Betriebssicherheit

Kapitel 2: Softwareentwicklung für sichere Systeme

Derk Rembold, 2020

Inhalt

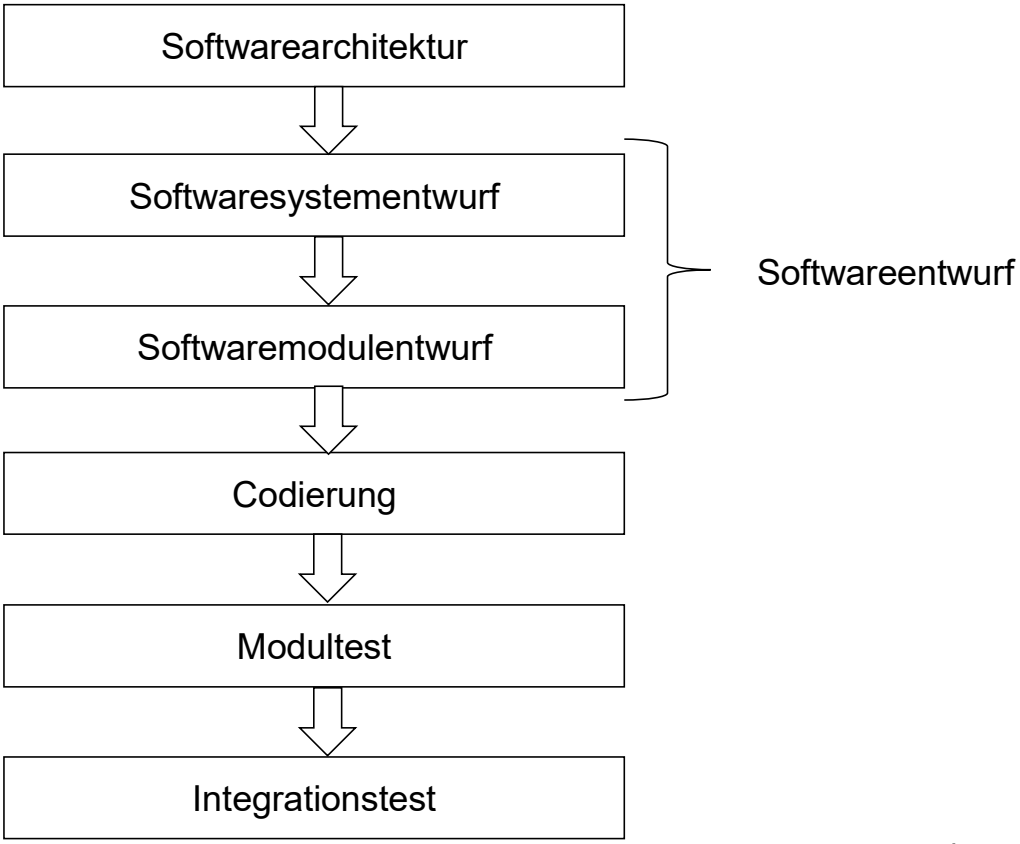
- Empfohlene Softwareverfahren
- Planung der Softwareentwicklung
- Softwareentwurf
- Codierung
- Modultest
- Integrationstest
- Testabdeckung

Empfohlene Softwareverfahren aus IEC 61508

Verfahren	SIL1	SIL2	SIL3	SIL4
Modularisierung und strukturierte Programmierung	++	++	++	++
Entwurfs- und Codierungsrichtlinien	+	++	++	++
Rechnergestützte Entwurfswerkzeuge	+	+	++	++
Defensive Programmierung	o	+	++	++
Semiformale Methoden	+	++	++	++
…				

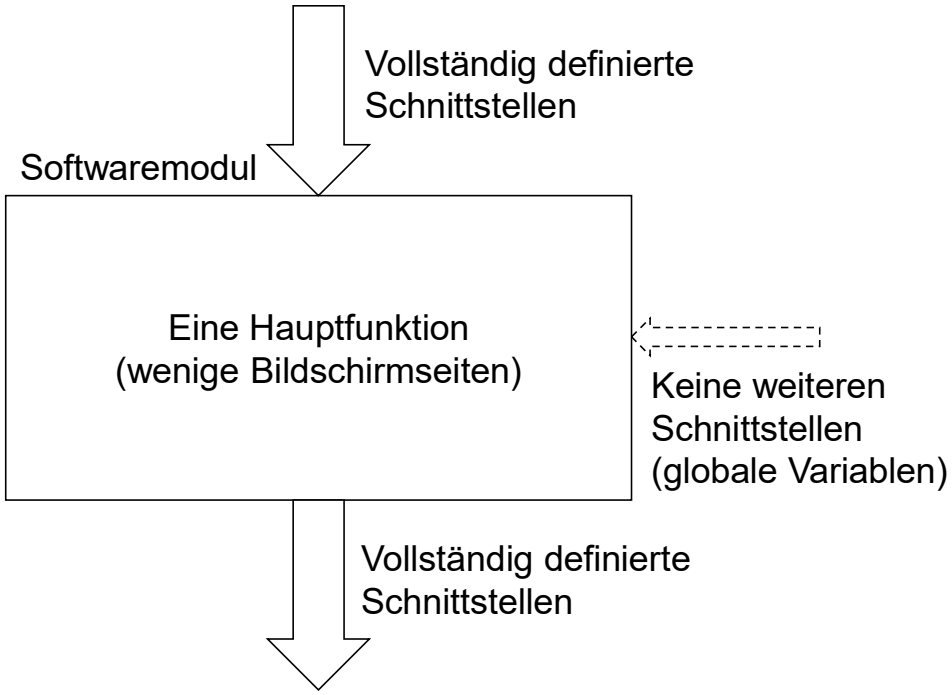
++=besonders empfohlen
+= empfohlen
O= keine Empfehlung

Phasen der Softwareentwicklung



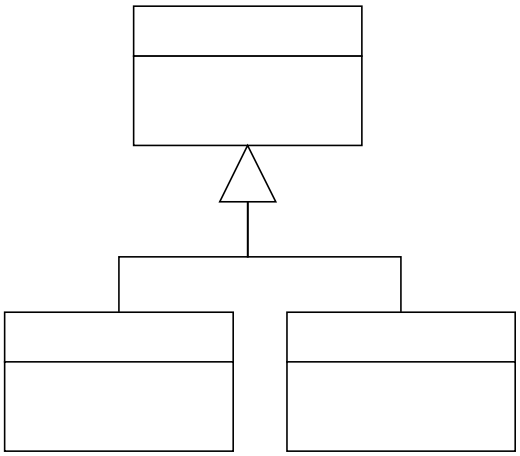
Softwareentwurf: Modularisierung

Ziel: Begrenzung der Komplexität

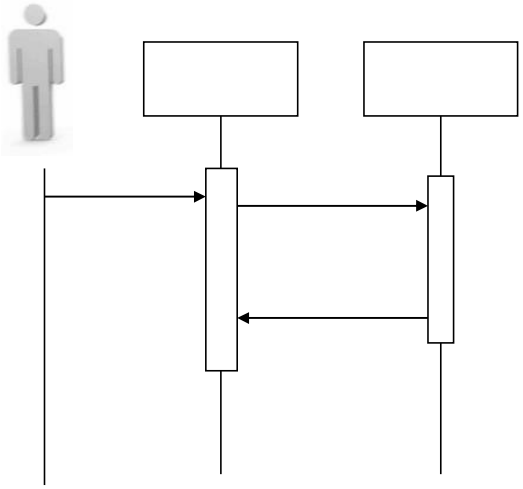


Softwareentwurf: Semiformale Methoden

UML-Klassendiagramme



UML-Sequenzdiagramme



Codierung: Codierungsrichtlinien

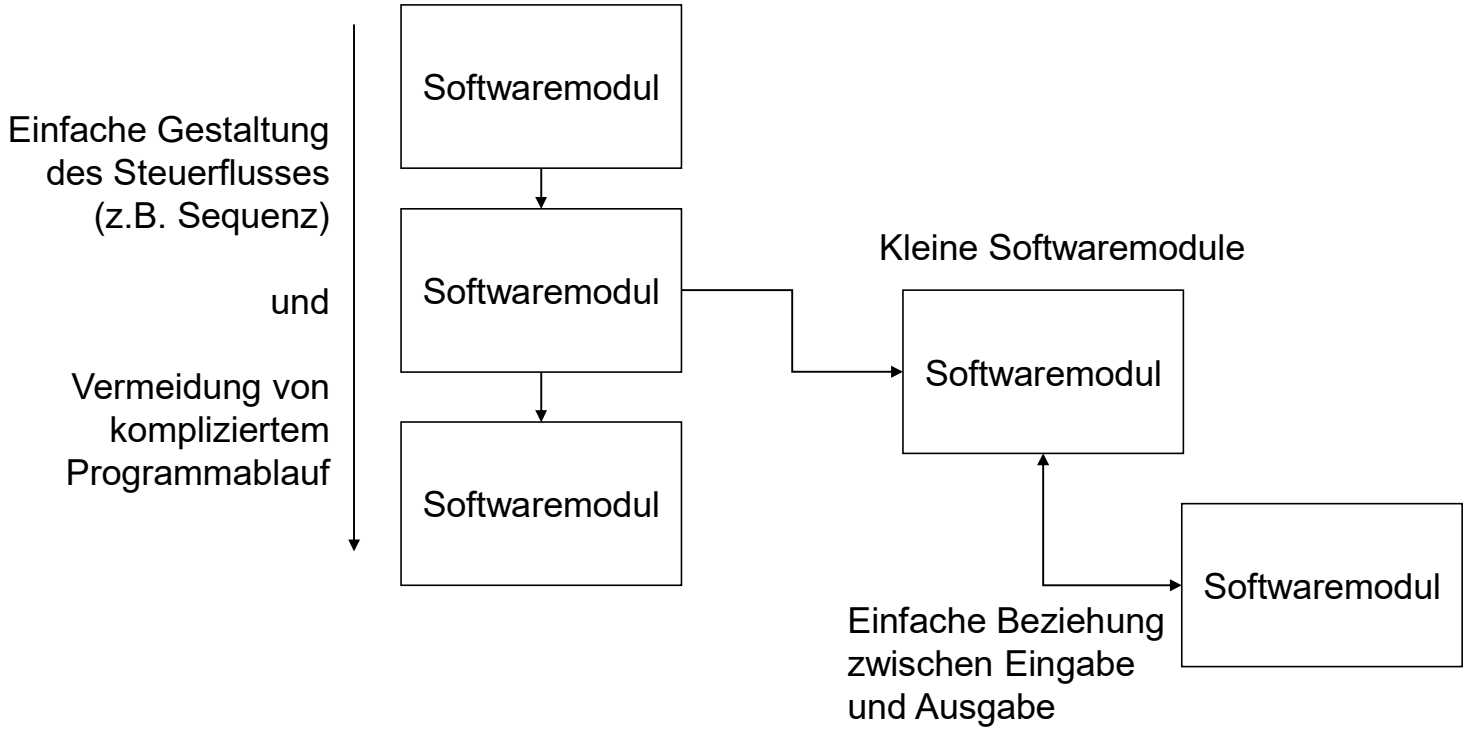
Ab SIL 3 besonders zu empfehlen
Ziel: Lesbarkeit und Verifizierbarkeit zu erhöhen

Verfahren	SIL1	SIL2	SIL3	SIL4
Eingeschränkte Verwendung von Pointer	o	+	++	++
Eingeschränkte Verwendung von Interrupts	+	+	++	++
Keine dynamischen Variablen	o	+	++	++
Alternativ: Überwachung Speicherplatz dyn. Variablen	o	+	++	++

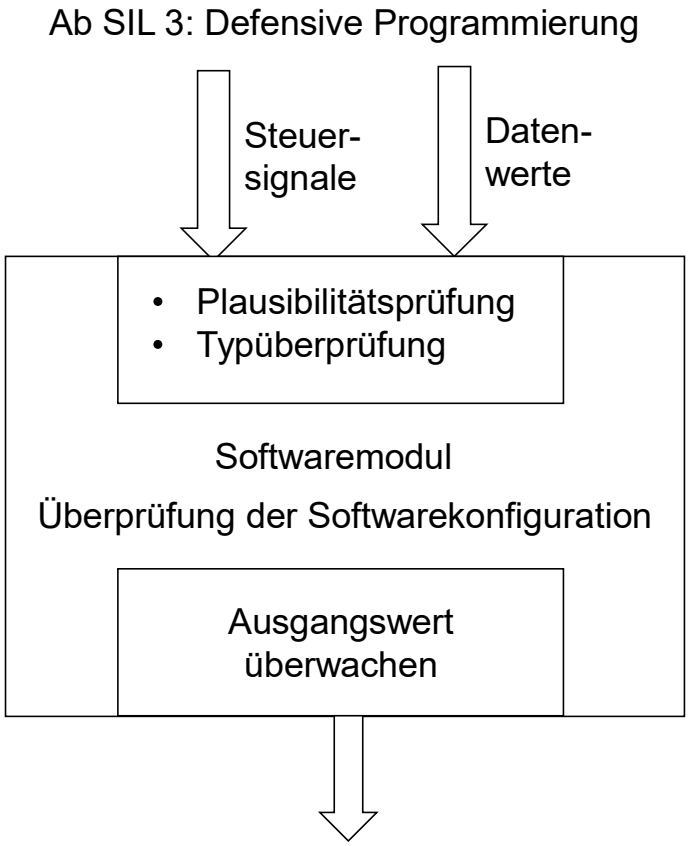
++=besonders empfohlen
+= empfohlen
O= keine Empfehlung

Codierung: Strukturierte Programmierung

Bei jedem SIL: strukturierte Programmierung



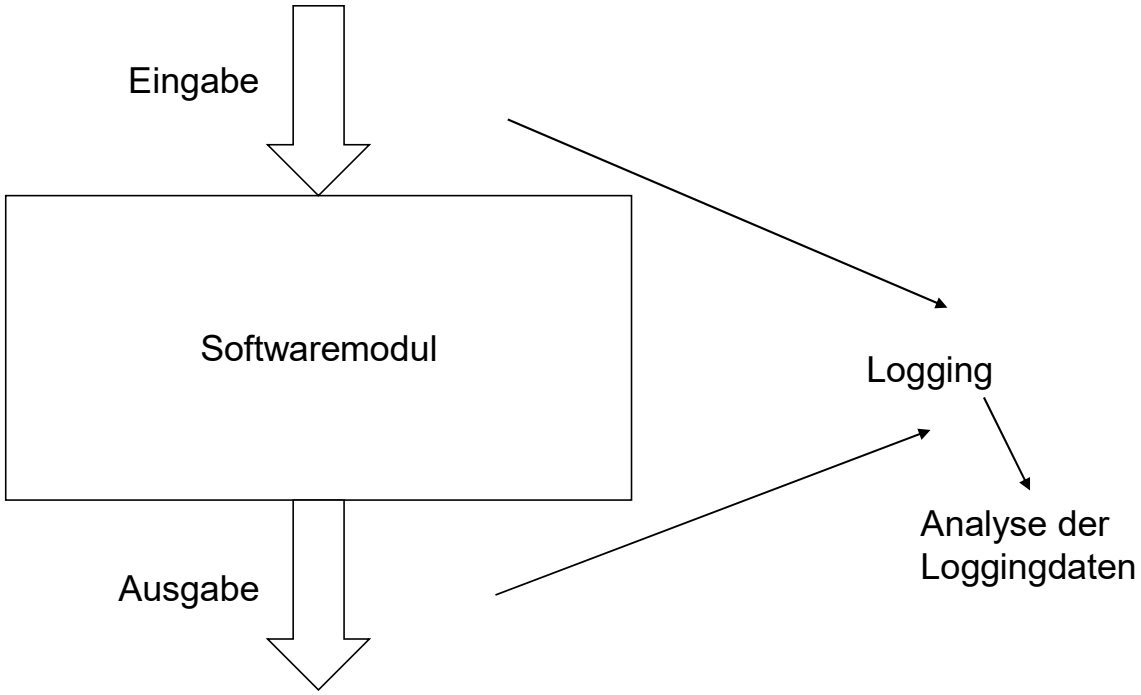
Codierung: Defensive Programmierung



Modultest

Ziel: Nachweis der bestimmungsgemäßen Funktion für jedes Modul

Werkzeuge:
- JUnit (Java)
- CppUnit (cpp)
- NUnit (.NET)
- Unittest (python)

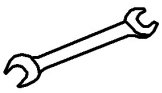


Integrationstest (Software)

Testen des Zusammenwirkens der Module

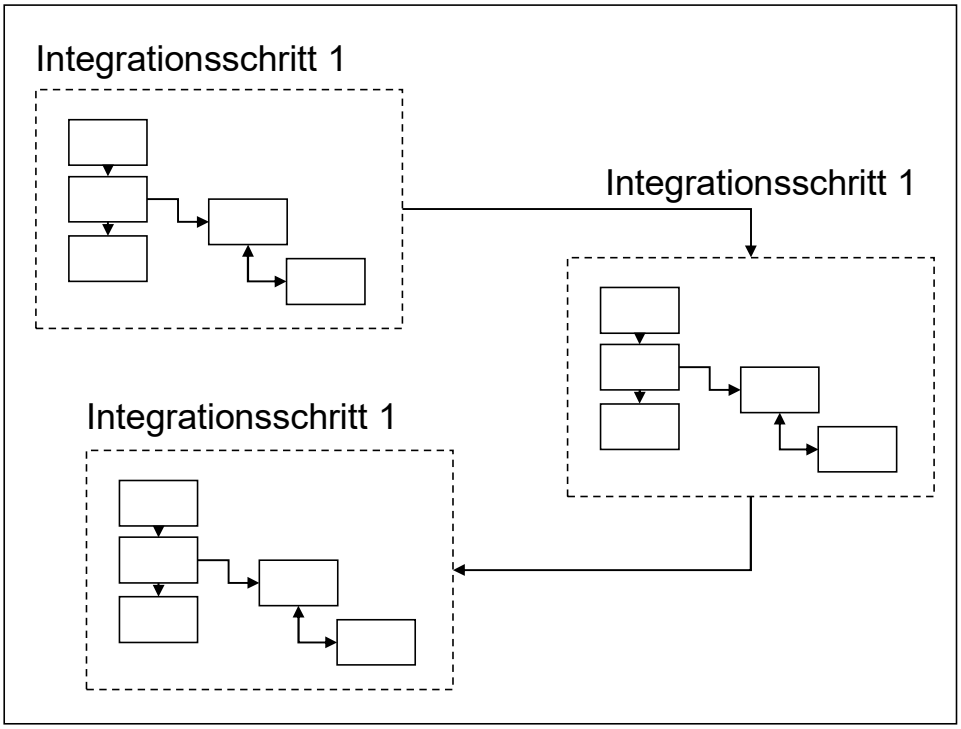
Testspezifikation

- Integrationsschritt 1
  - Testart …
- Integrationsschritt 2
  - Testart …
- Integrationsschritt 3
- …



Testwerkzeuge

Tester

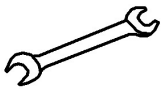


Integrationstest (Software und Hardware)

Testen des Zusammenwirkens von Software und PE Hardware

Testspezifikation

- Integrationsschritt 1
  - Testart …
  - Eingabeparameter
  - Ausgabewerte
- Integrationsschritt 2
  - Testart …
- Integrationsschritt 3
- …



Testwerkzeuge

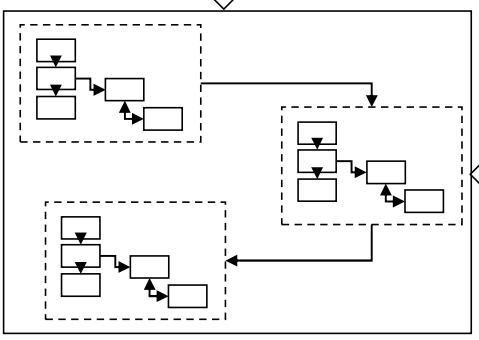
Tester

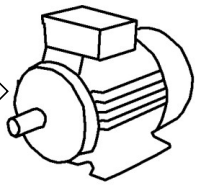
Integrationsschritt 2

Integrationsschritt 3

Eingabe

Ausgabe



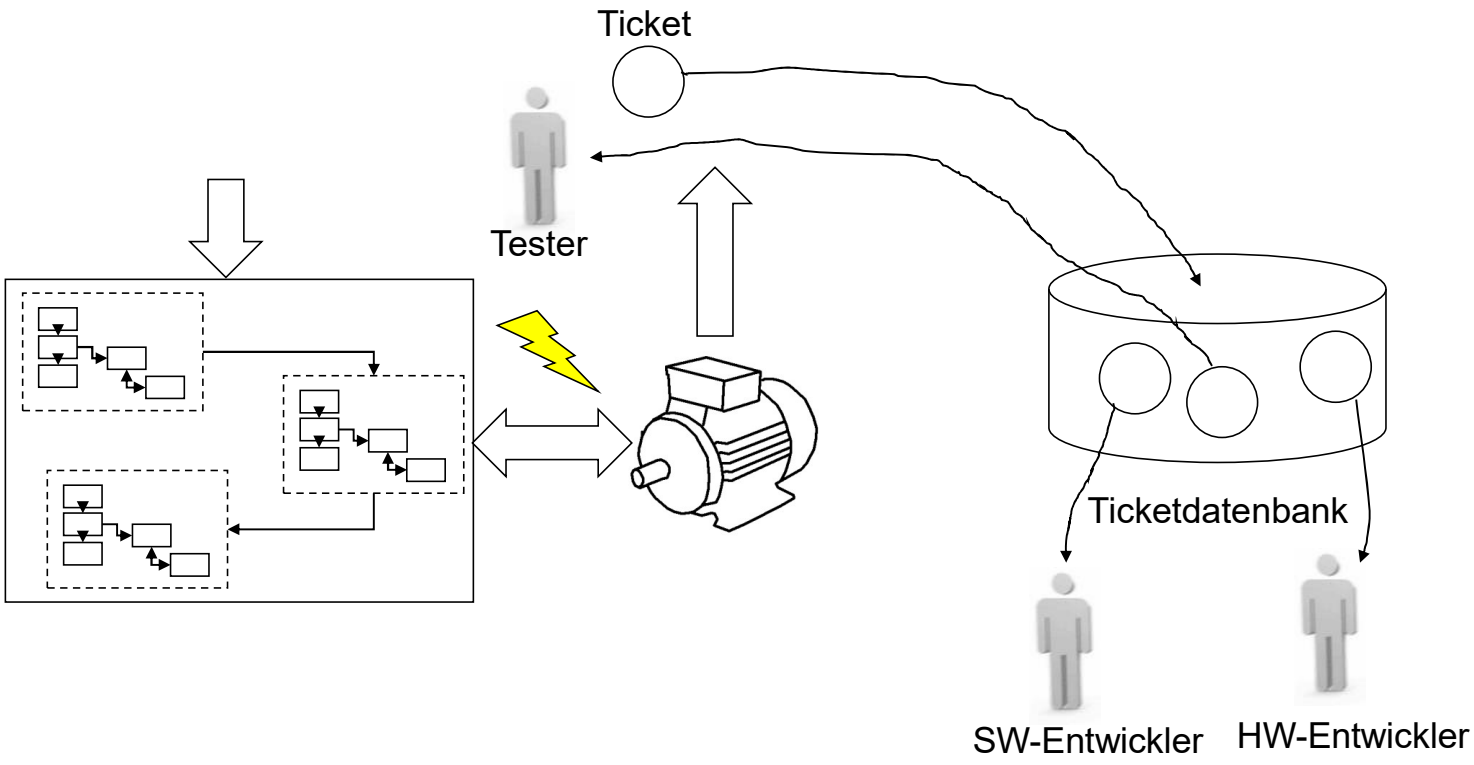


SW-Entwickler

HW-Entwickler

Integrationstest (Software und Hardware)

Dokumentation und Korrektur von Fehler



Integrationstest (Software)

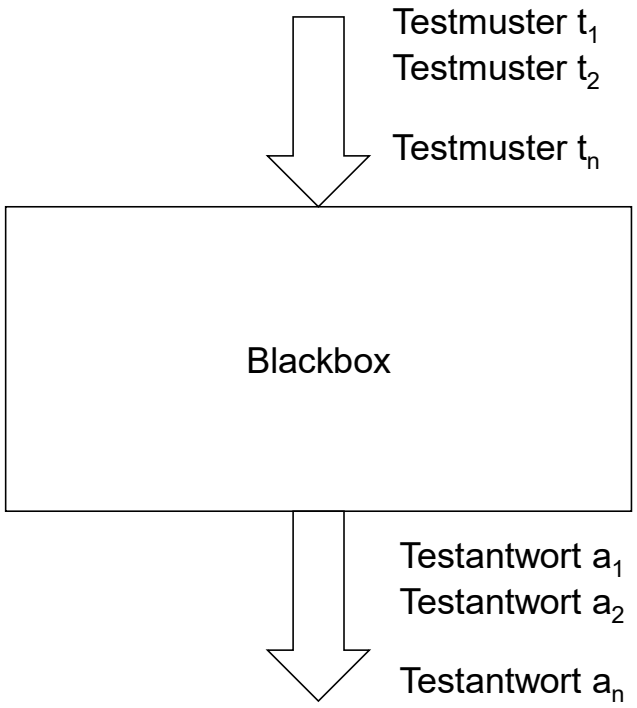
Empfehlungen für Testverfahren

Verfahren	SIL1	SIL2	SIL3	SIL4
Dynamische Analyse und Test	+	++	++	++
Datenaufzeichnung und Analyse	++	++	++	++
Funktionstest und Blackbox-Test	++	++	++	++
Leistungstest	+	+	++	++
Schnittstellentest	+	+	++	++

++=besonders empfohlen
+= empfohlen
O= keine Empfehlung

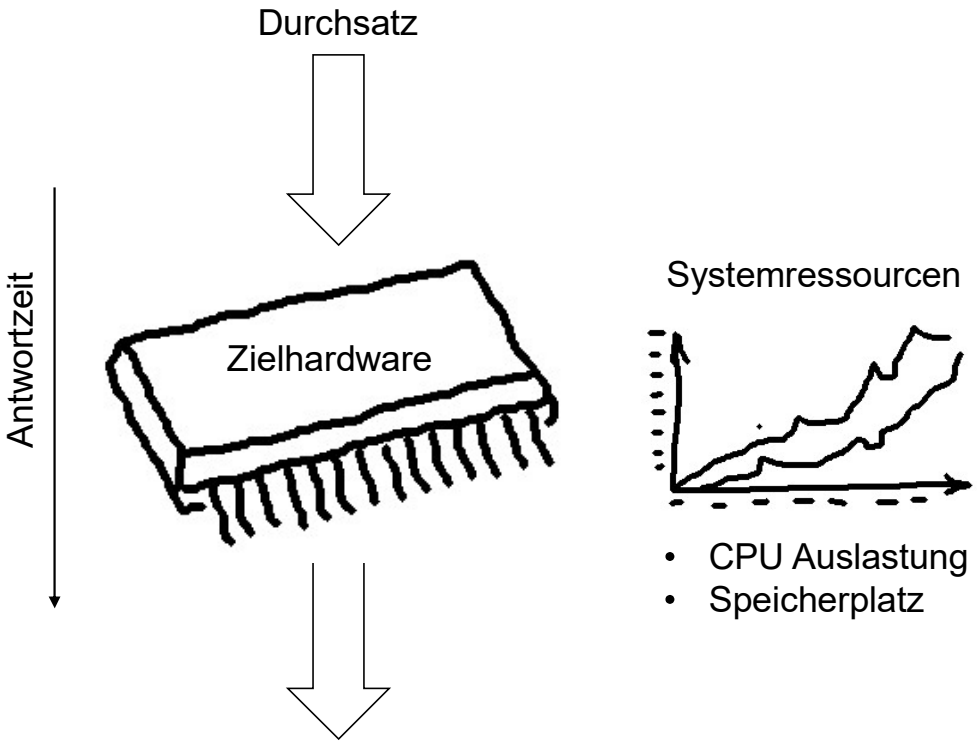
Integrationstest (Software und Hardware)

Blackbox Test



Integrationstest (Software und Hardware)

Leistungstest



Integrationstest (Software und Hardware)

Empfehlungen für Validierungsverfahren

Verfahren	SIL1	SIL2	SIL3	SIL4
Funktionstest und Blackbox-Test	++	++	++	++
Statistische Tests	o	+	+	++
Simulation	+	+	++	++

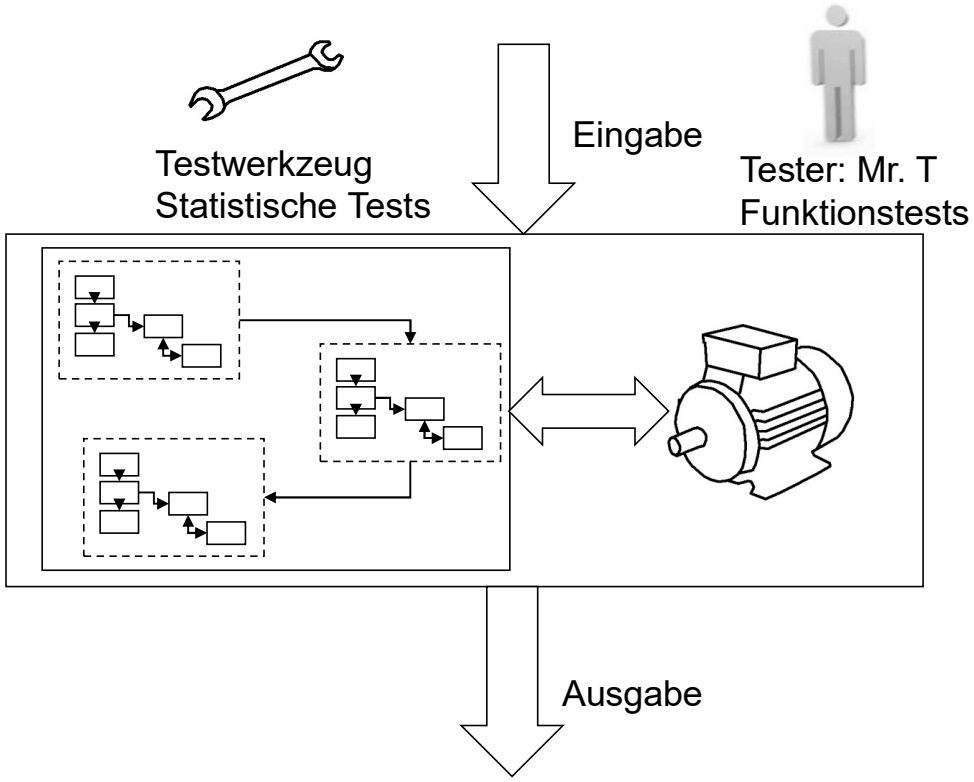
++=besonders empfohlen
+= empfohlen
O= keine Empfehlung

Integrationstest

Validierung bezüglich Sicherheit

Spezifikation der Sicherheitsanforderung

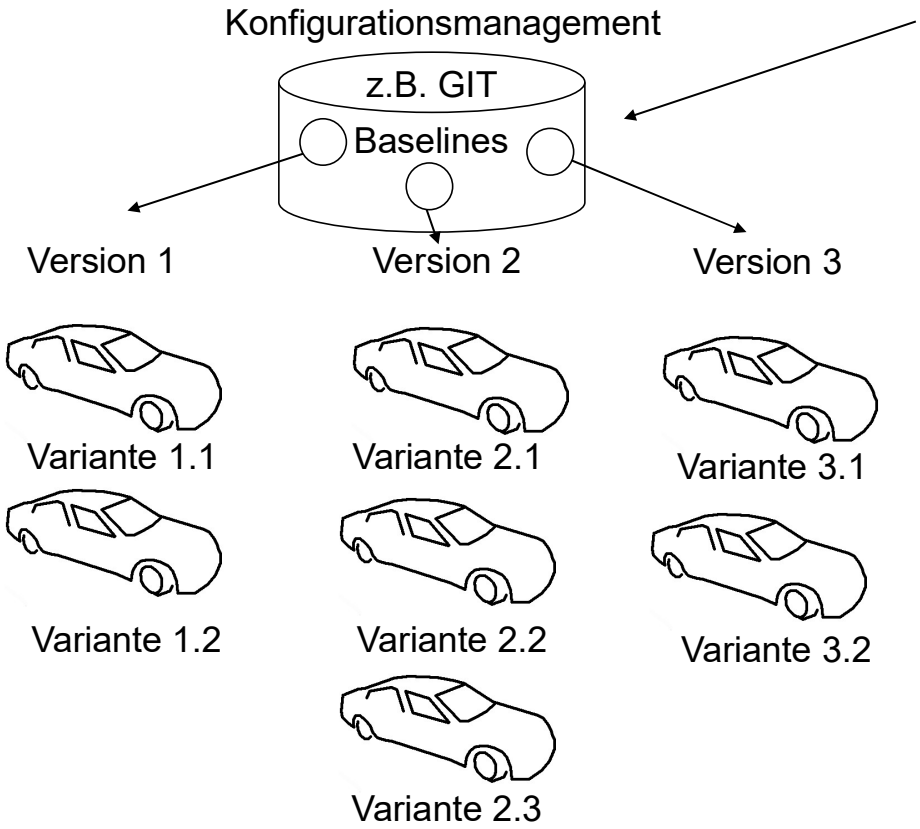
Sicherheitsanforderung 1
Person: Mr. T
Betriebsart: …
Verfahren: …
Testumgebung: …
Eingabeparameter: …
Ausgabewerte: …
Erfolgsbeurteilung: …



Ergebnisse der Validierung ist Teil des Sicherheitsnachweises!

Konfigurationsmanagement

Unabhängig von SIL muss Software unter Konfigurationskontrolle gestellt werden



Konfigurationselemente:

- Anforderungen
- Spezifikationen
- Code
- Testpläne
- Werkzeuge
- Entwicklungsumgebungen

Testabdeckung bei ISO 26262

Software darf keine unbeabsichtigte Funktionalität erhalten

Methoden	ASIL1	ASIL2	ASIL3	ASIL4
Anweisungsüberdeckung C0	++	++	+	+
Zweigüberdeckung C1	+	++	++	++
Modifizierte Bedingung / Entscheidungsüberdeckung MC / DC	+	+	+	++

++=besonders empfohlen
+= empfohlen
o= keine Empfehlung

Testabdeckung bei ISO 26262

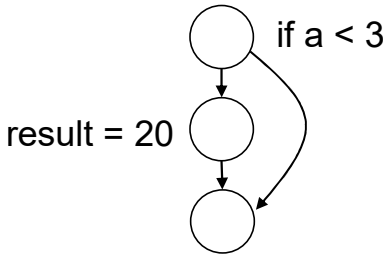
Kontrollflussgraph

Ein Kontrollflussgraph ist ein gerichteter Graph mit einer Menge von Knoten und gerichteten Kanten.

Knoten sind Anweisungen und Kanten stellen den Kontrollfluss dar.

Zum Beispiel:

```
if a < 3:
    result = 20
```

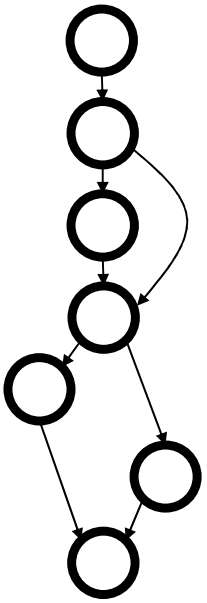


Testabdeckung bei ISO 26262

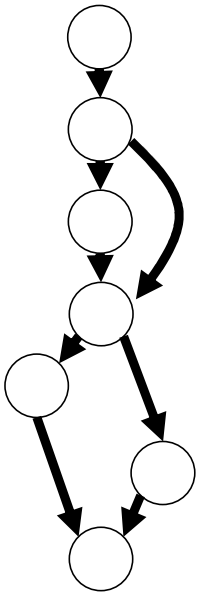
C0 und C1 Überdeckung

```python
def calc(x,y):

    result = 0

    if x > 0 and y > 0:
        result = 30

    if x > 10 and y < 10:
        result += 30
    else:
        result += 10

    return result
```

C0 Überdeckung



C1 Überdeckung



Kontrollflussgraphen

Testabdeckung bei ISO 26262

Modifizierte Bedingung / Entscheidungsüberdeckung MC / DC

Alle bool‘sche Ausdrücke sollen sowohl auf True als auch auf False getestet werden

```
def calc(x,y):

    result = 0

    if x > 0 and y > 0:
        result = 30

    if x > 10 and y < 10:
        result += 30
    else:
        result += 10

    return result
```