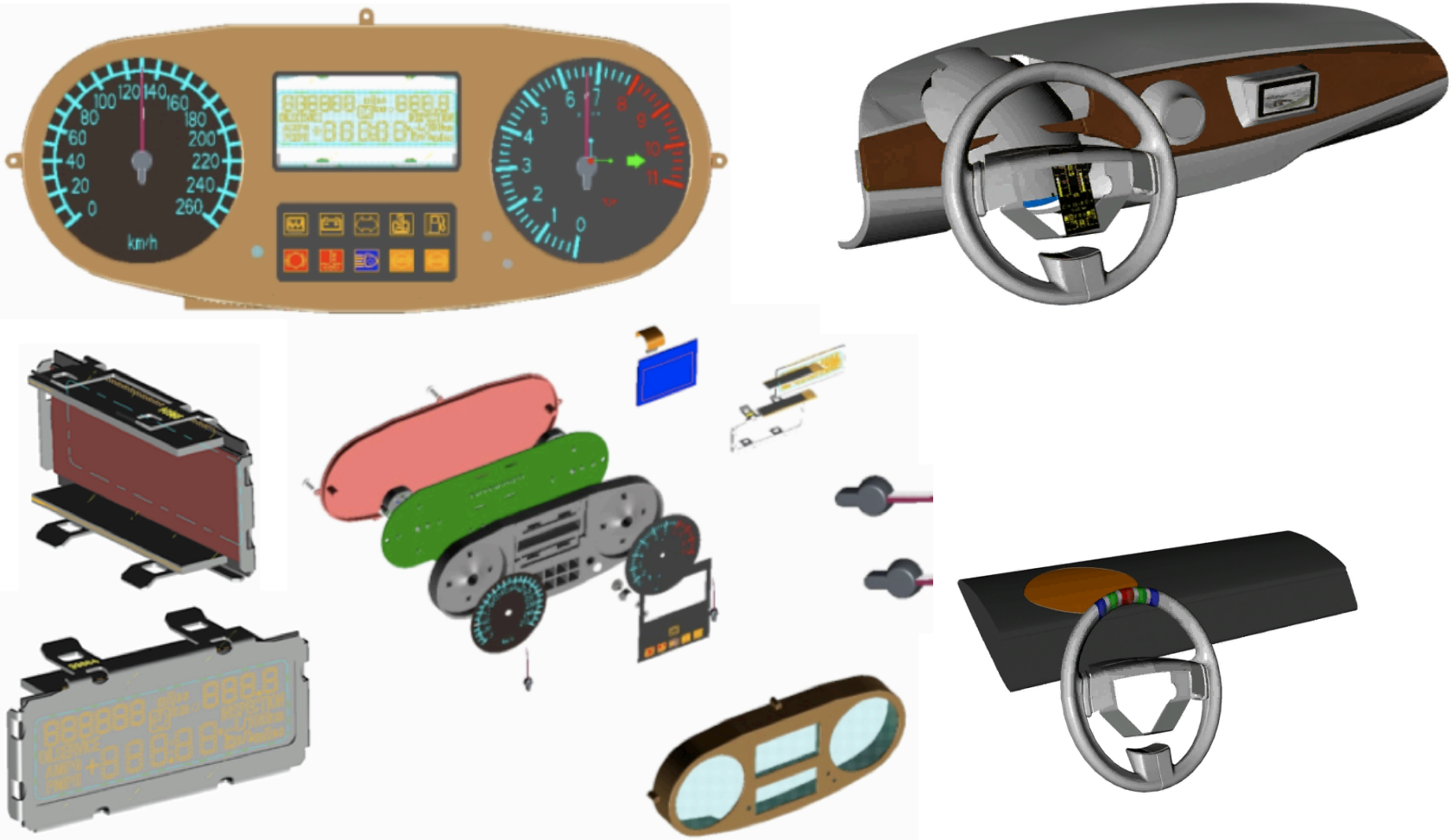


© J. Schäuffele, Th. Zurawka: Automotive Software Engineering, Vieweg, 2003

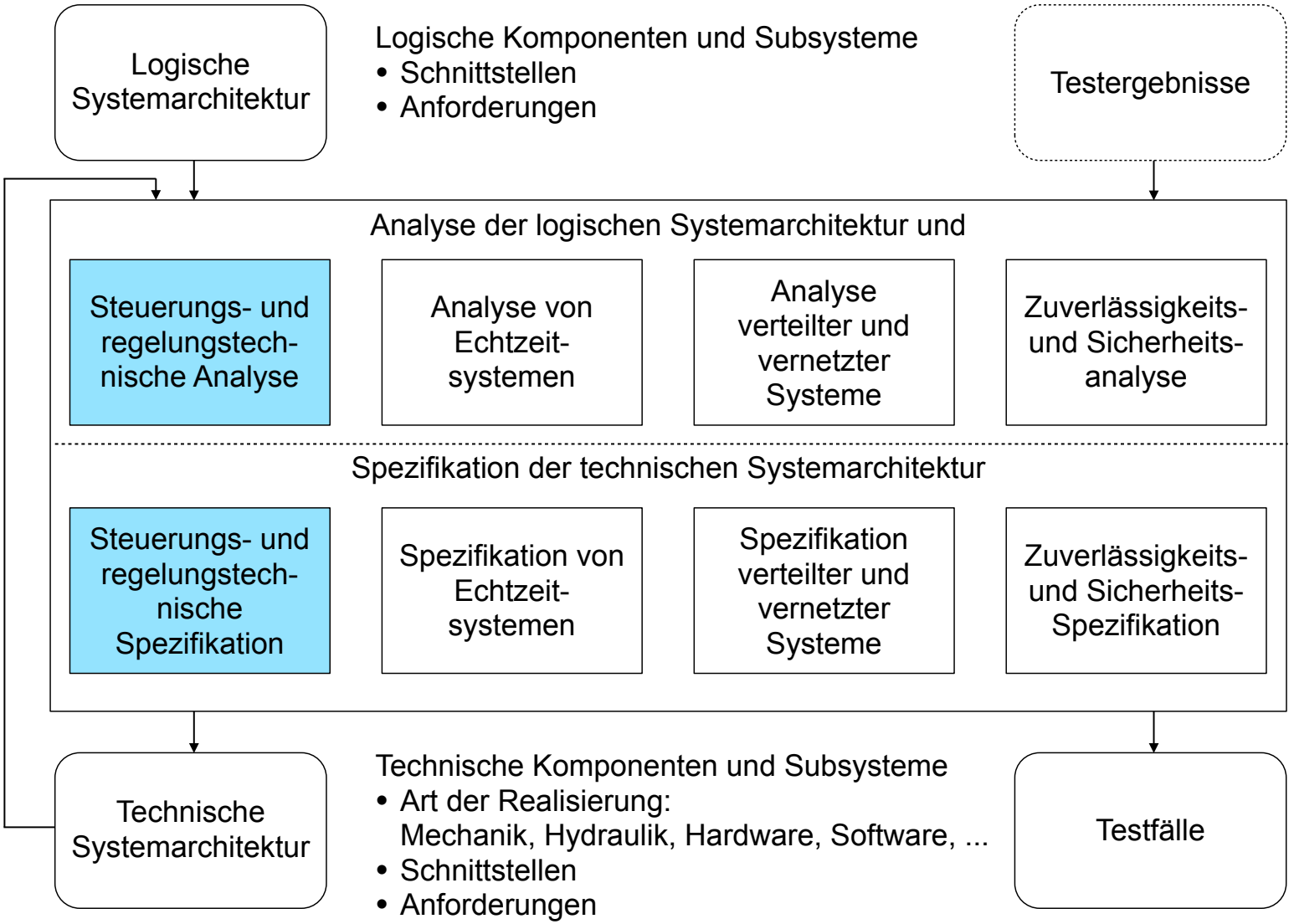
Hardware des Kombiinstruments



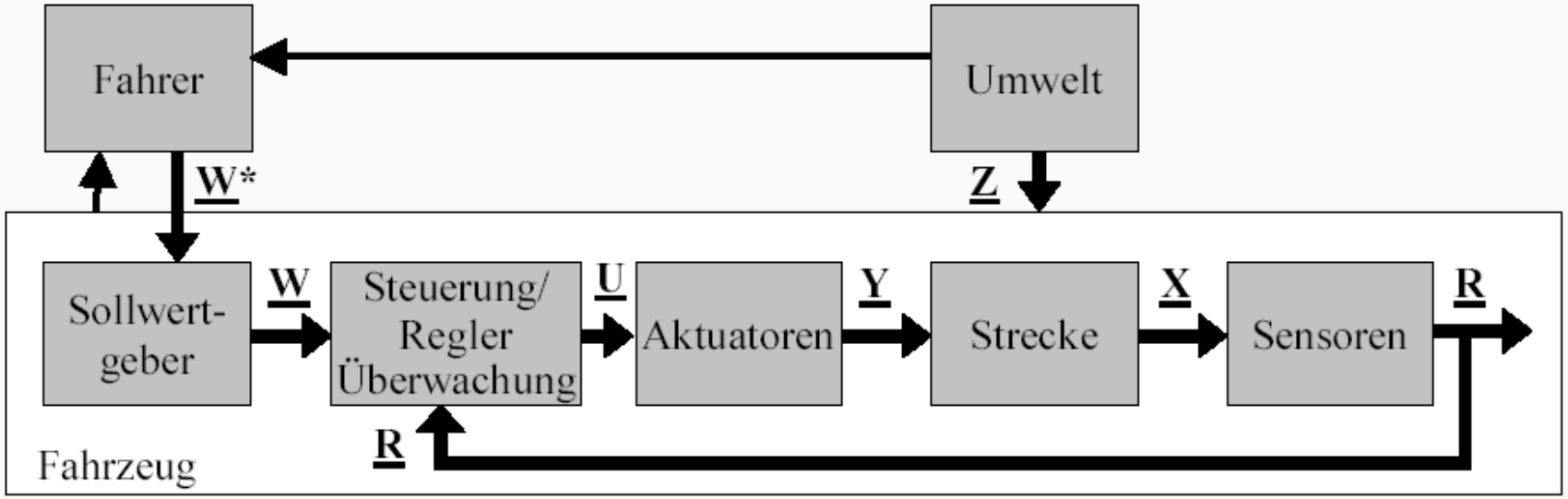
Entwicklungsobjekt: Kombiinstrument



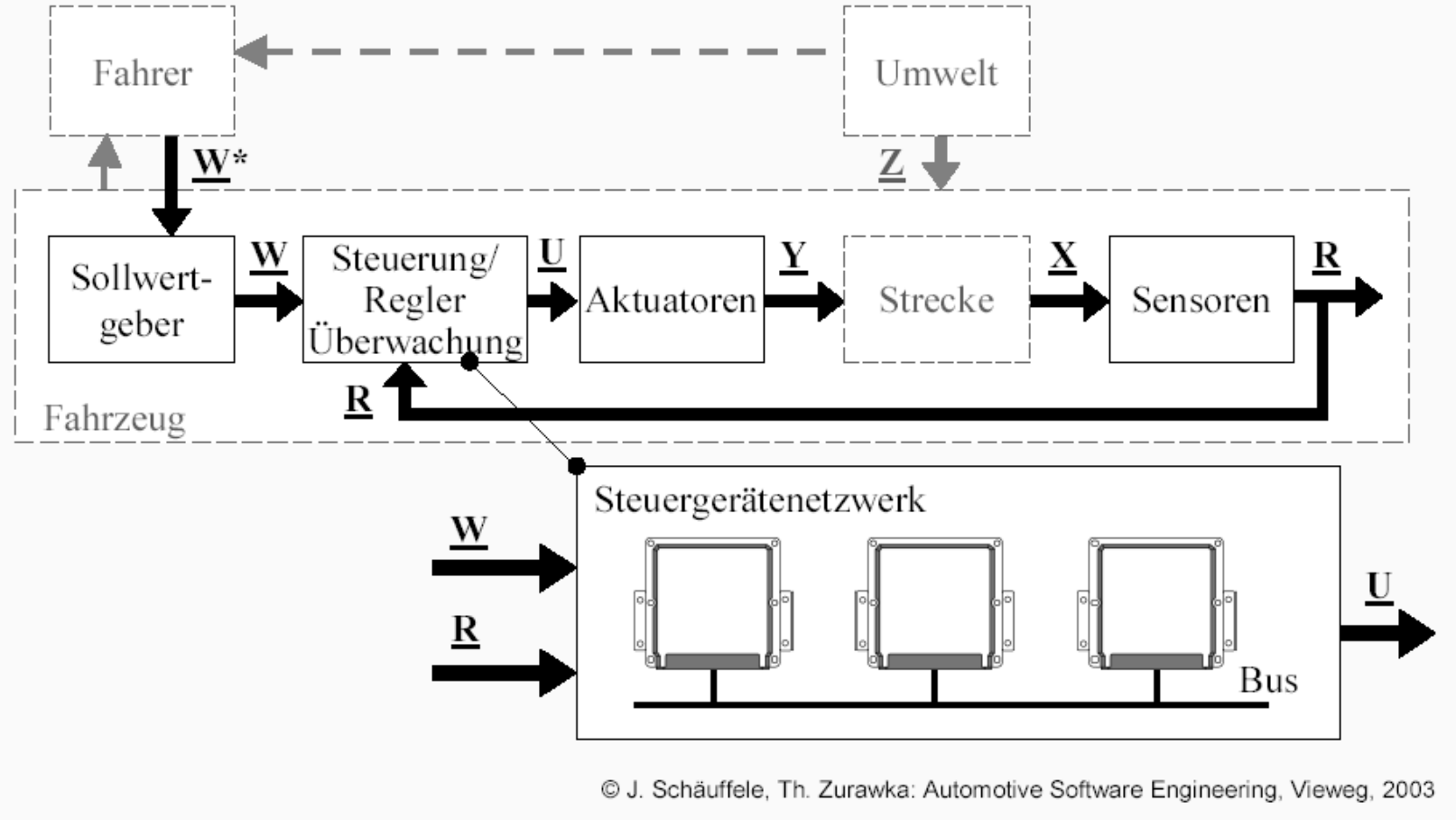
Analyse der logischen Systemarchitektur und Spezifikation der technischen Systemarchitektur (Nach Schäuffele, Zurawka)



Logische Systemarchitektur in der Steuerungs- und Regelungstechnik

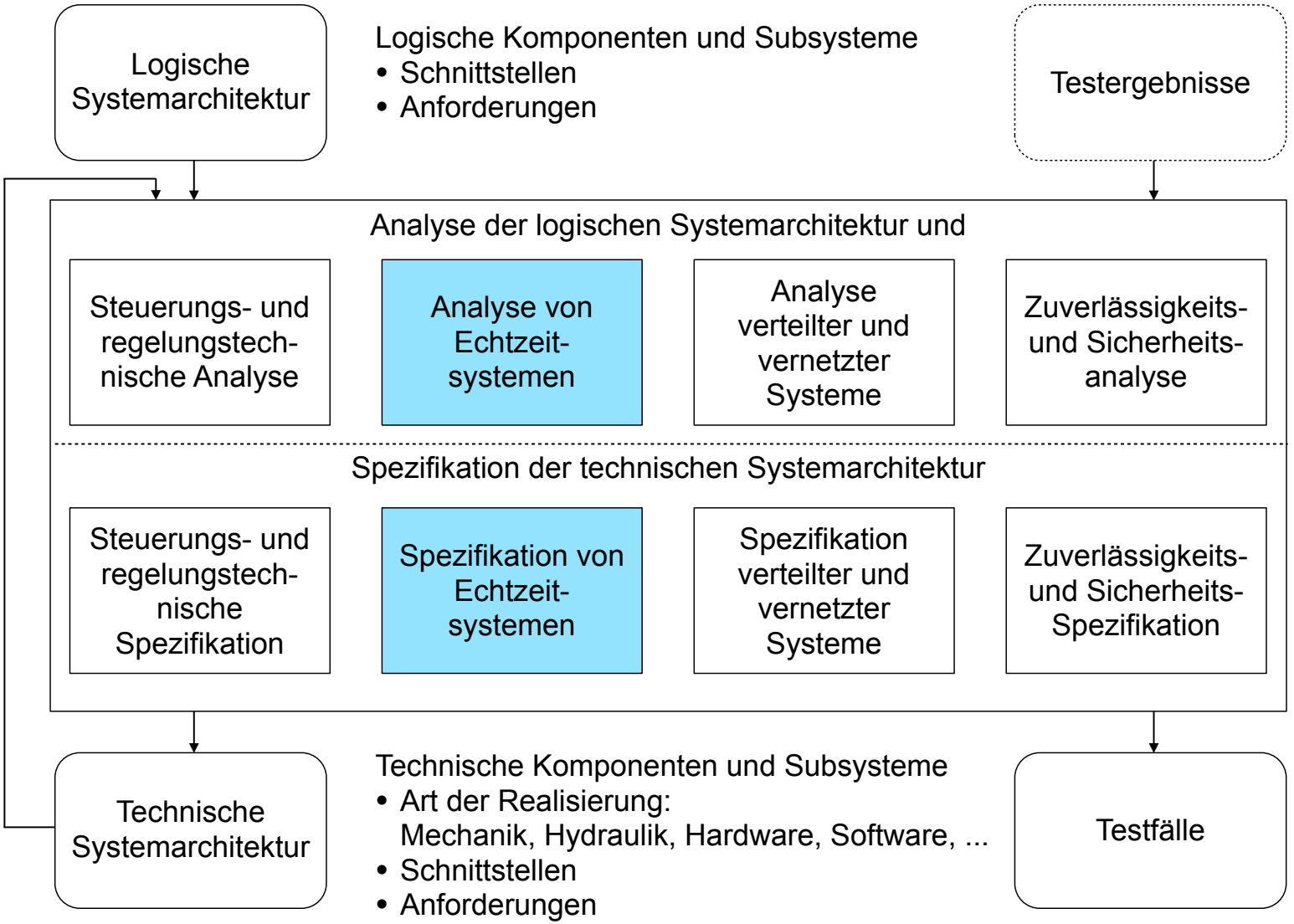


Technische Systemarchitektur in der Steuerungs- und Regelungstechnik



© J. Schäuffele, Th. Zurawka: Automotive Software Engineering, Vieweg, 2003

Analyse der logischen Systemarchitektur und Spezifikation der technischen Systemarchitektur (Nach Schäuffele, Zurawka)

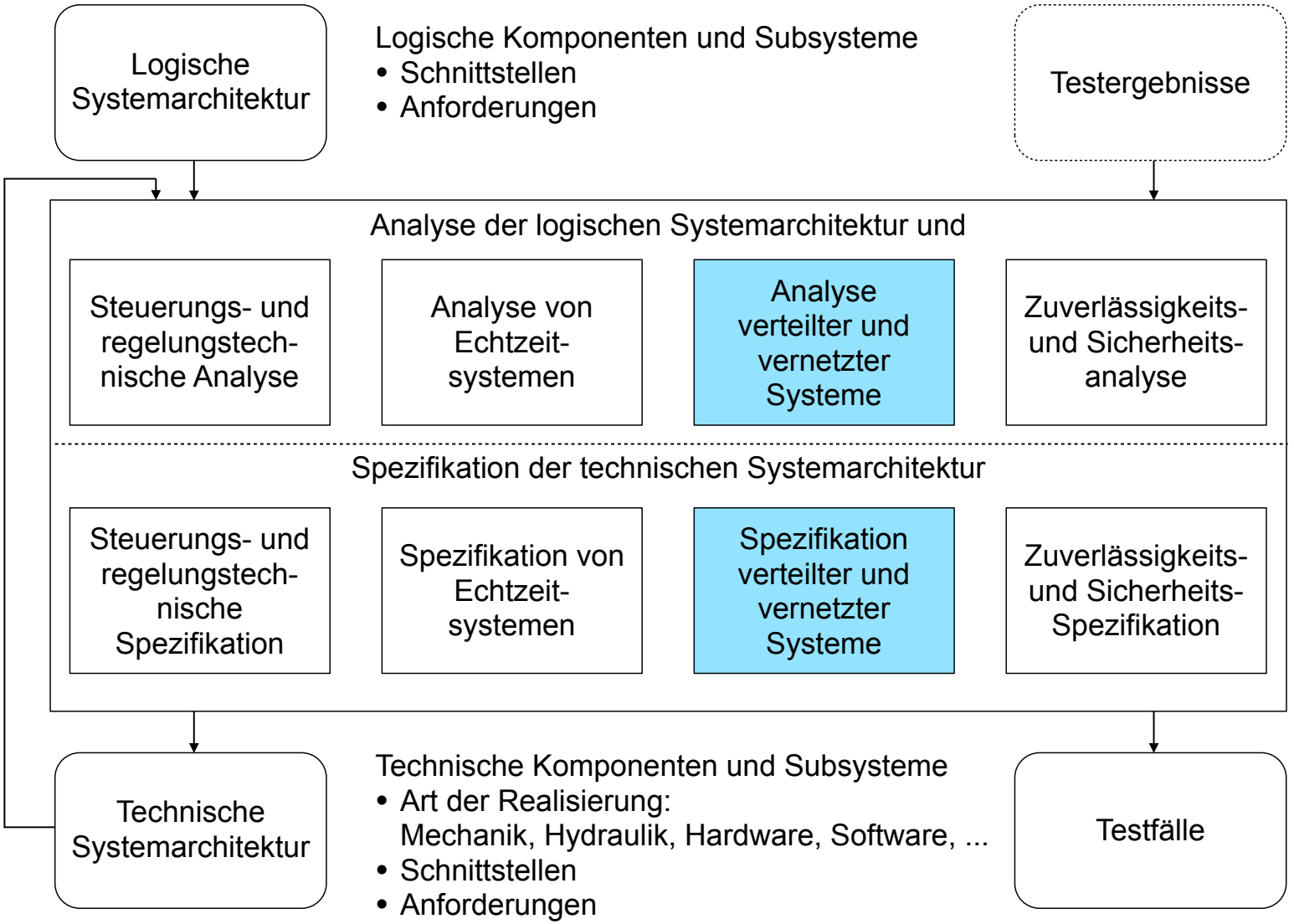


- Vorlesung Echtzeitsysteme Prof. Schürr

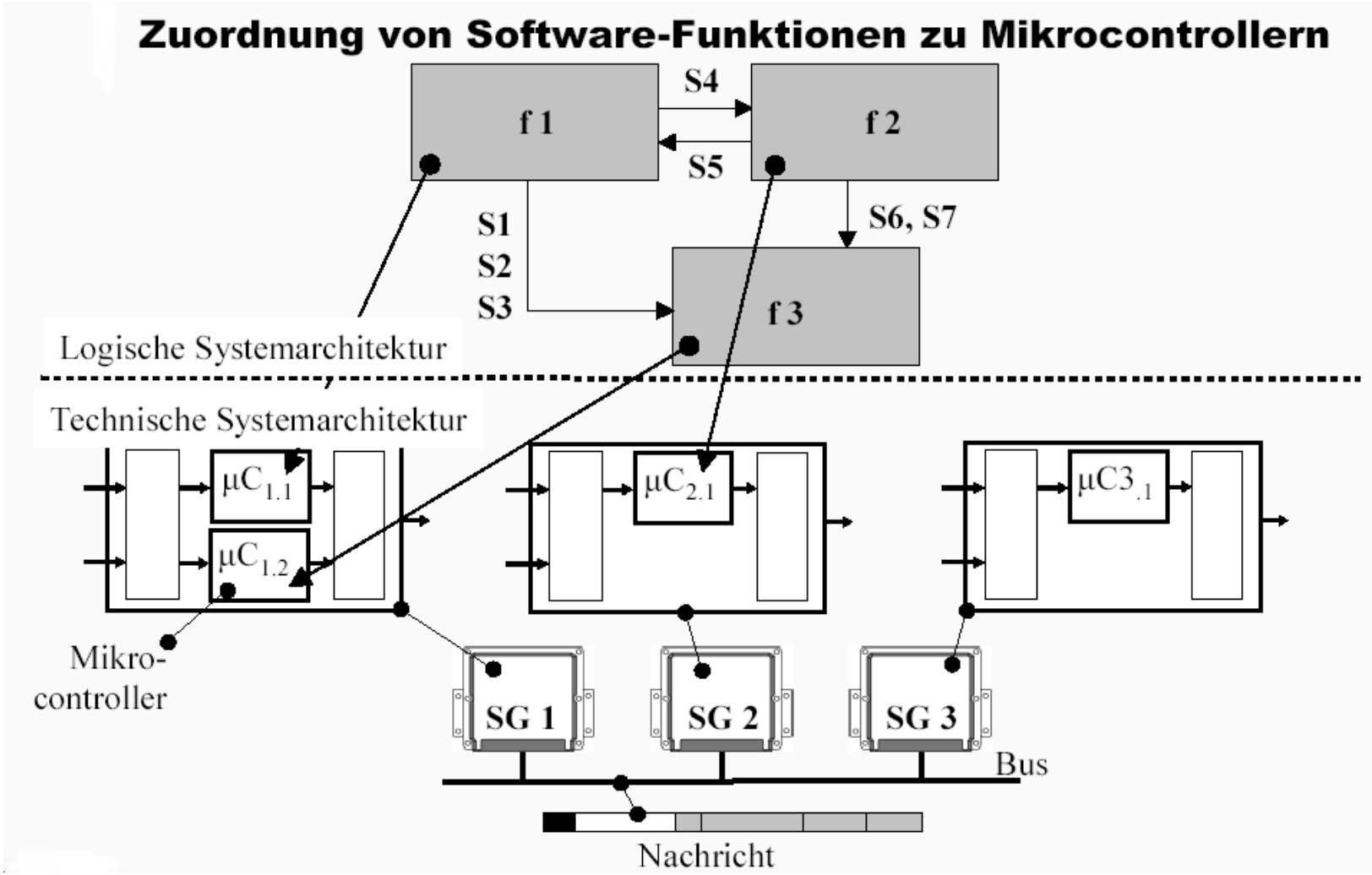
Analyse und Spezifikation von Echtzeitsystemen

- Bei Analyse und Spezifikation steuerungs- und regelungstechnischer Systeme werden Anforderungen bezüglich der Abtastrate festgelegt.
- Abtastraten bilden die Basis für Echtzeitanforderungen an Software-Funktionen.
- Bei verteiltem, vernetztem System ergeben sich aus den Abtastraten auch die Echtzeitanforderungen an das Kommunikationssystem.
- Realisierbarkeit wird mit geeigneten Methoden analysiert.
  - Bewertung technischer Realisierungsalternativen.
  - Eventuell Korrektur der Konfiguration des Echtzeitbetriebsystems.

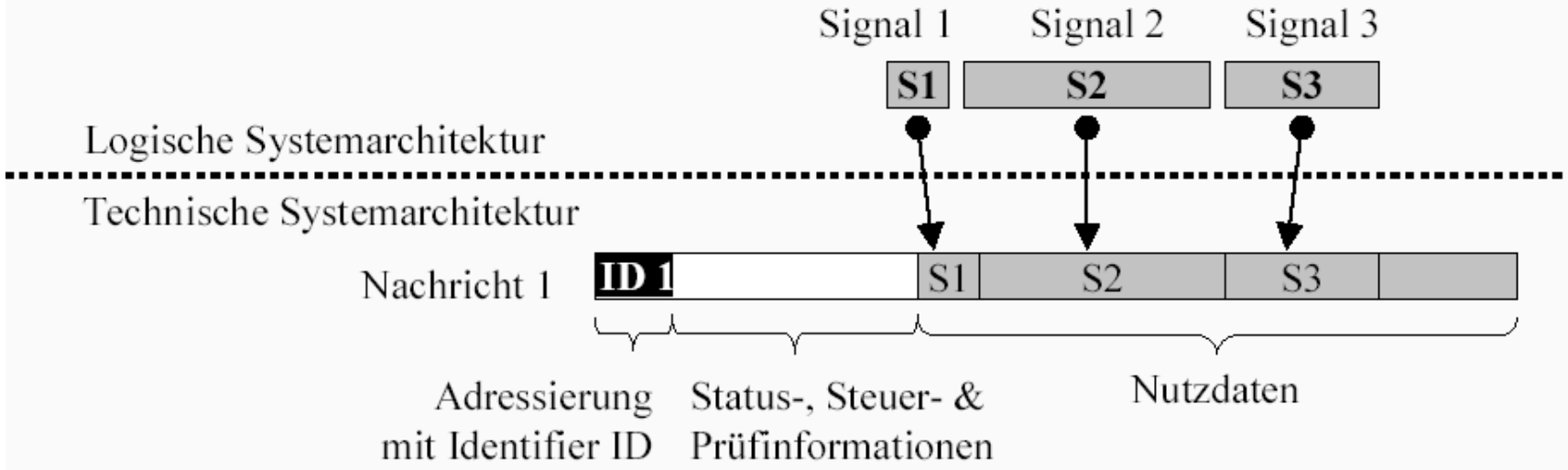
Analyse der logischen Systemarchitektur und Spezifikation der technischen Systemarchitektur (Nach Schäuffele, Zurawka)



Analyse der logischen Systemarchitektur und Spezifikation der technischen Systemarchitektur (Nach Schäuffele, Zurawka)



Zuordnung von Signalen zu Nachrichten

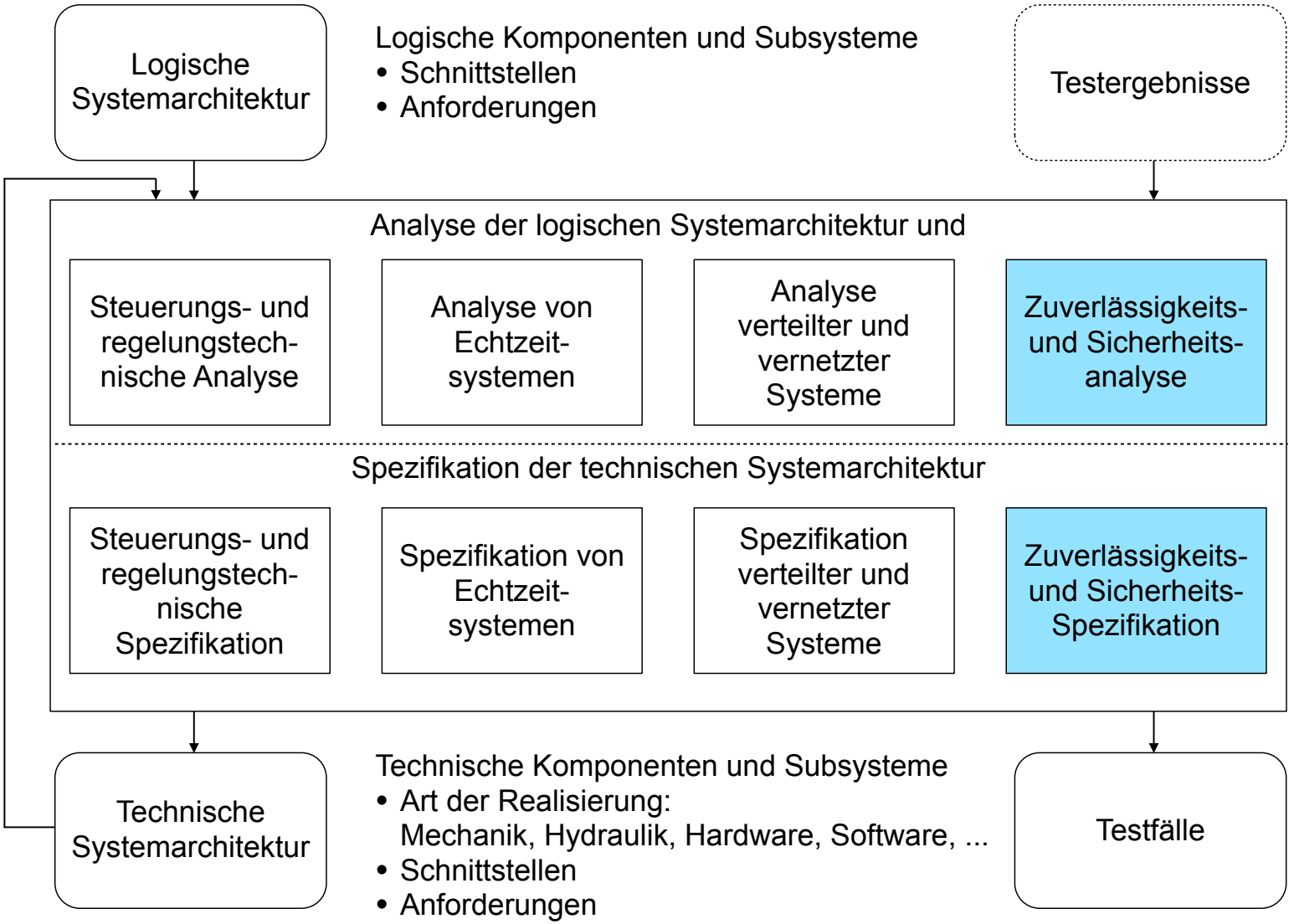


vgl. 5. E/E-Entwicklung

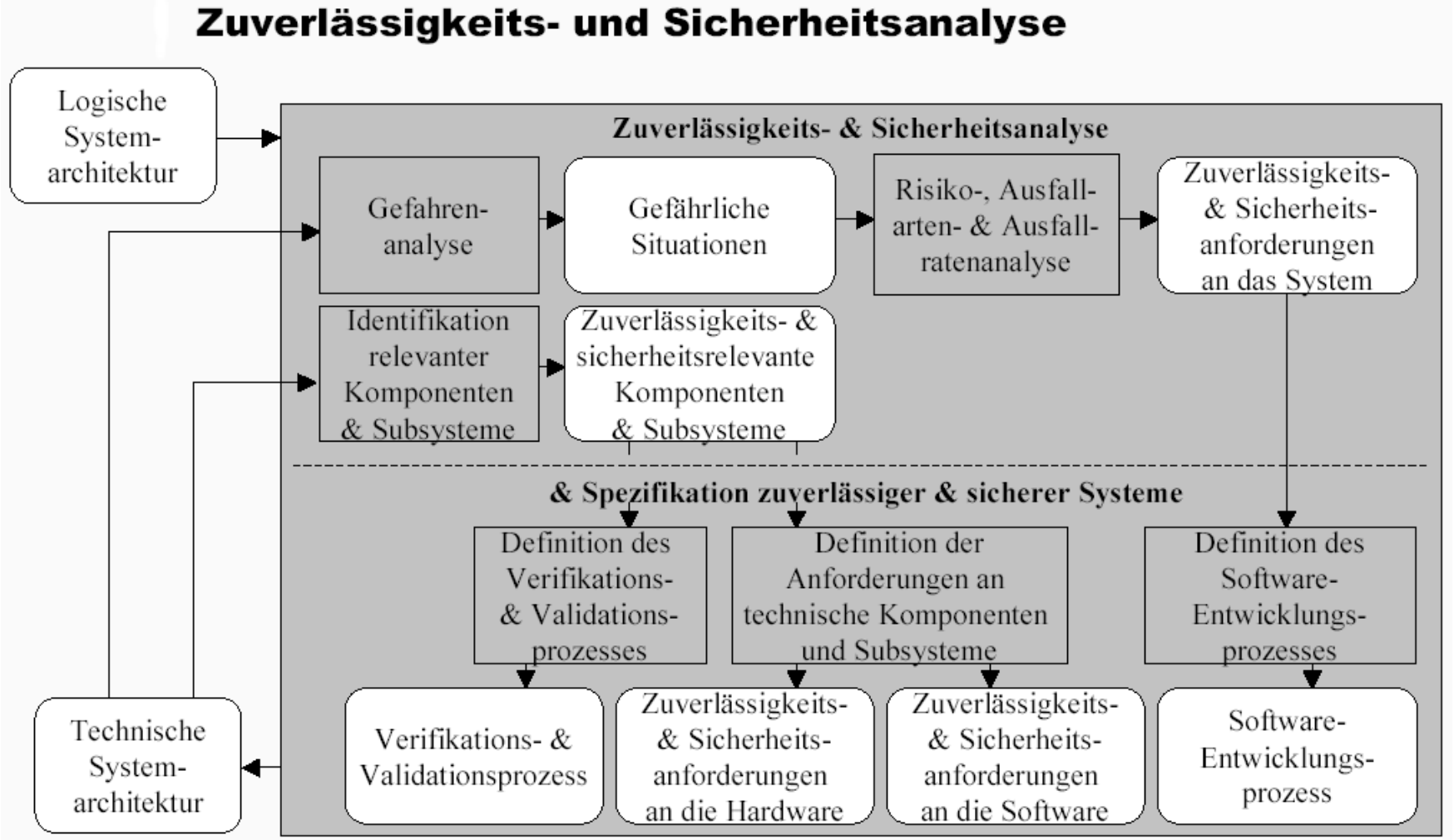
5. Bussysteme im Automobil

3. Zeitverhalten

Analyse der logischen Systemarchitektur und Spezifikation der technischen Systemarchitektur (Nach Schäuffele, Zurawka)



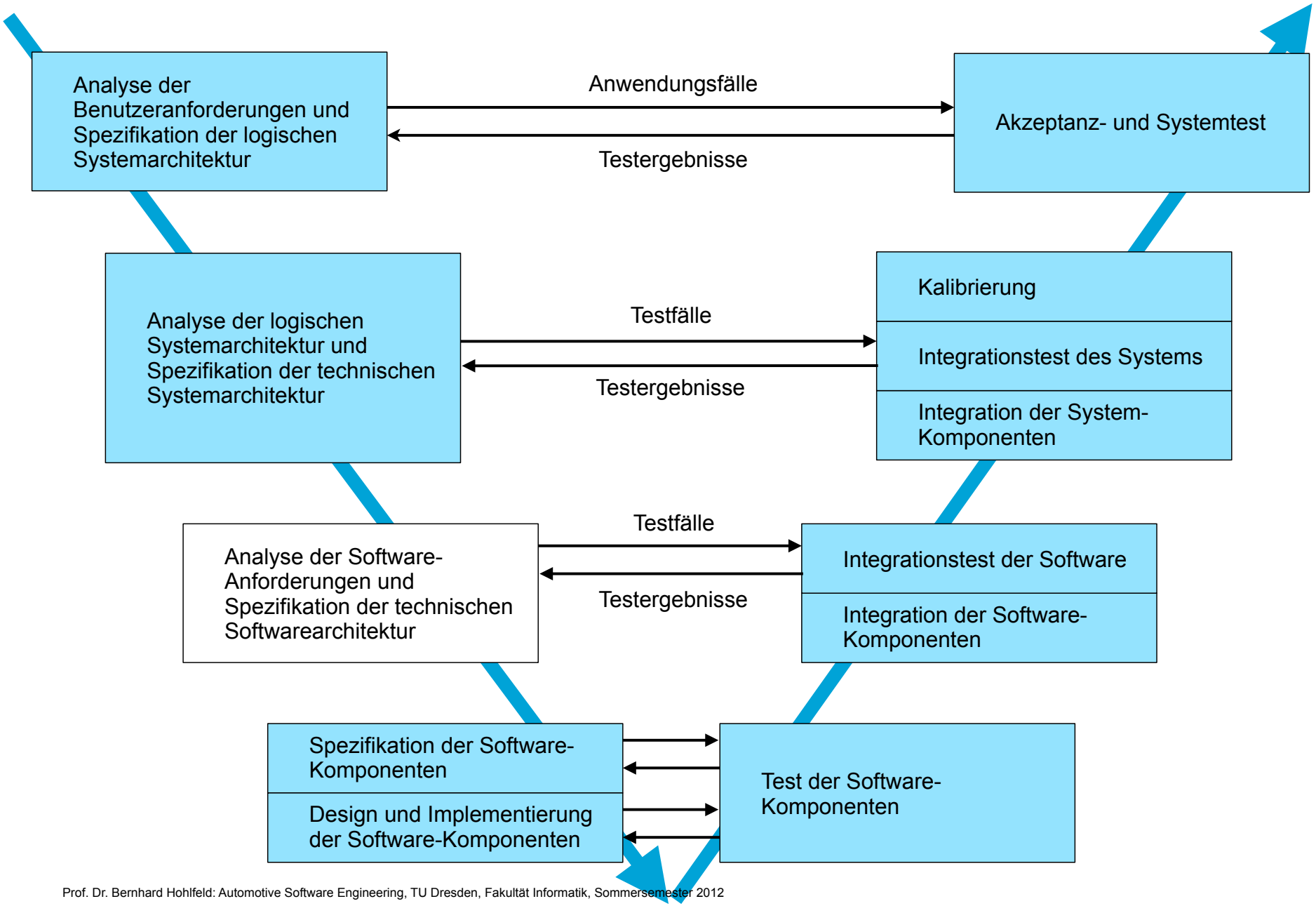
Analyse der logischen Systemarchitektur und Spezifikation der technischen Systemarchitektur



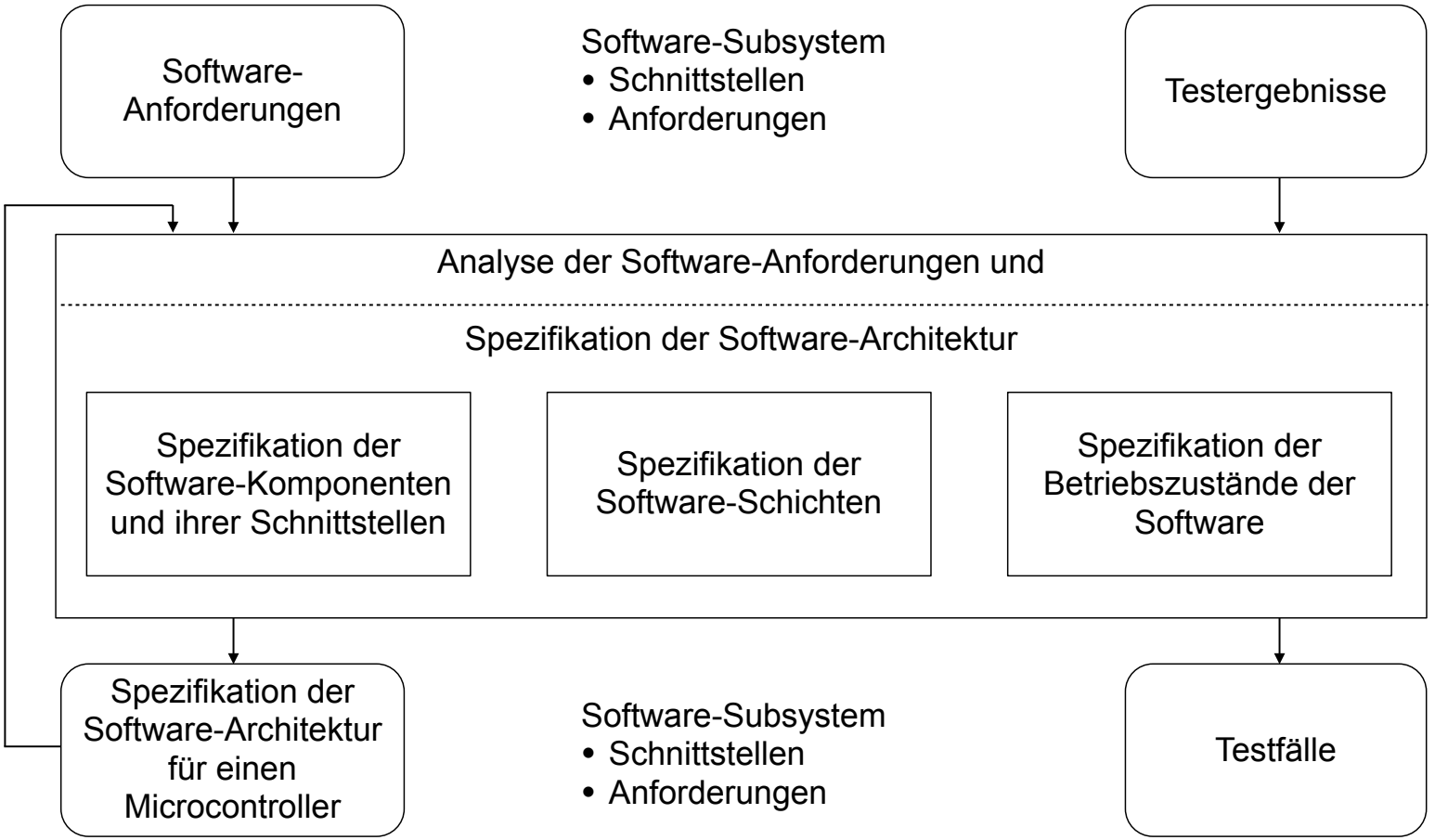
6. SW-Entwicklung / 1. Kernprozess
Kernprozess zur Entwicklung von elektronischen Systemen und Software

1. Grundbegriffe
2. Entwicklungsobjekt: Kombiinstrument
3. Analyse der Benutzeranforderungen und Spezifikation der logischen Systemarchitektur
4. Analyse der logischen Systemarchitektur und Spezifikation der technischen Systemarchitektur
5. **Analyse der Software-Anforderungen und Spezifikation der technischen Softwarearchitektur**
6. Spezifikation der Software-Komponenten
7. Design und Implementierung der Software-Komponenten
8. Test der Software-Komponenten
9. Integration der Software-Komponenten
10. Integrationstest der Software
11. Integration der System-Komponenten
12. Integrationstest des Systems
13. Kalibrierung
14. Akzeptanz- und Systemtest

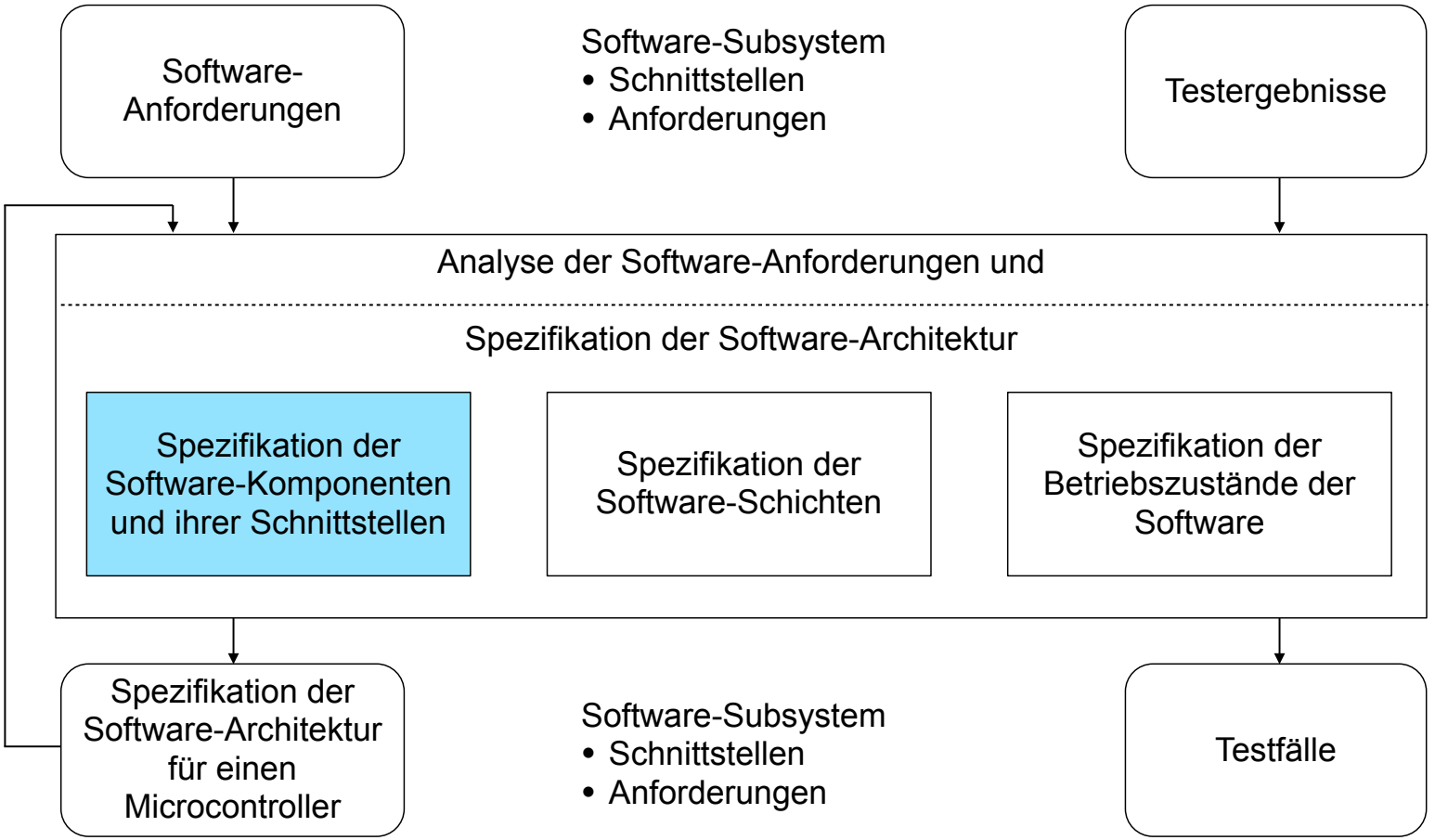
Analyse der Software-Anforderungen und Spezifikation der technischen Softwarearchitektur



Analyse der Software-Anforderungen und Spezifikation der technischen Software-Architektur (Nach Schäuffele, Zurawka)



Analyse der Software-Anforderungen und Spezifikation der technischen Software-Architektur (Nach Schäuffele, Zurawka)



Daten vs. Kontrolle

- Programmierung
  - Dateninformationen
  - Kontroll- oder Steuerinformationen
- Software- Schnittstellen
  - Datenschnittstellen
  - Kontroll- oder Steuerschnittstellen

Beispiel

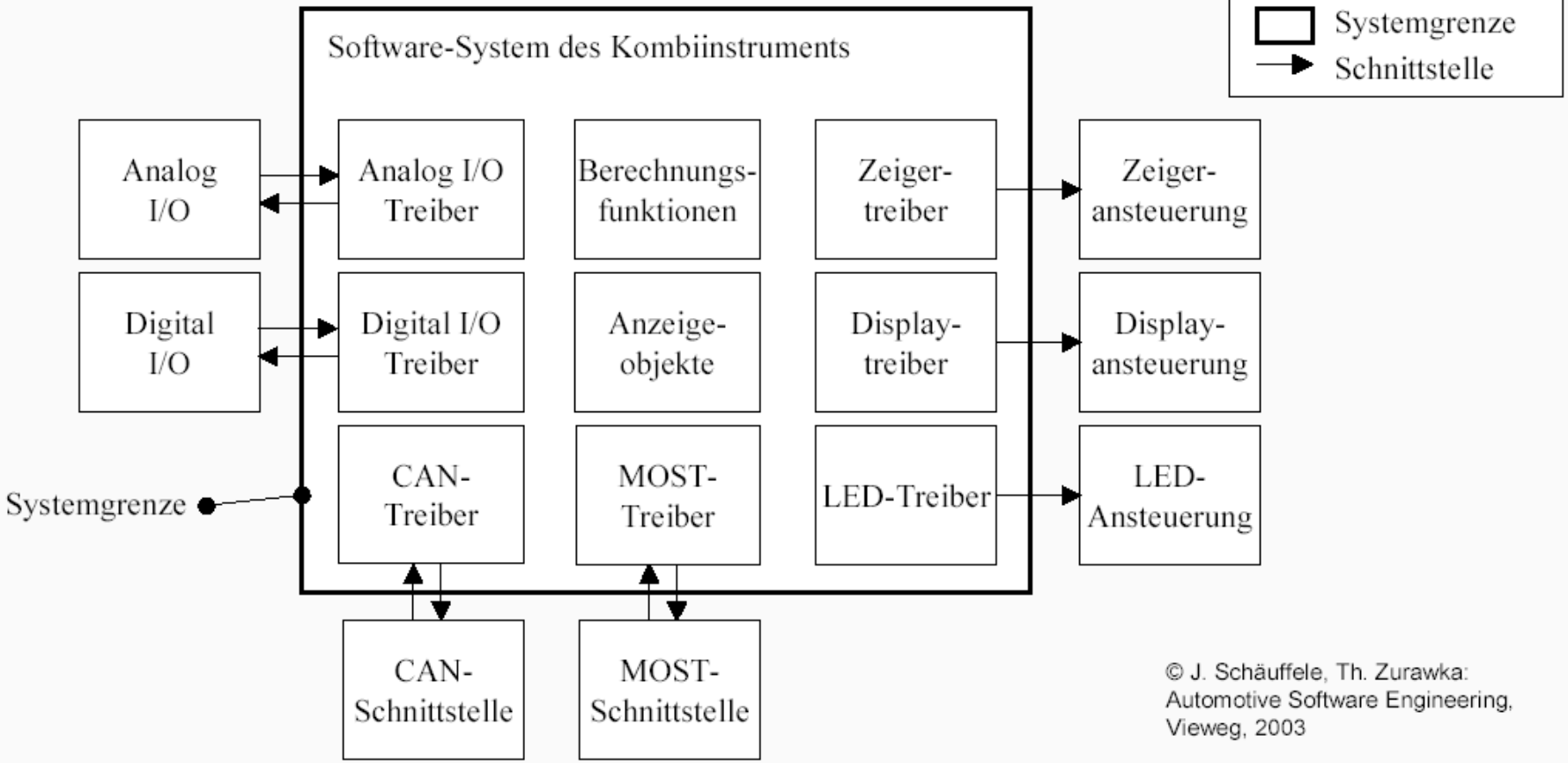
- Eintreffen einer CAN-Nachricht: Kontrollinformation
- Inhalt der Nachricht: Dateninformation

Spezifikation der On-Board-Schnittstellen

- Software-System und seine Umgebung (*Kontext*)
- *On-Board-Schnittstellen* des Steuergeräts zu Sollwertgebern, Sensoren, Aktuatoren, On-Board-Kommunikation mit anderen Steuergeräten

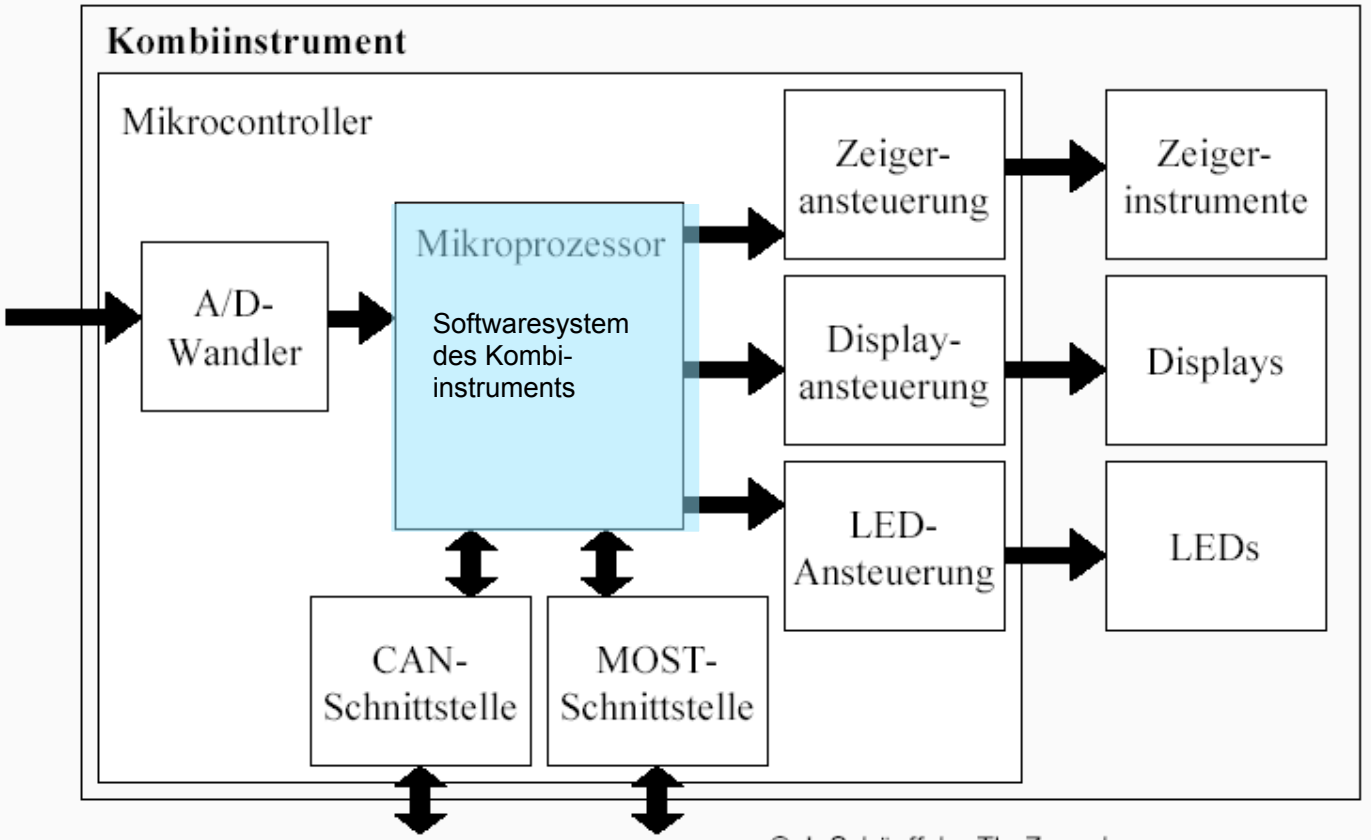
Analyse der Software-Anforderungen und Spezifikation der technischen Software-Architektur (Nach Schäuffele, Zurawka)

Kontext- und Schnittstellenmodell der Software des Kombiinstruments



© J. Schäuffele, Th. Zurawka: Automotive Software Engineering, Vieweg, 2003

Hardware des Kombiinstruments



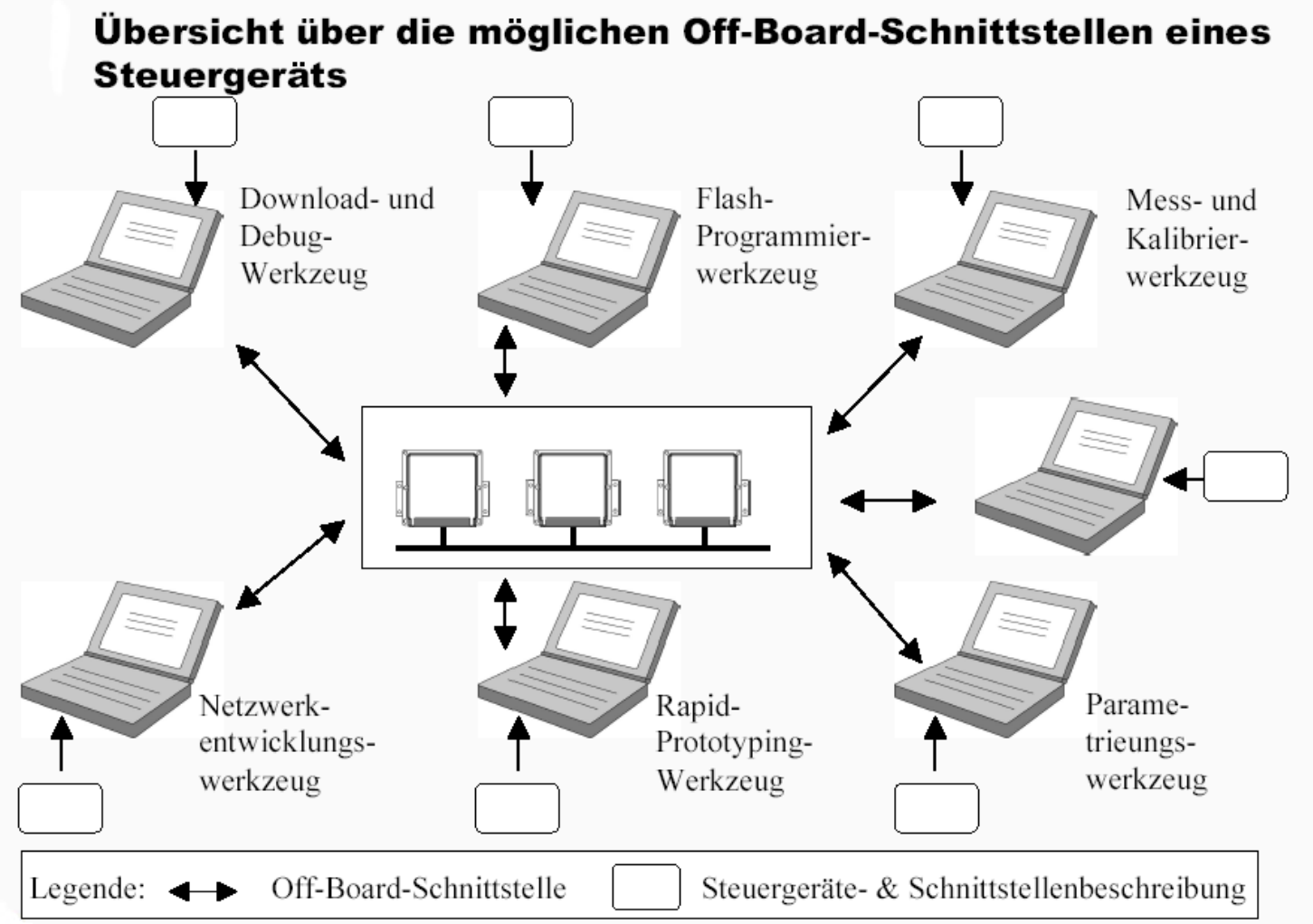
© J. Schäuffele, Th. Zurawka:
Automotive Software Engineering, Vieweg, 2003

- siehe auch 9. Integration der Software-Komponenten

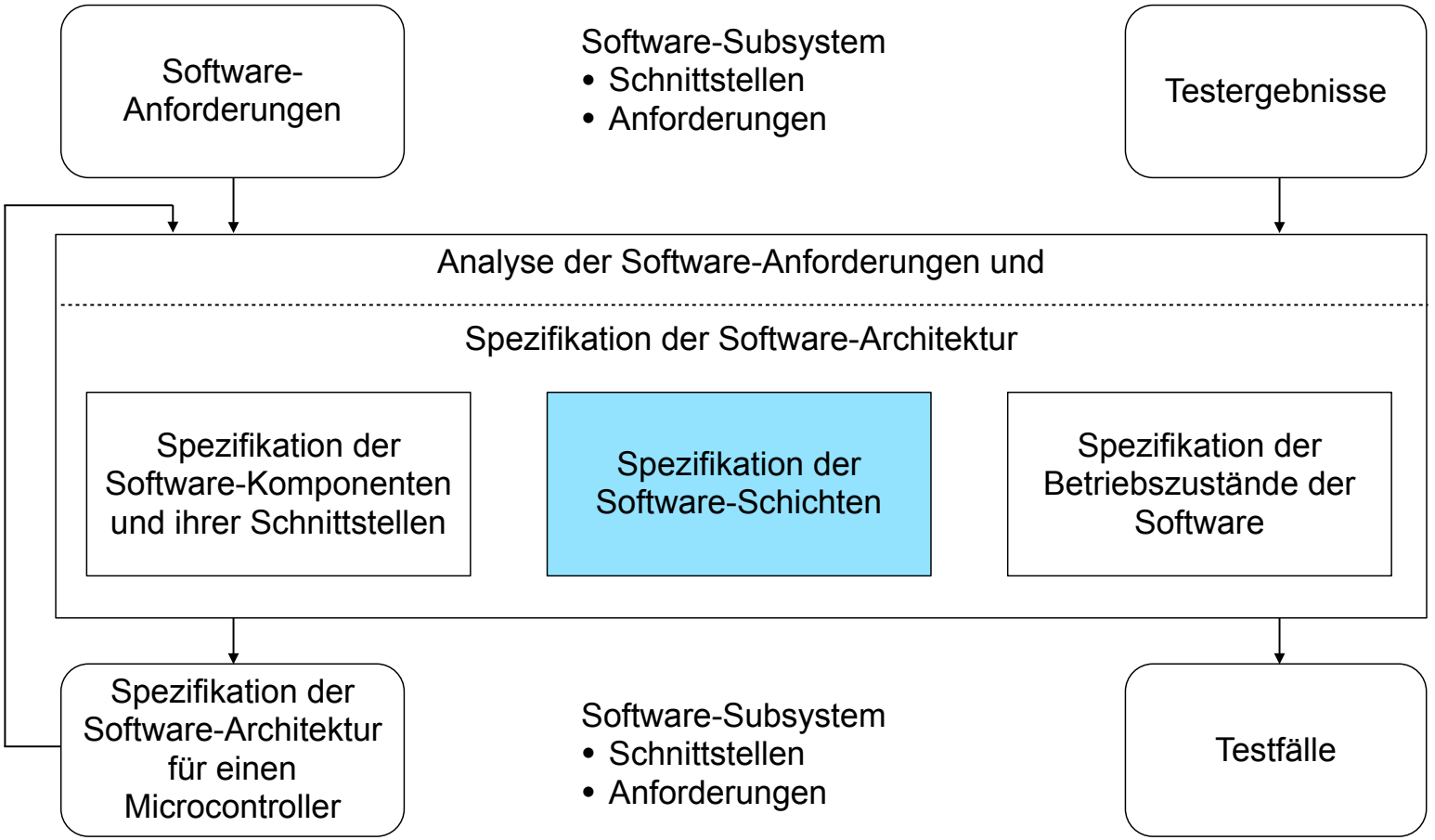
Spezifikation der Off-Board-Schnittstellen

- Software-Architektur muss alle Schnittstellen unterstützen, die während des Off-Board-Betriebs des Steuergeräts, als auch um Verlauf der Entwicklung, in Produktion und Service für die Off-Board-Kommunikation notwendig sind.
- Verschiedene Hardware- und Software-Varianten (modifizierte Off-Board-Schnittstellen)
- Verfahren für Messen, Kalibrieren, Diagnose, Flash-Programmierung in ASAM
- Beschreibungsdateien für Off-Board-Schnittstellen

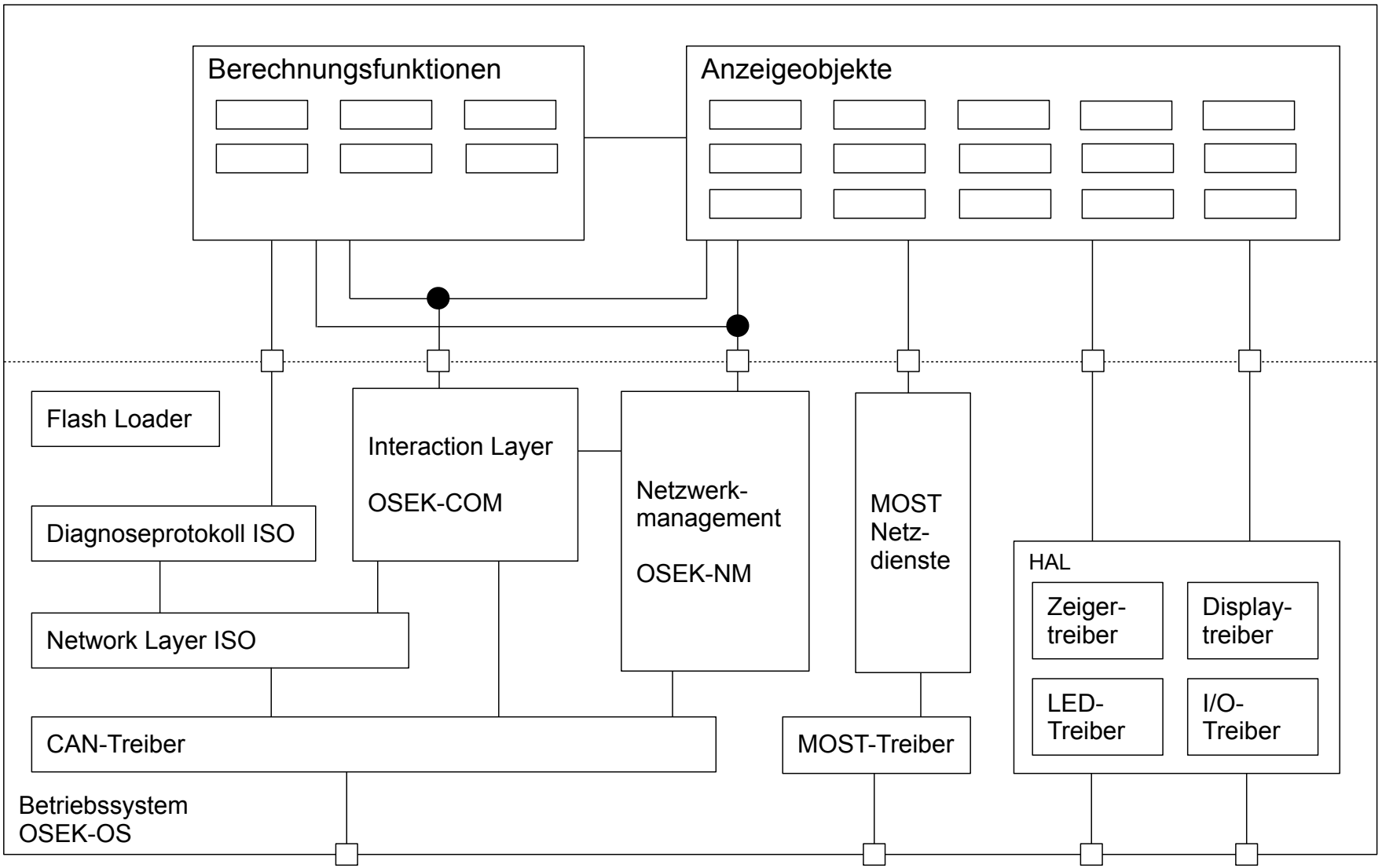
Analyse der Software-Anforderungen und Spezifikation der technischen Software-Architektur (Nach Schäuffele, Zurawka)



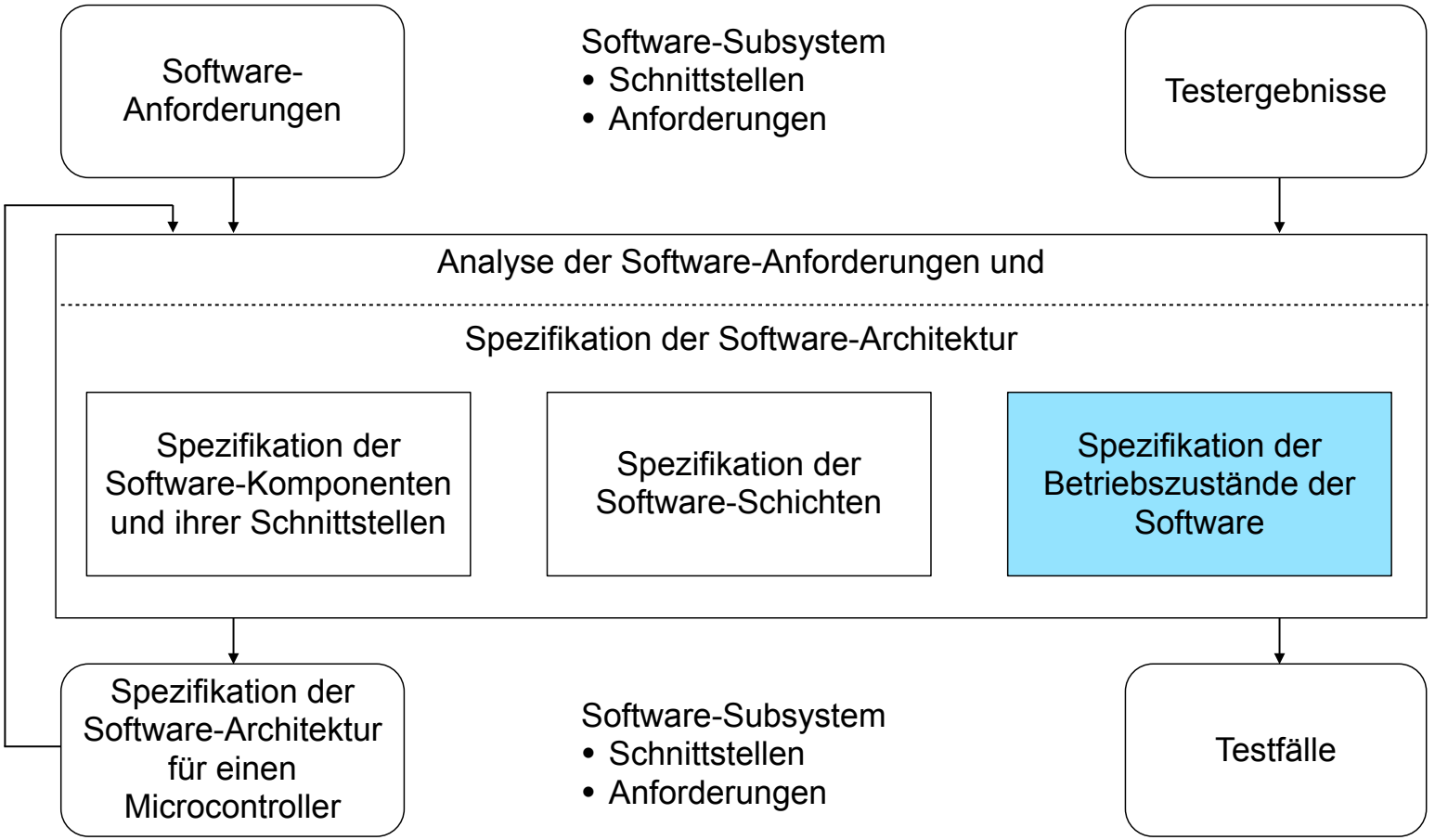
Analyse der Software-Anforderungen und Spezifikation der technischen Software-Architektur (Nach Schäuffele, Zurawka)



Softwarearchitektur des Kombiinstrumentes (Nach Schäuffele, Zurawka)



Analyse der Software-Anforderungen und Spezifikation der technischen Software-Architektur (Nach Schäuffele, Zurawka)

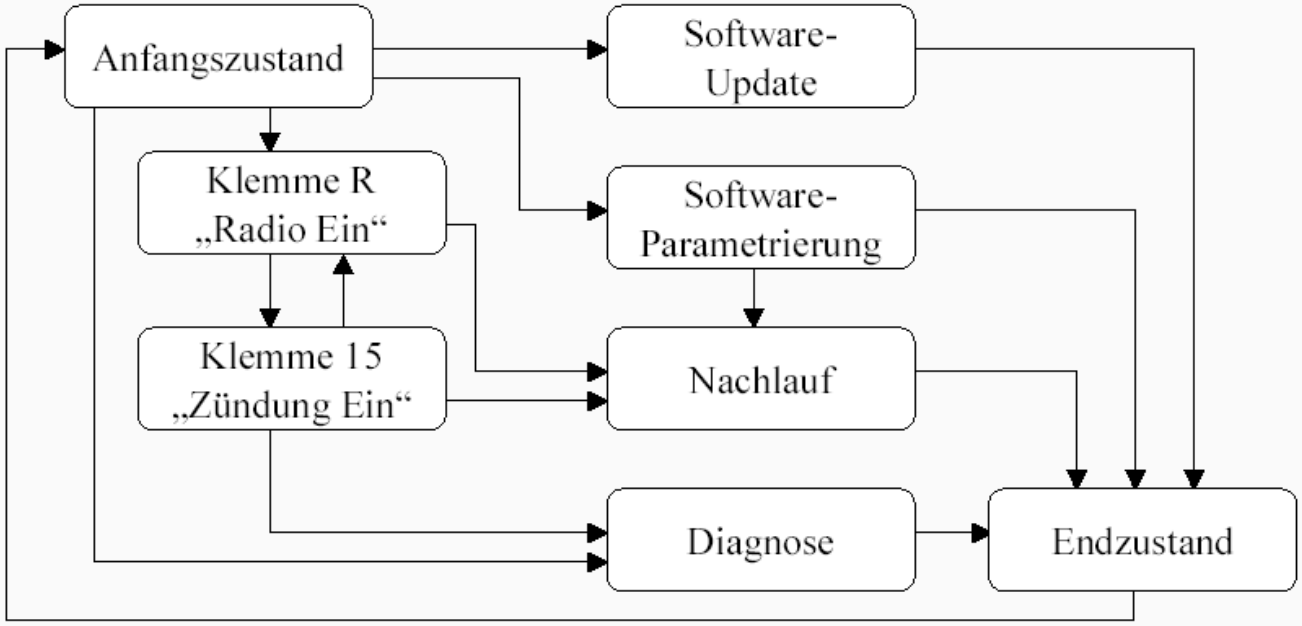


Betriebszustände

- Normalbetrieb
- Betriebszustand für Diagnose oder Update
- Notlauf

Spezifikation durch Zustandsautomaten

Betriebszustände und Übergänge für das Kombiinstrument

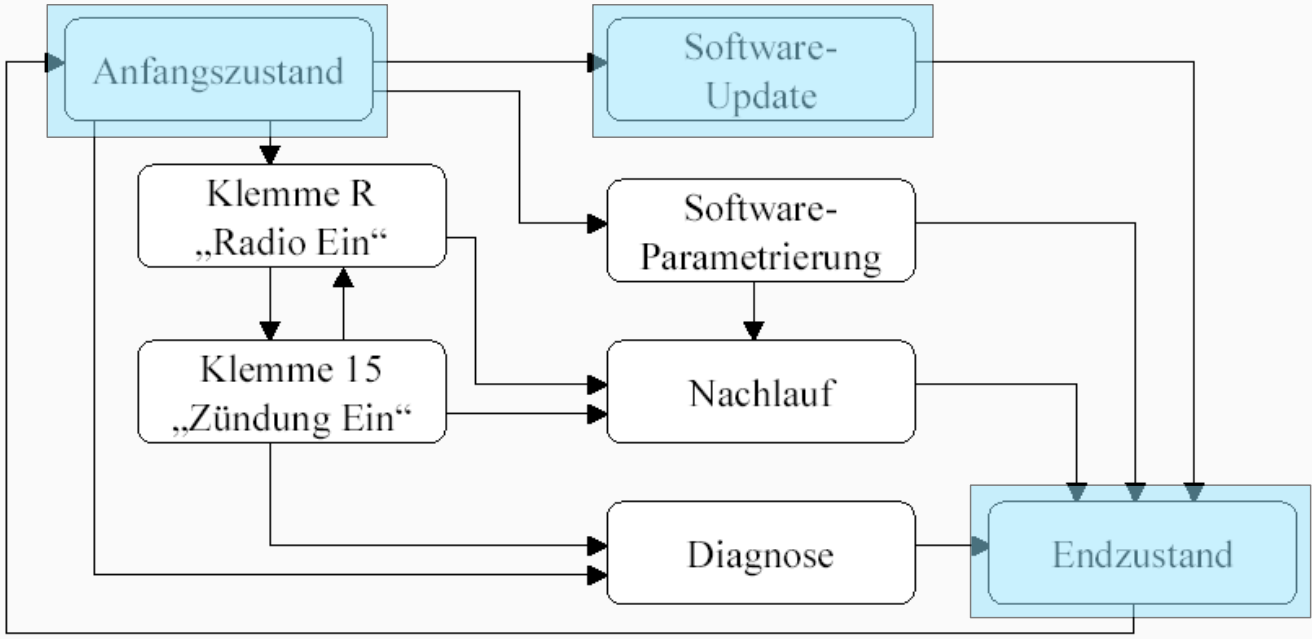


Betriebszustände

- Normalbetrieb
- Betriebszustand für Diagnose oder Update
- Notlauf

Spezifikation durch Zustandsautomaten

Betriebszustände und Übergänge für das Kombiinstrument

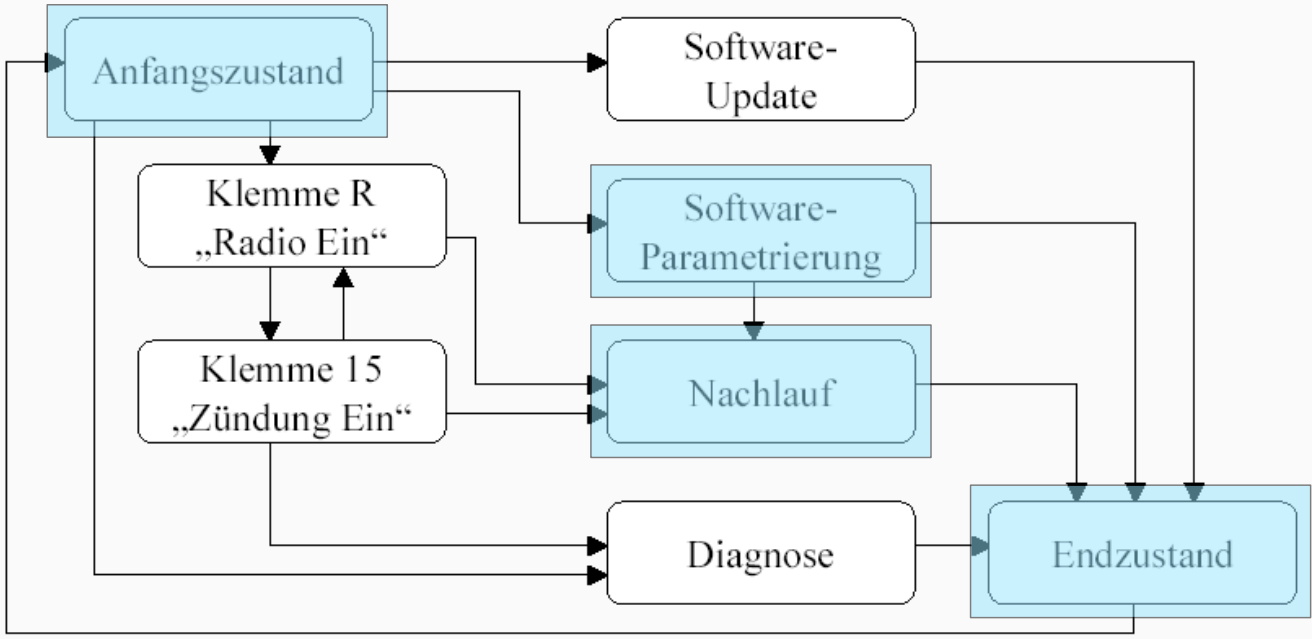


Betriebszustände

- Normalbetrieb
- Betriebszustand für Diagnose oder Update
- Notlauf

Spezifikation durch Zustandsautomaten

Betriebszustände und Übergänge für das Kombiinstrument

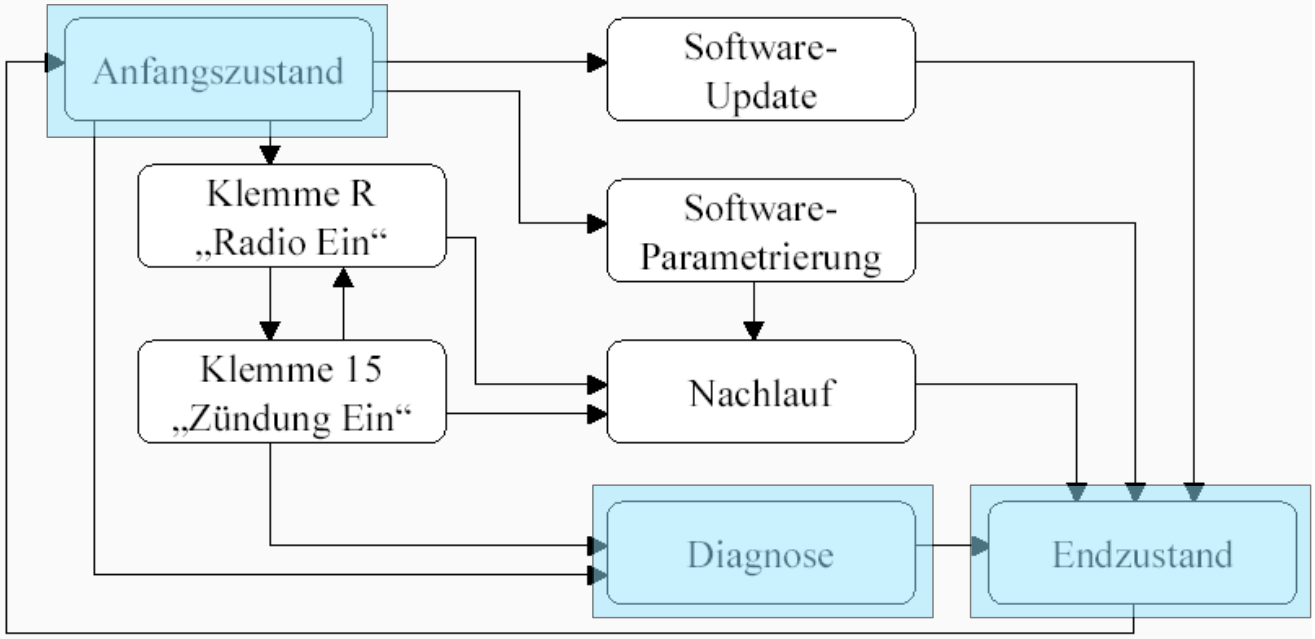


Betriebszustände

- Normalbetrieb
- Betriebszustand für Diagnose oder Update
- Notlauf

Spezifikation durch Zustandsautomaten

Betriebszustände und Übergänge für das Kombiinstrument

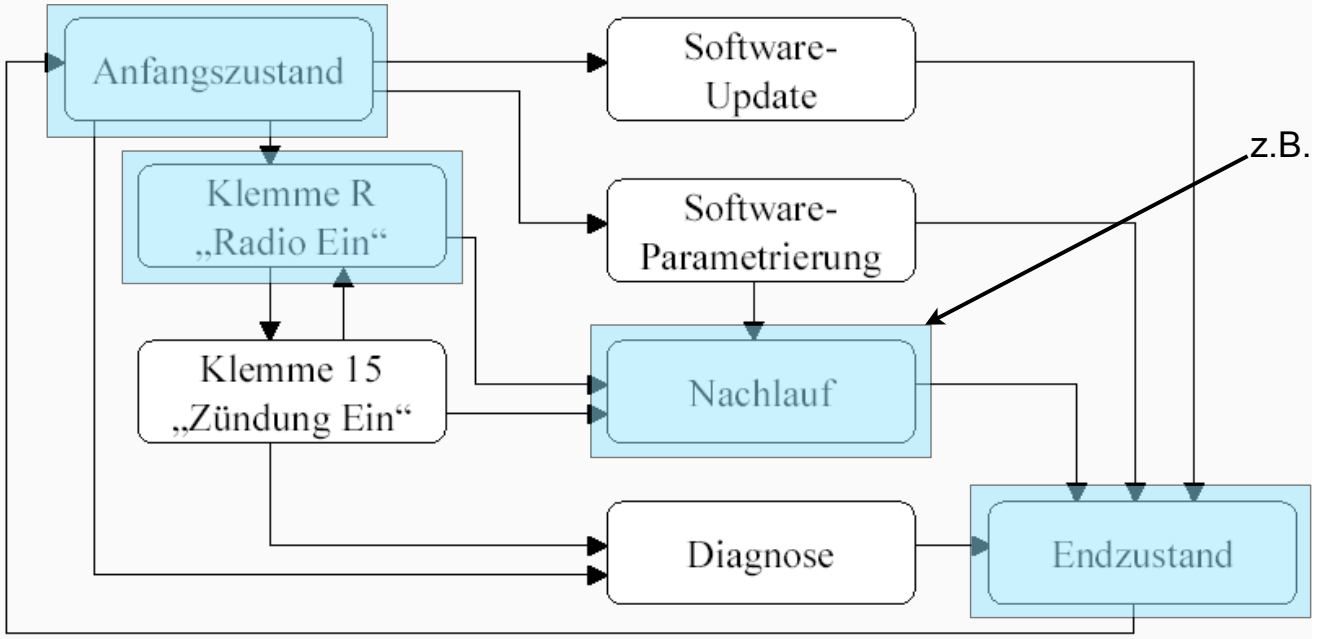


Betriebszustände

- Normalbetrieb
- Betriebszustand für Diagnose oder Update
- Notlauf

Spezifikation durch Zustandsautomaten

Betriebszustände und Übergänge für das Kombiinstrument



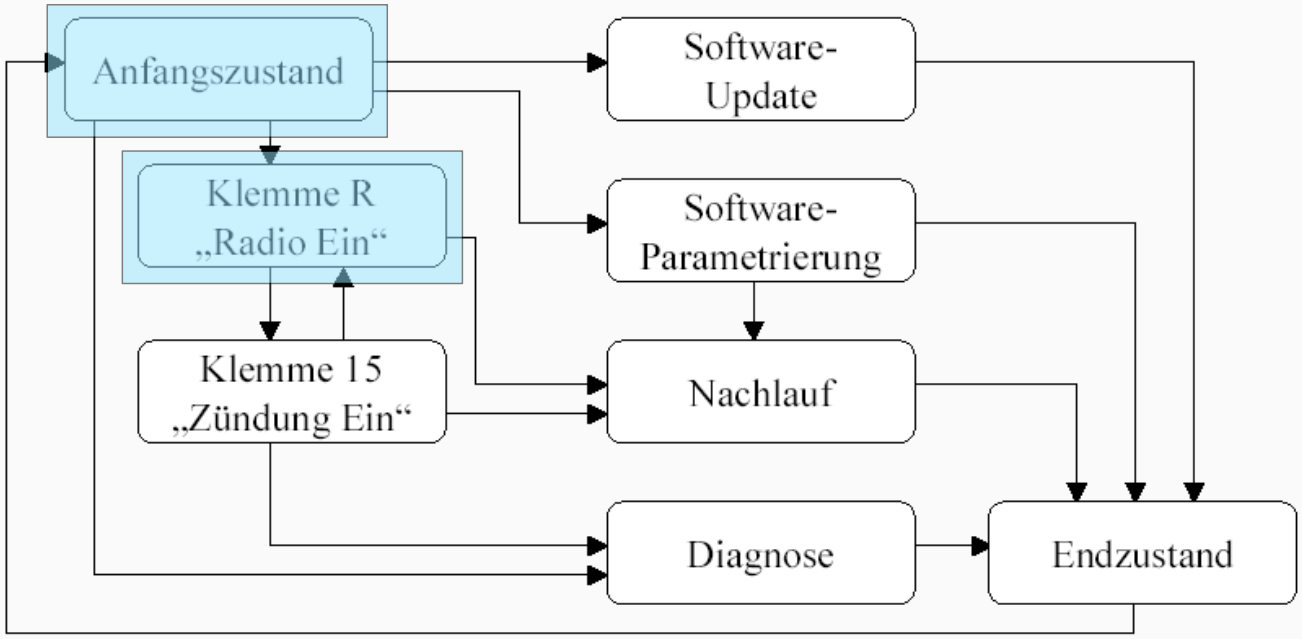
z.B. Merken des Senders

Betriebszustände

- Normalbetrieb
- Betriebszustand für Diagnose oder Update
- Notlauf

Spezifikation durch Zustandsautomaten

Betriebszustände und Übergänge für das Kombiinstrument

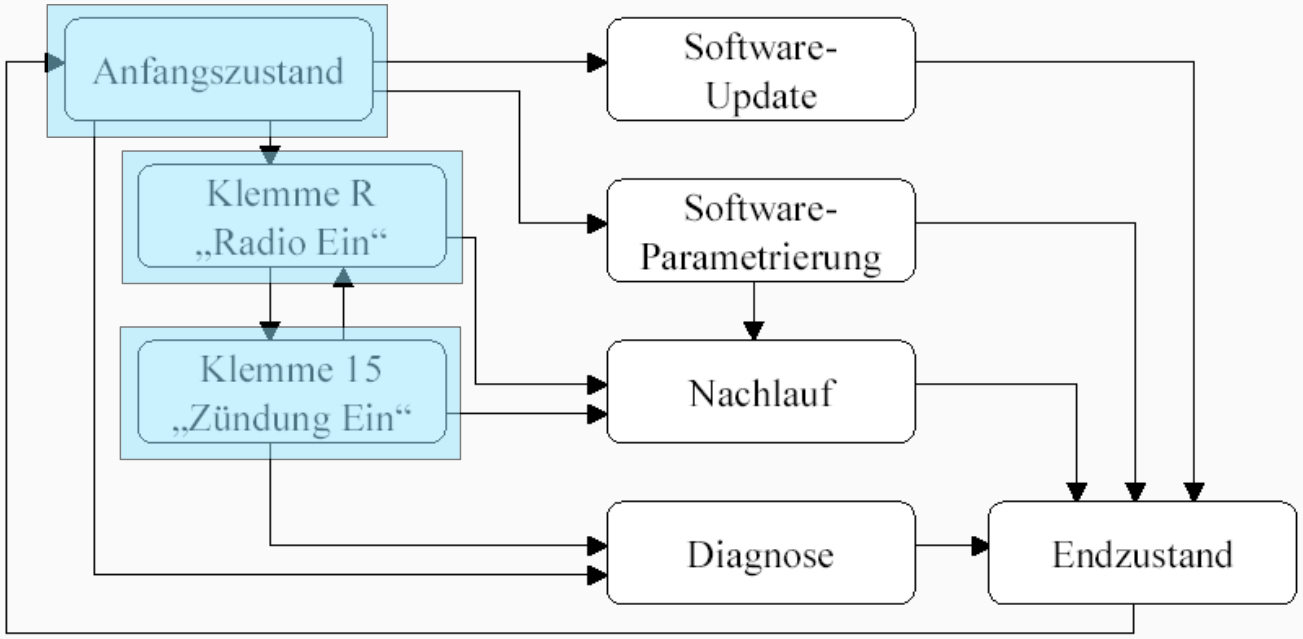


Betriebszustände

- Normalbetrieb
- Betriebszustand für Diagnose oder Update
- Notlauf

Spezifikation durch Zustandsautomaten

Betriebszustände und Übergänge für das Kombiinstrument



Betriebszustände

- Normalbetrieb
- Betriebszustand für Diagnose oder Update
- Notlauf

Spezifikation durch Zustandsautomaten

Betriebszustände und Übergänge für das Kombiinstrument