Vehicle concepts & -systems

Module 4 Basics to software development – part 2

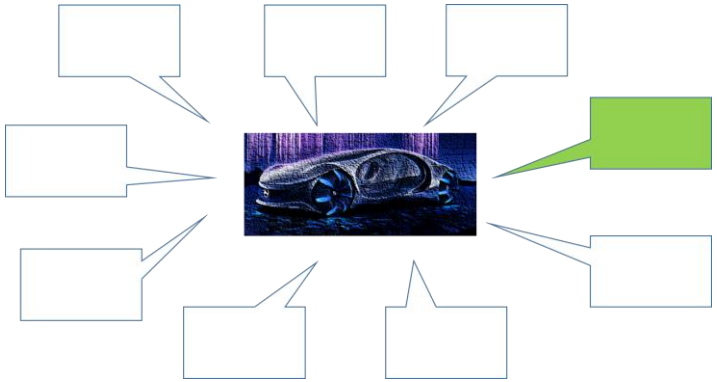
WS 2023/2024

Schedule

• Presentation & Feedback Teamwork module 4	08:00 – 09:00
• Software development – V Model part 2	09:00 – 10:00
• *Coffee break*	*10:00 – 10:15*
• Preperation of module 6 (practical experience)	10:15 – 12:00
• *Lunch break*	*12:00 – 13:00*
• Teamwork	13:00 – 16:00

Teamwork module 4

- Please choose one key mechatronic system of your vehicle concept and describe it from a customer perspective (requirements)
- Please describe on the basis of logical and technical layers the key function of this mechatronic system
- Which sensors do you need?
- Which voltage level(s) do you need? Why?
- For one function: please describe a (simple) PID controller



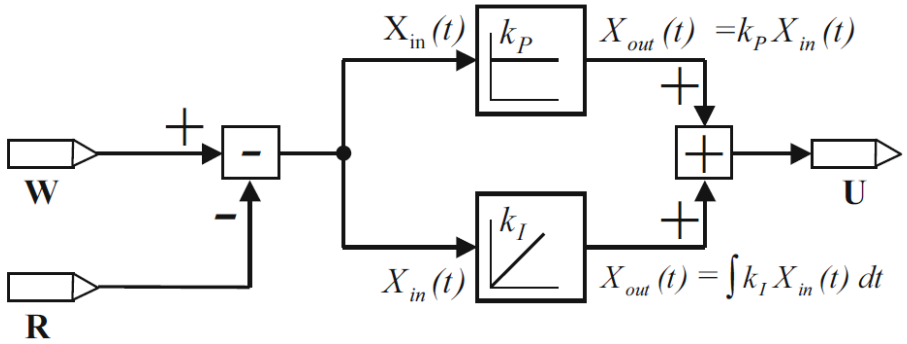
Design und Implementation of software components

Common CASE-Tools:

- ASCET (ETAS)
- MATLAB/Simulink (Mathworks)
- TargetLink (dSpace)

W = Führungsgröße
R = Regelgröße
U = Ausgangsgröße

Beispiel: ASCET ®



[Sch16]

PID controller - basics

P-Anteil:

- Nur proportionaler Anteil (Verstärkung Kp). Ausgangssignal (u) ist proportional zum Eingangssignal €

$$u(t) = K_p \cdot e(t)$$

- Kein Zeitverhalten; daher sofortige Regelung

I-Anteil:

- Integrierender Regler; zeitliche Integration der Regelabweichung e(t) auf die Stellgröße mit der Gewichtung durch die Nachstellzeit (Tn)

$$u(t) = \frac{1}{T_N} \int_0^t e(\tau) d\tau$$

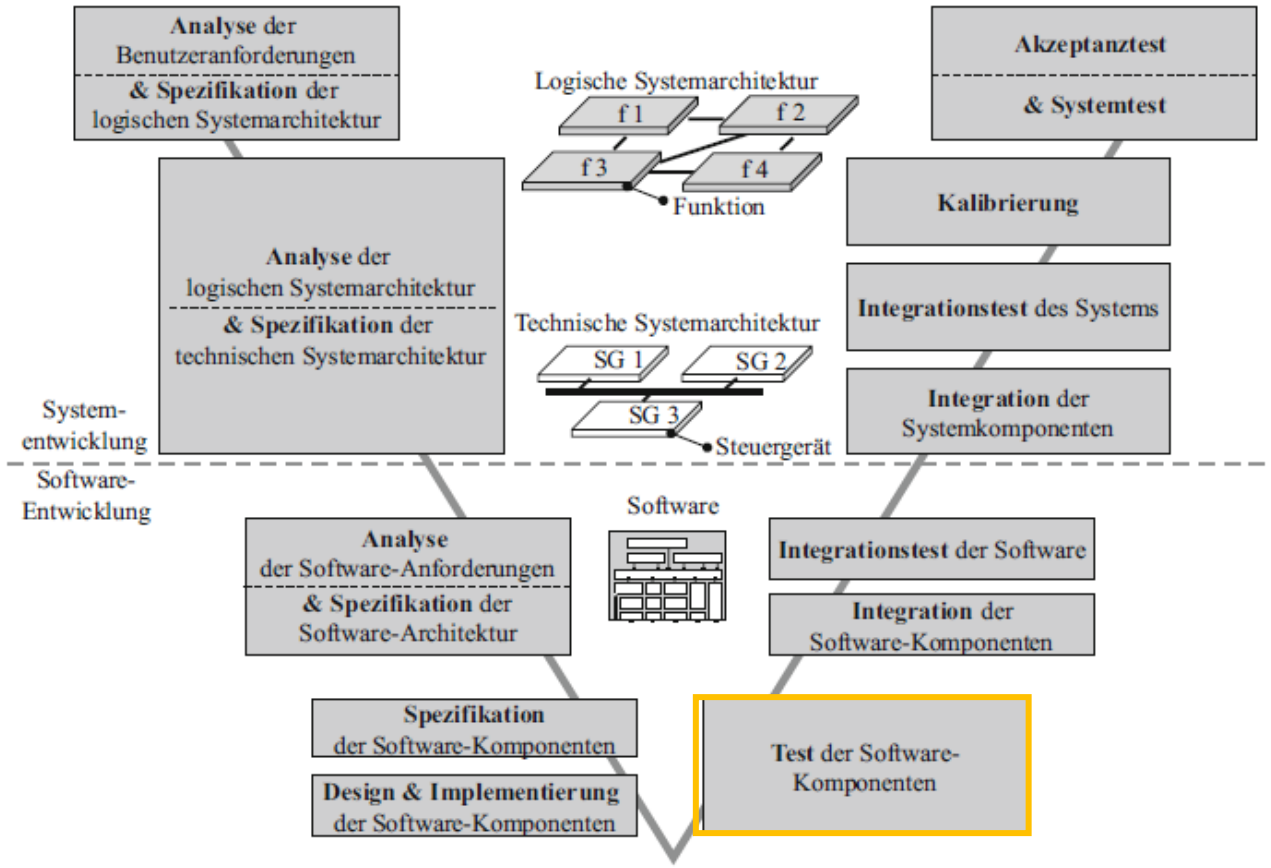
- Langsamer, genauer Regler

D-Glied:

- Differenzierer, der in Verbindung mit P/I Regler eingesetzt wird; Reaktion nur auf Änderungsgeschwindigkeit, nicht Regelabweichung

$$u(t) = T_\mathrm{V} \frac{\mathrm{d}}{\mathrm{d}t} e(t)$$

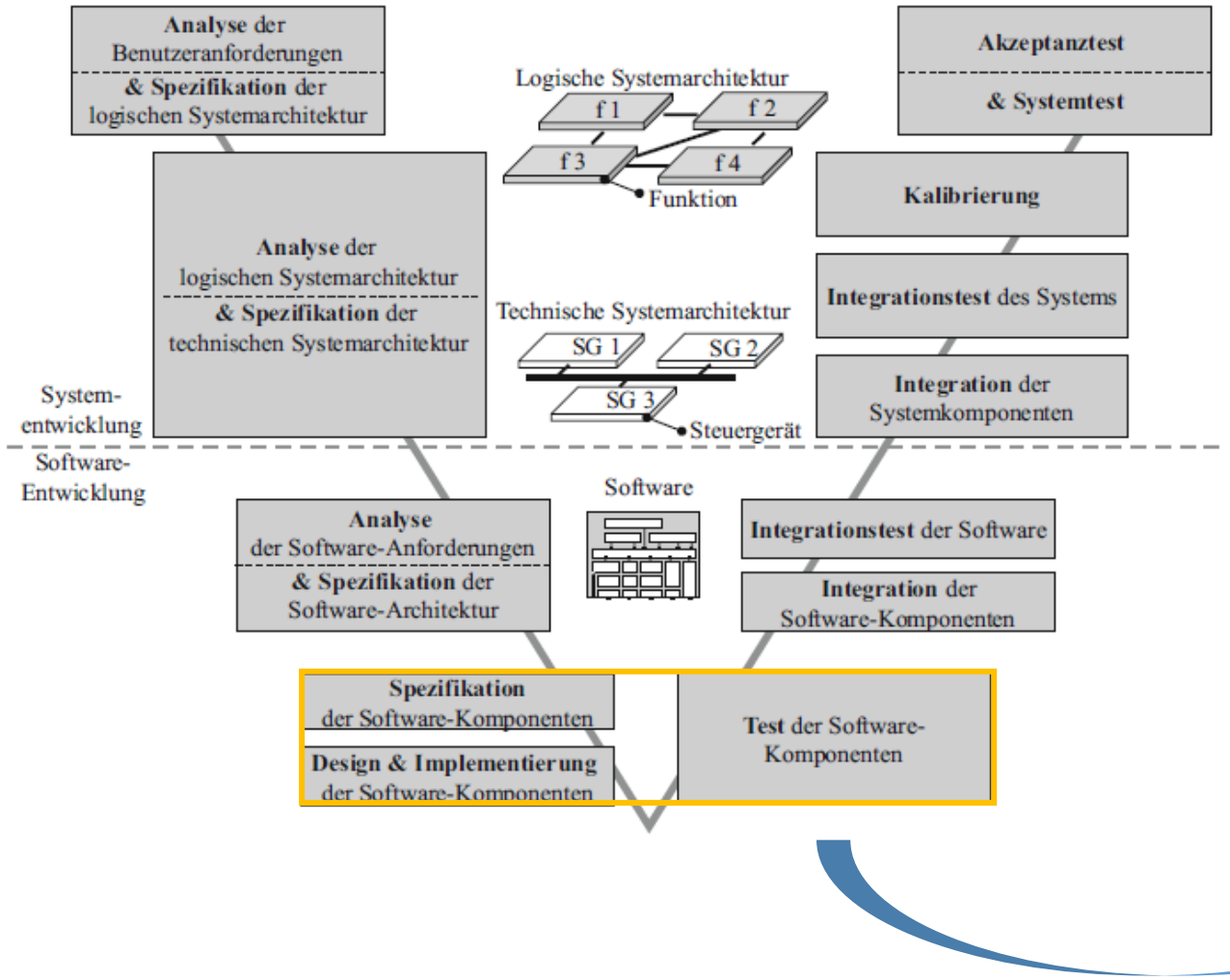
Das V-Modell – Test der Software-Komponente



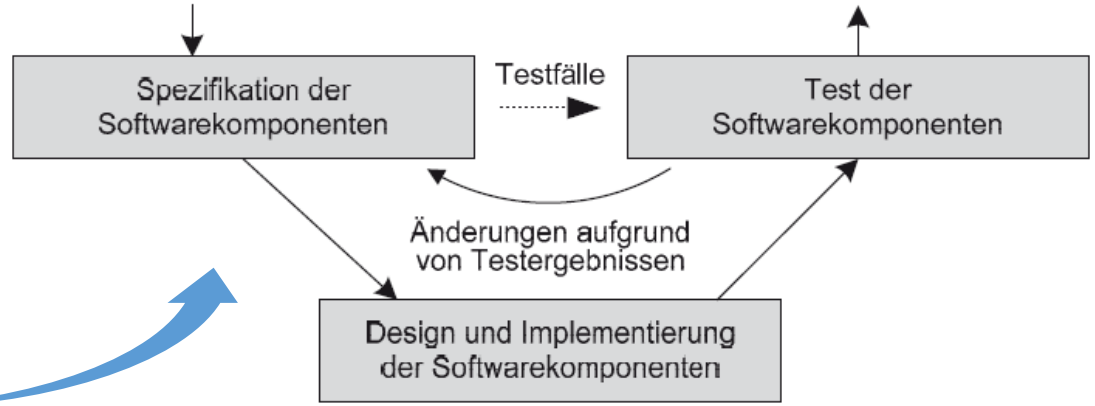
- Definition of test cases based on the software specification
- Through variation of input measures of the software component, the behaviour of the output signal can be observed

[Sch16]

Iterations based on component testing

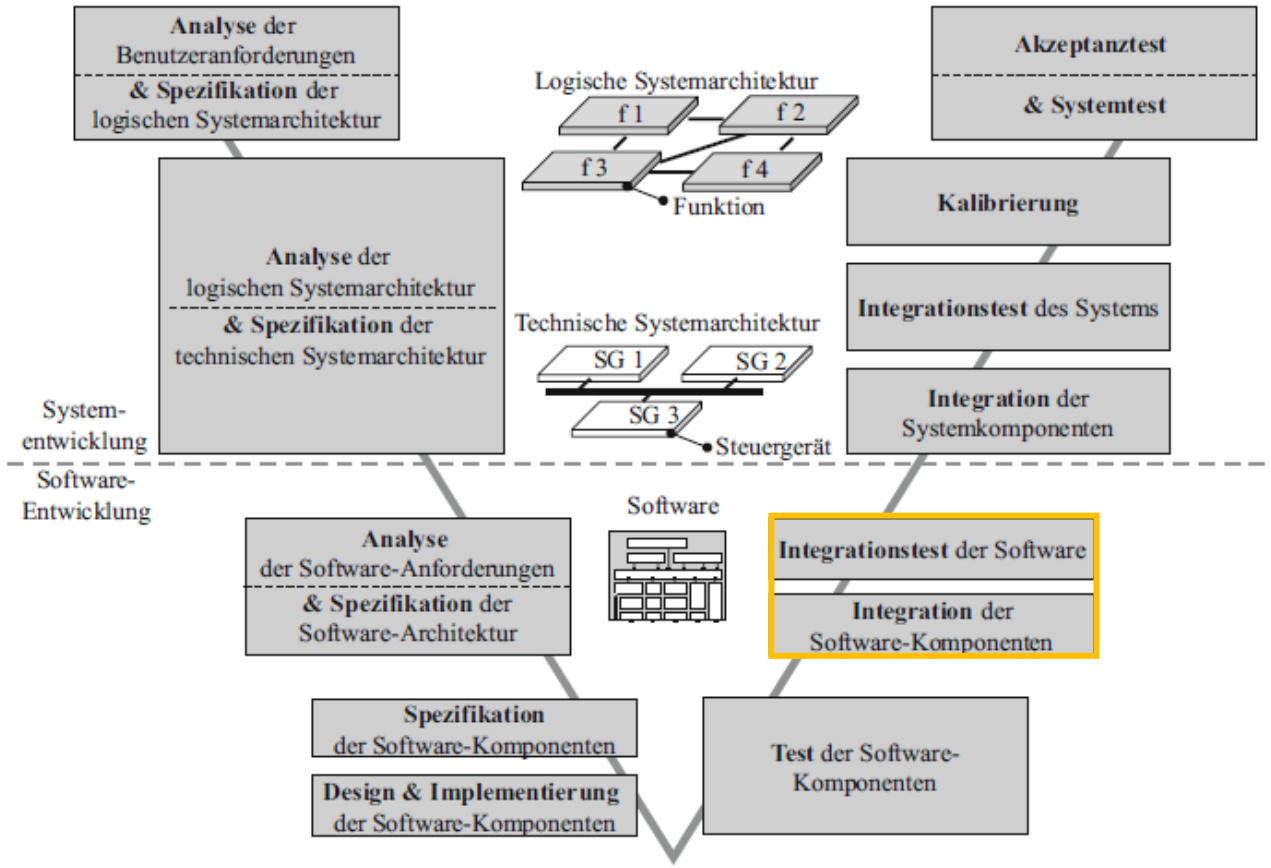


- Idealized process of V-model suggests a sequential processing of the V-model steps
- Iteration loops or necessary based on results of test cases
- Adjustments of software are necessary until the required output signals are ensured (statistics! Regarding spreads based on component tolerances)



[Sch16]

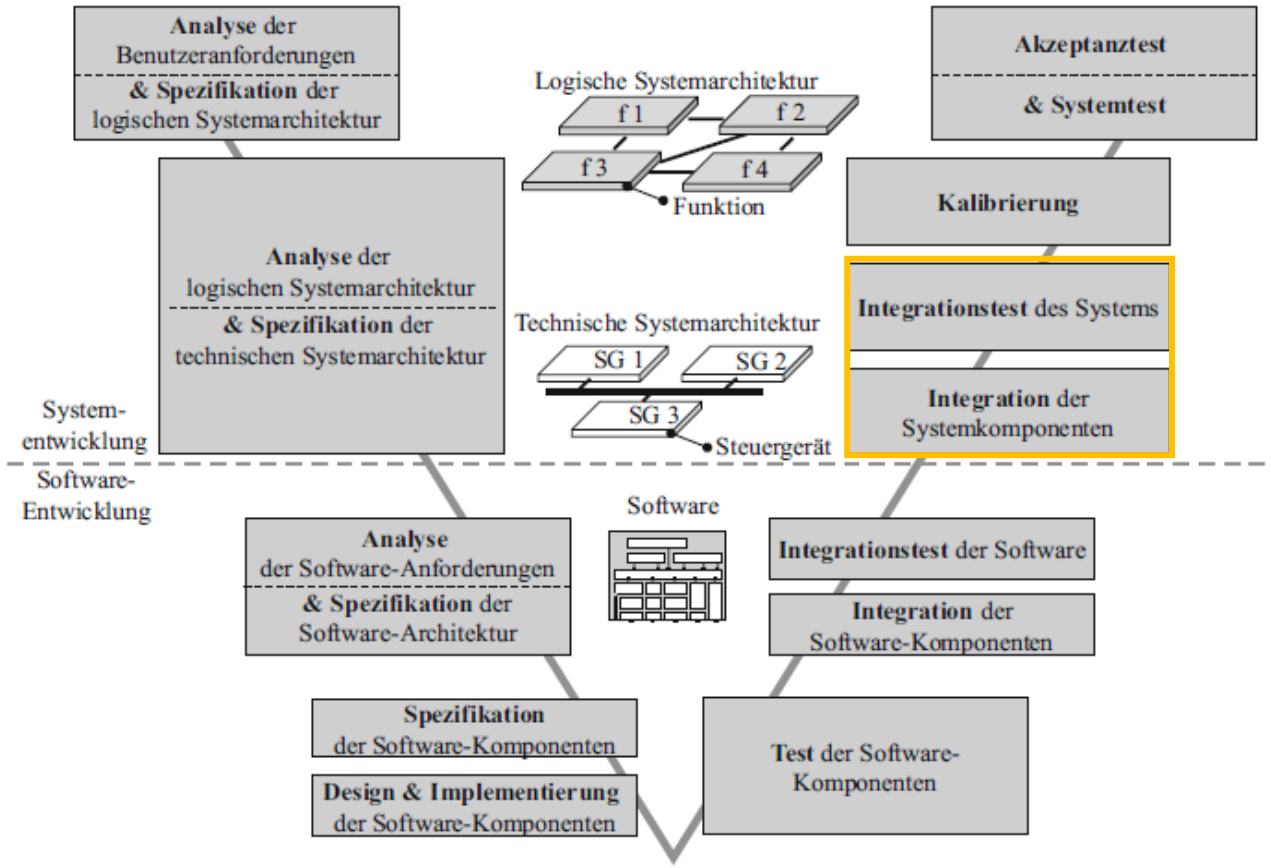
Integration & testing of the software (SIL)



- Testing of requirements regarding software architecture
- Compiling of software components to a functioning program & data set
- Software can be processed on a micro controller
- Tool based tests of a software
- Focus: Achieving software standards in automotive software engineering
- „Software in the Loop“ (SIL)

[Sch16]

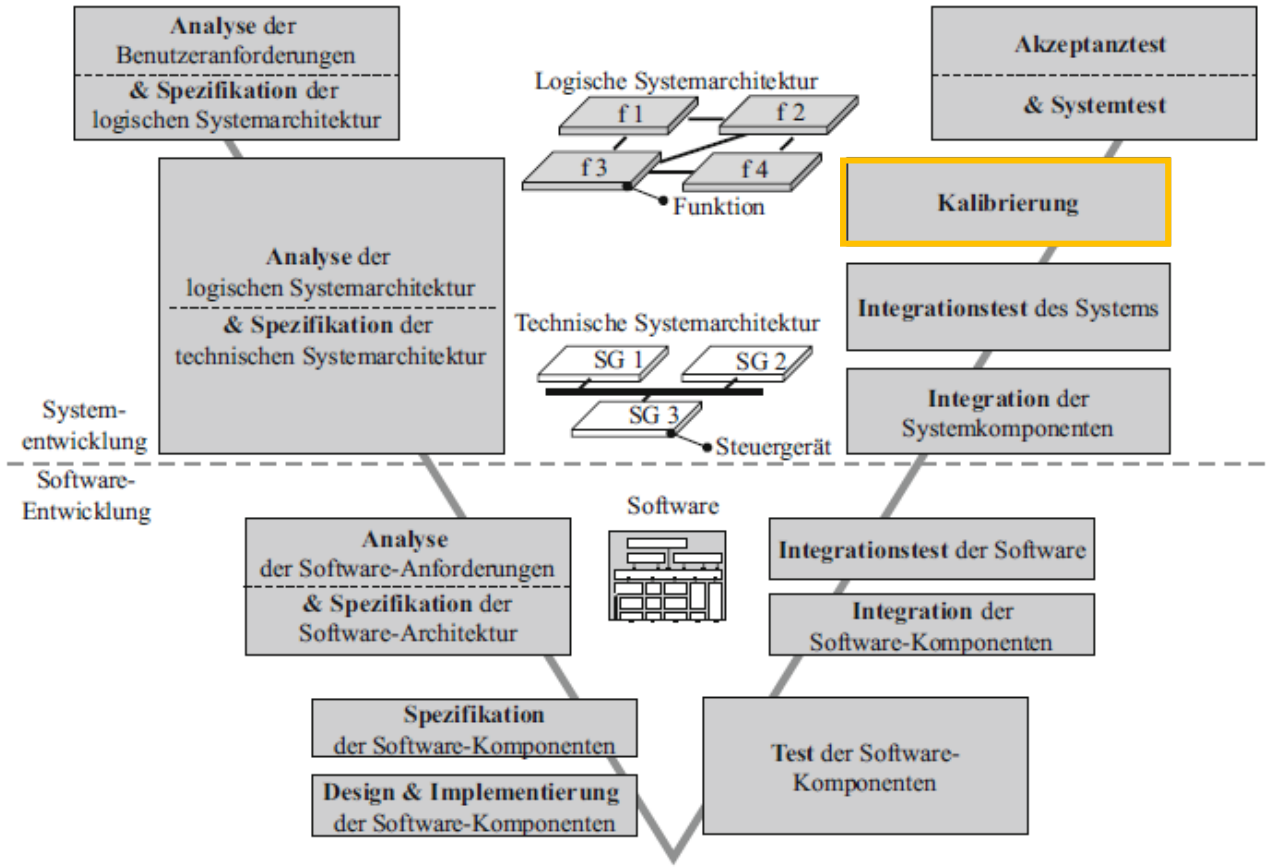
Integration & tests in system level (HIL)



- Testing of the requirements regarding logical & technical system architecture including the corelating specifications
- First testing of software on hardware control units (Hardware in the Loop HIL)
- Testing of software in combination with all relevant control units (Hardware in the Loop HIL)

[Sch16]

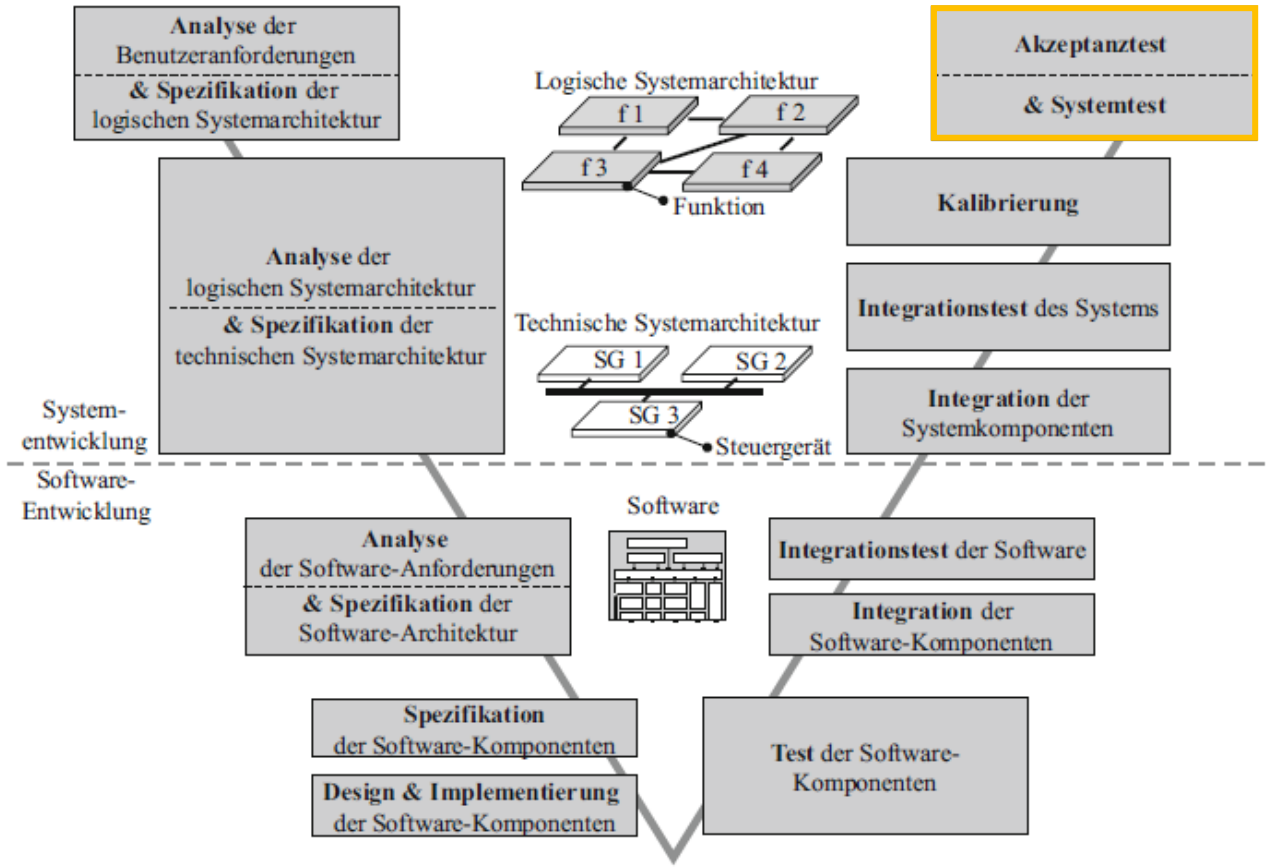
Calibration / application



- Detailed data calibration of software
- Adjustment of software on enviromental conditions
- Iterative process and data calibration

[Sch16]

Acceptance & system testing



- Testing of user requirements
- Confirmation of devolped functions
- Testing in real driving conditions

[Sch16]

Validation Levels in the V-Model

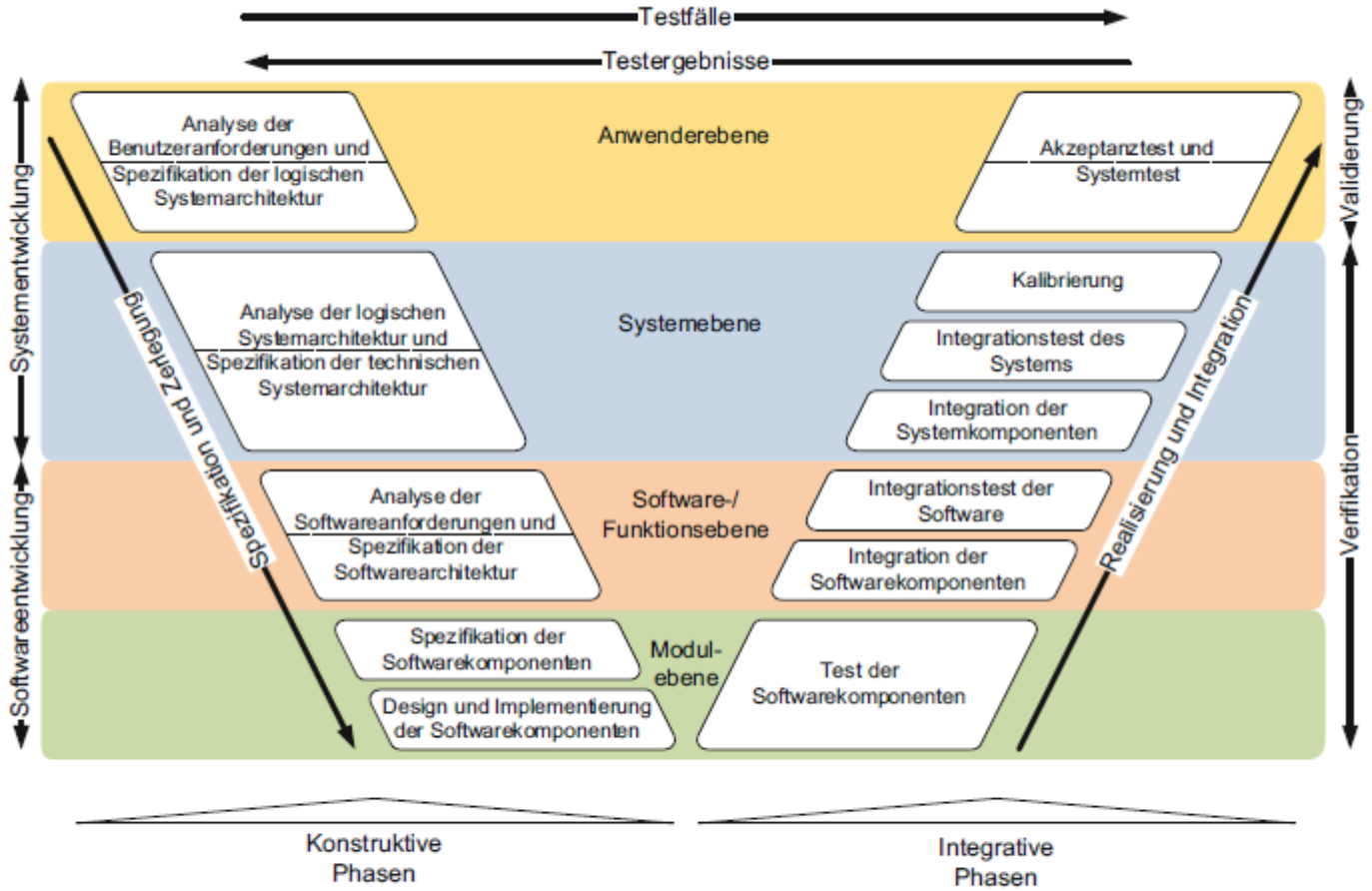


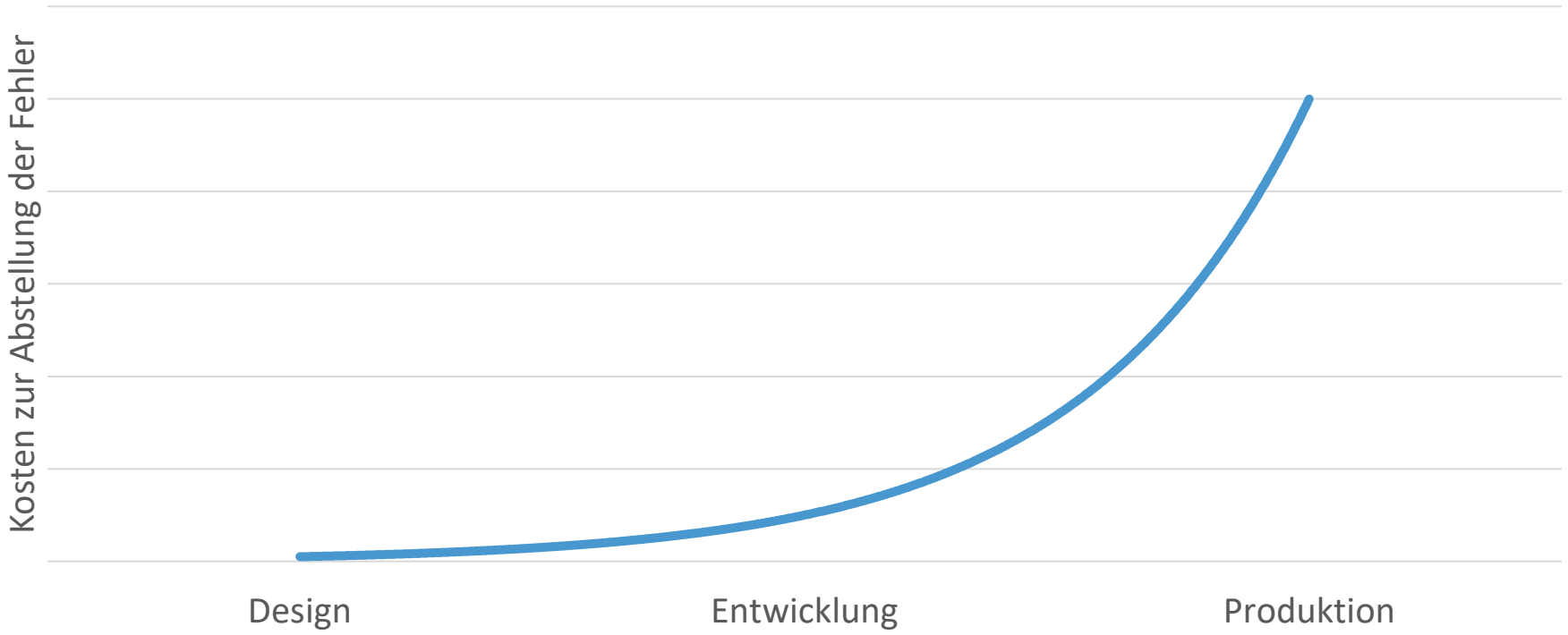
Abb. 4.8 Prüfebenen im V-Modell

[Wol18]

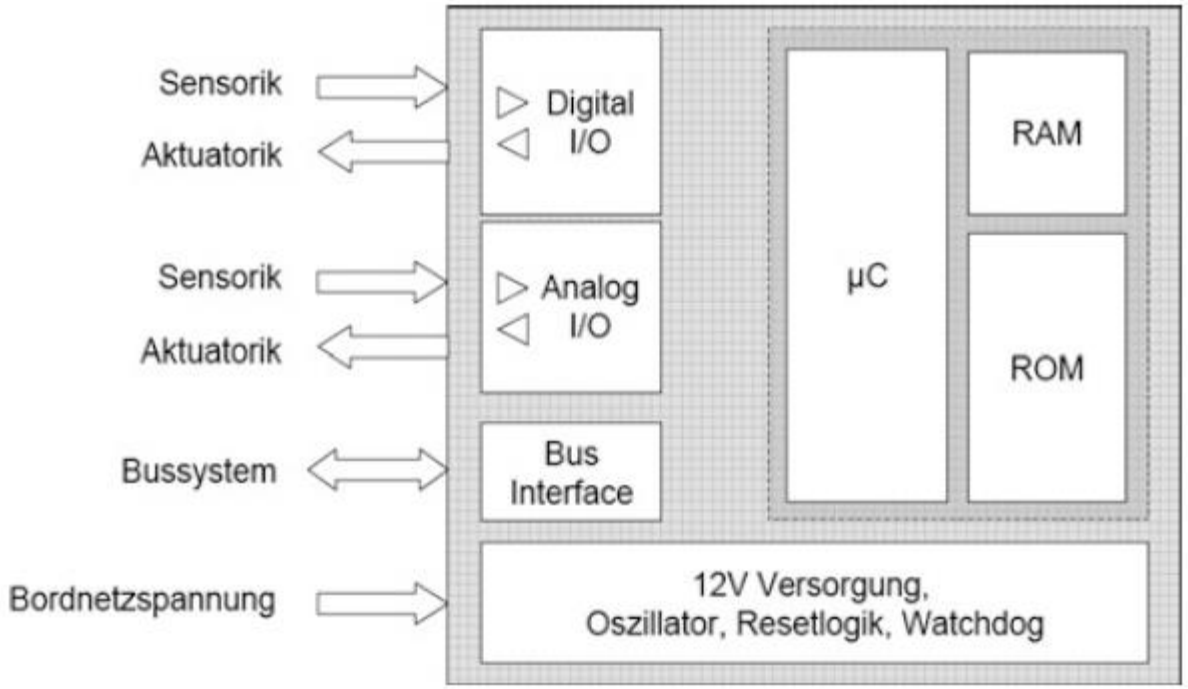
Validation & testing of software

Goal => identify failures in the software in the early phase of product development

- SIL – Software in the Loop
- HIL – Hardware in the Loop



Embedded control unit in automotive systems



Steuergerät im Kraftfahrzeug. (Form [9])

[WO18]

Domains regarding electric components

- **Hardware/Elektrik/Elektronik**
  Mikrocontroller (z. B. DSP), FPGA, Batterierelais, Schaltungen, Kabel, Stecker und weitere Verbindungstechnologien.
- **Software**
  Standardisierte Softwaremodule und -Architekturen, Basissoftware, Treibersoftware, Betriebssysteme, Kapselung, konkrete funktionale Algorithmen, offene Schnittstellen.
- **Vernetzung**
  CAN, Flexray, andere Kommunikationsprotokolle, Konnektivität.

[Wol18]

Recap module 2: Considering the mechatronic control system

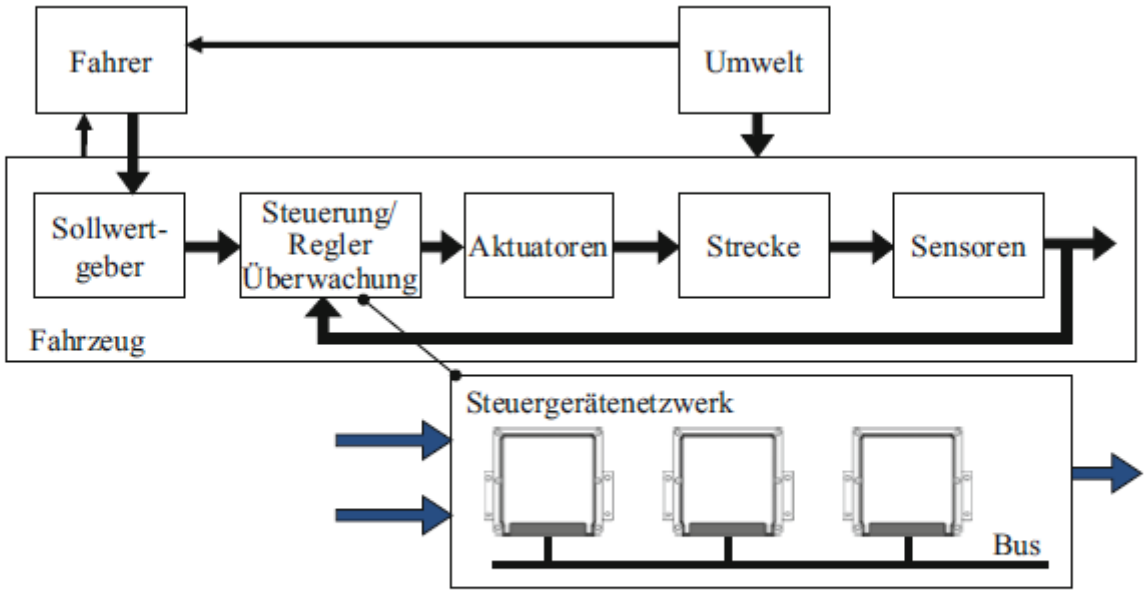


Abb. 1.4 Das Fahrzeug als mechatronisches Regelsystem. ([3] Abb. 1.2)

- Verkabelung mit Verbindungstechnik und Steckern.
- Sensoren.
- Leuchten und Displays.
- Aktoren (Elektromotoren, Ventile, Relais, . . .).
- Bussysteme (CAN, FlexRay, . . .).
- Energiespeicher (Standard-Batterie(n), Hybrid-Speicher, E-Fahrzeug, . . .).
- zukunftsorientierten Komponenten der Digitalisierung und Konnektivität.

[Wol18]

Voltage situation & communication interfaces

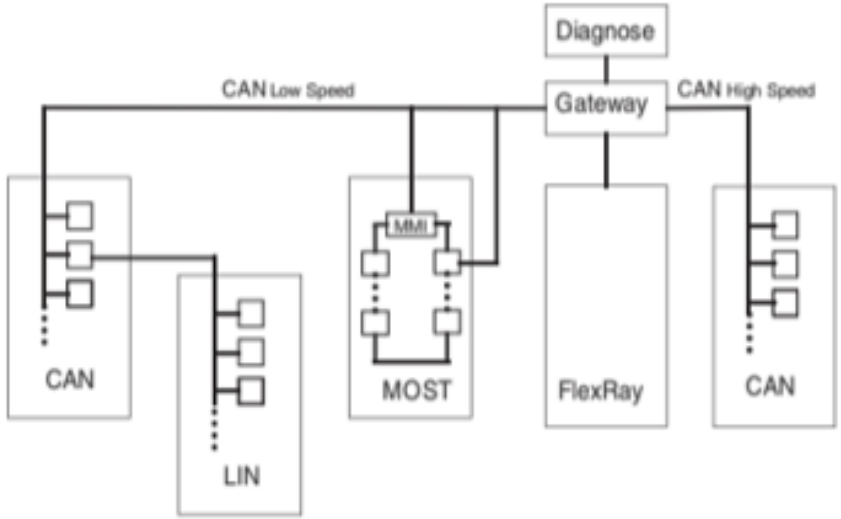
Elektrik

Klassisch:
- PKW: 12V
- LKW: 24V

Moderne PKW:
- 48V zusätzlich zum 12V Bordnetz
  Zusammenfassung Anlasser und Generator in Startergenerator (rSG, iSG) => Hybridantrieb
- Viele Bauteile haben ein integriertes Steuergerät – diese benötigen eine 12V Stromversorgung.

Kommunikation

- CAN Bus (Controller Area Network) zur digitalen Übertragung verschlüsselter Informationen
- Flexray-Bussystem: Weiterentwicklung mit höheren Anforderungen an Datensicherheit und –menge
- LIN-Subsysteme (Local Interconnect Network) zur Übertragung von kleineren Datenmengen (kostengünstig)
- MOST Bussysteme (Media Oriented System Transport) für Multi-Media Anwendungen und sehr hohe Datenmengen
- Die Verbindung der Bus-Systeme wird über Gateways sichergestellt.



[Hak17]

Recap Software engineering: mapping functions & control units

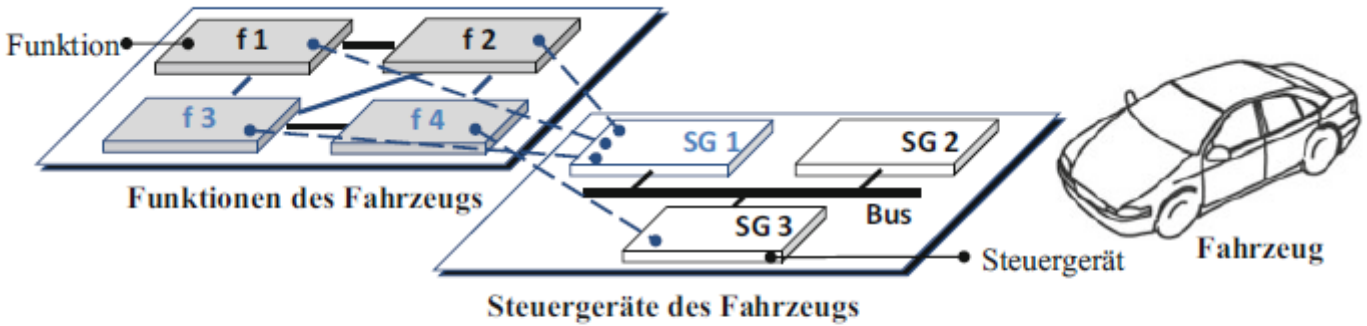


Abb. 1.6 Funktionsmapping: Zuordnung der Funktion auf die Steuergeräte. ([3] Abb. 1.12)

Software functions can be realized in vehicles trough control units

[Wol18]

Differentiate: Logical vs. technical level

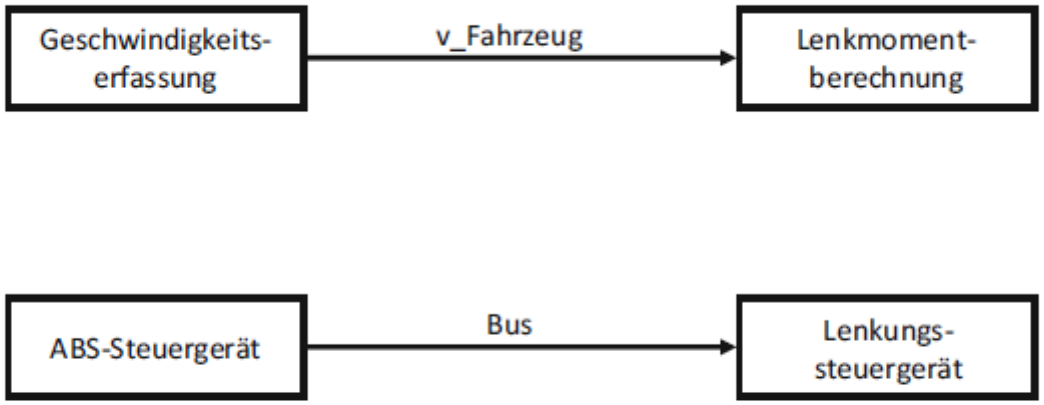


Abb. 2.1 Logische und technische Systemarchitektur

[Wol18]

Example for automotive software engineering & knowing the DIN: clamp namings in vehicles according to DIN72552

Die allgemeine Klemmenbelegung im Fahrzeug ist nach DIN 72552 genormt. Das Energiemanagement ist ein wesentlicher Faktor in der Automobilelektronik [1]. Die wesentlichen Klemmen sind:

- Klemme 15: Sie stellt Strom zur Verfügung, solange die Zündung an ist (Zündungsplus)
- Klemme 30: Sie stellt als feste Verbindung zur Batterie immer Strom zur Verfügung (Dauerplus)
- Klemme 31: Masse

[Wol18]

Example: starting a vehicle

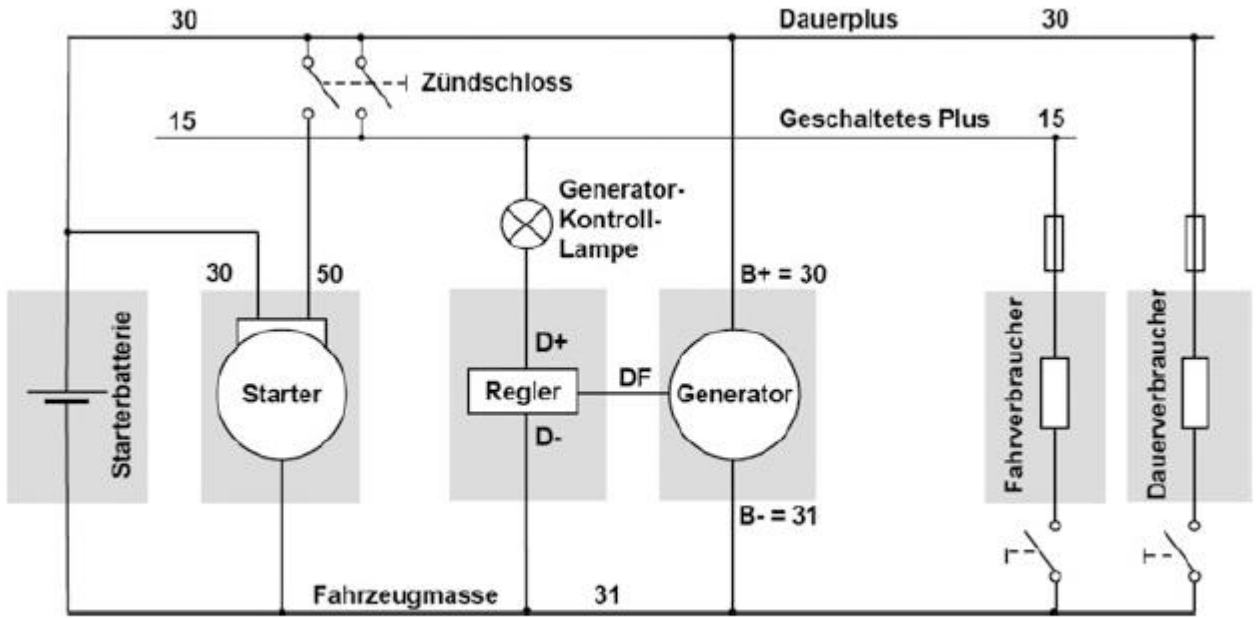


Abb. 1.5 Startfunktion und Klemmenbezeichnung

Dauerplus:
z.B. Warnblinker

Geschalteter Plus:
z.B. elektrische Fensterheber
(bei den meisten Herstellern)

[Wol18]

Example: Flash lights @ Golf V

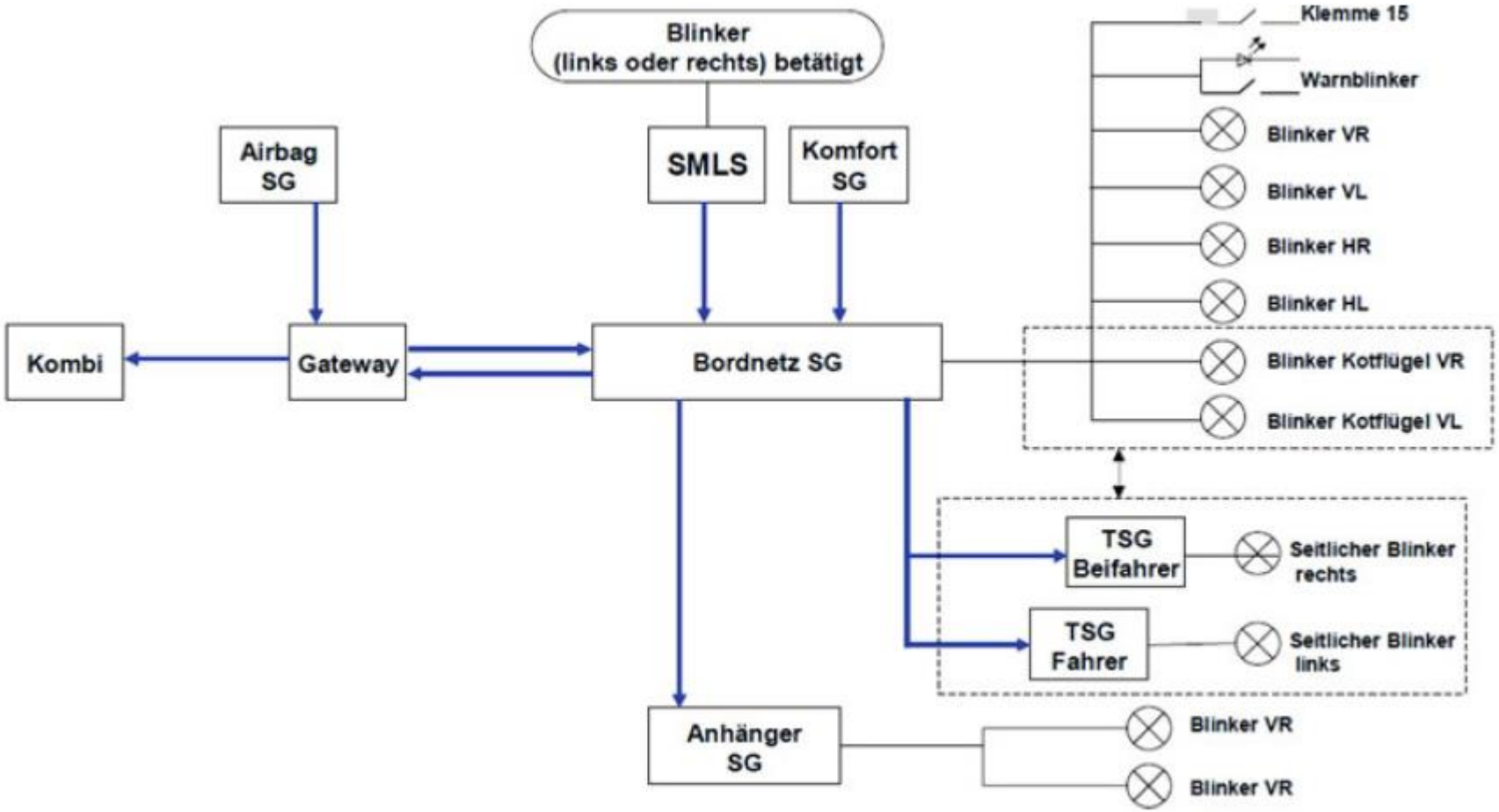


Abb. 1.3 Blinkfunktion im Golf V

[Wol18]

Example: functions & interfaces of door control unit

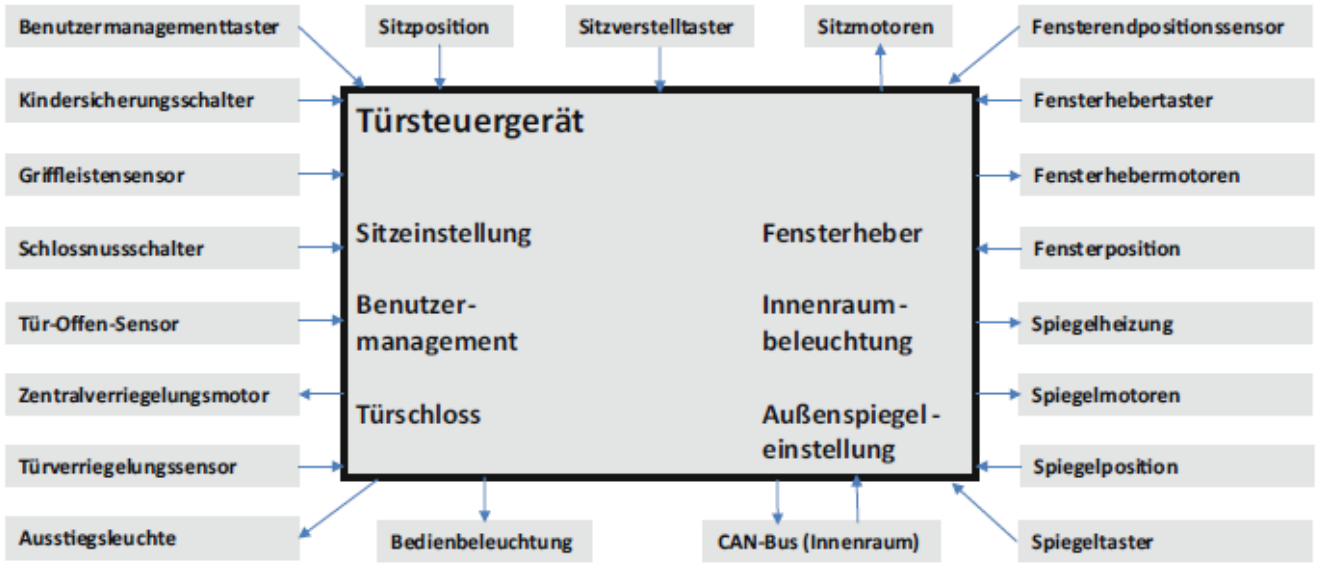


Abb. 2.11 Türsteuergerät

[Wol18]

Example: AC control unit

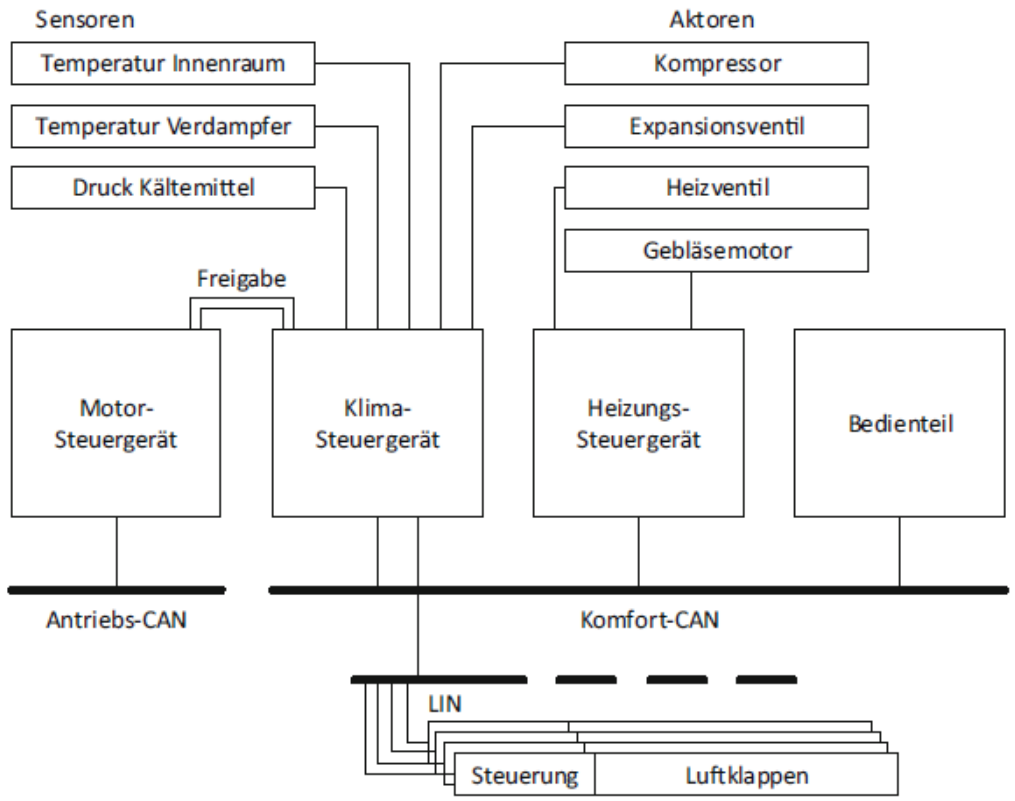


Abb. 2.8 Klimasteuerung. ([2])

Die funktionalen Anforderungen an eine Klimasteuerung in Abb. 2.8 sind je nach Modell, Leistung und Fahrzeug recht unterschiedlich in Bezug auf zu erreichende Temperaturen und Geschwindigkeiten. Hierzu gehören auch die möglichen Einsatzorte des Fahrzeugs mit ihren klimatischen Bedingungen.

Wesentliche Funktionale Anforderungen an ein Klimasteuergerät:

- Die Temperaturen für Fahrer und Beifahrer müssen getrennt einstellbar sein.
- Bei Ausfall des Klimakompressors ist ein Heiz- und Lüftungsbetrieb durchzuführen.
- Ein Beschlagen der Frontscheibe muss verhindert werden können.

Beispiele für nichtfunktionale Anforderungen an ein Klimasteuergerät können die Folgenden sein:

- Die Klimasteuerung darf die Echtzeitfähigkeit des Motorsteuergeräts nicht beeinflussen.

- Es ist eine Erhöhung des Motormoments beim Einschalten des Klimakompressors nötig.
- Die Klimasteuerung darf im Fehlerfall keine Rückwirkung auf das Bordnetz haben.

[Wol18]

Example: consequences of software bugs

- **Kampfflugzeug**
  Im Jahr 1984 verursachte der Autopilot das Drehen des Flugzeuges mit der Unterseite nach oben, wenn der Äquator überflogen wurde. Dies kam daher, dass man keine „negativen“ Breitengrade als Eingabedaten bedacht hatte. Dieser Fehler wurde sehr spät während der Entwicklung des Flugzeugs an Hand eines Simulators entdeckt und beseitigt.
- **Raketenabsturz**
  Im Jahr 1996 stürzte der Bordcomputer einer Rakete genau 36,7 s nach dem Start ab, als er versuchte, den Wert der horizontalen Geschwindigkeit von einer 64 Bit Gleitkommadarstellung in 16 Bit signed Integer umzuwandeln: –/+ b1 b2 … b15. Die entsprechende Zahl war größer als 2^{15} = 32.768 und erzeugte einen Overflow. Eine Begrenzung des Werts war nicht vorgesehen oder bedacht, da die Software von der Vorgängerversion übernommen und darum nicht weiter geprüft wurde. Die Software war zwar erfolgreich im Vorgängersystem getestet, dieses hatte jedoch nie die Geschwindigkeit der aktuellen Version erreicht. Das Lenksystem brach zusammen und gab die Kontrolle an eine zweite, identische Einheit ab. Die Selbstzerstörung wurde ausgelöst, da zu diesem Zeitpunkt bereits die Triebwerke abzubrechen drohten.
- **Flugzeugabsturz**
  Am 14. September 1993 startete in Frankfurt ein Passagierflugzeug zu einem Linienflug. Das Wetter war schlecht, es ging ein leichter Sturm und der Himmel hing voller Gewitterwolken. Trotzdem war es gut genug für eine sichere Landung. Der Kapitän des Passagierflugzeugs setzte zur Landung an. Routinemäßig brachte er das Flugzeug auf den Boden. Als er die Geschwindigkeit drosseln wollte, funktionierten die Bremssysteme nicht. Erst nach 13 s Bodenkontakt entfalteten sie ihre komplette Bremsleistung: zu spät. Ursache: Die Plausibilisierung in der Software des Fly-by-Wire für den Aufsetzdruck war aufgrund des wetterdingt so noch nicht aufgetretenen Luftdrucks falsch. Die Software des Fly-by-Wire-Systems aller Typen der Familie des Passagierflugzeugs wurde überarbeitet und der notwendige Aufsetzdruck von 12 t auf 2 t gesenkt.
- **Automobilindustrie**
  Prominente Fehler in der Motorsteuerung waren ungewollte Beschleuniger von Fahrzeugen eines Herstellers, obwohl das Fahrpedal nicht getreten wurde. Ein weiterer Softwarefehler führte zu erhöhtem Ruhestrombedarf der Fahrzeuge und damit zur Entladung der Batterie nach wenigen Tagen Standzeit.

[Wol18]

OBD in control units

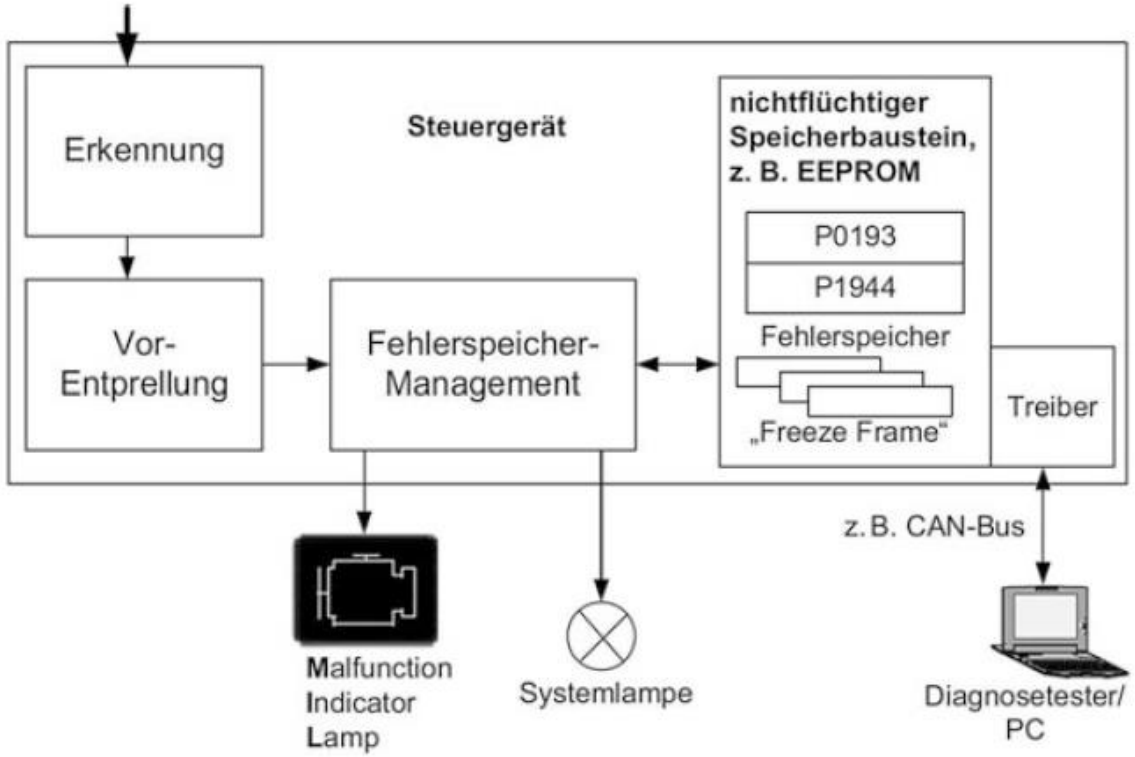
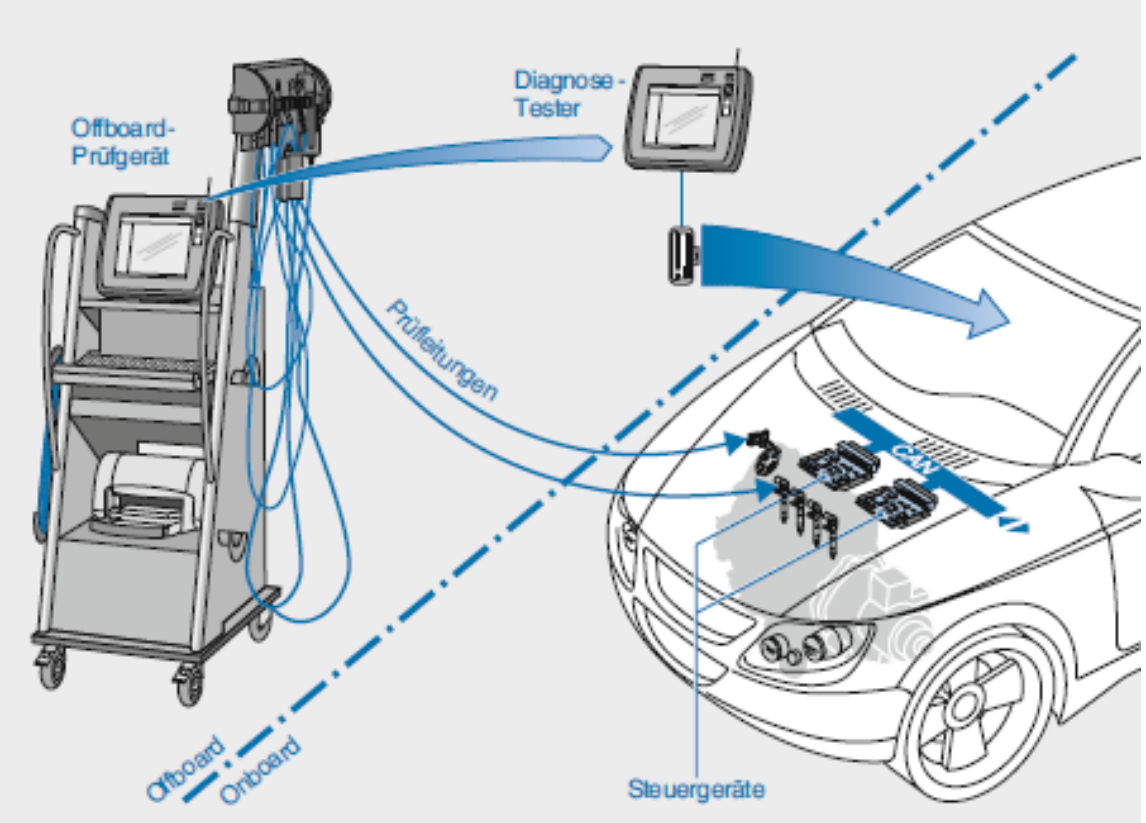


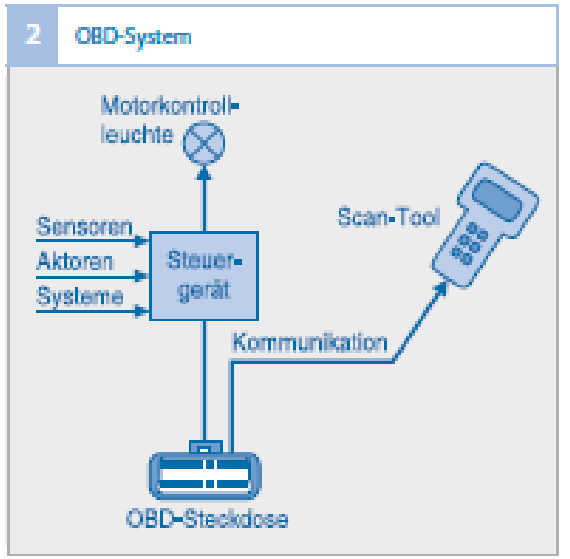
Abb. 2.7 Eigendiagnose von Steuergeräten

Beispiel:
Überwachung EGAS Funktion des elektrischen Fahrpedals (Steuergerät verarbeitet Meldung „Vollgas“ obwohl der Fahrer das Gaspedal nicht betätigt)

[Wol18]

Diagnosis system





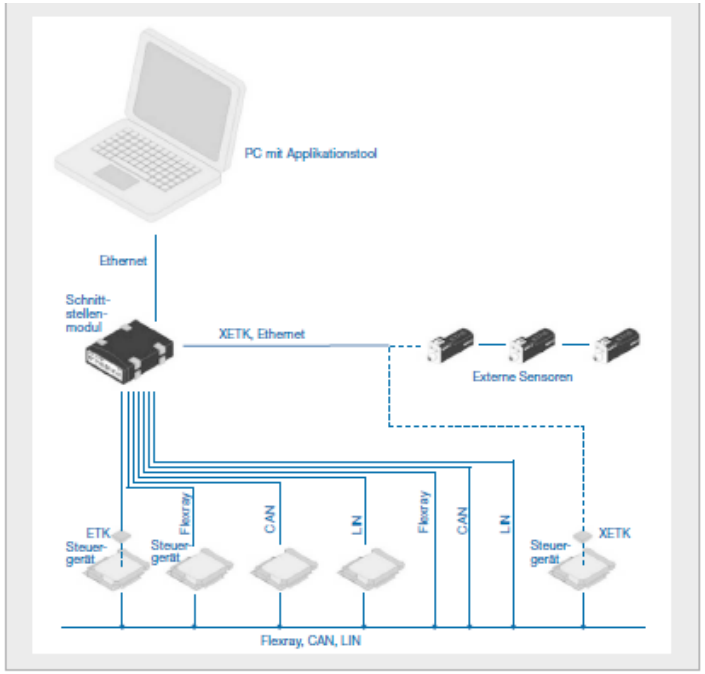
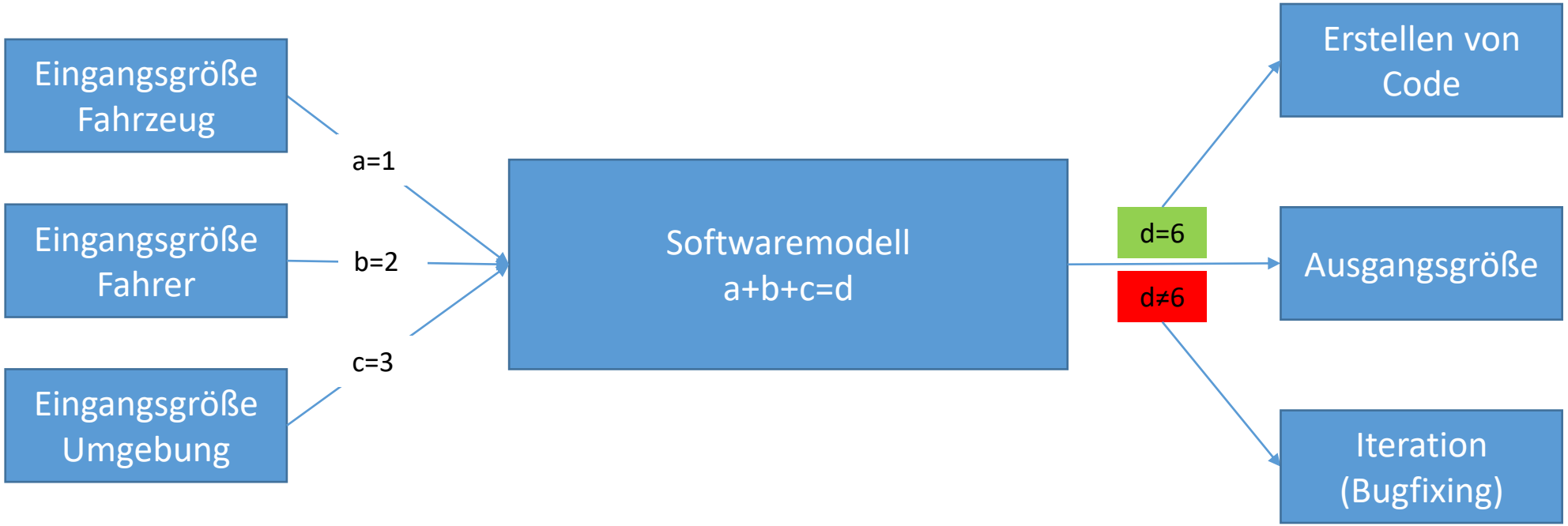


Bild 17
ETK Emulator-Tastkopf
XETK Tastkopf mit XCP-Protokoll

[Rei20]

MIL - Model in the Loop

- Komplett erstellte Software liegt noch nicht vor
- Prüfung einzelner Softwarekomponenten in einer virtuellen Umwelt-, Fahrer- und Fahrzeugumgebung
- Modell wird auf einem Computer ausgeführt

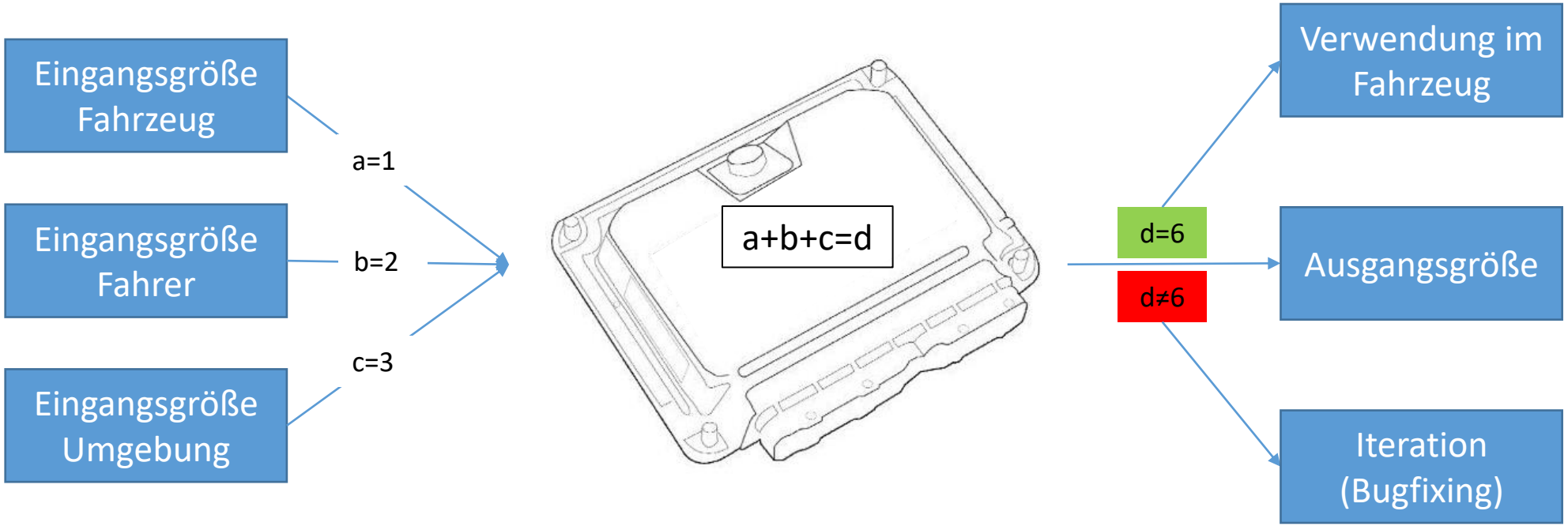


SIL - Software in the Loop

- Erstellte Software liegt vor
- Prüfung der Software in einer virtuellen Umwelt-, Fahrer- und Fahrzeugumgebung
- Software wird auf einem Computer ausgeführt

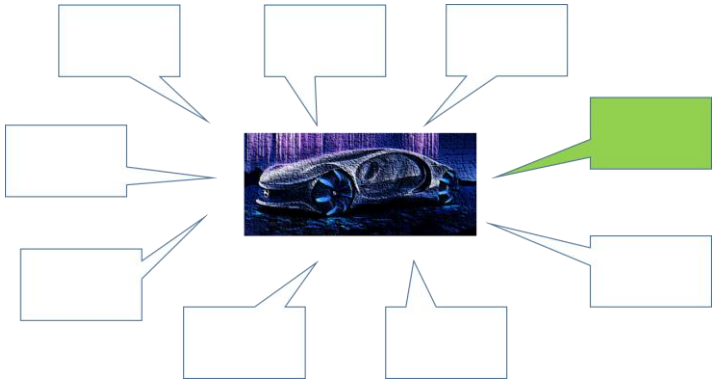
HIL - Hardware in the Loop

- Software ist auf dem Steuergerät geflasht
- Prüfung der Software in einer virtuellen Umwelt-, Fahrer- und Fahrzeugumgebung
- Software wird auf der finalen Hardware, dem Steuergerät, ausgeführt



Teamwork module 5

- Take your choosen mechatronic component you described already by sensors / voltage levels / PID controller
- Please describe test cases for your component
- Which test cases are necessary to include also extreme boundary conditions (e.g. temperature, altitude, humidity)



www.hs-kempten.de/adrive