INFORMATIK ▪ CONSULTING ▪ SYSTEMS AG

Vorlesung „Embedded Software-Engineering im Bereich Automotive“

Technische Universität Dresden, Fakultät Informatik,
Professur Softwaretechnologie

Sommersemester 2010

Dr. rer. nat. Bernhard Hohlfeld

bernhard.hohlfeld@daad-alumni.de

Inhalt

Inhalt

4. Kernprozess zur Entwicklung von elektronischen Systemen und Software

1. Grundbegriffe
2. Entwicklungsobjekt: Kombiinstrument
3. Analyse und Spezifikation der Benutzeranforderungen
4. Analyse und Spezifikation der technischen Anforderungen
5. Analyse und Spezifikation der Software-Anforderungen
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-Komponenten
11. Integration der System-Komponenten
12. Integrationstest des Systems
13. Kalibrierung
14. Akzeptanz- und Systemtest

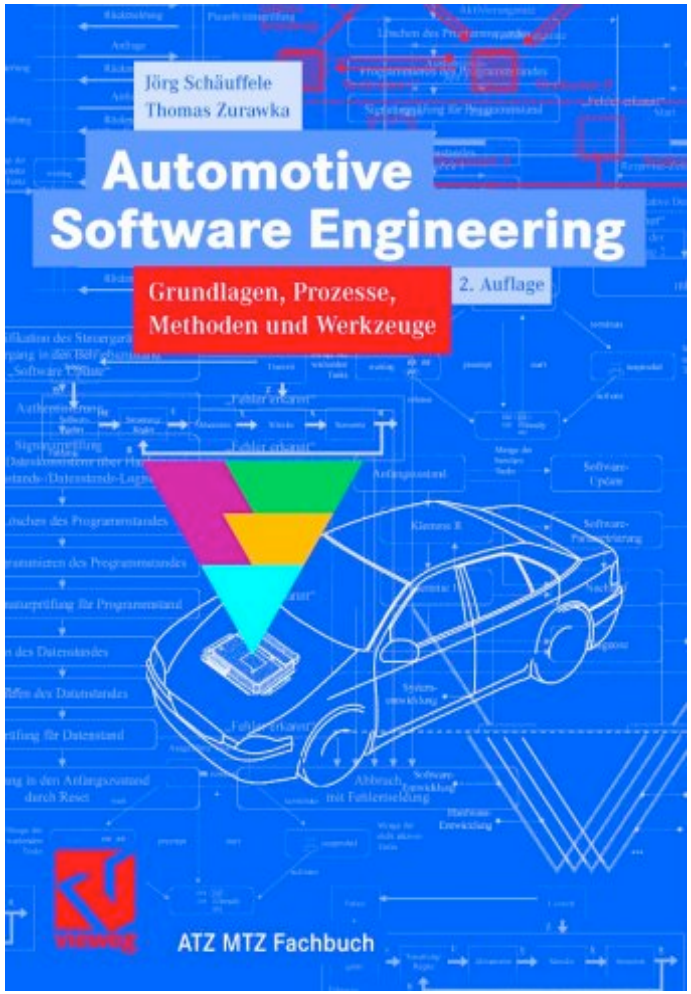
4. Kernprozess zur Entwicklung von elektronischen Systemen und Software

1. Grundbegriffe

2. Entwicklungsobjekt: Kombiinstrument
3. Analyse und Spezifikation der Benutzeranforderungen
4. Analyse und Spezifikation der technischen Anforderungen
5. Analyse und Spezifikation der Software-Anforderungen
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-Komponenten
11. Integration der System-Komponenten
12. Integrationstest des Systems
13. Kalibrierung
14. Akzeptanz- und Systemtest

Quelle

Kapitel 4



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
    - Automotive
    - Luftfahrt
    - Bahnen
    - Medizin
  - Hoher Bezug zu realer Umwelt
  - Systeme 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
- 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

Systems Engineering (1)

Systems Engineering ist die gezielte Anwendung von wissenschaftlichen und technischen Ressourcen

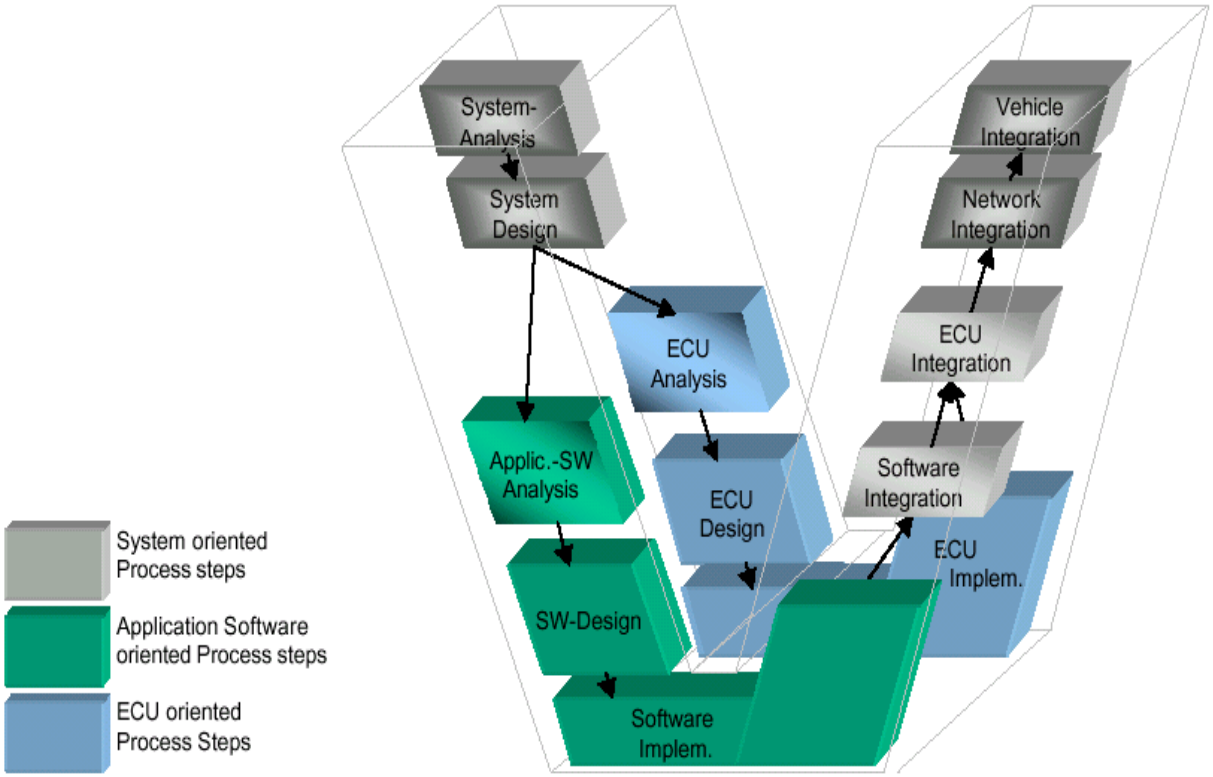
- Zur Transformation eines operationellen Bedürfnisses in die Beschreibung einer Systemkonfiguration unter bestmöglicher Berücksichtigung aller operativen Anforderungen und nach den Maßstäben der gebotenen Effektivität.
- Zur Integration aller technischen Parameter und zur Sicherstellung der Kompatibilität 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, Bezug zu CMMI http://www.sei.cmu.edu/cmmi und INCOSE http://www.incose.org)

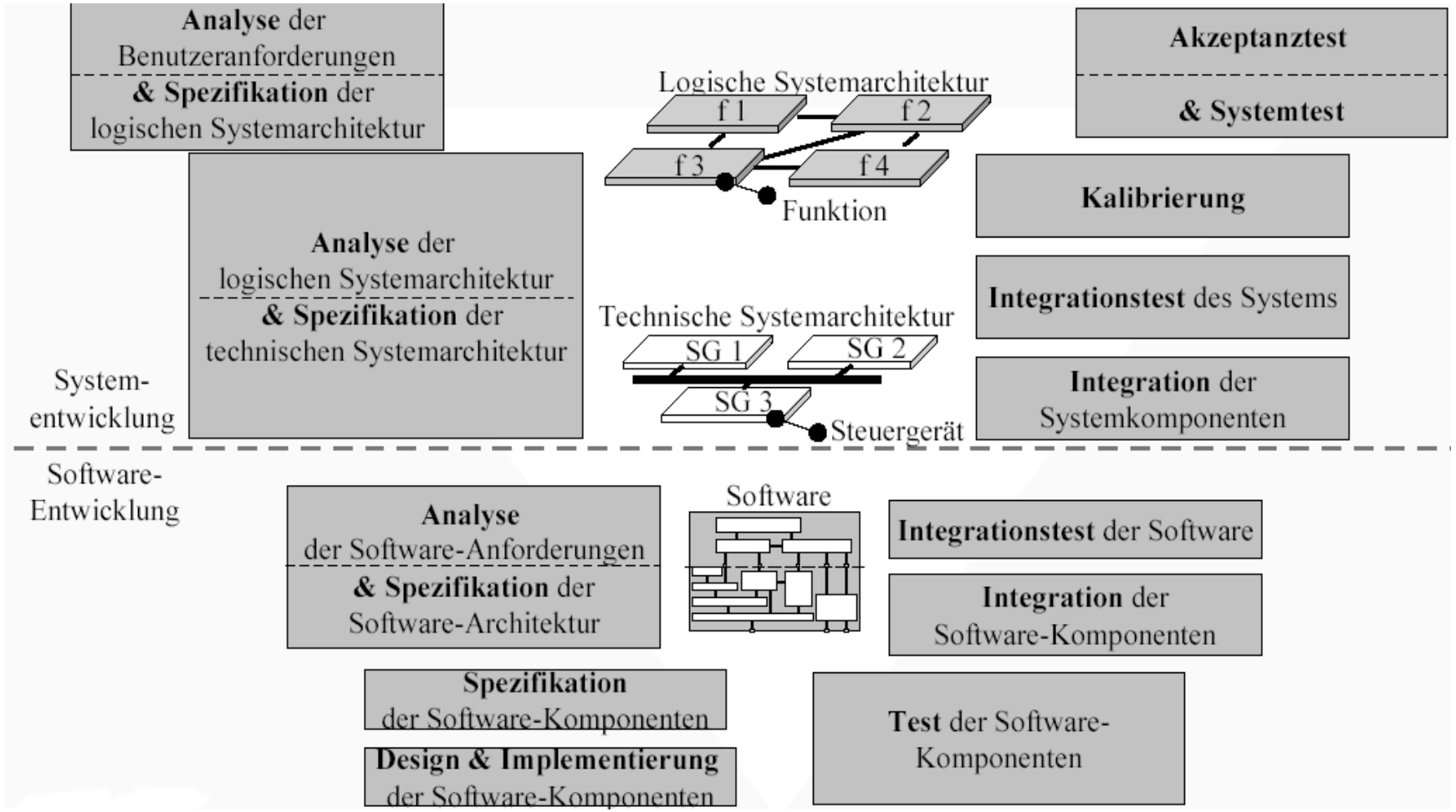
Systems Engineering (2)

Fachdisziplinen innerhalb Systems Engineering

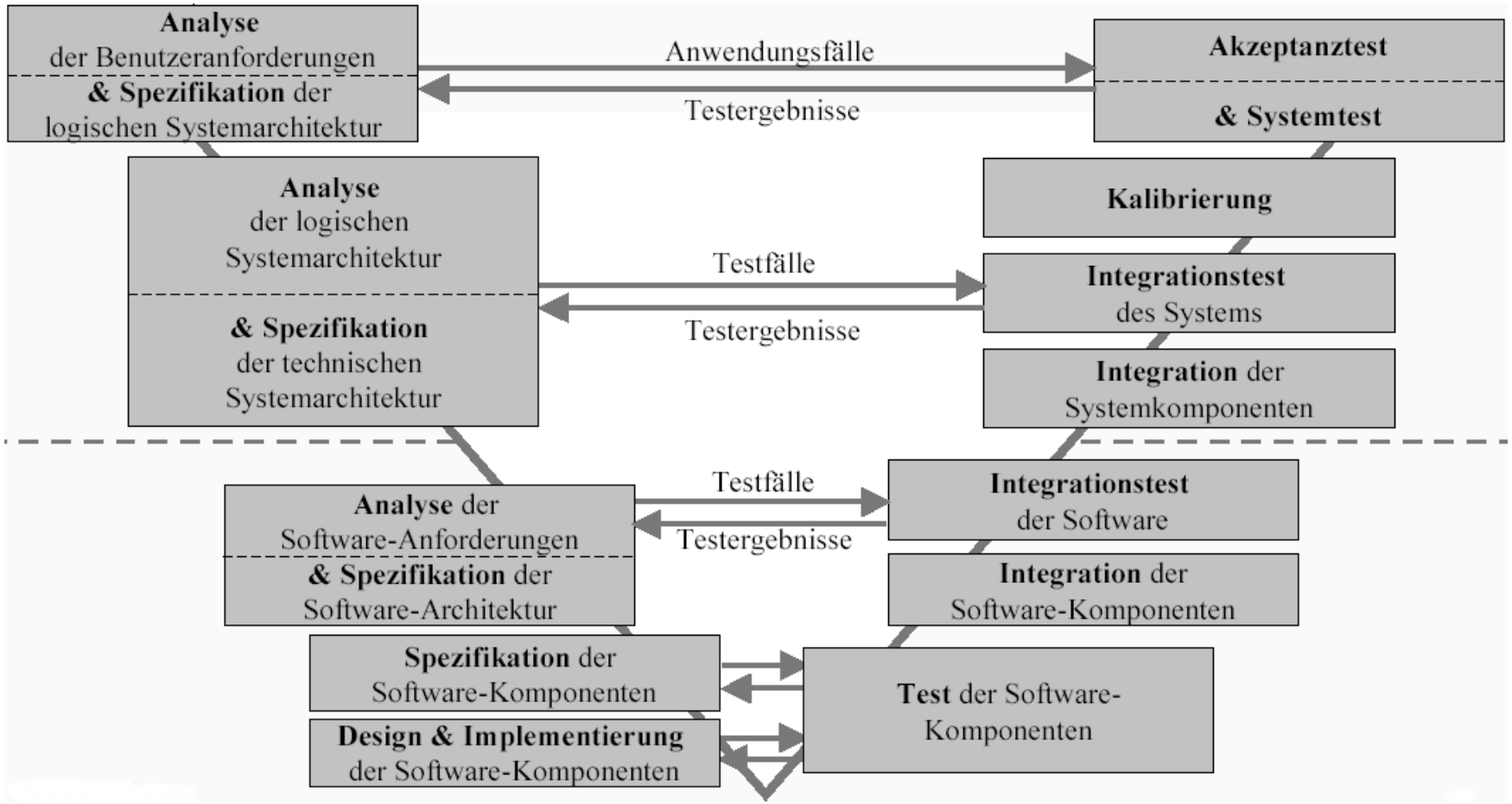
- Software-Entwicklung
- Hardware-Entwicklung
- Sensorik
- Aktuatorik
- ...



Übersicht V-Modell



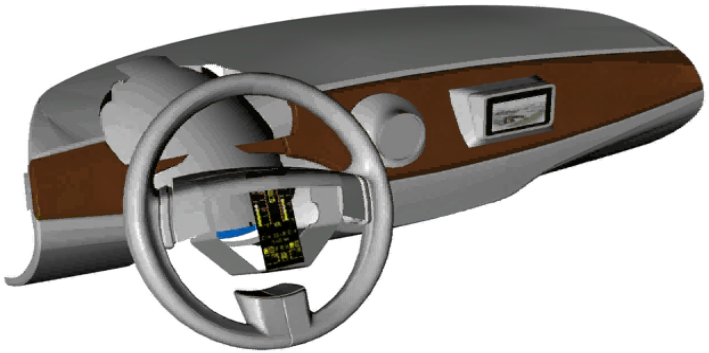
Übersicht V-Modell

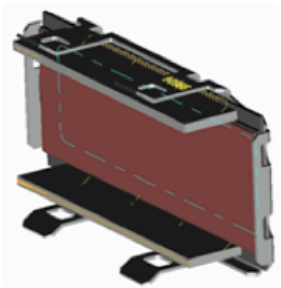


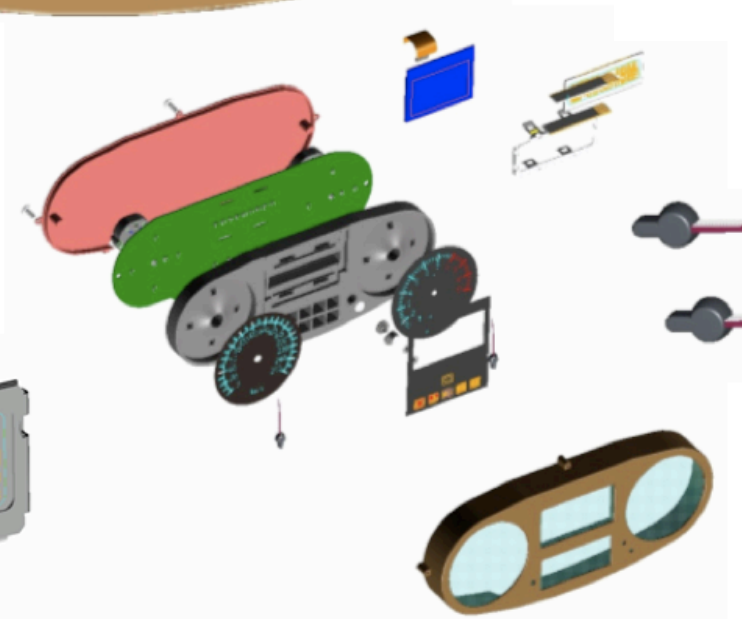
4. Kernprozess zur Entwicklung von elektronischen Systemen und Software

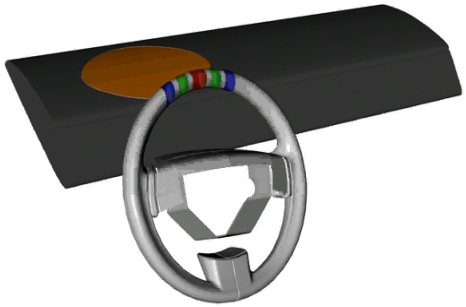
1. Grundbegriffe
2. **Entwicklungsobjekt: Kombiinstrument**
3. Analyse und Spezifikation der Benutzeranforderungen
4. Analyse und Spezifikation der technischen Anforderungen
5. Analyse und Spezifikation der Software-Anforderungen
6. Spezifikation der Software-Komponenten
7. Design und Implementierung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

Entwicklungsobjekt: Kombiinstrument





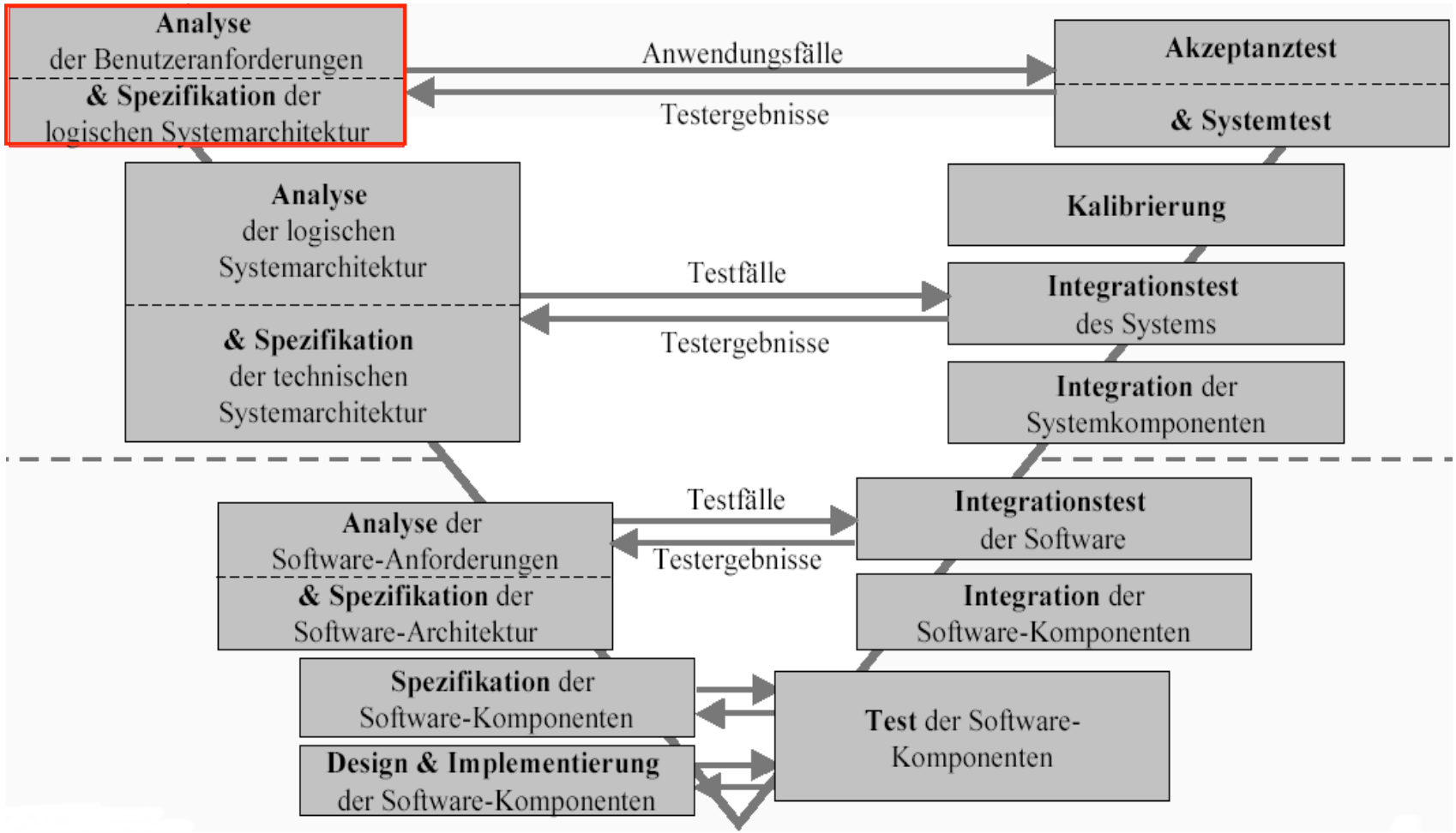




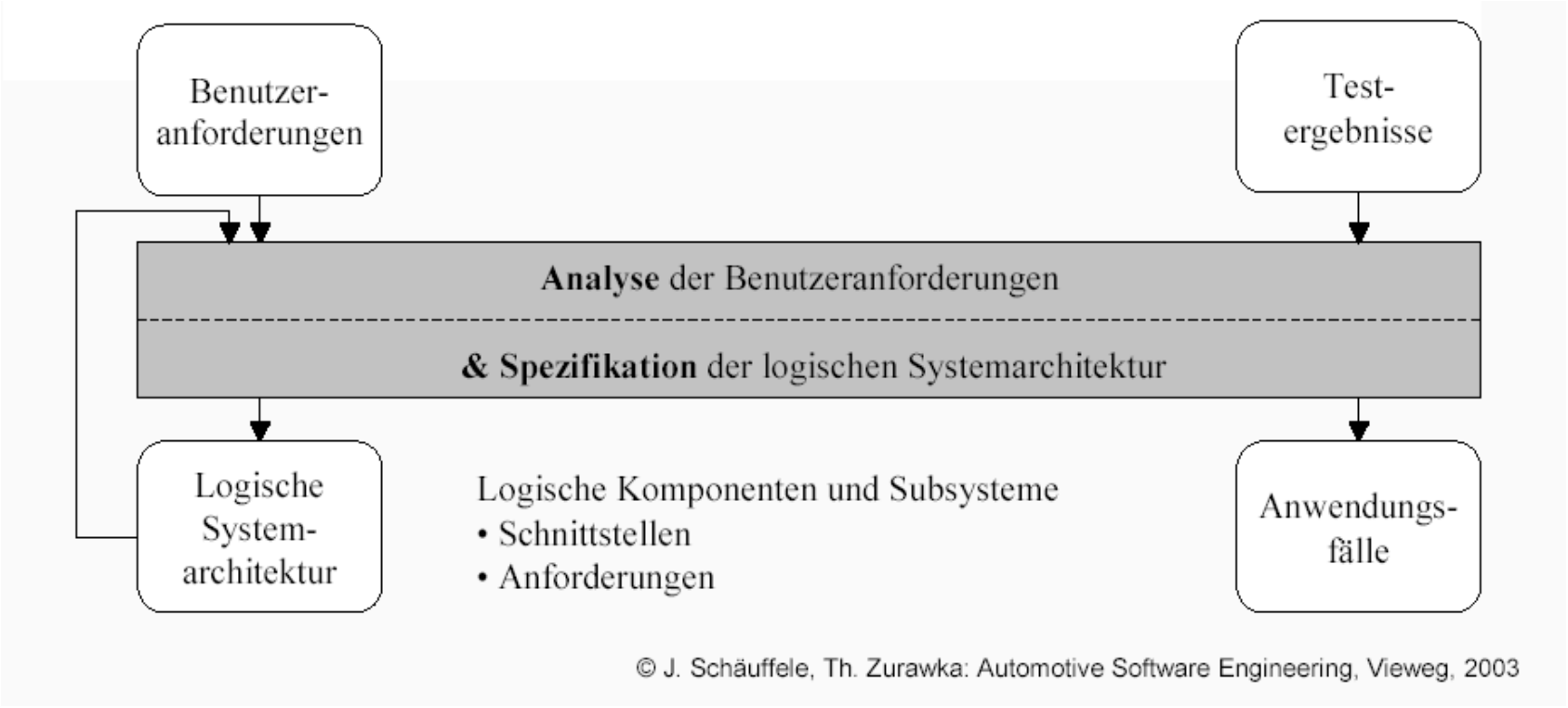
4. Kernprozess zur Entwicklung von elektronischen Systemen und Software

1. Grundbegriffe
2. Entwicklungsobjekt: Kombiinstrument
3. **Analyse und Spezifikation der Benutzeranforderungen**
4. Analyse und Spezifikation der technischen Anforderungen
5. Analyse und Spezifikation der Software-Anforderungen
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-Komponenten
11. Integration der System-Komponenten
12. Integrationstest des Systems
13. Kalibrierung
14. Akzeptanz- und Systemtest

Analyse und Spezifikation der Benutzeranforderungen



Analyse und Spezifikation der Benutzeranforderungen



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

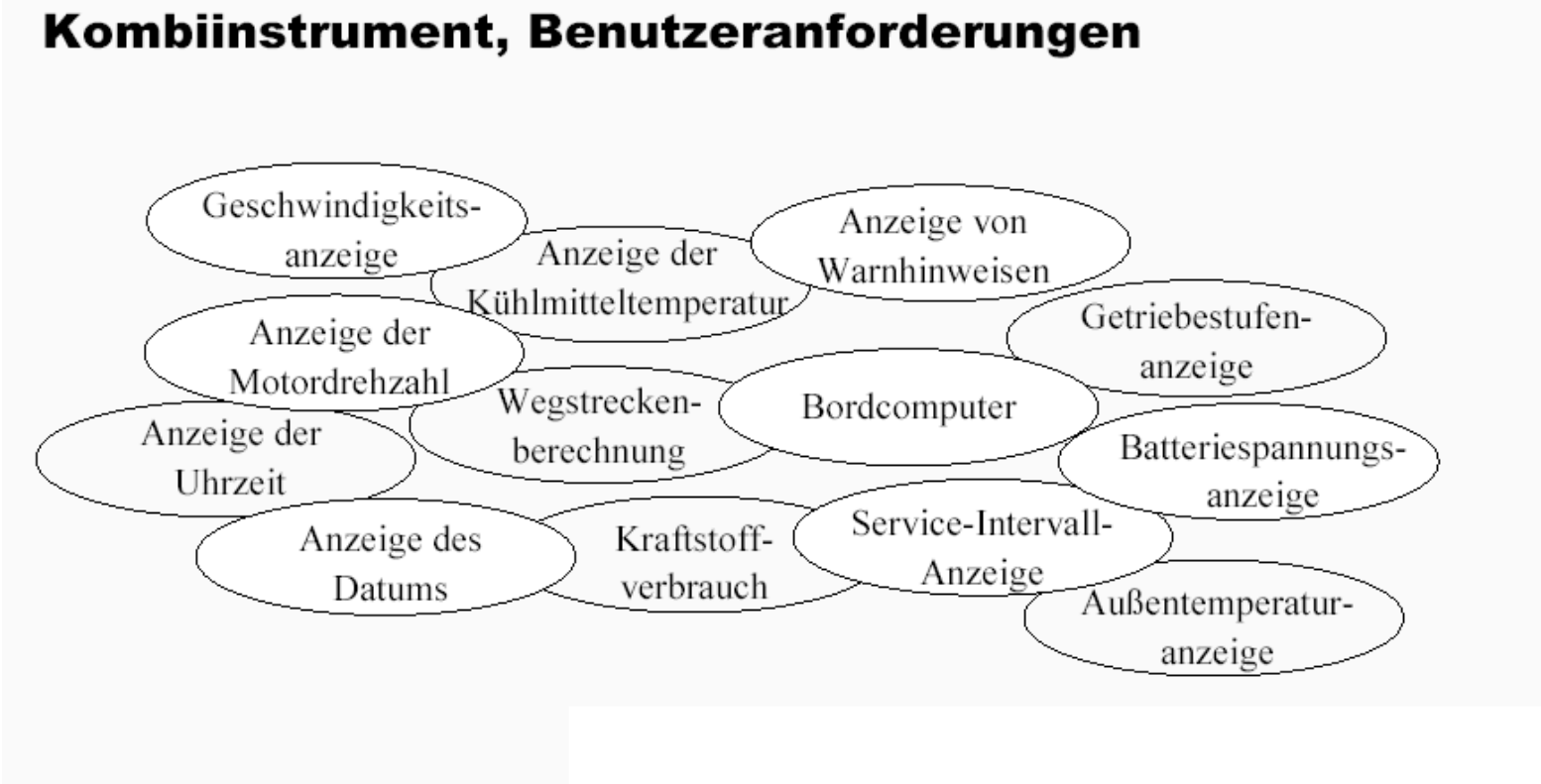
Analyse der Benutzeranforderungen

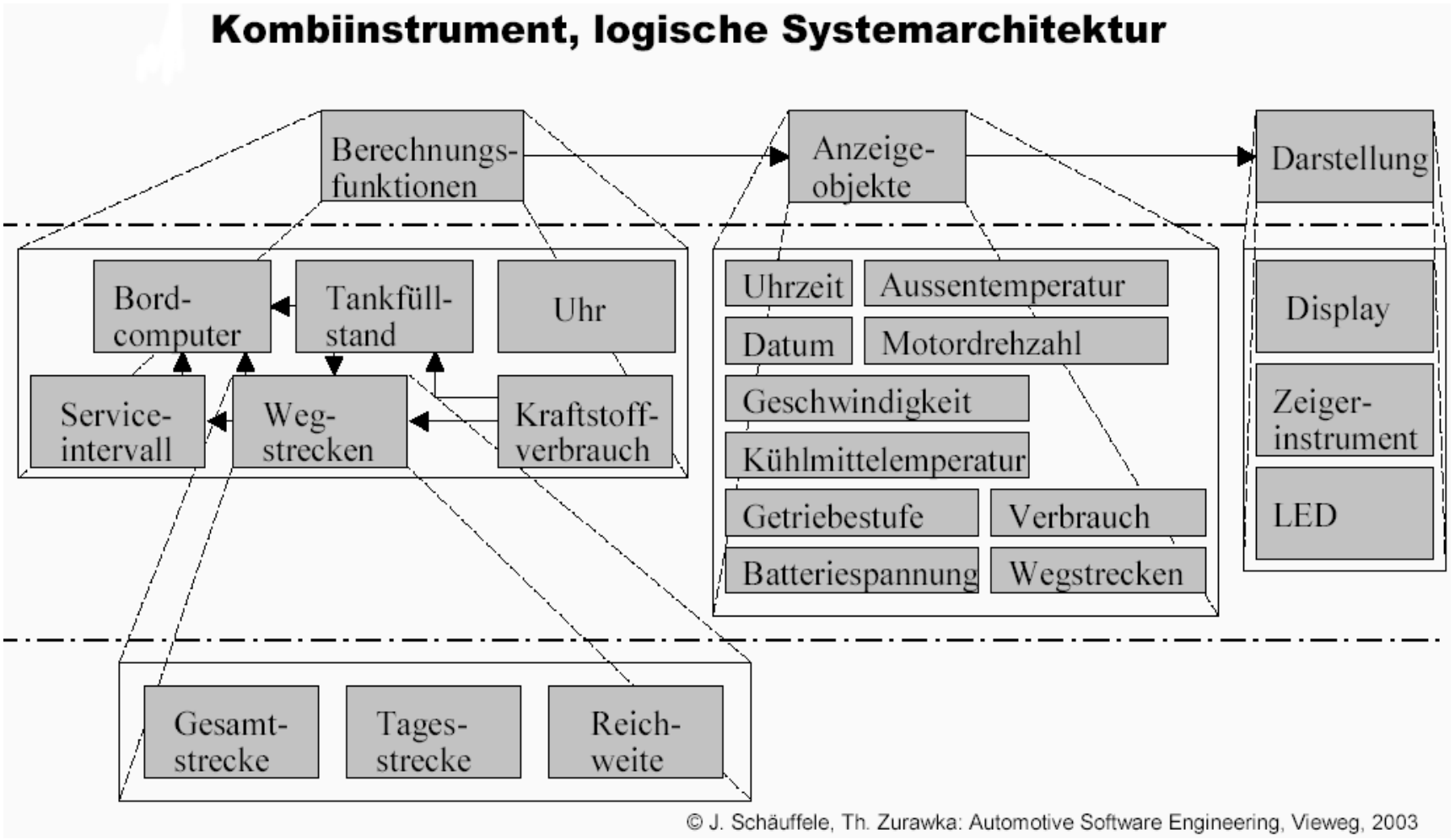
- Struckturierungsprozess für die Anforderungen und Randbedingungen aus Sicht der Benutzer in der frühen Phase der Systementwicklung
- logische Komponenten und Subsysteme definieren
- Funktionen, Anforderungen und Schnittstellen festlegen
- Anwendungsfälle für die Funktionen als Basis für den Systemtest festlegen

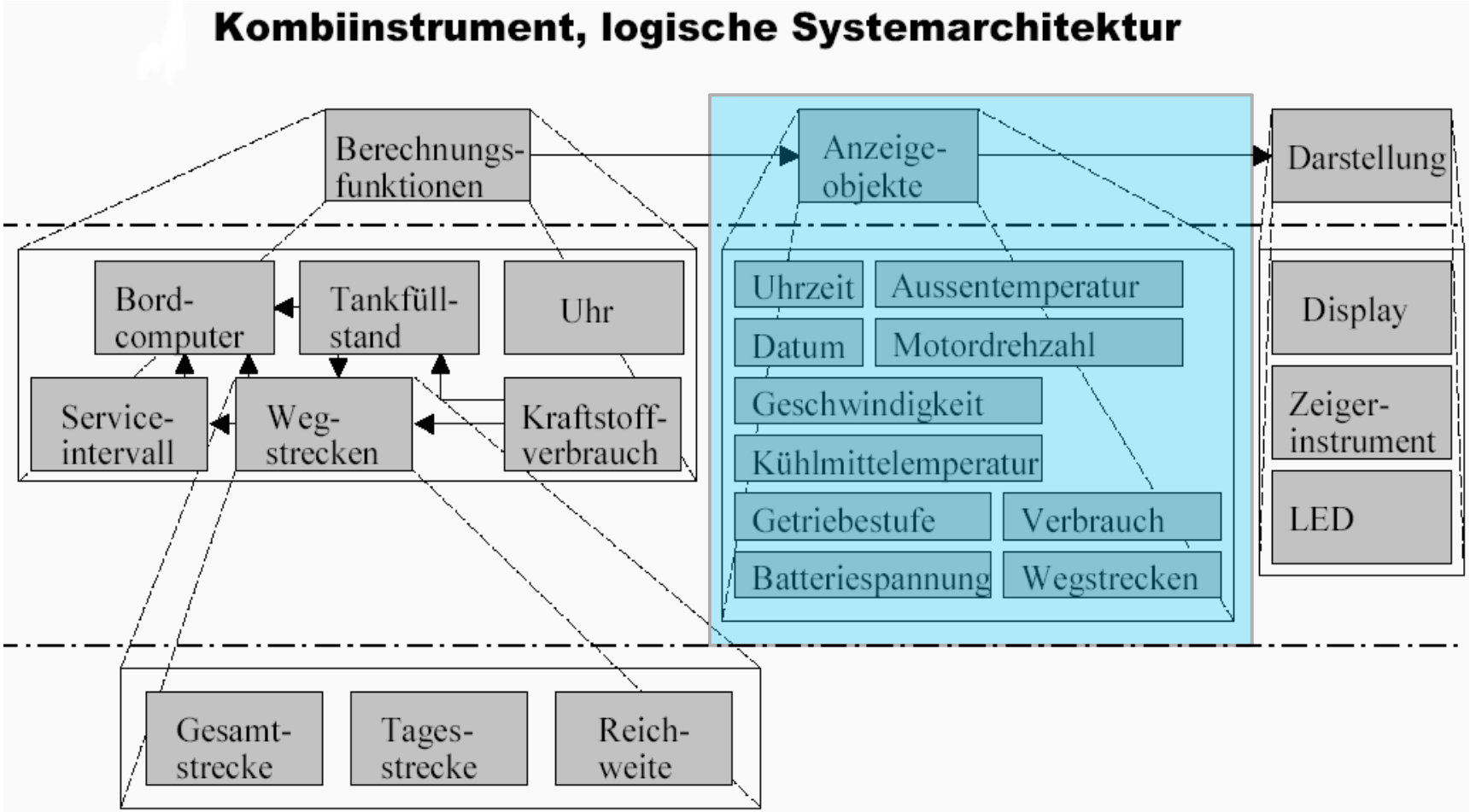
Logische Systemarchitektur

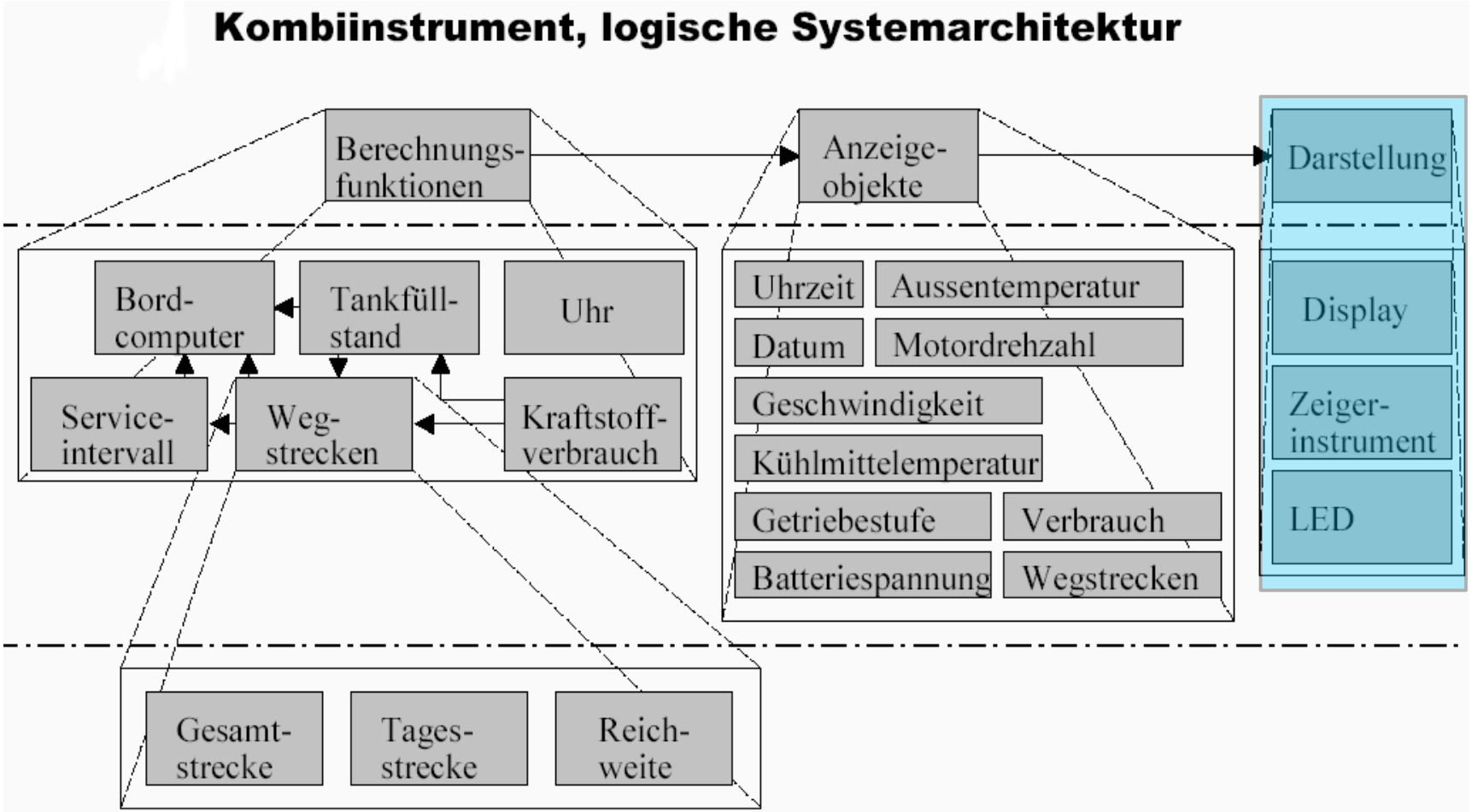
- abstrakte Lösung
- Bindeglied zwischen Benutzeranforderungen und technischer Systemarchitektur
- modellbasierte Darstellung (Blockdiagramme, Zustandsautomaten)
- schrittweise Zerlegung der Funktionen

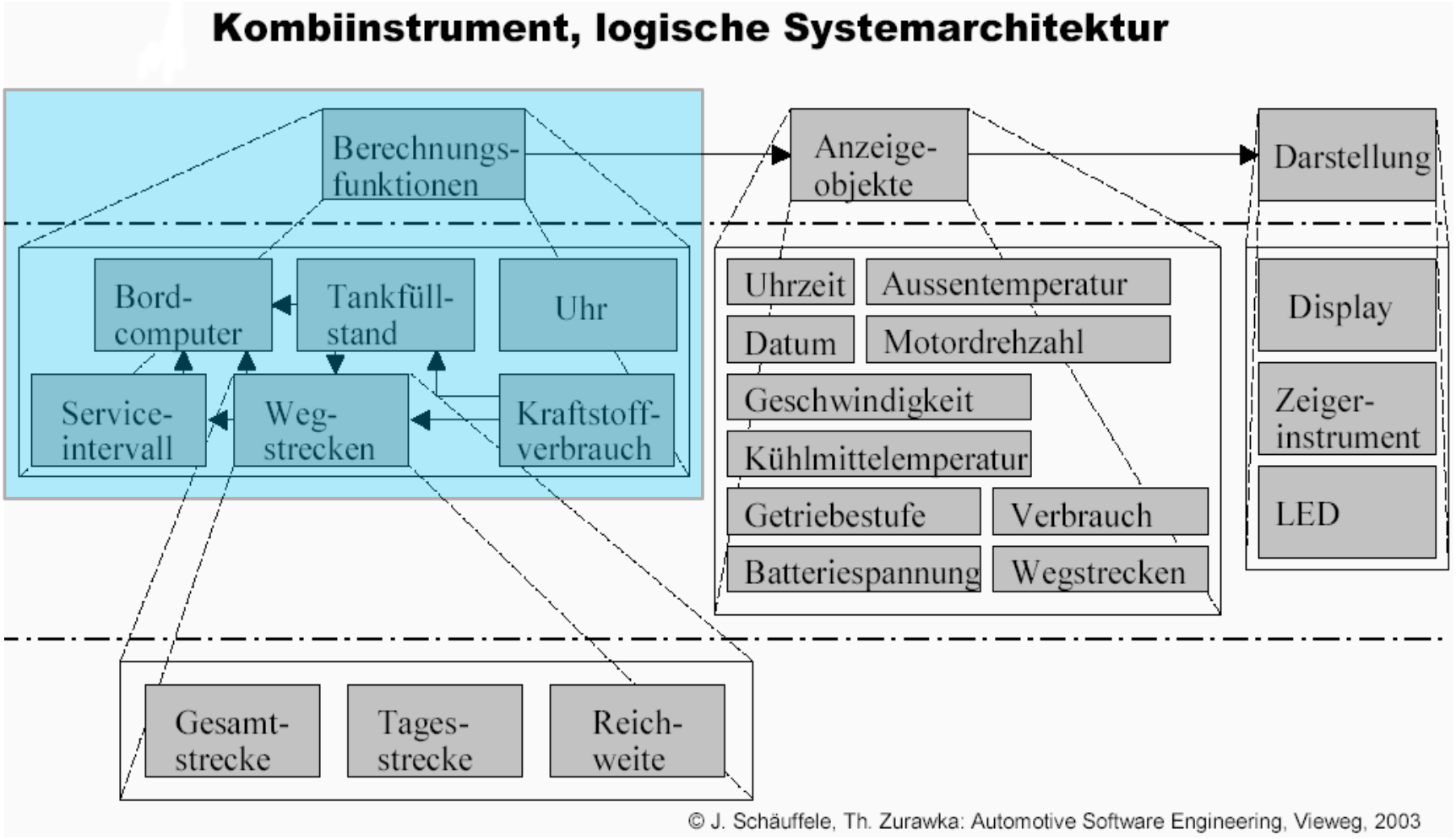
Analyse und Spezifikation der Benutzeranforderungen

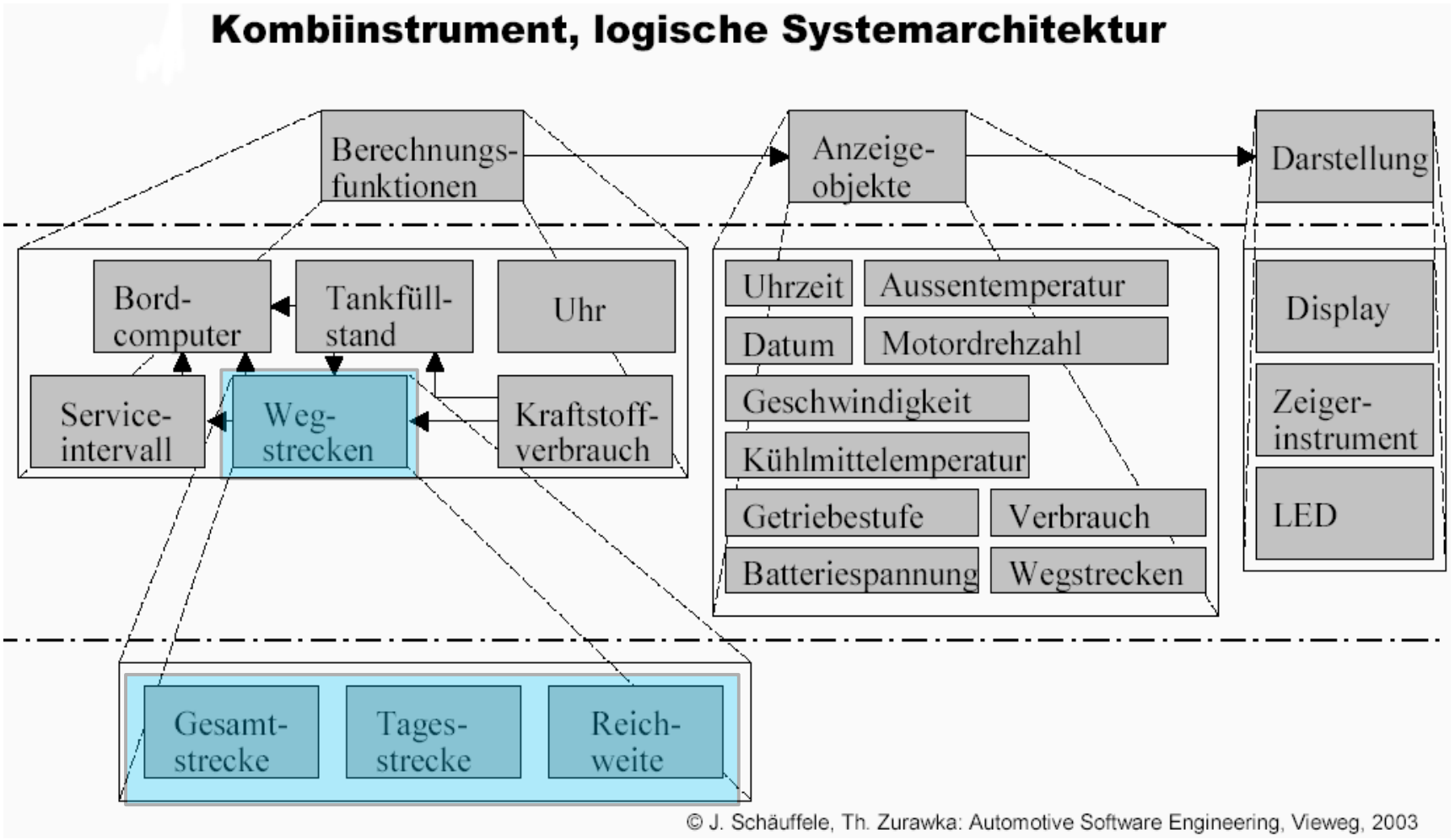



Kombiinstrument, logische Systemarchitektur
Berechnungs-funktionen
Anzeige-objekte
Darstellung
Bord-computer
Tankfüll-stand
Uhr
Service-intervall
Weg-strecken
Kraftstoff-verbrauch
Uhrzeit
Aussentemperatur
Datum
Motordrehzahl
Geschwindigkeit
Kühlmittelemperatur
Getriebestufe
Verbrauch
Batteriespannung
Wegstrecken
Display
Zeiger-instrument
LED
Gesamt-strecke
Tages-strecke
Reich-weite
© J. Schäuffele, Th. Zurawka: Automotive Software Engineering, Vieweg, 2003


Kombiinstrument, logische Systemarchitektur
Berechnungs-funktionen
Anzeige-objekte
Darstellung
Bord-computer
Tankfüll-stand
Uhr
Service-intervall
Weg-strecken
Kraftstoff-verbrauch
Uhrzeit
Aussentemperatur
Datum
Motordrehzahl
Geschwindigkeit
Kühlmittelemperatur
Getriebestufe
Verbrauch
Batteriespannung
Wegstrecken
Display
Zeiger-instrument
LED
Gesamt-strecke
Tages-strecke
Reich-weite
© J. Schäuffele, Th. Zurawka: Automotive Software Engineering, Vieweg, 2003


Kombiinstrument, logische Systemarchitektur
Berechnungs-funktionen
Anzeige-objekte
Darstellung
Bord-computer
Tankfüll-stand
Uhr
Service-intervall
Weg-strecken
Kraftstoff-verbrauch
Uhrzeit
Aussentemperatur
Datum
Motordrehzahl
Geschwindigkeit
Kühlmittelemperatur
Getriebestufe
Verbrauch
Batteriespannung
Wegstrecken
Display
Zeiger-instrument
LED
Gesamt-strecke
Tages-strecke
Reich-weite
© J. Schäuffele, Th. Zurawka: Automotive Software Engineering, Vieweg, 2003

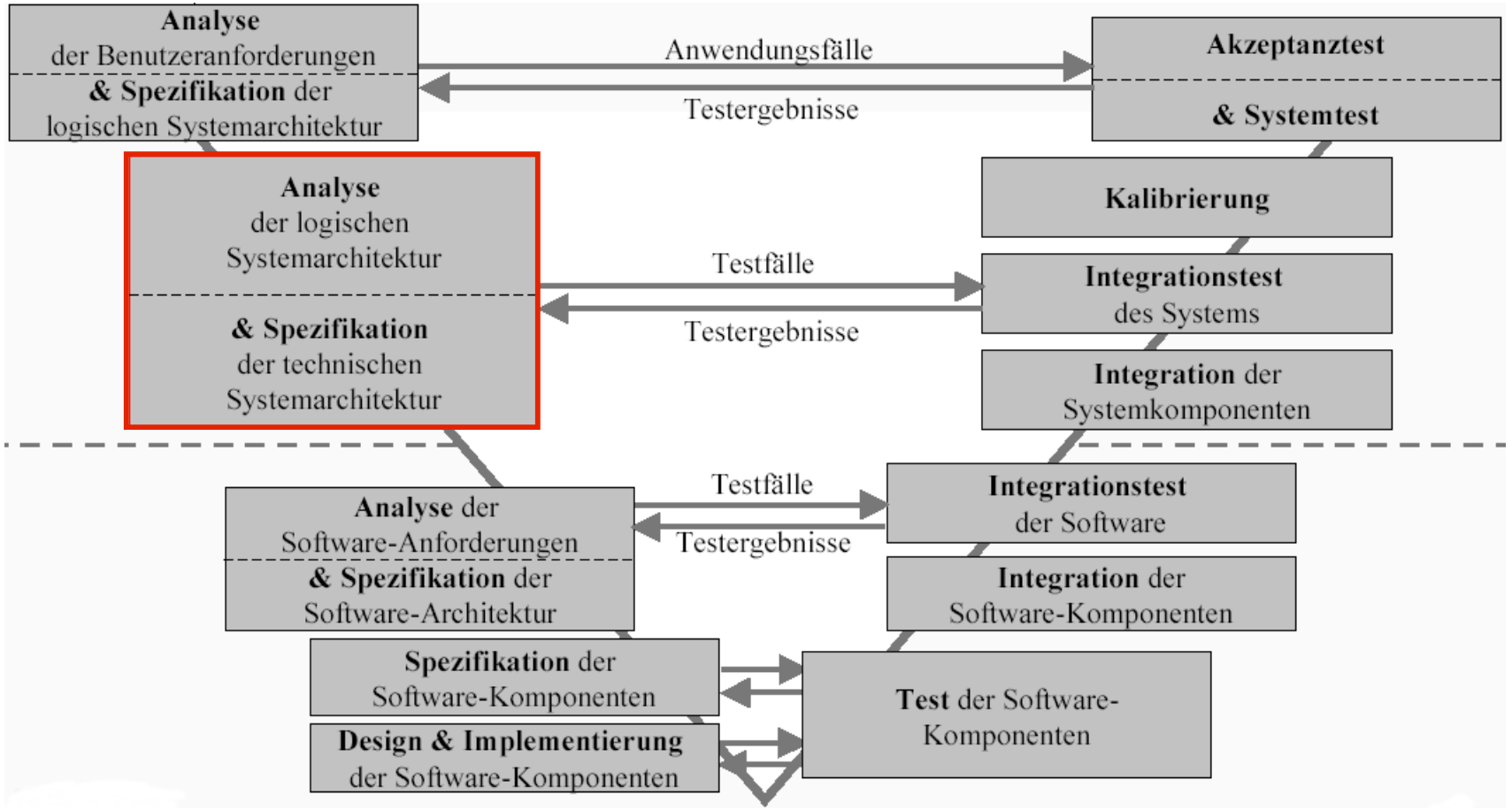

Kombiinstrument, logische Systemarchitektur
Berechnungs-funktionen
Anzeige-objekte
Darstellung
Bord-computer
Tankfüll-stand
Uhr
Service-intervall
Weg-strecken
Kraftstoff-verbrauch
Uhrzeit
Aussentemperatur
Datum
Motordrehzahl
Geschwindigkeit
Kühlmittelemperatur
Getriebestufe
Verbrauch
Batteriespannung
Wegstrecken
Display
Zeiger-instrument
LED
Gesamt-strecke
Tages-strecke
Reich-weite
© J. Schäuffele, Th. Zurawka: Automotive Software Engineering, Vieweg, 2003


Kombiinstrument, logische Systemarchitektur
Berechnungs-funktionen
Anzeige-objekte
Darstellung
Bord-computer
Tankfüll-stand
Uhr
Service-intervall
Weg-strecken
Kraftstoff-verbrauch
Uhrzeit
Aussentemperatur
Datum
Motordrehzahl
Geschwindigkeit
Kühlmittelemperatur
Getriebestufe
Verbrauch
Batteriespannung
Wegstrecken
Display
Zeiger-instrument
LED
Gesamt-strecke
Tages-strecke
Reich-weite
© J. Schäuffele, Th. Zurawka: Automotive Software Engineering, Vieweg, 2003

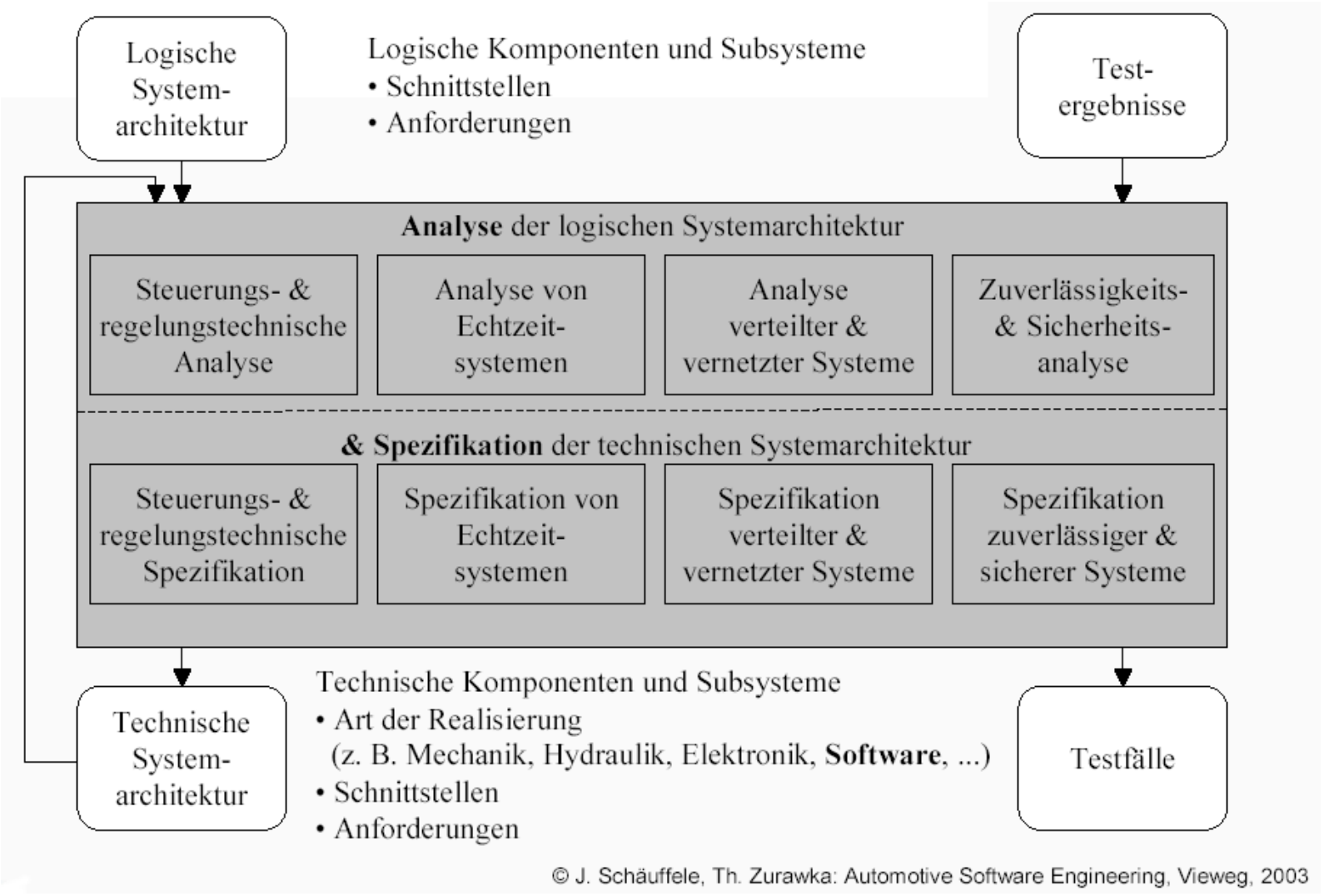
4. Kernprozess zur Entwicklung von elektronischen Systemen und Software

1. Grundbegriffe
2. Entwicklungsobjekt: Kombiinstrument
3. Analyse und Spezifikation der Benutzeranforderungen
4. **Analyse und Spezifikation der technischen Anforderungen**
5. Analyse und Spezifikation der Software-Anforderungen
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-Komponenten
11. Integration der System-Komponenten
12. Integrationstest des Systems
13. Kalibrierung
14. Akzeptanz- und Systemtest

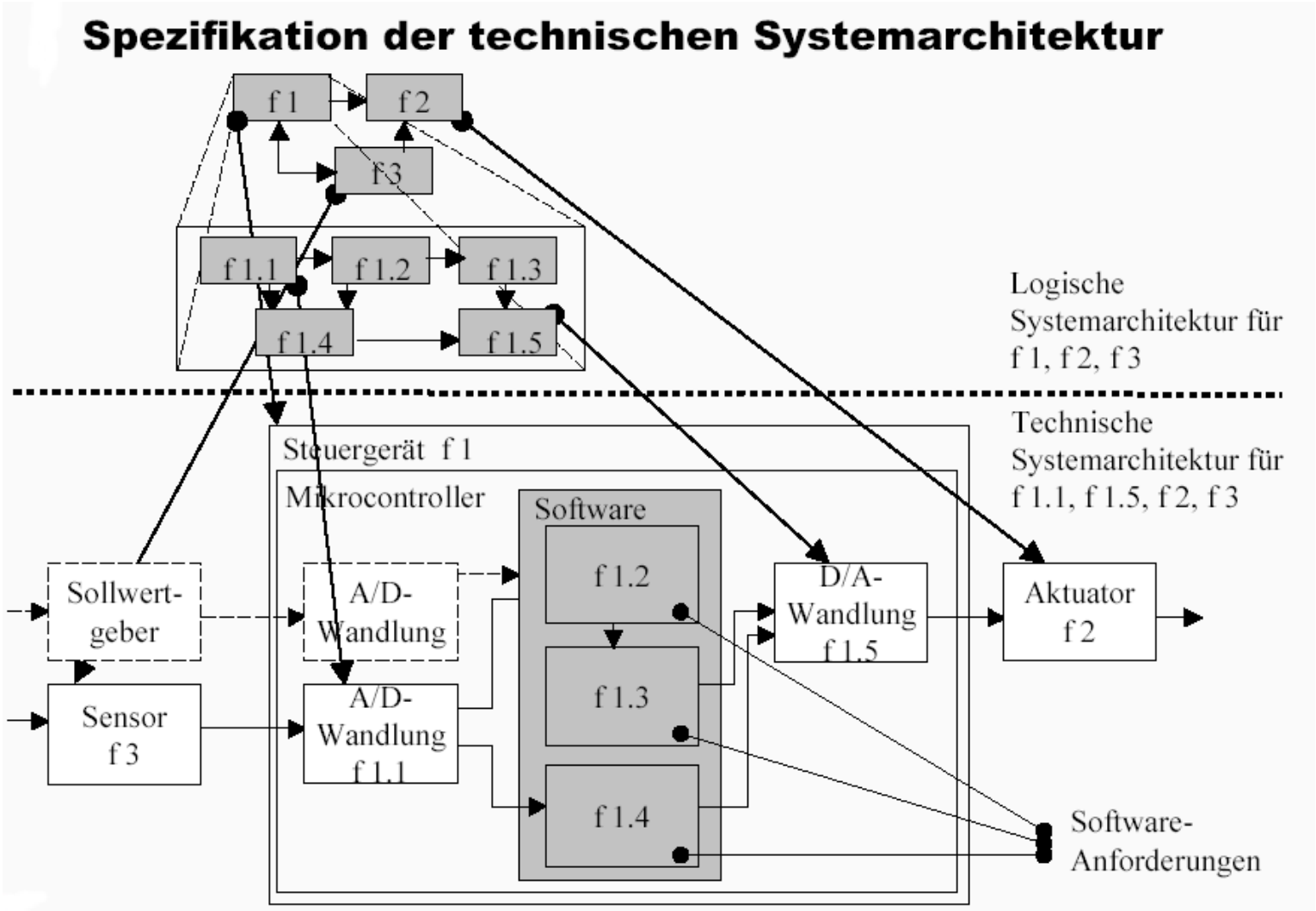
Analyse und Spezifikation der technischen Anforderungen



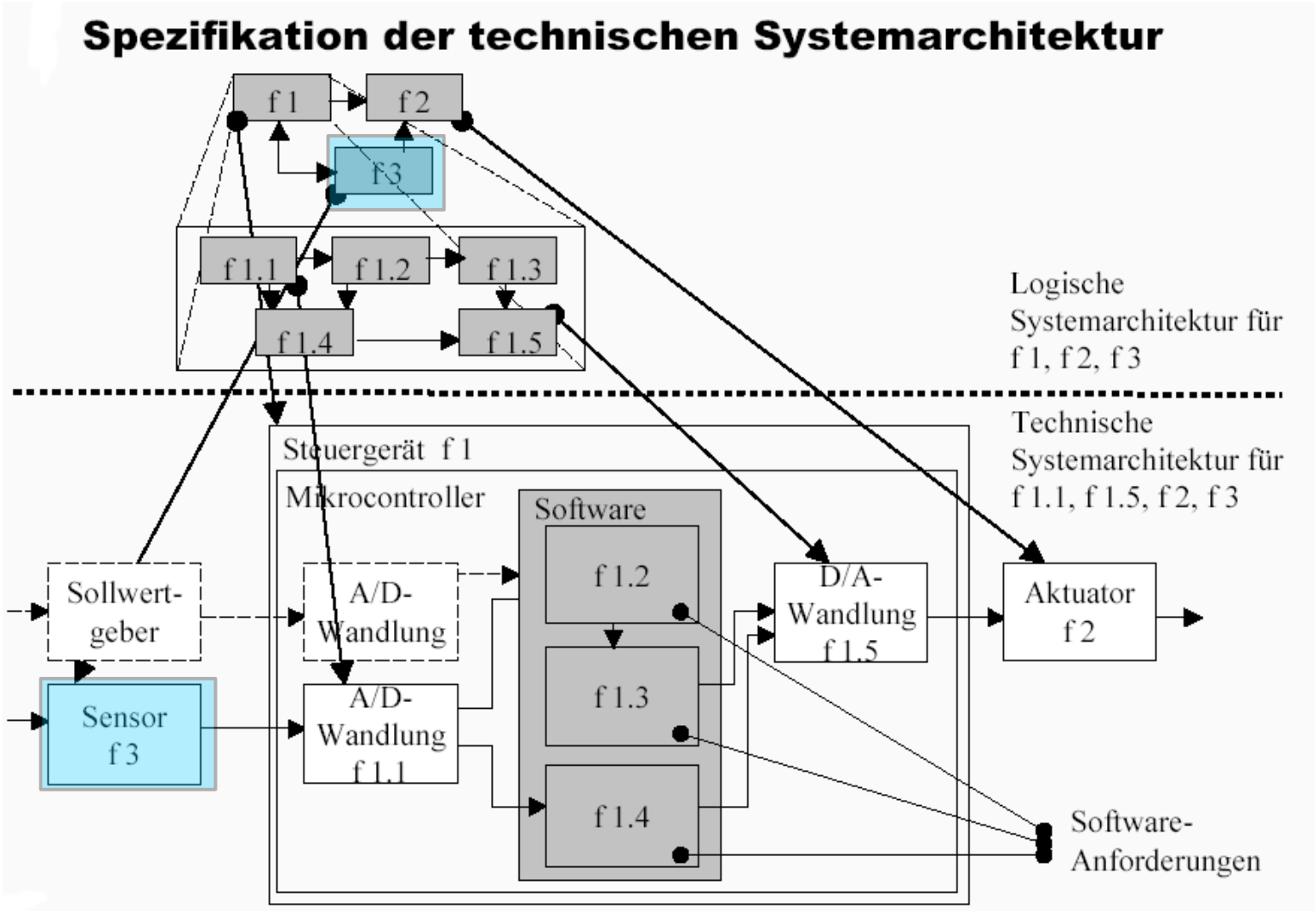
Analyse und Spezifikation der technischen Anforderungen



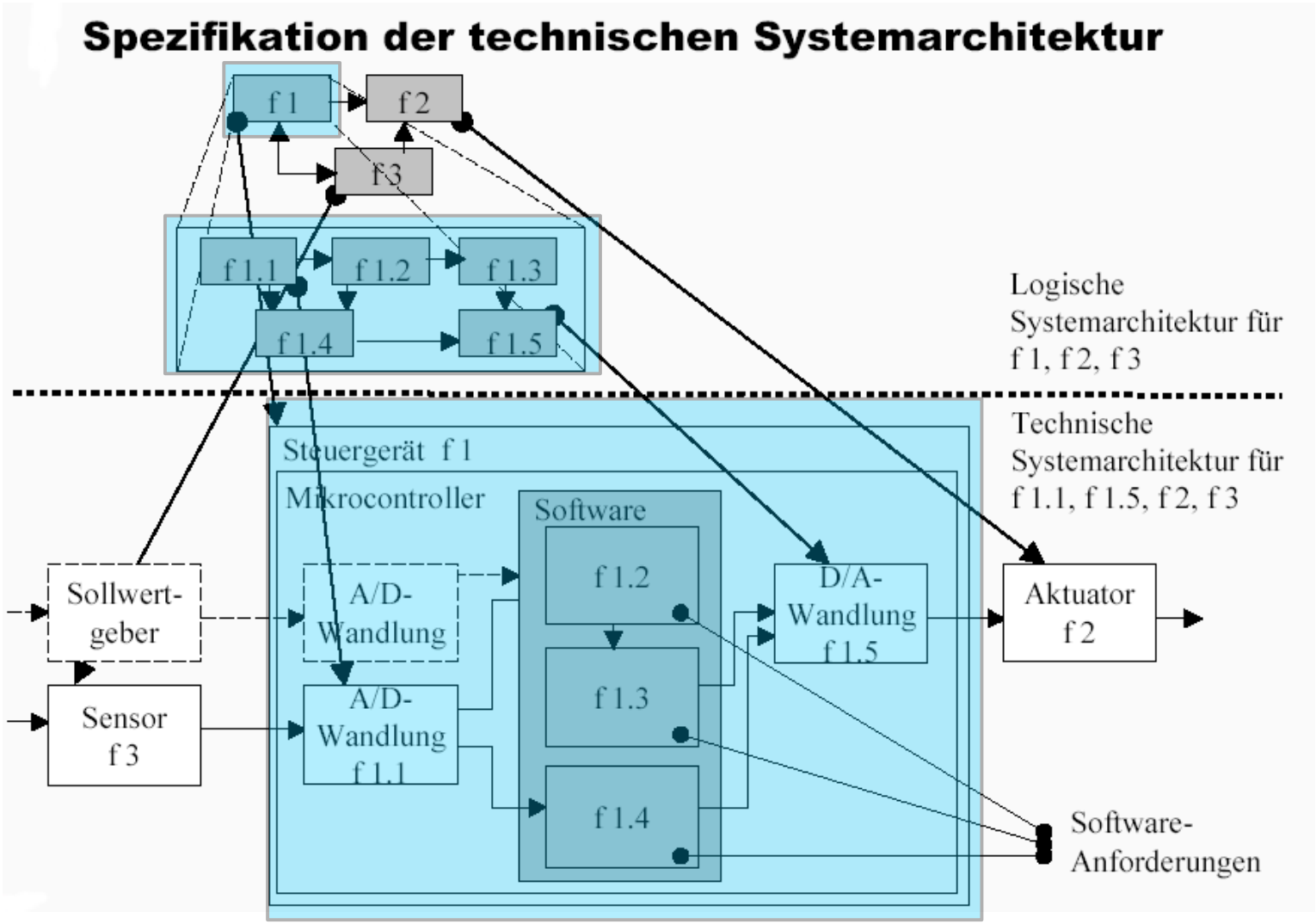
Analyse und Spezifikation der technischen Anforderungen



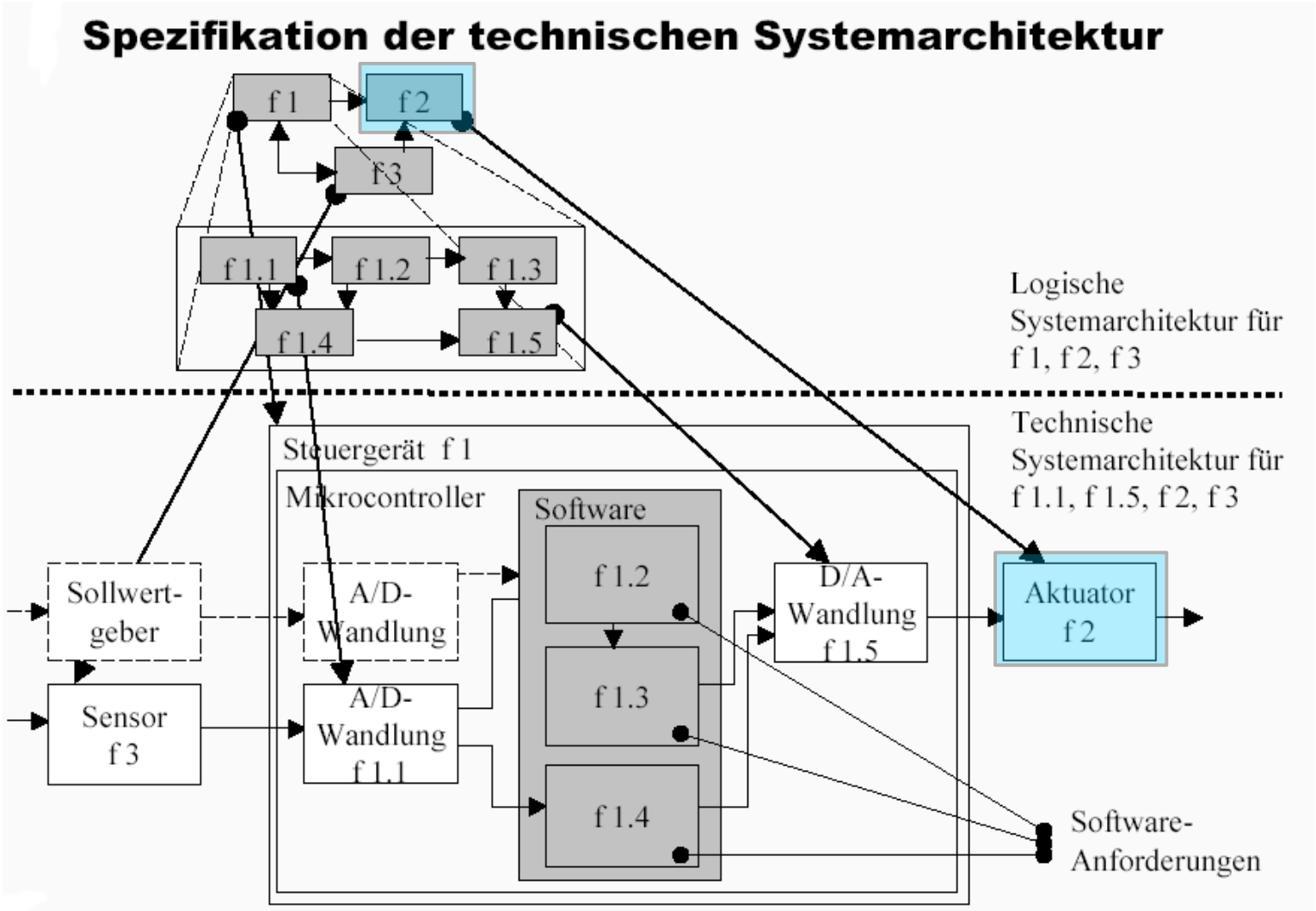
Analyse und Spezifikation der technischen Anforderungen



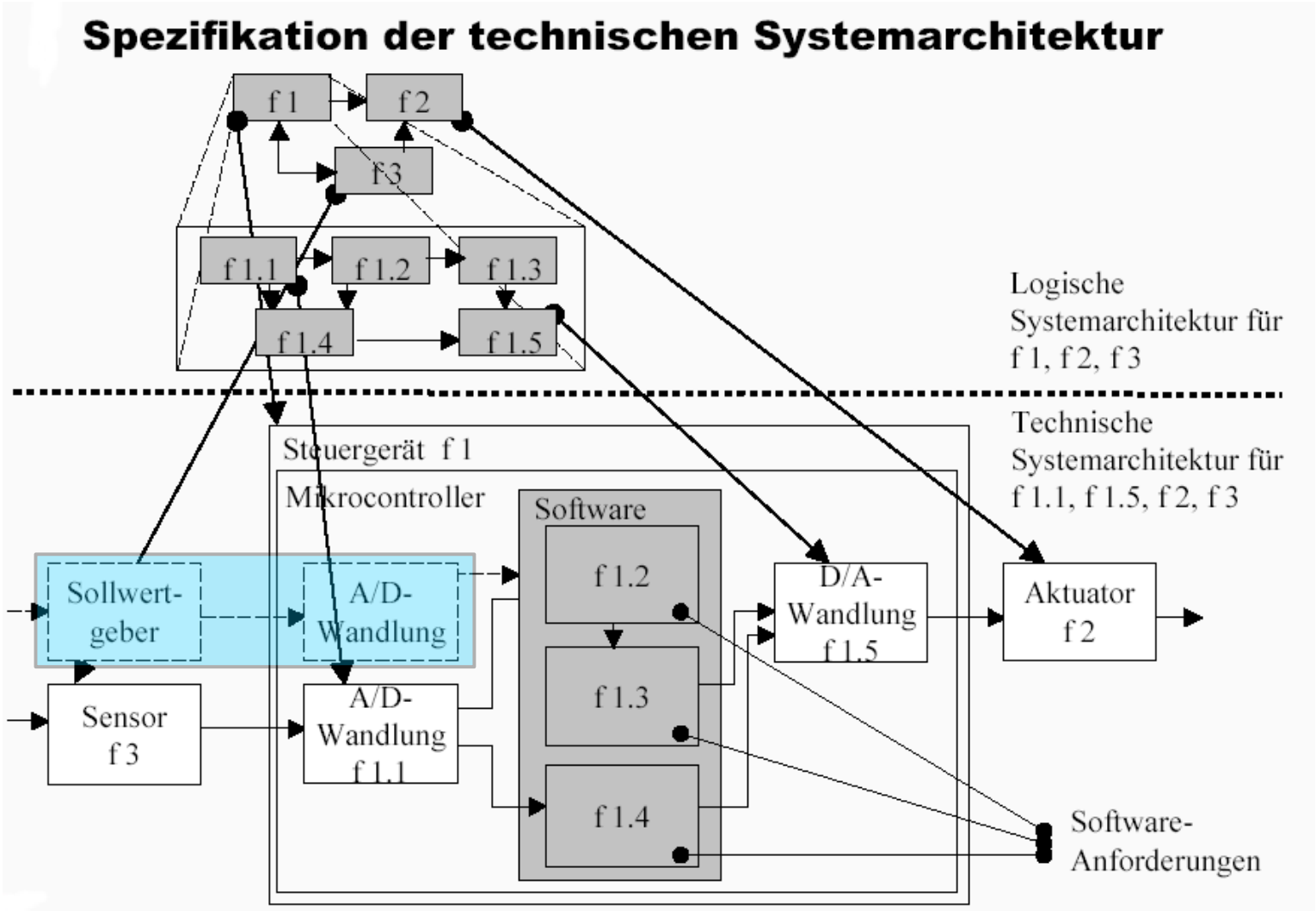
Analyse und Spezifikation der technischen Anforderungen



Analyse und Spezifikation der technischen Anforderungen



Analyse und Spezifikation der technischen Anforderungen



Randbedingungen für die technische Systemarchitektur

- Standards und Entwurfmuster
- Abhängigkeiten zwischen verschiedenen Systemen und Komponenten
- Ergebnisse von Machbarkeitsanalysen
- Produktions- und Serviceanforderungen
- Änderbarkeits- und Testbarkeitsanforderungen
- Aufwands-, Kosten- und Risikoabschätzungen

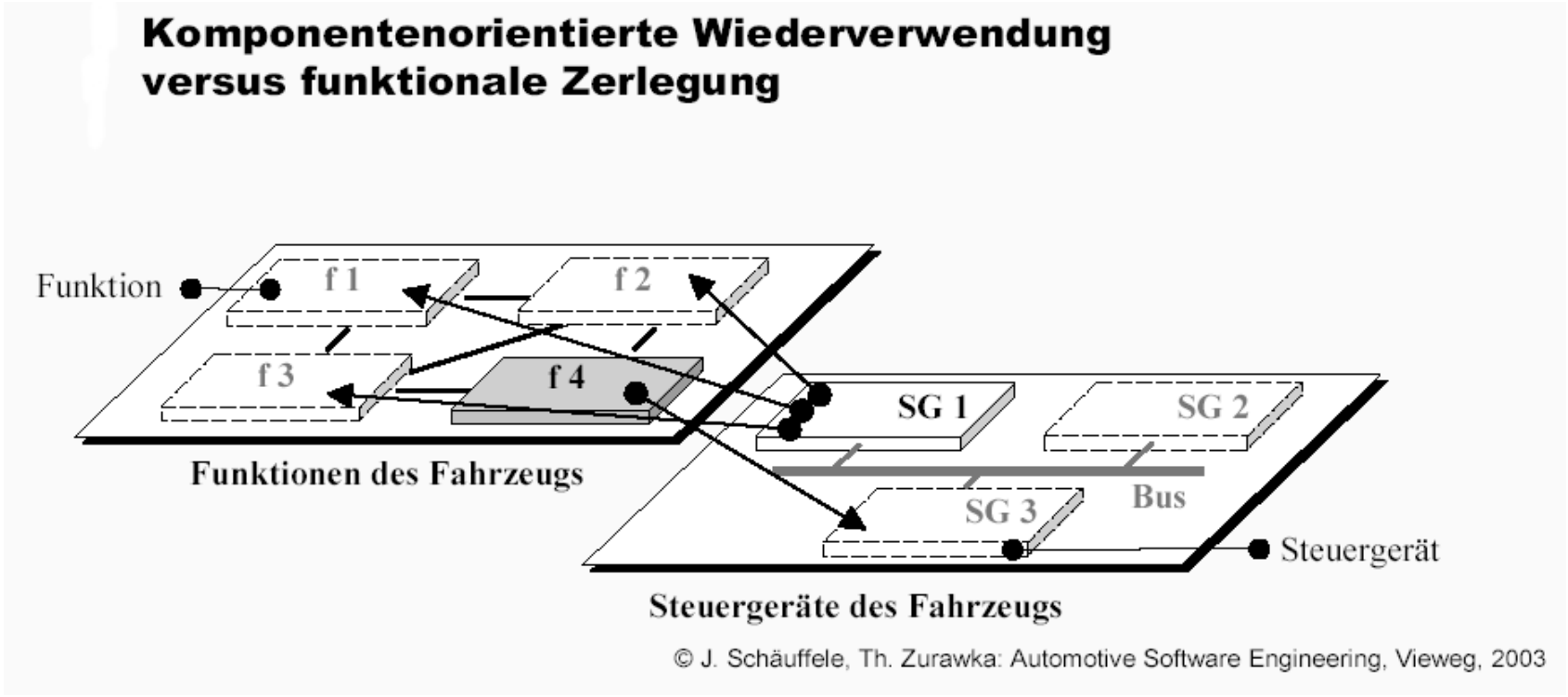
Beispiele

- Wiederverwendung von technischen Komponenten in verschiedenen Fahrzeugbaureihen
- Verschiedene Fahrzeugvarianten innerhalb einer Fahrzeugbaureihe
- Sonderausstattung versus Serienausstattung
- Länderspezifische Ausstattungsvarianten
- Komponentenorientierte Wiederverwendung

Randbedingungen und Zielkonflikte

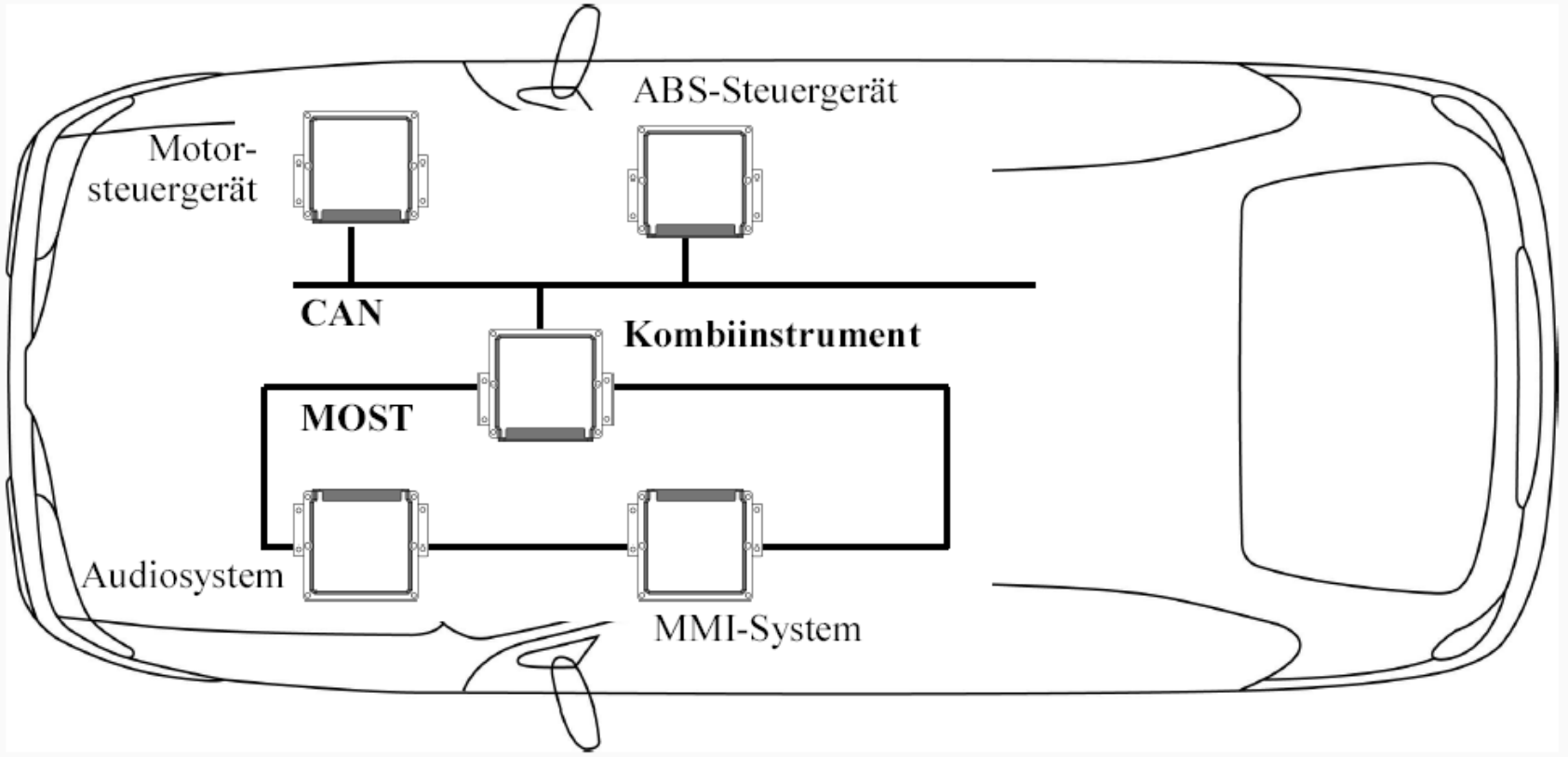
- Wiederverwendung von technischen Komponenten in verschiedenen Baureihen
  - Motoren
  - Getriebe
  - Einheitliche Motor- und Getriebesteuergeräte mit unterschiedlichem Programm und Datenstand
- Verschiedene Varianten innerhalb einer Baureihe
  - Schaltgetriebe
  - Automatikgetriebe
  - Trennung von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

Analyse und Spezifikation der technischen Anforderungen



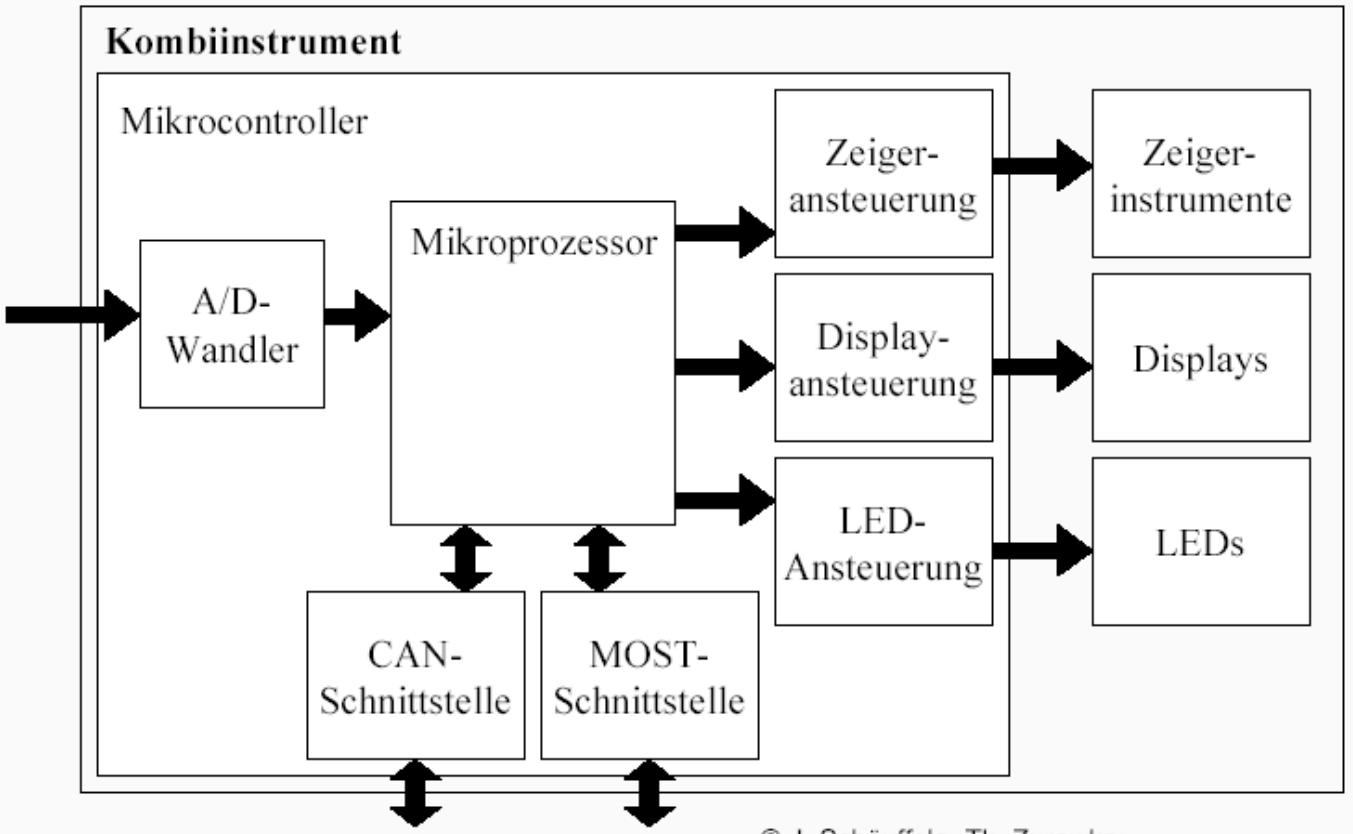
- Vorgabe:
  - Wiederverwendung des Steuergerätes SG1 mit den Funktionen f1, f2, f3
- Freiheitsgrad:
  - Zuordnung der Funktion f4 (z.B. auf SG 3)

Kombiinstrument als Komponente im Steuergerätenetzwerk



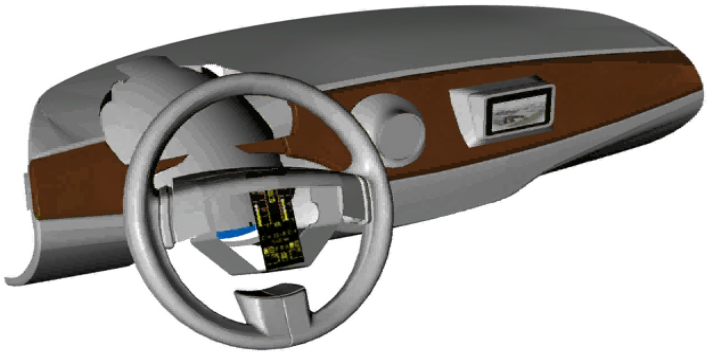
© J. Schäuffele, Th. Zurawka: Automotive Software Engineering, Vieweg, 2003

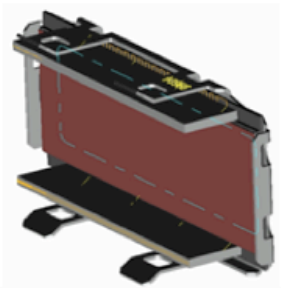
Hardware des Kombiinstruments

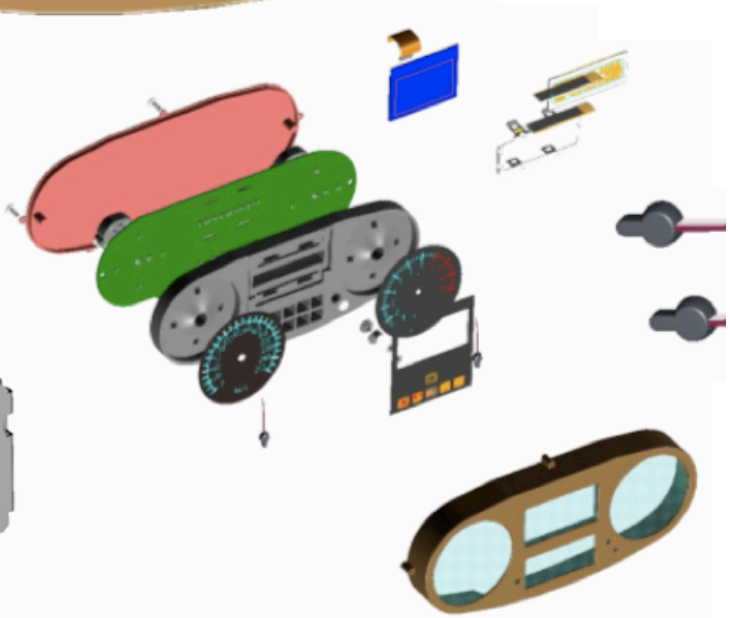


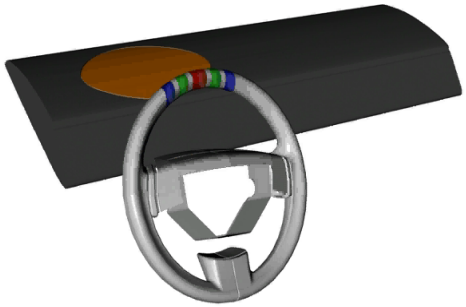
© J. Schäuffele, Th. Zurawka:
Automotive Software Engineering, Vieweg, 2003

Entwicklungsobjekt: Kombiinstrument

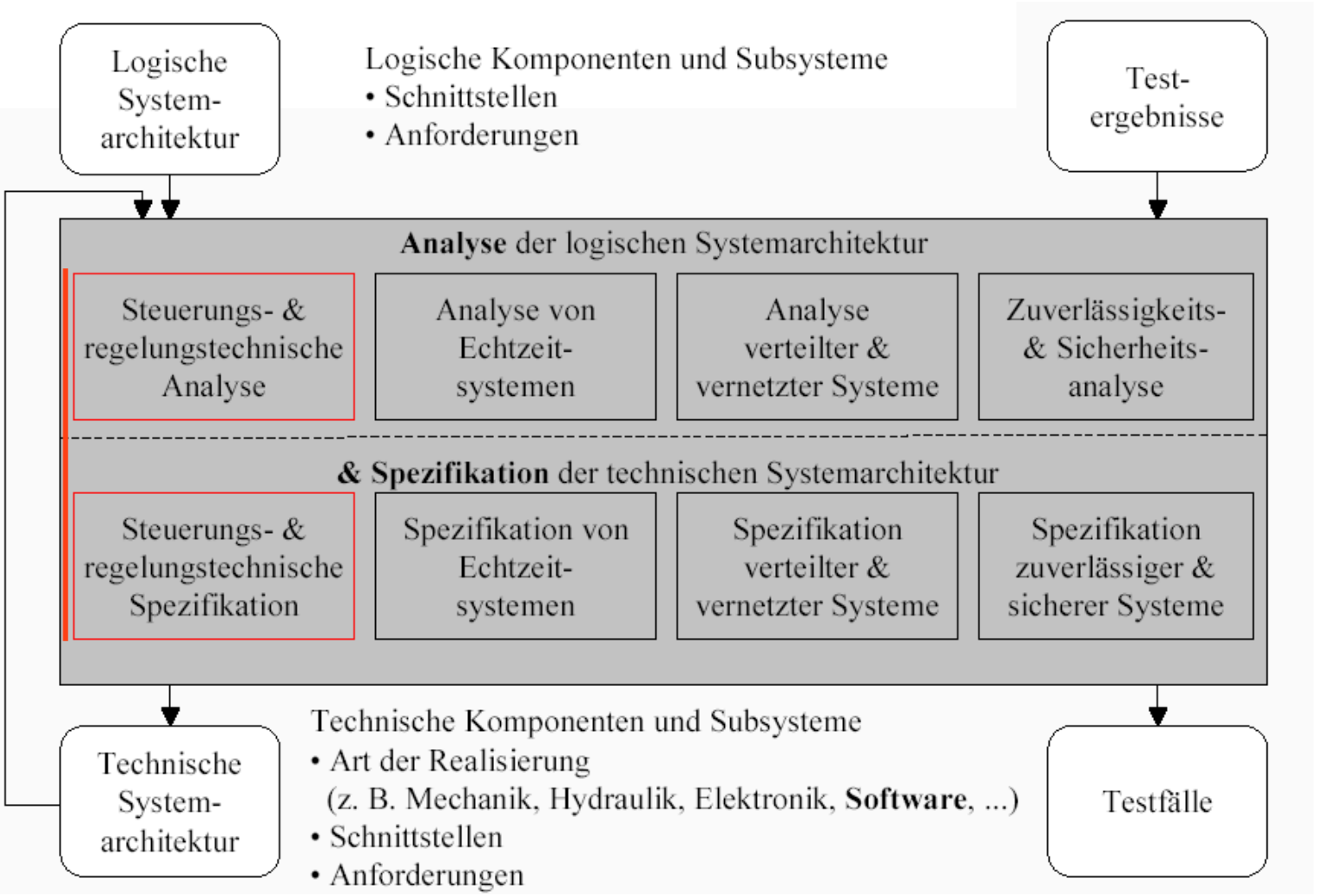




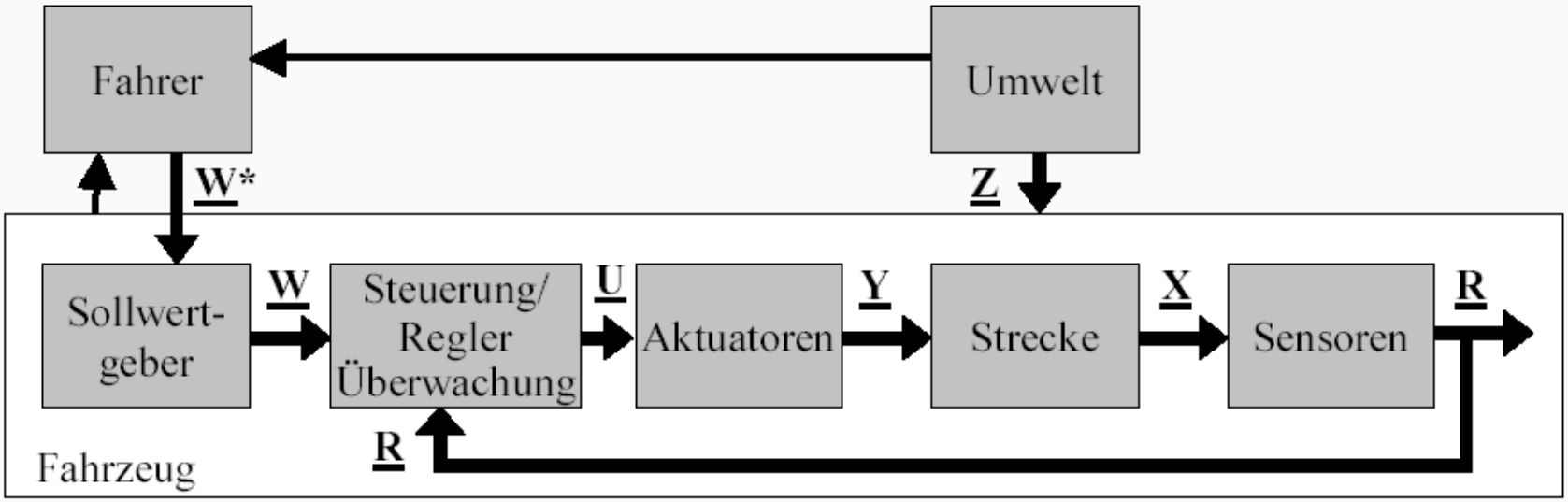




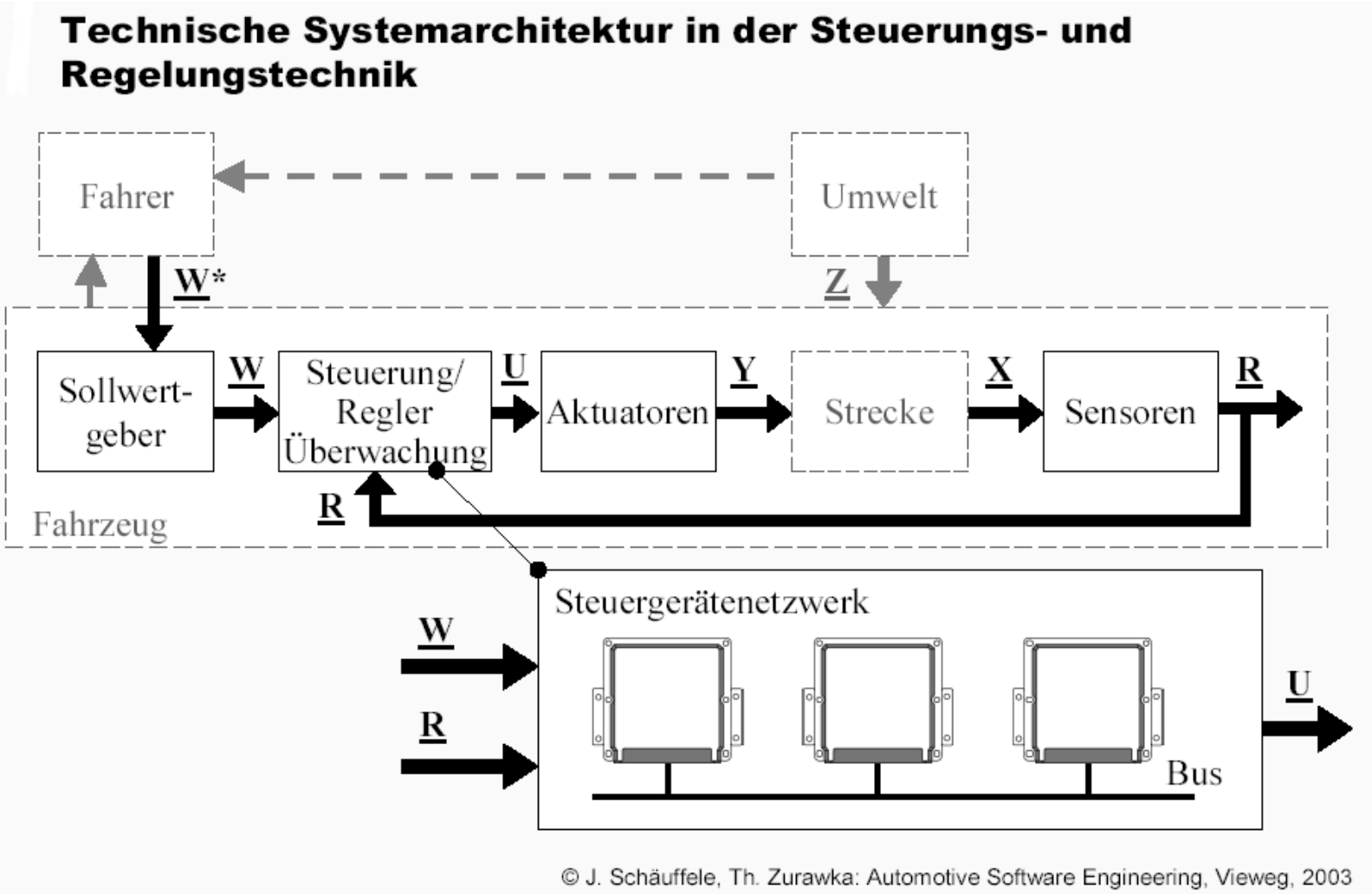
Analyse und Spezifikation der technischen Anforderungen



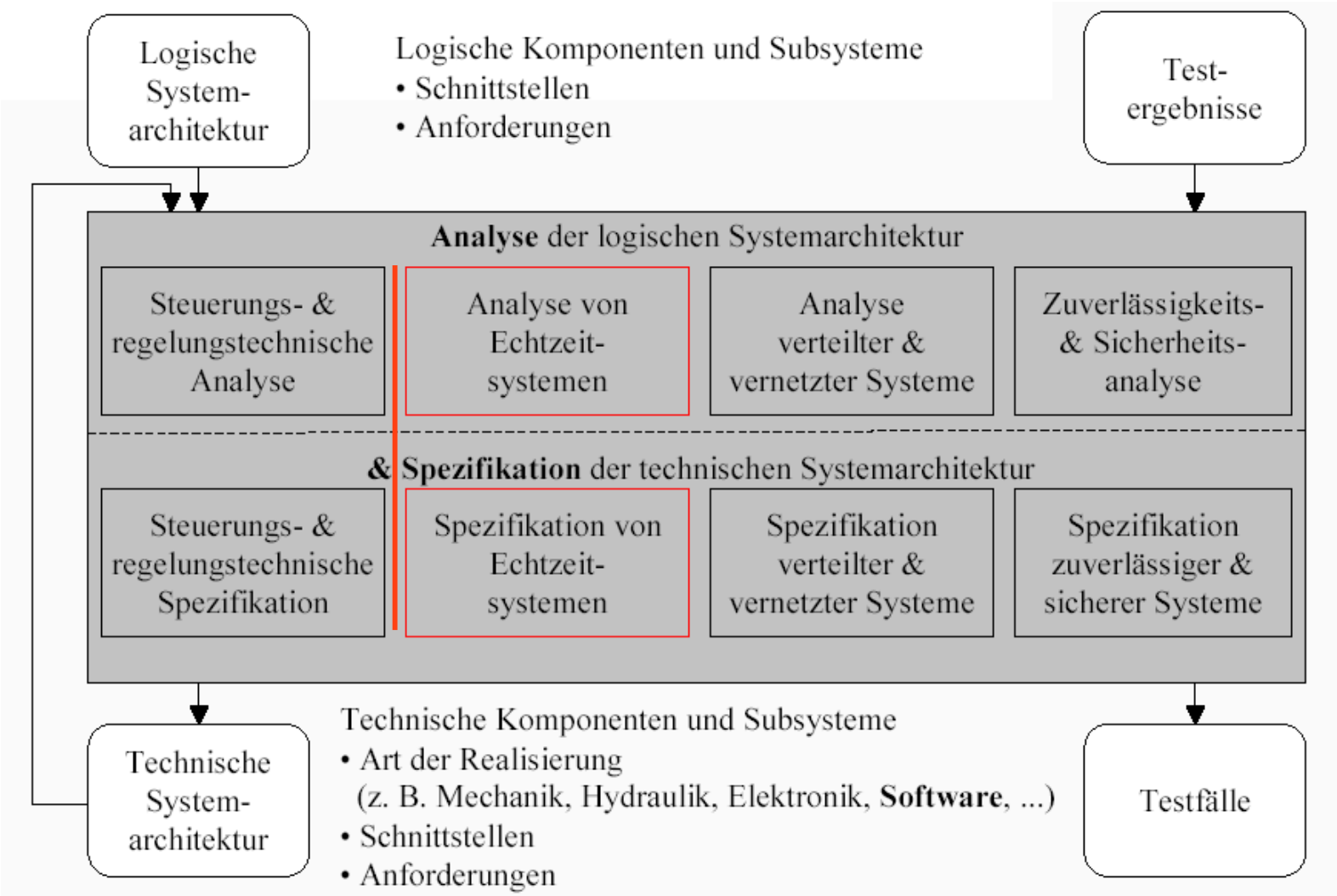
Logische Systemarchitektur in der Steuerungs- und Regelungstechnik



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


Technische Systemarchitektur in der Steuerungs- und Regelungstechnik
Fahrer
Umwelt
W*
Z
Sollwert-geber
W
Steuerung/ Regler Überwachung
U
Aktuatoren
Y
Strecke
X
Sensoren
R
R
Fahrzeug
Steuergerätenetzwerk
W
R
U
Bus
© J. Schäuffele, Th. Zurawka: Automotive Software Engineering, Vieweg, 2003

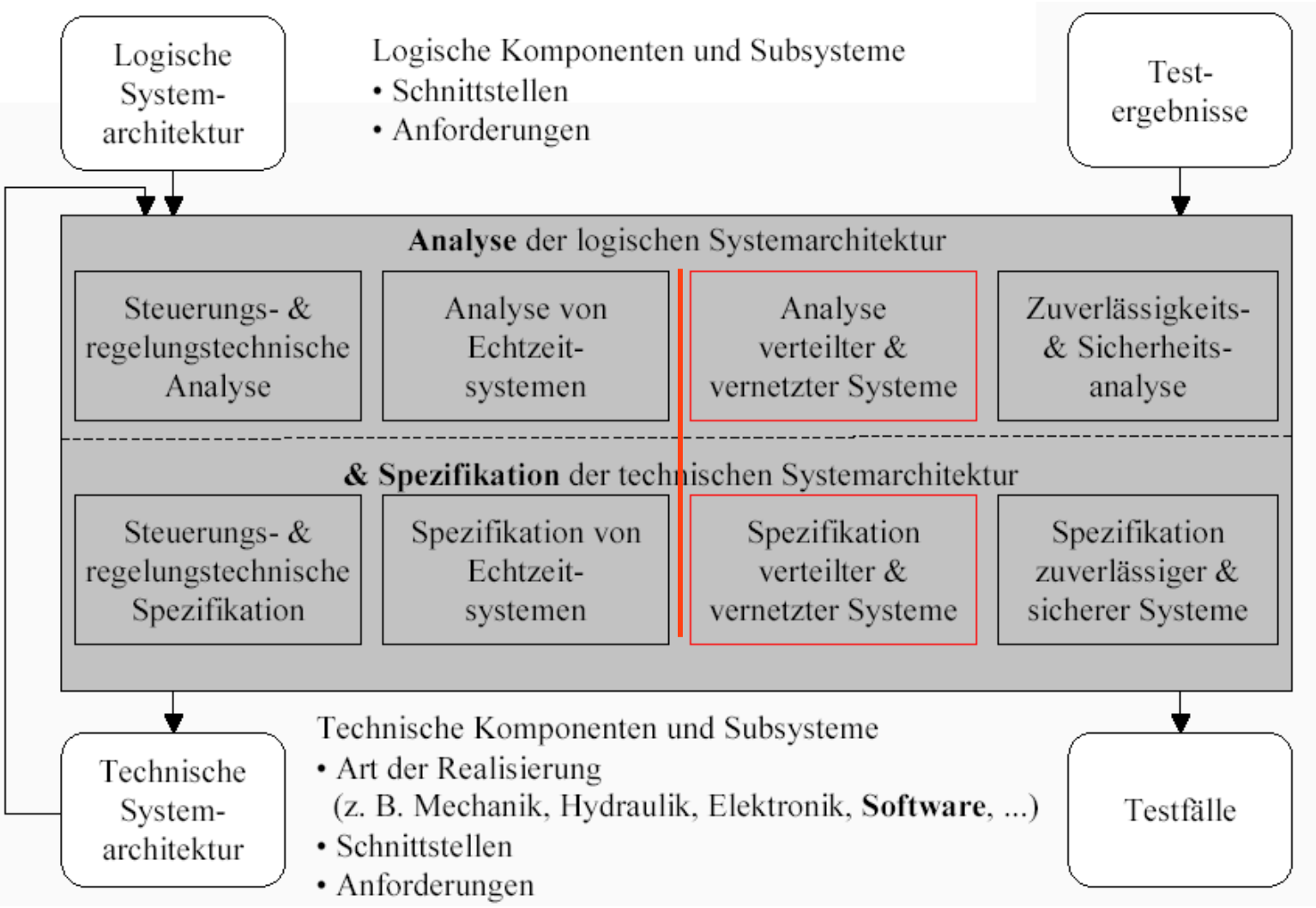
Analyse und Spezifikation der technischen Anforderungen

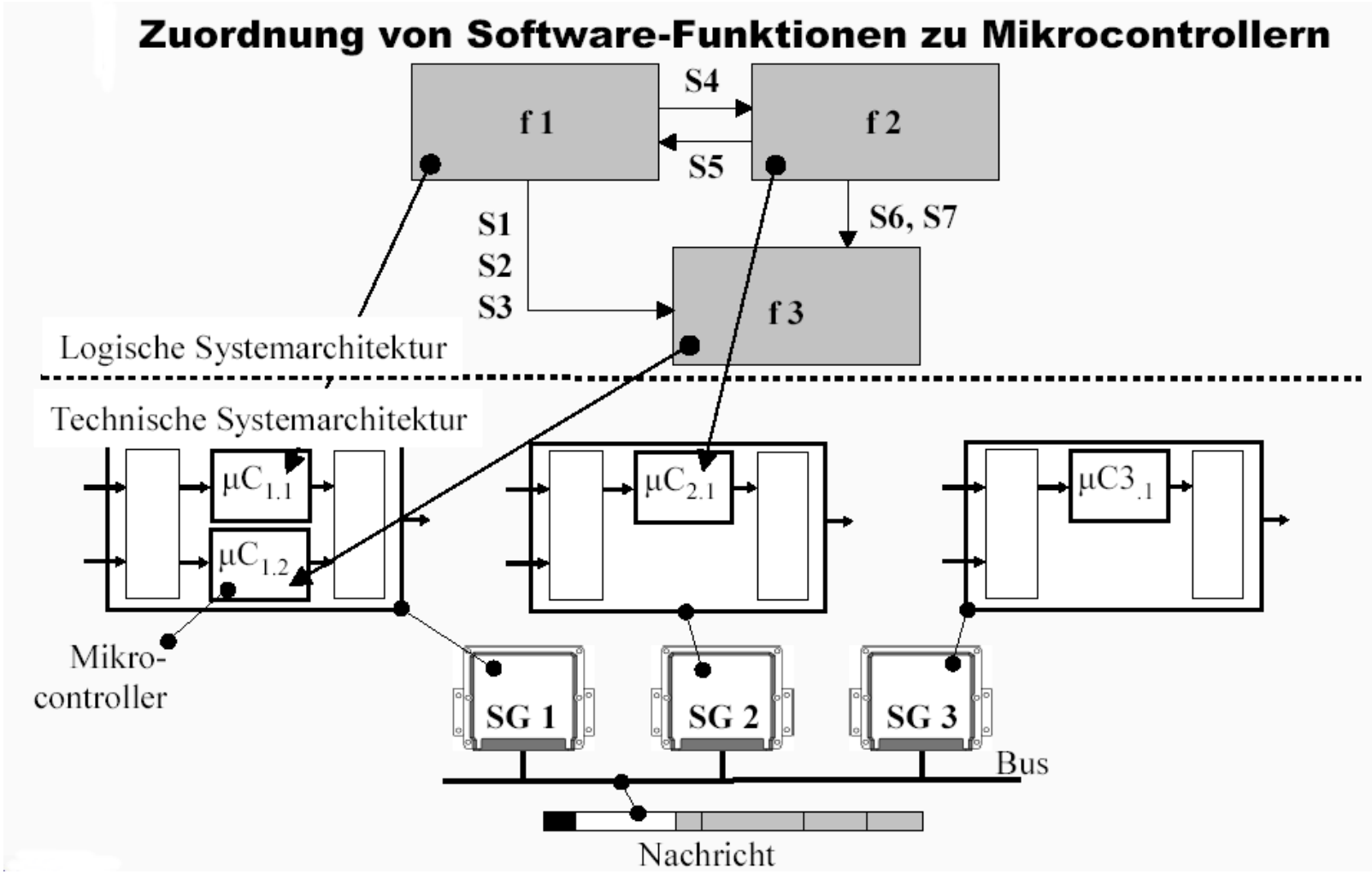


Analyse und Spezifikation von Echtzeitsystemen

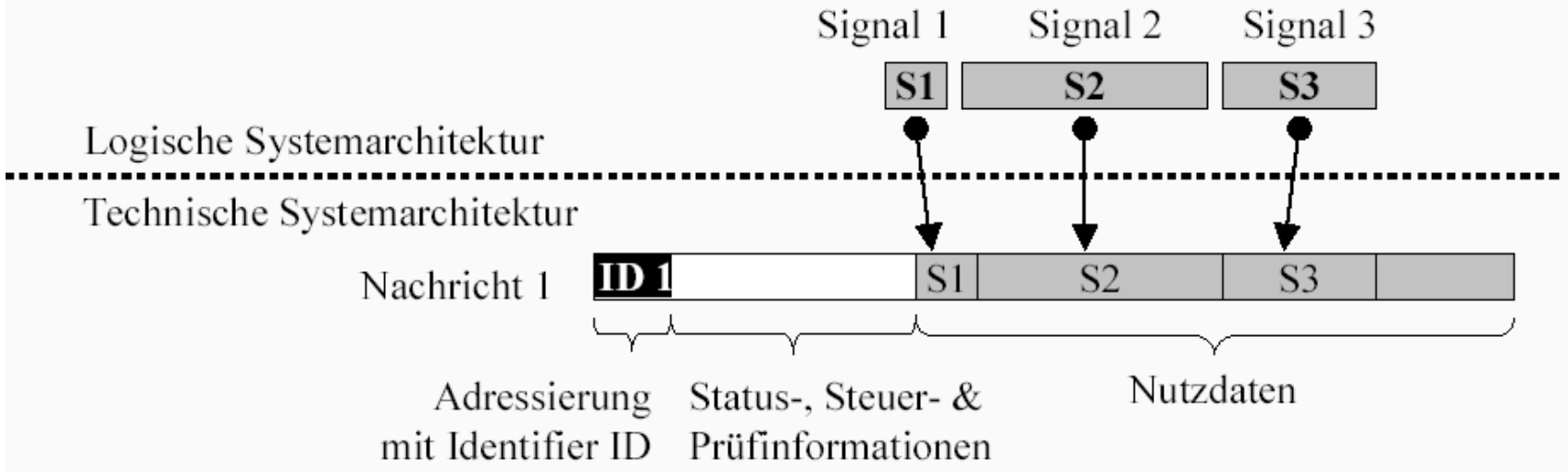
- Bei Analyse und Spezifikation steuerungs- und regelungstechnischer Systeme werden Anforderungen bezüglich der Abtastrate festgelegt.
- Abtastraten bilden die Basis für Echtzeitanforderungen an Software-Funktionen.
- Bei verteiltem, vernetztem System ergeben sich aus den Abtastraten auch die Echtzeitanforderungen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

Analyse und Spezifikation der technischen Anforderungen




Zuordnung von Software-Funktionen zu Mikrocontrollern
f 1
f 2
f 3
S4
S5
S1
S2
S3
S6, S7
Logische Systemarchitektur
Technische Systemarchitektur
µC1.1
µC1.2
µC2.1
µC3.1
Mikro-
controller
SG 1
SG 2
SG 3
Bus
Nachricht

Zuordnung von Signalen zu Nachrichten

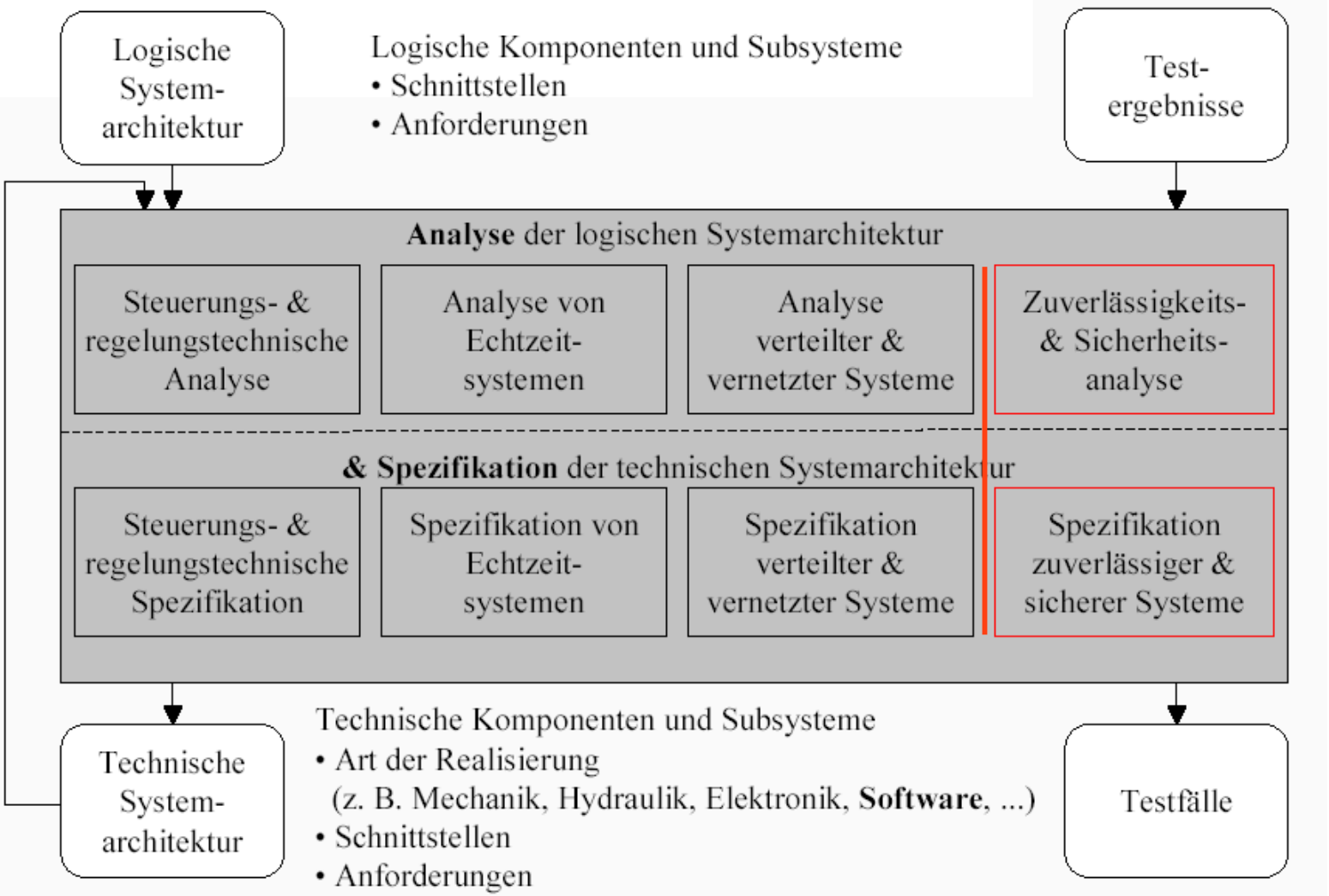


vgl. 3. Automotive Elektrik/Elektronik-Entwicklung (E/E)

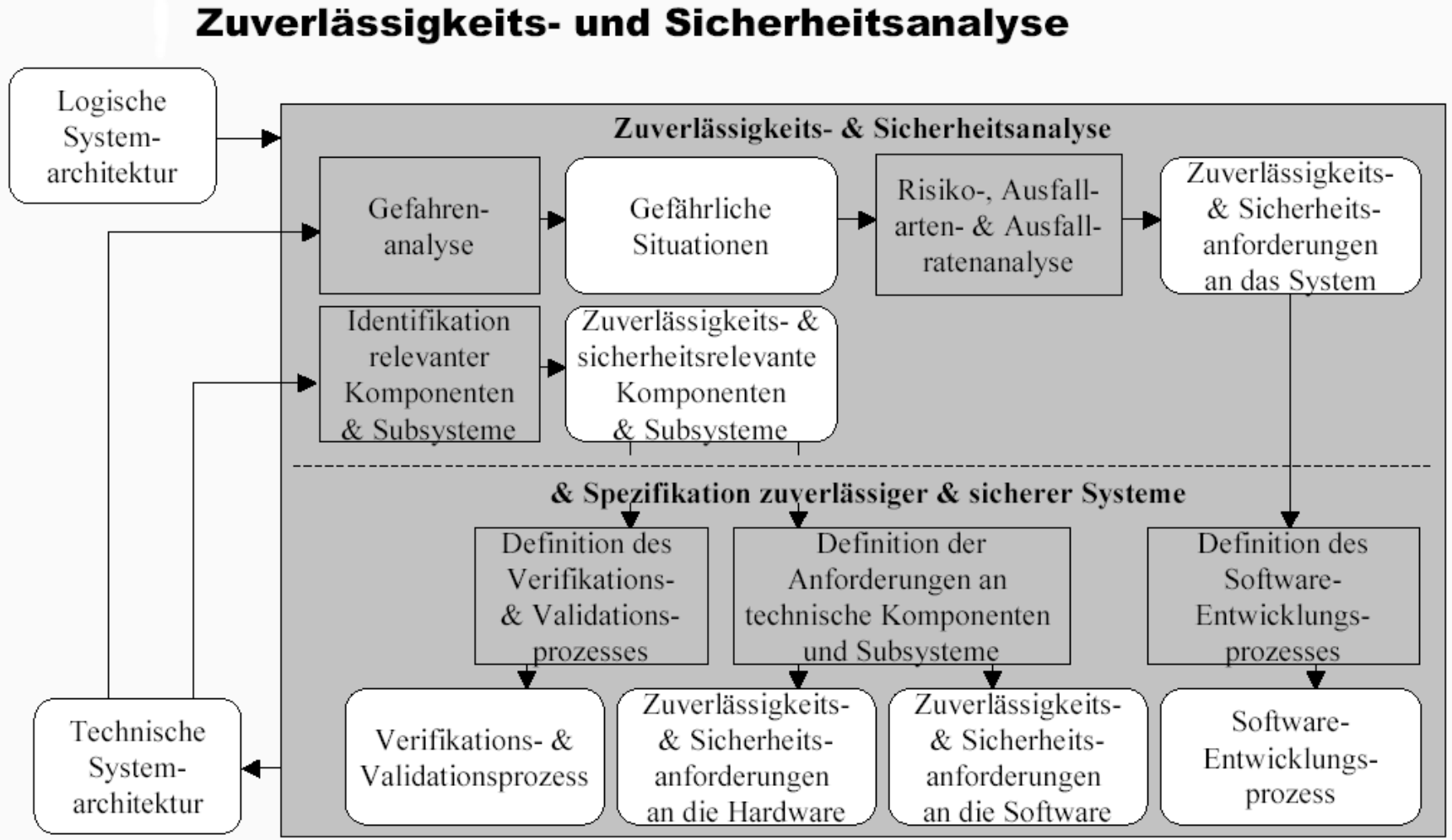
5. Bussysteme im Automobil

3. Zeitverhalten

Analyse und Spezifikation der technischen Anforderungen



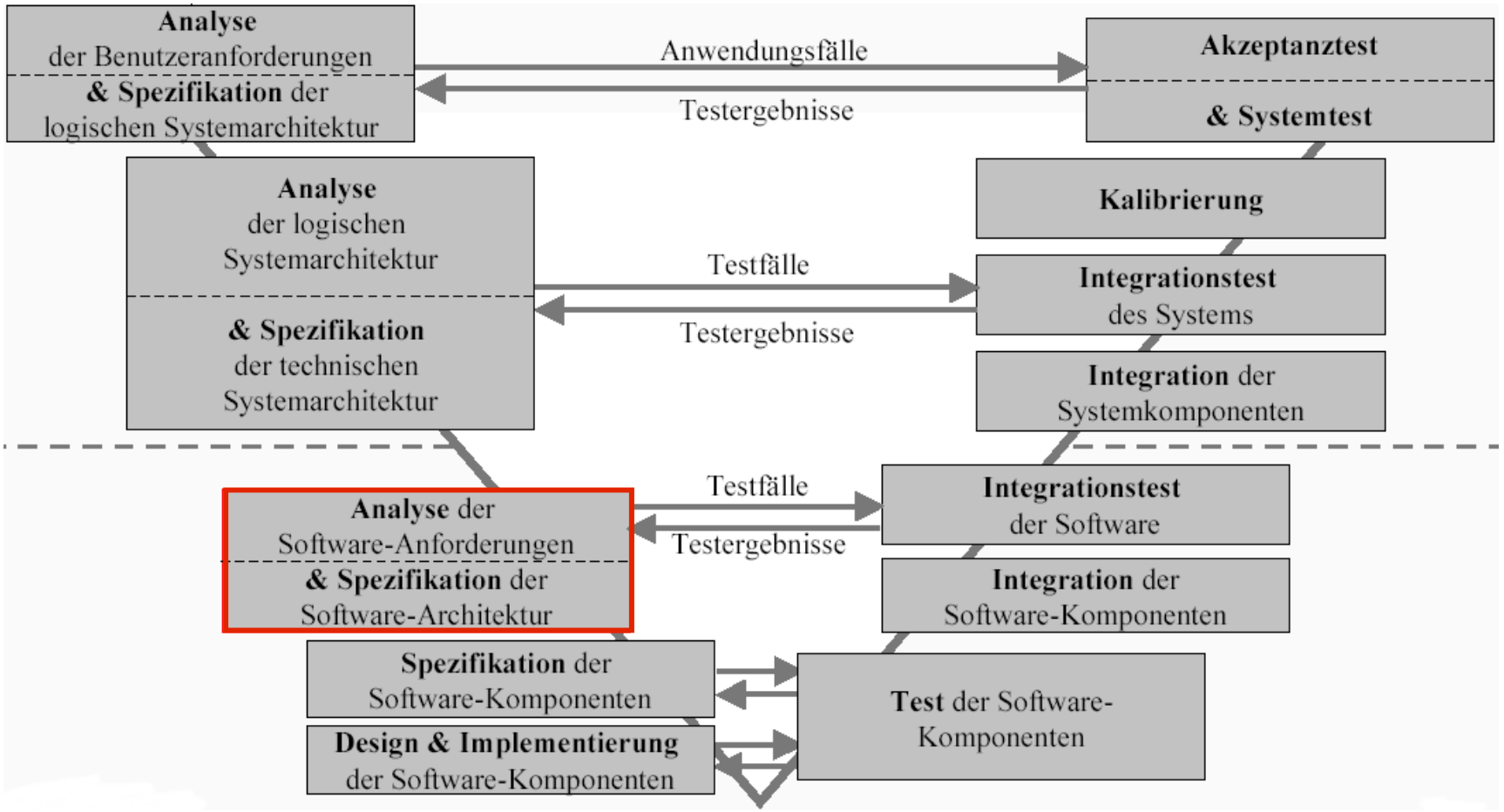
Analyse und Spezifikation der technischen Anforderungen



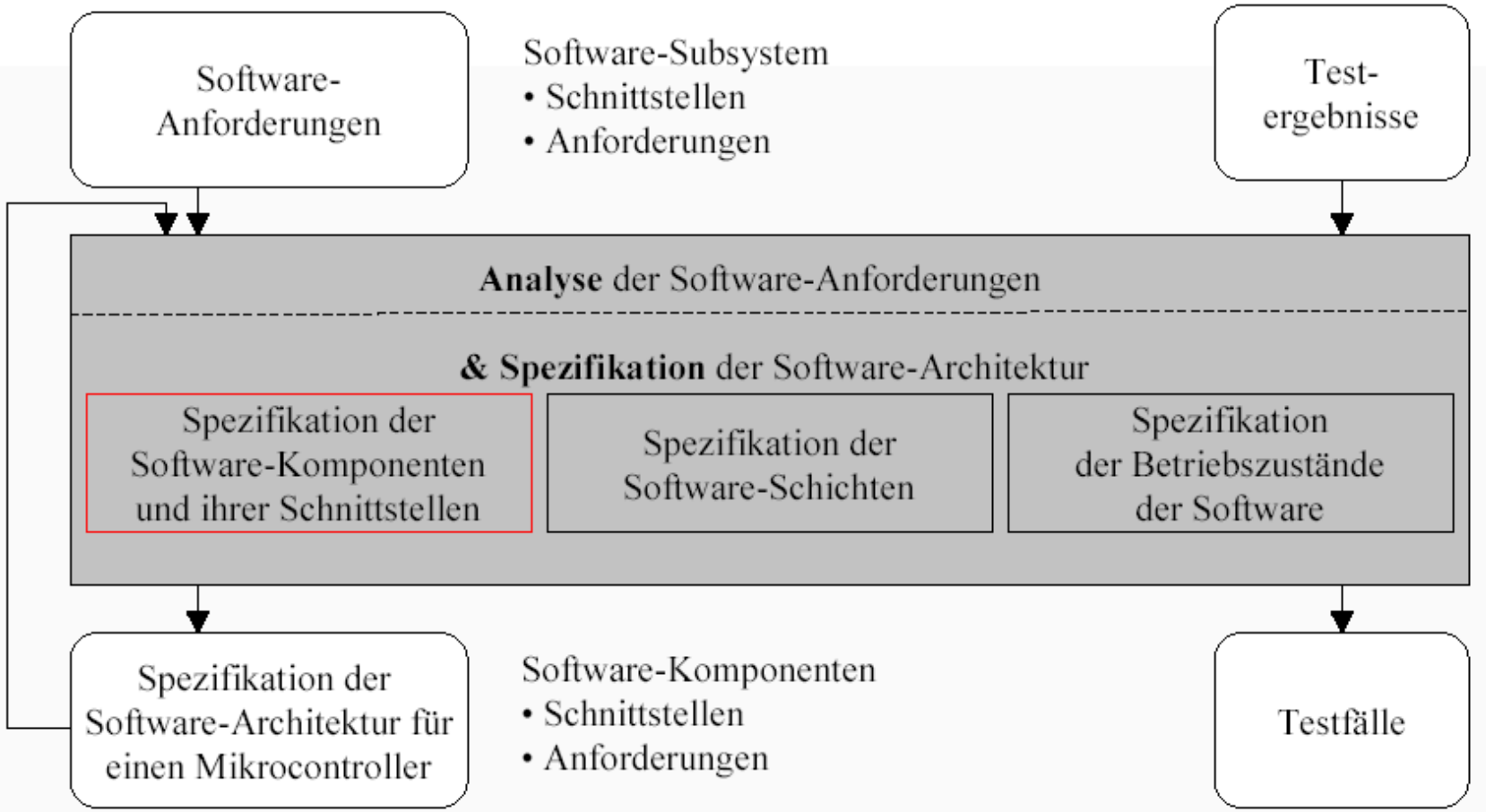
4. Kernprozess zur Entwicklung von elektronischen Systemen und Software

1. Grundbegriffe
2. Entwicklungsobjekt: Kombiinstrument
3. Analyse und Spezifikation der Benutzeranforderungen
4. Analyse und Spezifikation der technischen Anforderungen
5. **Analyse und Spezifikation der Software-Anforderungen**
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-Komponenten
11. Integration der System-Komponenten
12. Integrationstest des Systems
13. Kalibrierung
14. Akzeptanz- und Systemtest

Analyse und Spezifikation der Software-Anforderungen



Analyse und Spezifikation der Software-Anforderungen



Analyse und Spezifikation der Software-Anforderungen

Daten vs. Kontrolle

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

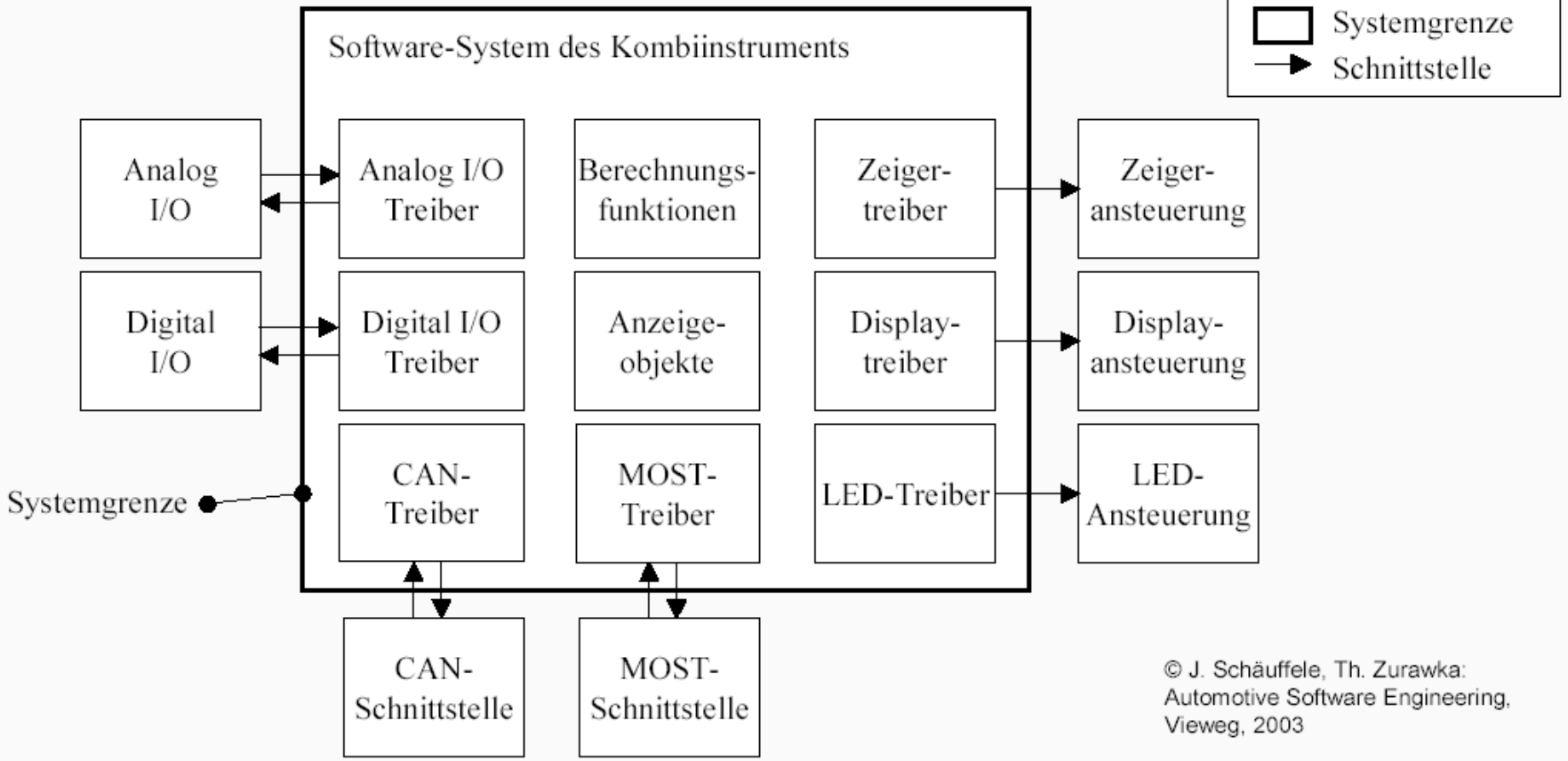
Beispiel

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

Spezifikation der On-Board-Schnittstellen

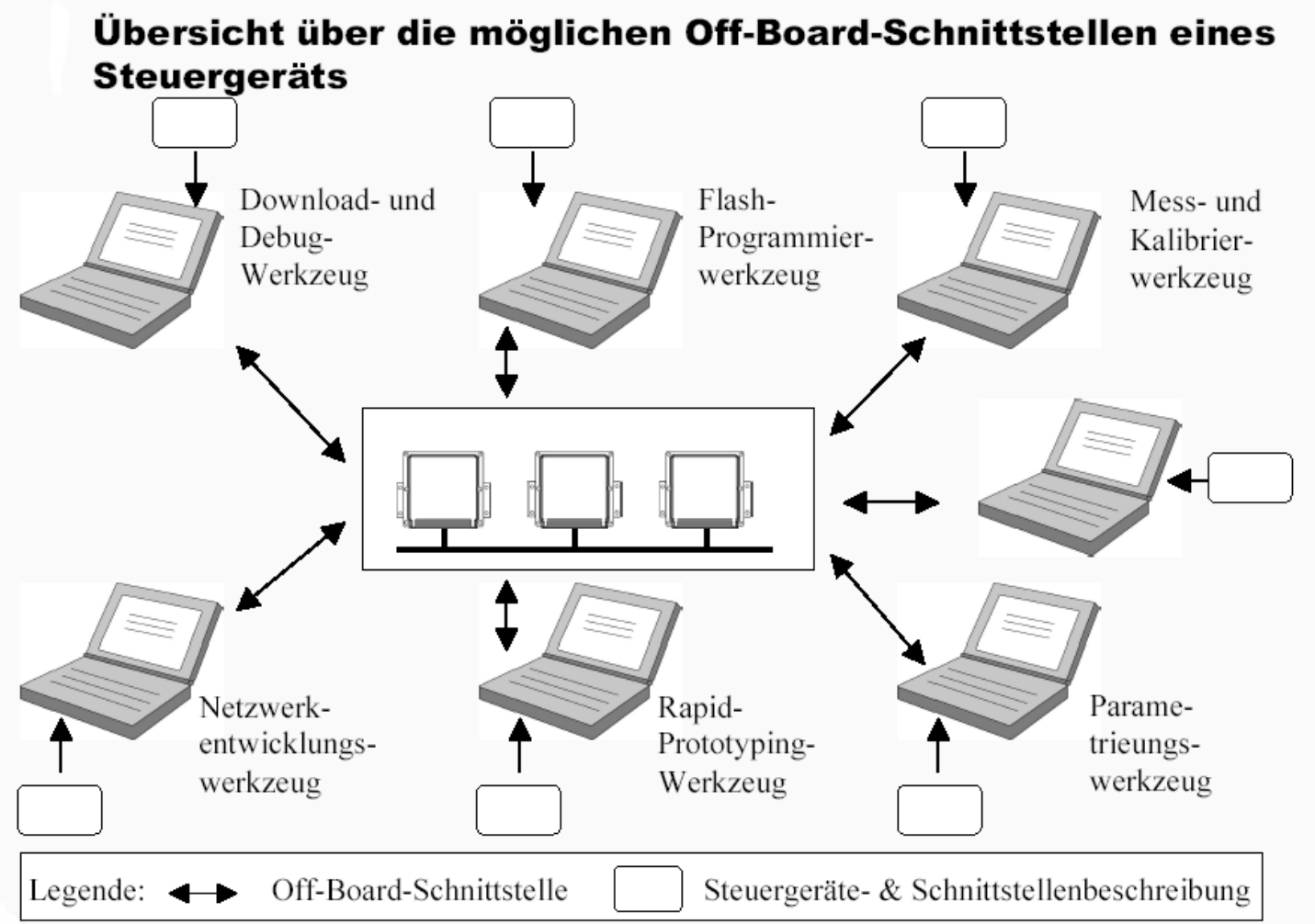
- Software-System und seine Umgebung (*Kontext*)
- *On-Board-Schnittstellen* des Steuergeräts zu Sollwertgebern, Sensoren, Aktuatoren, On-Board-Kommunikation mit anderen Steuergeräten

Kontext- und Schnittstellenmodell der Software des Kombiinstruments

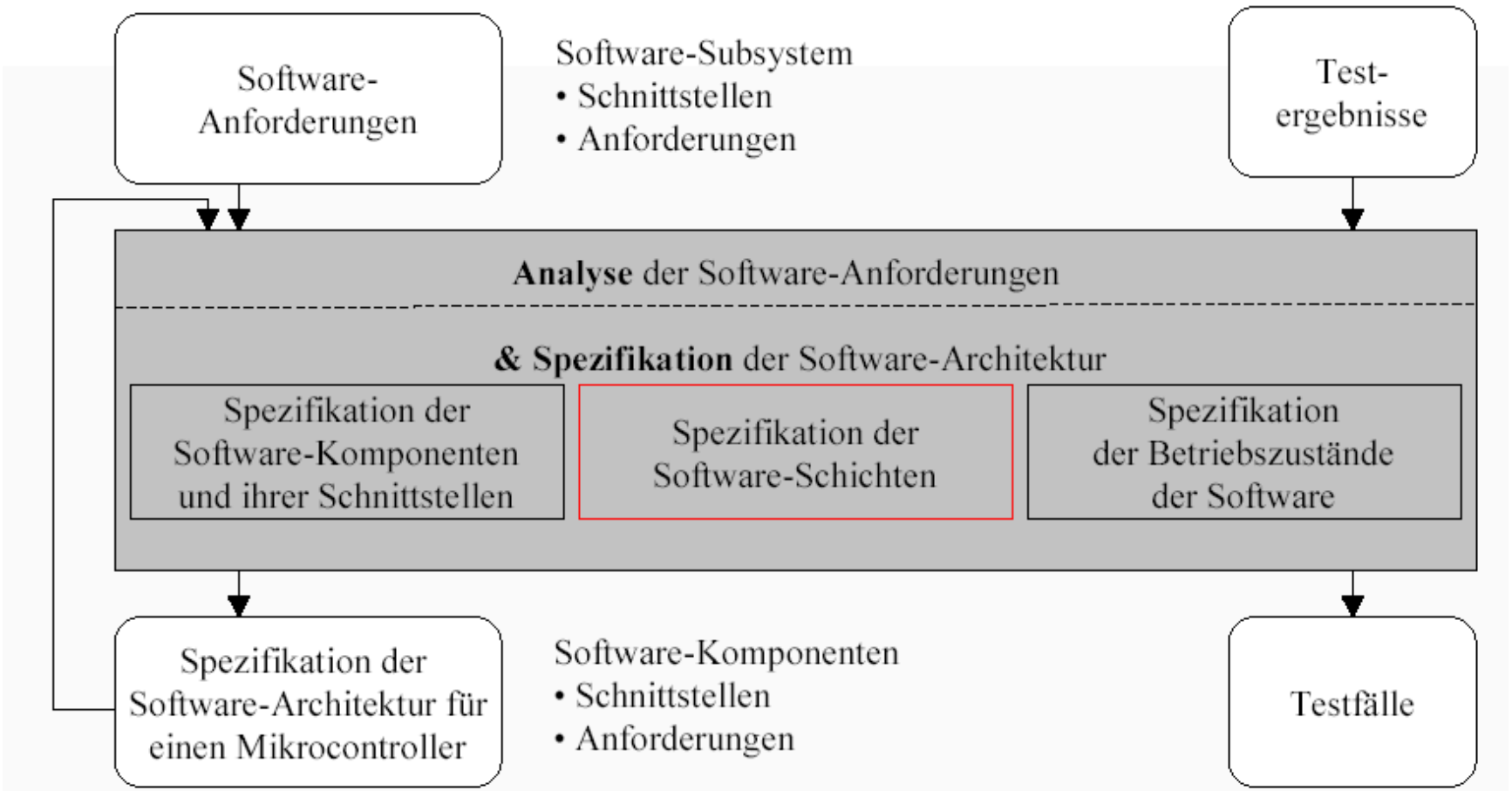


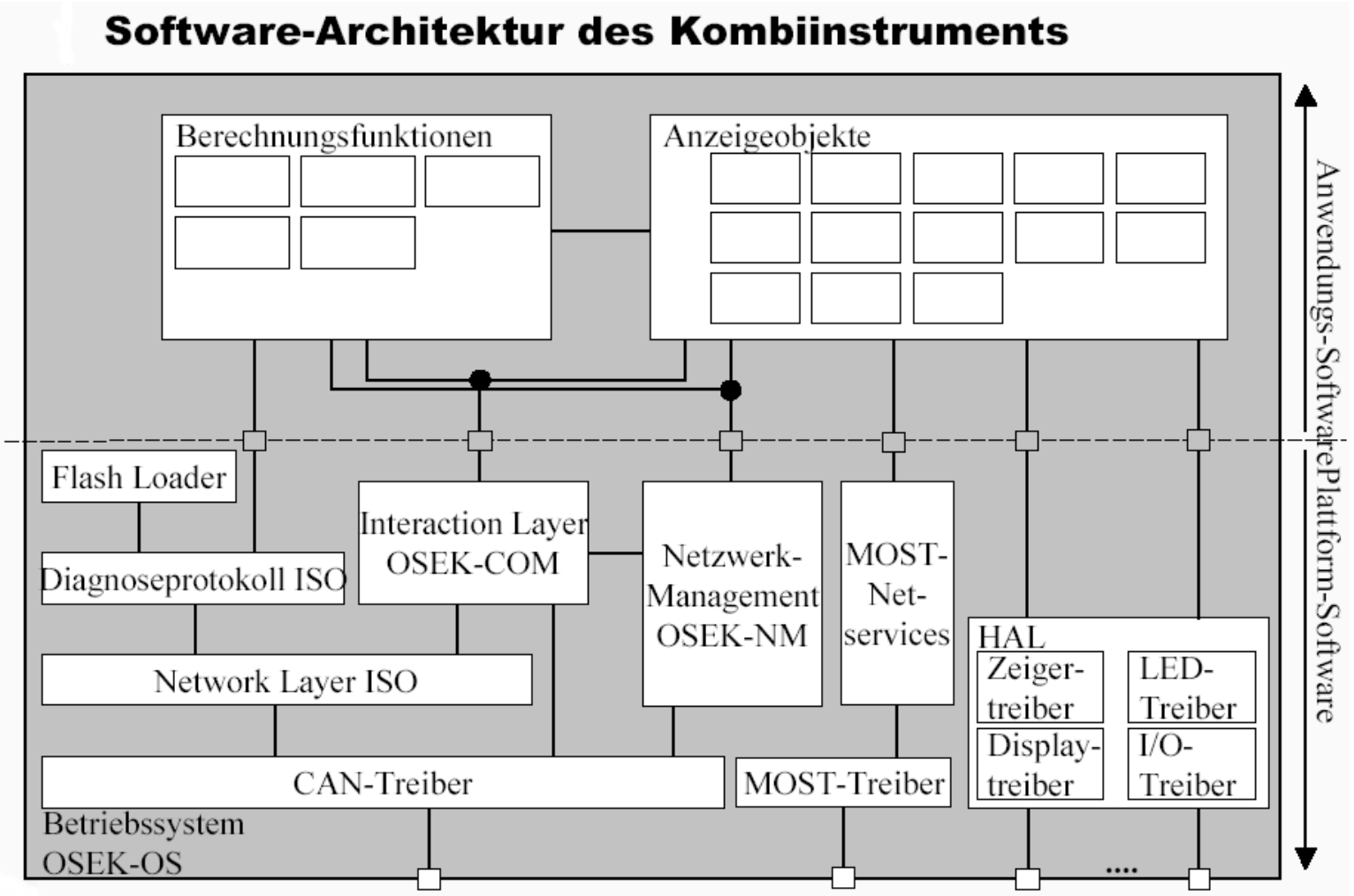
Spezifikation der Off-Board-Schnittstellen

- 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

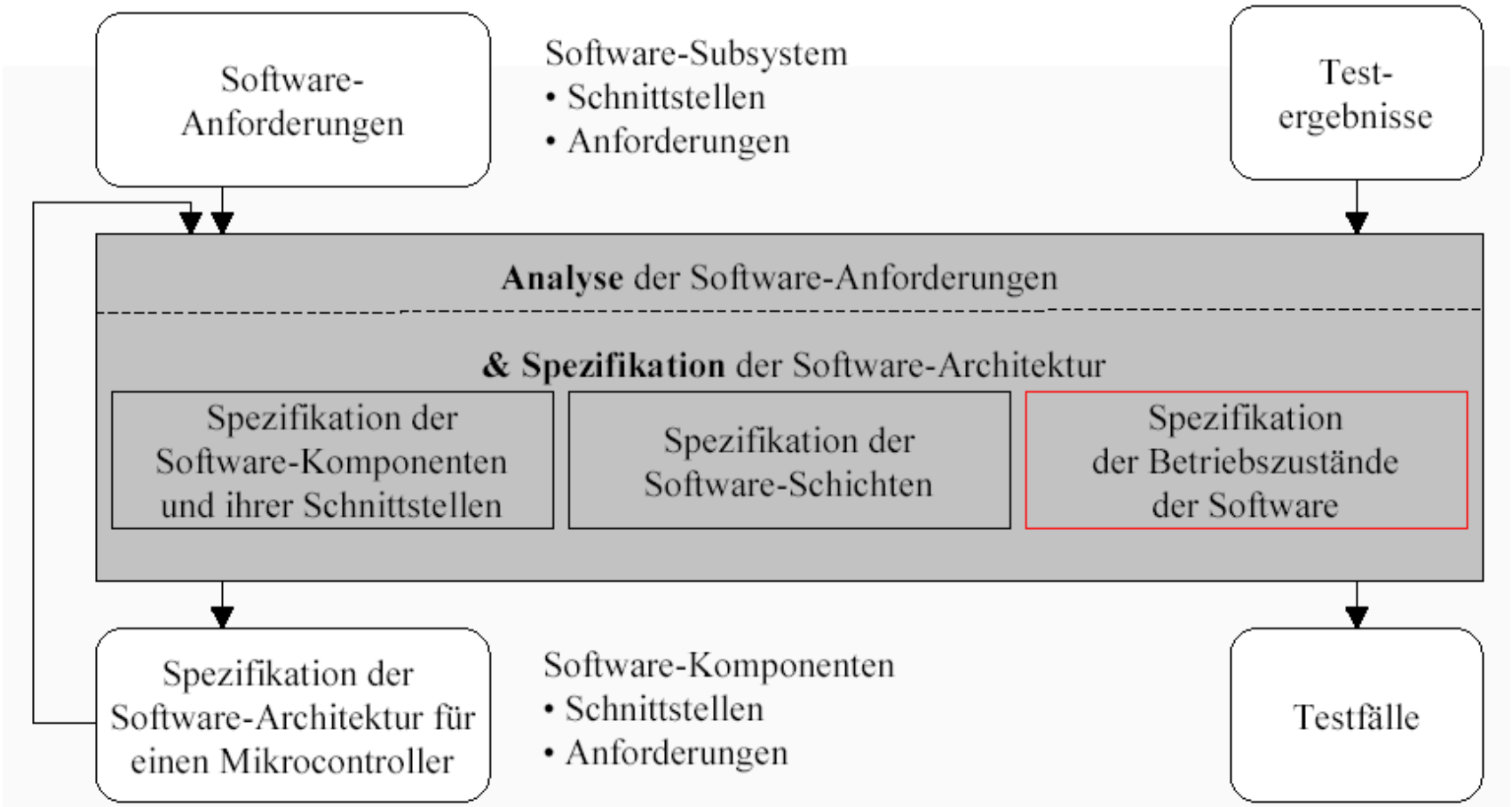

Übersicht über die möglichen Off-Board-Schnittstellen eines Steuergeräts
Download- und Debug-Werkzeug
Flash-Programmier-werkzeug
Mess- und Kalibrier-werkzeug
Netzwerk-entwicklungs-werkzeug
Rapid-Prototyping-Werkzeug
Parame-trieungs-werkzeug
Legende:
Off-Board-Schnittstelle
Steuergeräte- & Schnittstellenbeschreibung

Analyse und Spezifikation der Software-Anforderungen




Software-Architektur des Kombiinstruments
Berechnungsfunktionen
Anzeigeobjekte
Anwendungs-Softwareplattform-Software
Flash Loader
Diagnoseprotokoll ISO
Interaction Layer
OSEK-COM
Netzwerk-
Management
OSEK-NM
MOST-
Net-
services
HAL
Zeiger-
treiber
LED-
Treiber
Display-
treiber
I/O-
Treiber
Network Layer ISO
CAN-Treiber
MOST-Treiber
Betriebssystem
OSEK-OS
....

Analyse und Spezifikation der Software-Anforderungen

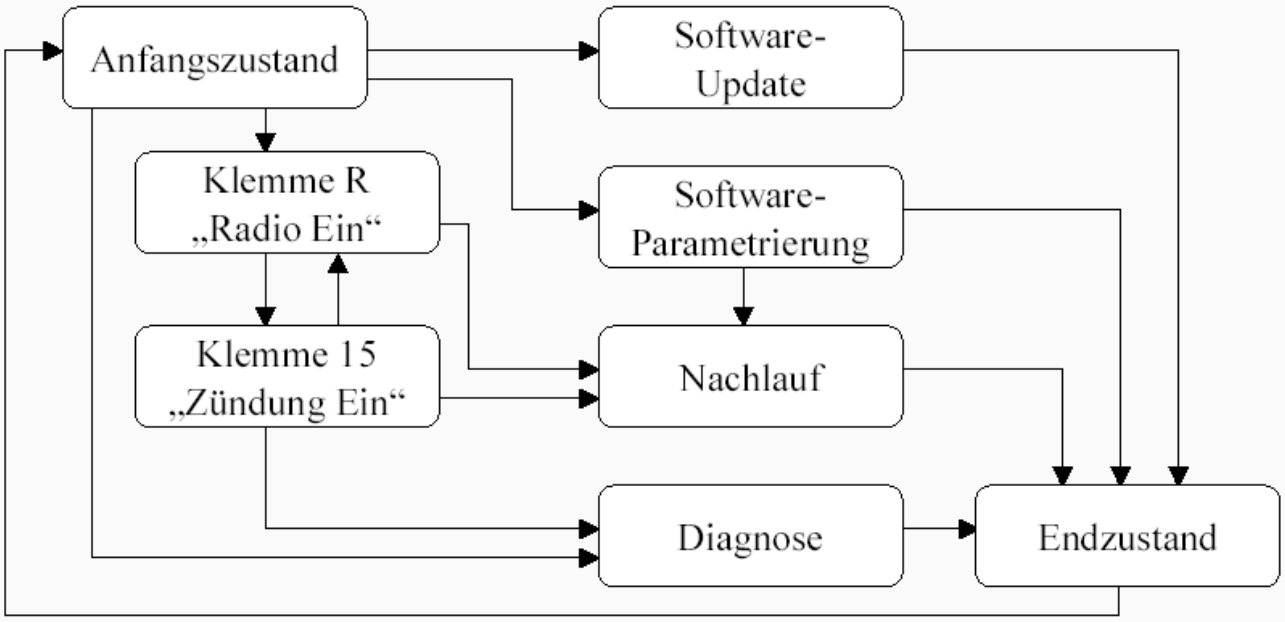


Betriebszustände

- Normalbetrieb
- Betriebszustand für Diagnose oder Update
- Notlauf

Spezifikation durch Zustandsautomaten

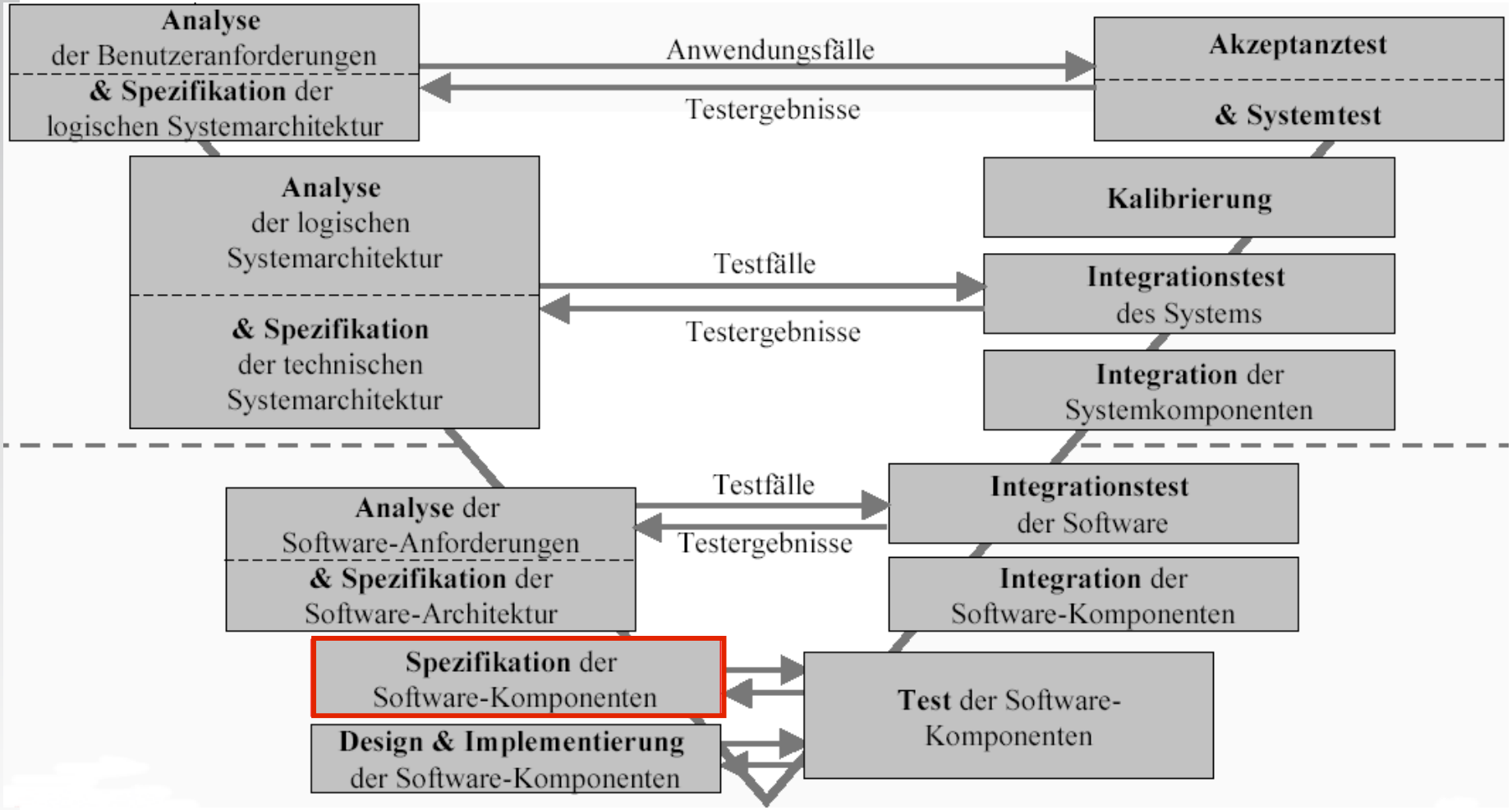
Betriebszustände und Übergänge für das Kombiinstrument



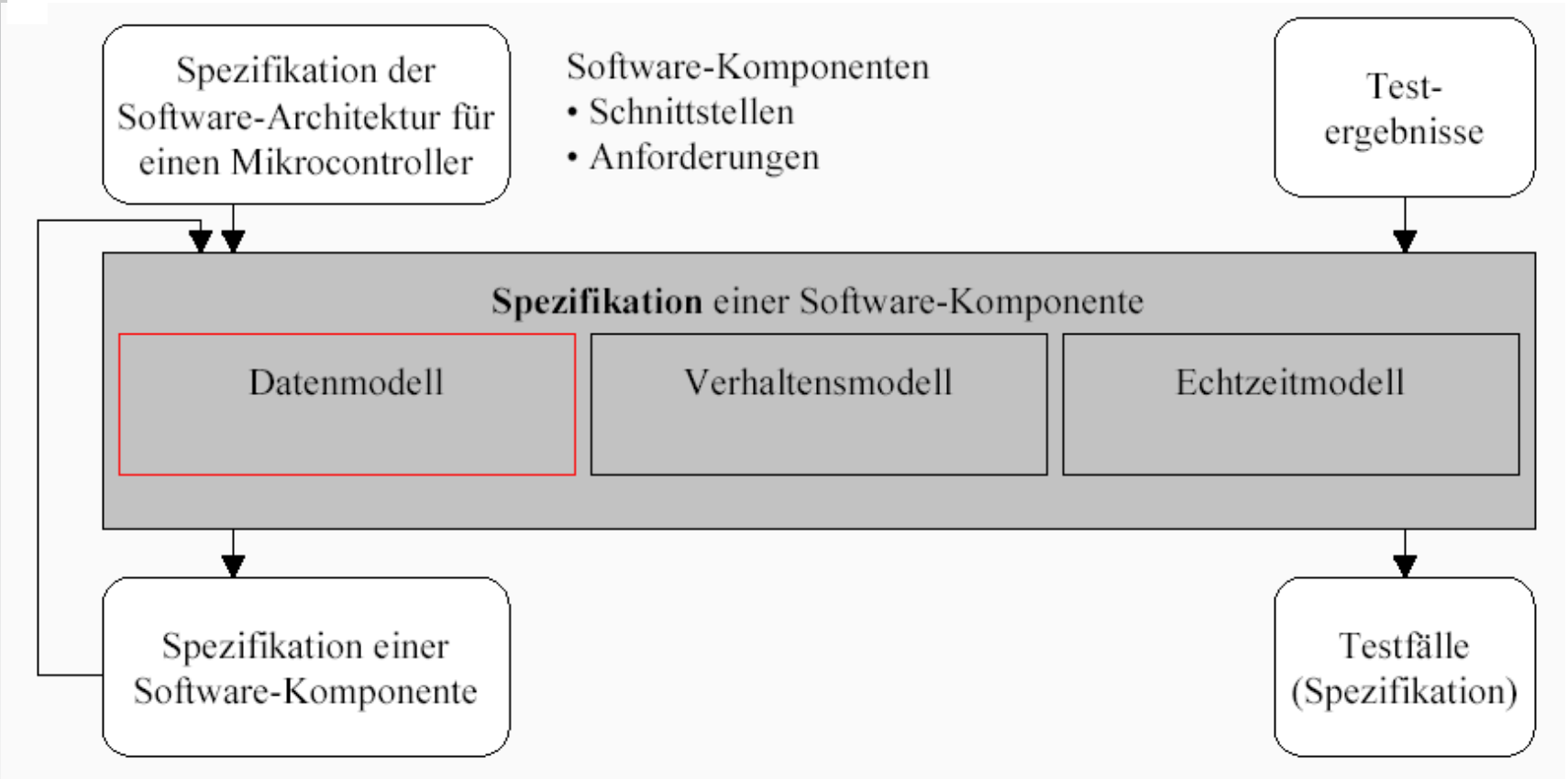
4. Kernprozess zur Entwicklung von elektronischen Systemen und Software

1. Grundbegriffe
2. Entwicklungsobjekt: Kombiinstrument
3. Analyse und Spezifikation der Benutzeranforderungen
4. Analyse und Spezifikation der technischen Anforderungen
5. Analyse und Spezifikation der Software-Anforderungen
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-Komponenten
11. Integration der System-Komponenten
12. Integrationstest des Systems
13. Kalibrierung
14. Akzeptanz- und Systemtest

Spezifikation der Software-Komponenten



Spezifikation der Software-Komponenten

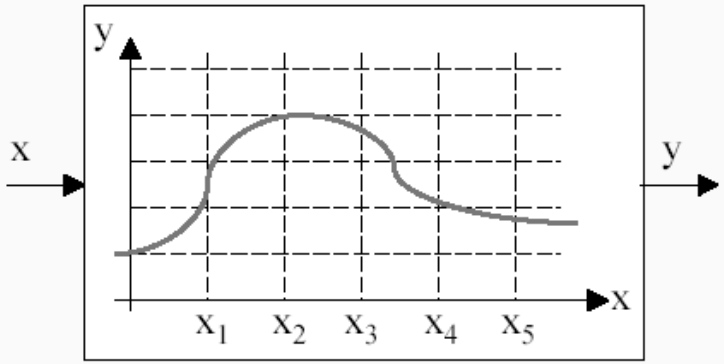


Datenstrukturen

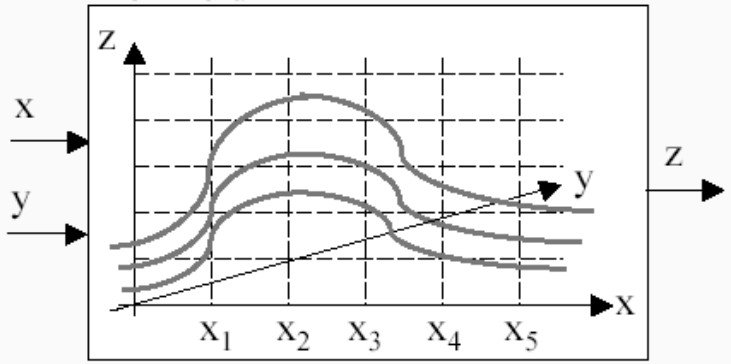
- Einfache Datenstrukturen
  (skalare Größen, Vektoren, Matritzen)
- komplexere Datenstrukturen

Grafische Darstellung:

Kennlinie



Kennfeld

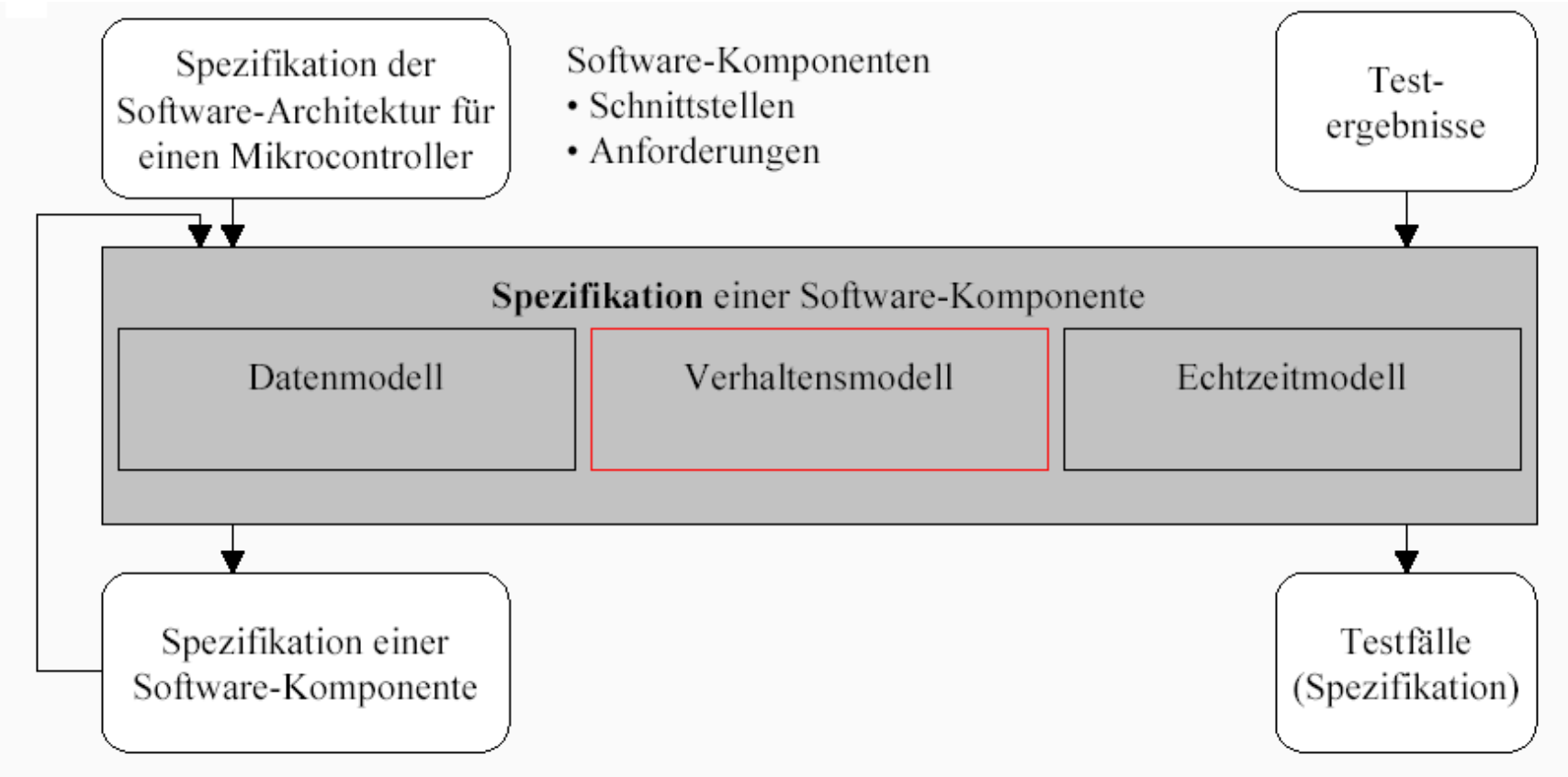


Tabellarische Darstellung:

x_1	x_2	x_3	x_4	x_5
y_1	y_2	y_3	y_4	y_5

	x_1	x_2	x_3	x_4	x_5
y_1	z_{11}	z_{12}	z_{13}	z_{14}	z_{15}
y_2	z_{21}	z_{22}	z_{23}	z_{24}	z_{25}
y_3	z_{31}	z_{32}	z_{33}	z_{34}	z_{35}

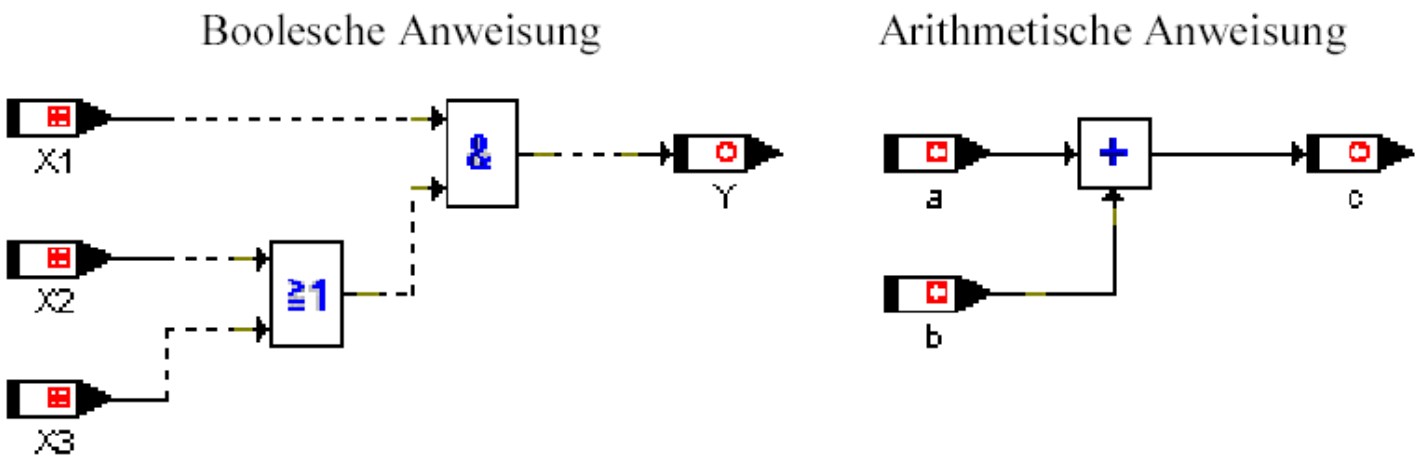
Spezifikation der Software-Komponenten



Spezifikation des Datenflusses

Beispiel: Datenfluss für eine boolsche und eine arithmetische Anweisung in ASCET-SD-Blockschaltbildern

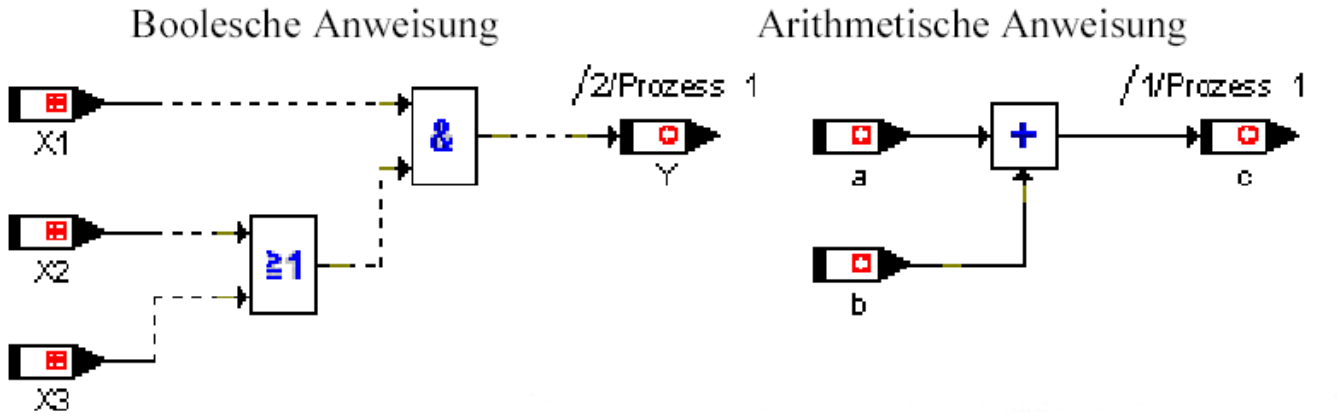
y= X1 & (X2 || X3)
c = a+b



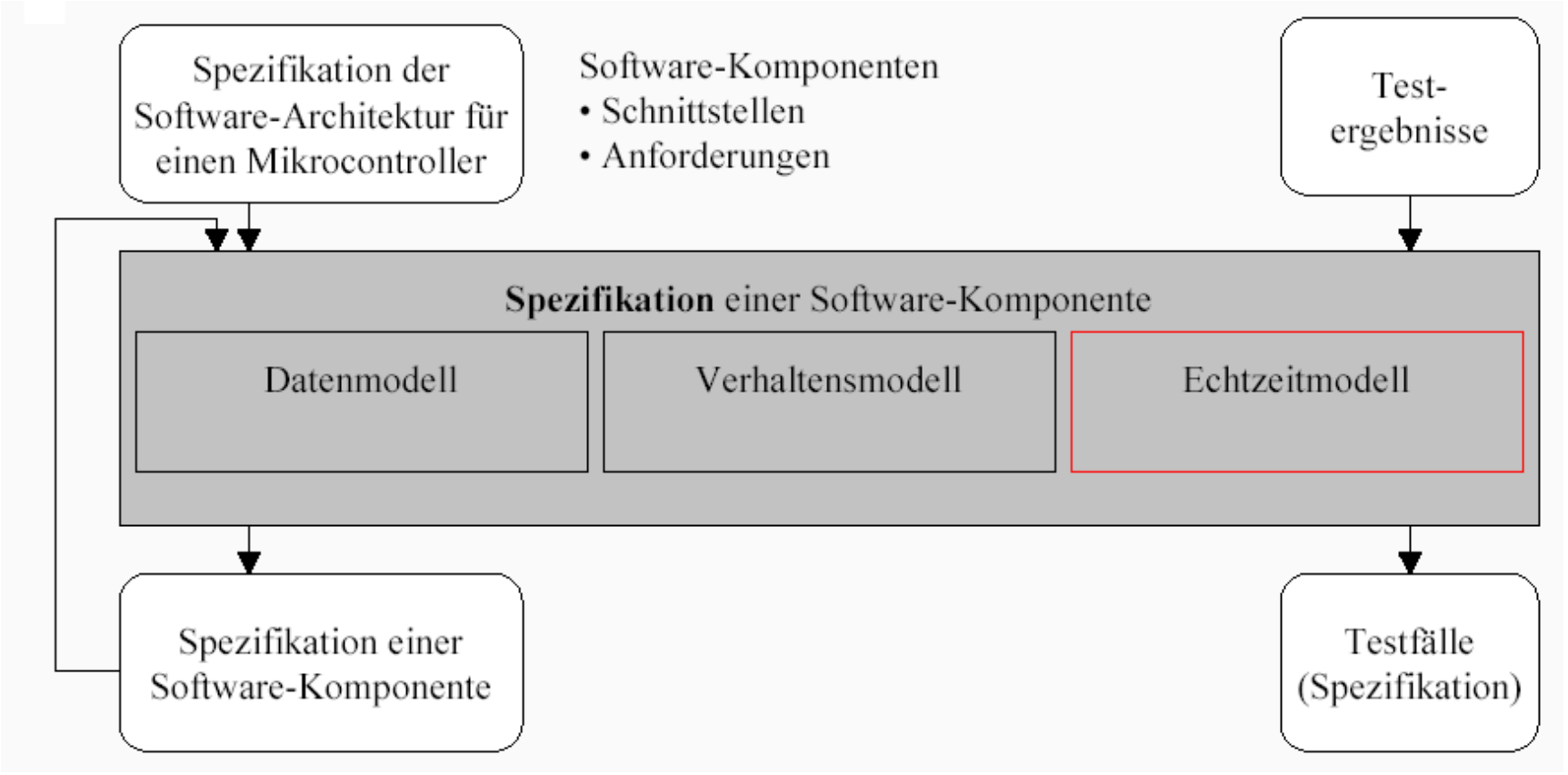
Spezifikation des Kontrollflusses

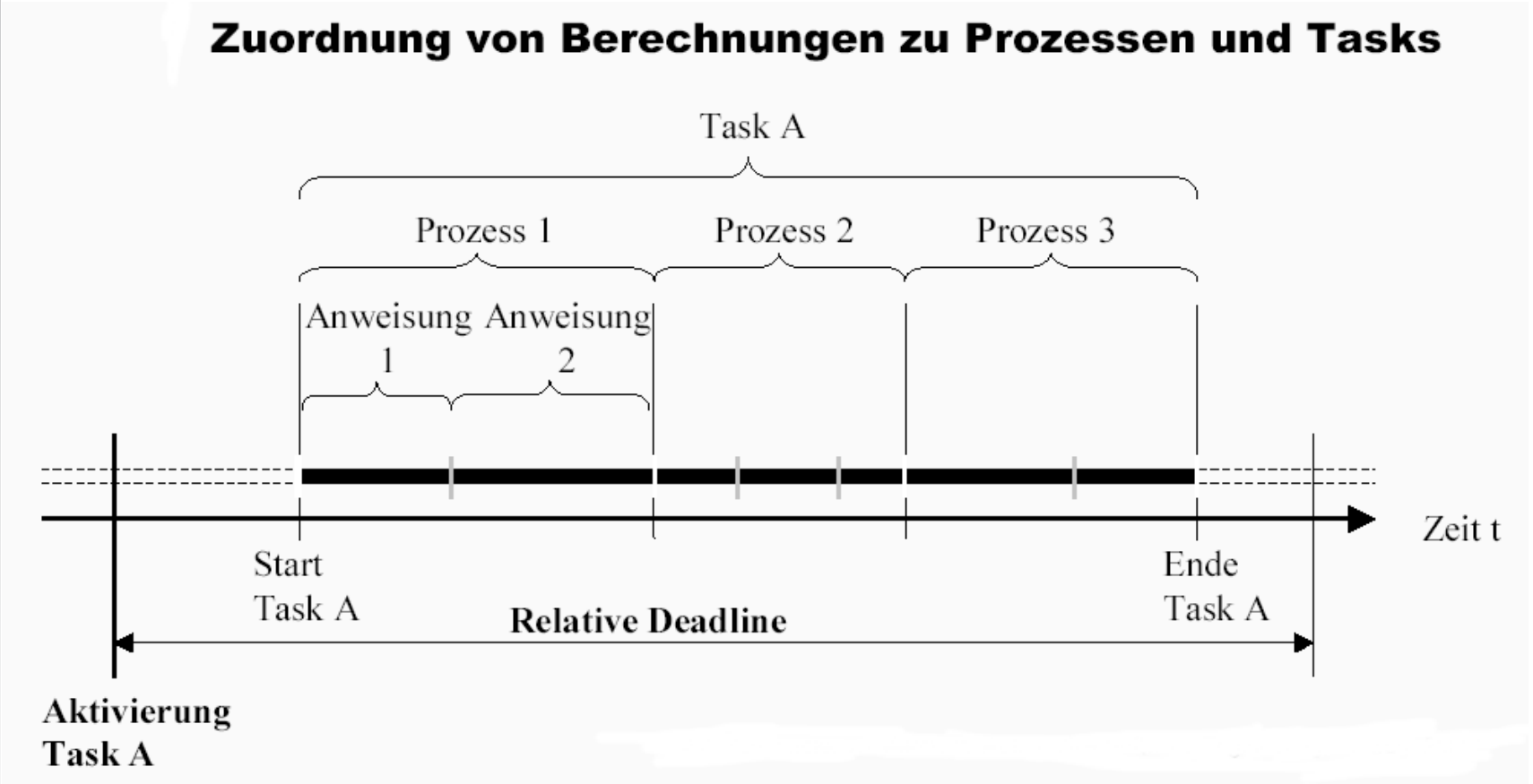
- klassische Ansätze aus der Programmierung, z.B. Struktogramme
- Ansätze aus dem objekt-orientierten Entwurf, z.B. für die Aufrufstruktur zwischen Software-Komponenten

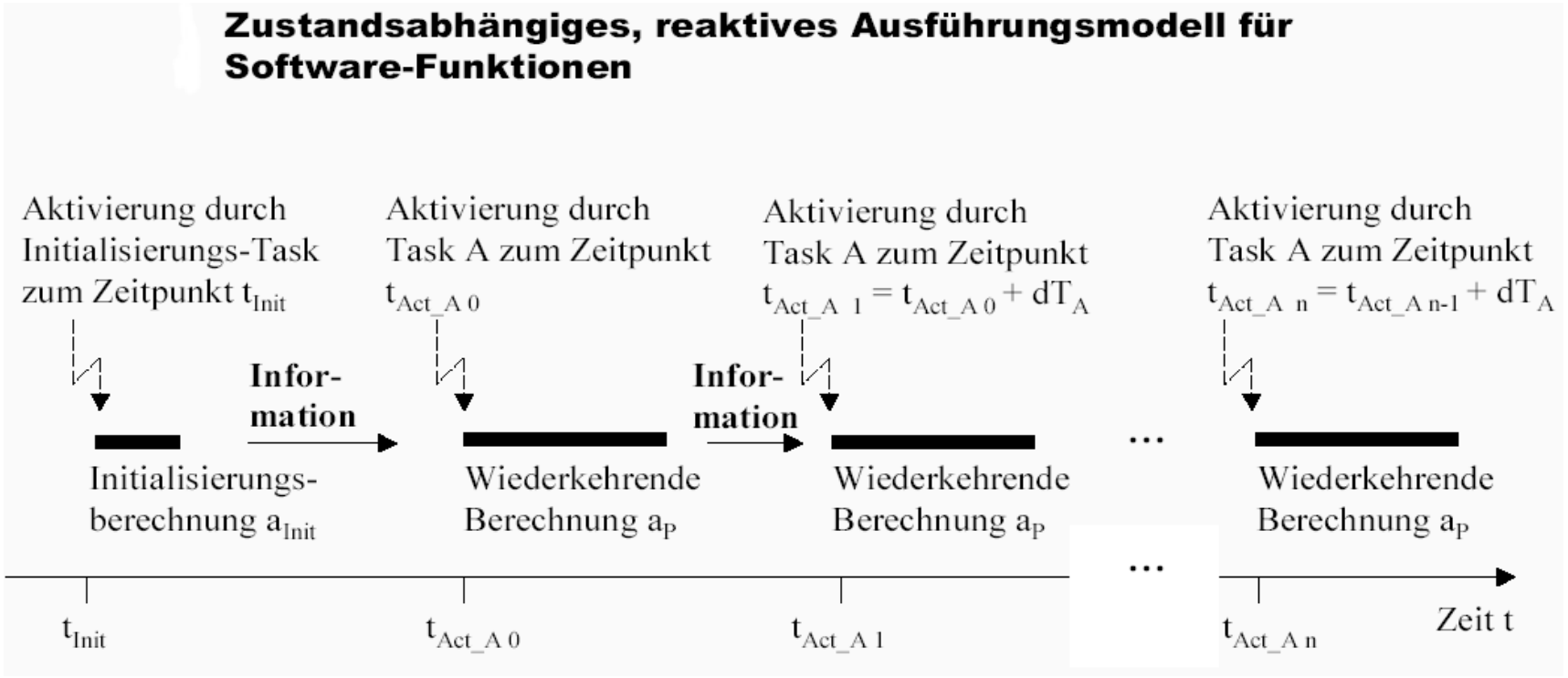
Kontrollfluss zur Festlegung der Ausführungsreihenfolge in ASCET-SD



Spezifikation der Software-Komponenten




Zuordnung von Berechnungen zu Prozessen und Tasks
Task A
Prozess 1
Prozess 2
Prozess 3
Anweisung 1
Anweisung 2
Zeit t
Start Task A
Ende Task A
Relative Deadline
Aktivierung Task A


Zustandsabhängiges, reaktives Ausführungsmodell für Software-Funktionen
Aktivierung durch Initialisierungs-Task zum Zeitpunkt t_{Init}
Aktivierung durch Task A zum Zeitpunkt $t_{Act\_A\ 0}$
Aktivierung durch Task A zum Zeitpunkt $t_{Act\_A\ 1} = t_{Act\_A\ 0} + dT_A$
Aktivierung durch Task A zum Zeitpunkt $t_{Act\_A\ n} = t_{Act\_A\ n-1} + dT_A$
Infor-mation
Infor-mation
Initialisierungs-berechnung a_{Init}
Wiederkehrende Berechnung a_P
Wiederkehrende Berechnung a_P
Wiederkehrende Berechnung a_P
t_{Init}
$t_{Act\_A\ 0}$
$t_{Act\_A\ 1}$
$t_{Act\_A\ n}$
Zeit t

Zustandsunabhängiges, reaktives Ausführungsmodell für Software-Funktionen