**engineering** the **embedded world**

embeX
Your embedded experts

Herzlich Willkommen!


# Software testen gemäß IEC 61508
# ganz einfach oder unmöglich?

Dr. Martin Lange
embeX GmbH

embeX
Your embedded experts

# Die embeX GmbH

- Dienstleister für die Entwicklung von embedded Systemen
- Hardware, Software, Firmware
- von der Idee bis zur Serie
- gegründet 2001 in Freiburg im Breisgau
- über 100 Mitarbeiter
- seit über 10 Jahren Entwicklung von sicheren Geräten nach IEC 61508
- außerdem: Avionik, Railway, Medizintechnik, Standardindustrie-automatisierung

- **Problemstellung**
- Lösungsansatz
- Konkrete Vorgehensweise
- Ergebnis

# Problemstellung

## IEC 61508

Funktionale Sicherheit sicherheitsbezogener elektrischer/ elektronischer/programmierbarer elektronischer Systeme

Basisnorm für u.a.:

- Industrie- und Prozessautomation
- Maschinensteuerungen
- Automotive
- Schienenverkehr

embeX
Your embedded experts

## IEC 61508

Funktionale Sicherheit sicherheitsbezogener elektrischer/ elektronischer/programmierbarer elektronischer Systeme

Teile:

1. Allgemeine Anforderungen
2. Anforderungen an sicherheitsbezogene elektrische/elektronische/ programmierbare elektronische Systeme
3. Anforderungen an Software
4. Begriffe und Abkürzungen
5. Beispiele zur Ermittlung der Stufe der Sicherheitsintegrität (safety integrety level)
6. Anwendungsrichtlinie für IEC 61508-2 und IEC 61508-3
7. Überblick über Verfahren und Maßnahmen

# Problemstellung

## IEC 61508-3: Anforderungen an Software

Anhänge A und B:

**Tabelle A.5 – Softwareentwurf und Softwareentwicklung – Test der Softwaremodule und**

(siehe 7.4.7 und 7.4.8)

Verweise auf IEC 61508-7

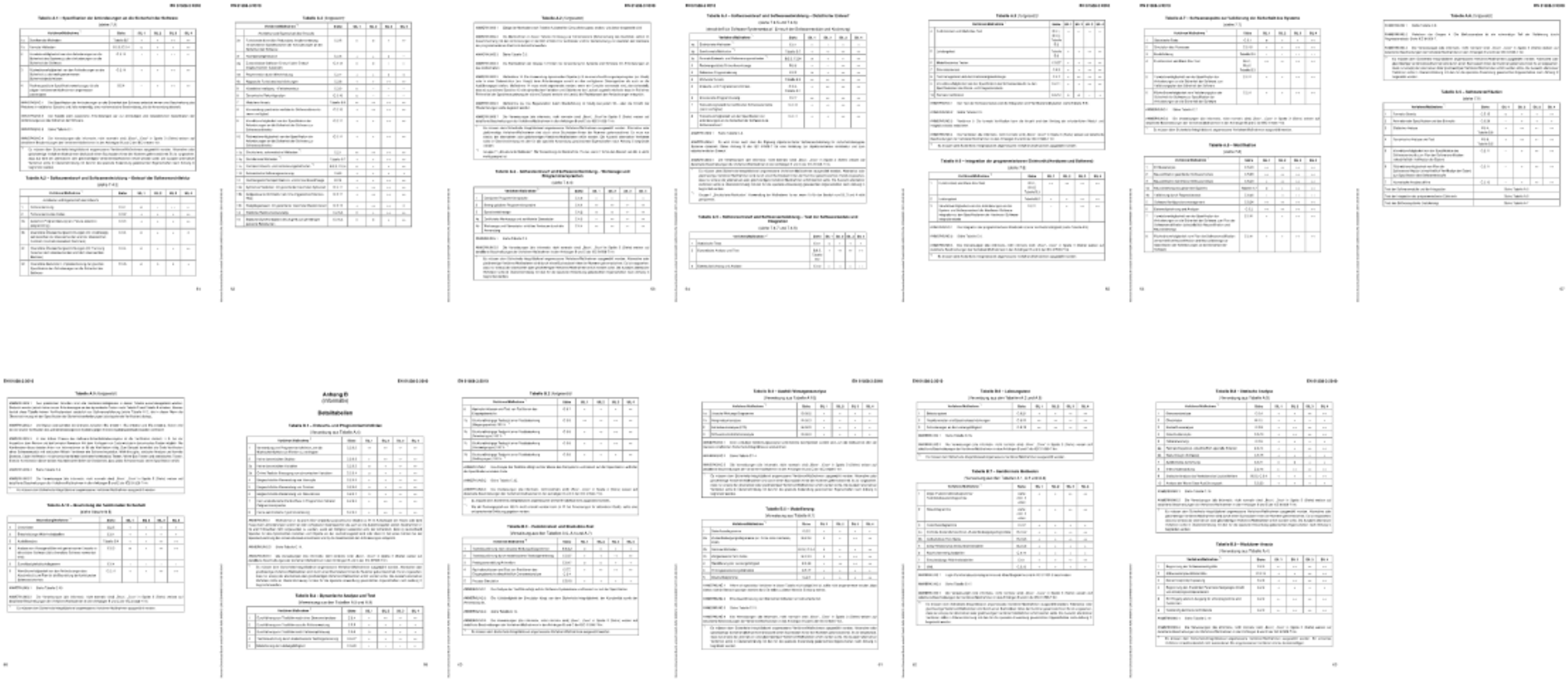
| | Verfahren/Maßnahme *) | Siehe | SIL 1 | SIL 2 | SIL 3 | SIL 4 |
|---|---|---|---|---|---|---|
| 1 | Statistische Tests | C.5.1 | o | + | + | + |
| 2 | Dynamische Analyse und Test | B.6.5, Tabelle B.2 | + | ++ | ++ | ++ |
| 3 | Datenaufzeichnung und Analyse | C.5.2 | ++ | ++ | ++ | ++ |
| 4 | Funktionstest und Black-Box-Test | B.5.1, B.5.2, B.3 | ++ | ++ | ++ | ++ |
| 5 | Leistungstest | Tabelle B.6 | + | + | ++ | ++ |
| 6 | Modellbasiertes Testen | C.5.27 | + | + | ++ | ++ |
| 7 | Schnittstellentest | C.5.3 | + | + | ++ | ++ |
| 8 | Testmanagement und Automatisierungswerkzeuge | C.4.7 | + | ++ | ++ | ++ |
| 9 | Vorwärtsverfolgbarkeit von der Spezifikation des Softwareentwurfs zu den Spezifikationen des Modul- und Integrationstests. | C.2.11 | + | + | ++ | ++ |
| 10 | Formale Verifikation | C.5.12 | o | o | + | + |

‚++' = sehr empfohlen
≠ verpflichtend!

# Problemstellung

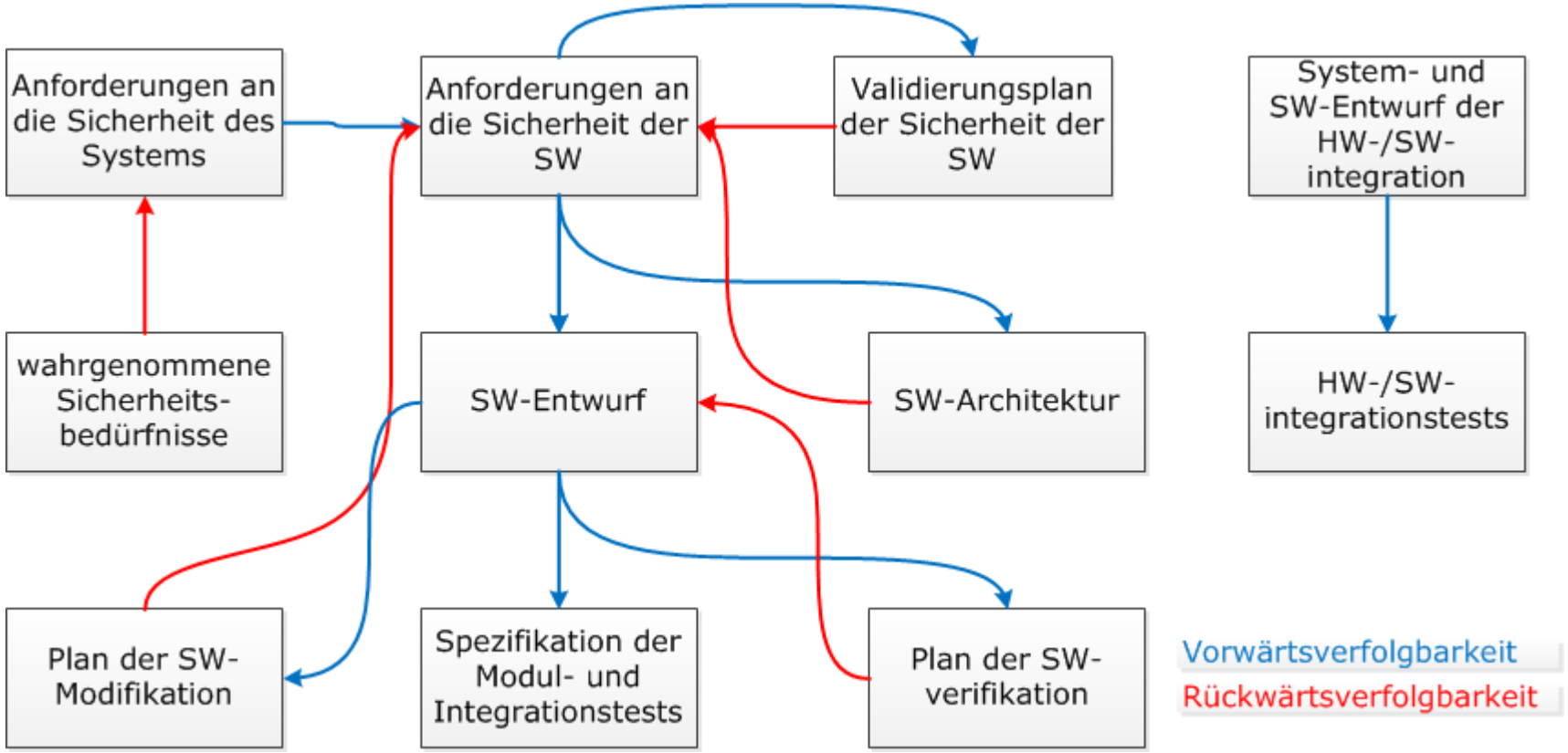
## IEC 61508-3: Anforderungen an Software

Anhänge A und B:



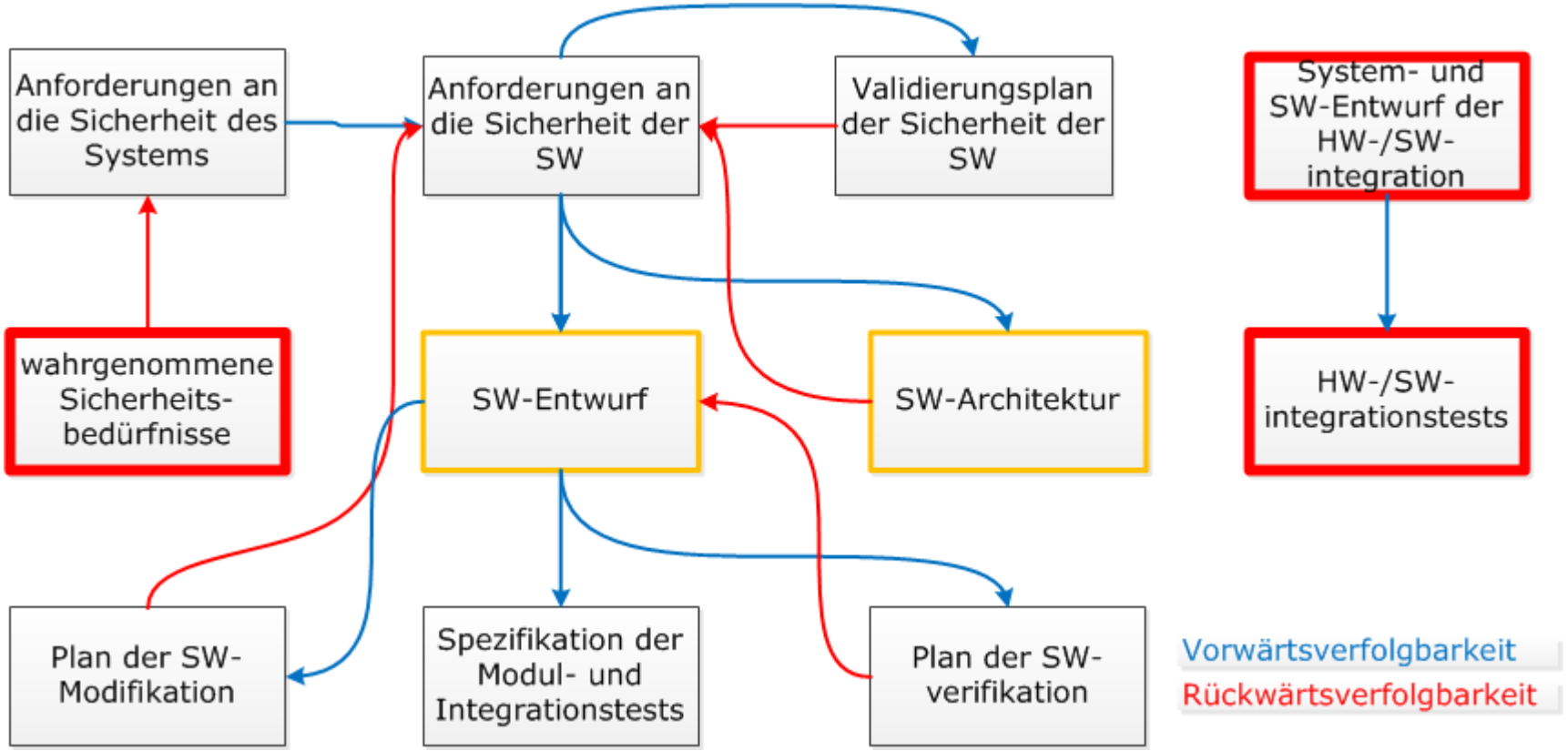
# Problemstellung

## Traceability



# Problemstellung

## Traceability



# Problemstellung

## Verwendung von Anhang B

**Tabelle B.3 – Funktionstest und Black-Box-Test**

(Verweisung aus den Tabellen A.5, A.6 und A.7)

| | Verfahren/Maßnahme *) | Siehe | SIL 1 | SIL 2 | SIL 3 | SIL 4 |
|---|---|---|---|---|---|---|
| 1 | Testfallausführung nach Ursache-Wirkungs-Diagrammen | B.6.6.2 | o | o | + | + |
| 2 | Testfallausführung durch modellbasierte Testfallgenerierung | C.5.27 | + | + | ++ | ++ |

Tabelle A.5: Softwareentwurf und Softwareentwicklung – Test der Softwaremodule und Integration

Tabelle A.6: Integration der programmierbaren Elektronik (Hardware und Software)

Tabelle A.7: Softwareaspekte zur Validierung der Sicherheit des Systems

embeX
Your embedded experts

## Was also tun?

Möglichkeit 1:

Ich definiere, wie ich in meinem konkreten Softwareprojekt die Software testen und verifizieren will.

Und lass mir dann vom Prüfer sagen, was noch fehlt.

Im nächsten Projekt wird dem Prüfer sicherlich etwas anderes fehlen …

Möglichkeit 2:

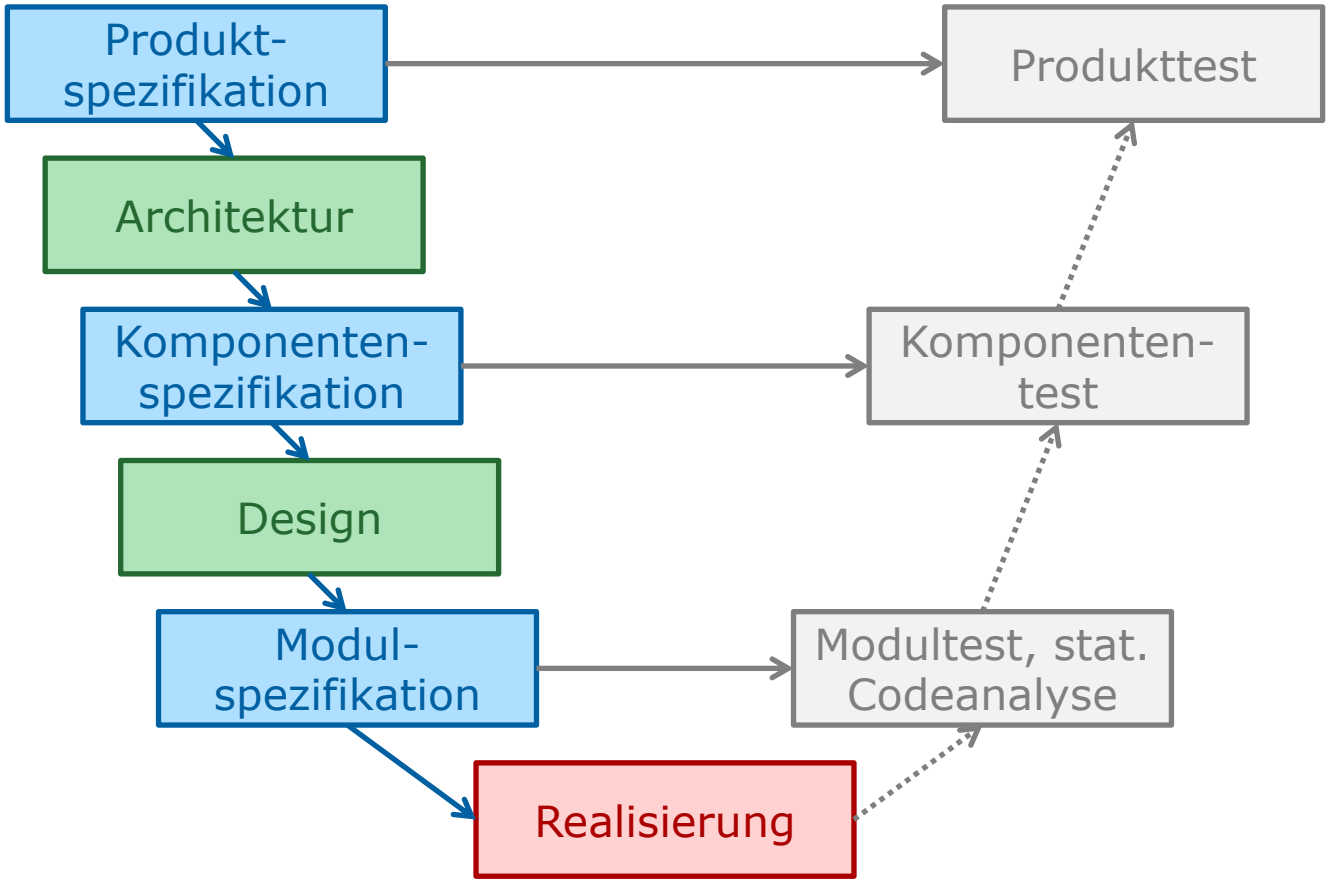
Ich definiere für meine zukünftigen Softwareprojekte, wie ich die Software testen und verifizieren will.

Ich weise einem Prüfer nach, dass das konform zur Norm ist.

Und lasse mir das von ihm bestätigen.

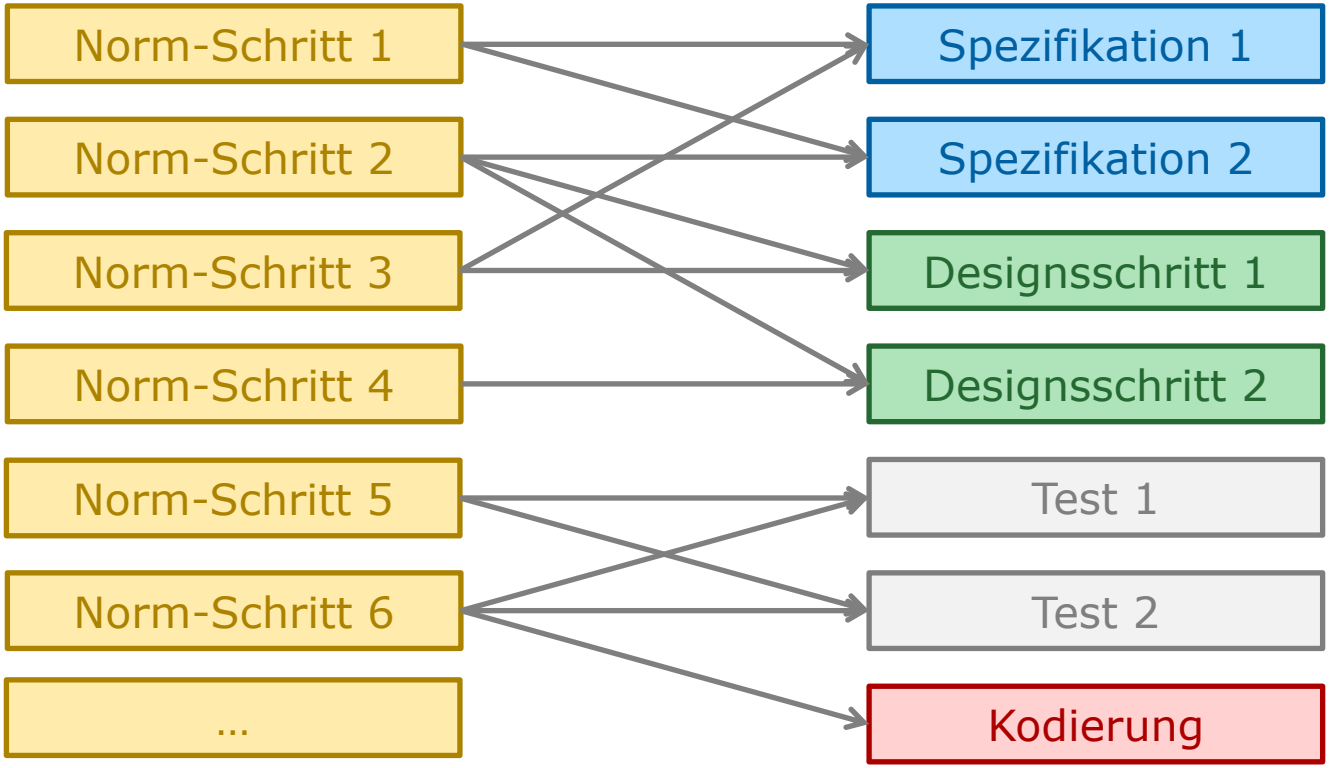
- Problemstellung
- **Lösungsansatz**
- Konkrete Vorgehensweise
- Ergebnis

## Definition eines generischen Entwicklungsprozesses

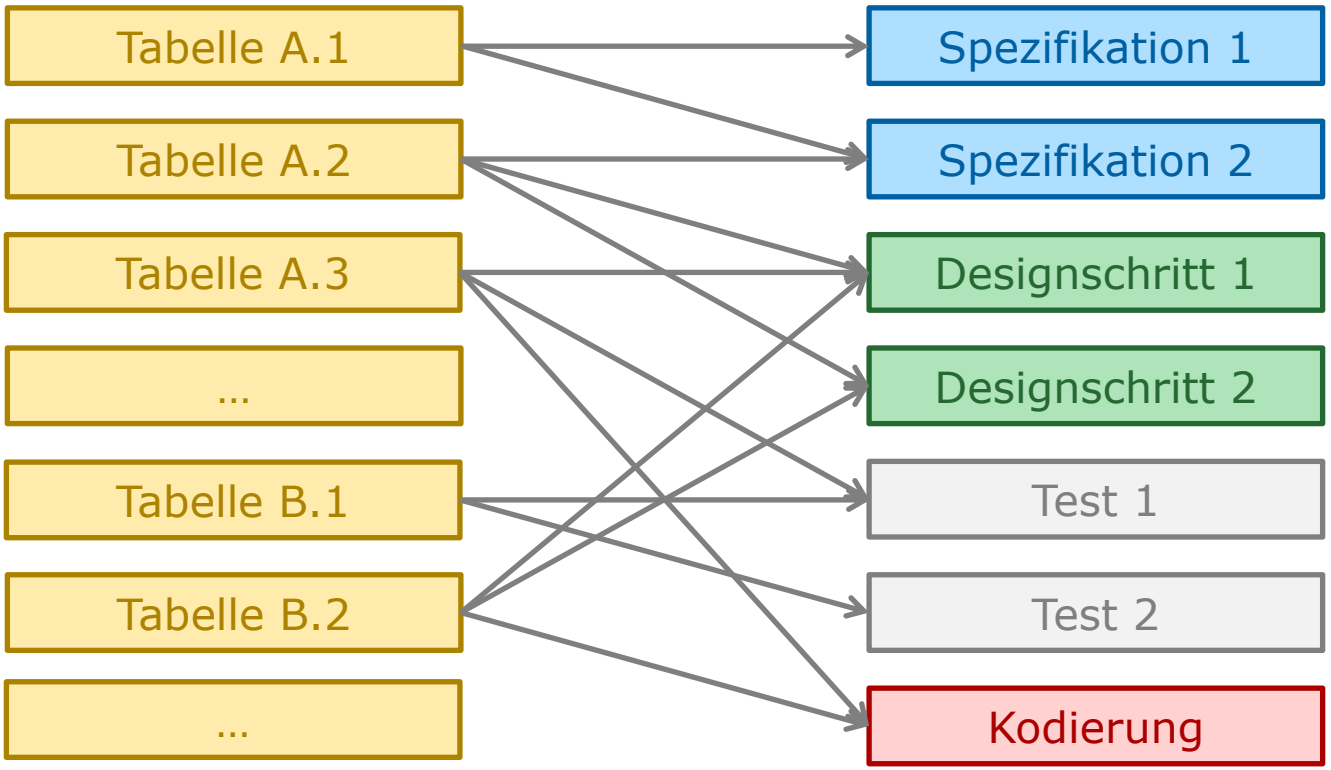


embeX
Your embedded experts

## Abbildung des Prozesses der IEC 61508 auf den eigenen Prozess



## Zuordnung der Tabellen aus IEC 61508-3 zu den eigenen Prozessschritten



embeX
Your embedded experts

## Definition einer generischen Vorgehensweise

- Auswahl der in den einzelnen Entwicklungs- und Verifikationsschritten geeigneten Maßnahmen und Methoden, ggf. mit
  - einer genaueren Beschreibung der geplanten Vorgehensweise
  - einem Verweis auf firmeninterne Vorlagen und Anweisungen
- Abstimmung mit einer Prüfstelle

➢ Einige offene Punkte, die projektspezifisch entschieden werden müssen

## Definition der projektspezifischen Vorgehensweise

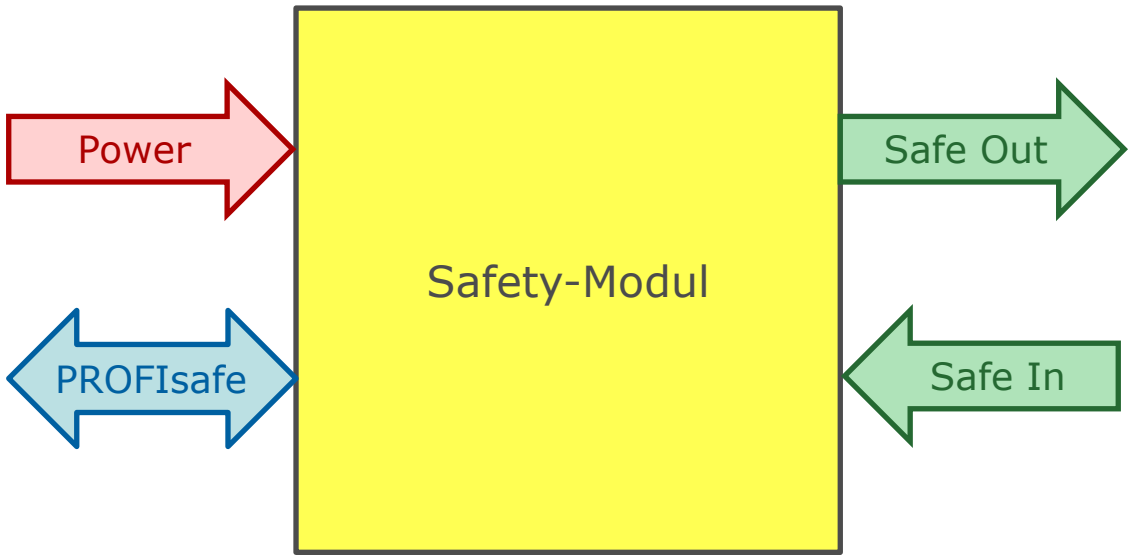
- Entscheidung über die offenen Punkte
- Abstimmung mit dem Prüfer auf Basis des generischen Prozesses

➢ eine präzise Zusammenfassung der normativen Anforderungen für den einzelnen Entwickler, Tester, Reviewer im Projekt

- Problemstellung
- Lösungsansatz
- **Konkrete Vorgehensweise**
- Ergebnis

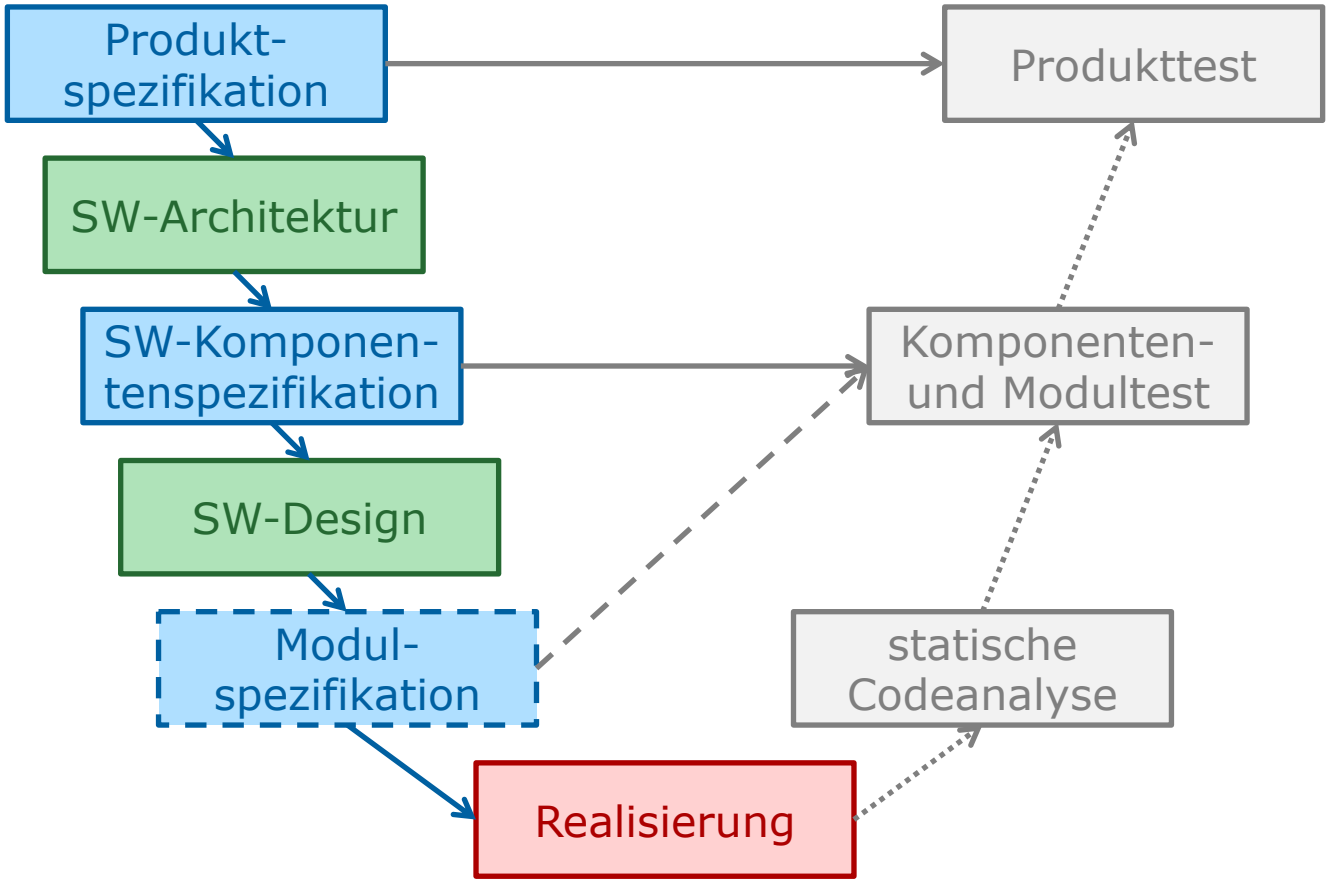
embeX
Your embedded experts

## Typischer Entwicklungsgegenstand



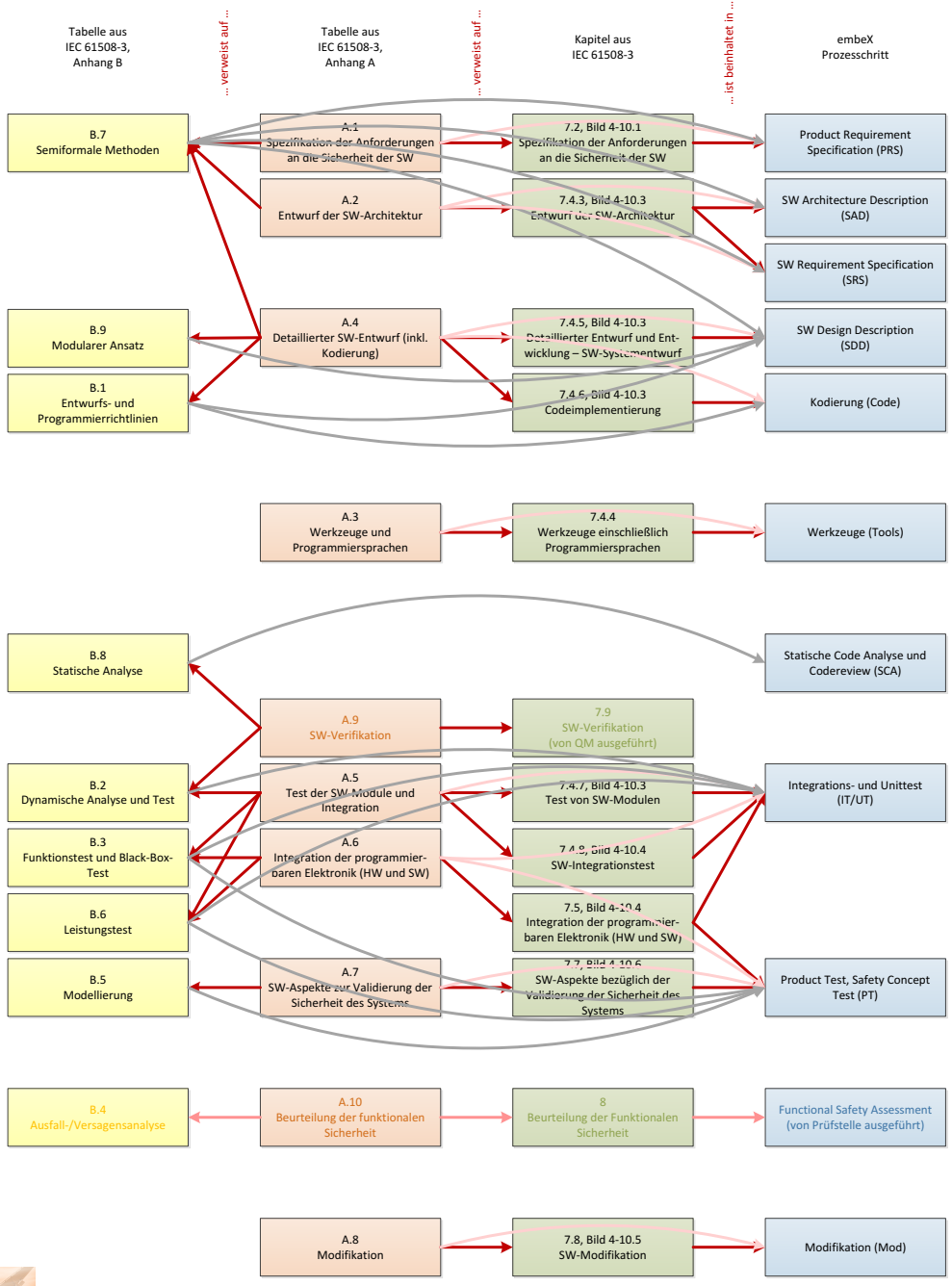
embeX
Your embedded experts

## Definition eines generischen Entwicklungsprozesses (Software)



# Vorgehensweise

**Abbildung der einzelnen Prozessschritte auf die IEC 61508-3**



embeX
Your embedded experts

# Definition der generischen Vorgehensweise (Ausschnitt)

Beispiel:
Komponenten- und Modultests (31 Einträge)

**Tabelle A.6 – Integration der programmierbaren Elektronik (Hardware und Software) (siehe 7.5)**

| | Verfahren/Maßnahme * | siehe | SIL 1 | SIL 2 | SIL 3 | generisch | Kommentar generisch |
|---|---|---|---|---|---|---|---|
| 1 | Funktionstest und Black-Box-Test | B.5.1<br>B.5.2<br>Tabelle B.3 | ++ | ++ | ++ | siehe Tabelle B.3 | |
| 2 | Leistungstest | Tabelle B.6 | + | + | ++ | siehe Tabelle B.6 | |
| 3 | Vorwärtsverfolgbarkeit zwischen den Anforderungen an den System- und Softwareentwurf der Hardware-/Software-integration und den Spezifikationen der Hardware-/Softwareintegrationstests | C.2.11 | + | + | ++ | teilweise | Die Integrationstests basieren auf der SRS und müssen deren Anforderungen vollständig abdecken.<br>Unittests beruhen auf den Funktionsbeschreibungen der SDD, werden aber nur bei Bedarf spezifiziert und durchgeführt. |

*ANMERKUNG 1 Die Integration der programmierbaren Elektronik ist eine Verifikationstätigkeit (siehe Tabelle A.9).*
*ANMERKUNG 2 Siehe Tabelle C.6.*
*ANMERKUNG 3 Die Verweisungen (die informativ, nicht normativ sind) „Bxxx", „Cxxx" in Spalte 3 (siehe) weisen auf detaillierte Beschreibungen der Verfahren/Maßnahmen in den Anhängen B und C der IEC 61508-7 hin.*
** Es müssen dem Sicherheits-Integritätslevel angemessene Verfahren/Maßnahmen ausgewählt werden.*

**Tabelle B.2 – Dynamische Analyse und Test (Verweisung aus den Tabellen A.5 und A.9)**

| | Verfahren/Maßnahme * | siehe | SIL 1 | SIL 2 | SIL 3 | generisch | Kommentar generisch |
|---|---|---|---|---|---|---|---|
| 1 | Durchführung von Testfällen nach einer Grenzwertanalyse | C.5.4 | + | ++ | ++ | siehe Tab. B.4, Pkt. 4 | |
| 2 | Durchführung von Testfällen aus der Fehlererwartung | C.5.5 | + | + | + | n/a | Tests nach Fehlererwartung auf Basis der Intuition des Prüfers sind erwünscht, lassen sich aber per Prozessdefinition nicht durchsetzen. |
| 3 | Durchführung von Testfällen nach Fehlereinpflanzung | C.5.6 | o | + | + | nein | Aufwand steht nicht in angemessenem Verhältnis zum Ergebnis. |
| 4 | Testfallausführung durch modellbasierte Testfallgenerierung | C.5.27 | + | + | ++ | projekt-abhängig | Hardwarenahe Software lässt sich i.d.R. nicht sinnvoll modellieren. Ggf. wird dieses Verfahren angewandt, falls die Software<br>- komplexe Zustandsmaschinen oder<br>- komplexe Algorithmik<br>enthält. |
| 5 | Modellierung der Leistungsfähigkeit | C.5.20 | + | + | + | projekt-abhängig | nur bei nicht-deterministischem (ereignisgesteuertem) Scheduling-Modell |
| 6 | Äquivalenzklassen und Test mit Partitionen des eingeschränktem Eingangsbereichs | C.5.7 | + | + | + | siehe Tab. B.4, Pkt. 4 | |
| 7a | Strukturabhängige Tests mit einer Testabdeckung (Eingangspunkte) 100 %** | C.5.8 | ++ | ++ | ++ | ja | Die Testabdeckung wird zunächst bei der Durchführung der Softwarekomponententests überprüft. Sofern erforderlich, werden zusätzlich Tests einzelner Funktionen (Unit-Tests mit Stubs für Hardwarezugriffe und Aufrufe weiterer Funktionen) durchgeführt. Sofern die Testabdeckung "Bedingungen" (7d) nicht sinnvoll erreicht werden kann, so wird min. die besonders empfohlene Testabdeckung (++) erfüllt oder eine entsprechende Begründung dokumentiert (z.B.: Konstrukte defensiver Programmierung). |
| 7b | Strukturabhängige Tests mit einer Testabdeckung (Anweisungen) 100 %** | C.5.8 | + | ++ | ++ | ja | |
| 7c | Strukturabhängige Tests mit einer Testabdeckung (Verzweigungen) 100 %** | C.5.8 | + | + | ++ | ja | |
| 7d | Strukturabhängige Tests mit einer Testabdeckung (Bedingungen) 100 %** | C.5.8 | + | + | + | ja | |

embeX
Your embedded experts

# Definition der projektspezifischen Vorgehensweise (Ausschnitt)

| | | Verfahren/Maßnahme * | siehe | SIL 1 | SIL 2 | SIL 3 | generisch | Kommentar generisch | angewandt | Kommentar projektspezifisch |
|---|---|---|---|---|---|---|---|---|---|---|
| | 3 | Datenaufzeichnung und Analyse | C.5.2 | ++ | ++ | ++ | ja | - Test werden vollständig dokumentiert<br>- Entscheidungen und ihre Grundlagen werden in Form von Protokollen dokumentiert<br>- Probleme und ihre Lösungen werden mit Hilfe eines Bug-Trackings-Tools (Mantis) dokumentiert | ja | |
| | 7 | Schnittstellentest | C.5.3 | + | + | ++ | ja | Die Schnittstellen der einzelnen Softwarekomponenten werden jeweils (so weit möglich) vollständig getestet. | ja | |
| | 8 | Testmanagement und Automatisierungswerkzeuge | C.4.7 | + | ++ | ++ | ja | Testtreiber für die Komponententests und erwartete Testergebnisse werden toolgesetützt erstellt (Tessy, embeX-CI) | ja | |
| | 9 | Vorwärtsverfolgbarkeit zwischen der Spezifikation des Softwareentwurfs und den Spezifikationen des Modul- und Integrationstests. | C.2.11 | + | + | ++ | projekt-abhängig | Die Integrationstests basieren auf der SRS und müssen deren Anforderungen vollständig abdecken.<br>Unittests beruhen auf den Funktionsbeschreibungen der SDD, werden aber nur bei Bedarf spezifiziert und durchgeführt, wenn eine vollständige Codeabdeckung durch die Integrationstests alleine nicht erzielt werden konnte..<br>Projektabhängig kann auch fesgelegt werden, dass alle Softwarefunktionen einem vollständigen Unittest unterzogen werden. | teilweise | Unittests werden nur bei Bedarf durchgeführt. |
| | 3 | Vorwärtsverfolgbarkeit zwischen den Anforderungen an den System- und Softwareentwurf der Hardware-/Software-integration und den Spezifikationen der Hardware-/Softwareintegrationstests | C.2.11 | + | + | ++ | teilweise | Die Integrationstests basieren auf der SRS und müssen deren Anforderungen vollständig abdecken.<br>Unittests beruhen auf den Funktionsbeschreibungen der SDD, werden aber nur bei Bedarf spezifiziert und durchgeführt. | teilweise | |
| | 4 | Testfallausführung durch modellbasierte Testfallgenerierung | C.5.27 | + | + | ++ | projekt-abhängig | Hardwarenahe Software lässt sich i.d.R. nicht sinnvoll modellieren. Ggf. wird dieses Verfahren angewandt, falls die Software<br>- komplexe Zustandsmaschinen oder<br>- komplexe Algorithmik<br>enthält. | nein | wg. geringer Komplexität |
| | 5 | Modellierung der Leistungsfähigkeit | C.5.20 | + | + | + | projekt-abhängig | nur bei nicht-deterministischem (ereignisgesteuertem) Scheduling-Modell | nein | wg. deterministischem Scheduling-Modell (Zeitscheiben-Scheduler) |
| | 7a | Strukturabhängige Tests mit einer Testabdeckung (Eingangspunkte) 100 %** | C.5.8 | ++ | ++ | ++ | ja | Die Testabdeckung wird zunächst bei der Durchführung der Softwarekomponententests überprüft. Sofern erforderlich, werden zusätzlich Tests einzelner Funktionen (Unit-Tests mit Stubs für Hardwarezugriffe und Aufrufe weiterer Funktionen) durchgeführt. Sofern die Testabdeckung "Bedingungen" (7d) nicht sinnvoll erreicht werden kann, so wird min. die besonders empfohlene Testabdeckung (++) erfüllt oder eine entsprechende Begründung dokumentiert (z.B.: Konstrukte defensiver Programmierung). | ja | |
| | 7b | Strukturabhängige Tests mit einer Testabdeckung (Anweisungen) 100 %** | C.5.8 | + | ++ | ++ | ja | | ja | |
| | 7c | Strukturabhängige Tests mit einer Testabdeckung (Verzweigungen) 100 %** | C.5.8 | + | + | ++ | ja | | ja | |
| | 7d | Strukturabhängige Tests mit einer Testabdeckung (Bedingungen) 100 %** | C.5.8 | + | + | + | ja | | ja | |

embeX
Your embedded experts

# Projektspezifische Vorgehensweise (gefiltert, komplett)

| | | Verfahren/Maßnahme * | siehe | SIL 1 | SIL 2 | SIL 3 | generisch | Kommentar generisch | angewandt | Kommentar projektspezifisch |
|---|---|---|---|---|---|---|---|---|---|---|
| | 3 | Datenaufzeichnung und Analyse | C.5.2 | ++ | ++ | ++ | ja | - Test werden vollständig dokumentiert<br>- Entscheidungen und ihre Grundlagen werden in Form von Protokollen dokumentiert<br>- Probleme und ihre Lösungen werden mit Hilfe eines Bug-Trackings-Tools (Mantis) dokumentiert | ja | |
| | 7 | Schnittstellentest | C.5.3 | + | + | ++ | ja | Die Schnittstellen der einzelnen Softwarekomponenten werden jeweils (so weit möglich) vollständig getestet. | ja | |
| | 8 | Testmanagement und Automatisierungswerkzeuge | C.4.7 | + | ++ | ++ | ja | Testtreiber für die Komponententests und erwartete Testergebnisse werden toolgesetützt erstellt (Tessy, embeX-CI) | ja | |
| | 9 | Vorwärtsverfolgbarkeit zwischen der Spezifikation des Softwareentwurfs und den Spezifikationen des Modul- und Integrationstests. | C.2.11 | + | + | ++ | projekt-abhängig | Die Integrationstests basieren auf der SRS und müssen deren Anforderungen vollständig abdecken.<br>Unittests beruhen auf den Funktionsbeschreibungen der SDD, werden aber nur bei Bedarf spezifiziert und durchgeführt, wenn eine vollständige Codeabdeckung durch die Integrationstests alleine nicht erzielt werden konnte..<br>Projektabhängig kann auch fesgelegt werden, dass alle Softwarefunktionen einem vollständigen Unittest unterzogen werden. | teilweise | Unittests werden nur bei Bedarf durchgeführt. |
| | 3 | Vorwärtsverfolgbarkeit zwischen den Anforderungen an den System- und Softwareentwurf der Hardware-/Software-integration und den Spezifikationen der Hardware-/Softwareintegrationstests | C.2.11 | + | + | ++ | teilweise | Die Integrationstests basieren auf der SRS und müssen deren Anforderungen vollständig abdecken.<br>Unittests beruhen auf den Funktionsbeschreibungen der SDD, werden aber nur bei Bedarf spezifiziert und durchgeführt. | teilweise | |
| | 7a | Strukturabhängige Tests mit einer Testabdeckung (Eingangspunkte) 100 %** | C.5.8 | ++ | ++ | ++ | ja | Die Testabdeckung wird zunächst bei der Durchführung der Softwarekomponententests überprüft. Sofern erforderlich, werden zusätzlich Tests einzelner Funktionen (Unit-Tests mit Stubs für Hardwarezugriffe und Aufrufe weiterer Funktionen) durchgeführt. Sofern die Testabdeckung "Bedingungen" (7d) nicht sinnvoll erreicht werden kann, so wird min. die besonders empfohlene Testabdeckung (++) erfüllt oder eine entsprechende Begründung dokumentiert (z.B.: Konstrukte defensiver Programmierung). | ja | |
| | 7b | Strukturabhängige Tests mit einer Testabdeckung (Anweisungen) 100 %** | C.5.8 | + | ++ | ++ | ja | | ja | |
| | 7c | Strukturabhängige Tests mit einer Testabdeckung (Verzweigungen) 100 %** | C.5.8 | + | + | ++ | ja | | ja | |
| | 7d | Strukturabhängige Tests mit einer Testabdeckung (Bedingungen) 100 %** | C.5.8 | + | + | + | ja | | ja | |
| | 4 | Äquivalenzklassen und Test mit eingeschränktem Eingangsbereich einschließlich Grenzwertanalyse | C.5.7<br>C.5.4 | + | ++ | ++ | ja | Bestimmung von Grenzwerten / ungültigen Parameterbereichen bei der Testfallerstellung für Komponenten. Typischerweise ergeben sich Äquivalenzklassen der Testverktoren schon durch die angestrebte Testabdeckung. | ja | |
| | 5 | Prozess-Simulation | C.5.18 | + | + | + | ja | bei den Softwarekomponententests wird die Umgebung (= andere Komponenten) durch die Testtreiber simuliert. | ja | |
| | 2 | Reaktionszeiten und Speicherbeschränkungen | C.5.22 | ++ | ++ | ++ | ja | 1. max. Ausführungszeiten einzelner Tasks werden in der Spezifikationen gefordert und in den Softwarekomponententests überprüft.<br>2. Es erfolgt keine dynamische Speicherverwaltung; insofern sind Tests der Speicherbeschränkungen nicht erforderlich | ja | |

(9 Einträge)

- Problemstellung
- Lösungsansatz
- Konkrete Vorgehensweise
- **Ergebnis**

# Ergebnis

## Was haben wir erreicht?

- Definition einer einheitlichen Vorgehensweise für (Software-)Entwicklung und –Verifikation in Safety-Projekten.
- Die Konformität mit den Anforderungen der IEC 61508 lässt sich Schritt für Schritt nachweisen.
- Die Vorgehensweise lässt sich leicht an die Erfordernisse eines konkreten Entwicklungsprojekts anpassen.
- Der einzelne Entwickler/Tester/Reviewer erhält eine überschaubare und selbsterklärende Übersicht der in seinem Arbeitspaket umzusetzenden Anforderungen der Norm.

- Dieser Ansatz lässt sich auch für den Entwicklungsprozess **Ihrer Firma** umsetzen.

Vielen Dank für Ihre Aufmerksamkeit!

**Kontakt**

Dr. Martin Lange
Leiter Fachbereich Funktionale Sicherheit

Fon: +49 (0) 761 479799 14
Mobil: +49 (0) 151 42232538

m.lange@embex.de

**embeX GmbH**

Heinrich-von-Stephan-Straße 23
D-79100 Freiburg

Fon: +49 (0) 761 479799 0
Fax: +49 (0) 761 479799 99

www.embex.de
info@embex.de