



- | | | |
|-----|--------------------------------|--------------------|
| 1. | Einführung | - Prof. Zimmermann |
| 2. | Aspekte des Systementwurfs | - Prof. Zimmermann |
| 3. | Modellbasierter Entwurf | - Prof. Zimmermann |
| 4. | Echtzeitsysteme | - Prof. Zimmermann |
| 5. | Scheduling | - Prof. Zimmermann |
| 6. | Sicherheit und Zuverlässigkeit | - Prof. Zimmermann |
| 7. | Softwaretechnische Aspekte | - Prof. Fengler |
| 8. | Hardware-Software-Codesign | - Prof. Fengler |
| 9. | Rechnerarchitektur Aspekte | - Prof. Fengler |
| 10. | Kommunikation | - Prof. Fengler |
| 11. | Energieeffizienz | - Prof. Fengler |
| 12. | Domäne Automotive | - Prof. Fengler |



1. Vorgehensmodelle

2. Echtzeit Programmierung

3. Testen

1. Allgemeines

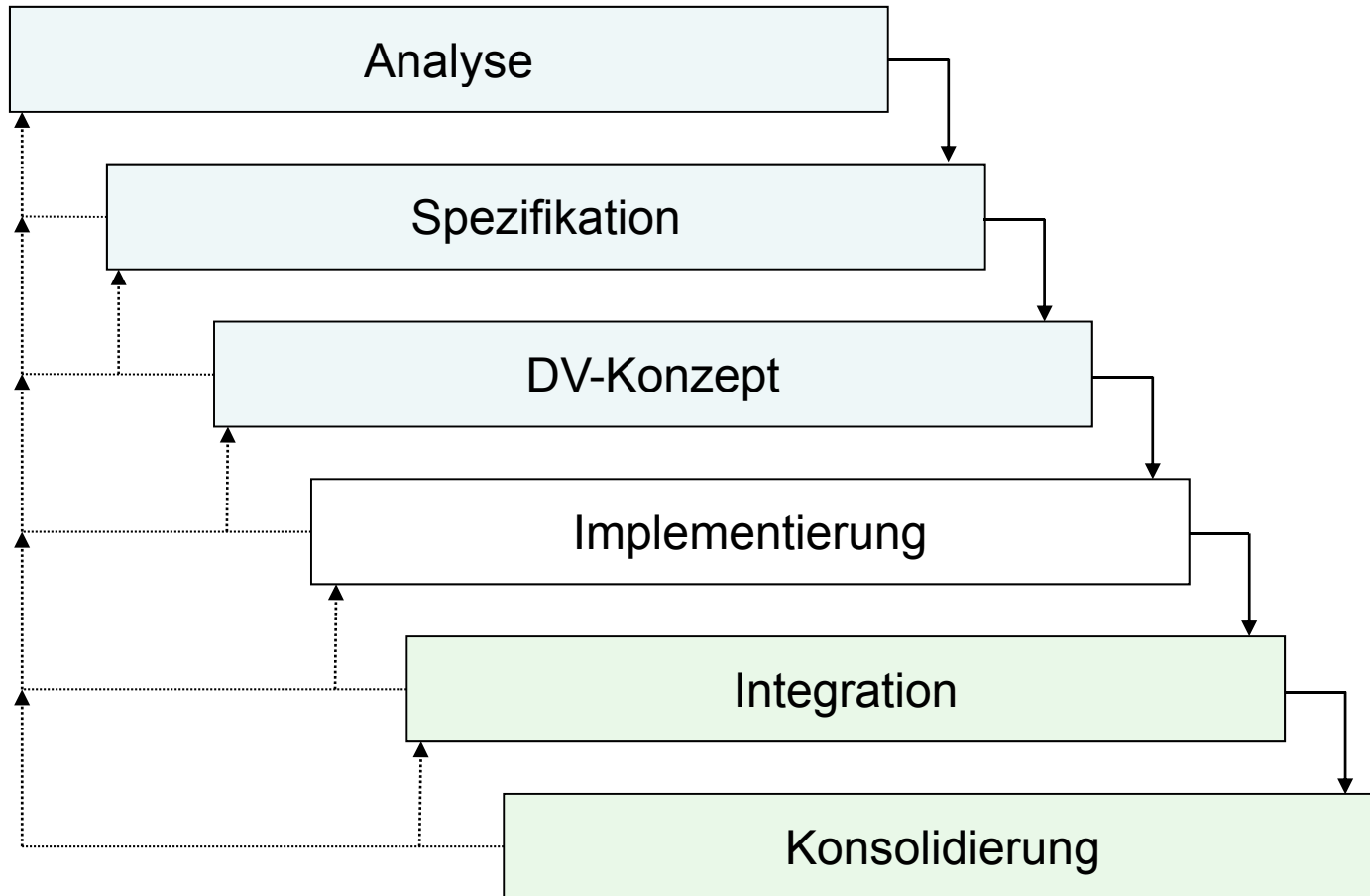
2. Remote Debugging

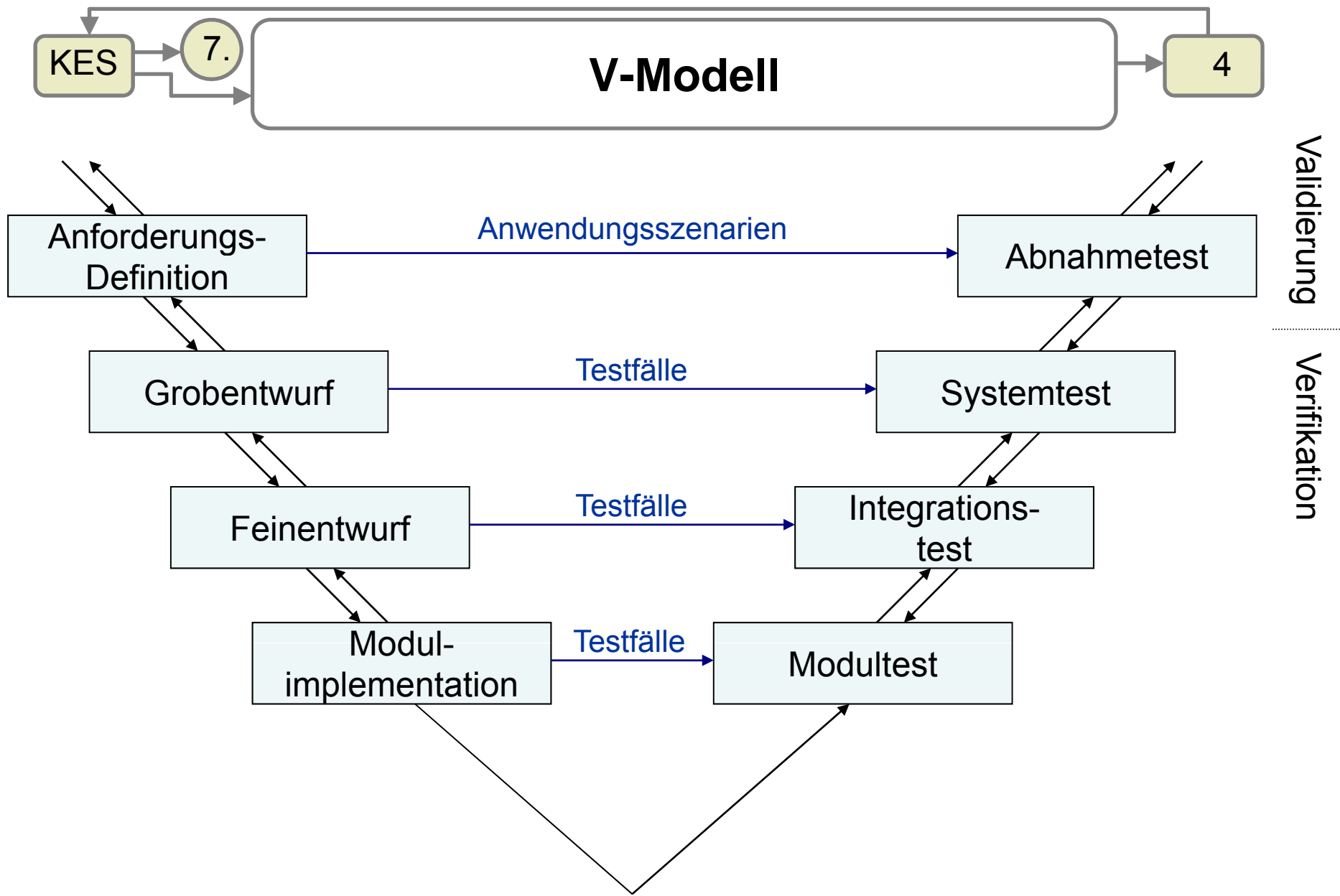
3. In Circuit Emulation

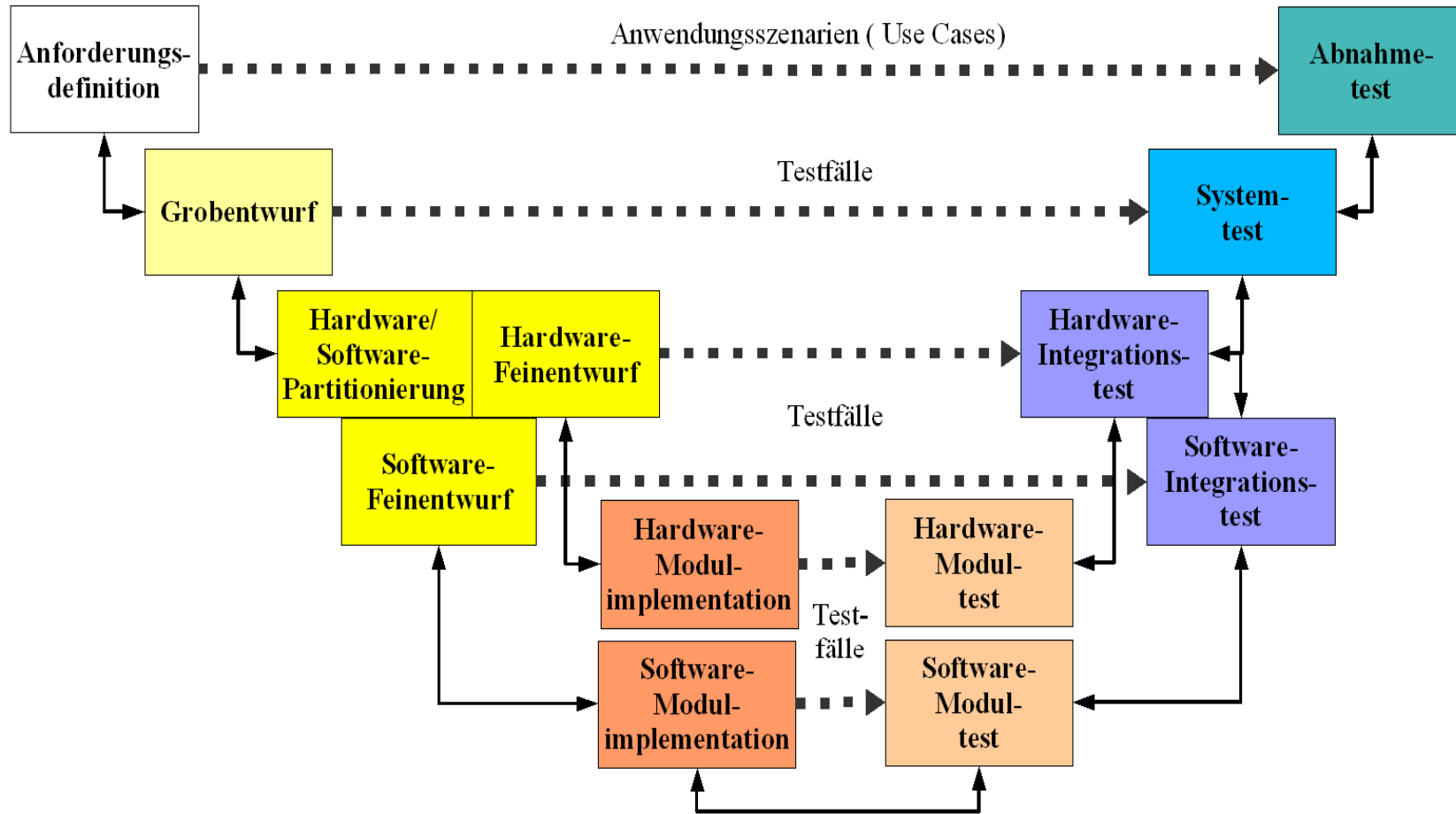
4. Hardware in the loop / Software in the loop

5. JTAG

4. Autosar

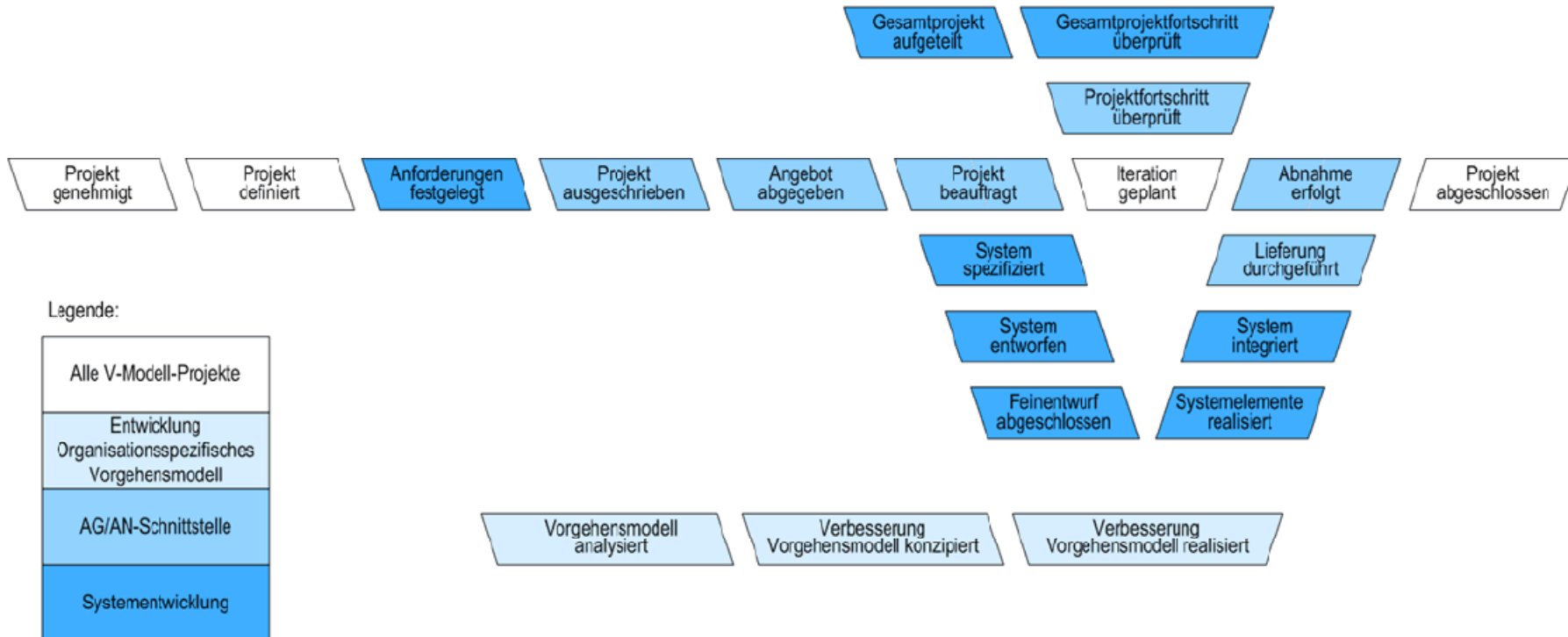


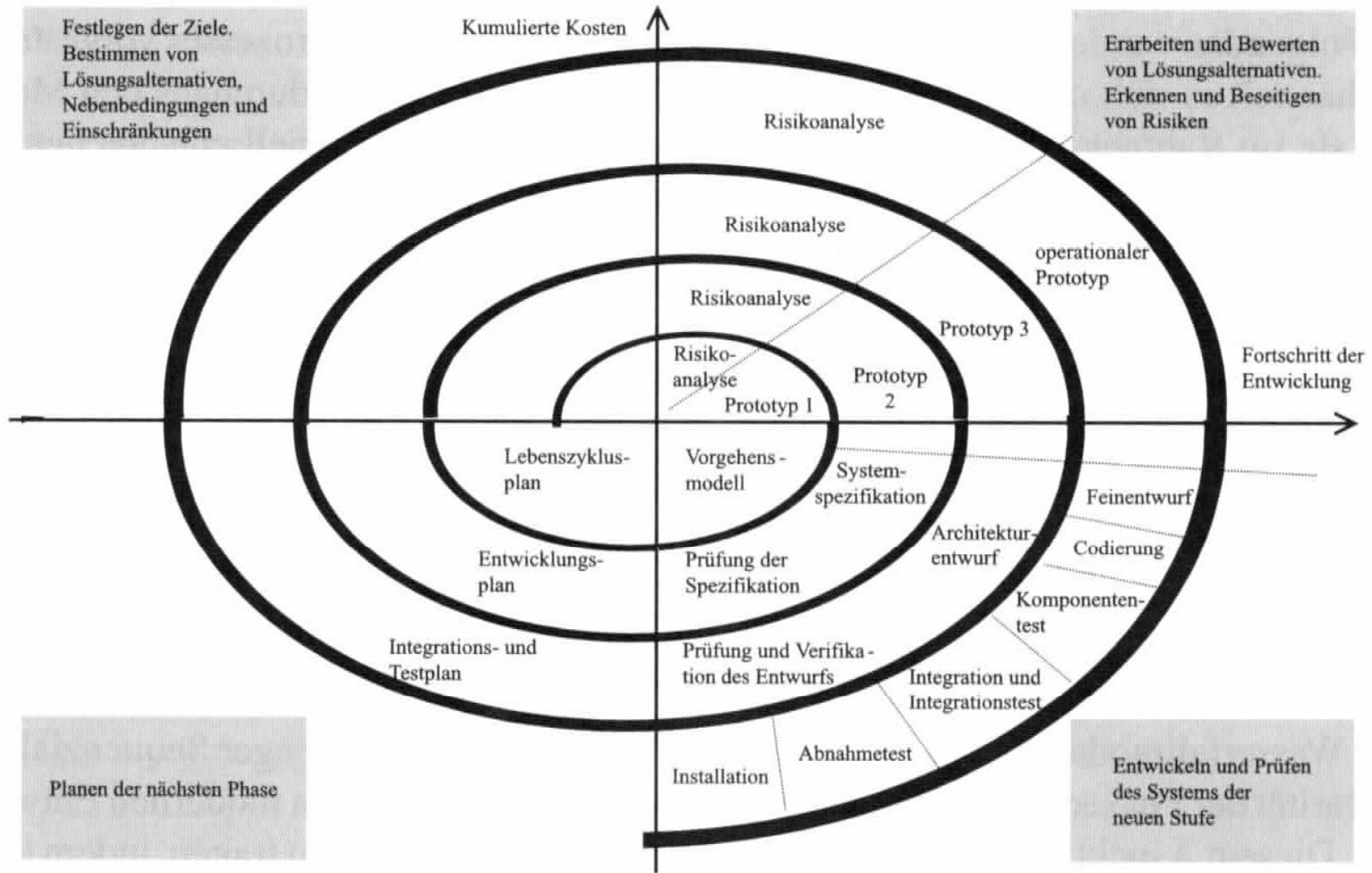
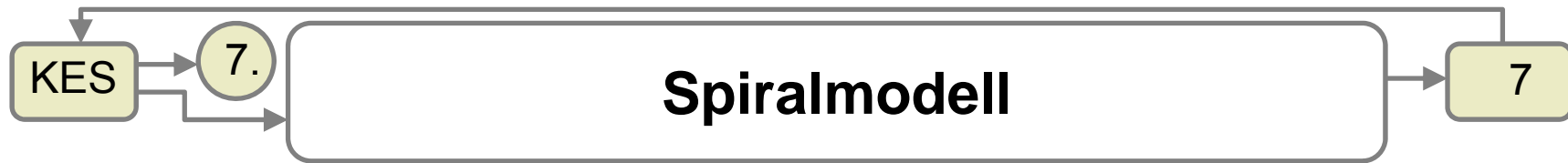




Quelle: Müller, M.; Implementation eines UML Entwurfs; Studienarbeit; TU Ilmenau, 2005

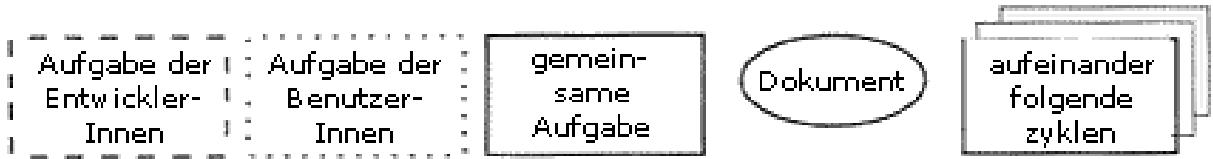
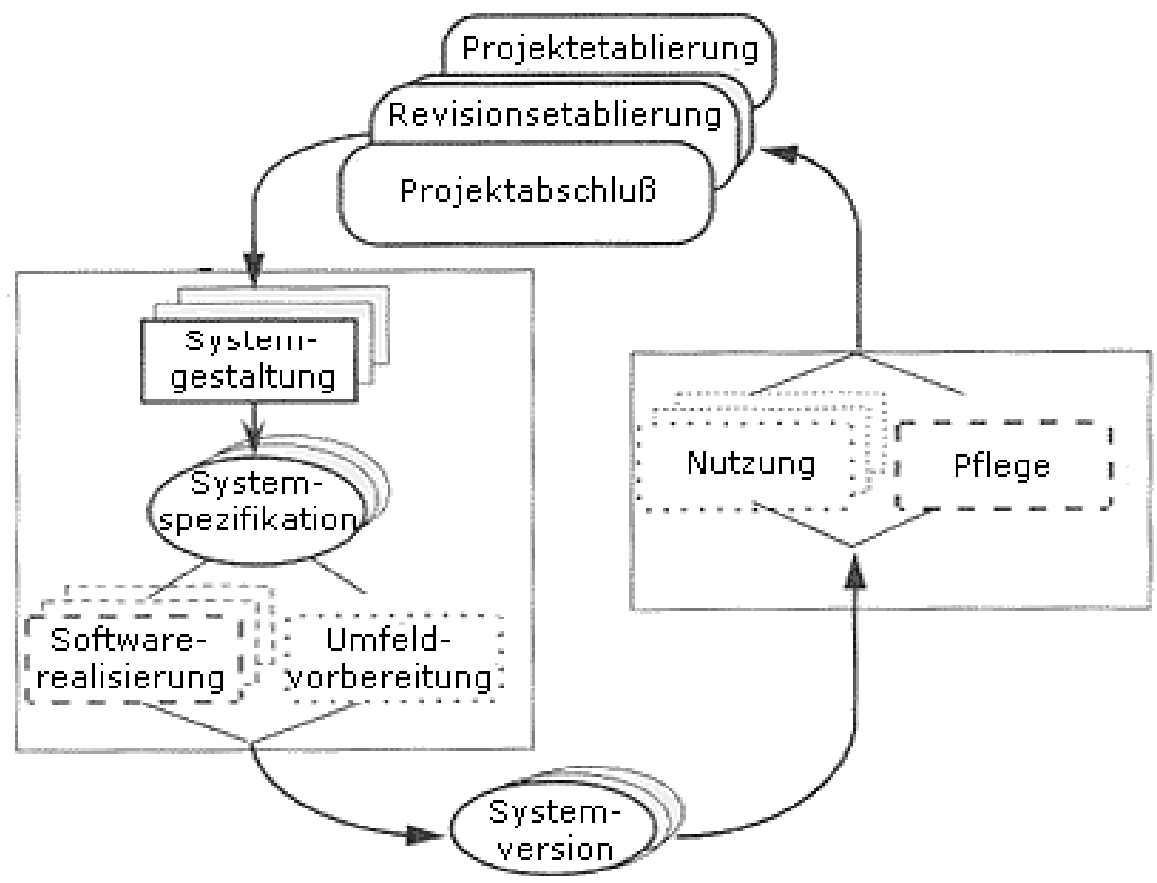
Fengler, Zimmermann 03-2009





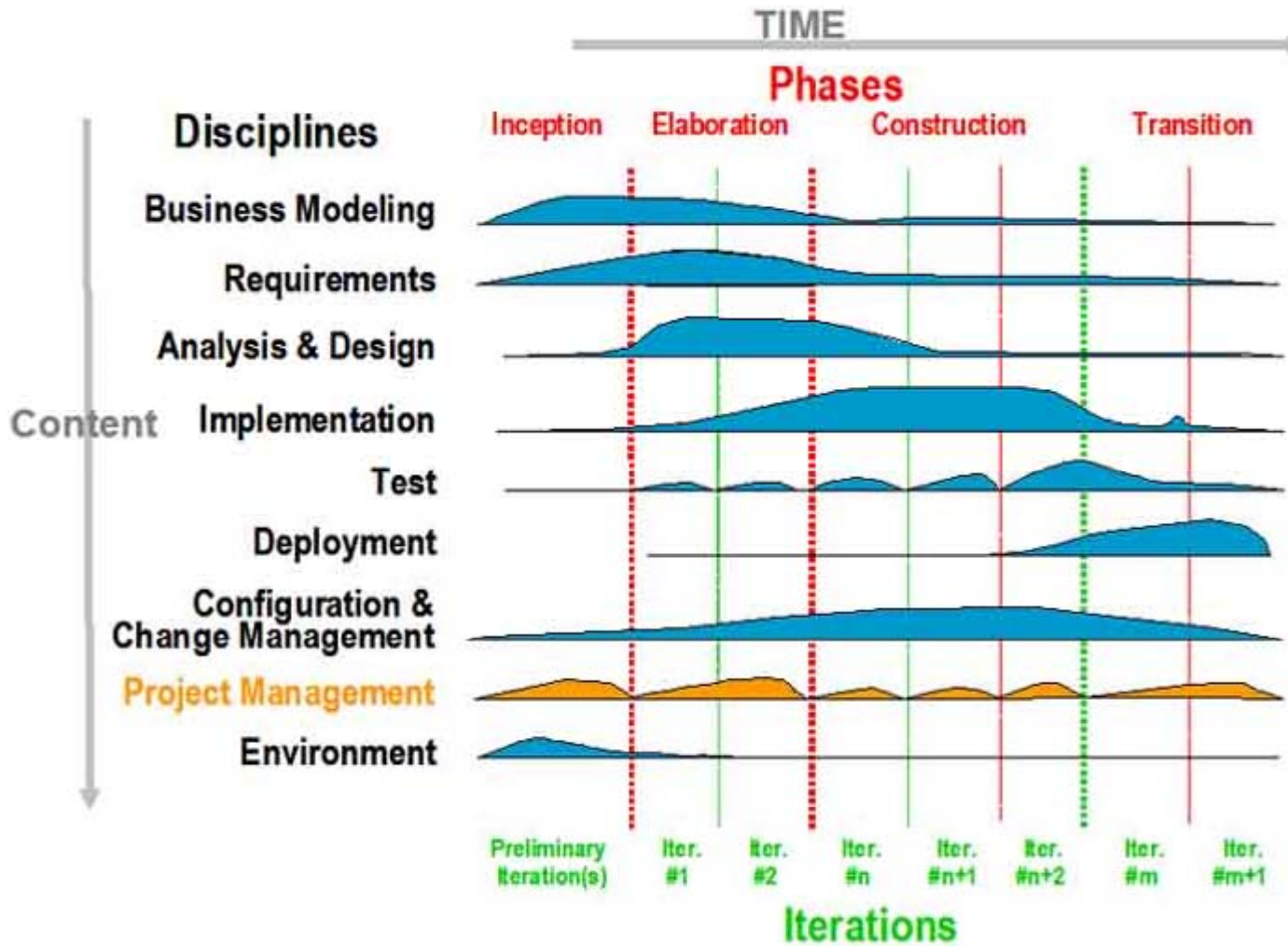
Quelle: A. Silkora, R. Drescher: Software-Engineering und Hardware-Design, Hanser, 2002

Fengler, Zimmermann 03-2009



Quelle: C. Floyd u.a., in: Informatik-Spektrum Heft 1, Feb. 1997.

Fengler, Zimmermann 03-2009



Quelle: IBM



1. Vorgehensmodelle

2. Echtzeit Programmierung

3. Testen

1. Allgemeines

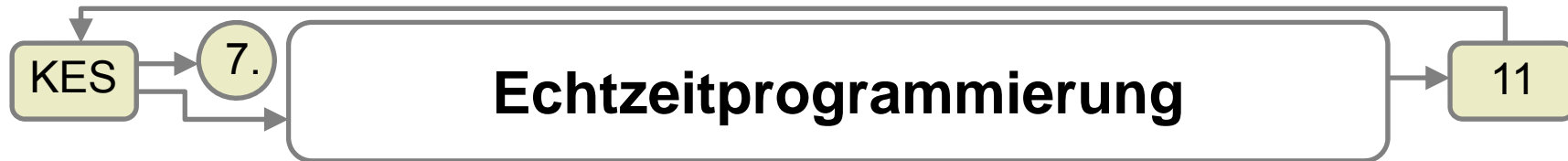
2. Remote Debugging

3. In Circuit Emulation

4. Hardware in the loop / Software in the loop

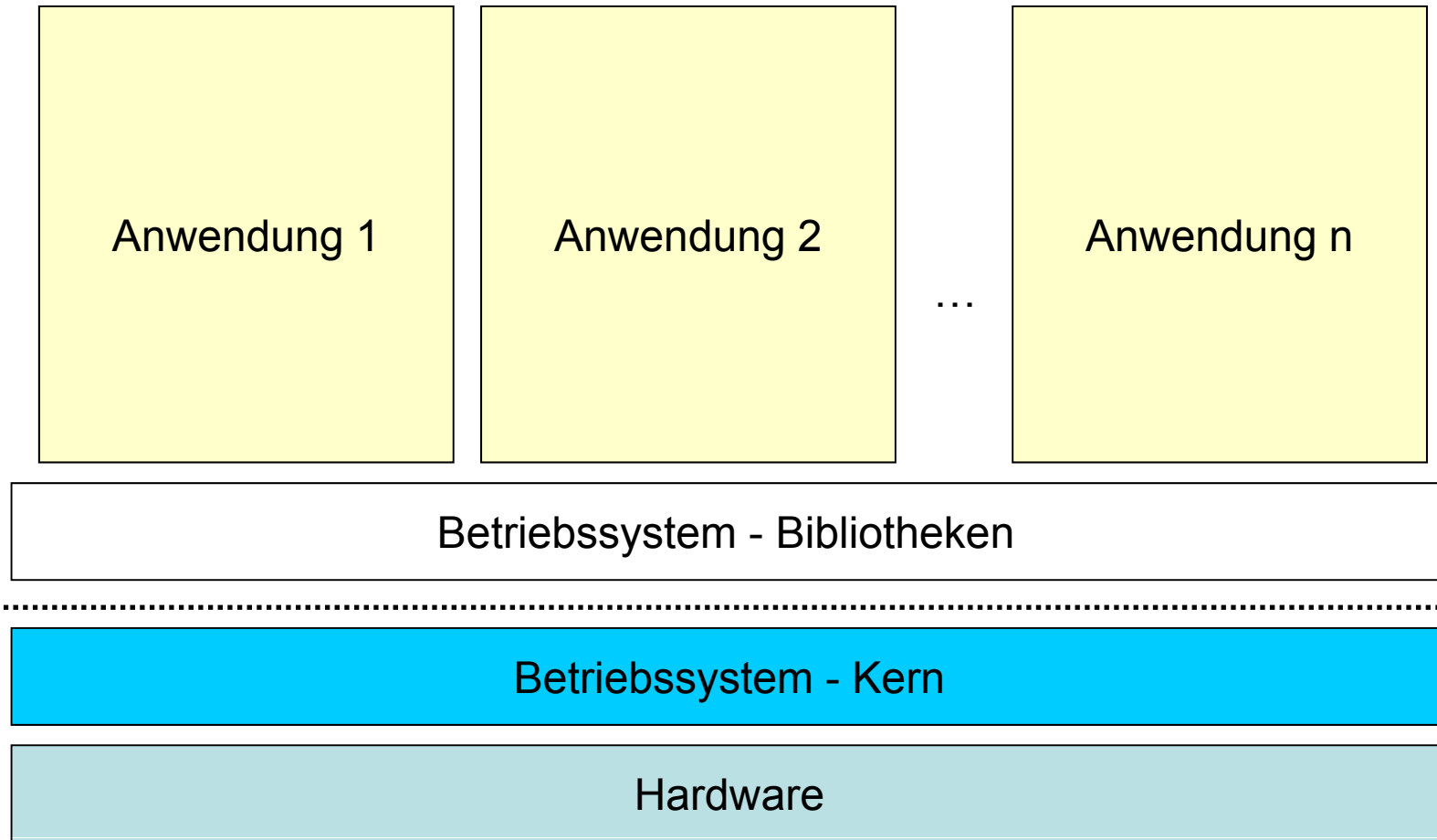
5. JTAG

4. Autosar



„Echtzeitbetrieb ist ein **Betrieb eines Rechnersystems**, bei dem Programme zur Verarbeitung anfallender Daten **ständig** derart betriebsbereit sind, dass die Verarbeitungsergebnisse **innerhalb einer vorgegebenen Zeitspanne** verfügbar sind.“ (DIN 44300)

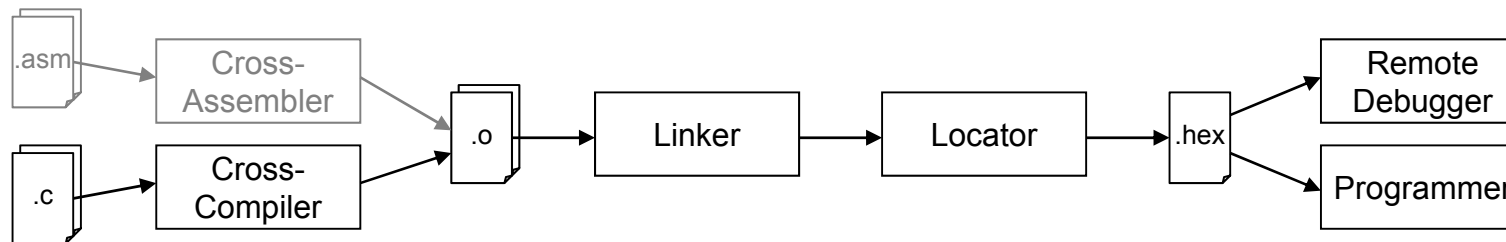
→ siehe 1. Teil der Vorlesung





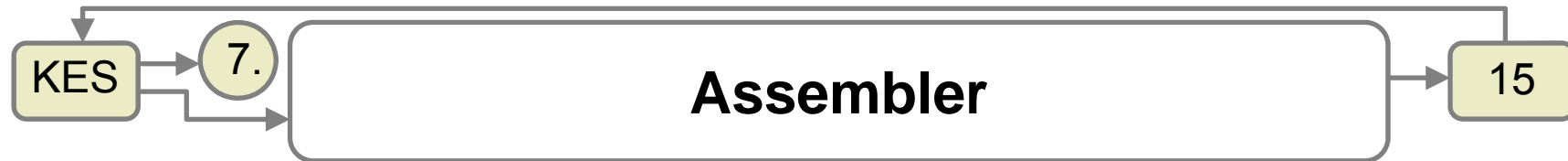
- Übersetzungsvorgang

- Cross-Compiler übersetzt Quelltext in Binärcode für die Zielarchitektur
- Locator passt Speicheradressen für Programm- und Datenbereiche entsprechend der vorhanden ROM / RAM Adressen an





```
void led_out(char ziffer, char dot, int stelle) {  
  /* Ziffer: 0-15 (A-F), '-' , '<Space>'  
  Dot: >0:  
  Stelle: 0-7 */  
  
  /* 7 - Segment Repräsentation der Ziffern 0-9, a-f, -) */  
  unsigned char ztab[18] = {0x3F,0x06,0x5b,0x4f,0x66,0x6d,0x7d,  
    0x07,0x7F,0x6F,0x77,0x7c,0x39,0x5e,0x79,0x71,0x40,0};  
  /* Adressen der 7-Segment Anzeigen */  
  unsigned char digit[8]={0x20, 0x10, 0x40, 0x30, 0x60, 0x50,  
    0x80, 0x70};  
  unsigned char out;  
  if (ziffer=='-') ziffer=16;  
  if (ziffer==' ') ziffer=17;  
  if (ziffer>17) ziffer=17;  
  out = ztab[ziffer];  
  if (dot) out |= 0x80; // Punkt einschalten, falls erforderlich  
  DP1L = 0xFF; // Richtung Datenbusport: Ausgabe  
  P1H = digit[stelle]; // Adresse auf Adressbus ausgeben  
  P1L = out; // Wert auf Datenbus ausgeben  
  P1H = 0; // Adressbus zurücksetzen  
}
```



```

MOV DX, 0B0H ; 1. Ausgabeadresse in DX
MOV EDI,10
MOV ESI,OFFSET ziff ;Adresse von ziff ins Register bringen
m2: MOV AL,[ESI+EDI-1]
    OUT DX,AL
    INC DX ; nächste Displaystelle
    DEC EDI ; nächste Ziffer
    JNZ m2 ; Solange wie EDI nicht 0 zurück zu m2

ziff DB 3FH,03H,6DH,67H,53H,76H,7EH,23H,7FH,77H
  
```

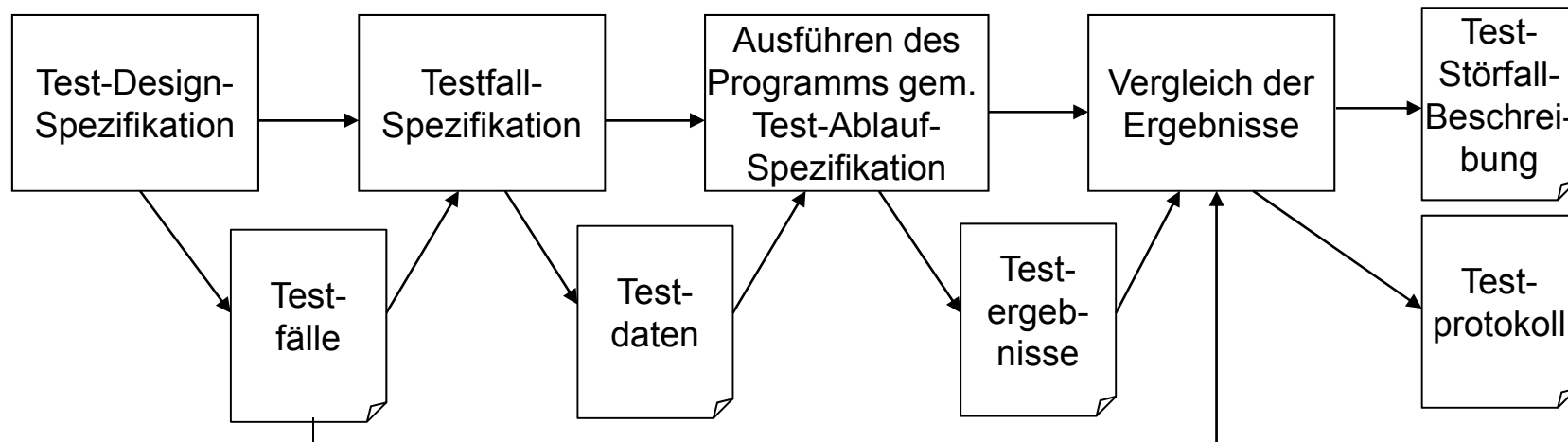


1. Vorgehensmodelle
2. Echtzeit Programmierung
3. Testen

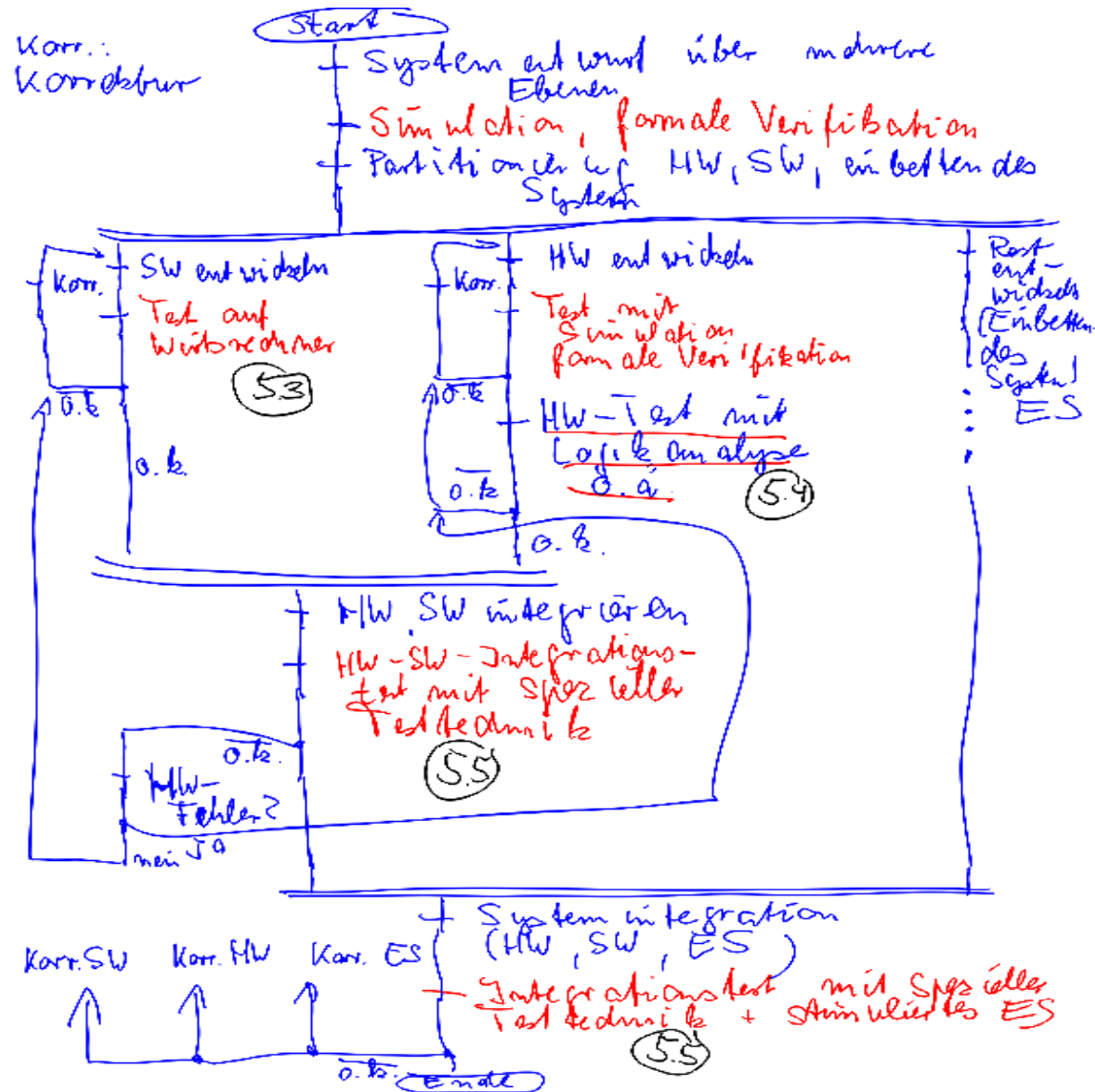
1. Allgemeines

2. Remote Debugging
3. In Circuit Emulation
4. Hardware in the loop / Software in the loop
5. JTAG

4. Autosar

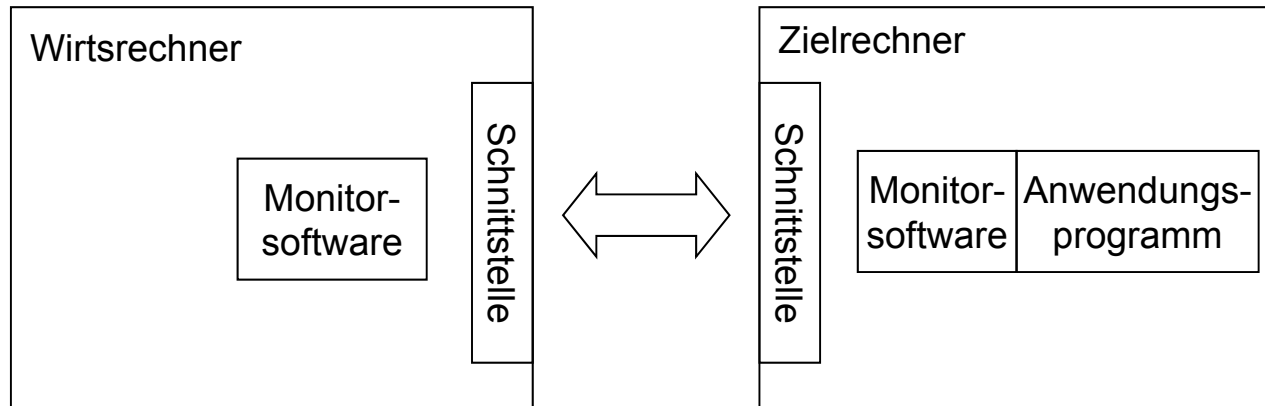


Einordnung von Inbetriebnahme und Test in den Entwicklungsablauf



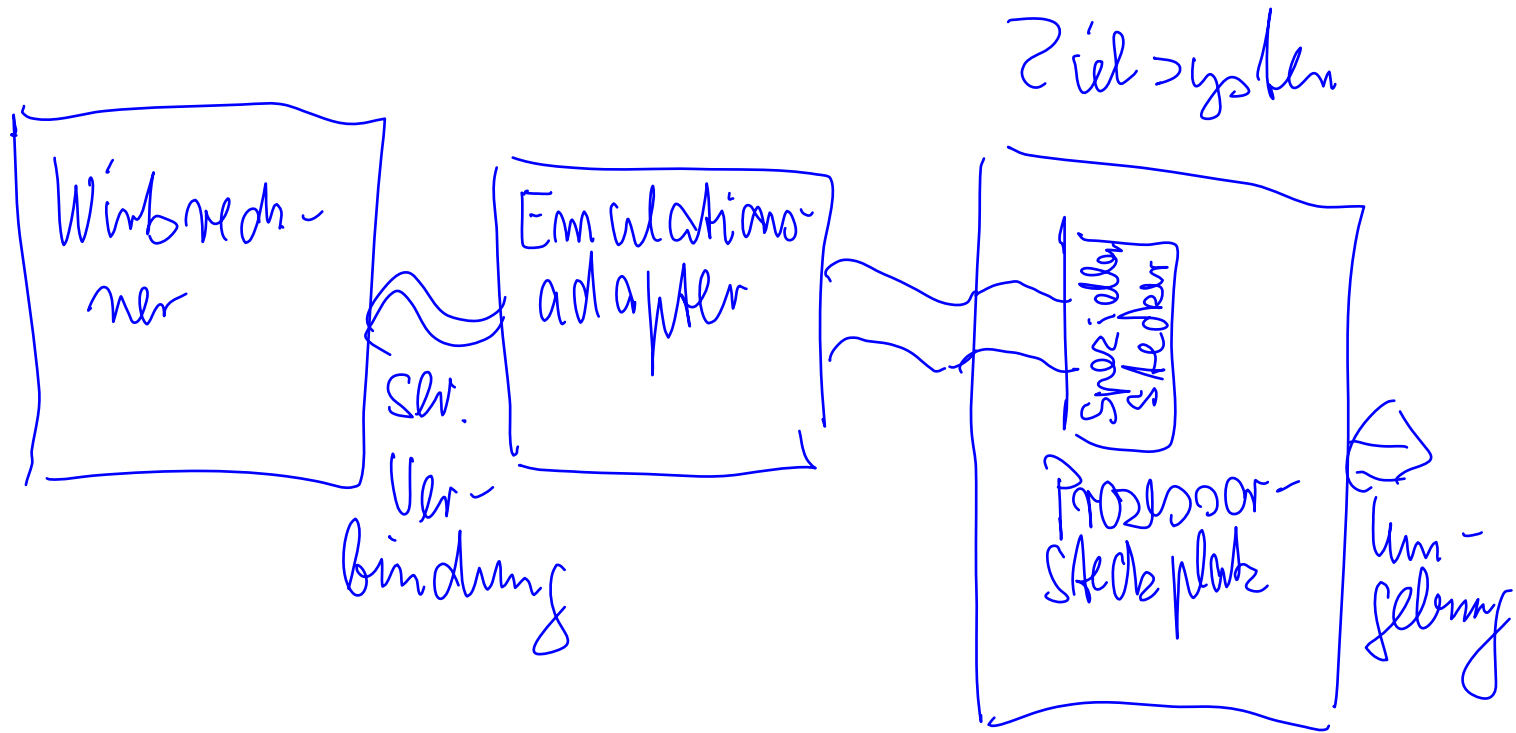


1. Vorgehensmodelle
2. Echtzeit Programmierung
3. Testen
 1. Allgemeines
 2. Remote Debugging
 3. In Circuit Emulation
 4. Hardware in the loop / Software in the loop
 5. JTAG
4. Autosar

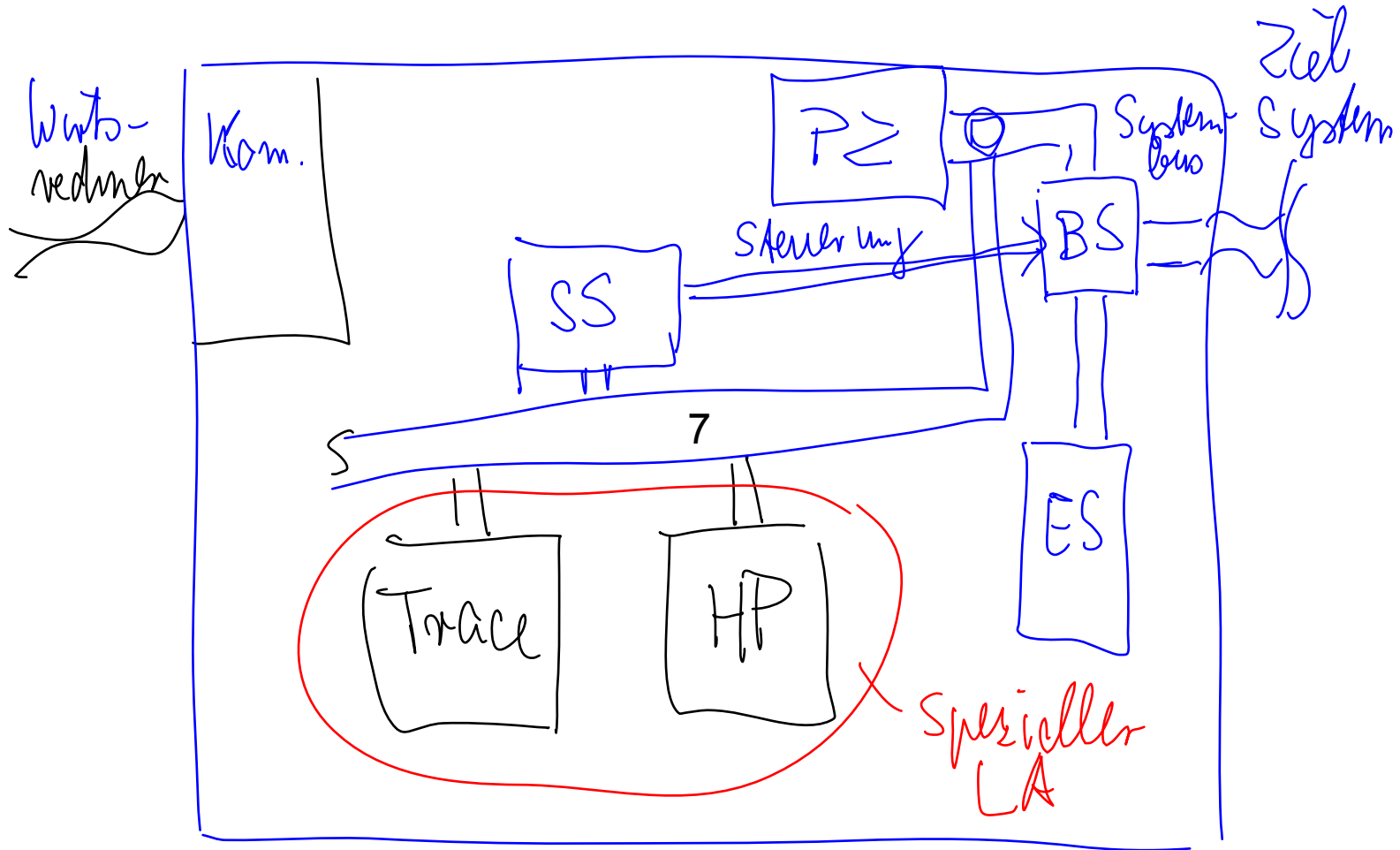




1. Vorgehensmodelle
2. Echtzeit Programmierung
3. Testen
 1. Allgemeines
 2. Remote Debugging
 3. In Circuit Emulation
 4. Hardware in the loop / Software in the loop
 5. JTAG
4. Autosar

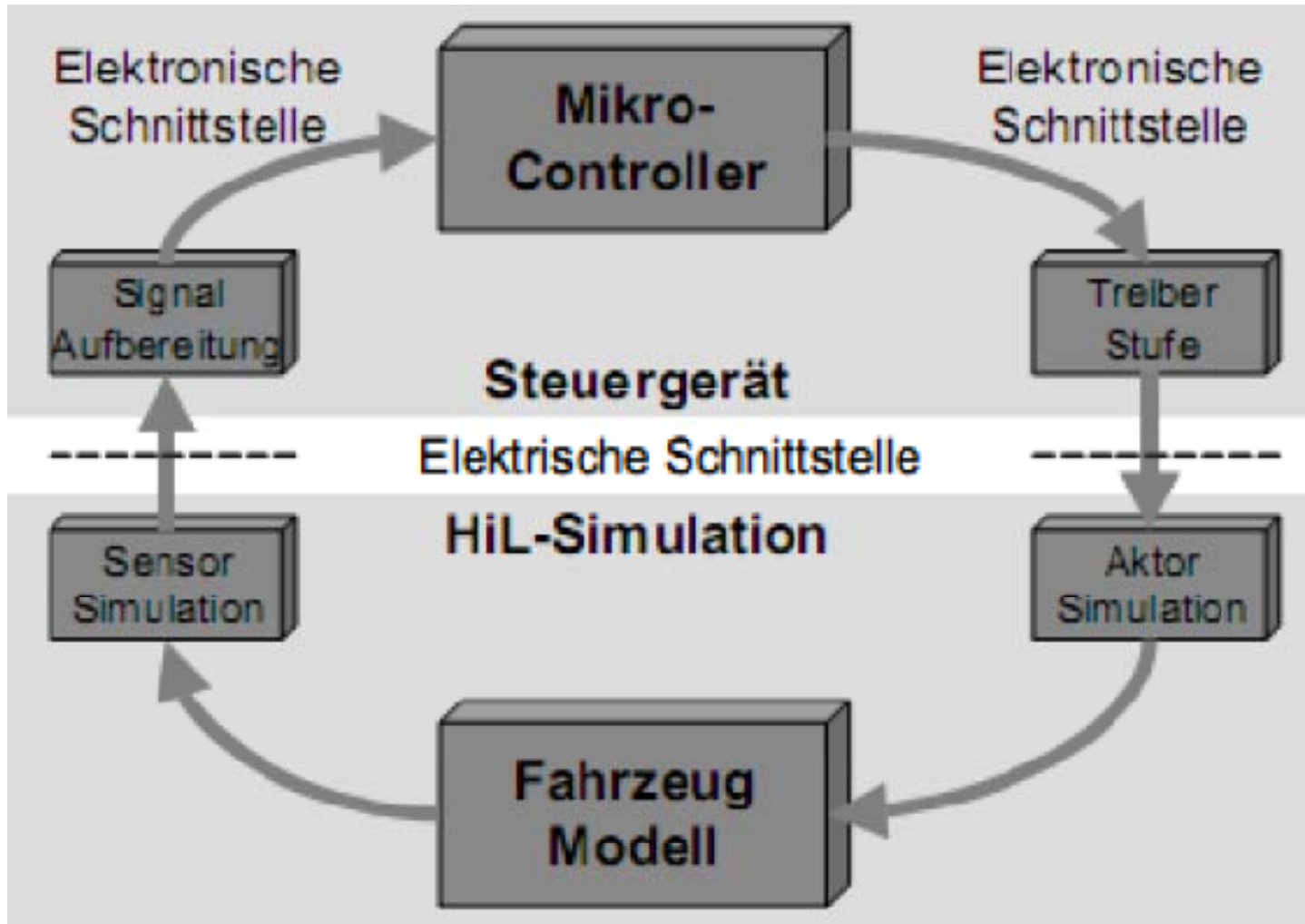


KES → 7. → **Struktur des Emulationsadapters** →



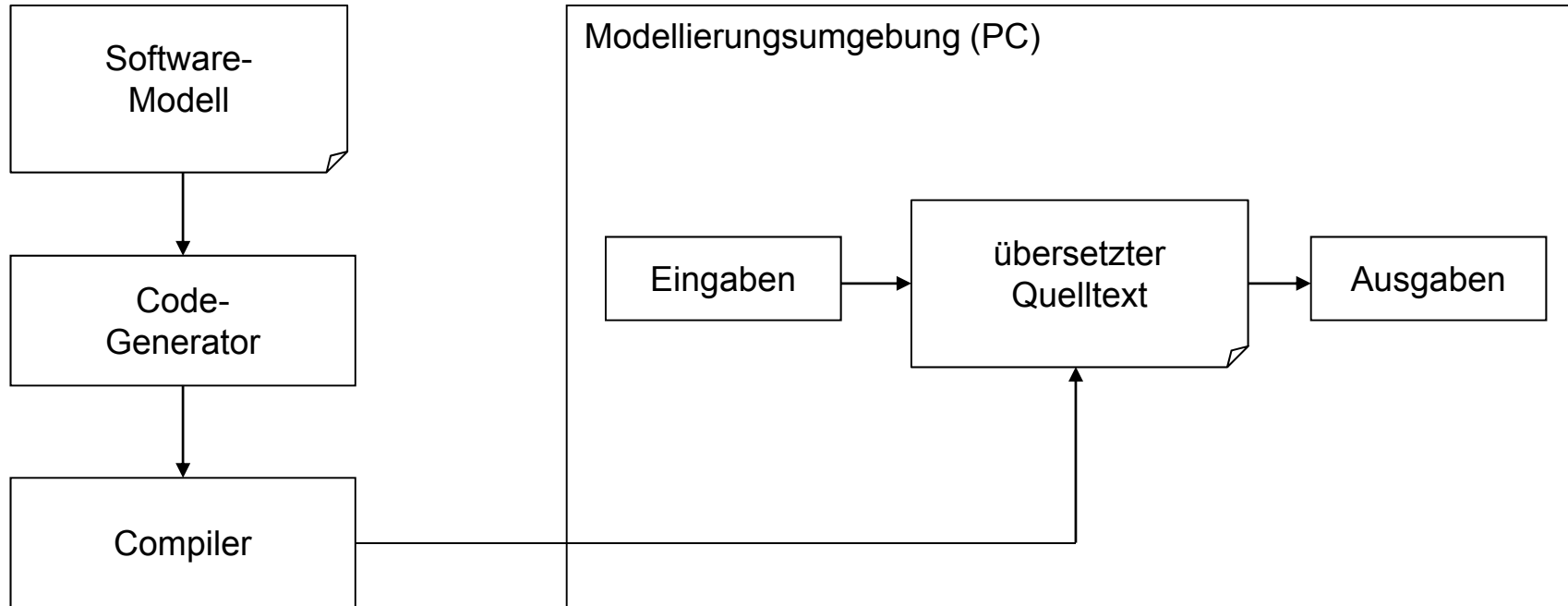


1. Vorgehensmodelle
2. Echtzeit Programmierung
3. Testen
 1. Allgemeines
 2. Remote Debugging
 3. In Circuit Emulation
 4. Hardware in the loop / Software in the loop
 5. JTAG
4. Autosar



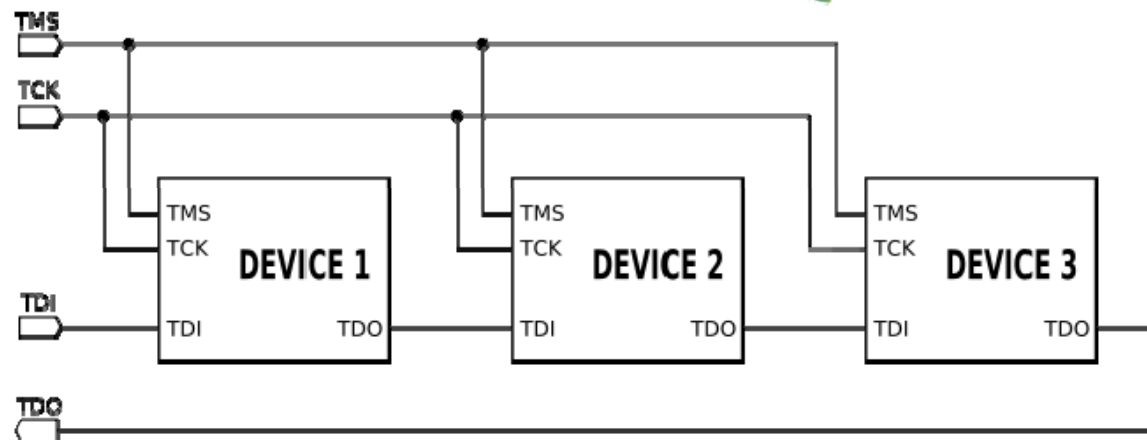
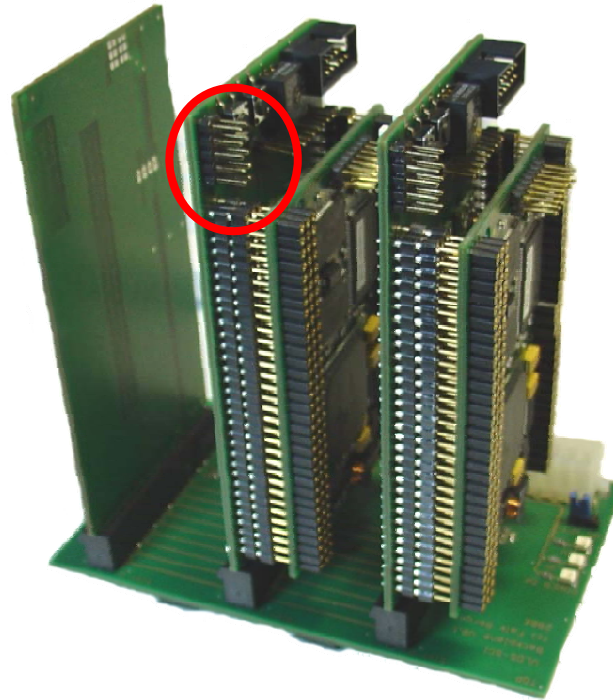
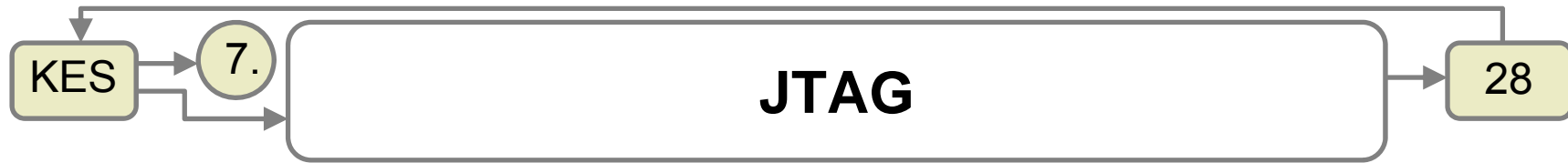
Quelle: Dissertation „Modellbasierter Hardware-in-the-Loop Test von eingebetteten elektronischen Systemen“, Dipl.-Ing. Bernhard Spitzer, Mannheim, 2001

Fengler, Zimmermann 03-2009



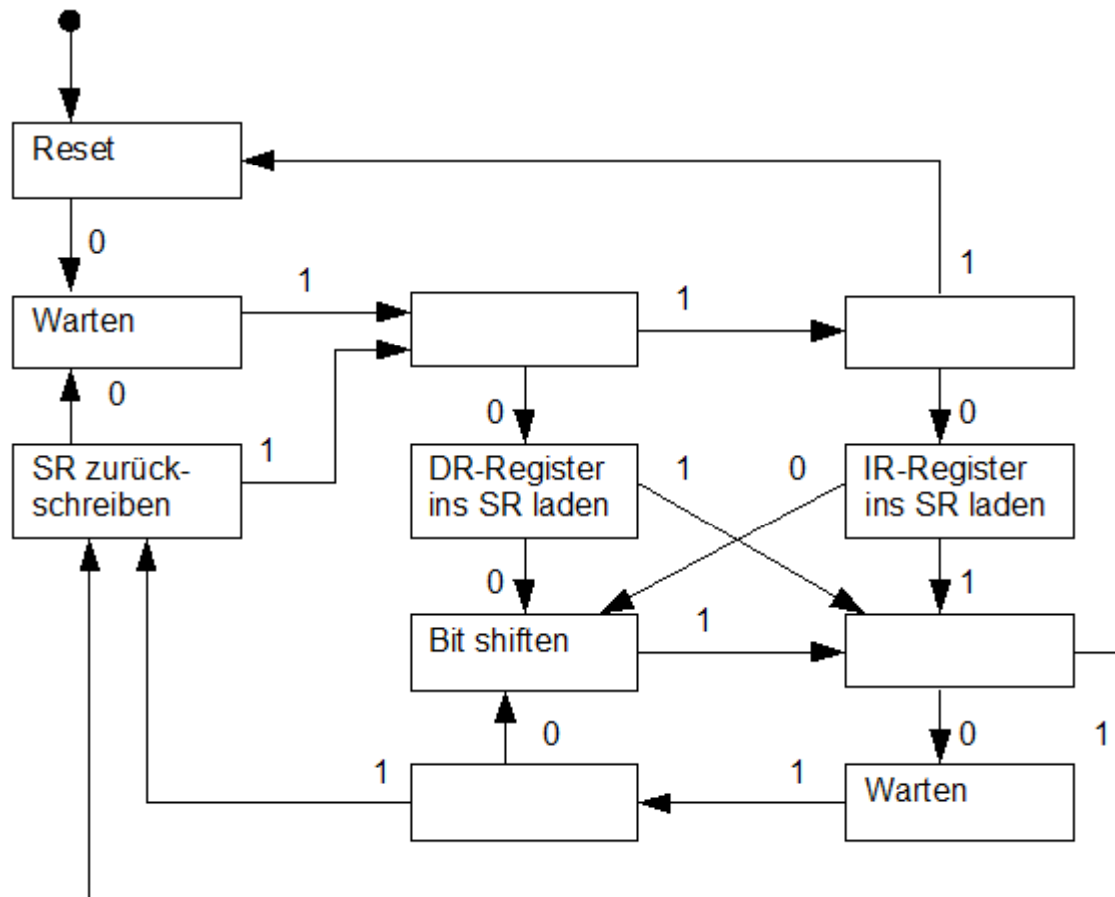


1. Vorgehensmodelle
2. Echtzeit Programmierung
3. Testen
 1. Allgemeines
 2. Remote Debugging
 3. In Circuit Emulation
 4. Hardware in the loop
 5. JTAG
4. Autosar





Auswahl des Registers über das TMS Signal



Legende

SR – Schieberegister

DR – Datenregister

IR – Instruction Register



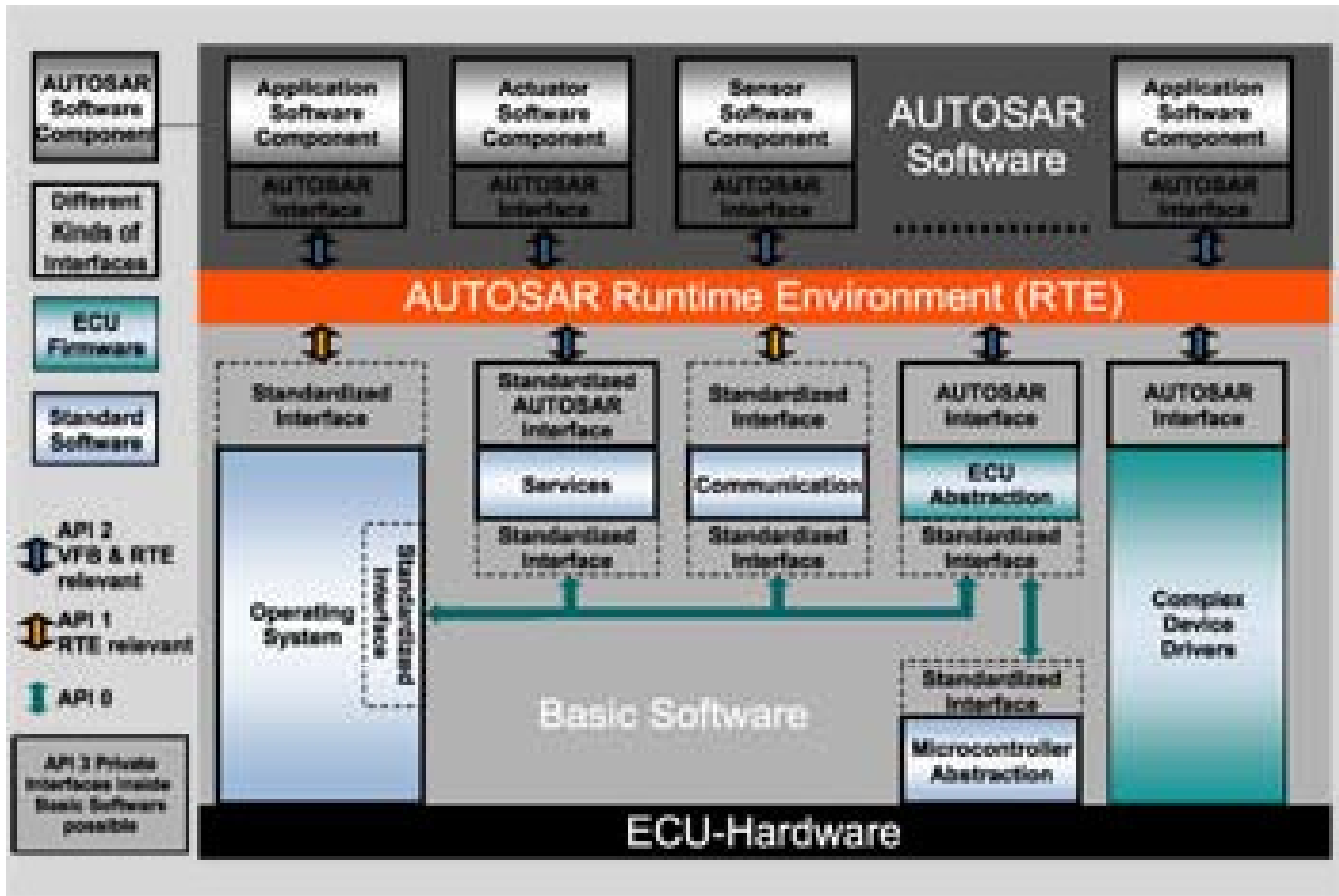
1. Vorgehensmodelle
2. Echtzeit Programmierung
3. Testen
 1. Allgemeines
 2. Remote Debugging
 3. In Circuit Emulation
 4. Hardware in the loop / Software in the loop
 5. JTAG
4. Autosar

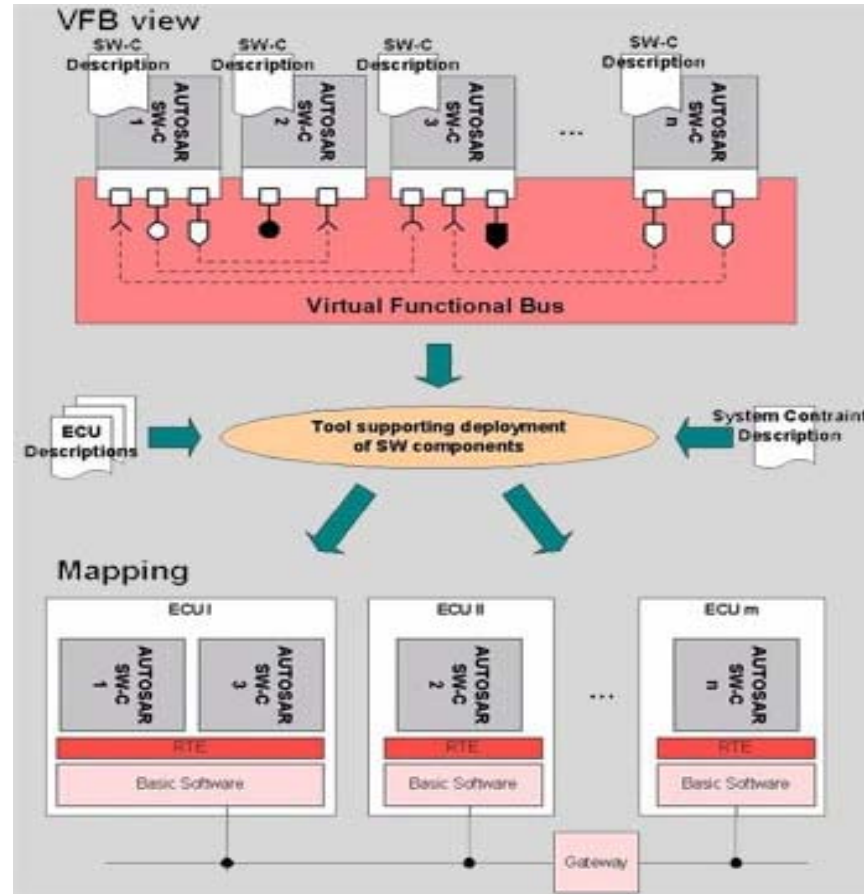


AUTOSAR (AUTomotive Open System ARchitecture) is an open and standardized automotive software architecture, jointly developed by automobile manufacturers, suppliers and tool developers.

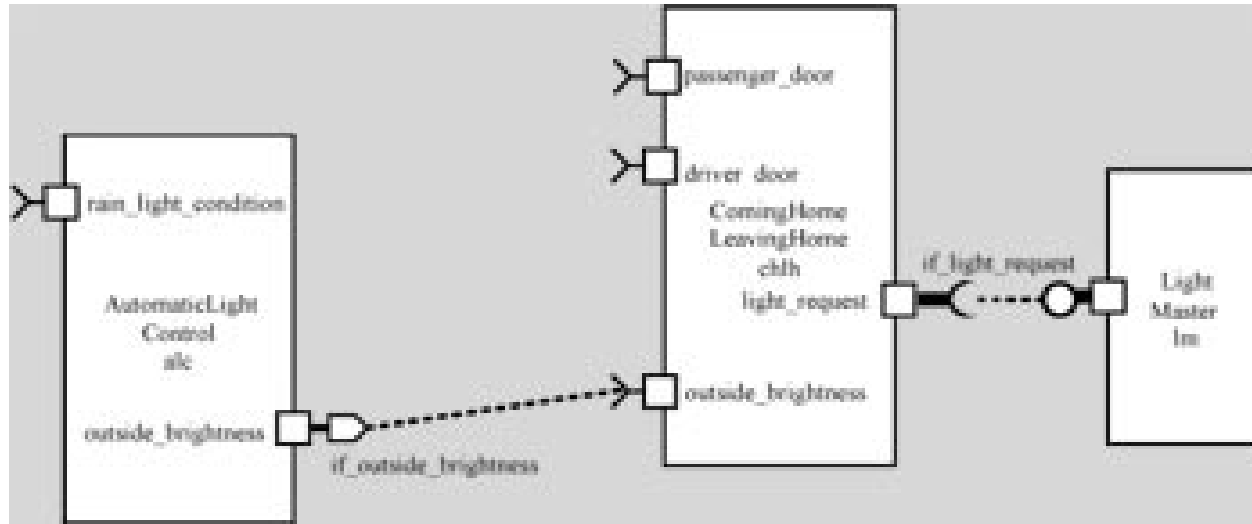
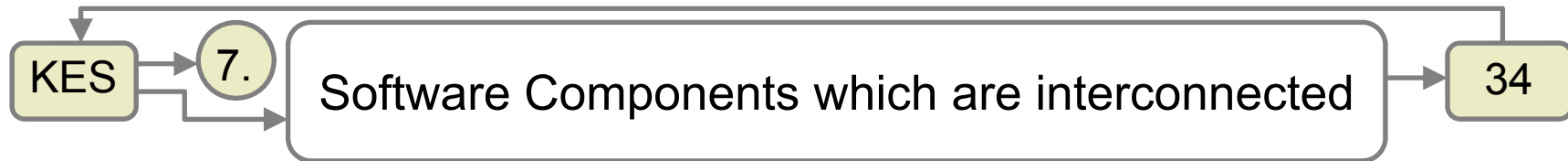
<http://www.autosar.org/>

KES → 7. **Structure of the software for an ECU** →

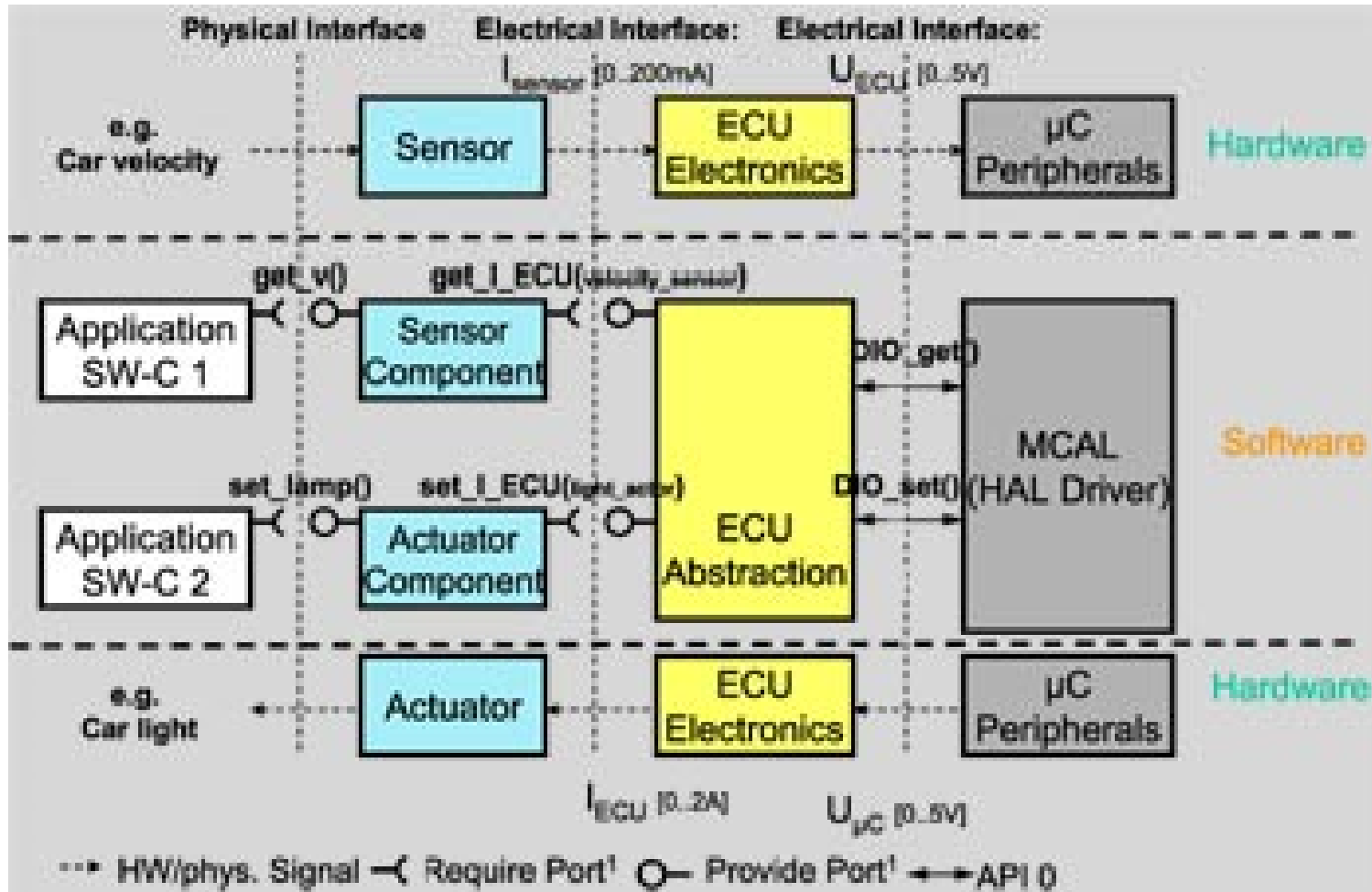


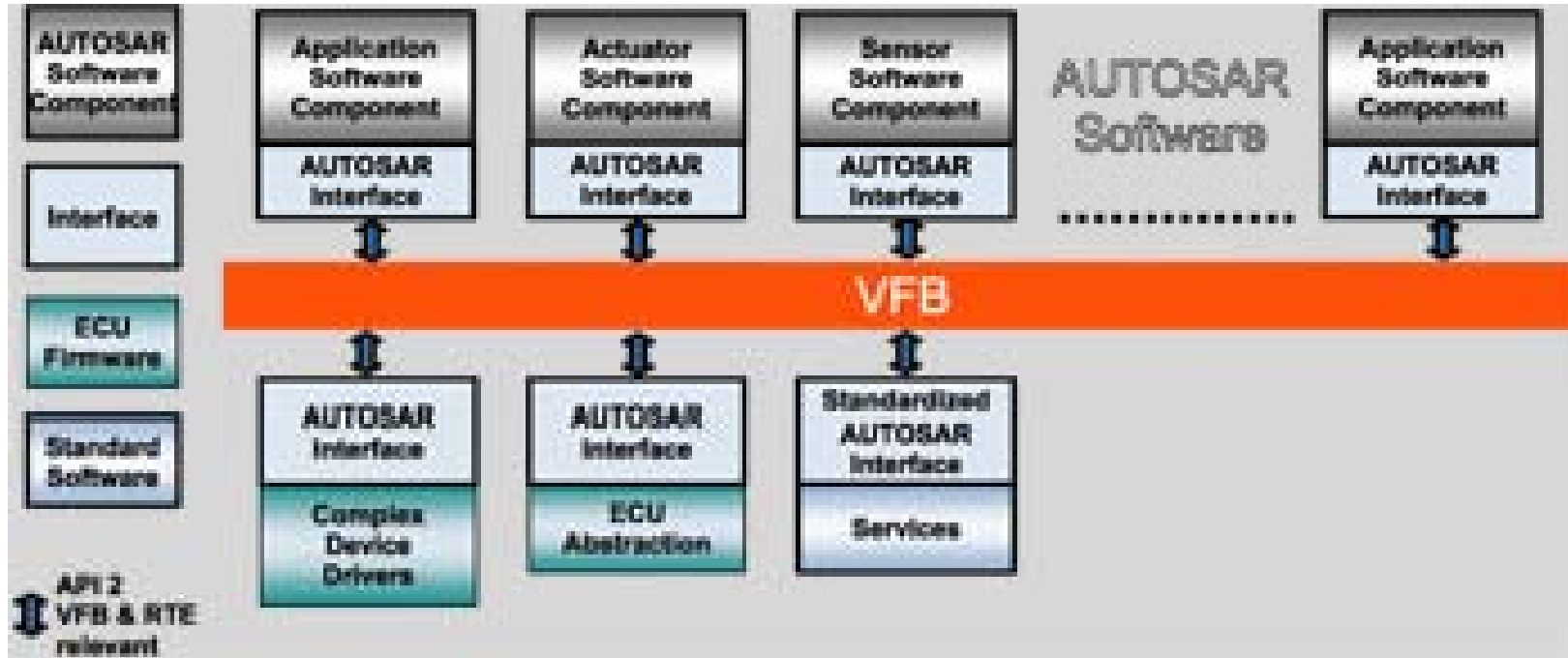


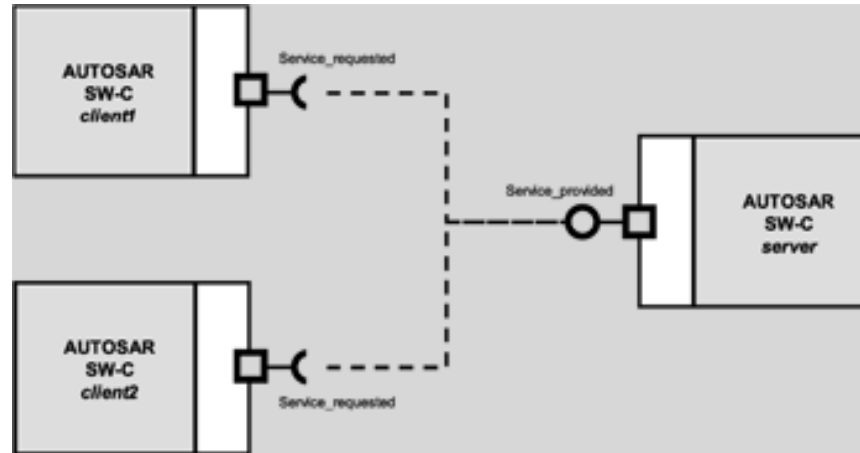
Following the AUTOSAR Methodology, the E/E architecture is derived from the formal description of software and hardware components.



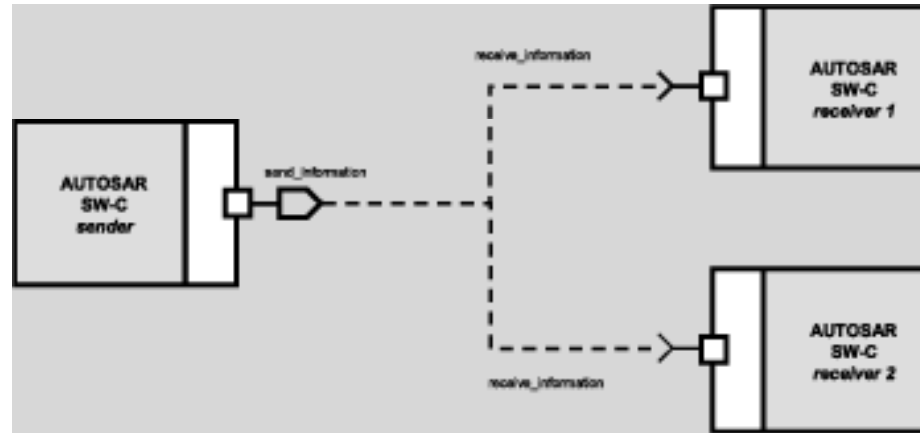
The image shows an application consisting of three AUTOSAR Software Components which are interconnected by several "connectors".







Client-server communication in the VFB view.



Data distribution by asynchronous non-blocking communication in the VFB view.

KES → 7. **Schematic view of AUTOSAR ECU software architecture.** → 39

