Zuverlässigkeits-bewertung unter neuen Anforderungen an Sicherheitsleittechnik in Kernkraftwerken: Analysen der Anwendungspraxis

GRS - 441

Gesellschaft für Anlagen-
und Reaktorsicherheit
(GRS) gGmbH

Zuverlässigkeits-bewertung unter neuen Anforderungen an Sicherheitsleittechnik in Kernkraftwerken: Analysen der Anwendungspraxis

Manuela Jopen
Claudia Quester
Sarah Römer
Dagmar Sommer
Jan Stiller
Birte Ulrich

Oktober 2016

Anmerkung:
Das diesem Bericht zugrunde liegende FE-Vorhaben wurde mit Mitteln des Bundesministeriums für Umwelt, Naturschutz, Bau und Reaktorsicherheit (BMUB) unter dem Kennzeichen 3613R01322 durchgeführt.

GRS - 441
ISBN 978-3-946607-23-6

Deskriptoren:

Sicherheitsleittechnik, softwarebasierte Leittechnik, digitale Leittechnik, nichtnukleare Industrie, Bewertungsmethoden, Software-Metriken

Kurzfassung

In vielen deutschen Kernkraftwerken sind inzwischen rechnerbasierte oder programmierbare leittechnische Einrichtungen im Bereich der betrieblichen Leittechnik und bei Regelungen und Begrenzungen eingesetzt. Da angesichts des Ausstiegsbeschlusses absehbar ist, dass für deutsche Kernkraftwerke keine kerntechnikspezifischen Austauschkomponenten mehr entwickelt werden, ist sogar mit einem verstärkten Einsatz rechnerbasierter und programmierbarer leittechnischer Einrichtungen zu rechnen. Dieser verstärkte Einsatz kann sich dann auch auf sicherheitstechnisch wichtige leittechnische Einrichtungen ausweiten.

Vor allem bei Umrüstungsmaßnahmen in deutschen Kernkraftwerken an sicherheitstechnisch wichtigen leittechnischen Einrichtungen muss in Betracht gezogen werden, dass ein Teil der leittechnischen Funktionen noch weit über das Jahr 2022 hinaus, d. h. auch bei langfristigem Nichtleistungsbetrieb nach der Abschaltung der Kernkraftwerke sowie während des Stilllegungsbetriebs im Rahmen des Rückbaus, benötigt werden. Problematisch an dieser Entwicklung ist, dass zum Zeitpunkt der Antragsstellung keine speziell für rechnerbasierte oder programmierbare leittechnische Einrichtungen formulierte Anforderungen im deutschen kerntechnischen Regelwerk zur Verfügung standen. Es existierten jedoch bereits DIN-Normen, in denen Regelungen und Anforderungen rechnerbasierten und programmierbare Einrichtungen gestellt werden. Mittlerweile wurden darüber hinaus Anforderungen an rechnerbasierte und programmierbare Einrichtungen im Rahmen der Überarbeitung des kerntechnischen Regelwerks und der Sicherheitsanforderungen an Kernkraftwerke aufgenommen. Die RSK ist jedoch der Auffassung, dass zusätzliche, über den zum Zeitpunkt der Stellungnahme aktuellen Stand der Normen hinausgehende Verfahren erforderlich sind. Daher ist eine eingehende Untersuchung von internationalen nuklearen sowie nicht-nuklearen Regelwerken, Standards und Normen notwendig, um eine mögliche Übertragbarkeit in das deutsche kerntechnische Regelwerk zu diskutieren.

Insgesamt liefert das Vorhaben somit einen Beitrag zur Weiterentwicklung der Bewertungsmaßstäbe rechnerbasierter oder programmierbarer Baugruppen und Systeme für eine Anwendung im nationalen, nuklearen Bereich. Im Fokus dieser Weiterentwicklung stehen neben konkreten Anforderungen an die Softwareentwicklung und den Softwarelebenszyklus die Prozesse und Methoden der Qualifizierung und der Bewertung der Systemzuverlässigkeit sowie die Anwendbarkeit von Software-Metriken.

Inhaltsverzeichnis

1 Einleitung

In vielen deutschen Kernkraftwerken sind inzwischen rechnerbasierte oder programmierbare leittechnische Einrichtungen im Bereich der betrieblichen Leittechnik und bei Regelungen und Begrenzungen eingesetzt. Je nach Anlage und System unterscheidet sich der Umfang des rechnerbasierten oder programmierbaren Anteils jedoch signifikant. Der Anteil kann sich auf einzelne Betriebsmittel (z.B. Messumformer) bis hin zu komplett rechnerbasierten oder programmierbaren Regelungs- oder Begrenzungssysteme erstrecken. Einsatzbereiche rechnerbasierter oder programmierbarer Technik in leittechnischen Systemen deutscher Kernkraftwerke sind beispielsweise

- Reaktorregelungen in den Kernkraftwerken Emsland und Brokdorf,
- Reaktorleistungsbegrenzungen und -regelungen in den Kernkraftwerken Philippsburg-1 und 2, Unterweser, Isar-1 und Neckarwestheim-1,
- USUS-System im Kernkraftwerk Philippsburg-1,
- Teile der Turbinensteuerung im Kernkraftwerk Neckarwestheim-2 und
- Leistungsverteilungsüberwachung im Kernkraftwerk Grohnde.

Im internationalen Umfeld sind Reaktorschutzsysteme in unterschiedlichem Umfang mit rechnerbasierten oder programmierbaren Systemen geplant oder bereits umgesetzt. Hierzu zählen beispielsweise

- In Umsetzung befindliche Umrüstvorhaben in der finnischen Anlage Loviisa (Block 1 und 2),
- Abgeschlossene Umrüstungsmaßnahmen der leittechnischen Einrichtungen im Reaktorschutz im schweizerischen Kernkraftwerk Beznau, in den tschechischen Anlagen Temelín (Block 1 und 2) und Dukovany-3, in der schwedischen Anlage Ringhals-1 sowie in der US-amerikanischen Anlage Oconee (Block 1, 2, und 3),
- Einsatz einer FPGA-basierten Leittechnikplattform in den ukrainischen Anlagen Rovno 1-4, Khmelnitsky 1 und 2, Südukraine 1 und 2, Kozloduy 5 und 6 sowie Zaporozhye 1-3,
- Einsatz vollständig rechnerbasierter oder programmierbarer Leittechnik in der kanadischen Anlage Darlington (Blöcke 1 und 2), der französischen Anlagen

Chooz-B (Block 1 und 2) und Civeaux (Block 1 und 2) sowie der chinesischen Anlage Tianwan (Block 1 und 2).

Vor allem bei Umrüstungsmaßnahmen in deutschen Kernkraftwerken an sicherheitstechnisch wichtigen leittechnischen Einrichtungen muss in Betracht gezogen werden, dass ein Teil der leittechnischen Funktionen noch weit über das Jahr 2022 hinaus, d. h. auch bei langfristigem Nichtleistungsbetrieb nach der Abschaltung der Kernkraftwerke sowie während des Stilllegungsbetriebs im Rahmen des Rückbaus, benötigt werden. Es ist daher im Rahmen der Restlaufzeit der Kernkraftwerke und darüber hinaus mit Erneuerungen an sicherheitstechnisch wichtigen leittechnischen Einrichtungen und elektronischen Baugruppen (z.B. Einsatz von FPGAs auf analogen und digitalen E/A-Baugruppen oder Einsatz von Messumformern mit FPGAs) zu rechnen. Da angesichts des Ausstiegsbeschlusses absehbar ist, dass für deutsche Kernkraftwerke keine kerntechnikspezifischen Austauschkomponenten mehr entwickelt werden, ist sogar mit einem verstärkten Einsatz rechnerbasierter und programmierbarer leittechnischer Einrichtungen zu rechnen. Ebenfalls denkbar ist, dass dabei leittechnische Komponenten zum Einsatz kommen könnten, die für andere, nicht-nukleare Bereiche und für andere Einsatzzwecke entwickelt wurden.

Problematisch an dieser Entwicklung ist, dass zum Zeitpunkt der Antragsstellung des Vorhabens nur wenig konkrete speziell für rechnerbasierte oder programmierbare leittechnische Einrichtungen formulierte Anforderungen im deutschen kerntechnischen Regelwerk zur Verfügung standen. Es existierten jedoch bereits DIN-Normen, in denen Anforderungen an rechnerbasierte und programmierbare leittechnische Einrichtungen gestellt werden. Mittlerweile wurden darüber hinaus Anforderungen an rechnerbasierte und programmierbare Einrichtungen im Rahmen der Überarbeitung des kerntechnischen Regelwerks und der Sicherheitsanforderungen an Kernkraftwerke aufgenommen. Die RSK ist jedoch der Auffassung, dass zusätzliche, über den zum Zeitpunkt der Stellungnahme aktuellen Stand der Normen hinausgehende Verfahren erforderlich sind, um eine optimale und nachweisbare Auslegung gegen gemeinsam verursachte Ausfälle auf Systemebene zu gewährleisten /RSK 11/. Daher ist eine eingehende Untersuchung von internationalen nuklearen sowie nicht-nuklearen Regelwerken, Standards und Normen notwendig, um eine mögliche Übertragbarkeit in das deutsche kerntechnische Regelwerk zu diskutieren. Insgesamt liefert das Vorhaben somit einen Beitrag zur Weiterentwicklung der Bewertungsmaßstäbe rechnerbasierter oder programmierbarer Baugruppen und Systeme für eine Anwendung im nationalen, nuklearen Bereich. Im Fokus dieser

Weiterentwicklung stehen neben konkreten Anforderungen an die Softwareentwicklung und den Softwarelebenszyklus die Prozesse und Methoden der Qualifizierung und der Bewertung der Systemzuverlässigkeit sowie die Anwendbarkeit von Software-Metriken.

2 Stand von Wissenschaft und Technik

2.1 Definition verwendeter Begriffe

In diesem Abschnitt werden zunächst die in den betrachteten Normen, Regelwerken und Standards verwendeten Begriffe mit Bezug zu rechnerbasierten und programmierbaren Einrichtungen definiert. Sie fußen auf unterschiedlichsten Quellen aus gänzlich verschiedenen Gebieten. Teilweise werden daher die Begriffe in verschiedenen mehr oder weniger abweichenden Bedeutungen verwendet. Im Folgenden werden daher für jeden Begriff die Definitionen angegeben, welche für die GRS maßgebend sind und in diesem Dokument gelten, solange keine explizite oder implizite anderslautende Definition im Text angegeben ist. In Anhang 4.3.1 sind die jeweils originalen Definitionen aus den für dieses Dokument relevanten Normen und Standards aufgeführt.

- **Anforderungen an Software (engl.: requirements)**

Vorgaben bezüglich der Beschaffenheit und den Fähigkeiten einer Software, durch die diese Software Probleme lösen bzw. Ziele erreichen kann. Sind die Anforderungen an eine Software durch Vereinbarungen, Standards (Normen), Spezifikationen oder andere formale Dokumente festgelegt, so spricht man von spezifizierten Anforderungen an diese Software.

- **Anwendungsfunktion (engl.: application function)**

Funktion eines leittechnischen Systems, die eine Aufgabe in Verbindung mit dem gesteuerten Prozess und nicht mit dem Funktionieren des Systems selbst erfüllt.

- **Anwendungssoftware (engl.: application software)**

Teil der Software eines leittechnischen Systems, durch den Anwendungsfunktionen realisiert werden.

- **Betriebssoftware (engl.: operational system software)**

Teil der Systemsoftware, dessen ausführbarer Code während des Systembetriebs auf dem Zielsystem läuft.

- **Computerprogramm (engl.: computer program)**

Ein Computerprogramm (oder kurz Programm) ist eine Kombination von Anweisungen und Datendefinitionen, die es der Hardware ermöglicht, bestimmte Funktionen bzw. Aufgaben oder Probleme zu bearbeiten oder zu lösen.

Die Darstellung bzw. Beschreibung eines Programms in einer bestimmten Programmiersprache wird häufig als (Quell-)Code bezeichnet.

- **Computersystem (engl.: computer system)**

Ein System aus einem oder mehreren Computern und der dazugehörigen Software.

- **Daten (engl.: data)**

Darstellung von Informationen oder Anweisungen in einer für die Übertragung, Auswertung oder Verarbeitung mittels Rechnern geeigneten Weise.

- **Dissimilarität (engl.: dissimilarity)**

Auf rechnerbasierte oder programmierbare Systeme bezogener Unterbegriff der Diversität.

- **Diversität (engl.: diversity)**

Diversität ist das Vorhandensein von zwei oder mehreren unterschiedlichen Verfahren oder Mitteln, um ein bestimmtes Ziel zu erreichen. Diversität wird insbesondere als Schutzmaßnahme gegen ein Versagen aus gemeinsamer Ursache eingesetzt.

- **Software-Diversität (engl.: software diversity)**

Diversität von Software bezeichnet den Gebrauch von zwei oder mehreren unterschiedlichen Programmen, um kritische Funktionen auszuführen. Ein Höchstmaß an Diversität wird durch einen funktionalen Unterschied erreicht. Eine weitere Möglichkeit ist die Verwendung unterschiedlicher Programme von unabhängigen Entwicklungs-Teams unter Verwendung unterschiedlicher Softwaredesigns und Programmiersprachen. Zusätzlich sollte die Software mit unterschiedlicher Hardware (z.B. Prozessoren, Speichermedien) zum Einsatz kommen.

- **E/E/PE-System (engl.: E/E/PE system für electrical/electronic/programmable electronic system)**

Ein Steuerungs-, Regelungs-, Begrenzungs-, Schutz- oder Überwachungssystem aus einem oder mehreren elektrischen/elektronischen/programmierbaren elektronischen Geräten, inklusive aller Bestandteile des Systems, wie Stromversorgung, Sensoren und andere Eingabegeräten, Kommunikationseinrichtungen und -wege, Aktoren und andere Ausgabegeräten.

- **Fehler (engl.: fault)**

Mangel an einer Hardware-, Software- oder Systemkomponente.
Auf eine genauere Diskussion des Begriffes Softwarefehler wird in Absatz 2.2 „Softwarefehler“ eingegangen.

- **Fehlertoleranz (engl.: fault tolerance)**

Fähigkeit eines Systems, trotz des Vorhandenseins einer begrenzten Anzahl von Hardware- oder Softwarefehlern, kontinuierlich korrekt weiterzuarbeiten.

- **Formale Methoden (engl.: formal methods)**

Die Anwendung eines mathematischen Verfahrens für die Überprüfung der Einhaltung des spezifizierten Softwaredesigns.

- **Funktionale Diversität (engl.: functional diversity)**

Anwendung der Diversität auf der Funktionsebene (z. B. Ableitung eines Abschaltkriteriums sowohl aus Druck- als auch Temperaturgrenzwerten).

- **Funktionale Validierung (engl.: functional validation)**

Prüfung der Übereinstimmung von Spezifikationen der Anwendungsfunktionen mit den originären funktionalen und Leistungsfähigkeitsanforderungen der Anlage. Die funktionale Validierung ist als Ergänzung zur Systemvalidierung zu sehen, bei der die Übereinstimmung des Systems mit den Spezifikationen der Anwendungsfunktionen geprüft wird.

- **Hardware (engl.: (computer) hardware)**

Physische (mechanische und elektronische) Komponenten eines Rechners.

- **Implementierung (engl.: implementation)**

Die Implementierung ist die Umsetzung eines Softwaredesigns in ein Computersystem unter Berücksichtigung der Rahmenbedingungen, Regeln und Zielvorgaben, also im Sinne einer Spezifikation. Häufig wird der Begriff auch für die Installation und Anpassung einer Software-Lösung oder den Übergang einer Software in den Betrieb benutzt.

- **Initialisierung (engl.: initialization)**

Setzen von Zählern, Schaltern, Adressen oder Inhalten von Speichern auf Null oder andere Startwerte, zu Beginn oder an bestimmten Punkten des Betriebs eines Rechnerprogramms.

- **Integration (engl.: integration)**

Die Kombination von Softwarebestandteilen, Hardwarebestandteilen oder beidem zu einem Gesamtsystem.

- **Integrationstests (engl.: integration tests)**

Tests, die während des Integrationsvorgangs von Hardware/Software vor Validierung des Rechner-Systems durchgeführt werden, um die Kompatibilität der Software und Hardware des Rechners zu verifizieren.

- **Kategorie einer leittechnischen Funktion (engl.: category of an I&C function)**

Eine von drei möglichen Zuordnungen (A, B, C) von leittechnischen Funktionen, die die Sicherheitsrelevanz der durchzuführenden Funktionen beschreiben. Wenn die Funktion für die Sicherheit nicht signifikant ist, erfolgt keine Sicherheitszuordnung (Kategorisierung).

- **Klassifizierung (engl.: classification)**

Die Einordnung (einer Software) in einer Kategorie

- **Komplexität (engl.: complexity)**

Objektiver Schwierigkeitsgrad der Verständlichkeit und Verifizierbarkeit eines Systems oder einer Komponente aufgrund von Auslegung, Realisierung oder Verhalten.

- **Konzept der gestaffelten Schutzmaßnahmen (engl.: defence in depth)**

Anwendung von mehr als einer Schutzmaßnahme zum Erreichen eines gegebenen Sicherheitsziels, sodass dieses erreicht wird, auch wenn eine der Schutzmaßnahmen versagt.

- **Metrik (engl.: metric)**

Eine Softwaremetrik, oder kurz Metrik, ist eine meist mathematische Funktion, die eine Eigenschaft von Software in einen Zahlenwert, auch Maßzahl genannt, abbildet. Hierdurch werden formale Vergleichs- und Bewertungsmöglichkeiten geschaffen.

- **PE-System (engl.: programmable electronic system)**

Ein Steuerungs-, Regelungs-, Begrenzungs-, Schutz- oder Überwachungssystem aus einem oder mehreren programmierbaren elektronischen Geräten, inklusive aller Bestandteile des Systems, wie Stromversorgung, Sensoren und andere Eingabegeräten, Kommunikationseinrichtungen und -wege, Aktoren und andere Ausgabegeräten.

- **Programmierbares, rechnerbasiertes System (engl.: computer-based system)**

Leittechnisches System, dessen Funktionen meistens von Mikroprozessoren, programmierten elektronischen Einheiten oder Rechnern abhängen oder durch die Verwendung derartiger Gerätschaften zur Gänze durchgeführt werden. Synonym werden auch die Begriffe digitales System, softwarebasiertes System, programmierbares System benutzt.

- **Programmiersprache (engl.: programming language)**

Eine künstliche Sprache, in der Computerprogramme als sogenannter (Quell-)Code geschrieben werden. Der Code enthält Anweisungen und Datendefinitionen, die von einem Rechner ausgeführt werden können.

- **Qualifizierung (engl.: qualification)**

Der Nachweisprozess, dass eine Einheit (beispielsweise ein Computersystem) die Anforderungen erfüllt bzw. für den betrieblichen Einsatz geeignet ist.

- **Qualität (engl.: quality)**

Der Grad, zu dem ein System, ein Bestandteil oder ein Prozess vorgegebene Anforderungen erfüllt.

- **Qualitätssicherung (engl.: quality assurance)**

In Bezug auf Software ist die Qualitätssicherung jede geplante und systematische Tätigkeit zur Bewertung des Entwicklungs- und Fertigungsprozesses, die verwirklicht und dargelegt wird, um Vertrauen dahingehend zu schaffen, dass die Software die Qualitätsanforderungen erfüllt und eine konstante Qualität sicherstellt.

- **Rechner (engl.: computer)**

Programmierbare Funktionseinheit, die aus einem oder mehreren Prozessoren und peripherem Gerät besteht, und die durch interne Programme gesteuert wird und wesentliche Rechnungen durchführen kann, einschließlich umfangreicher arithmetischer oder logischer Rechenvorgänge, ohne menschlichem Eingriff während eines Rechenlaufs.

- **Rechnerprogramm (engl.: computer program)**

Synonym für: Computerprogramm (siehe Computerprogramm).

- **Redundanz (engl.: redundancy)**

Bereitstellung unterschiedlicher (identischer oder verschiedener) Strukturen, Systeme oder Komponenten zur Erfüllung der gleichen Funktion, so dass jede Struktur, die nicht unmittelbar vom Betriebszustand oder Versagen einer anderen betroffen ist, die geforderte Funktion ausüben kann.

Anmerkung: In diesem Sinne wären diversitäre Systeme zugleich immer auch redundant, aber nicht umgekehrt. In der Kerntechnik wird unter Redundanz häufig das Vorhandensein mehrerer gleicher Systeme für die Ausführung derselben Funktion verstanden, wodurch diversitäre Systeme nicht gleichzeitig als redundant bezeichnet werden.

- **Software (engl.: software)**

Programme, Daten, Regeln einschließlich zugehöriger Dokumentation, die zum Betrieb eines rechnerbasierten Systems gehören.

- **Softwarebasiertes System**

Synonym für: rechnerbasiertes System (siehe Programmierbares, rechnerbasiertes System).

- **Softwaredesign (engl.: software design)**

Softwaredesign ist der Entwurfsprozess zur Planung einer Software-Lösung und Teil des gesamten Softwareentwicklungsprozesses.

Dabei wird unter Anwendung von wissenschaftlichen Grundsätzen, technischen Informationen und des menschlichen Einfallsreichtums die Definition eines Softwaresystems angestrebt, welches spezifizierte Funktionen bei größtmöglicher Wirtschaftlichkeit und Leistungsfähigkeit ermöglicht.

- **Softwareentwicklung (engl.: software development)**

Phase des Softwarelebenszyklus, die zur Erzeugung der Software für ein leittechnisches System oder eines anderen Softwareprodukts führt. Sie umfasst alle Tätigkeiten von der Anforderungsspezifikation bis zur Installation in der Anlage.

- **Softwaremodifizierung (engl.: software modification)**

Änderung eines bereits abgestimmten Dokuments, die zu einer Veränderung des ablauffähigen Codes führt, oder Änderungen am Code selbst.

Anmerkung: Softwaremodifizierungen können während der Entwicklung von Software auftreten (z. B. bei der Behebung von Fehlern, die in einem späteren Stadium der Entwicklung gefunden wurden), oder nachdem die Software bereits in Betrieb genommen wurde.

- **Software-Version (engl.: software version)**

Jeweilige Ausgabe eines Software-Produkts, die durch Änderung oder Korrektur eines vorhergehenden Software-Produkts erstellt wurde.

- **Softwarevalidierung (engl.: software validation)**

Test und Überprüfung der integrierten Software, um Übereinstimmung mit den in der Spezifikation des leittechnischen Systems vorgegebenen Anforderungen an Funktionalität, Leistungsfähigkeit und Schnittstellen sicherzustellen. Außerdem wird nachgewiesen, dass die Software die von ihr geforderte Funktion in der vorgesehenen Umgebung erfüllt.

- **Spezifikation (engl.: specification)**

Dokument, in dem die Anforderungen, Funktionsweisen oder andere Eigenschaften eines Systems oder einer Komponente, oft auch die Verfahren zur Überprüfung der Einhaltung dieser Angaben, in kompletter, präziser und verifizierbarer Weise angegeben sind.

Anmerkung: Es gibt verschiedene Arten von Spezifikationen, z. B. Anforderungs- oder Systemspezifikationen.

- **Systemsoftware (engl.: system software)**

Teil der Software eines leittechnischen Systems, der für ein bestimmtes Rechnersystem oder eine Rechnerfamilie erstellt wurde, um Entwicklung, Betrieb und Modifikation des Systems und der zugehörigen Programme zu erleichtern.

Anmerkung: Die Systemsoftware der Gerätefamilien besteht üblicherweise aus der Betriebssoftware und der Software unterstützender Systeme (Softwarewerkzeuge).

- **Systemvalidierung (engl.: system validation)**

Bestätigung durch Prüfung und Beibringen anderer Nachweise, dass ein System zur Gänze die Anforderungsspezifikation erfüllt (Funktionalität, Ansprechzeit, Fehlertoleranz, Robustheit).

- **Verifizierung (engl.: verification)**

Durch Prüfen und Vorlegen objektiver Beweise sicherstellen, dass die Ergebnisse einer Tätigkeit den Zielen und Anforderungen entsprechen, die für diese Tätigkeit definiert wurden.

Versagen (engl.: failure)

Abweichen des vorliegenden Verhaltens von dem beabsichtigten Verhalten.

- **Versagen gemeinsamer Ursache (engl.: common cause failure (CCF), Gemeinsam verursachter Ausfall, GVA)**

Gleichzeitiges Versagen von zwei oder mehreren Strukturen, Systemen oder Komponenten infolge eines einzelnen spezifischen Ereignisses oder Grundes.

- **Vorgefertigte Software (engl.: pre-developed software (PDS))**

Software, die bereits vorgefertigt, als kommerzielles oder gesetzlich geschütztes Produkt verfügbar und für den Einsatz in einem rechnerbasierten System vorgesehen ist. Handelt es sich um ein handelsübliches Produkt oder Massenware, so wird auch der Begriff (Commercial-)Off-The-Shelf-Software verwendet.

- **Zertifizierung (engl.: certification)**

Als Zertifizierung bezeichnet man ein Verfahren, mit dessen Hilfe die Einhaltung der spezifizierten Anforderungen nachgewiesen wird. Als Ergebnis des Verfahrens wird schriftlich garantiert, dass ein System oder Bestandteil den Voraussetzungen entspricht und der Einsatz im betrieblichen Gebrauch zulässig ist.

- **Zuverlässigkeit (engl.: reliability)**

Die Zuverlässigkeit eines Systems oder Bestandteils ist eine Eigenschaft, die angibt, wie verlässlich das System die zugewiesenen Funktionen unter festgesetzten Bedingungen für einen definierten Zeitraum durchführt bzw. ein spezifiziertes Leistungsniveau unter spezifizierten Bedingungen aufrechterhalten wird.

2.2 Bisherige Arbeiten der GRS

Auswertung der Betriebserfahrung mit rechnerbasierten und programmierbaren Einrichtungen

Im Rahmen des Vorhabens 3610R01361 „Entwicklung und Einsatz von Analysemethoden zur Beurteilung software-basierter leittechnischer Einrichtungen in deutschen Kernkraftwerken“ wurde die Betriebserfahrung mit rechnerbasierten und programmierbaren Einrichtungen unterhalb der Meldeschwelle untersucht /GRS 15c/. Da rechnerbasierte und programmierbare Komponenten unterschiedliche Ausfallmechanismen und Fehlerursachen aufweisen können, sollte mit der Auswertung der Betriebserfahrung ermittelt werden, ob die bisherigen Abläufe zur Bewertung der Zuverlässigkeit solcher Komponenten beibehalten werden können oder angepasst werden müssen. Die Auswertung der Betriebserfahrung hat ergeben, dass in den untersuchten deutschen Anlagen zwar bereits einige leittechnische Komponenten durch programmierbare oder rechnerbasierte Modelle ersetzt wurden, ein großer Teil der Anlagentechnik aber noch mit herkömmlichen, konventionellen Komponenten betrieben wird. Dennoch wurden folgende Softwarefehler in diesem Projekt identifiziert:

- Baugruppeninterner Fehler im Diagnosepuffer
  → Baugruppe wurde ausgetauscht
- Kommunikationsstörung
  → Programm nach Urlöschen der CPU neu übersetzt und übertragen
- Kompletter Programmverlust der Sicherheitssteuerung trotz betriebsbereiter Pufferbatterien
  → Programm neu übertragen
- Ausfall einer Systemfunktion eines Überwachungssystems
  → Ertüchtigten Softwarebaustein übertragen
- Automatisierungsprozessor wegen Speicherfehler ausgefallen
  → Urgelöscht und zurückgesetzt
- Siebbänder schalten sporadisch im Automatikbetrieb nicht zu
  → Programm der Steuerung angepasst
- Wartenmeldung steht nur ca. 20s an und quittiert sich dann selbst
  → Programm geändert
- Steuerschrank in Störung
  → Software geladen und geprüft
- Baugruppe zeigt „Run“ und „Stop“ gleichzeitig an
  → Telegrammaufträge beim Kommunikationsprozessor neu geladen
- Ausfall der CPU
  → Busverbindungstelegramme neu geladen und synchronisiert
- Untergruppensteuerung in Störung (Rückmeldung „Ein“ fehlt)
  → Neustart der Steuerung
- Programmverlust der Betriebssteuerung
  → Software wurde wieder eingespielt

Wesentliche Beiträge zu Fehlern an Softwarekomponenten haben gemäß /GRS 15c/ Ereignisse geliefert, bei denen Pufferbatterien ausgefallen sind. Pufferbatterien kommen in den Anlagen beispielsweise in CPUs zum Einsatz. Dort werden sie benötigt, um bei Ausfall der Spannungsversorgung den Inhalt des RAM-Speichers zu erhalten. Fällt die Pufferbatterie bei abgeschalteter Spannungsversorgung aus, führt dies dazu, dass der

Inhalt des RAM-Speichers verloren geht. Ohne Speicherinhalt läuft die Steuerung bei Wiedereinschalten der Spannungsversorgung nicht an. Folgende Ursachen für den Ausfall der Pufferbatterien wurden in /GRS 15b/ ermittelt:

- Chargenproblem
  Die Pufferbatterien einer Charge wiesen nicht die gewünschten Eigenschaften auf. Nach Entdeckung dieses Problems wurden die Batterien dieser Charge vollständig ausgetauscht.

- Ständig belastete Pufferbatterien
  Auch wenn eine Spannungsversorgung, in der eine Pufferbatterie eingebaut ist, beispielsweise nur während der Revisionszeit benötigt wird, werden die vorhandenen Pufferbatterien zum Erhalt des RAM-Speichers dauerhaft belastet. Dies führt dazu, dass die Batterien häufiger ausfallen und dadurch ein entsprechender Austausch erforderlich ist. Um die Ausfälle zu verhindern, wurde ein jährliches Austauschintervall statt den typischen 2 Jahren für diese Pufferbatterien eingeführt.

Weiterhin hat sich gezeigt, dass Programmierfehler auftreten können. Diese Fehler sind meist schwer zu detektieren, da sie beispielsweise nur sporadisch oder bei bestimmten Betriebszuständen auftreten. In den in /GRS 15c/ betrachteten Fällen wurden vorgefundene Fehler durch ein Firmware-Update im Rahmen einer vorbeugenden Instandhaltung behoben. Diese wird im Allgemeinen von der Herstellerfirma geliefert. Den Mitarbeitern der Anlage ist dabei meistens nicht bekannt, welche Details sich in der neuen Firmware geändert haben. Da für betriebliche Komponenten keine entsprechende Anforderung aus dem Regelwerk besteht, werden typischerweise hierzu auch keine weiteren Nachforschungen seitens des Betreibers veranlasst. Für sicherheitstechnisch wichtige Komponenten bestehen entsprechende Anforderungen, so dass hier eine Firmware nur mit einer genehmigten Versionsnummer aufgespielt werden darf. /GRS 15c/

Diversitätsmerkmale rechnerbasierter und programmierbarer Einrichtungen

Die GRS hat sich bereits im Rahmen des Vorhabens 3611R01355 „Aufstellung von Kriterien und Kenngrößen zur deterministischen Prüfung der Eignung von Redesign-Komponenten für den Einsatz in der Sicherheitsleittechnik von Kernkraftwerken“ mit dem Thema der Diversität von programmierbaren und rechnerbasierten Einrichtungen auseinandergesetzt. Da nach Ansicht der GRS die betrachteten Normen und Standards keinen ausreichenden Anwendungsbezug aufweisen und zu allgemein formuliert sind und

darüber hinaus unterschiedliche Herangehensweisen bei der Beurteilung der Diversität bestehen, hat die GRS eine eigene Diversitätsmatrix für Bestandteile eines generischen Leittechniksystems entwickelt /GRS 15b/. Diese Matrix enthält Diversitätsmerkmale, die anhand des Lebenszyklus eines leittechnischen Systems und seiner Komponenten strukturiert sind.

Die Diversität eines leittechnischen Systems liegt nach /GRS 15b/ erst dann vor, wenn Diversität für alle relevanten Diversitätsmerkmale nachweisbar ist. Im Bereich der Softwareerstellung werden folgende Diversitätsmerkmale gefordert /GRS 15b/:

- System- und Anwendungssoftware
- Vorgehensmodell oder Verfahren zur Softwareentwicklung
- Programmierverfahren
- Softwarewerkzeuge
- Programmiersprachen
- Unterstützende Bibliotheken
- Hersteller einschließlich Unterauftragnehmer
- Vorgefertigte Software

Nur wenn für alle diese Punkte entsprechend den Anforderungen aus /GRS 15b/ Diversität nachgewiesen werden kann, ist die Software aus Sicht der GRS diversitär.

Softwarefehler

Im aktuell laufenden Vorhaben 3614R01304 „Auswirkungsbereiche von Softwarefehlern in sicherheitstechnisch wichtigen Einrichtungen von Kernkraftwerken“ wird die Betriebserfahrung mit softwarebasierten leittechnischen Einrichtungen ausgewertet und generisch die verfahrenstechnischen Auswirkungen von Softwarefehlern in sicherheitstechnischen Einrichtungen untersucht. Neben bereits aufgetretenen Softwarefehlern sollen mögliche Softwarefehler postuliert und ihre Auswirkungen untersucht werden.

Probabilistische Zuverlässigkeitsbewertung digitaler Leittechnik

Im Vorhaben SR 2418 „Fachliche Unterstützung des BMU bei der Entwicklung probabilistischer Bewertungsmethoden – Anpassung und Erprobung von Methoden zur probabilistischen Bewertung digitaler Leittechnik“ /GRS 04/ wurden Methoden zur probabilistischen Zuverlässigkeitsbewertung digitaler Leittechnik zusammengestellt, angepasst

und erprobt. Dazu wurden zunächst international angewandte Methoden zur Zuverlässigkeitsbewertung digitaler Leittechnik sowie die Betriebserfahrung hinsichtlich der Zuverlässigkeit digitaler Leittechnik ausgewertet und die für eine Periodische Sicherheitsüberprüfung relevanten Zuverlässigkeitsdaten ermittelt. Unter anderem wurden in diesem Projekt auch Software-Metriken betrachtet. Es wurden verschiedene Metriken beschrieben. Dazu zählen zum einen flache Zähl-Metriken, die den Software-Code direkt zählend erfassen oder analytische Metriken, die den Code in abstrakte Strukturen (z.B. Graphen) umsetzen und diese Strukturen analysieren. Bei der praktischen Anwendung insbesondere analytischer Metriken werden Programme, sog. Software-Analysatoren, verwendet, die den Software-Code des Messobjekts einlesen, den Code dann in eine abstrakte Struktur transferieren, diese Struktur dann analysieren und daraus einen Messwert für die entsprechende analytische Metrik ermitteln. In dem Vorhaben wurden u. a. verschiedene Metriken eingesetzt, um exemplarisch eine Komplexitätsbewertung von Funktionsbausteinen und Funktionsplänen durchzuführen. Es wurde darüber hinaus am Beispiel einer Leittechnikfunktion gezeigt, wie eine Zuverlässigkeitsbewertung der Software in die Fehlerbaummethode zur Zuverlässigkeitsbewertung der Leittechnik-Hardware integriert werden kann.

2.3 Stellungnahme der RSK zum Einsatz rechnerbasierter Leittechnik

Die RSK geht in ihrer Stellungnahme „Rechnerbasierte Sicherheitsleittechnik für den Einsatz in der höchsten Sicherheitskategorie in deutschen Kernkraftwerken“ /RSK 11/ ausführlich auf die Maßnahmen zur Vermeidung und Beherrschung des GVA ein. Darin wird insbesondere die Frage diskutiert, ob im Rahmen von Nachrüstungen oder Modernisierungsprojekten ein Einsatz rechnerbasierter Leittechnik in Reaktorschutzsystemen erfolgen kann. Als Vorteile einer rechnerbasierten Sicherheitsleittechnik werden u.a. die bessere Wartbarkeit, die inhärente Möglichkeit der Selbstüberwachung und der Fehlererkennung, das Vermeiden von driftbedingtem Abweichen von Grenzwerten sowie eine umfassendere und schnellere Störungsanalyse durch entsprechende Diagnosetools aufgezählt. Als Nachteile gelten beispielsweise die hohe Funktionsdichte auf einzelnen Baugruppen und daraus resultierende Fehlerauswirkungen auf viele Funktionen bei einem Baugruppenausfall, umfangreiches und sehr komplexes Design von Hard- und Software sowie keine vollständige Prüfbarkeit aufgrund der hohen Komplexität der anwendungsunabhängigen Systemhardware und –software. Als besonders kritisch wird das Potenzial für GVA im Bereich der Systemsoftware angesehen.

Die verschiedenen bestehenden kerntechnischen Normen des IEC-, DIN-IEC und DIN-EN liefern laut /RSK 11/ einen Rahmen für die Betrachtung von GVA und die zu ergreifenden Gegenmaßnahmen. Die RSK ist jedoch der Auffassung, dass zusätzliche, über den zum Zeitpunkt der Stellungnahme aktuellen Stand der Normen hinausgehende Verfahren erforderlich sind, um eine optimale und nachweisbare GVA-Auslegung auf Systemebene zu gewährleisten.

Die RSK gibt in /RSK 11/ zur systematischen Analyse potentieller GVA-Mechanismen und Common-Cause-Auslöser folgende Empfehlungen:

- GVA-Analyse: Überprüfung jeder redundanten Systemkonfiguration auf anzunehmende Common-Cause-Auslöser (CCI – Common Cause Initiator)
- Analyseleitfaden: Es wird die Schaffung eines Leitfadens gefordert, der als Basis für ein einheitliches Vorgehen zur Durchführung von GVA-Analysen auf Geräte- und Systemebene dienen soll. In ihm sollen die in den verschiedenen Normen genannten Einzelverfahren FTA (Fehlerbaumanalyse), FMEA (Failure Mode Effect Analysis) und CCA (Common Cause Analysis) zusammengeführt werden. Neben den nuklearen Normen und Standards sollen dazu insbesondere die Normen DIN EN 61508 /DIN 61b/ sowie ECSS-Q-80 /ECSS 80/ verwendet werden.
- CCI-Katalog: Für die GVA-Analysen soll ein Katalog erstellt werden, der die mindestens zu betrachtenden Auslöser für Soft- und Hardware-GVA zusammenstellt. Dieser Katalog soll dem aktuellen Kenntnisstand hinsichtlich möglicher CCI entsprechen und immer auf aktuellem Stand gehalten werden.
- HALT-Tests: Nach dem Vorbild der Luftfahrt, bei der sogenannte HALT-Testverfahren durchgeführt werden (Highly Accelerated Lifetime Testing), soll die Widerstandskraft eines Teilsystems bei gleichzeitigem Einwirken einer Vielzahl externer Einflüsse ermittelt werden. Wenn durch solche Testverfahren neue Erkenntnisse zu erwarten sind, sollen für rechnerbasierte Systeme, die im Rahmen von Reaktorschutzanwendungen eingesetzt werden, HALT-Tests durchgeführt werden.
- Verfolgung von Betriebserfahrung: Werden rechnerbasierte Leittechniksysteme für Funktionen der Kategorie A oder B eingesetzt, ist zur Sicherstellung des erforderlichen hohen Zuverlässigkeitsniveaus ein Informationsaustauch über Einsatzerfahrungen und Systeminformationen Behörden und Gutachtern zu gewährleisten.

Die RSK bemängelt, dass die Anforderungen des IEC-Regelwerks für leittechnische Systeme, die Kategorie-A Funktionen erfüllen, in der Praxis nicht ausreichend umgesetzt werden. Ursache dafür ist, dass für manche Anwendungen keine IEC ausgelegte Gerätetechnik zur Verfügung steht und daher industrielle Standardprodukte verwendet werden. Die RSK fordert daher:

- Für die Ausführung von Kategorie-A-Funktionen dürfen grundsätzlich nur Geräte eingesetzt werden, die den Anforderungen des IEC-Regelwerks zur Arbeitsweise rechnerbasierter Leittechnik, die Kategorie-A-Funktionen ausführt, genügen (IEC 61513, IEC 60880, IEC 62340). Diese rechnergestützten Geräte sollen auf einer streng sequentiellen und zyklischen Software-Verarbeitung basieren und einen hohen Grad an deterministischem Verhalten aufweisen.
- Werden Geräte eingesetzt, die nicht nach IEC ausgelegt sind, ist nachzuweisen, dass zu unterstellende GVA keine unzulässigen sicherheitstechnischen Auswirkungen auf die Anlage haben.

Um Fehler in der Aufgabenstellung und bei der Implementierung der Aufgabenstellung zu vermeiden, definiert die RSK folgende Qualitätsanforderungen zum Erlangen funktionaler Diversität:

- Berücksichtigung aller sicherheitsrelevanter Merkmale (Vollständigkeit)
- Eindeutigkeit der Aufgabenstellung und formalisierte Spezifikation
- Einfachheit der Aufgabenstellung, so dass eine vollständige Prüfung möglich ist
- Beschreibung von Testfällen und erwarteten Testergebnissen, welche die Korrektheit der Implementierung überprüfen
- Aussagen über Betriebszustände, in denen die entsprechende Funktion verfügbar sein muss
- Berücksichtigung der unterschiedlichen Zustände an den Schnittstellen aufgrund von Prüf- und Reparaturtätigkeiten

Um Fehler zu reduzieren wird zudem empfohlen eine unabhängige Implementierung der Aufgabenstellung mittels unterschiedlicher Entwicklungsumgebungen durch verschiedene Entwicklerteams zu realisieren. Wenn eine funktionale Diversität nicht möglich ist, muss die Diversifizierung der internen Zustände der Rechner durch leittechnische Mittel oder Geräte-Diversität erfolgen. Geeignete Maßnahmen sind:

- unterschiedliche Belegung der Ein-/Ausgabekanäle,

- unterschiedliche Abarbeitungsfolge der leittechnischen Funktionen
- unterschiedliche Normierung der verwendeten Messsignale,
- unsymmetrische Implementierung zusätzlicher Überwachungsfunktionen,
- unterschiedliche Implementierung der Funktionen.

Es werden Anforderungen für die Erstellung und Prüfung von Software beschrieben. Darin wird gefordert, dass Software nach einem Phasenmodell (Erstellung der Anforderungsspezifikation, Systemspezifikation, Entwurf, Installation sowie Betrieb und Wartung) zu erstellen ist. Um die konstruktive Qualität der Software der Kategorie 1 zu gewährleisten, wird gefordert, die Software mit rechnergestützten Werkzeugen zu erstellen. Die Softwareeinheiten sind mit möglichst geringem Funktionsumfang in klar abgegrenzten Einheiten aufzubauen. Die Ergebnisse der einzelnen Phasen sind unter Anwendung formaler Analysemethoden und zusätzlicher Tests vollständig zu verifizieren. Nach der Installation soll das Verhalten des Hardware- und Softwaresystems validiert werden. Es ist eine vollständige Entwicklungs-, Qualitätssicherungs- und Benutzerdokumentation zu erstellen. Für Software der Kategorien 2 und 3 gelten jeweils abgeschwächte Anforderungen.

2.4 Fachvorträge

Im Rahmen des Projekts wurden zum Ausbau der Expertise Fachvorträge auf insgesamt drei Konferenzen besucht.

Für die Erweiterungen der Expertise in den Themenbereichen Softwarezuverlässigkeit und Testmethoden wurde an der „14th International Conference on Software QA and Testing on Embedded Systems“ teilgenommen. Zur Unterstützung der Bearbeitung konkret von Arbeitspaket 4 (AP 4) und Arbeitspaket 5 (AP 5) wurden auf der Konferenz Vorträge zum Thema Softwarezuverlässigkeit in sicherheitskritischen Anwendungen sowie Methoden und Metriken besucht. Die im Projekt durchgeführten Arbeiten konnten bestätigt und ergänzt werden.

Die thematischen Schwerpunkte der „Nordic PSA Castle Meeting 2015“ lagen auf der Bewertung und Berechnung von Ausfallwahrscheinlichkeiten sowie der Diskussion der digitalen Automatisierung. So wurden zum Beispiel im Rahmen eines Vortrages die Vor- und Nachteile einer Digitalisierung diskutiert. Die gewonnen Erkenntnisse wurden in die Diskussionen des Arbeitspaketes 4 (AP 4) aufgenommen.

Auf der Konferenz „NKS-R MODIG and PLANS, Joint workshops on reliability analysis and safety demonstrations of digital I&C, 2015“ konnte die Expertise in den Themenschwerpunkten Zuverlässigkeitsanalyse, Bewertung der Fehlertoleranz und Führen von Sicherheitsnachweisen von digitalen Leittechniksystemen in Kernkraftwerken ausgebaut werden. Informationen und Erkenntnisse zu Themenstellungen der Konferenz wurden in die Diskussion des Arbeitspaketes 4 (AP 4) und des Arbeitspaketes 5 (AP 5) mit einbezogen.

Darüber hinaus konnten im Rahmen der Literaturrecherche vier weitere Vorträge als für das Projekt relevant identifiziert werden. Sie werden in den thematisch entsprechenden Kapiteln in die Diskussion einbezogen.

- Prof. Dr. Fähnrich, K.-P.: Vorlesung Softwaretechnik – Vorgehensmodelle, V-Modell XT, Universität Leipzig, erreichbar unter http://tinyurl.com/hbcblqj, zitiert am 12.02.2016.
- Finnish Centre for Radiation and Nuclear Safety: Probabilistic Safety Analyses (PSA), Helsinki, YVL 2.8, pp. 1-10, 1997.
- Prof. Dr. Koschke R.: Softwaretechnik, Fachbereich Mathematik und Informatik, Universität Bremen, 2013, erreichbar unter https://www.informatik.uni-bremen.de/st/lehre/swt13/prozesse.pdf, zitiert am 12.02.2016.
- Saglietti, F.: Computer Safety, Reliablitity, and Security, 27th International Conference, SAFECOMP 2007.

2.5 Fachliteratur

Im Rahmen der Arbeiten zum Projekt wurde eine Literaturrecherche durchgeführt. Es existiert eine Vielzahl an Veröffentlichungen, die sich im Allgemeinen mit den Themen Zuverlässigkeit und Qualität von Software, Zuverlässigkeitsbewertung sowie Methoden und Metriken befassen. Die in den Veröffentlichungen konkret behandelten Themen entsprechen jedoch oft nicht den Fragestellungen des Projekts oder behandeln die Themen so oberflächlich, dass sie keinen inhaltlichen Gewinn für das Projekt darstellen.

Folgende Veröffentlichungen haben sich hingegen für das durchgeführte Projekt als relevant erwiesen und werden in den thematisch entsprechenden Kapiteln in die Diskussion einbezogen.

- Bless, M.: Scrum und die IEC 62304, Medizinische Software mit agilen Methoden normenkonform entwickeln, Version 1.8, Januar 2014. Erreichbar unter http://agilecoach.de/wp-content/uploads/2013/07/Marc-Bless-Scrum-und-die-IEC-62304-eBook-v1.8.pdf, zitiert am 12.02.2016.
- Chu, T.-L., Yue, M., Martinez-Guridi, G., Lehner, J.: Review of quantitative software reliability methods, Brookhaven National Laboratory, Letter Report, Digital System Software PRA, JCN N-6725, September 2010.
- Fusani, M.: Examining software engineering requirements in safety-related standards, РАДІОЕЛЕКТРОННІ І КОМП'ЮТЕРНІ СИСТЕМИ, 2009, № 7, 02.02.2009.
- Koord, Y., Krauter, V.: Überblick Vorgehensmodelle im Projektmanagement, FOM Hamburg, 2016, http://tinyurl.com/he2jkhs/, zitiert am 12.02.2016.
- Littlewood, B.: The Need for Evidence from Disparate Software to Evaluate Software Safety, Directions in Safety Critical Systems, Springer-Verlag, 1993.
- Littlewood, B., Verrall, J. L: A Bayesian Reliability Model with a Stochastically Monotone Failure Rate, IEEE Transactions on Reliability, Vol. R-23, June 1974.
- Lyu, M. (ed.): Software Fault Tolerance, John Wiley and Sons, Inc., 1995.
- Lyu, M. R.: Handbook of Software Reliability Engineering, April 1996.
- Leveson, N.G.: Safeware: System Safety and Computers, Addison-Wesley, 1995.
- Musa, J.: Software Reliability Engineering, McGraw Hill, 1999.
- Rees, R.A.: Detectability of Software Failures, Reliability Review, Vol 14, 1994.
- Scholz, F.: Bewertungsmöglichkeit der Softwarezuverlässigkeit digitaler Leittechnik, August 2001.

3 Überblick über nationale und internationale Normen und Standards

Zahlreiche Normen und Standards beschäftigen sich mit der Entwicklung von sicherheitsrelevanter Software. Einige davon gelten allgemein für sicherheitskritische Bereiche und damit sowohl für nukleare als auch für nicht-nukleare Anwendungen. Andere sind speziell auf einen dieser Bereiche, wie beispielsweise Avionik, Schienenverkehr oder Wehrtechnik ausgerichtet. Eine vollständige Aufzählung dieser Normen zu geben ist aufgrund ihrer Vielzahl nicht zielführend. Daher wurde zunächst im Rahmen einer intensiven Recherche ermittelt, welche Standards und Normen eine besondere Relevanz für dieses Projekt aufweisen. Sie werden in den folgenden Abschnitten strukturiert dargestellt und ihre Inhalte in Anhang B kurz beschrieben. Diese Vorarbeit dient neben der Ermittlung des Standes von Wissenschaft und Technik als Basis für einen Vergleich der Anforderungen an die Software-Entwicklung und die Methoden zur Qualifizierung und Zuverlässigkeitsbewertung sowie eine detaillierte Diskussion der entsprechenden Arbeitspakete 2-5 (AP2-AP5) in den nachfolgenden Abschnitten.

3.1 Allgemeine Standards und Normen

Für sicherheitskritische Bereiche in deutschen Industrieanlagen werden allgemeingültige Anforderungen an den Einsatz von Software in der Norm

- DIN EN 61508 „Funktionale Sicherheit sicherheitsbezogener elektrischer/ elektronischer/ programmierbarer elektronischer Systeme“, vor allem Teil 3 „Anforderungen an Software“ /DIN 61d/
- DIN EN ISO 13849 „Sicherheit von Maschinen – Sicherheitsbezogene Teile von Steuerungen“ /ISO 15/, /ISO 13/

formuliert. International stehen hierfür zusätzlich die folgenden Standards zur Verfügung

- IEEE 1074 “Standard for Developing Software Life Cycle Processes“ /IEEE 74/
- IEEE 1012 “Standard for System and Software Verification and Validation” /IEEE 12/.

Speziell für die Bewertung der Systemzuverlässigkeit sind für dieses Vorhaben die folgenden allgemein für sicherheitskritische Bereiche gültigen Standards und Berichte relevant:

- IEEE 1413 “IEEE Standard Framework for Reliability Prediction of Hardware“ /IEEE 13/
- IEEE 1633 “IEEE Recommended Practice on Software Reliability” /IEEE 33/
- BS5760 “Reliability of systems, equipment and components, Part 8: Guide to assessment of reliability of systems containing software” /BSI 98/

Diese Normen und Standards werden in Anhang B beschrieben.

3.2 Nukleare Standards und Normen

3.2.1 Nationale Regelwerke, Standards und Normen

Die Nutzung der Kernenergie ist in Deutschland durch verschiedene Gesetze, Verordnungen, Regelungen, Leit- und Richtlinien geregelt. Die rechtlichen Voraussetzungen für die friedliche Nutzung der Kernenergie schafft in Deutschland das Atomgesetz. Die dort enthaltenen Forderungen werden durch das kerntechnische Regelwerk konkretisiert. Dieses beinhaltet u.a. die Sicherheitsanforderungen an Kernkraftwerke mit ihren Interpretationen sowie die Regeln des Kerntechnischen Ausschusses (KTA). Dieses kerntechnische Regelwerk wird zudem durch das konventionelle technische Regelwerk unterstützt, dem beispielsweise DIN-Normen unterlagert sind.

Grundsätzliche Anforderungen an Kernkraftwerke und damit an die hier zu betrachtenden Leittechniksysteme werden zunächst in den folgenden Dokumenten behandelt:

- Sicherheitsanforderungen an Kernkraftwerke /BMU 12/
- Interpretationen zu Sicherheitsanforderungen an Kernkraftwerke /BMU 13/.

Damit konkrete Anforderungen an Leittechniksysteme definiert werden können, werden in /BMU 13/ die verschiedenen Leittechnikfunktionen zunächst entsprechend ihrer sicherheitstechnischen Bedeutung in die Kategorien A, B und C eingeteilt. Die Kategorie A umfasst demnach alle Leittechnik-Funktionen, die erforderlich sind, um Ereignisse der Sicherheitsebene 3 zu beherrschen. Kategorie B umfasst alle Leittechnik-Funktionen,

die erforderlich sind, um Ereignisse der Sicherheitsebene 2 zu beherrschen sowie das Eintreten von Ereignissen der Sicherheitsebene 3 zu vermeiden. Kategorie C umfasst alle übrigen sicherheitstechnisch wichtigen Leittechnik-Funktionen.

Darüber hinaus umfasst

- DIN EN 61226 „Kernkraftwerke – Leittechnische Systeme mit sicherheitstechnischer Bedeutung – Kategorisierung leittechnischer Funktionen“ /DIN 09/

Anforderungen daran, wie leittechnische Funktionen den Kategorien A, B und C zuzuordnen sind. Entsprechend dieser Kategorien beschäftigen sich die folgenden Normen mit Softwareaspekten rechnerbasierter leittechnischer Systeme:

- DIN IEC 60880 „Kernkraftwerke – Leittechnische Systeme mit sicherheitstechnischer Bedeutung: Softwareaspekte für rechnerbasierte Systeme zur Realisierung von Funktionen der Kategorie A“ /DIN 07/
- DIN EN 62138: „Kernkraftwerke – Leittechnik für Systeme mit sicherheitstechnischer Bedeutung: Softwareaspekte für rechnerbasierte Systeme zur Realisierung von Funktionen der Kategorien B oder C“ /DIN 10a/

Hier werden zusätzlich zu der Kategorisierung der Leittechnikfunktionen in A, B und C die Systeme, welche die entsprechenden Leittechnikfunktionen ausführen, in drei Klassen unterteilt. Leittechnische Systeme der Klasse 1 dienen im Wesentlichen der Realisierung von Leittechnik-Funktionen der Kategorie A, Systeme der Klasse 2 von Leittechnik-Funktionen der Kategorie B und Systeme der Klasse 3 von Leittechnik-Funktionen der Kategorie C. Grundsätzlich können die so klassifizierten Systeme zusätzlich Leittechnik-Funktionen aus niedrigeren Kategorien ausführen.

Vertiefend wird dann in folgenden Regeln des KTA auf die verschiedenen Aspekte der Leittechniksysteme eingegangen:

- KTA 3501 „Reaktorschutzsystem und Überwachungseinrichtungen des Sicherheitssystems“ /KTA 31/
- KTA 3503 „Typprüfung von elektrischen Baugruppen der Sicherheitsleittechnik“ /KTA 33/

Darüber hinaus werden verschiedene Themen in Normen und Stellungnahmen betrachtet, welche auch für Softwareaspekte der Leittechnik relevant sind. Dazu zählen:

- DIN EN 61513 „Kernkraftwerke – Leittechnik für Systeme mit sicherheitstechnischer Bedeutung: Allgemeine Systemanforderungen“ /DIN 13/
- DIN EN 62340 „Kernkraftwerke - Leittechnische Systeme mit sicherheitstechnischer Bedeutung - Anforderungen zur Beherrschung von Versagen aufgrund gemeinsamer Ursache“ /DIN 10b/
- VdTÜV „Stellungnahme zu den erforderlichen Vorsorgemaßnahmen gegen systematisches Versagen von digitalen leittechnischen Einrichtungen in kerntechnischen Anlagen, die Leittechnikfunktionen der Kategorie 1 ausführen“ /VDT 08/
- VDI/ VDE 3528 „Anforderungen an Serienprodukte und Kriterien für deren Einsatz in der Sicherheitsleittechnik von Kernkraftwerken“ /VDI 11/

Die für diesen Bericht wesentlichen Aspekte der hier aufgeführten Dokumente werden in Anhang B beschrieben.

3.2.2 Internationale Standards und Normen

Im internationalen Bereich ist länderübergreifend die International Atomic Energy Agency (IAEA) tätig. Hierbei handelt es sich um eine autonome wissenschaftlich-technische Organisation, die 1957 von den Vereinten Nationen gegründet wurde und für insgesamt 159 Mitgliedsstaaten Leistungen erbringt. Zu ihren Aufgaben gehört unter anderem die Erstellung von Sicherheits-Standards für kerntechnische Anlagen. Diese sind für die Mitgliedstaaten nicht gesetzlich bindend, sollen aber nach Möglichkeit in den nationalen Regelwerken umgesetzt werden. Die in den Sicherheitsstandards gegebenen Empfehlungen sind daher als „weiche Forderungen“ (Should-Statements) /IAEA 00/, /IAEA 02/ formuliert. Neben den vorgeschlagenen Maßnahmen sind auch alternative Maßnahmen zur Erfüllung der Anforderungen erlaubt.

Folgende Sicherheits-Standards der IAEA beschäftigen sich mit Leittechniksystemen bzw. Software, welche in Leittechniksystemen zum Einsatz kommt.

- IAEA Safety Standard Series “Instrumentation and Control Systems Important to Safety in Nuclear Power Plants” Safety Guide No. NS-G-1.3 / IAEA 02/

- IAEA Safety Standard Series “Software for Computer Based Systems Important to Safety in Nuclear Power Plants” Safety Guide No. NS-G-1.1 /IAEA 00/
- IAEA Technical Report Series No. 367 “Software Important to Safety in Nuclear Power Plants” /IAEA 94/
- IAEA Technical Report Series No. 384 “Verification and Validation of Software Related to the Safety of Nuclear Power Plant Instrumentation and Control” /IAEA 99/
- IAEA Nuclear Energy Series “Protection against Common Cause Failures in Digital I&C Systems of Nuclear Power Plants” No. NP-T-1.5 / IAEA 09/

Wie in Deutschland verfügen verschiedene Länder über eigene, rechtlich bindende Regelwerke. Beispielsweise wird in Kanada die Software-Entwicklung für sicherheitstechnisch wichtige Software in folgendem Standard reglementiert:

- CE-1001-STD, Revision 2 „Standard for Software Engineering of Safety Critical Software” /ONT 99/

In den USA wird die Gesetzgebung vom Code of Federal Regulations (CFR) geregelt. Die entsprechenden Gesetze umfassen insgesamt 50 Titel. Hiervon beschäftigt sich ein Titel mit dem Sektor „Energie“ (CFR 10), in dem auch Vorgaben und Regulierungen für den Bereich der Kerntechnik zu finden sind. Konkrete Vorgaben zu softwarebasierter, sicherheitstechnisch wichtiger Leittechnik konnten hier jedoch nicht gefunden werden.

Das Energieministerium der Vereinigten Staaten (Department of Energy, DoE) ist verantwortlich für die nukleare Sicherheit. Zusammen mit der Aufsichtsbehörde NRC (Nuclear Regulatory Commission) überwachen sie die Einhaltung der gesetzlichen Bestimmungen in der Kerntechnik.

Die NRC hat über die gesetzlichen Bestimmungen hinaus eine Vielzahl zusätzlicher Standards, Regelwerke und Forschungsanalysen veröffentlicht. Diese sind in den USA zwar nicht gesetzlich bindend, jedoch werden einige Vorgaben der NRC in den USA für Genehmigungsverfahren herangezogen. Zu Leittechniksystemen und zum Einsatz von Software in Leittechniksystemen hat die NRC folgende Dokumente veröffentlicht:

- NUREG 800, Appendix 7.0-A “Review Process for Digital Instrumentation and Control Systems” /NUR 07/

- Regulatory Guide 1.152 “Criteria for Use of Computers in Safety Systems of Nuclear Power Plants” /NRC 11/
- NUREG/CR-6734 “Digital Systems Software Requirements Guidelines” /NUR 01/
- NUREG/CR-7007 “Diversity Strategies for Nuclear Power Plant Instrumentation and Control Systems” /NUR 08/

Darüber hinaus haben sich verschiedene europäische Aufsichtsbehörden und technische Sachverständigenorganisationen hinsichtlich der Bewertung von Software in Leittechniksystemen von Kernkraftwerken ausgetauscht und beraten und ein gemeinsames Positionspapier veröffentlicht:

- Licensing of safety critical software for nuclear regulators, Common position of seven European nuclear regulators and authorized technical support organizations /ENR 13/

Die für diesen Bericht wesentlichen Aspekte der hier aufgeführten Dokumente werden in Anhang B beschrieben.

3.3 Nicht-nukleare Standards und Normen

Neben Normen und Standards, die allgemein für sicherheitskritische Bereiche in verschiedensten Industriezweigen gelten und die, die speziell auf die Kerntechnik abgestimmt sind, liegen auch verschiedene Normen und Standards in anderen nicht-nuklearen Bereichen vor. Dazu zählt vor allem der Schienenverkehr, die Automobilindustrie, die Luft- und Raumfahrt und die Wehrtechnik.

National besteht für den Einsatz und die Qualifizierung von Software im Schienenverkehr die Norm

- DIN EN 50126 „Bahnanwendungen – Spezifikation und Nachweis der Zuverlässigkeit, Verfügbarkeit, Instandhaltbarkeit, Sicherheit (RAMS)“ /DIN 99/
- DIN EN 50128 „Bahnanwendungen – Telekommunikationstechnik, Signaltechnik und Datenverarbeitungssysteme – Software für Eisenbahnsteuerungs- und Überwachungssysteme“ /DIN 50/

- DIN EN 50129 „Bahnanwendungen – Telekommunikationstechnik, Signaltechnik und Datenverarbeitungssysteme – Sicherheitsrelevante elektronische Systeme für Signaltechnik“ /DIN 03/

Im Bereich der Luft- und Raumfahrt sind im Hinblick auf Software-Aspekte die folgenden internationalen Standards relevant:

- RTCA DO-178C „Software Considerations in Airborne Systems and Equipment Certification“ /RTCA 11/
- FAA Order 8110.49 „Software Approval Guidelines“ /FAA 03/
- NASA-STD-8719.13C: „Software Safety Standard“ /NASA 87/
- ECSS-Q-80, “Software Product Assurance” /ECSS 80/

Im Bereich der Automobilindustrie wurde die Normenreihe

- ISO 26262 „Road Vehicles – Functional Safety“ /ISO 11/

verfasst, in der die Anforderungen der allgemeinen Norm DIN EN 61508 auf die Anwendung in der Automobilbranche angepasst und konkretisiert wurden.

Im Bereich der Wehrtechnik gibt es verschiedene Standards und Richtlinien zum Einsatz von Software. Dazu zählen:

- MIL-STD-882E „Department of Defense Standard Practice - System Safety“ /MIL 88/
- Department of Defense, „Joint Software Systems Safety Engineering Handbook“ /DOD 10/
- NATO AOP-52 „Guidance on Software Safety Design and Assessment of Munition-Related Computing Systems“ /AOP 52/
- Defence Standard 00-42 „Reliability and maintainability assurance guides - part 2: software“ /DEF 42/
- Defence Standard 00-55 „Requirements for safety related software in defence equipment - part 1: requirements; part 2: guidance“ /DEF 55/

Die für diesen Bericht wesentlichen Aspekte der hier aufgeführten Dokumente werden in Anhang B beschrieben.

4 Auswertung der relevanten Standards und Normen

Im vorigen Kapitel wurden zunächst Standards und Normen in den verschiedenen Industriezweigen aufgeführt, welche für Software-Aspekte in der Sicherheitsleittechnik relevant sind. In diesem Kapitel werden die Inhalte dieser Standards und Normen der Kerntechnik gegenüber gestellt und eine Übertragbarkeit diskutiert. Die in diesem Kapitel durchgeführten Arbeiten entsprechen somit dem Arbeitspaket 2 (AP 2).

Da die Software-Entwicklung ein Teil des Software-Lebenszyklus und damit Bestandteil des System-Lebenszyklus des Gesamtsystems ist, wird im Folgenden mit einem Vergleich der System-Sicherheitslebenszyklen in den verschiedenen Industriezweigen begonnen. Anschließend wird auf die Unterschiede im Software-Sicherheitslebenszyklus eingegangen.

Schlussendlich werden die Anforderungen zu verschiedenen Themengebieten des Software-Sicherheitslebenszyklus konkret verglichen und eine Übertragbarkeit auf die Kerntechnik untersucht. Hierzu wird ergänzend eine Vergleichbarkeit der kerntechnischen Kategorien und der Sicherheitsintegritätslevel der unterschiedlichen Industriezweige erarbeitet und diskutiert.

Im Fazit des Kapitels wird zusammengefasst, in welchen Themenschwerpunkten eine Übertragung von Anforderungen an die Software auf die Kerntechnik möglich wäre.

4.1 System-Sicherheitslebenszyklus

In diesem Abschnitt werden zunächst allgemein die Phasen eines System-Sicherheitslebenszyklus in den verschiedenen Industriezweigen verglichen, um zu untersuchen, wie die allgemeinen Anforderungen aus DIN EN 61508 in der Anwendung umgesetzt werden und um Unterschiede darzustellen. Hierfür wurden die in Tab. 4.1 genannten, in Anhang B beschriebenen Normen herangezogen. In Tab. 4.2 sind die System-Sicherheitslebenszyklen zu den unterschiedlichen Industriezweigen vergleichend dargestellt.

Tab. 4.1 Zum Vergleich des System-Sicherheitslebenszyklus herangezogene Normen

Bereich	Norm	Titel
Allgemein	DIN EN 61508, Teil 1	Funktionale Sicherheit sicherheitsbezogener elektrischer/ elektronischer/ programmierbarer elektronischer Systeme
Bahn	DIN EN 50126	Bahnanwendungen – Spezifikation und Nachweis der Zuverlässigkeit, Verfügbarkeit, Instandhaltbarkeit, Sicherheit (RAMS)
Auto	ISO 26262, Teil 4	Road Vehicles – Functional Safety
Kerntechnik	DIN EN 61513	Kernkraftwerke – Leittechnik für Systeme mit sicherheitstechnischer Bedeutung: Allgemeine Systemanforderungen

Der in DIN EN 50126 dargestellte Lebenszyklus unterscheidet sich nur sehr wenig von dem allgemeinen Lebenszyklus in DIN EN 61508. Der einzige Unterschied hier ist, dass im Gegensatz zum allgemeinen Lebenszyklus E/E/PE-Systeme als Teil des Gesamtsystems nicht gesondert betrachtet werden. Hier wird von der Konzeptentwicklung bis zur Stilllegung und Entsorgung nur ein System betrachtet. Es gibt keine separaten Phasen für spezielle Aspekte von Teilsystemen oder für Hardware und Software.

Während in DIN EN 61508 nur sehr allgemein gefordert wird, dass für jede Phase des Lebenszyklus ein Verifizierungsplan aufzustellen ist und die Ergebnisse zu dokumentieren sind, wird beispielsweise in DIN EN 50126 in jeder Phase aufgeführt, welche Verifizierungsaufgaben zu erfüllen und welche Ergebnisse und Informationen zu dokumentieren sind. Ein eingehender Vergleich der Verifizierung und Validierung wird in Rahmen von Kapitel 5 durchgeführt.

Der Lebenszyklus in ISO 26262 ist auf die Serienproduktion der Automobilindustrie angepasst. Daraus resultiert ein etwas anderer Ablauf der einzelnen Phasen im Lebenszyklus als in DIN EN 61508. In DIN EN 61508 erfolgt beispielsweise die Validierung erst nach der Installation und Inbetriebnahme eines Systems. In der Automobilindustrie muss dies bereits vor der Freigabe zur Serienproduktion erfolgen, um nachzuweisen, dass die Erzeugnisse der Serienproduktion alle Anforderungen an das Produkt erfüllen. Außerdem beinhaltet der Lebenszyklus eine gesonderte Phase zur Produktionsfreigabe mit vorher durchgeführter Bewertung der funktionalen Sicherheit. Im Gegensatz zu DIN EN

61508 erfolgt eine Betrachtung von Änderungs- oder Nachrüstmaßnahmen im Lebenszyklus in ISO 26262 nicht erst zum Zeitpunkt des Betriebs des Systems, sondern bereits zu Beginn des Lebenszyklus, da jede Änderung Einfluss auf die Serienproduktion hat und damit bereits sehr früh bewertet werden muss.

Der Lebenszyklus in DIN EN 61513 unterscheidet sich vor allem in den ersten Phasen vom allgemeinen Lebenszyklus in DIN EN 61508. In der Kerntechnik ist es üblich, die sicherheitstechnische Auslegungsgrundlage der Anlage nach Prinzipien der IAEA sowie nach IEC-Normen und nationalen Regeln, die außerhalb des Anwendungsbereichs der DIN EN 61513 liegen, zu entwickeln /DIN 13/. Diese Auslegungsgrundlage beinhaltet bereits die Betrachtung von relevanten Ereignisabläufen, Risikoanalysen und eine Kategorisierung der Funktionen, welche von den Systemen ausgeführt werden sollen. Der Lebenszyklus in DIN EN 61513 baut daher auf diesen Eingangsinformationen für die erste Phase des Lebenszyklus auf, während in anderen Industriezweigen diese Betrachtungen im Lebenszyklus enthalten sind.

Nachfolgend werden in Tabelle 4.2 die einzelnen Elemente des System-Sicherheitslebenszyklus‘ aus den verschiedenen Normen einander gegenüber gestellt.

Zusammenfassend lässt sich festhalten, dass eine grundsätzliche Ähnlichkeit des System-Sicherheitslebenszyklus‘ in den verschiedenen Industriezweigen gegeben ist, jeder Anwendungsbereich jedoch einzelne Phasen aufgrund seiner spezifischen Gegebenheiten etwas anders oder in anderer Reihenfolge darstellt. Im Folgenden wird auf die einzelnen Phasen anhand des Lebenszyklus in DIN EN 61508 eingegangen.

4.1.1 Konzept

Der Lebenszyklus in DIN EN 61508 und DIN EN 50126 beginnt mit der Konzeptentwicklung, während in ISO 26262 mit der Definition des Gesamtsystems begonnen wird. In allen Fällen soll in dieser Phase zunächst ein ausreichendes Verständnis des Gesamtsystems entwickelt werden, um alle nachfolgenden Aufgaben zufriedenstellend erfüllen zu können. Hierzu werden Anforderungen formuliert, welche Informationen zur Verfügung stehen müssen und welche Aspekte betrachtet werden sollen. Dazu zählen beispielsweise anzuwendende rechtliche und regulatorische Anforderungen und die Wechselwirkungen mit anderen Systemen.

Tab. 4.2 System-Sicherheitslebenszyklus unterschiedlicher Industriezweige

DIN EN 61508	DIN EN 61513	ISO 26262	DIN EN 50126
Konzept			Konzept
Definition des Anwendungsbereichs		Definition des Items	Systemdefinition und Anwendungsbedingung
		Beginn des Sicherheitslebenszyklus	
Gefährdungs- und Risikoanalyse		Gefährdungsanalyse und Risikobewertung	Risikoanalyse
Gesamte Sicherheitsanforderungen	Anforderungsspezifikation für die gesamte Leittechnik	Konzept der funktionalen Sicherheit	Systemanforderungen
Zuordnung der Sicherheitsanforderungen	Auslegung und Planung der leittechnischen Gesamtarchitektur		Zuteilung der Systemanforderungen
Gesamtplanung von Betrieb, Instandhaltung, Sicherheitsvalidierung und Installation und Übergabe	Zuordnung der leittechnischen Funktionen zu den einzelnen leittechnischen Systemen	Planung von Betrieb und Produktion	
Sicherheitsanforderungsspezifikation für das E/E/EP-System			
		Produktentwicklung (System-, HW-, SW-Ebene)	Entwicklung/ Konstruktion und Implementierung
Realisierung des E/E/EP-Systems	Realisierung und Planung der einzelnen leittechnischen Systeme		Fertigung
Gesamte Installation und Freigabe	Gesamtintegration, Inbetriebnahme		Installation und Montage
Validierung der Gesamtsicherheit		Sicherheitsvalidierung	Systemvalidierung
		Bewertung der funktionalen Sicherheit	
		Freigabe zur Serienproduktion	Systemabnahme
		Produktion	
Betrieb, Instandhaltung und Instandsetzung	Gesamtbetrieb und Instandhaltung	Betrieb, Service und Stilllegung	Betrieb und Instandhaltung
Modifikationen		Modifikationen	Anforderungen an Nachrüstungen
Stilllegung und Entsorgung			Stilllegung und Entsorgung

Während DIN EN 61508 bereits hier auf die Definition von möglichen Gefahrensituationen eingeht, liegt in ISO 26262 der Fokus zunächst auf der Funktionalität des Systems. DIN EN 50126 fordert, die Gefahrenquellen, die die Leistungsfähigkeit und die Sicherheit des Systems beeinträchtigen können, aufzuzeigen. Im nuklearen Bereich ist diese Phase nicht im Anwendungsbereich der DIN EN 61513 enthalten.

4.1.2 Definition des Anwendungsbereichs des Gesamtsystems

Während in DIN EN 61508 und in DIN EN 50126 als nächstes die Grenzen des betrachteten Systems und der Umfang der Gefahren- und Risikoanalyse spezifiziert werden sollen, fordert ISO 26262 als nächsten Schritt die Festlegung, ob es sich um eine Neuentwicklung oder eine Modifikation handelt. Handelt es sich um eine Modifikation, muss der bereits bestehende Ablauf des Lebenszyklus für dieses System entsprechend angepasst werden. Es werden daher bei einer Modifikation entsprechende Anforderungen formuliert, um sicherzustellen, dass Änderungen am System die geforderte funktionale Sicherheit nicht beeinträchtigen. Eine solche Unterscheidung ist im Lebenszyklus nach DIN EN 61508 nicht vorgesehen. Anforderungen an Modifikationen eines bestehenden Systems werden erst zu einem späteren Zeitpunkt im Lebenszyklus formuliert. Da in der Automobilindustrie aber vor Beginn der Serienproduktion sichergestellt werden muss, dass die vorgesehenen Modifikationen keinen Einfluss auf die funktionale Sicherheit haben, kann der in DIN EN 61508 gezeigte Ablauf des Lebenszyklus nicht eingehalten werden. Im nuklearen Bereich ist diese Phase nicht im Anwendungsbereich der DIN EN 61513 enthalten.

4.1.3 Gefährdungs- und Risikoanalyse

Sowohl im Lebenszyklus nach DIN EN 61508 als auch nach ISO 26262 und DIN EN 50126 erfolgt als nächster Schritt eine Gefahren- und Risikoanalyse. In DIN EN 61508 wird sehr allgemein gefordert, dass in der Analyse folgende Aspekte zu berücksichtigen sind:

- Ermittlung aller möglichen Gefahren und Bestimmung der betroffenen Komponente
- Konsequenzen und Wahrscheinlichkeiten für das Auftreten eines Ereignisses
- Maßnahmen, um Gefahren und Risiken zu vermeiden oder zu reduzieren

In ISO 26262 wird gefordert, dass Gefahren, die durch Fehlfunktionen des Systems entstehen können, identifiziert und kategorisiert werden. Mögliche vorgesehene Sicherheitsmechanismen zur Verminderung von Gefahren sollen hierbei zunächst unberücksichtigt bleiben. Die identifizierten Betriebszustände und Situationen, in denen ein Fehlverhalten zu gefährlichen Ereignissen führen kann, sollen dann beschrieben und entsprechend klassifiziert werden. Hierbei werden in ISO 26262 drei Eigenschaften zur Klassifizierung genannt:

- Schwere der Auswirkung
- Häufigkeit der Fahrsituation
- Beherrschbarkeit der Fehlfunktion in der jeweiligen Fahrsituation

Anhand dieser Eigenschaften kann dann auf Basis einer in ISO 26262 enthaltenen Tabelle ein Sicherheitsintegritätslevel für Automobile (ASIL, Automotive Safety Integrity Level) bestimmt werden. Für jedes betrachtete Ereignis soll dann entsprechend des zugeordneten ASILs mindestens ein Sicherheitsziel definiert werden. Abschließend hat eine Verifizierung der Analyse zu erfolgen, die nachweist, ob alle Ereignisse und Gefahren abdeckend betrachtet wurden, diese mit der Item-Definition übereinstimmen und ob der Bezug von ASIL zum Gefahrenereignis konsistent ist. Ein Vergleich der ASIL mit SIL und kerntechnischen Sicherheitskategorien erfolgt in Kapitel 4.5.

Nach DIN EN 50126 sollen im Rahmen einer Risikoanalyse die Gefahren, die mit einem System verbunden sind, und die Ereignisse, die diese Gefahren auslösen, identifiziert werden. Außerdem soll das mit den Gefahren verbundene Risiko ermittelt werden. Dazu soll die Häufigkeit des Eintretens und das Ausmaß der Auswirkungen der jeweiligen Gefahren ermittelt werden. Abschließend soll ein Prozess für ein kontinuierliches Risikomanagement erstellt werden. Als Basis für das Risikomanagement soll ein Gefahrenprotokoll erstellt werden, an dessen Inhalt Anforderungen gestellt werden.

Im nuklearen Bereich ist diese Phase nicht im Anwendungsbereich der DIN EN 61513 enthalten.

4.1.4 Zuordnung der Sicherheitsanforderungen

In DIN EN 61508 wird gefordert, dass anhand der Ergebnisse der Gefahren- und Risikobewertung die Sicherheitsanforderungen für das System in Hinblick auf Sicherheitsfunktionen und Sicherheitsintegrität spezifiziert werden. Diese Anforderungen sollen dann in einem nächsten Schritt den dafür vorgesehenen E/E/PE-Systemen zugeordnet werden. Diese Forderung wird in ISO 26262 aufgegriffen und in Form eines funktionalen Sicherheitskonzepts umgesetzt. Im Rahmen dessen sollen nach ISO 26262 anhand der Ergebnisse der Klassifizierung von Ereignissen, ihrer Zuordnung zu ASIL und der Definition von Sicherheitszielen Sicherheitsanforderungen abgeleitet und diese im vorläufigen architektonischen Design des Items bestimmten Teilsystemen zugeordnet werden. Im Gegensatz zur DIN EN 61508 wird formuliert, welche Aspekte im Rahmen dieses Konzepts betrachtet und berücksichtigt werden sollen. Beispielsweise wird gefordert, dass es für jedes definierte Sicherheitsziel mindestens eine funktionale Sicherheitsanforderung geben muss. Kann innerhalb eines bestimmten Zeitraums kein sicherer Zustand erreicht werden, muss für die entsprechende Situation eine Notfallmaßnahme spezifiziert werden. Es muss ein Konzept für Warnsysteme geben, um den Fahrer bei Ausfällen oder Störungen rechtzeitig zu warnen.

Während DIN EN 61508 erst nach Zuordnung von Sicherheitsanforderungen zu Teilsystemen eine Zuordnung zu SIL fordert, erfolgt dies in ISO 26262 bereits im Rahmen der Risiko- und Gefahrenanalyse. Hier erfolgt eine ASIL-Zuordnung demnach zunächst unabhängig vom betroffenen Teilsystem.

Nach DIN EN 50126 sollen in dieser Phase die RAMS[1]-Anforderungen an das jeweilige System und die Nachweis- und Abnahmekriterien spezifiziert werden. Anschließend soll ein Programm für die Überwachung der RAM-Aufgaben während der folgenden Lebenszyklusphasen erstellt werden. Dazu wird genau beschrieben, welche Informationen die Spezifikationen enthalten müssen und was bei dem Überwachungsprogramm enthalten sein muss. Sind die funktionalen und die Sicherheitsanforderungen den entsprechenden Subsystemen, Komponenten und externen Einrichtungen zugeordnet worden, soll abschließend eine Überprüfung des Validierungs- und Sicherheitsplans erfolgen, um zu gewährleisten, dass die geplanten Maßnahmen auch den Systemanforderungen nach der Zuteilung entsprechen.

[1] RAMS: Reliability, Availability, Maintainability, Safety

Der Lebenszyklus nach DIN 60513 beginnt mit dieser Phase. Zur Bestimmung der Sicherheitsanforderungen werden zunächst die Leittechnikanforderungen aus der sicherheitstechnischen Auslegung der Anlage abgeleitet. Anschließend erfolgt eine Überprüfung der in der Auslegung festgelegten Kategorisierung der leittechnischen Funktionen, Systeme und Geräte. Es werden Anforderungen daran formuliert, welche Randbedingungen, Gefährdungen und Schnittstellen im Rahmen der Überprüfung betrachtet werden sollen. Anhand dessen werden dann weitere Anforderungen hinsichtlich der Unterteilung der leittechnischen Architektur in separate Teilsysteme formuliert.

4.1.5 Gesamtplanung

In DIN EN 61508 werden Anforderungen hauptsächlich dahingehend formuliert, dass alle Aktivitäten derart zu planen sind, dass sich bei der Planung und Durchführung möglichst keine negativen Einwirkungen auf das System ergeben. Dazu wird beispielsweise ausgeführt, welche Aspekte bei der Planung der Sicherheitsvalidierung und auch bei der Instandhaltung zu berücksichtigen sind. Während sich die Anforderungen in DIN EN 61508 zum Thema Planung also schwerpunktmäßig mit dem Erhalt eines sicheren Zustands des Systems beschäftigen, geht die ISO 26262 mehr auf die Planung der Serienproduktion und den Betrieb des Items im Fahrzeug ein. Beispielsweise beschreibt die ISO 26262 detailliert, welche Informationen ein Nutzerhandbuch enthalten soll, während DIN EN 61508 lediglich allgemein eine Dokumentation zum betrachteten System fordert.

In DIN EN 50126 gibt es keine separate Phase für die Planung. Planungsaufgaben werden in den nachfolgenden Phasen an unterschiedlichen Stellen durchgeführt.

Auch DIN EN 61513 formuliert Anforderungen an die Planung. Dabei beschäftigt sie sich im Wesentlichen mit dem Qualitätssicherungsprogramm, der Planung des Zugriffsschutzes, der Planung der Gesamtintegration und -inbetriebnahme der Leittechnik, dem Gesamtbetriebsplan, dem Instandhaltungsplan und der Planung des Trainings.

4.1.6 Realisierung

Grundsätzlich unterscheiden sich die betrachteten Normen dahingehend, dass es sich in DIN EN 61508 (Allgemein) um die Realisierung eines E/EE/EP/-Systems, bei DIN EN 61513 (Kerntechnik) um die Realisierung eines Systems und bei DIN EN 50126 (Bahn) grundsätzlich um die Fertigung handelt. In ISO 26262 (Auto) wird erst nach Ablauf der

abschließenden Bewertung der funktionalen Sicherheit auf die Freigabe und die Durchführung der Serienproduktion eingegangen.

In DIN EN 61508 (allgemein) wird bei der Realisierung eines E/E/PE-Systems auf die Teile der Norm verwiesen, welche Anforderungen an Software und Hardware enthalten. Der Schwerpunkt liegt hier auf der Entwicklung und der anschließenden Integration. Diese Themen sind in Abschnitt 4.1.5 und Abschnitt 4.1.7 behandelt.

In DIN EN 61513 (Kerntechnik) sollen im Rahmen der Realisierung die Rechenprogramme entwickelt bzw. beschafft sowie die Anwendungssoftware entwickelt und programmiert werden. Es wird diesbezüglich auf Anforderungen in DIN IEC 60880 /DIN 07/ verwiesen, die in Abschnitt 4.2 behandelt werden.

Nach DIN EN 50126 (Bahn) hat die Fertigungsphase das Ziel, einen Fertigungsprozess zu implementieren, der RAMS-validierte Subsysteme und Komponenten erzeugt, sowie einen RAMS-bezogene Prozesssicherung und Support-Vorkehrungen für Subsysteme und Komponenten zu etablieren. Zu den Support-Maßnahmen gehören die Vorbereitung, Verifizierung und Validierung der zugehörigen Dokumente sowie der Betriebs- und Instandhaltungsverfahren und der Schulungsunterlagen.

Vor Freigabe zur Serienproduktion hat nach ISO 26262 die Verifizierungs- und Validierungsphasen und die Bewertung der funktionalen Sicherheit zu erfolgen. Das zu erstellende Freigabedokument dient dann als Basis für die Produktion. Die Produktion ist zunächst zu planen. Dieser Plan soll die verschiedenen Produktionsschritte beschreiben, sowie die Abläufe und Werkzeuge, um die funktionale Sicherheit zu erreichen. Um sicherzustellen, dass die korrekte eingebettete Software und die dazugehörigen Kalibrierungsdaten als Teil des Produktionsprozesses geladen werden, soll eine entsprechende Prozedur erstellt werden. Darüber hinaus soll ein Kontrollplan erstellt werden, welcher die Vorgehensweisen und Methoden zur Kontrolle der Produktion beschreiben. Vorhersehbare Prozessausfälle sollen identifiziert und entsprechende Maßnahmen entwickelt werden, um diesen Ausfällen vorzubeugen. Der Produktionsprozess unterteilt sich in eine Vorproduktion und die eigentliche Serienproduktion. Die Anforderungen zielen im Wesentlichen darauf ab, Unterschiede zwischen den beiden Prozessen zu identifizieren und zu bewerten sowie mögliche vorhersehbare Ausfallarten zu identifizieren und entsprechende Maßnahmen einzurichten.

4.1.7 Installation und Inbetriebnahme des Gesamtsystems

Während diese Phase des Lebenszyklus für das E/E/PE-System in DIN EN 61508 Anforderungen an die Planung und Entwicklung der Inbetriebnahme über die Inbetriebnahme bis hin zu Modifikation und Instandhaltung beinhaltet, endet in ISO 26262 diese Phase bei der Freigabe des entsprechenden Produkts für die Serienproduktion. In beiden Normen werden an dieser Stelle ausführliche Anforderungen an Hardware- und Softwarekomponenten gestellt. Während in DIN EN 61508 immer ein System und seine Entwicklung bis hin zu Modifikationen und Instandhaltung betrachtet wird, bezieht sich diese Phase der ISO 26262 ausschließlich auf die Erzeugung eines Prototyps. Auch Prüfungen, Verifizierungen und Integration ins Fahrzeug werden hier gefordert, beziehen sich jedoch auch immer nur auf den Prototyp. Kann anhand des Prototyps nachgewiesen werden, dass der betrachtete Gegenstand ins Fahrzeug integriert werden kann und alle Sicherheitsanforderungen erfüllt, kann er für die Serienproduktion freigegeben werden.

DIN EN 61508 bezieht sich immer auf das tatsächlich verbaute System, welches dann im weiteren Verlauf modifiziert und getestet werden kann. Ein Unterschied zwischen DIN EN 61508 und ISO 26262 ist, dass in ISO 26262 eine Sicherheitsvalidierung bereits im Rahmen der Produktentwicklung vor Freigabe für die Serienproduktion gefordert wird, während in DIN EN 61508 eine Sicherheitsvalidierung erst nach Installation und Inbetriebnahme des Systems zu erfolgen hat.

Nach DIN EN 50126 werden in dieser Phase die Subsysteme und Komponenten entwickelt und konstruiert. Anschließend muss ein Nachweis erbracht werden, dass sie die RAMS-Anforderungen erfüllen. Dazu soll ein grundsätzlicher Systemsicherheitsnachweis erarbeitet werden, durch den nachgewiesen wird, dass das System in der Gestaltung und unabhängig von der Anwendung den Sicherheitsanforderungen genügt. Basierend auf dem Sicherheitsnachweis werden dann die nachfolgenden Lebenszyklusaufgaben im Zusammenhang mit RAMS geplant. Dies beinhaltet auch die Erstellung eines Produktionsverfahrens, das in der Lage ist, RAMS-validierte Subsysteme und Komponenten zu erzeugen.

Nach DIN EN 50126 werden alle Subsysteme, Komponenten und externe Einrichtungen in Übereinstimmung mit dem Montageplan montiert und zusammengebaut, die erforderlich sind, um das Gesamtsystem zu bilden. Es werden Anforderungen an die Dokumentation der Montage formuliert. Abschließend wird gefordert, dass der Sicherheitsplan nach Abschluss der Montage zu überprüfen und anzupassen ist, um zu gewährleisten,

dass jegliche Änderung, entweder am System oder an den Verfahren, aufgezeichnet ist und in künftigen Lebenszyklusaufgaben wirkungsvoll gehandhabt wird. Darüber hinaus soll mit der Schulung des Personals begonnen sowie die Lagerhaltung von Ersatzteilen und Werkzeugen festgelegt werden.

DIN EN 61513 fordert in dieser Phase, dass die Tätigkeiten entsprechend einer Strategie systematisch durchgeführt werden sollen, die in Übereinstimmung mit den Errichtungsplänen des Systems, den Gesamtintegrations- und Inbetriebnahmeplänen und der Zugriffsschutzplanung entwickelt wurde. Die Gesamtintegration soll erst durchgeführt werden, nachdem alle betroffenen leittechnischen Systeme errichtet und einzeln geprüft wurden. Software und Datenbasen mit Parametern müssen geladen sein und die gespeicherten Werte müssen geprüft und verifiziert sein. Hardware und Software müssen dem Konfigurationsmanagement unterliegen. Verifizierung und Validierung aller sicherheitstechnisch wichtigen Funktionen müssen abgeschlossen sein, bevor diese Funktionen in Betrieb genommen werden.

4.1.8 Validierung der Sicherheit des Gesamtsystems

Nach DIN EN 61508 muss eine Validierung erfolgen, um zu zeigen, dass das sicherheitsrelevante E/E/PE-System der Spezifikation entspricht. Dazu wird gefordert, dass Validierungsaktivitäten durchgeführt werden und dass alle Gerätschaften, die für eine quantitative Messung als Teil der Validierungsaktivitäten genutzt werden, entsprechend der Spezifikationen eines nationalen Standards oder der Spezifikationen des Lieferanten kalibriert werden. Außerdem werden Anforderungen an die Dokumentation während der Validierung beschrieben. Auch Abweichungen zwischen erwarteten und tatsächlichen Ergebnissen und über die Entscheidungen für das weitere Vorgehen sind zu dokumentieren.

In ISO 26262 ist das Ziel dieser Phase der Nachweis, dass die Sicherheitsziele eingehalten werden und dass das funktionale Sicherheitskonzept für das Gesamtsystem angemessen ist. Außerdem soll gezeigt werden, dass die Sicherheitsziele korrekt und vollständig sind und auf Fahrzeugebene erfüllt werden. Es wird gefordert, dass die Sicherheitsziele für das Gesamtsystem, welches in einem repräsentativen Fahrzeug integriert ist, validiert werden. Außerdem wird beschrieben, welche Eigenschaften im Rahmen der Validierung überprüft werden sollen. Werden Prüfungen zur Validierung durchgeführt, sollen die gleichen Anforderungen an die Prüfungen wie bei der Verifizierung gelten.

Ziel dieser Phase ist in DIN EN 50126 die Validierung und Inbetriebsetzung der gesamten Subsysteme, Komponenten und externen Einrichtungen, die Vorbereitung und ggf. Abnahme des anwendungsspezifischen Sicherheitsnachweises des Systems sowie die Durchführung der Datenerfassung und Auswertung. Für alle Aufgaben werden Anforderungen an die Dokumentation formuliert. Abschließend werden Anforderungen an die Systemabnahme formuliert, welche nach DIN EN 50126 eine weitere Phase des Lebenszyklus darstellt.

Nach DIN EN 61513 ist das Ziel dieser Phase die Prüfung des integrierten Systems, um Übereinstimmung mit den Spezifikationen bezüglich Funktionalität, Leistungsfähigkeit und Schnittstellen nachzuweisen. In dieser Phase müssen Prüfungen durchgeführt werden, um das System und seine Software, die Programmierungs- und Konfigurationsdaten zu validieren und die Übereinstimmung mit den Systemanforderungen nachzuweisen. Diese Prüfungen müssen am System in der endgültigen Konfiguration einschließlich der endgültigen Version der Software und anderen Programmierungsdaten durchgeführt werden.

Anforderungen an die Validierung werden in Kapitel 5 behandelt.

4.1.9 Betrieb, Instandhaltung und Reparatur des Gesamtsystems

In ISO 26262 werden speziell für die Serienproduktion Anforderungen gestellt, nach denen Prozesse zu entwickeln sind, um während der Produktion die funktionale Sicherheit jederzeit zu gewährleisten. Ein besonderer Fokus wird hier auf mögliche Unterschiede in der Prototypentwicklung und der eigentlichen Serienproduktion gelegt. Solche Unterschiede müssen erkannt und bewertet werden, um eine Übertragbarkeit der Prozesse für die Prototyperzeugung auf die Serienproduktion zu ermöglichen. Auch sollen Prozessausfälle während der Serienproduktion analysiert werden, um mögliche Auswirkungen auf die funktionale Sicherheit zu identifizieren und zu bewerten. DIN EN 61508 geht auf diese Art der Produktion nicht ein, sondern fordert im Wesentlichen, dass den verantwortlichen Personen für den Betrieb, Instandhaltung und bei Modifikationen alle notwendigen Anforderungen zur Verfügung gestellt werden. Schwerpunkt der Anforderungen liegt darauf, bei Nachrüstungen und Instandhaltung des betrachteten Systems die geforderte funktionale Sicherheit aufrechtzuerhalten.

In DIN EN 50126 wird in der Phase die Überwachung der Ausführung des Systems und der Betriebs- und Instandhaltungsverfahren sowie die langfristige Erfüllung der RAMS-

Anforderungen an das System gefordert. In einer anschließenden Phase werden Anforderungen an die Erfassung der Leistungsfähigkeit des Systems formuliert. Dazu soll ein Verfahren für das Zusammentragen, die Analyse und Bewertung der Leistungserfüllung erstellt werden.

Nach DIN EN 61513 darf der Betrieb der leittechnischen Systeme beginnen, wenn die Auswertung der Inbetriebnahmeberichte einen erfolgreichen Abschluss der Inbetriebnahmetätigkeiten zeigt. Die leittechnischen Systeme sollen dann so betrieben und instandgehalten werden, dass die Einhaltung der Anforderungen an die sicherheitstechnisch wichtigen leittechnischen Funktionen erhalten bleibt.

4.1.10 Modifikation und Nachrüstung des Gesamtsystems

Nach DIN EN 61508 sind vor einer Modifikation oder Nachrüstung Prozeduren zu planen. Es soll eine Auswirkungsanalyse durchgeführt werden, die eine Bewertung der Auswirkungen der Modifikation oder Nachrüstung beinhaltet. Die Bewertung soll auch eine Risiko- und Gefahrenanalyse beinhalten und den Einfluss anderer konkurrierender Modifikations- oder Nachrüstmaßnahmen berücksichtigen. Bei allen Modifikationen oder Nachrüstungen, die die funktionale Sicherheit beeinflussen, soll zur entsprechenden Lebenszyklusphase zurückgekehrt und alle nachfolgenden Phasen erneut betrachtet werden.

Diese Phase ist in ISO 26262 bereits die zweite Phase im Lebenszyklus, da Änderungen immer auch Auswirkungen auf die Serienproduktion haben und entsprechend früh berücksichtigt werden müssen. Im Falle von Änderungen soll eine Auswirkungsanalyse durchgeführt werden. Diese Analyse soll aufzeigen, welche Gebiete von den Änderungen betroffen sind. Es sollen zudem die Betriebsmodi und Betriebssituationen, die Schnittstellen zur Umgebung und die Installationscharakteristika wie beispielsweise der Einbauort im Fahrzeug, die Fahrzeugkonfiguration und -variante berücksichtigt werden. Abschließend soll der Einfluss der Änderung auf die funktionale Sicherheit identifiziert und beschrieben werden.

Nach DIN EN 50126 muss bei Nachrüstungen oder Änderungen des Systems zunächst ein Sicherheitsplan erstellt werden. Anschließend werden Anforderungen daran formuliert, was bei Änderungen oder Nachrüstungen zu beachten und zu kontrollieren ist.

Nach DIN EN 61513 muss die Realisierung von Systemänderungen in Übereinstimmung mit festgelegten Prozeduren erfolgen. Nach erfolgter Änderung ist eine Prüfung des einwandfreien Arbeitens des Systems durchzuführen. Wird Ersatzhardware angefordert, muss nachgewiesen werden, dass der Ersatz die Spezifikation der ursprünglichen Hardware erfüllt.

4.1.11 Stilllegung und Entsorgung

Nach DIN EN 61508 müssen auch für den Abbau und die Entsorgung des Systems Prozeduren definiert werden. Vor jeder Außerbetriebnahme oder Entsorgung muss eine Auswirkungsanalyse durchgeführt werden, die auch eine Bewertung der Auswirkungen auf die funktionale Sicherheit jedes E/E/PE-Systems beinhaltet. Wird eine Auswirkung ermittelt, soll in die betroffene Phase zurückgekehrt und alle nachfolgenden Phasen erneut bearbeitet werden.

Nach ISO 26262 sollen Instandhaltung, Reparatur und Entsorgung von Gesamtsystemen, ihren Teilsystemen oder Elementen verwaltet und dokumentiert werden.

Nach DIN EN 50126 sollen zunächst die Auswirkungen der Stilllegung und Entsorgung auf Systeme oder externe Einrichtungen, die mit dem stillzulegenden System in Verbindung stehen, festgestellt werden. Anschließend soll die Stilllegung geplant werden.

Im nuklearen Bereich ist diese Phase nicht im Anwendungsbereich der DIN EN 61513 enthalten.

4.2 Software-Sicherheitslebenszyklus

Nachdem im letzten Abschnitt zunächst auf den System-Sicherheitslebenszyklus in den verschiedenen Industriezweigen eingegangen wurde, wird im Folgenden der Software-Sicherheitslebenszyklus der verschiedenen Industriezweige verglichen. Hierfür wurden die in Tab. 4.3 genannten, in Anhang B beschriebenen Normen betrachtet.

Tab. 4.3 Für den Vergleich des Software-Sicherheitslebenszyklus herangezogene Normen

Bereich	Norm	Titel
Allgemein	DIN EN 61508, Teil 3	Funktionale Sicherheit sicherheitsbezogener elektrischer/ elektronischer/ programmierbarer elektronischer Systeme, Teil 3: Software-Anforderungen
Kerntechnik	DIN EN 60880	Kernkraftwerke – Leittechnik für Systeme mit sicherheitstechnischer Bedeutung: Softwareaspekte für rechnerbasierte Systeme zur Realisierung von Funktionen der Kategorie A
Auto	ISO 26262, Teil 6	Road Vehicles – Functional Safety, Part 6: Product development at the software level
Bahn	DIN EN 50128	Bahnanwendungen – Telekommunikationstechnik, Signaltechnik und Datenverarbeitungssysteme – Software für Eisenbahnsteuerungs- und Überwachungssysteme

DIN EN 50128 und DIN EN 61508 lassen die Wahl des Software-Sicherheitslebenszyklus offen, sofern er die Anforderungen der entsprechenden Norm umsetzt. ISO 26262 gibt einen konkreten Lebenszyklus vor. In der Kerntechnik wird das V-Modell (siehe Anhang C) für die Vorgehensweise bei der Software-Entwicklung gefordert, da dieses bereits durch IAEA NS-G-1.3 /IAEA 02/ aufgegriffen und verbreitet wurde sowie notwendige Anpassungen erlaubt.

In Tab. 4.4 ist der Software-Sicherheitslebenszyklus in den betrachteten Industriezweigen vergleichend aufgeführt. Auch hier ist der grundsätzliche Ablauf vergleichbar. Im Folgenden wird auf Anforderungen zu verschiedenen Lebenszyklus-Phasen eingegangen, wobei der Fokus auf den Unterschieden zwischen den betrachteten Industriezweigen liegt.

Ergänzend zu den einzelnen Phasen des Software-Sicherheitslebenszyklus‘ werden zunächst allgemeine Anforderungen an das Management und die Organisationsstruktur betrachtet.

Tab. 4.4 Software-Lebenszyklus in verschiedenen Industriezweigen

Allgemein	**Kerntechnik**	**Bahn**	**Auto**
Grundsätzliche Anforderungen an den Lebenszyklus	Allgemeine Anforderungen für Software-Projekte - Softwaretypen - Vorgehensweise zur SW-Entwicklung - SW-Projektmanagement - SW-QS-Plan - Konfig.-Management - Zugriffsschutz		Einleiten der Produktentwicklung auf SW-Ebene
Spezifikation der SW-Anforderungen	SW-Anforderungen - Spezifikation - Selbstüberwachung - Periodische Prüfungen - Dokumentation	SW-Anforderungen	Spezifikation der SW-Sicherheitsanforderungen
Validierungsplan für SW-Aspekte der System-Sicherheit SW-Design und Entwicklung	Auslegung und Implementierung - Prinzipien - Sprachen, Übersetzer, Werkzeuge	SW-Architektur SW-Entwurf SW-Komponentenentwurf	SW-Architekturentwurf SW-Komponentenentwurf
	SW-Verifizierung	Komponenten-Implementierung und Test	Implementierung der SW-Komponente Test der SW-Komponente
Integration von programmierbarer Elektronik	SW-Aspekte der System-Integration	SW-Integration	SW-Integration und Test
SW-Betrieb und Modifikationsprozeduren			
SW-Validierung	SW-Aspekte der Systemvalidierung	SW-Validierung	
SW-Modifikation	SW-Modifizierung		
	Installation der SW auf der Anlage	SW-Bereitstellung	
SW-Verifizierung		SW-Wartung	Verifizierung
Bewertung funktionaler Sicherheit der SW			

4.2.1 Management und Organisationsstruktur

DIN EN 50128 geht detailliert auf Management- und Organisationsstrukturen in Hinblick auf Software ein. Es werden in diesem Zusammenhang Anforderungen an Lieferanten, Gutachter, Validierer und an die Kompetenz der Mitarbeiter gestellt. Außerdem wird eine bevorzugte Organisationsstruktur aufgezeigt. In einem Anhang wird für jede Rolle aufgeführt, welche Verantwortlichkeiten mit dieser Position verbunden sind und welche Kompetenzen aufzuweisen sind. Diese Themen werden in den anderen Industriezweigen in diesem Detailgrad nicht behandelt.

In DIN EN 60880 wird in Hinblick auf das Thema Management auf das Software-Projektmanagement, den Software-Qualitätssicherungsplan und das Konfigurationsmanagement eingegangen. Speziell auf das Thema Konfigurationsmanagement gehen auch DIN EN 61508 und ISO 26262 ein, wobei letztere für das Konfigurationsmanagement für die Software-Entwicklung auf Anforderungen in ISO IEC 12207 /ISO 08/ verweist. DIN EN 61508 beschäftigt sich neben dem Konfigurationsmanagement mit dem Management der funktionalen Sicherheit. Hierbei werden Verantwortlichkeiten für ein E/E/PE-System definiert. Im Gegensatz zu DIN EN 50128, in der für jeden Posten sehr klare Anforderungen formuliert werden, sind die Anforderungen in DIN EN 61508 eher allgemeiner Natur.

4.2.2 Lebenszyklus, allgemeine Anforderungen

In DIN EN 50128 und 61508 wird allgemein die Anwendung eines Lebenszyklusmodells für die Software-Entwicklung gefordert. In beiden Normen wird nicht explizit vorgegeben, welches Modell anzuwenden ist. Es soll aber die in der jeweiligen Norm genannten Anforderungen erfüllen. In beiden Normen wird als Ziel des Einsatzes eines Lebenszyklus die Softwareentwicklung in festgelegten Phasen und Aktivitäten formuliert. DIN EN 50128 liefert zwei mögliche Beispiele eines Lebenszyklus. In ISO 26262 wird ein Lebenszyklusmodell vorgegeben, welches auf den Einsatz in der Softwareentwicklung angepasst werden soll. In DIN IEC 60880 (Kerntechnik) wird aktuell das V-Modell als Lebenszyklusmodell empfohlen.

DIN EN 50128 gibt in einer Tabelle detailliert vor, welche Dokumente im Rahmen des Lebenszyklus SIL-abhängig erstellt werden müssen. Auf das Thema Dokumentation wird in Kapitel 4.3 eingegangen. Für die Reihenfolge der zu erstellenden Dokumente

wurde als Lebenszyklusmodell ein lineares Wasserfallmodell (siehe Anhang C) zugrunde gelegt. Es wird jedoch hier noch mal explizit darauf hingewiesen, dass die Reihenfolge an das angewandte Lebenszyklusmodell angepasst werden kann. In Anhang C werden drei weit verbreitete Phasenmodelle beschrieben und ihre Vor- und Nachteile erläutert.

4.2.3 Software-Anforderungsspezifikation

DIN EN 50128, ISO 26262 und DIN IEC 60880 enthalten konkrete Anforderungen daran, welche Informationen in der Software-Anforderungsspezifikation enthalten sein müssen. Diese Normen fordern, dass die Betriebsarten der Software, die Verhaltensweisen, speziell das Ausfallverhalten sowie die Schnittstellen zu anderen Systemen beschrieben werden müssen. DIN EN 50128 fordert darüber hinaus, dass Schnittstellen zwischen Hardware und Software beschrieben, Software-Funktionen mit und ohne Sicherheitsanforderungen gekennzeichnet und Software-Selbsttests und Hardware-Prüfungen durch Software berücksichtigt werden müssen. DIN IEC 60880 geht speziell auf das Thema Selbsttests separat ein. Zusätzlich wird gefordert, dass für Parameter der Software, die während des Betriebs von Hand geändert werden können, ihre Aufgaben, Typen, Bereiche und Randbedingungen zu beschreiben sind. Darüber hinaus muss nach DIN IEC 60880 beschrieben werden, was die Software nicht darf oder was sie verhindern muss. DIN IEC 60880 weist bei der Software-Anforderungsspezifikation explizit darauf hin, dass dieser Phase im Lebenszyklus eine hohe Bedeutung zukommt und daher konsequent durchzuführen ist.

DIN EN 50128 gibt an, dass die Software-Anforderungsspezifikation durch folgende Techniken und Maßnahmen unterstützt werden muss. Hieraus muss eine Kombination von Techniken ausgewählt und die Auswahl begründet werden:

- Formale Verfahren (beruhend auf einem mathematischen Ansatz)
- Modellierung
- Strukturierte Verfahren
- Entscheidungstabellen

4.2.4 Entwurf, Architektur, Entwicklung und Implementierung

DIN EN 61508, DIN EN 50128 und ISO 206262 unterteilen die Phasen zwischen der Anforderungsspezifikation und der Integration in die Phasen

- Architektur und Entwurf,
- Komponentenentwurf,
- Implementierung und Test der Softwarekomponenten.

Zunächst soll nach DIN EN 50128 eine Software-Architektur entwickelt werden, die die an die Software zu stellenden Anforderungen erfüllt. Wechselwirkungen zwischen Hardware und Software sollen identifiziert und ihre Bedeutung für die Sicherheit bewertet werden. Anschließend soll ein Entwurfsverfahren ausgewählt und die Software entsprechend ihrem Software-SIL entworfen werden. Dieses Entwurfsverfahren muss unter anderem eine klare Darstellung von Funktionalität, Informationsfluss zwischen Komponenten, Parallelverarbeitung und Datenstrukturen und -eigenschaften unterstützen. Die gesamte Software muss in einzelne Software-Komponenten zerlegt werden. Für jede dieser Komponenten muss eine Software-Komponentenentwurfsspezifikation erstellt werden. Darüber hinaus müssen Codierstandards festgelegt werden. Auf diese wird in Kapitel 4.3.3 eingegangen. Nach Abschluss des Entwurfs wird eine Software erstellt, die analysierbar, testbar, verifizierbar und wartbar sein soll. Diese Phase umfasst dann auch den Test der entwickelten Softwarekomponenten.

Wie im Schienenverkehr soll nach ISO 26262 zunächst ein Architekturentwurf der Software erstellt werden, welcher die Anforderungsspezifikation erfüllt. Dieser Entwurf soll anschließend verifiziert werden. Dieser Architekturentwurf stellt alle Software-Komponenten und ihre Wechselwirkungen in einer hierarchischen Struktur dar. Für die Beschreibung des Software-Architekturentwurfs sollen je nach ASIL informelle, semi-formelle oder formale Notationen verwendet werden. Um Ausfälle aufgrund der hohen Komplexität von Software zu vermeiden, sollen Prinzipien angewandt werden, um Eigenschaften wie Modularität und Einfachheit sicherzustellen. Zu diesen Prinzipien zählen:

- Hierarchische Struktur der Software-Komponenten,
- Begrenzte Größe der Software-Komponenten,
- Begrenzter Umfang der Schnittstellen,

- Begrenzte Kopplung zwischen Software-Komponenten,
- Begrenzter Einsatz von Interrupts,
- Angemessene Eigenschaften der Planung des zeitlichen Ablaufs.

Entsprechend des Architekturentwurfs sollen im nächsten Schritt ähnlich wie im Schienenverkehr die Software-Komponenten spezifiziert und der entsprechende Entwurf implementiert werden. Anschließend soll auch hier die Implementierung verifiziert werden. Im Rahmen von Tests soll gezeigt werden, dass die Software-Komponenten die Entwurfsspezifikation erfüllen und keine unerwünschten Funktionalitäten aufweisen.

Auch in DIN EN 61508 werden in dieser Phase Anforderungen an das Entwurfsverfahren und den Software-Architekturentwurf gestellt. Im Gegensatz zu ISO 26262 und DIN IEC 60880 geht DIN EN 61508 zusätzlich auf Werkzeuge und Programmiersprachen ein. Programmiersprachen sollen beispielsweise einem geeigneten Codierstandard entsprechen. Konkrete Anforderungen an den Codierstandard werden jedoch nicht gestellt. Hierauf wird in Kapitel 4.3.3 eingegangen. Anschließend werden Anforderungen an den Software-Entwurf formuliert. Auch hier wird eine Unterteilung der Software in Software-Komponenten gefordert. Abschließend werden Anforderungen an die Tests der Software-Komponenten beschrieben.

In DIN EN 60880 wird die Phase nach der Erstellung der Software-Anforderungsspezifikation etwas anders bezeichnet und strukturiert als in den anderen Industriezweigen. Hier liegt der Schwerpunkt der Anforderungen neben der Implementierung auf der Auslegung der Software. Im Gegensatz zur Automobilindustrie und zum Schienenverkehr, wo grundsätzliche und allgemeine Anforderungen formuliert werden, wird in DIN EN 60880 genau beschrieben, worauf bei der Auslegung und der Implementierung konkret zu achten ist. Beispielsweise wird gefordert, dass die Software modifizierbar sein muss, wie eine Änderungskontrolle erfolgen muss und was bei der Software-Struktur zu beachten ist. Auch werden Anforderungen an die Software-Module gestellt. Hierzu zählt beispielsweise, dass die Module klar und verständlich sein müssen, jedes Modul einer bestimmten Funktion entspricht und nur eine begrenzte Größe aufweist. Die Schnittstellen zwischen Modulen sollten so einfach wie möglich sein. Darüber hinaus werden Anforderungen an die Betriebssoftware, die Verwendung von Interrupts und arithmetischen Ausdrücken, der Selbstüberwachung und der Speicherinhalte formuliert.

4.2.5 Integration

Nach DIN EN 50128 wird in dieser Phase die Integration der Software ausgeführt und nachgewiesen, dass die Software mit der Hardware richtig zusammenwirkt, um ihre vorgesehene Funktion auszuführen. Dazu wird gefordert, dass die Integration der Software-Komponenten ein Prozess der fortschreitenden Kombination einzelner und vorher getesteter Komponenten zu einem verbundenem Ganzen sein muss, damit die Komponentenschnittstellen und die zusammengefügte Software vor der Systemintegration und dem Systemtest angemessen geprüft werden können. Während der Integration muss jede Modifizierung oder Veränderung des integrierten Systems einer Auswirkungsanalyse unterzogen werden, in der alle betroffenen Komponenten und die erforderlichen Aktivitäten zur Reverifikation identifiziert werden. Darüber hinaus werden Anforderungen an die in dieser Phase zu erstellenden Berichte formuliert.

Auch in ISO 26262 besteht diese Phase aus der Integration der Software-Komponenten und der Demonstration, dass der Software-Architekturentwurf korrekt umgesetzt wurde. Dies kann sowohl sicherheitsrelevante als auch nicht sicherheitsrelevante Software-Komponenten beinhalten. Es werden im Wesentlichen Anforderungen an die Planung der Integration und die Testmethoden formuliert. Für die Softwareintegration werden folgende Testmethoden genannt:

- Anforderungsbasierender Test,
- Schnittstellentest,
- Fehlerinjektionstest,
- Ressourcen-Verbrauchstest,
- Vergleichstest zwischen Modell und Code, falls möglich.

DIN EN 61508 formuliert in dieser Phase im Wesentlichen Anforderungen an die Integrationstests. Diese Tests sollen bereits in der Entwurfs- und Entwicklungsphase spezifiziert werden. Auch hier wird gefordert, dass sämtliche Änderungen während der Integration einer Auswirkungsanalyse unterzogen werden sollen, um zu bestimmen, welche Software-Module betroffen sind und einer Reverifizierung unterzogen werden müssen. Die Ergebnisse der Integrationstests müssen dokumentiert werden.

DIN IEC 60880 betrachtet die Software-Integration als Teil der Systemintegration. Daher werden Anforderungen an die Softwareaspekte im Systemintegrationsplan formuliert. Die Anforderungen beziehen sich daher immer sowohl auf Hardware als auch auf Software.

4.2.6 Verifizierung

Bei der Verifizierung liegt der Schwerpunkt der Anforderungen in der kerntechnischen Norm DIN EN 60880 beim Verifizierungsteam. Darüber hinaus werden Anforderungen an die verschiedenen Verifizierungstätigkeiten gestellt. Hierzu zählen die Erstellung eines Verifizierungsplans, die Verifizierung der Auslegung und der Realisierung.

Auch in DIN EN 50128 werden u.a. Anforderungen an den Verifizierungsplan gestellt. Hier wird zusätzlich zu den Anforderungen in der Kerntechnik gefordert, dass die Rollen und Verantwortlichkeiten der am Prüfprozess beteiligten Personen in der Prüfspezifikation festgeschrieben werden müssen. Zusätzlich wird außerdem gefordert, dass die Prüfungen wiederholbar sein müssen und falls praktikabel mit automatischen Hilfsmitteln durchgeführt werden. Die Test-Skriptfiles für die automatische Testausführung müssen verifiziert werden.

Anforderungen zur Verifizierung werden in Kapitel 5 behandelt.

4.2.7 Validierung

In DIN EN 50128 wird bei diesem Punkt ausschließlich auf den Inhalt der geforderten Berichte eingegangen. Hierzu gehören ein Software-Validierungsplan, ein Software-Validierungsbericht und ein Software-Validierungs/Verifizierungsbericht.

Auch DIN IEC 60880 gibt Vorgaben zu einem Validierungsplan. Hier wird explizit gefordert, dass die Validierung von Personen durchgeführt werden muss, die an den Auslegungs- und Realisierungsarbeiten nicht beteiligt waren. Darüber hinaus wird gefordert, dass statische und dynamische Testfälle im Validierungsplan enthalten sein müssen. In DIN EN 50128 werden hier mehrere Auswahlmöglichkeiten zur Verfügung gestellt:

- Manuelle oder automatische oder beide Techniken,

- Statische oder dynamische oder beide Techniken,
- Analytische oder statische oder beide Techniken,
- Testen in realer oder simulierter oder in beiden Umgebungen.

Anforderungen zur Validierung werden in Kapitel 5 behandelt.

4.3 Anforderungen an die Software

In den letzten Abschnitten wurde auf die Phasen im System- und im Softwarelebenszyklus eingegangen. Im Folgenden werden konkrete Anforderungen zu verschiedenen Themen, die in den in diesem Vorhaben betrachteten Normen und Standards angesprochen werden, für die betrachteten Industriezweige verglichen.

Ein Vergleich verschiedener Normen wurde bereits 2009 in /FUS 09/ in Hinblick auf verschiedene Kriterien bezüglich Software durchgeführt. Als Ergebnis wurde eine Tabelle erstellt, welche zum einen die verschiedenen Themen, die in den betrachteten Normen behandelt wurden, darstellt und zum anderen den Detailgrad des jeweiligen Themas bewertet (Siehe Abb. 4.1). Die eingetragenen Zahlen von eins bis vier zeigen an, wie ausgiebig das jeweilige Thema in dem entsprechenden Standard behandelt wird (0=wenig, 4=sehr ausgiebig). In /FUS 09/ wurden Standards aus der Luftfahrtindustrie, der Automobilindustrie, der Kerntechnik und dem Militär verglichen. Dabei wurde die Einordnung der Relevanz des jeweiligen Themas durch den Autor in der Spalte „rank“ angegeben. Ergebnis dieser Auswertung war, dass nicht alle Standards immer die innovativsten Technologien beinhalten. Darüber hinaus kann gezeigt werden, dass es sowohl Standards gibt, die einen Großteil der Themen behandeln, als auch Standards, die sich auf spezielle Themen wie Management konzentrieren. Der Detaillierungsgrad der Behandlung von Themen ist dabei jedoch sehr unterschiedlich. /FUS 09/

Die in /FUS 09/ verglichenen Standards sind in den letzten Jahren überarbeitet worden, so dass teilweise neue Aspekte hinzugekommen sind. Die im Rahmen dieses Projekts untersuchten Normen und Standards wurden deshalb im Folgenden nun ebenfalls thematisch untersucht und hinsichtlich ihrer Anforderungen aufgearbeitet. Dazu wurde ein Vergleich der konkreten Anforderungen anhand der bereits in Kapitel 3 beschriebenen Standards und Normen durchgeführt.

Dass bestimmte Themen darüber hinaus auch in anderen Normen und Standards behandelt werden, kann nicht ausgeschlossen werden. Deswegen werden die Anforderungen grundsätzlich auf die Kerntechnik bezogen, verglichen und eine Übertragbarkeit wird diskutiert. Eine Bewertung in umgekehrter Richtung wird nicht durchgeführt. Des Weiteren sei erwähnt, dass für diesen konkreten Vergleich lediglich die nationalen, in Kapitel 3 genannten Standards und Normen im Bereich der Kerntechnik herangezogen wurden. Die nationalen Regularien in der Kerntechnik werden jedoch durch eine Vielzahl von Dokumenten, wie z.B. das BHB, Fachanweisungen, aber auch Querverweise auf allgemeine Standards und Normen ergänzt. Dass in diesen Dokumenten weiterführende konkrete Anforderungen an Software gestellt werden, kann nicht ausgeschlossen werden.

Tab. 4.5 gibt eine Übersicht darüber, zu welchen Themenbereichen Anforderungen in den verschiedenen nicht-nuklearen Industriezweigen oder der Kerntechnik behandeltet werden. Hierbei sind nur solche Themenbereiche gekennzeichnet, zu denen hinreichend konkrete Anforderungen gestellt werden. Lediglich peripher erwähnte Themenbereiche werden nicht berücksichtigt.

		CRITERIA		Safety & SW Std's							SW Eng Std's					SW Eng Comp		
		keywords	rank	IEC 60880-2	DO-178B	EN 50128	ISO 26262-6/8	IEC 61508-2/3	MIL 882D	CMMI +Safe	ISO/IEC 12207	CMMI	ISO/IEC 15504	ISO/IEC 90003	ISO/IEC 9126	SWEBOK	ISTQB Syllabus	SEI Process Framework
System - SW - Quality	1	interpretation for conformity	4	2	3	3	3	3	4	1	2	3	3	2	2	-	-	-
	2	requirements analysis	4	-	3	2	4	3	-	3	3	4	4	3	-	2	-	3
	3	process concept	4	-	4	1	3	1	0	4	4	4	4	2	-	4	2	4
	4	management	3	2	3	3	4	3	0	3	2	4	4	4	-	2	2	2
	5	evolving technology	3	2	2	3	1	3	0	3	3	3	3	-	2	4	3	4
	6	system engineering	3	2	3	2	3	3	0	3	3	3	2	-	2	-	1	2
	7	safety culture	3	1	3	3	3	3	3	4	-	-	-	-	-	-	-	-
	8	human factors	3	-	1	1	1	2	0	3	1	2	2	-	-	2	-	3
	9	abstraction levels	3	3	4	2	3	2	1	2	3	3	4	3	-	3	2	4
	10	Controller/controlled separation	3	2	3	0	2	2	0	1	-	-	-	-	-	-	-	-
	11	Integrity levels	2	-	4	4	4	4	4	1	-	-	-	-	-	-	-	-
	12	purpose and stakeholders	2	2	3	3	3	3	4	2	1	3	3	3	-	-	-	-
	13	safe components vs. functions	2	2	2	3	3	3	1	-	-	-	-	-	-	-	-	-
	14	independent certification	1	2	4	2	3	3	3	1	2	3	2	2	2	-	-	-
	15	operational phase	1	3	3	2	4	2	1	1	2	2	2	1	2	-	-	-
SW - general	16	COTS / PDS	4	4	3	2	3	2	2	-	-	-	-	0	3	-	3	1
	17	tools selection / certification	4	4	2	3	4	3	1	-	-	-	-	1	4	-	-	-
	18	SW isolation	3	3	4	2	2	3	0	-	-	-	-	-	-	-	-	-
	19	impact analysis / traceability	3	4	3	3	3	3	2	2	0	2	2	1	-	-	-	-
	20	technique selection criteria	2	-	2	3	1	3	-	2	-	1	1	-	-	3	-	1
	21	configuration data	1	-	2	4	0	-	-	-	-	-	-	-	-	-	-	-
	22	SW - System borderline	1	-	4	2	3	3	1	-	0	2	1	1	-	-	-	-
SW - Techniques	23	Formal Methos	4	-	1	1	1	1	-	-	-	-	-	-	-	2	-	-
	24	Model checking	4	-	0	0	0	0	-	-	-	-	-	-	-	-	-	-
	25	model based developent	3	-	0	0	3	0	-	-	-	-	-	-	-	-	-	-
	26	SW requirements analysis	3	-	3	2	3	2	-	3	3	4	4	2	-	4	-	-
	27	verification and testing	3	3	3	3	3	3	2	3	3	3	3	2	2	4	4	-
	28	diversity	2	4	3	2	2	2	0	-	-	-	-	-	-	0	-	-
	29	reliability testing	1	-	1	-	-	-	0	1	-	-	-	-	-	4	2	-

Abb. 4.1 Vergleich verschiedener Normen und Standards in Hinblick auf die Abdeckung verschiedener Kriterien zu Softwareaspekten /FUS 09/

Tab. 4.5 Übersicht über die behandelten Themenbereiche in den verschiedenen nicht-nuklearen und nuklearen Bereichen

	Allgem. Normen	Kern-technik	Bahn	Auto	Wehr-technik	Luftfahrt	Raum-fahrt
Personal und Management							
Teams, Rollen und Verantwortlichkeiten	x		x		x		
Kompetenz der Mitarbeiter			x				
Organisationsstruktur			x		x		
Prozess-Management	x	x			x		x
Konfigurationsmanagement		x		x	x	x	x
Lebenszyklus							
Allgemeine Anforderungen	x	x	x		x	x	x
Dokumentation und Rückverfolgbarkeit	x	x	x		x	x	x
Software-Anforderungsspezifikation	x	x	x	x	x	x	
Schnittstellen (Hardware/Software, Software/Software)		x	x	x			
Software-Architektur	x		x	x	x	x	
Planung	x		x	x	x	x	x
Entwurf	x		x	x	x	x	x
Implementierung	x	x	x	x	x	x	x
Integration	x	x	x	x	x	x	
Verifizierung	x	x	x	x	x	x	x

Validierung	x	x	x		x	x	x
Tests		x	x	x	x	x	
Selbstüberwachung		x					
Prozeduren zur Fehlerbehebung		x			x		
Änderungen	x	x	x	x	x	x	x
Instandhaltung			x		x	x	x
Betrieb							x
Stilllegung							x
Sonstige Themen							
Werkzeuge und Sprachen	x	x	x		x	x	x
Handling und Speichermedien					x		
Begutachtung			x	x			
Codierstandards			x		x	x	
CCF/ Diversität		x			x		
Zugriffsschutz		x			x		

In den folgenden Abschnitten werden die konkreten Anforderungen zu den Themenbereichen separat und detailliert diskutiert, zu denen in der Kerntechnik bisher keine konkreten Anforderungen vorliegen, oder bestehende Anforderungen sinnvoll ergänzt werden können.

4.3.1 Personal und Management

Zum Thema Personal und Management werden in den verschiedenen betrachteten Standards und Normen die folgenden Aspekte behandelt (siehe Tab. 4.5):

- Teams, Rollen und Verantwortlichkeiten
- Kompetenz der Mitarbeiter
- Organisationsstruktur
- Prozess-Management
- Konfigurationsmanagement

Während in DIN EN 61508 (Allgemein) zum Thema Personal nur gefordert wird, dass die Verantwortlichkeiten während der Sicherheitsplanung festgelegt werden müssen, behandelt DIN EN 50128 (Bahn) dieses Thema sehr ausführlich. Neben Anforderungen an den Entwickler, Implementierer und Designer der Software werden hier auch Anforderungen an den Lieferanten und den Gutachter der Software gestellt. Darüber hinaus werden Anforderungen an die Begutachtung der Software gestellt, wobei auch hier neben Anforderungen an den Begutachtungsplan der Schwerpunkt beim Gutachter liegt. Für jede Rolle gibt es eine genaue Beschreibung im normativen Anhang der DIN EN 50128, in der die notwendigen Kompetenzen und die Verantwortlichkeiten aufgeführt sind, die die jeweilige Rolle zu erfüllen hat. DIN EN 50128 stellt darüber hinaus auch detaillierte Anforderungen an die Organisationsstruktur, die sich je nach SIL unterscheidet. In dieser Organisationsstruktur wird beispielsweise festgelegt, welche Aufgaben durch welche Personen zu erfüllen sind. Dabei wird vor allem darauf eingegangen, welche Verantwortungsbereiche durch eine Person abgedeckt werden können und welche Verantwortungsbereiche auf mehrere Personen aufgeteilt werden müssen. Je nach SIL wird dies im Schienenverkehr unterschiedlich gefordert.

Das Handbuch der Wehrtechnik /DOD 10/ stellt Anforderungen an Arbeitsgruppen. Diese müssen sich aus Personen mit geeigneter Expertise zusammensetzen. Aufgabe

der Arbeitsgruppe ist die Identifizierung und Untersuchung von Anwendungsproblemen sowie die Erstellung einer entsprechenden Lösung und einer damit verbundenen Minimierung des Versagensrisikos. Darüber hinaus fordert der Standard, dass der Arbeitsgruppe Unterstützung von Programmierern und Technikern zur Verfügung gestellt werden muss. Insgesamt werden vier Arbeitsgruppen zur Steigerung der Zuverlässigkeit gebildet. Die Arbeitsgruppen decken die Arbeitsfelder Systemsicherheit, Tests, Implementierung und Mensch-Maschine-Schnittstelle ab. Auf diesen vier Ebenen soll somit eine Erhöhung der Zuverlässigkeit des Gesamtsystems erreicht werden.

Keine der betrachteten nationalen, nuklearen Normen und Standards im kerntechnischen Bereich ist bei den Themen Personal, Verantwortlichkeiten und Organisationstruktur derart detailliert wie diese Standards der Wehrtechnik und der Bahnindustrie. Die DIN EN 60880 formuliert in diesem Zusammenhang ausschließlich Anforderungen an das Training des Anlagenpersonals. International wird in /IAEA 94/ auf die Qualifikation des Personals und ihr Training eingegangen. Auch hier geht es jedoch um das Anlagenpersonal und nicht um das Personal für die Software-Entwicklung. Eine Übertragung entsprechender Anforderungen an das Personal, Verantwortlichkeiten und die Organisationstruktur bei der Softwareentwicklung ist zur Erhöhung der Zuverlässigkeit des Gesamtsystems sinnvoll. Entsprechende Anforderungen sollten in nationale, kerntechnische Standards und Normen integriert werden. Ein weiterer Aspekt, der für entsprechende kerntechnische Standards und Normen übernommen werden kann, ist ein grundsätzlicher Aspekt des Aufbaus von /DOD 10/ (Wehrtechnik). Das Handbuch weist zu Beginn jedes Kapitels darauf hin, für welchen Leser das Kapitel relevant ist. Dies trägt erheblich zur leichteren und verständlicheren Anwendung des Handbuchs bei.

4.3.2 Lebenszyklus

Zum Thema Lebenszyklus werden in den betrachteten Normen folgende Themen behandelt (siehe Tab. 4.5):

- Allgemeine Anforderungen
- Dokumentation und Rückverfolgbarkeit
- Software-Anforderungsspezifikation
- Schnittstellen
- Software-Architektur

- Planung, Entwurf
- Implementierung und Integration
- Validierung und Verifizierung
- Tests, Selbstüberwachung
- Prozeduren zur Fehlerbehebung
- Änderungen, Instandhaltung
- Betrieb
- Stilllegung

Auf die Unterschiede der einzelnen Lebenszyklus-Phasen wurde bereits im letzten Abschnitt eingegangen. Im Folgenden werden daher darüber hinaus gehende Unterschiede der nicht-nuklearen Normen zur Kerntechnik dargestellt.

Dokumentation und Rückverfolgbarkeit

In DIN EN 50128 (Bahn) ist eine ausführliche Liste enthalten, welche Dokumente im Laufe des Lebenszyklus erstellt werden müssen. Für die Erstellung der verschiedenen Berichte sind unterschiedliche Personen verantwortlich (siehe nachfolgende Tabellen). Die Normen und Standards der Kerntechnik gehen nicht in diesem Detail auf die zu erstellenden Berichte ein und fordern auch nicht die Menge an Dokumenten, die im Schienenverkehr gefordert werden. In den Tabellen wird im Vergleich angezeigt, welche Berichte in der Kerntechnik gemäß DIN IEC 60880 gefordert werden.

Tab. 4.6 Dokumente, die im Schienenverkehr vom Entwerfer zu erstellen sind

Dokument	**Kerntechnik**
SW-Architekturspezifikation	
SW-Entwurfsspezifikation	
SW-Schnittstellenspezifikation	
SW-Komponentenentwurfsspezifikation	
Freigabemitteilung	

Tab. 4.7 Dokumente, im Schienenverkehr vom Verifizierer zu erstellen sind

Dokument	**Kerntechnik**
SW-QS-Plan	X
SW-QS-Verifikationsbericht	
SW-Verifikationsplan	X
SW-Verifikationsbericht	
SW-Anforderungsverifikationsbericht	
SW-Architektur- und Entwurfsverifikationsbericht	
SW-Komponentenentwurfsverifikationsbericht	
SW-Quellcodeverifikationsbericht	X
SW-Integrationsverifikationsbericht	X
SW-Validierungsverifizierungsbericht	
Anwendungsdaten-/ -algorithmen-Verifikationsbericht	
Bereitstellungs-Verifikationsbericht	
SW-Wartungsverifikationsbericht	
SW-Begutachtungsverifikationsbericht	

Tab. 4.8 Dokumente, die im Schienenverkehr vom Validierer zu erstellen sind

Dokument	**Kerntechnik**
SW-Validierungsplan	
SW-Validierungsbericht	

Tab. 4.9 Dokumente, die im Schienenverkehr vom Anforderungsmanager zu erstellen sind

Dokument	Kerntechnik
SW-Anforderungsspezifikation	X
Anwendungs-Anforderungsspezifikation	
Anwendungs-Generierungsplan	

Tab. 4.10 Dokumente, die im Schienenverkehr vom Tester zu erstellen sind

Dokument	Kerntechnik
Gesamt-SW-Testspezifikation	x
SW-Komponententestspezifikation	
SW-Komponententestbericht	x
Gesamt-SW-Testbericht	
Anwendungs-Testspezifikation	
Anwendungs-Testbericht	

Tab. 4.11 Dokumente, die im Schienenverkehr vom Integrator zu erstellen sind

Dokument	Kerntechnik
SW-Integrationstestbericht	
SW-/HW-Integrationstestbericht	

Tab. 4.12 Dokumente, die im Schienenverkehr vom Gutachter zu erstellen sind

Dokument	Kerntechnik
SW-Begutachtungsplan	
SW-Begutachtungsbericht	

DIN EN 50128 behandelt als einzige betrachtete Norm das Thema der Rückverfolgbarkeit von Dokumenten. Hierzu wird gefordert, dass die Rückverfolgbarkeit gegeben sein muss, indem jedes Dokument eine eindeutige Referenznummer und einen definierten und dokumentierten Bezug zu anderen Dokumenten besitzt. Jedes Dokument muss alle

anwendbaren Bedingungen und Anforderungen des Vorgängerdokuments enthalten und darf seinem Vorgängerdokument nicht widersprechen. Alle Dokumente müssen darüber hinaus so strukturiert sein, dass eine fortlaufende Erweiterung während des Entwicklungsprozesses möglich ist.

Entsprechend detaillierte Anforderungen an die Dokumentation und Rückverfolgbarkeit werden in den nationalen, kerntechnischen Normen und Standards nicht gestellt. Im Schienenverkehr wird auf Normenebene eine große Anzahl an Dokumenten während des Lebenszyklus gefordert. In den kerntechnischen Normen ist die Anzahl der geforderten Dokumente kleiner. Dies bedeutet jedoch nicht, dass weniger Dokumentation in der Kerntechnik erfolgt, da viele der Berichte auch zu einem Bericht zusammengefasst werden könnten. Eine entsprechende Auflistung verdeutlicht jedoch, welche Dokumente mindestens erstellt werden müssen. Zusammen mit den Anforderungen an die Rückverfolgbarkeit der Dokumente trägt dies zur Erhöhung der Qualität bei der Softwareentwicklung bei.

Schnittstellen und Hardware

Sowohl die betrachteten Normen aus der Kerntechnik als auch aus der Automobilindustrie und dem Schienenverkehr beschreiben Anforderungen an die Schnittstellen zwischen Hardware und Software. Während die Anforderungen an die Schnittstellen in ISO 26262 (Auto) sehr allgemein gehalten sind, werden in DIN EN 50128 (Bahn) detailliertere Anforderungen an die Schnittstellen formuliert. Nach DIN EN 50128 (Bahn) muss z.B. eine Schnittstellenspezifikation erstellt werden, welche neben den festgelegten Grenzwerten u.a. auch auf die Beschreibung des Verhaltens bei Erreichen und bei Überschreitung dieser Grenzwerte eingehen soll. Darüber hinaus müssen zeitliche Beschränkungen und Anforderungen an den korrekten Betrieb sowie die Behandlung von Ausnahmen beschrieben werden. Sofern anwendbar, wird ein zugewiesener Speicher für die Schnittstellen-Pufferspeicher sowie Mechanismen zur Erkennung, ob der Speicher zugewiesen werden kann oder alle Pufferspeicher belegt sind, gefordert. Zusätzlich müssen Synchronisationsmechanismen zwischen den Funktionen existieren. Schnittstellen müssen laut DIN EN 50128 bereits in frühen Lebenszyklus-Phasen definiert und bei Änderungen immer aktualisiert werden. Bei der Systemintegration müssen sie angemessen geprüft werden können. Anhand einer Testspezifikation muss gezeigt werden, dass jede Softwarekomponente über festgelegte Schnittstellen zu anderen Komponenten verfügt, indem diese Komponenten gemeinsam ausgeführt werden. Abschließend wird gefordert,

dass eine strenge Trennung zwischen generischer Software und Anwendungsdaten vorliegen muss. Es muss also möglich sein, entweder die generische Software oder die Anwendungsdaten zu kompilieren, ohne die jeweils andere Software aktualisieren zu müssen.

In der Wehrtechnik werden ebenfalls Anforderungen an die Hardware-Umgebung, in der die Software betrieben wird, gestellt. So werden unter Anderem konkrete Anforderungen an die verwendete CPU gestellt. In /AOP 52/ wird zum Beispiel gefordert, dass Prozessoren mit separaten Datenspeichern und Schnittstellen solchen mit kombinierten vorzuziehen sind. Des Weiteren sind CPUs, Mikroprozessoren und Computer, die (in Analysen) mathematisch dargestellt werden können zu bevorzugen.

Entsprechend konkrete Anforderungen an die Hardware-Komponenten des Gesamtsystems werden in den nationalen, kerntechnischen Standards und Normen nicht gestellt. Hier wird für die Schnittstellen gefordert, dass die Software-Schnittstellen beschrieben und die wechselseitigen Anforderungen zwischen Hardware und Software in Hinblick auf Ausfallentdeckung und Reaktionen auf Ausfallanzeigen bestimmt werden müssen. Eine Übertragung von Anforderungen an die Schnittstellen und Hardwarekomponenten ist daher sinnvoll.

4.3.3 Sonstige Themen

Neben Themen zu Personal und Management und zum Lebenszyklus werden in den betrachteten Normen und Standards weitere Themen behandelt. Hierzu gehören (siehe Tab. 4.5):

- Werkzeuge und Sprachen
- Handling und Speichermedien
- Codierstandards
- Zugriffsschutz
- CCF/ Diversität

Werkzeuge

Sowohl DIN EN 50128 (Bahn) und ISO 26262 (Auto) als auch DIN EN 60880 (Kerntechnik) und DIN EN 61508 (allgemein) stellen Anforderungen an die einzusetzenden Werkzeuge bei der Software-Entwicklung. Während DIN EN 61508 wie bei den anderen Themen nur sehr allgemeine Anforderungen stellt, sind DIN EN 50128 und DIN IEC 60880 hier sehr viel detaillierter.

In ISO 26262 liegt der Schwerpunkt der Anforderungen zu Software-Werkzeugen auf dem Nachweis der Vertrauenswürdigkeit und der Qualifizierung des Werkzeugs. Diese Themen werden auch in DIN IEC 60880 behandelt.

In DIN EN 50128 wird als Ziel der Anforderungen die Erbringung des Nachweises genannt, dass mögliche Fehler der Werkzeuge das Ausgangsprodukt nicht in einer sicherheitsrelevanten Weise beeinträchtigen. Zu diesem Zweck wurden die Werkzeuge in drei Klassen eingeteilt:

- T1: Das Werkzeug erzeugt keine Ausgaben, die direkt oder indirekt zum ausführbaren Code der Software beitragen.
- T2: Das Werkzeug unterstützt den Test oder die Verifizierung des Entwurfs oder des ausführbaren Codes, bei dem Fehler im Werkzeug zur Nicht-Erkennung von Fehlern führen können, jedoch in der ausführbaren Software keine Fehler direkt erzeugen können.
- T3: Das Werkzeug erzeugt Ausgangsdaten, die direkt oder indirekt zum ausführbaren Code des sicherheitsrelevanten Systems beitragen.

Entsprechend dieser Klassifizierung werden in DIN EN 50128 unterschiedliche Anforderungen an die verschiedenen Werkzeug-Klassen gestellt. Eine vergleichbare Klassifizierung von Werkzeugen gibt es in der Kerntechnik nicht und konnte auch in den anderen betrachteten Normen unterschiedlicher Industriezweige nicht identifiziert werden. Eine Übertragung der Anforderungen an die Klassifizierung der Werkzeuge für die Softwareentwicklung ist dementsprechend möglich und sinnvoll.

Handling und Speichermedien

Die Wehrtechnik stellt als einziger betrachteter Industriezweig konkrete Anforderungen an den Umgang mit und an das Speichern der entwickelten Software. Der Entwickler

muss laut /DEF 95/ eine Prozedur festlegen, um den sicheren Umgang und das zuverlässige Speichern der Software zu gewährleisten. Hierbei müssen folgende Bedingungen erfüllt sein:

- Die Software muss in einer Weise kopiert und gesichert sein, die sie gegen Verlust bei Katastrophen geschützt ist.
- Es muss ein System installiert sein, dass einen Zugriff auf die Software nur über einen autorisierten Zugang erlaubt.
- Die Umgebung muss so beschaffen sein, dass eine Beeinträchtigung von physikalischen Speichermedien minimiert wird.

Darüber hinaus fordert /DOD 10/ (Wehrtechnik), dass für das Speichern der Software langzeitgeeignete, nicht wiederbeschreibbare ROM – Speicher verwendet werden. /DEF 55/ stellt außerdem Anforderungen an die Vervielfältigung der Software.

Des Weiteren wird in der Wehrtechnik gefordert, dass die Speichermedien und die Datenübertragung periodisch geprüft werden müssen.

Entsprechende Anforderungen an den Umgang mit oder die Speicherung der Software werden in den betrachteten nationalen, nuklearen Standards und Normen nicht aufgeführt. Eine entsprechende Ergänzung zu Anforderung an den Umgang mit der Software sowie deren Speicherung und Sicherung sind jedoch sinnvoll und sollten in der Kerntechnik ergänzt werden.

Codierstandards

In der Wehrtechnik werden konkrete Anforderungen an den Software-Code gestellt. So wird in /AOP 52/ gefordert, dass er modular aufgebaut und möglichst wenig komplex gestaltet sein soll. Desweitere soll sicherheitskritischer Code nicht unterbrochen sein und es darf keinen nicht genutzten Code geben. Der Code muss vor Änderungen (auch durch sich selbst) geschützt sein. Weitere konkrete Anforderungen sind zum Beispiel, dass Variablen eindeutig definiert sein müssen und sicherheitsrelevante Variablen von nicht sicherheitsrelevanten zu unterscheiden sein müssen. Neben diesen und weiteren konkreten Anforderungen an die Codierung stellt /AOP 52/ ebenfalls konkrete Anforderungen an die Programmiersprache. Diese soll folgende Eigenschaften aufweisen:

- Formal definierte Syntax

- Starke Typisierung
- Blockstruktur
- Vorhersehbare Programmausführung
- Vermeidung von mehrdeutigen Merkmalen
- Möglichkeit der Variablendefinierung (Konventionen)
- Leicht lesbares Format

Ähnliche Anforderungen an die Codierung werden auch in der Luftfahrt /RTCA 11/ gestellt. Dort werden sie sogar dahingehend noch ergänzt, dass konkrete Anforderungen an das Schreiben des Codes gestellt werden, wie zum Beispiel:

- Begrenzung der Zeilenlänge
- Verwendung von Tabs und Leerzeilen
- Dokumentation des Quellcodes mit
    - Name des Autors
    - Revisionshistorie
    - Betroffene Daten und Größen
- Eindeutige Benennung von Subsystemen

Auch in DIN EN 50128 (Bahn) wird auf Anforderungen an Codierstandards eingegangen. Hierzu zählen eine gute Programmierpraxis, der Einsatz von Maßnahmen zur Vermeidung oder Erkennung von Fehlern sowie von Verfahren zur Quellcode-Dokumentation. Darüber hinaus muss das Entwurfsverfahren Merkmale besitzen, die Folgendes unterstützen:

- Klare und genaue Darstellung von Funktionalität, Informationsfluss zwischen den Komponenten, Reihenfolge und zeitbezogenen Informationen, Parallelverarbeitung, Datenstrukturen und –eigenschaften
- Verständlichkeit für den Menschen
- Verifikation und Validierung
- Software-Wartung.

Entsprechend konkrete Anforderungen an die Codierung und die Programmiersprache sind in den nationalen, kerntechnischen Standards und Normen nicht enthalten. Anforderungen an die Codierung und die Programmiersprache können jedoch bereits im Entwicklungsprozess die Zuverlässigkeit der Software erhöhen und sollten in den Bereich der Kerntechnik übertragen werden.

CCF/ Diversität

Auf das Thema Diversität und gemeinsam verursachte Ausfälle geht die Kerntechnik von allen betrachteten Industriezweigen am detailliertesten ein. Die Wehrtechnik beschäftigt sich zwar ebenfalls mit diesem Thema, jedoch bei weitem nicht in dem Detailgrad. Zu diesem Thema sind demnach keine neuen Erkenntnisse aus anderen Industriezweigen abzuleiten.

Zugriffsschutz

Auf das Thema Zugriffsschutz geht die Kerntechnik von allen betrachteten Industriezweigen am detailliertesten ein. Die Wehrtechnik beschäftigt sich zwar ebenfalls mit diesem Thema, jedoch bei weitem nicht in dem Detailgrad. Zu diesem Thema sind demnach keine neuen Erkenntnisse aus anderen Industriezweigen abzuleiten.

4.4 Fazit

4.4.1 System-Sicherheitslebenszyklus

Der System-Sicherheitslebenszyklus unterscheidet sich in den verschiedenen Industriezweigen nur geringfügig. Der Ablauf der Phasen ist beim Schienenverkehr, der Automobilindustrie und der Kerntechnik grundsätzlich vergleichbar. Beim Schienenverkehr wird sich sehr genau an den in DIN EN 61508 beschrieben Verlauf des Lebenszyklus gehalten. Die Automobilindustrie unterscheidet sich dahingehend, dass der gesamte Lebenszyklus auf die Serienproduktion angepasst wurde. Dadurch verschieben sich im Vergleich zu den anderen Industriezweigen manche Phasen etwas in der Reihenfolge. Auch die Anforderungen sind auf die Sicherstellung der funktionalen Sicherheit in der Serienproduktion angepasst. Die Kerntechnik unterscheidet sich von den anderen Industriezweigen dadurch, dass der Lebenszyklus nach DIN EN 61513 erst später einsetzt, da die Bestimmung der Auslegungsgrundlagen nicht als Phase im Lebenszyklus vorgesehen ist, sondern als Vorarbeit in die erste Phase des Lebenszyklus eingeht.

So wird in DIN EN 61513 zum Beispiel keine Phase zur Konzeptentwicklung aufgeführt. Diese Phase dient in den anderen Industriezweigen dazu, ein ausreichendes Verständnis für das Gesamtsystem zu entwickeln. Entsprechende Anforderungen werden in DIN EN 61513 nicht an diese Arbeiten formuliert. Auch eine Gefährdungs- und Risikoanalyse wird in DIN EN 61513 im Gegensatz zu den anderen Normen nicht konkret aufgegriffen. Sie beginnt ihren System-Sicherheitslebenszyklus erst mit der Bestimmun und Zuordnung der Sicherheitsanforderungen des Systems. DIN EN 61513 behandelt darüber hinaus die Phasen der Stilllegung und Entsorgung im Gegensatz zu den anderen Industriezweigen nicht. Neue Erkenntnisse lassen sich aus diesen Unterschieden jedoch nicht ableiten, da die DIN EN 61513 die Erfüllung und Umsetzung der allgemeinen Norm DIN EN 61508 fordert und somit die entsprechenden Phasen des System-Sicherheitslebenszyklus sowie die entsprechenden Anforderungen an diesen trotzdem bereits enthält.

Eine Festlegung und Berücksichtigung der Fragestellung, ob es sich bei der Entwicklung um eine Neuentwicklung oder eine Modifikation der Software handelt wird lediglich in ISO 26262 (Auto) als früher Bestandteil des System-Sicherheitslebenszyklus betrachtet. Dieser Aspekt findet sich in keinem anderen Industriezweig und auch nicht in der allgemeinen Norm oder der Kerntechnik. Eine frühe Betrachtung dieser Fragestellung ist jedoch sinnvoll und kann durchaus auf die Kerntechnik übertragen werden.

4.4.2 Software-Sicherheitslebenszyklus

Im Standard DIN EN 61508-3 wird die Erfüllung der Anforderungen an den Software.Sicherheitslebenszyklus gefordert, wobei das zu verwendende Vorgehensmodell nicht festgelegt wird. Die Anforderungen sind dabei jedoch so gestellt, das nur das V-Modell als Vorgehensmodell verwendet werden kann. Dies wird aktuell auch in der Kerntechnik empfohlen. Laut Konferenzvortrag /MYK 15/ soll die Norm DIN EN 61508-3 jedoch so überarbeitet werden, dass als mögliche Methoden sowohl das V-Modell als auch das Scrum-Modell normenkonform verwendet werden können. Im Schienenverkehr und der Öl- und Gasindustrie wird SafeScrum vom TÜV Nord und TÜV Rheinland akzeptiert /MYK 15/. Auch andere Zertifizierungsstellen, wie beispielsweise TÜV Süd oder Lloyds akzeptieren Agile Entwicklungsmethoden. Da der Standard DIN EN 60880 darauf aufmerksam macht, dass er mit DIN EN 61513 und den darin verwiesen Standards (u.a. DIN EN 61508) ein konsistentes Set an Dokumenten darstellt, ist die Kerntechnik von dieser Änderung ebenfalls betroffen.

Darüber hinaus ergeben sich keine neuen Erkenntnisse für die Kerntechnik aus der Betrachtung des Software-Lebenszyklus der anderen Industriezweige. In der Regel sind die Anforderungen der Kerntechnik detaillierter als in den anderen Industriezweigen. So wird zum Beispiel in der Kerntechnik konkret auf den Aspekt der Selbstüberwachung eingegangen. Dieser Aspekt wird in den Normen der anderen Industriezweige nicht angesprochen.

Aktuell besteht keine Notwendigkeit, Aspekte aus dem Software-Sicherheitslebenszyklus auf die Kerntechnik zu übertragen. Welche konkreten Anforderungen an die Software und die Softwareentwicklung übertragen werden können, ist im folgenden Absatz zusammengefasst.

4.4.3 Anforderungen an die Software

In Abschnitt 4.3 wurden konkrete Anforderungen zu verschiedenen Themen für die betrachteten Industriezweige verglichen. Hierbei hat sich gezeigt, dass in einigen Themenbereichen durchaus eine Übertragung von konkreten Anforderungen aus anderen Industriezweigen sinnvoll ist. Konkret sind Anforderungen aus den Bereichen Personal und Management, Codierstandards, Dokumentation, Schnittstellen und Hardware, Werkzeuge, Handling und Speicherung sowie Begutachtung zu nennen.

Darüber hinaus können Empfehlungen zum Aufbau und zur Strukturierung von Regelwerken, Standards und Normen übernommen werden. Für den Aufbau eines kerntechnischen Regelwerks mit integrierten Forderungen zu softwarebasierter Leittechnik ist die Übertragung einer guten Strukturierung und Anwendungsempfehlung, wie sie in der Wehrtechnik zu finden ist, zu empfehlen. So wird zum Beispiel im Handbook /DOD 10/ jedes Kapitel damit eingeleitet, wer aus dem beteiligten Team das entsprechende Kapitel braucht und lesen soll. Weiterhin ist /DEF 42/ wie ein Leitfaden, der durch den Entwicklungsprozess führt, aufgebaut. Beides sorgt für eine bessere Übersichtlichkeit und ein effizienteres Arbeiten bei der Anwendung des Regelwerks.

In den Bereichen Selbstüberwachung und Diversität gehen die Anforderungen aus dem Kerntechnischen Regelwerk über die der anderen Bereiche hinaus. Eine Übertragung von Anforderungen aus der nicht-nuklearen Industrie ist somit nicht gegeben.

4.5 Vergleich kerntechnischer Kategorien und SIL

In den verschiedenen in Kapitel 3 beschriebenen Regelwerken, Standards und Normen werden unterschiedliche Kategorisierungsschemata benutzt. Um einen Vergleich zu ermöglichen, werden im Folgenden Beziehungen zwischen verschiedenen Kategorisierungen hergestellt. Dies betrifft die Sicherheitsanforderungsstufe (SIL) gemäß IEC 61508 /DIN 61a-h/, das Performance Level (PL) gemäß DIN EN ISO 13849 /ISO 15/, /ISO 13/ sowie die Kategorisierung nach den Sicherheitsebenen und Leittechnikkategorien des übergeordneten deutschen Regelwerks, d. h. der Sicherheitsanforderungen an Kernkraftwerke (SiAnf) /BMU 12/ und Interpretationen zu den Sicherheitsanforderungen an Kernkraftwerke /BMU 13/.

4.5.1 Beziehung zwischen Sicherheitsanforderungsstufe (SIL), Performance Level (PL) und Leittechnikkategorien des übergeordneten deutschen Regelwerks

Im Folgenden werden zum Herstellen einer Verbindung zwischen den verschiedenen Kategorisierungsarten der verschiedenen Regelwerke zwei Ansätze verfolgt. Zum einen werden Kategorisierungen für Einrichtungen nach den verschiedenen Regelwerken vorgenommen, um so zu einer Verknüpfung der verschiedenen Klassifizierungen zu kommen. Zum anderen werden Ausfallraten bzw. Ausfallwahrscheinlichkeiten, die in den Normen genannt werden, verglichen.

4.5.2 Herleitung einer Beziehung aufgrund der Klassifizierung nach verschiedenen Regelwerken

Als Objekt der Klassifizierungen werden leittechnische Einrichtungen in Hebezeugen in Kernkraftwerken verwendet. Für diese Einrichtungen gibt es eine kürzlich überarbeitete Regel KTA 3902 /KTA 12/ des Kerntechnischen Ausschusses, die bereits explizit auf softwarebasierte Leittechnik Bezug nimmt und nach Auffassung der GRS den aktuellen Stand der Wissenschaft und Technik auf diesem Gebiet darstellt. In dieser Regel werden neben der Klassifizierung nach DIN EN ISO 13849 auch Aussagen zu alternativen, als äquivalent zu betrachtenden Einstufungen nach IEC 61508 getroffen. In dieser Norm charakterisierte Sicherheitsfunktionen können auch Kategorien des übergeordneten deutschen Regelwerks zugeordnet werden.

Es ist zu betonen, dass diese Beziehungen nur zum Zwecke des Vergleiches der Anforderungen an die Softwareentwicklung verschiedener Regelwerke hergeleitet wurden.

Daraus ergibt sich keine allgemeine Äquivalenz oder Vergleichbarkeit der Kategorien verschiedener Regelwerke. Insbesondere kann aus den Beziehungen keine Argumentation abgeleitet werden, dass durch Erfüllung sicherheitstechnischer Anforderungen eines Regelwerkes zu den sicherheitstechnischen Anforderungen eines anderen Regelwerkes äquivalente Anforderungen erfüllt würden.

In Anhang E.1 von /KTA 12/ ist das Klassifizierungsschema (dort Einstufungsschema genannt) dieser Norm beschrieben. Im Folgenden sind die Kategorien, angefangen bei der höchsten, beschrieben:

1. Die höchste Kategorie beschreibt „Sicherheitsfunktionen, bei deren Versagen als Folge eine Überschreitung der Störfallplanungswerte nach § 49 StrlSchV unterstellt werden muss“. Solche Leittechnikfunktionen werden der Kategorie A des deutschen Regelwerks zugeordnet.

   Nach KTA 3902 ist, falls eine Ausführung mittels softwarebasierter Systeme vorgesehen ist, die Funktion in zweifacher Ausführung erforderlich, wobei eine Ausführung in Performance Level (PL) e und eine zweite hiervon unabhängige und verschiedenartige Ausführung mindestens in Performance Level c zu realisieren ist. Die Funktionen müssen unabhängig von den betrieblichen Steuerungsfunktionen sein. Fehler in den betrieblichen Steuerungsfunktionen dürfen diese Funktionen nicht unwirksam machen.

2. Die nächstniedrigere Kategorie umfasst sonstige Leittechnikfunktionen, die zur Beherrschung von Störungsereignissen vorgesehen sind, bei denen die Gefahren gemäß Abschnitt 4.3 oder 4.4 der KTA 3902 zu besorgen sind und keine anderweitigen Möglichkeiten zur Vermeidung der Gefährdung oder zur Begrenzung der Schadensauswirkungen (z. B. durch manuelles Eingreifen) bestehen. Abschnitt 4.3 betrifft Hebezeuge, bei denen beim Transport von Kernbrennstoffen, sonstigen radioaktiven Stoffen, radioaktiven Anlagenteilen oder sonstigen Lasten durch das Versagen des Hebezeugs die Gefahr eines Kritikalitätsunfalls oder die Gefahr einer Aktivitätsfreisetzung, als deren Folge die maximal zulässigen Ableitungen in die Umgebung gemäß Genehmigung überschritten werden können oder die Strahlenexposition in der Umgebung des Kernkraftwerkes für Einzelpersonen der Bevölkerung oberhalb der Grenzwerte der StrlSchV liegen kann, zu besorgen ist, während Abschnitt 4.4 die Brennelement-Wechselanlagen für Leichtwasserreaktoren betrifft.

Da eine Überschreitung der Störfallplanungswerte nach § 49 StrlSchV ausgeschlossen werden kann, werden diese Funktionen der Kategorie B des deutschen Regelwerks zugeordnet.

Nach KTA 3902 ist für solche Funktionen ein Performance Level e erforderlich. Sie müssen unabhängig von den betrieblichen Steuerungsfunktionen sein. Fehler in den betrieblichen Steuerungsfunktionen dürfen diese Funktionen nicht unwirksam machen.

3. Die nächstniedrigere Kategorie umfasst sonstige Leittechnikfunktionen, die zur Beherrschung von Störungsereignissen vorgesehen sind, bei denen die Gefahren gemäß Abschnitt 4.2 der KTA 3902 zu besorgen sind und keine anderweitigen Möglichkeiten zur Vermeidung der Gefährdung oder zur Begrenzung der Schadensauswirkungen (z. B. durch manuelles Eingreifen) bestehen. Abschnitt 4.2 betrifft Hebezeuge, bei denen beim Transport von Kernbrennstoffen, sonstigen radioaktiven Stoffen, radioaktiven Anlagenteilen oder sonstigen Lasten durch das Versagen des Hebezeuges unmittelbar die Gefahr einer Aktivitätsfreisetzung, als deren Folge eine Strahlenexposition von Personen in der Anlage mit einer effektiven Dosis durch innere Exposition über 1 mSv oder durch eine externe Exposition über 5 mSv eintreten kann, zu besorgen ist oder ein nicht absperrbarer Reaktorkühlmittelverlust oder eine über die Redundanz hinausgehende Beeinträchtigung von Sicherheitseinrichtungen, die notwendig sind, den Reaktor jederzeit abzuschalten, in abgeschaltetem Zustand zu halten oder die Nachwärme abzuführen, zu besorgen ist, und keine Gefahren gemäß Abschnitt 4.3 zu besorgen sind. Weiterhin umfasst sie Funktionen, die zur Beherrschung von Störungsereignissen vorgesehen sind, bei denen die Gefahren gemäß Abschnitt 4.3 oder 4.4 zu besorgen sind und anderweitigen Möglichkeiten zur Vermeidung der Gefährdung oder zur Begrenzung der Schadensauswirkungen bestehen.

   Da die maximal zulässigen Ableitungen in die Umgebung gemäß Genehmigung nicht überschritten werden können und die Strahlenexposition in der Umgebung des Kernkraftwerkes für Einzelpersonen der Bevölkerung nicht oberhalb der Grenzwerte der StrlSchV liegen kann, werden diese Funktionen der Kategorie B des deutschen Regelwerks zugeordnet. Dies ist in Übereinstimmung mit den SiAnf, in denen gefordert wird, auf den Sicherheitsebenen 1 und 2 die Strahlenexposition des Personals bei allen Tätigkeiten unter Berücksichtigung aller Umstände des Einzelfalls auch unterhalb der Grenzwerte der Strahlenschutzverordnung so gering wie möglich zu halten. Der Grenzwert der Strahlenschutzverordnung für beruflich strahlenexponierter

Personen bei deren Berufsausübung ist 20 mSv, während die oben angeführte Definition einen deutlich geringeren Wert verwendet.

Nach KTA 3902 ist für solche Funktionen ein Performance Level d erforderlich. Sie müssen unabhängig von den betrieblichen Steuerungsfunktionen sein. Fehler in den betrieblichen Steuerungsfunktionen dürfen diese Funktionen nicht unwirksam machen.

4. Die nächstniedrigere Kategorie umfasst sonstige Leittechnikfunktionen, die zur Beherrschung von Störungsereignisse vorgesehen sind, bei denen die Gefahren gemäß Abschnitt 4.2 der KTA 3902 zu besorgen sind und bei denen Möglichkeiten gegeben sind, die Gefährdung (z. B. durch manuelles Eingreifen) zu begrenzen oder zu vermeiden, oder die den Funktionen der PL d oder e vorgelagert sind.

   Diese Funktionen werden der Kategorie C des deutschen Regelwerks zugeordnet, da es sich um sicherheitstechnisch wichtige Funktionen handelt, sie jedoch nicht zwingend erforderlich sind, um Ereignisse der Sicherheitsebenen 2 oder 3 zu beherrschen.

   Nach KTA 3902 ist für solche Funktionen ein Performance Level c erforderlich. Sie müssen unabhängig von den betrieblichen Steuerungsfunktionen sein. Fehler in den betrieblichen Steuerungsfunktionen dürfen diese Funktionen nicht unwirksam machen.

5. Die nächstniedrigere Kategorie umfasst sonstige Leittechnikfunktionen, die betriebliche Funktionen ausführen, denen eine mittelbare sicherheitstechnische Bedeutung zuzuordnen ist, d.h. die für den sicheren Betrieb des Hebezeugs einen unterstützenden Beitrag liefern. Diese sind nach dem deutschen Regelwerk nicht kategorisiert.

   Nach KTA 3902 ist für solche Funktionen ein Performance Level b erforderlich. Forderungen nach einer Unabhängigkeit von sonstigen betrieblichen Funktionen werden nicht gestellt.

6. Die niedrigste Kategorie umfasst betriebliche Funktionen, für die eine gewisse sicherheitstechnische Relevanz gegeben ist, z. B. wenn deren Zuverlässigkeit die Ansprechhäufigkeit von Funktionen beeinflusst, für die ein PL b, c, d oder e gefordert wird. Diese sind nach dem deutschen Regelwerk nicht kategorisiert.

Nach KTA 3902 ist für solche Funktionen ein Performance Level a erforderlich. Forderungen nach einer Unabhängigkeit von sonstigen betrieblichen Funktionen werden nicht gestellt.

7. Sonstige betriebliche Funktionen sind nach KTA 3902 nicht kategorisiert und es werden keine speziellen Forderungen an sie gestellt. Diese sind nach dem deutschen Regelwerk auch nicht kategorisiert.

Nach KTA 3902 dürfen statt der Performance Level nach DIN EN ISO 13849-1 die in der Normenreihe DIN EN 61508 definierten Safety Integrity Levels (SIL) verwendet werden, wobei die Anforderungen gemäß SIL 2 nach DIN EN 61508 als gleichwertig mit den Anforderungen gemäß Performance Level d nach DIN EN ISO 13849-1 und die Anforderungen gemäß SIL 3 als gleichwertig mit den Anforderungen gemäß Performance Level e gelten.

Die sich aus den Ausführungen ergebenden Beziehungen sind in Tab. 4.13 zusammengefasst.

Tab. 4.13 Beziehung der Kategorisierungen der KTA 3902 unter Verwendung von PL nach EN ISO 13849-1 und SIL nach DIN EN 61508 gemäß KTA 3902 und Kategorisierungen nach deutschem kerntechnischen Regelwerk zum Zwecke des Vergleiches der Anforderungen an die Softwareentwicklung

PL	**SIL**	**Sicherheits-ebene**	**Kategorie**
keine	-	-	-
a	Keine Angabe	-	-
b	Keine Angabe	-	-
c	Keine Angabe	1	C
d	2	2	B
e	3	2	B
Zwei diversitäre Einrichtungen mit e und c	Keine Angabe	3	A

4.5.3 In den Normen angegebene Beziehung

In der ISO 13849-1 ist eine Beziehung zwischen dem Performance Level (PL) der ISO 13849-1 und dem Safety Integrity Level (SIL) der IEC 61508–1 angegeben.

Tab. 4.14 Beziehung zwischen dem Performance Level (PL) der ISO 13849-1 und der Safety Integrity Level (SIL) der IEC 61508–1 gemäß ISO 13849-1

PL	SIL
a	Keine Entsprechung
b	1
c	1
d	2
e	3
Keine Entsprechung	4

Diese Beziehung stimmt mit den Beziehungen aus dem vorigen Abschnitt (Tab. 4.13) bezüglich der PL d und PL e mit SIL 2 bzw. SIL 3 überein.

4.5.4 Herleitung einer Beziehung aufgrund quantitativer Zuverlässigkeitswerte

Dem Vergleich quantitativer Zuverlässigkeitswerte sind nur diejenigen Normen und Regeln zugänglich, in der solche Werte genannt werden. Dies ist zutreffend für

- EN ISO 13849-1 (allgemein),
- IEC 61508 (allgemein),
- ISO 26262-5 (Auto),
- DIN EN 50128 (Bahn),

nicht jedoch für Leittechnikkategorien des übergeordneten kerntechnischen deutschen Regelwerks.

EN ISO 13849

In EN ISO 13849-1: 2006 sind die Performance Level mit der Rate eines gefährlichen Ausfalls verknüpft (siehe Tab. 4.15)

Tab. 4.15 Performance Level nach EN ISO 13849-1 und Rate eines gefährlichen Ausfalls

	Rate gefährlicher Ausfälle pro Stunde	
PL	**Minimum**	**Maximum**
a	10^{-5}	10^{-4}
b	$3 * 10^{-6}$	10^{-5}
c	10^{-6}	$3 * 10^{-6}$
d	10^{-7}	10^{-6}
e	10^{-8}	10^{-7}

DIN EN 61508

Für die SIL nach DIN EN 61508 sind für die Rate von Ausfällen pro Stunde bei kontinuierlichem Betrieb oder im „high demand mode“, d. h. mit mehr als einer erwarteten Anforderung pro Jahr, Zielwerte definiert. Diese sind in Tab. 4.16 aufgeführt.

Tab. 4.16 SIL nach DIN EN 61508 und Zielwert der Ausfallrate bei kontinuierlichem Betrieb oder im „high demand mode“

	Rate gefährlicher Ausfälle pro Stunde	
SIL	**Minimum**	**Maximum**
1	10^{-6}	10^{-5}
2	10^{-7}	10^{-6}
3	10^{-8}	10^{-7}
4	10^{-9}	10^{-8}

Für die SIL nach DIN EN 61508 gelten für die Ausfallwahrscheinlichkeiten bei Anforderung im „low demand mode“, d. h. bei weniger als einer Anforderung pro Jahr Zielwerte, die in Tab. 4.17 aufgeführt sind.

Tab. 4.17 Zielwerte der Ausfallwahrscheinlichkeiten nach DIN EN 61508 („low demand mode“)

	Wahrscheinlichkeit eines Ausfalls pro Anforderung	
SIL	**Minimum**	**Maximum**
1	10^{-2}	10^{-1}
2	10^{-3}	10^{-2}
3	10^{-4}	10^{-3}
4	10^{-5}	10^{-4}

ISO 26262

In ISO 26262-5 sind mögliche Zielwerte für zufällige Hardwareausfälle, die zu einem Versagen führen, beschrieben. Diese sind jeweils auf ein Sicherheitsziel bezogen und stellen in diesem Standard eine von mehreren Möglichkeiten dar, solche Zielwerte festzulegen. Diese werden in Tab. 4.18 den ASIL-Kategorien der Automobilindustrie zugeordnet.

Tab. 4.18 ASIL nach DIN EN 26262 und Zielrate zufälliger Hardwareausfälle, die zu einem Versagen bzgl. eines Schutzzieles führen

ASIL	**Zielrate zufälliger Hardwareausfälle, die zu einem Versagen bzgl. eines Schutzzieles führen, pro Stunde**
A	Keine Angabe
B	$<10^{-7}$ (Empfehlung)
C	$<10^{-7}$
D	$<10^{-8}$

DIN EN 50128

Die DIN EN 50128 (Bahn) führt Techniken und Maßnahmen für 5 Software-Sicherheits-Integritätslevel (Software-SIL 0 bis 4) auf, wobei die geforderten Techniken für Level 1 dieselben sind wie die für Level 2 und die geforderten Techniken für Level 3 dieselben wie für Level 4.

Die Norm gibt keine Hinweise darauf, welche Software-SIL für ein gegebenes Risiko angemessen ist, sondern verweist auf die Normen DIN EN 50126 „Spezifikation und Nachweis der Zuverlässigkeit, Verfügbarkeit, Instandhaltbarkeit, Sicherheit (RAMS)“ für den Bahnbereich sowie auf DIN EN 50129 „Bahnanwendungen - Telekommunikationstechnik, Signaltechnik und Datenverarbeitungssysteme - Sicherheitsrelevante elektronische Systeme für Signaltechnik“.

In DIN EN 50129 werden im normativen Anhang A tolerierbare Raten gefährlicher Ausfälle pro Stunde und pro Funktion angegeben, wobei angemerkt wird, dass diese Tabelle auf sicherheitsrelevante Funktionen oder Teilsysteme angewandt werden kann, die eine oder mehrere solcher Funktionen beinhalten (siehe Tab. 4.19).

Die Werte sind identisch mit denen der Norm DIN EN 61508 für den „high demand mode“ (siehe Tab. 4.15). Werte für ein gefährliches Versagen pro Anforderung („low demand mode“ aus DIN EN 61508) sind in DIN EN 50129 nicht angegeben. Im Text wird darauf hingewiesen, dass Tab. 4.18 unter Hinzuziehung von DIN EN 61508 entstanden ist. Somit wird für die folgenden Betrachtungen davon ausgegangen, dass die SIL nach EN 61508 und EN 50129 identifiziert werden können.

Tab. 4.19 SIL nach DIN EN 50129 und tolerierbare Gefährdungsrate pro Stunde und pro Funktion

	Tolerierbare Gefährdungsrate THR pro Stunde und pro Funktion	
Sicherheitsanforderungs-stufe	Minimum	Supremum
1	10^{-6}	10^{-5}
2	10^{-7}	10^{-6}
3	10^{-8}	10^{-7}

4	10^{-9}	10^{-8}

Ergebnis

Der Vergleich der Ausfallraten aus EN ISO 13849 mit denjenigen aus DIN EN 61508 ist in Übereinstimmung mit der Tabelle 4.14. Die Diskussion zur DIN EN 50128 zeigt, dass die SIL dieser Norm mit den SIL der DIN EN 61508 identifiziert werden können.

Für die ISO 26262 stellt sich die Situation komplizierter dar, da

1. sich die in der Norm angegebenen Raten nur auf Hardwareausfälle beziehen,
2. sich die Raten auf ein Schutzziel, nicht auf die einzelne Funktion beziehen und
3. nur für ASIL C und D verbindliche Raten angegeben werden, während für ASIL B nur ein Wert empfohlen wird und für ASIL A gar keine Angabe gemacht wird. Die Werte für ASIL B und C sind identisch, d. h. der Unterschied besteht nur in der Verbindlichkeit.

Nur durch das Treffen von Annahmen über den Anteil der Hardwareausfälle an den gefährlichen Ausfällen und der Anzahl von Sicherheitsfunktionen, die einem Sicherheitsziel dienen, lassen sich Beziehungen zwischen ASIL und SIL ableiten. Wenn man davon ausgeht, dass Hardwarefehler nur einen Teil der gesamten Rate gefährlicher Ausfälle verursachen (z. B. 30%), und die Anzahl der Sicherheitsfunktionen für jedes Sicherheitsziel klein ist (z. B. 4), ergibt sich eine ungefähre Korrespondenz von ASIL C zu SIL 2 und ASIL D zu SIL 3.

In Tab. 4.20 werden die aus dem Vergleich der Ausfallraten bestimmten Entsprechungen dargestellt.

Tab. 4.20 Entsprechungen der Kategorien basierend auf in den Standards angegebenen Ausfallraten

Gefährliche Ausfallrate		EN ISO 13849-1	DIN EN 61508	DIN EN 50129	ISO 26262[2]
Supremum	**Minimum**	**PL**	**SIL**	**SIL**	**ASIL**
10^{-4}	10^{-5}	a			
10^{-5}	$3 * 10^{-6}$	b	1	1	
$3 * 10^{-6}$	10^{-6}	c			
10^{-6}	10^{-7}	d	2	2	C
10^{-7}	10^{-8}	e	3	3	D
10^{-8}	10^{-9}		4	4	

4.5.5 Vergleich von SIL und ASIL basierend auf in den Normen genannten Vorgehensweisen

Um die Vergleichbarkeit insbesondere auch der Kategorien A und B des Automotive Safety Integrity Level (Standard ISO 26262) zu untersuchen, wurde ein weiter Ansatz entwickelt, der die auf in den Normen genannten beispielhaften Vorgehensweisen zur Bestimmung von SIL bzw. ASIL basiert.

Es wurden in ISO 26262 und ISO 61508 aufgeführte Vorgehensweisen betrachtet, bei der aufgrund von Eingangsgrößen wie möglichen Konsequenzen, Beherrschbarkeit bei Versagen der Sicherheitsleittechnik die SIL- bzw. ASIL-Kategorie bestimmt wird. Der Vergleich basiert auf ISO 26262-3 Abschnitt 7.4.4 Determination of ASIL and safety goals und der im informativen Anhang E-6 von ISO 61508-5 dargestellten „Risk graph"-Vorgehensweise (Abbildung E.1 zusammen mit Tabelle E.2).

In ISO 26262-3 werden folgende Eingangsgrößen, die jeweils 4-5 Stufen annehmen können, verwendet:

- Severity S0 – S3 (Konsequenzen)

[2] ungefähre Übereinstimmung basierend auf oben genannten Annahmen

- Probability of exposure regarding operational situations E0 – E4
- Controllability C0 – C3

In ISO 61508-5 Anhang E-6 werden folgende Eingangsgrößen verwendet:

- Consequence C_A – C_D
- Frequency of, and exposure time in, the hazardous zone F_A – F_B
- Possibility of avoiding the hazardous event P_A – P_B
- Probability of the unwanted occurrence W_A – W_C

Die Eingangsgrößen „Severity“ und „Consequence“ beschreiben die möglichen Konsequenzen. Sie haben eine unterschiedliche Skala. Allerdings lassen sich nach den Bezeichnungen drei weitgehend äquivalente Stufen identifizieren (siehe Tab. 4.21).

Tab. 4.21 Vergleich der Klassen der Konsequenzen

61508		**26262**	
		S0	No injuries
C_A	Minor injury	S1	Light and moderate injuries
C_B	Erwartete Anzahl Tote 0,01 bis 0,1	S2	Severe and life-threatening injuries (survival probable)
C_C	Erwartete Anzahl Tote 0,1 bis 1	S3	Life-threatening injuries (survival uncertain), fatal injuries
C_D	Erwartete Anzahl Tote > 1		

Die Eingangsgrößen „Probability of exposure regarding operational situations“ und „Frequency of, and exposure time in, the hazardous zone“ beschreiben den Anteil der Zeit bzw. die Häufigkeit, bei denen Gefahren auftreten können. Sie haben ebenfalls eine unterschiedliche Skala, wobei die ISO 61508-5 nur zwei Stufen aufweist. Eine Gegenüberstellung findet sich in Tabelle 4.22. Die Eingangsgrößen „Controllability“ und „Possibility of avoiding the hazardous event“ beschreiben die Möglichkeit, durch Handeingriffe das Ereignis zu beherrschen. Sie haben ebenfalls eine unterschiedliche Skala, wobei ISO 61508-5 wiederum nur zwei Stufen aufweist. Eine Gegenüberstellung findet sich in Tabelle 4.23.

Tab. 4.22 Vergleich der Klassen der Anteil der Zeit bzw. der Häufigkeit, bei denen Gefahren auftreten können

61508		26262	
		E0	Incredible
		E1	Very low probability
F_A	Rare to more often exposure in the hazardous zone. Occupancy less than 0,1	E2	Low probability
		E3	Medium probabiity
F_B	Frequent to permanent exposure in the hazardous zone	E4	High probability

Tab. 4.23 Vergleich der Klassen der Möglichkeit, bei Ausfall der Sicherheitseinrichtung durch Handmaßnahmen einen Schaden zu vermeiden

61508		26262	
		C0	Controllable in general
P_A	P_A should only be selected if all the following are true: − facilities are provided to alert the operator that the safety instrumented systems has failed; − independent facilities are provided to shut down such that the hazard can be avoided or which enable all persons to escape to a safe area; − the time between the operator being alerted and a hazardous event occurring exceeds 1 h or is definitely sufficient for the necessary actions.	C1	Simply controllable
		C2	Normally controllable
P_B	Adopted if all the conditions are not satisfied	C3	Difficult to control or uncontrollable

In ISO 61508-5 Anhang E-6 wird eine weitere Eingangsgröße betrachtet, die die Wahrscheinlichkeit beschreiben soll, dass es ohne Eingriff eines Sicherheitssystems zum unerwünschten Ereignis kommt. Da es keine Entsprechung in ISO 26262-3 gibt, wird für den Vergleich die höchste Rate W_3 zugrunde gelegt, weil davon ausgegangen werden kann, dass beim Autoverkehr im Allgemeinen gefährliche Situationen nicht ohne einen Eingriff (manuell oder durch ein automatisches Sicherheitssystem) beherrscht werden können.

Es werden nun die verschiedenen Kombinationen von als entsprechend angesehenen Klassifizierungen betrachtet und die aus dem Graphen bzw. der Tabelle der SIL bzw. ASIL-Wert bestimmt; z. B. entspricht C_A „Minor injury“ S1 „Light and moderate injuries“. Aus C_A folgt direkt, dass kein SIL-Niveau erforderlich ist. Aus S1 folgt — je nach den weiteren Klassifizierungen — kein ASIL (QM), ASIL A oder ASIL B. Die so gefundenen Entsprechungen wurden als „x“ in Tab. 4.24 eingetragen.

Tab. 4.24 Nach dem beschriebenen Verfahren gefundene Entsprechungen der SIL und ASIL (x) und nach Plausibilitätsbetrachtung ergänzte weitere Entsprechungen (p) sowie aus Tab. 4.20 übernommene Entsprechung aufgrund probabilistischer Überlegungen (n)

	Kein ASIL (QM)	**ASIL A**	**ASIL B**	**ASIL C**	**ASIL D**
Kein SIL	x	x	x		
SIL 1	x	p	p		
SIL 2	x	x	p	n	
SIL 3			x	x	x, n
SIL 4					x

Wie Tab. 4.24 zu entnehmen ist, gibt es keine einfache Entsprechung von SIL und ASIL. Weiterhin sind „Lücken“ erkennbar, z. B. ergab das Verfahren bei Kombinationen eine Entsprechung von „SIL 1“ mit „kein ASIL“, während „Kein SIL“ mit „kein ASIL“, „ASIL 1“ und „ASIL 2“ ergab. Da die Sicherheitsanforderungen mit steigendem SIL und ASIL ansteigen und die Forderungen niedriger Stufen diejenigen hoher Stufen umfassen (D. h. wenn z. B. eine Einrichtung mit SIL 1 zulässig ist, ist auch eine mit SIL 2 erlaubt)

wurden auch alle sich so ergebenen Kombinationen in Tab. 4.24 zusätzlich einbezogen. Diese wurden in Tab. 4.24 mit „p“ gekennzeichnet. Weiterhin wurden die Ergebnisse aus dem Vergleich der Ausfallraten (Abschnitt 4.5.4, Tab. 4.20) als „n“ eingetragen.

4.5.6 Zusammenfassung

Die Beziehung der verschiedenen Kategorien wird zusammenfassend in Tab. 4.25 dargestellt.

Tab. 4.25 Beziehung der Kategorisierungen nach EN ISO 13849, DIN EN 61508,DIN EN 50129, ISO 26262 und dem deutsches Kerntechnischen Regelwerk zum Zwecke des Vergleiches der Anforderungen an die Softwareentwicklung

EN ISO 13849-1	**DIN EN 61508**	**DIN EN 50129**	**ISO 26262[3]**	**Deutsches Kerntechnisches Regelwerk**	
PL	**SIL**	**SIL**	**ASIL**	**Sicherheitsebene**	**Kategorie**
a					
b	1	1	(A, B)		
c				1	C
d	2	2	C, (A, B)	2	B
e	3	3	D, (B, C)		
	4	4	(D)		
Zwei diversitäre Einrichtungen mit e und c				3	A

[3] **Angegeben sind die sich aus den Ausfallraten unter den in Abschnitt 4.5.4 ergebenen Zuordnungen. Weitere Zuordnungen nach**

Tab. 4.24 sind in Klammern gesetzt.

5 Qualifizierung

Bei der Qualifizierung handelt es sich nach DIN EN 61513 um einen Vorgang, in Rahmen dessen die Eignung eines Systems oder einer Komponente für den betrieblichen Gebrauch bestimmt wird. In der Literatur ist der Begriff Qualifizierung daher eng verbunden mit den Begriffen Verifizierung und Validierung. Je nach Anwendungsgebiet wird die Qualifizierung als Oberbegriff für Verifizierung und Validierung oder als Teilmenge der Verifizierung verwendet. Vereinfacht dargestellt soll die Validierung die Frage „Wird das richtige Produkt entwickelt“ und Verifizierung die Frage „Wird das Produkt richtig erstellt“ beantwortet werden.

Im Rahmen des Vergleichs des Sicherheitslebenszyklus und des Software-Lebenszyklus wurde in Kapitel 3 bereits auf Anforderungen an Verifizierung und Validierung von Software eingegangen. Im Folgenden werden weiterführende Anforderungen zur Verifizierung und Validierung von Software, die über den Lebenszyklus hinausgehen, behandelt und beschrieben. Im Rahmen dieses Kapitels werden somit die Arbeiten zu Arbeitspaket 3 (AP 3) durchgeführt.

Die Systemvalidierung wird in DIN EN 60880 (Kerntechnik) definiert (siehe Anhang A) als die Bestätigung durch Prüfung und Beibringen anderer Nachweise, dass ein System zur Gänze die Anforderungsspezifikation erfüllt (Funktionalität, Ansprechzeit, Fehlertoleranz, Robustheit). Sie prüft die Übereinstimmung des Systems mit den Spezifikationen der Anwendungsfunktionen geprüft wird. Ergänzend dazu dient die funktionale Validierung der Prüfung der Übereinstimmung von Spezifikationen der Anwendungsfunktionen mit den originären funktionalen und Leistungsfähigkeitsanforderungen der Anlage. (DIN IEC 60880)

Neben den systembezogenen Definitionen von Validierung wird in DIN EN 62138 die Softwarevalidierung definiert als der Test und die Überprüfung der integrierten Software, um Übereinstimmung mit den in der Spezifikation des leittechnischen Systems vorgegebenen Anforderungen an Funktionalität, Leistungsfähigkeit und Schnittstellen sicherzustellen. Außerdem wird nachgewiesen, dass die Software die von ihr geforderte Funktion in der vorgesehenen Umgebung erfüllt.

Die Verifizierung soll nach DIN EN 62138 durch Prüfen und Vorlegen objektiver Beweise sicherstellen, dass die Ergebnisse einer Tätigkeit den Zielen und Anforderungen entsprechen, die für diese Tätigkeit definiert wurden.

5.1 Anforderungen aus allgemeinen Normen an die Verifizierung und Validierung

Nach DIN EN 61508 muss eine Validierung erfolgen, um zu zeigen, dass das sicherheitsrelevante E/E/PE-System der Spezifikation entspricht. Dazu wird gefordert, dass Validierungsaktivitäten durchgeführt werden sollen und dass alle Geräte, die für eine quantitative Messung als Teil der Validierungsaktivitäten genutzt werden, entsprechend der Spezifikationen eines nationalen Standards oder der Spezifikationen des Lieferanten kalibriert werden sollen. Außerdem werden Anforderungen an die Dokumentation der Validierung beschrieben. Auch Abweichungen zwischen erwarteten und tatsächlichen Ergebnissen und über die Entscheidungen für das weitere Vorgehen sind zu dokumentieren. Die hauptsächliche Validierungsmethode für Software ist nach DIN EN 61508 der Test, der durch Analyse, Simulation und Modellbildung unterstützt werden darf. Außerdem muss der Lieferant und/ oder Entwickler der Software dem Systementwickler alle Validierungsergebnisse bezüglich der Sicherheit und alle weiteren relevanten Dokumente verfügbar machen.

Teil 2 der Norm DIN EN ISO 13849 befasst sich ausschließlich mit dem Thema der Validierung. Dabei wird auf Validierungsverfahren, -leitsätze und -pläne eingegangen. Zweck des Validierungsverfahrens ist die Bestätigung, dass die Gestaltung der SRP/CS[4] die Spezifikation der Sicherheitsanforderungen der Maschinen unterstützt. Dabei muss gezeigt werden, dass die in Teil 1 der Norm formulierten Anforderungen durch das SRP/CS erfüllt werden. Die Validierung sollte von Personen durchgeführt werden, die unabhängig von der Gestaltung der SRP/CS sind. Die Validierung besteht aus der Durchführung der Analyse und von Funktionsprüfungen unter vorhersehbaren Bedingungen in Übereinstimmung mit dem Validierungsplan. Sowohl an die Validierung durch Analyse als auch durch Prüfung werden in Teil 2 der Norm Anforderungen gestellt. Dabei muss die Validierung für das SRP/CS oder eine Kombination von SRP/CSs nachweisen, dass die in der Spezifikation der Sicherheitsanforderungen geforderten PL und Kategorien erfüllt werden. Grundsätzlich erfordert dies eine Fehleranalyse anhand von Schaltplänen und, falls die Fehleranalyse nicht schlüssig ist, Prüfungen durch Fehlersimulationen in den tatsächlich vorhandenen Steuerkreisen und Simulation des Verhaltens der Steuerung beim Auftreten von Fehlern. /ISO 15/

[4] SRP/CS: Sicherheitsbezogenes Teil einer Steuerung.

Außerdem werden Anforderungen an die Validierung der Spezifikation von Sicherheitsanforderungen an die Sicherheitsfunktion und der Sicherheitsfunktionen selber sowie der sicherheitsbezogenen Software formuliert. Weiterhin wird die Validierung der Umgebungsanforderungen, der Instandhaltungsanforderungen und der technischen Dokumentation und Benutzerinformation behandelt. Im informativen Anhang wird zudem auf Validierungswerkzeuge für mechanische, pneumatische, hydraulische und elektrische Systeme eingegangen. /ISO 15/

In IEEE 1012 sind zum Thema Verifizierung und Validierung von Systemen sowohl Systeme mit Hardware- als auch mit Software-Komponenten inbegriffen. Der Fokus liegt dabei auf der Durchführung des eigentlichen Verifizierungs- und Validierungsprozesses (V&V-Prozess) auf Basis des System-Lebenszyklus und den geforderten Eingangs- und Ausgangsdaten. Ziel dieses Standards ist es, eine allgemeingültige Vorgehensweise für einen V&V-Prozess vorzustellen, der vom System-, Software- oder auch Hardware-Lebenszyklus unterstützt wird. Darüber hinaus soll der Inhalt für den Verifizierungs- und Validierungsplan definiert werden. Die Intensität und Rigorosität des V&V-Prozesses soll hierbei immer der zugrunde liegenden SIL-Klasse angepasst werden. Eine Übersicht, welche V&V-Teilprozesse für Systeme mit bestimmten SIL-Klasse relevant sind, wird in dem Standard ebenfalls gegeben.

Der V&V-Prozess zeigt nach IEEE 1012, ob die Anforderungen korrekt, vollständig, genau, konsistent und prüfbar sind. Die Aufgabe umfasst hierbei die Bewertung, Analyse, Einschätzung, Kontrolle, Inspektion und die Überprüfung der Produkte und Prozesse. Wichtig ist laut Standard hierbei, dass der V&V-Prozess parallel zu allen Schritten des Lebenszyklus durchgeführt wird und nicht als deren Abschluss erfolgt.

Neben einer Erläuterung des Zusammenhangs des V&V-Prozesses und dem Lebenszyklus geht der Standard konkret auf die V&V-Prozesse für die folgenden Themengebiete ein:

- Management
- Akquisition
- Lieferplanung
- Projektierung
- Konfigurationsmanagement

- Anforderungsformulierung des Kunden
- Systembedarfserstellung
- Systemarchitekturerstellung, Software- oder Hardwarekonzept
- Systemimplementierung (Konzept, Design, Qualifizierung…)
- Systemintegration
- Software- und Hardwaremodifikation
- Betrieb
- Wartung
- Außerbetriebnahme.

Zusätzlich beinhaltet IEEE 1012 einige administrative Anforderungen sowie Anforderungen an die Berichterstattung und Dokumentation.

5.2 Anforderungen in der Kerntechnik

Bei der Verifizierung im Software-Lebenszyklus liegt in der kerntechnischen Norm DIN IEC 60880 der Schwerpunkt der Anforderungen beim Verifizierungsteam. Darüber hinaus werden Anforderungen an die verschiedenen Verifizierungstätigkeiten gestellt. Hierzu zählen die Erstellung eines Verifizierungsplans, die Verifizierung der Auslegung und der Realisierung. In DIN IEC 60880 werden Vorgaben zu einem Validierungsplan gemacht. Hier wird explizit gefordert, dass die Validierung von Personen durchgeführt werden muss, die an den Auslegungs- und Realisierungsarbeiten nicht beteiligt waren. Darüber hinaus wird gefordert, dass statische und dynamische Testfälle im Validierungsplan enthalten sein müssen.

Auch der Technical Report IAEA 384 /IAEA 99/ beschäftigt sich mit der Verifizierung und Validierung von Software für die Anwendung in sicherheitstechnisch wichtigen Systemen in Kernkraftwerken. Zu Beginn des Berichts wird die Notwendigkeit der Verifizierung und Validierung diskutiert. Dazu wird zunächst dargestellt, dass eine erfolgreiche Verifizierung und Validierung dann erreicht ist, wenn die Anzahl der Softwarefehler bis auf ein akzeptables Minimum reduziert wurde. Hierbei sind zwei verschiedene Arten von Fehlern relevant: Fehler, die durch eine falsch formulierte Anforderung jedoch mit korrekter Einbindung auftreten und Fehler bei der Umsetzung von Anforderungen.

Im weiteren Verlauf werden verschiedene Arten von Software sowie deren Klassifizierung vorgestellt. Für ein besseres Verständnis der verschiedenen Arten von Software erläutert der Bericht die Unterschiede von neu entwickelter Software, bestehender Software aus ähnlicher Anwendung, bestehender Software aus einer anderen Anwendung und konfigurierter Software.

Weiterhin präsentiert der Bericht die Aufgaben und Notwendigkeiten der verschiedenen Entwicklungsschritte. Hierzu zählen die Spezifikation der Systemanforderungen und des Computersystems, der Software-Entwurf sowie die Integration, Validierung, Abnahme, Betrieb, Wartung und Modifikation von Software. Anschließend wird die Verifizierung der einzelnen Entwicklungsphasen eingehend beschrieben. Jede Phase soll hinsichtlich ihrer geforderten Eingangsdaten, den spezifischen Ausführungen und den erwarteten Ausgangsdaten geprüft werden. Ziel der Verifizierung ist dann, für jede einzelne Phase zu bestätigen, dass das entwickelte Produkt den Anforderungen entspricht. Neben konkreten Empfehlungen für eine erfolgreiche Umsetzung werden auch konkrete Techniken für die Verifizierung und Validierung von Software vorgestellt. Auch wird kurz auf die Verifizierung von bereits bestehender Software sowie die abgeschwächten Anforderungen an die Verifizierung von Software in Leittechniksystemen, die Funktionen der Kategorie B und C ausführen, eingegangen.

Bei der Validierung der Software muss demonstriert werden, dass die Software seine Funktion entsprechend den Spezifikationsanforderungen korrekt ausführt. Sind diese jedoch bereits fehlerhaft, verhindert auch eine erfolgreiche Validierung nicht das Versagen des Systems. Die Systemvalidierung wird nach der Integration der Software, jedoch vor deren Inbetriebnahme, durchgeführt und sollte alle Bestandteile des Gesamtsystems (z.B. auch Sensoren etc.) einschließen. Der Bericht weist weiterhin darauf hin, dass nicht alle Anforderungen, die an die Validierung für Software der Kategorie A gestellt werden, auch für Kategorie B und C erfüllt werden müssen. Dennoch sollte auch bei Software der Kategorie B und C gezeigt werden, dass die an das System gestellten Anforderungen erfüllt sind.

5.3 Anforderungen in der Automobilindustrie

Zweck von Verifizierungsaktivitäten ist nach ISO 26262 der Nachweis, dass die Ergebnisse jeder einzelnen Aktivität mit den spezifizierten Anforderungen übereinstimmen. Zu den Verifizierungsaktivitäten zählen neben der Design-Verifizierung die Sicherheitsana-

lysen, Integration der Hardware und Software und ihr Test. Die Validierung des integrierten Elements in einem repräsentativen Fahrzeug dient dazu, den Nachweis der Eignung für den vorgesehenen Einsatz zu erbringen und die Eignung der Sicherheitsmaßnahmen für eine Klasse oder einen Satz von Fahrzeugen zu bestätigen. Die Sicherheitsvalidierung dient auf Basis von Prüfungen und Tests dem Nachweis, dass die definierten Sicherheitsziele ausreichend sind und erfüllt werden.

Nach ISO 26262 soll die Verifizierung sicherstellen, dass die Arbeitsprodukte ihren Anforderungen entsprechen. Die Verifikation soll in der Konzeptphase, der Produktentwicklungsphase und der Produktions- und Betriebsphase erfolgen. Grundsätzlich werden daher Anforderungen an die Planung, die Spezifikation und die Durchführung der Verifizierung formuliert. Für die in einer Komponente integrierte Software soll verifiziert werden, ob die eingefügte Software die Software-Sicherheitsanforderungen erfüllt. Dies soll innerhalb der Zielumgebung erfolgen. In ISO 26262 werden u. a. folgende Anforderungen an die Verifizierung der Software gestellt:

- Um zu testen, dass die Software die Sicherheitsanforderungen erfüllt, sollen Tests in einer Testumgebung erfolgen. Als Testumgebung kann sowohl ein Fahrzeug als auch ein Testaufbau dienen. Die Tests sollen auf der Zielhardware durchgeführt werden.
- Die Ergebnisse der Verifizierung der Software-Sicherheitsanforderungen sollen in Hinblick auf die Übereinstimmung mit den erwarteten Ergebnissen, der Abdeckung der definierten Software-Sicherheitsanforderungen und den Erfüllungs- oder Ausfallkriterien ermittelt werden.

Ziel der Sicherheitsvalidierung ist nach ISO 26262 der Nachweis, dass die Sicherheitsziele eingehalten werden und dass das funktionale Sicherheitskonzept für das Gesamtsystem angemessen ist. Außerdem soll gezeigt werden, dass die Sicherheitsziele korrekt und vollständig sind und auf Fahrzeugebene erfüllt werden.

In ISO 26262 werden u. a. folgende Anforderungen an die Sicherheitsvalidierung gestellt:

- Sicherheitsziele sollen für die in einem repräsentativen Fahrzeug integrierten Komponente validiert werden.
- Werden Tests zur Validierung eingesetzt, gelten die gleichen Anforderungen wie an Tests bei der Verifizierung.

- Die Sicherheitsziele der Komponente sollen auf Fahrzeugebene unter Berücksichtigung der Steuerbarkeit und der Wirksamkeit von Sicherheitsmaßnahmen zur Kontrolle zufälliger oder systematischer Fehler, von externen Maßnahmen sowie von Elementen anderer Technologie validiert werden.

- Zur Validierung auf Fahrzeugebene sollen basierend auf Sicherheitszielen, Anforderungen an die funktionale Sicherheit und den Anwendungsbereich (Konfiguration, Umgebungsbedingungen, Fahrsituationen, etc.) Validierungsprozeduren und Testfälle für jedes Sicherheitsziel, inkl. Ausfall- und Erfüllungskriterien, verwendet werden.

- Ein angemessener Satz der folgenden Methoden soll angewandt werden:
    - Wiederholbare Tests mit spezifizierten Testprozeduren, Testfällen und Erfüllungs-/Ausfallkriterien
    - Analysen wie z. B. FMEA, Fehlerbaumanalyse, Ereignisbaumanalyse, Simulation
    - Langzeittests
    - Anwendertests unter realen Bedingungen
    - Reviews.

5.4 Anforderungen beim Schienenverkehr

Nach DIN EN 50126 dient die Verifizierung durch Überprüfung und objektiven Nachweis der Bestätigung, dass die besonderen Anforderungen für einen spezifischen, bestimmungsgemäßen Gebrauch erfüllt werden. Verifizierungsaktivitäten sind zum Abschluss jeder Phase im System-Lebenszyklus erforderlich. Neben für jede Phase spezifischen Verifizierungsaktivitäten werden in jeder Phase folgende Aktivitäten gefordert:

- Bewertung der Angemessenheit der Informationen sowie ggf. der Daten und sonstigen Statistiken, die als Eingangsgrößen in dieser Phase verwendet werden.

- Bewertung der Angemessenheit der eingesetzten Verfahren, Werkzeuge und Techniken.

- Bewertung der Qualifikation aller Mitarbeiter, welche die Aufgaben in dieser Phase durchführen.

Auf Anforderungen an die Verifizierung und Validierung in Hinblick auf Software wird in DIN EN 50128 eingegangen. Ziel der Software-Verifizierung ist nach DIN EN 50128 die Untersuchung und der Nachweis, dass die Ergebnisse einer bestimmten Entwicklungsphase den Anforderungen und Plänen hinsichtlich der Vollständigkeit, Richtigkeit und Konsistenz genügen. Diese Aktivitäten werden vom Verifizierer gelenkt. Die Verifizierung muss mindestens durch einen Software-Verifizierungsplan und Verifizierungsberichte dokumentiert werden. Sie sind unter Verantwortung des Verifizierers zu erstellen. Im Verifizierungsplan müssen die durchzuführenden Aktivitäten sowie alle anzuwendenden Kriterien, Werkzeuge und Techniken beschrieben werden. In jeder Entwicklungsphase muss gezeigt werden, dass die Anforderungen an Funktion, Leistungsverhalten und Sicherheit erfüllt werden. Die Verifizierungsberichte müssen folgendes dokumentieren:

- Identität und Konfiguration der verifizierten Teile,
- Name des Verifizierers,
- getroffene Annahmen,
- entdeckte Fehler und Mängel,
- dem Problem unzureichend angepasste Komponenten, Strukturen, Daten und Algorithmen und
- Punkte, die nicht der Spezifikation entsprechen.

Im Verifizierungsplan gemäß DIN EN 50128 wird im Vergleich zu den Anforderungen in der Kerntechnik gefordert, dass die Rollen und Verantwortlichkeiten der am Prüfprozess beteiligten Personen in der Prüfspezifikation festgeschrieben werden müssen. Außerdem wird gefordert, dass die Prüfungen wiederholbar sein müssen und falls praktikabel mit automatischen Hilfsmitteln durchgeführt werden. Die Test-Skriptfiles für die automatische Testausführung müssen verifiziert werden.

Während in DIN EN 61508 nur sehr allgemein gefordert wird, dass für jede Phase des Lebenszyklus ein Verifizierungsplan aufzustellen ist und die Ergebnisse zu dokumentieren sind, wird beispielsweise in DIN EN 50126 in jeder Phase aufgeführt, welche Verifizierungsaufgaben zu erfüllen und welche Ergebnisse und Informationen zu dokumentieren sind.

Ein weiteres Ziel der Software-Validierung nach DIN EN 50128 ist der Nachweis durch Analyse und Tests, dass die Software dem festgelegten SIL entspricht, die Software-

Anforderungen erfüllt werden und die Software sich für ihre vorgesehene Anwendung eignet. Basierend auf den Ergebnissen sind Anomalien und Mängel bezüglich ihrer Sicherheitsrelevanz zu bewerten. Diese Aktivität wird vom Validierer durchgeführt. Wie bei der Verifizierung muss auch bei der Validierung ein Software-Validierungsplan und ein Software-Validierungsbericht vom Validierer erstellt werden.

Der Software-Validierungsplan muss die Validierungsstrategie beschreiben. Hierbei sollen verschiedene Techniken eingesetzt werden, für die gemäß DIN EN 50128 mehrere Auswahlmöglichkeiten zur Verfügung stehen:

- Manuelle oder automatische oder beide Techniken,
- Statische oder dynamische oder beide Techniken,
- Analytische oder statische oder beide Techniken,
- Testen in realer oder simulierter oder in beiden Umgebungen.

Darüber hinaus muss der Software-Validierungsplan aufführen, welche Schritte notwendig sind, um die Angemessenheit

- jeder Software-Spezifikation bei der Erfüllung der Sicherheitsanforderungen und
- der Gesamtsoftware-Testspezifikation als ein Test gegenüber der Software-Anforderungsspezifikation zu demonstrieren.

Die Ergebnisse der Validierung müssen dann im Software-Validierungsbericht dokumentiert werden. Mängel und Fehler in der Software sowie ihr Einfluss auf die Anwendung der Software müssen verdeutlicht werden und hinsichtlich des entsprechenden SILs bewertet werden.

Zu jeder Software muss darüber hinaus eine Freigabemitteilung erstellt werden, die alle Einschränkungen bei der Nutzung der Software enthalten muss. Diese Einschränkungen werden aus den festgestellten Fehlern, den Nichtübereinstimmungen mit der Norm, dem Grad der Erfüllung der Anforderungen und der Pläne abgeleitet.

In einem abschließenden Software-Validierungsbericht muss die Konsistenz der Validierung und des Validierungsplans geprüft und dargelegt werden.

5.5 Anforderungen in der Wehrtechnik

Das Handbuch der Wehrtechnik /DOD 10/ beschreibt eine formale Methode zur Software-Qualifizierung. Diese besteht aus einer spezifischen Folge von Testfällen, die eingesetzt werden, um Belege für die Sicherheitszertifizierung zur Verfügung zu stellen. Formale Qualifizierungstests sollen auf höchster Stufe der Integration durchgeführt werden, da Tests, die auf der gesamten Softwareanwendung (alle Software-Module und Komponenten stehen miteinander und mit der entsprechenden Hardware in Wechselwirkung) laufen, am genauesten die tatsächliche Hardware-Software-Konfiguration wiedergeben. Dennoch ist es manchmal notwendig, Testfälle auf niedrigerer Integrationsstufe durchzuführen, um die notwendige Test-Abdeckung zu erreichen (z. B. zur Ausführung des gesamten Software-Codes). Alle Testfälle sollten dabei immer mit der Angabe der Integrationsstufe dokumentiert werden.

Werden bei den Qualifizierungstests Ergebnisse erzielt, die eine Überarbeitung der Software notwendig machen, muss es möglich sein, Änderungen am Code rückgängig zu machen und erneute Änderungen einzufügen. In /DOD 10/ werden daher folgende Kriterien definiert:

- Durchführung von Regressionstests für jede neue Software-Kompilierung zur Verifizierung, dass zwischenzeitlich neu durchgeführte Kompilierungen nicht die vorher getestete und qualifizierte sicherheitskritische Funktionalität des Systems beeinträchtigt.
- Durchführung von Sicherheits-Regressionstests für Software-Modifizierungen und Revisionen innerhalb einer kompilierten Version der Software, die nach der formalen Qualifizierung durchgeführt wurde.
- Erzeugung eines minimalen Satzes an Testfällen, um die gesamte sicherheitskritische Funktionalität zu testen.

Bei der Planung von Software-Tests müssen alle Simulatoren, Modelle, Emulatoren und Software-Werkzeuge berücksichtigt werden, die von dem Test-Team eingesetzt werden. Dies soll sicherstellen, dass alle Prozesse, Anforderungen und Prozeduren für die Validierung etabliert sind. Diese Validierung muss vor dem tatsächlichen Einsatz der Software durchgeführt werden. Ein Software-Sicherheitstestplan soll nach /DOD 10/ Vorgaben dazu enthalten, in welcher Form der Software-Sicherheitsingenieur an den Meetings der Arbeitsgruppe zum Testen der Software teilnehmen soll und welche Informationen zur Verfügung gestellt werden müssen.

Der Sicherheitsmanager muss die Planung der Software-Sicherheitstest in den gesamten Software-Testplan integrieren. Diese Integration soll die Identifikation sämtlichen sicherheitskritischen Codes enthalten.

Der Standard /AOP 52/ behandelt die Qualifizierung von Prozessen und Werkzeugen zur Prüfung. Demnach müssen Nachweise erbracht werden, dass für die verwendeten Prozesse und Werkzeuge ausreichende Sicherheitskontrollen durchgeführt wurden, so dass sichergestellt wird, dass sie die Integrität von komplexen elektronischen Elementen nicht beeinträchtigen. Darüber hinaus müssen Nachweise über die geforderte Kompetenz der Mitarbeiter, die für die Durchführung der Prozesse verantwortlich sind, erbracht werden.

5.6 Anforderungen in der Luftfahrt

Die Verifizierung ist nach DO-178C /RTCA 11/ eine technische Bewertung der Ergebnisse des Software-Planungsprozesses, der Software-Entwicklungsprozesse und des Software-Verifizierungsprozesses. Der Ablauf des Software-Verifizierungsprozesses wird im Software-Planungsprozess und im Software-Verifizierungsplan festgelegt. Verifizierungsaktivitäten bestehen typischerweise aus einer Kombination von Reviews, Analysen, der Entwicklung von Testfällen und Prozeduren sowie der zwischenzeitlichen Durchführung dieser Testprozeduren.

Ziel des Verifizierungsprozesses ist nach DO-178C die Erkennung und die Dokumentation von Fehlern, die während des Software-Entwicklungsprozesses eingefügt worden sein könnten. Das Entfernen solcher Fehler ist eine Aufgabe des Software-Entwicklungsprozesses.

Bei der Verifizierung ist folgendes zu beachten:

- Ist der getestete Code nicht identisch zu der im Flugzeug verbauten Software, müssen die Unterschiede spezifiziert und begründet werden.
- Wenn es nicht möglich ist, spezifische Software-Anforderungen durch Ausführung der Software in einer realistischen Testumgebung zu verifizieren, müssen andere Test-Mittel verfügbar sein. Die Begründung für die Anwendung dieser Mittel muss im Software-Verifizierungsplan oder in den Software-Verifizierungsergebnissen zu finden sein.

- Mängel und Fehler, die während des Verifizierungsprozesses entdeckt werden, sollen berichtet, in der nachfolgenden Lebenszyklusphase geklärt und wenn möglich behoben werden. Eine anschließende, erneute Verifizierung der Änderungen soll sicherstellen, dass die Modifizierung korrekt implementiert wurde und keinen Einfluss auf die bereits verifizierte Funktionalität hat.
- Um die Unabhängigkeit zu gewährleisten, soll eine andere Person als der Entwickler der Software die Verifizierung durchführen.

5.7 Fazit

In allen Normen und Standards wird als Ziel der Qualifizierung beziehungsweise der Verifizierung und Validierung genannt, dass der Nachweis erbracht werden soll, dass alle Anforderungen korrekt umgesetzt wurden und während der Entwicklung keine Fehler aufgetreten sind. Wie dieser Nachweis in den einzelnen nicht-nuklearen Bereichen erbracht werden muss, ist in den vorangegangen Kapiteln beschrieben. Vergleicht man die an die Qualifizierung gestellten Anforderungen mit denen aus nationalen und internationalen Normen und Standards aus dem Bereich der Kerntechnik, so wird deutlich, dass in kerntechnischen Standards und Normen bereits detaillierte Anforderungen enthalten sind. Andere Industriezweige sind bei diesem Thema größtenteils weniger detailliert als die vorhandenen Anforderungen in der Kerntechnik. Deswegen besteht keine Notwendigkeit Anforderungen zur Qualifizierung aus nicht-nuklearen Bereichen zu übertragen.

Für den Aspekt der Dokumentation und Berichterstattung gehen die Vorgaben und Empfehlungen insbesondere im Schienenverkehr jedoch über die Kerntechnik hinaus. Es ist durchaus sinnvoll entsprechend detaillierte Vorgaben an die Dokumentation und Berichterstattung auch für die Kerntechnik aufzustellen. Eine Übertragung aus dem Schienenverkehr ist hierbei möglich.

Auch hat sich gezeigt, dass im Schienenverkehr und in der Automobilindustrie konkrete Vorgehensweisen und Empfehlungen für Methoden der Qualifizierung gegeben werden. So wird zum Beispiel in der Automobilindustrie ein angemessener Satz der Methoden

- Wiederholbare Tests mit spezifizierten Testprozeduren, Testfällen und Erfüllungs-/Ausfallkriterien,

- Analysen wie z. B. FMEA, Fehlerbaumanalyse, Ereignisbaumanalyse, Simulation,
- Langzeittests,
- Anwendertests unter realen Bedingungen,
- Reviews

empfohlen. In der Kerntechnik werden lediglich statische und dynamische Tests im Validierungsplan gefordert. Konkretere Anforderungen an die Methoden werden nicht gestellt. Eine detaillierte Diskussion der Übertragbarkeit von Methoden und Modellen findet sich in Kapitel 6.

6 Zuverlässigkeitsbewertung

In diesem Kapitel werden die qualitativen und quantitativen Methoden zur Bewertung und Quantifizierung der Zuverlässigkeit sowie die diesbezüglich relevanten Anforderungen in der Kerntechnik und in nicht-nuklearen Industriezweigen vorgestellt. Im weiteren Verlauf werden die Anforderungen an die Zuverlässigkeitsbewertung sowie Empfehlungen zu Methoden verglichen, um anschließend zu diskutieren, inwiefern einzelne Aspekte auf den Bereich der Kerntechnik übertragen werden können. Die in diesem Kapitel durchgeführten Arbeiten entsprechen der Bewertung und Quantifizierung der Systemzuverlässigkeit zu Arbeitspaket 4 (AP 4) und der Beschreibung der Methoden zu Arbeitspaketen 5 (AP 5).

6.1 Qualitative Methoden

In diesem Abschnitt werden qualitative Methoden zur Bewertung der Zuverlässigkeit vorgestellt. Oftmals können qualitative Methoden bei entsprechender Kenntnis der Ausfallwahrscheinlichkeit auch quantitativ eingesetzt werden. Gerade bei Software sind diese Ausfallwahrscheinlichkeiten jedoch sehr schwer zu ermitteln.

In dem nicht-nuklearen Standard der European Cooperation for Space Standardization “Software Dependability and safety” /ECSS 80/ werden die folgenden Methoden beschrieben:

- SFMEA (Software Failure Modes and Effects Analysis)
    - SFMECA (Software Failure Modes, Effects and Criticality Analysis)
- SFTA (Software Fault Tree Analysis)
- SCCA (Software Common Cause Analysis)
- Dynamic Flowgraph methodology (DFM)

Im Folgenden wird kurz auf die Methoden eingegangen, ihre Anwendbarkeit auf Software und ihre Vor- und Nachteile kurz erläutert. Eine detailliertere Beschreibung dieser Methoden findet sich in Anhang D.

- **Software Failure Modes and Effects Analysis (SFMEA)**

  Aufgrund der wachsenden Bedeutung rechnerbasierter und programmierbarer Systeme wurde die ursprünglich für Hardware entwickelte Fehlerzustandsart- und auswirkungsanalyse (Failure Modes and Effects Analysis, FMEA) auch für Software weiterentwickelt und der Begriff Software FMEA (SFMEA) eingeführt /STUK 02/.

  Die in unterschiedlichen Industriezweigen eingesetzte FMEA verwendet verschiedene, auf die einzelnen Bereiche zugeschnittene Standards mit dem Ziel, durch ein systematisches Vorgehen bei der Analyse, mögliche Fehlerzustandsarten, ihre Ursachen und ihre Auswirkungen auf das Systemverhalten zu ermitteln. In der Luft- und Raumfahrt wird der MIL-STD-1629A Standard /MIL 80/ verwendet, die Automobilindustrie befolgt den Standard SAE J-1739 /SAE 09/ und viele andere Industriezweige verwenden den Standard IEC 60812 /IEC 01/.

  Die Methode zeichnet sich dadurch aus, dass sie vorzugsweise in der frühen Entwicklungsphase eingesetzt wird, um eine frühzeitige Fehlerbehebung und gegebenenfalls Entwurfsänderungen zu ermöglichen, und iterativ den gesamten Entwurfsprozess begleitet /IEC 01/.

- **Software Failure Modes, Effects and Criticality Analysis (SFMECA)**

  Der Vorteil der SFMECA ist, dass sie ein frühzeitiges Erkennen von Schwächen im Entwurf ermöglicht. Auf diese Weise lassen sich Ausfälle erkennen, die alleine oder in Kombination mit anderen Ausfällen auftreten und unzulässige oder den Betrieb ernstlich gefährdende Auswirkungen haben.

  Der Nachteil der SFMECA ist, dass sie bei der Analyse von GVA nur sehr begrenzt einsetzbar ist. Der Grund dafür ist, dass einzelne Ausfallarten und die Auswirkungen dieser Ausfallarten auf das System als voneinander unabhängig betrachtet werden. Zwar gibt es qualitative Methoden der SFMECA zur Analyse von Ausfällen mit gemeinsamer Ursache, es wird jedoch empfohlen auf andere Methoden wie beispielsweise die Markov-Methode oder die Fehlerbaumanalyse zurückzugreifen /IEC 01/.

- **Software Fault Tree Analysis (SFTA)**

  Bei der SFTA wird der vorhandene Quellcode direkt in die FTA-Systemanalyse eingebunden. Unerwünschte Ereignisse, die von Softwarekomponenten stammen, werden dann einer Software-Fehlerbaumanalyse unterzogen. Die SFTA wird rückwärts von der Ausgabe zur Eingabe hin analysiert (Top-down Analyse), so dass die Ein-

gaben ermittelt werden können, die zu Fehlern führen. Schrittweise wird jede Programmanweisung (while-Schleifen, if-Anweisungen und Zuweisungen etc.) mit Hilfe sogenannter Fehlerbaummuster (fault tree templates) analysiert. Im Unterschied zur Hardware, deren Fehlerbäume auch probabilistischer Natur sein können, folgt der Quellcode einem logischen Konstrukt und ist daher deterministisch. Gelingt es zu zeigen, dass die getroffenen Annahmen für das Eintreten eines Basisereignisses der Logik widersprechen, kann das Basisereignis als Ursache für ein Systemversagen ausgeschlossen werden /THU 04/.

Da Softwarekomponenten im Unterschied zu technischen Systemen häufig keine klar getrennte Aufgabenverteilung aufweisen und sehr komplexe Abhängigkeiten besitzen, ist die Erstellung des Software-Fehlerbaums komplexer als für Hardware /THU 04/.

- **Software Common Cause Analysis (SCCA)**

  Mit der Software Common Cause Analysis lassen sich einerseits GVA-Auslöser identifizieren, deren Auftrittswahrscheinlichkeiten nicht vernachlässigbar gering sind und die nicht ausreichend abgesichert sind. Andererseits ermöglicht sie auch zu zeigen, dass die Wahrscheinlichkeit für einen Ausfall aufgrund gemeinsamer Ursache „ausreichend gering“ ist, verglichen mit einem Ausfall bei Anforderung. /SAG 07/

- **Dynamic Flowgraph Methodology (DFM)**

  Die DFM ist vor allem geeignet für die Identifikation und Bewertung von Systemversagen, verursacht durch fehlerhaftes oder unvollständiges Softwaredesign bzw. Spezifikationen, die vor allem in Verbindung mit der Hardware auftreten /BNL 10/. Vor allem in der Softwareentwicklungsphase beim Design der Software ist diese Methode einsetzbar, aber auch bei bereits bestehenden Systemen kann sie angewendet werden. Zur Bestimmung von Komponentenausfällen und Softwarezuverlässigkeit ist die Methode nur begrenzt geeignet.

6.2 Quantitative Methoden

Die Zuverlässigkeit einer Software ist ein dynamisches (z.B. von den zeitabhängigen Eingangsdaten abhängiges) Verhaltensmerkmal, das nicht direkt messbar ist. Dennoch existieren diverse Methoden zur quantitativen Abschätzung der Systemzuverlässigkeit. Sie lassen sich, je nach Messgröße von der sie abgeleitet werden, in die folgenden beiden Kategorien einteilen:

- **Metrik-Modelle auf Grundlage von Software-Eigenschaften**

  Diese Modelle nutzen Messgrößen, die sich aus dem Quellcode der Software bestimmen lassen und die vor allem die statischen Eigenschaften der Software beschreiben. Als Indikator für die Zuverlässigkeit werden Metriken definiert, die verschiedene Merkmale des Quellcodes in einen Zahlenwert abbilden. Dabei werden die Metriken so definiert, dass sie die Komplexität der Software in Korrelation zur Fehlerzahl in der Software setzen.

- **Stochastische Modelle auf Grundlage von Ausfalldaten**

  Stochastische Modelle benötigen keine Informationen über die Struktur der einzelnen Komponenten der Software, sondern verwenden Testdaten und Daten aus der Betriebserfahrung, um diese statistisch zu analysieren. Je nach verwendetem Modell werden unterschiedliche Annahmen bei der Beschreibung der Zuverlässigkeit gemacht /BSI 98/.

Oft werden beide Modelle miteinander kombiniert. Eine Zuverlässigkeitsbewertung auf Grundlage von Software-Eigenschaften benötigt zur Validierung der getroffenen Annahmen auch immer die Anwendung stochastischer Modelle. Nur so kann festgestellt werden, wie gut die Metriken wirklich mit der Zuverlässigkeit korrelieren /BSI 98/

Im Folgenden sind beide Kategorien und die entsprechenden Metriken sowie ihre Vor- und Nachteile beschrieben.

6.2.1 Metrik-Modelle auf Grundlage von Softwareeigenschaften

Software-Metriken dienen dazu, unterschiedliche Eigenschaften einer Software in einen Zahlenwert abzubilden. Nach /IEEE 09/ werden Metriken mit dem Ziel definiert, Qualitätsanforderungen an Software allgemeingültig, messbar und überprüfbar zu machen. Dabei wird Qualität definiert als „Grad mit dem eine Software eine verlangte Kombination an Qualitätsmerkmalen erfüllt." Je nach Entwicklungsstufe im Softwarelebenszyklus lassen sich unterschiedliche Qualitätsanforderungen definieren, um

- von Beginn an Qualitätsanforderungen für ein System zu definieren und fortlaufend zu überprüfen,
- Anomalien aufzudecken und auf mögliche Probleme zu verweisen,
- Fehleranfälligkeit bei Systemänderungen zu beschreiben.

Aufgrund der vielfältigen Qualitätsmerkmale im Softwarelebenszyklus gibt es eine große Anzahl von Metriken, die häufig noch weiter kategorisiert werden, beispielsweise in Prozessmetrik, Produktmetrik oder Ressourcenmetrik (siehe /SCH 01/).

Um die Qualität einer Software hinsichtlich möglicher Fehler zu bestimmen, eignen sich vor allem Metriken, die stark mit der Anzahl der Softwarefehler korrelieren, wie beispielsweise Umfangs- und Komplexitätsmetriken, Qualitätsmetriken und Strukturmetriken, die den Quellcode bezüglich der ihm zugrundeliegenden Struktur analysieren /IST 10/, /SCH 01/. Einen Überblick der verschiedenen Software-Metriken liefert Tab. 6.1. Im Folgenden werden beispielhaft einige der genannten Metriken beschrieben.

Tab. 6.1 Übersicht über die Einteilung verschiedener Software-Metriken /SCH 01/

Ebene 1	**Ebene 2**	**Ebene 3**
Prozessmetriken	Life-Cycle-Metrik	Problemdefinitionsmetriken, Entwurfsmetriken, Analyse-/Spezifikationsmetriken
	Maturity-Metriken	Organisationsniveau, Technologieniveau, Datenmanagement, Dokumentationsstandards
Produktmetriken	Umfangsmetriken	Elementeanzahl (Lines of Code, Anzahl der Dokumentationsseiten) Entwicklungsmetriken (Anzahl Testfälle, Ressourcenanteile), Komponentenumfang (Modulzahl)
	Architekturmetriken	Komponentenzahl, Sprach-/Paradigmenzahl, Schichten
	Strukturmetriken	Tiefe, Breite, Kopplung
	Qualitätsmetriken	Effizienzmetriken (Zeitverhalten, Ressourcenverhalten), Wartbarkeitsmetriken (Analysierbarkeit, Testbarkeit, Änderbarkeit, Stabilität), Funktionalitätsmetriken (Eignung, Korrektheit, Sicherheit, Interoperabiliät,Standardgerechtheit), Zuverlässigkeitsmetriken (Fehlerhäufigkeit, Fehlertoleranz, Wiederanlaufmöglichkeit), Handhabbarkeitsmaßnahmen (Lesbarkeit, Lernaufwand, Handhabungsaufwand)

	Komplexitäts-metriken	Algorithmische Komplexitätsmetriken, psychologische Komplexitätsmetriken (Steuerfluss, Datenfluss, Topologie)
Ressourcen-metriken	Personalmetriken	Programmiererfahrung, Produktivität, Kommunikationsniveau, Teamstruktur
	Softwaremetriken	Leistungsmerkmale, Paradigmenabdeckung, Erneuerungsrate

Umfangs- und Komplexitätsmetriken

Mit Zunahme der Komplexität einer Software wächst auch die Wahrscheinlichkeit für Fehler. Komplexitätsmetriken analysieren die Struktur des Programms und liefern Zahlenwerte, die ein Maß für die Fehleranfälligkeit der Software sind.

- **Softwareumfang – Lines of Code (LOC)**

  Der Softwareumfang eines Quellcodes lässt sich durch die Anzahl seiner Zeilen bestimmen. Dabei werden unter dem Begriff „Lines of Code“ auch Kommentar- und Leerzeilen mitberücksichtigt, die nicht vom Programm ausgeführt werden. Für eine zweckmäßigere Bewertung des Softwareumfangs, gibt es auch Umfangsmetriken, die Leerzeichen und Kommentare nicht mitzählen (Source Lines of Code (SLOC)) oder Umfangsmetriken, die die Anzahl der Funktionen und Operatoren bestimmen (Netto Lines of Code (NLOC)).

  Umfangsmetriken haben den Vorteil, dass sie sich leicht bestimmen lassen. Da sie jedoch keine Verzweigungen und Schleifen im Quellcode berücksichtigen und somit die unterschiedlichen Aufrufhäufigkeiten von Anweisungen nicht beachten, sind sie lediglich geeignet eine Aussage hinsichtlich des Softwareumfangs, jedoch nicht hinsichtlich der Komplexität des Kontrollflusses zu machen /SCH 01/.

- **Knotenmetrik**

  Um die Strukturiertheit eines Softwaremoduls zu bestimmen wurde die sogenannte Knotenmetrik eingeführt. Sie bestimmt die Anzahl der „unstrukturierten Sprünge“ innerhalb eines Kontrollflussgraphen. Ein solcher Knoten entsteht, wenn sich Pfade entlang des Kontrollflussgraphen überkreuzen.

In einem Computerprogramm werden Sprunganweisungen verwendet, wenn das Programm nicht mit dem direkt nachfolgenden Befehl fortgesetzt werden soll, sondern an einer anderen Stelle im Programm fortgeführt werden soll.

Je nach Richtung der beiden Sprünge werden unterschiedliche Klassifikationen gemacht. Als Rückwärtssprung wird ein Sprung definiert, der als Zielzeile eine kleinere Nummer hat, als die Nummer der Ausgangszeile (hierzu werden häufig Schleifen verwendet, bei denen Teile des Programms noch einmal durchlaufen werden). Beim Vorwärtssprung ist die Nummer der Zielzeile größer als die Nummer der Ausgangszeile (hierzu kann beispielsweise der Befehl „goto“ verwendet werden). Zur Überschneidung zweier Sprünge kann es durch zwei Rückwärts-, zwei Vorwärts- oder ein Rückwärts- und ein Vorwärtssprung kommen /IST 10/.

Die Knotenmetrik ist stark abhängig von der Linearisierung des gerichteten Graphen und kann daher auf schlecht strukturierten Quellcode und hohe Komplexität hinweisen. Zudem ist die Knotenmetrik ein Maß für die Unabhängigkeit der einzelnen Programmblöcke und die Übersichtlichkeit /IST 10/.

- **Zyklomatische Komplexität nach McCabe**

  Die zyklomatische Komplexität ist ein Maß für die Komplexität eines Softwaremoduls und gibt an, wie viele unterschiedliche Möglichkeiten es gibt das Modul zu durchlaufen. Sie lässt sich aus dem Kontrollflussgraphen eines Moduls bestimmen, wenn die Anzahl der Kanten e, die Anzahl der Knoten n und die Anzahl der Komponenten p bestimmt wird:

$$V(g) = e - n + 2p$$

  wobei die Knoten der Sprungfunktion und die Kanten dem Sprung entsprechen.

  Häufig wird gefordert, dass die zyklomatische Komplexität einen Wert > 10 nicht überschreitet, diese Forderung ist jedoch sehr umstritten und lässt sich leicht vermeiden, wenn das Modul in zwei Module zerlegt wird, wodurch die Makro-Komplexität erhöht wird /LYU 96/. Im Falle von sequentiellen Anweisungen beträgt die zyklomatische Komplexität V(g) = 1 /IST 10/.

- **Wesentliche zyklomatische Komplexität**

  Zur Bestimmung der wesentlichen zyklomatischen Komplexität wird zunächst der Kontrollflussgraph strukturell reduziert, d.h. strukturierte Programmteile mit je einem Ein- und Ausgang werden aus dem Quellcode entfernt, um den Kontrollflussgraphen

zu minimieren. Dann werden die Knoten aus dem neu erzeugten Kontrollflussgraphen bestimmt. Die Knoten sind ein Maß für die Unstrukturiertheit des Quellcodes, die aufgrund kompliziert programmierter Konstrukte entsteht. Die wesentliche zyklomatische Komplexität ergibt sich nach der gleichen Formel, wie die zyklomatische Komplexität, aus dem reduzierten Kontrollflussgraphen /IST 10/. Laut /LDRA 05/ hat ein strukturiertes Programm keine wesentlichen Knoten und somit eine „wesentliche zyklomatische Komplexität“ der Größe eins.

- **Linear Code Sequence and Jump (LCSAJ)**

  Die „Linear Code Sequence and Jump“-Metrik ist ebenfalls ein kontrollflussorientiertes Testverfahren, das den Überdeckungsgrad der Anweisungs- und Zweigabdeckung überprüft. Als LCSAJ wird eine lineare Anweisungsfolge bestehend aus einem oder mehreren aufeinanderfolgenden Basic Blocks bezeichnet, die durch ein Tripel (A,B,C) von Zeilennummern definiert ist. Dabei ist A die Nummer der Startzeile, B die Nummer der Endzeile und C die Nummer der Zielzeile /IST 10/.

  Die Berechnung der LCSAJ ermöglicht die Bestimmung des Überdeckungsgrads, um sicherzustellen, dass es keine Anweisungen im Quellcode gibt, die niemals durchlaufen werden. Dabei ist der Überdeckungsgrad definiert als Quotient aus der Zahl der erfassten Anweisungen bzw. Zweige durch die Gesamtzahl der Anweisungen bzw. Zweige. Darüber hinaus kann die LCSAJ-Dichte berechnet werden, die der Summe der sich überlappenden LCSAJ (d.h. der doppelten Zeilennummern) entspricht und ein Maß für die Komplexität des Programms ist. Bei Änderungen oder Korrekturen innerhalb eines bestimmten Codebereichs kann so die Fehleranfälligkeit bestimmt werden /IST 10/.

- **Halstead-Metrik**

  Bei der Halstead-Metrik wird davon ausgegangen, dass logische Einfachheit die Anzahl von Fehlern reduziert und eine syntaktische Analyse der Sprache des Quellcodes ein Maß für die logische Komplexität liefert. Dazu wird die Anzahl der Operanden und Operatoren aus dem Quellcode bestimmt. Zu den Operanden sind beispielsweise Variablen und Konstanten zu zählen, während Operatoren vor allem logische Operatoren, Vergleichsoperatoren etc. sind.

  N_1 = Gesamtzahl der Operatoren, N_2 = Gesamtzahl der Operanden

  η1 = Gesamtzahl der Arten von Operatoren einer Art, η2 = Gesamtzahl der Arten von Operanden einer Art Daraus ergibt sich:

Programmlänge: $N = N_1 + N_2$

Programmvolumen: $V = N \log_2 (\eta_1 + \eta_2)$

geschätzter Aufwand: $\hat{E} = V \left[\frac{\eta_1 N_2}{2\eta_2}\right]$

Aus diesen Basisgrößen lassen sich weitere Kennzahlen berechnen, wie beispielsweise die Implementierungszeit, Programmniveau, Schwierigkeitsgrad etc.

Da die Halstead-Methode nur die sprachliche Komplexität misst, ist es sinnvoll sie mit der McCabe-Komplexität, die die Komplexität der Programmverzweigung bestimmt, zu kombinieren /SCH 01/, /LYU 96/.

Kombination von Software-Metriken

Angesichts der oben beschriebenen großen Vielzahl an Metriken ist zunächst unklar, welche Auswahl bzw. Kombination von Metriken eine aussagekräftige Zuverlässigkeitsbewertung der Software ermöglicht. Zwar korrelieren die meisten Metriken stark mit der Anzahl der Softwarefehler, aber viele der Metriken korrelieren auch untereinander und haben gemeinsame Elemente die zur Bestimmung der Komplexität verwendet werden können (z.B. Halstead-Metrik, McCabe, LOC). Diese Metriken sind daher Kombinationen dieser gemeinsamen Elemente. Um ein kleineres Set von Variablen zu ermitteln, das alle Informationen vereint, wird die sogenannte Hauptkomponentenanalyse (engl. principal components analysis, PCA) angewandt /LYU 96/. Die Hauptkomponentenanalyse wird im Folgenden kurz beschrieben. Eine eingehende Beschreibung findet sich in Anhang E.

Die Hauptkomponentenanalyse ist eine statistische Methode, die es ermöglicht multivariate Datensätze durch Hauptachsentransformation in ihrer Dimension zu vermindern und vorhandene Redundanzen (in Form von Korrelationen in den Datenpunkten) zusammenzufassen, ohne dass wichtige Informationen verloren gehen. Bei der Hauptkomponentenanalyse wird davon ausgegangen, dass die am stärksten streuenden Daten die meiste Information beinhalten /LYU 96/. Die Hauptkomponentenanalyse ist ein bewährtes Verfahren, das im Bereich des maschinellen Lernens häufig angewendet wird, beispielsweise bei der computergestützten Gesichtserkennung.

Die Hauptkomponentenanalyse kombiniert die oben beschriebenen Metriken auch mit der dynamischen Komplexität und ist ein Beispiel dafür, wie Metriken verwendet werden können, um die Qualität einer Software zu bestimmen. Die Schritte sind im Einzelnen:

1. Reduktion der ausgewählten, in Wechselbeziehung zueinander stehenden Metriken zu einer kleineren Menge an orthogonalen, d.h. unabhängigen Metriken durch Hauptkomponentenanalyse. Weitere Reduktion der Metriken zu einer einzelnen Metrik.

2. Quantifizierung des Ausführungsprofils, das die Wahrscheinlichkeit p_i beschreibt, mit der die Ausführung einer Funktion in Modul m_i stattfindet.

3. Kombination der statischen Komplexitätsmetrik mit der dynamischen Komplexität des Ausführungsprofils.

6.2.2 Stochastische Modelle auf Grundlage von Ausfalldaten

Ziel der quantitativen, stochastischen Modelle zur Zuverlässigkeitsbewertung ist es, Ausfallwahrscheinlichkeiten von softwarebasierten Komponenten oder Systemen abzuschätzen und Ausfallwahrscheinlichkeiten anzugeben. Grundsätzlich müssen hierbei zwei Ausfallarten unterschieden werden:

- Ausfall bei Überwachung / Regelung, z. B. Fehlauslösung
- Ausfall bei Anforderung, d. h. keine Auslösung trotz Bedarf

Es ist möglich, dass hierfür der Einsatz mehrerer Methoden sinnvoll bzw. notwendig ist.

Stochastische Zuverlässigkeitsmodelle modellieren die Mechanismen von Systemausfällen probabilistisch. Dazu benötigen diese Methoden Test- oder Betriebsdaten.

Stochastische Methoden zur Zuverlässigkeitsbewertung lassen sich in mehrere Kategorien unterteilen, je nachdem welche Annahmen und mathematischen Modelle für den Fehlermechanismus gemacht werden (siehe Abb. 6.1).

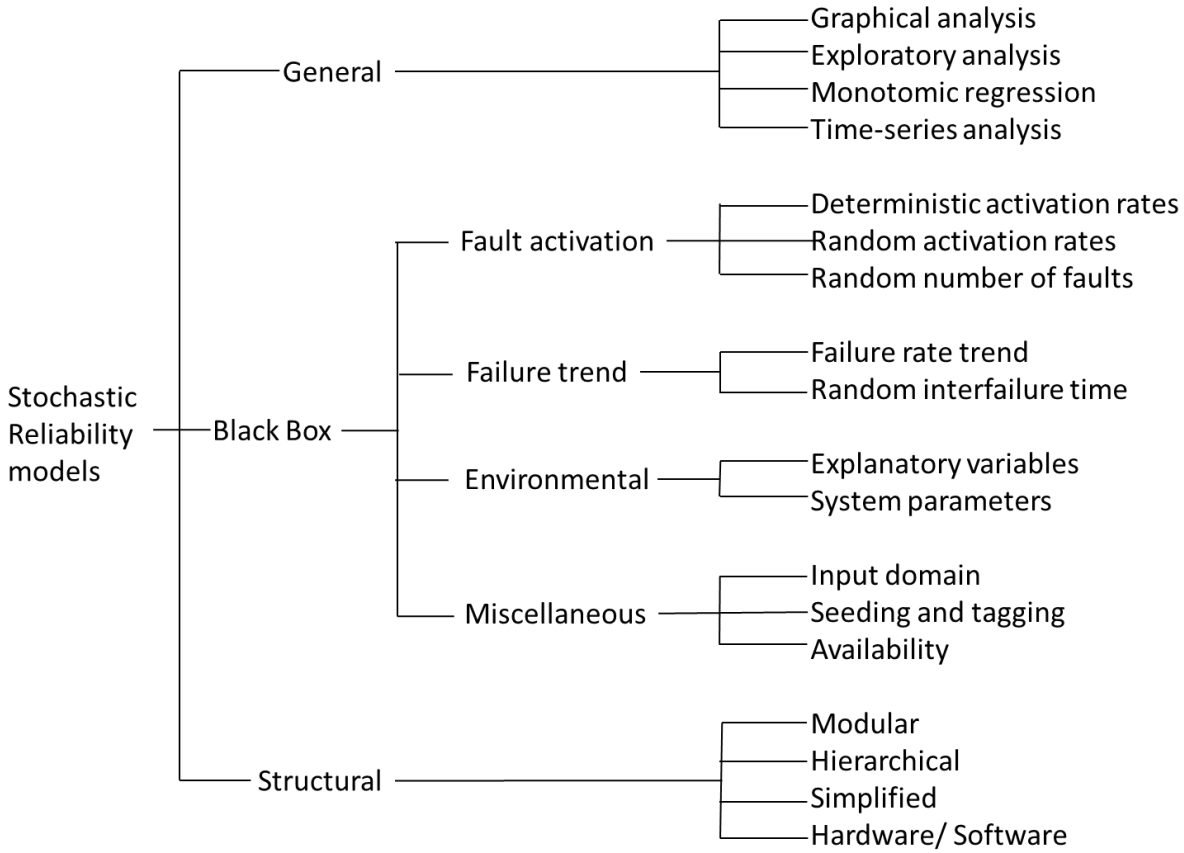

Abb. 6.1 Unterteilung der stochastischen Methoden /BSI 98/

Es sind vier Hauptkategorien zu unterscheiden. Diese sind im Folgenden kurz beschrieben.

Allgemein anwendbare statistische Modelle

Sie setzen keine Annahmen oder Modelle über den Fehlermechanismus voraus. Dennoch eignen sie sich zur Untersuchung von allgemeinen Merkmalen von Testdaten wie beispielsweise zeitliche Trends, Ausreißer, Datencluster etc. In diese Kategorie fallen graphische Analysen, explorative Datenanalyse (EDA), Regressionsanalyse und Zeitreihenanalyse /BSI 98/. Eine detaillierte Beschreibung findet sich in Anhang D.

Parametrische Black-Box-Modelle

Bei diesen Modellen wird die interne Struktur des Systems bei der Analyse nicht berücksichtigt („Black Box“), sondern nur die Fehlerhaftigkeit des Outputs des Systems analysiert. Aus der Ausfallstatistik werden unbekannte Parameter zur Beschreibung der Fehlermechanismen bestimmt. Häufig verwendete Modelle sind Fault-activation-, Failure-trend- und Environmental-factors-Modelle, die unterschiedliche Herangehensweisen bei

der Bestimmung der Zuverlässigkeit haben. Eine detaillierte Beschreibung einiger typischer Black-Box-Modelle findet sich in Anhang D.

6.2.2.1 Strukturelle Modelle

Strukturelle Modelle untersuchen zunächst die Zuverlässigkeit der einzelnen Komponenten, Module etc. und bestimmen dann die Zuverlässigkeit des Gesamtsystems. Zur Bestimmung der Zuverlässigkeit des Gesamtsystems verwenden diese Modelle mathematische Beschreibungen der internen Wechselwirkungen zwischen den Systemkomponenten. Für die Zuverlässigkeitsbewertung der einzelnen Komponenten werden häufig „Black Box"-Modelle verwendet /BSI 98/.

6.2.2.2 Bayessche Netze (Bayesian Belief Networks, BBN)

Bei Bayesschen Netzen handelt es sich um probabilistische, grafische Modelle, die einen Satz von Zufallsvariablen (dargestellt als Knoten) und deren kausale Abhängigkeiten (dargestellt als gerichtete Kanten bzw. Pfeile) in einem gerichteten azyklischen Graphen beschreiben /BNL 10/. Eine detaillierte Beschreibung findet sich in Anhang D.

Bayessche Netze haben den großen Vorteil, dass sie verschiedenartige Informationen verwenden und auch Unsicherheiten der Parameter mit modellieren können. Allerdings stellt die Schätzung der Modellparameter eine Herausforderung dar. Insbesondere bei geringer Datenlage ist die Schätzung der statistischen Anfangsparameter oft mit großen Unsicherheiten behaftet. Dies macht den Nachweis kleiner Ausfallwahrscheinlichkeiten, wie sie bei Systemen mit sicherheitstechnischer Bedeutung gefordert werden, teilweise schwierig. Auch ist es fraglich, ob die Unsicherheiten der zugrundeliegenden Annahmen (Modellunsicherheit und Schätzung) klein genug werden können, um Aussagen über sehr hohe Zuverlässigkeiten zu ermöglichen /BNL 10/. Die Anwendung Bayesscher Netze im Bereich der Zuverlässigkeitsbewertung sicherheitsrelevanter, rechnerbasierte oder programmierbarer Systeme wurde beispielsweise im Rahmen des „Finnish Nuclear Safety Research Programmes" und vom Institut für Sicherheitstechnologie (ISTec) untersucht. Im Folgenden wird ein kurzer Einblick über die Ansätze dieser beiden Zuverlässigkeitsbewertungen STUK 01/, /IST 10/ sowie ihre Vor- und Nachteile gegeben.

6.3 Anwendungsbeispiele

Im Folgenden werden zwei konkrete Anwendungsbeispiele für die Zuverlässigkeitsbewertung vorgestellt, diskutiert und bewertet.

6.3.1 Einsatz von Software-Metriken und stochastischen Modellen für die Zuverlässigkeitsbewertung (ISTec)

Im Rahmen eines Forschungsvorhabens wurde vom Institut für Sicherheitstechnologie (ISTec) ein Konzept zur Vorhersage der Zuverlässigkeit eines softwarebasierten digitalen Leittechniksystems entwickelt. Die Methode, die Messungen der Komplexität mit statistischen Methoden zur Zuverlässigkeitsbewertung kombiniert, wurde speziell für softwarebasierte Leittechnik entwickelt, die anhand graphischer Spezifikationen mittels Code-Generatoren erzeugt wurde /IST 10/.

Für die bottom-up-Analyse wurden zunächst zwei unterschiedliche probabilistische Komplexitätsbewertungen der Funktionsbausteine entwickelt: Die Black-Box-Methode kann angewandt werden, wenn kein Zugriff auf den Programmtext der Funktionsbausteine möglich ist, während die White-Box-Methode detaillierte Kenntnisse des Programms benötigt. Zusätzlich erfolgt eine probabilistische Komplexitätsbewertung der Funktionspläne sowie weitere Komplexitätsmaße für Speicher und Parametrierbarkeit /IST 10/.

Für die Entwicklung des Konzepts wurde beispielhaft das digitale Leittechniksystem TELEPERM XS der Firma Siemens verwendet. Ziel des Projekts war es jedoch prototypisch die Automatisierbarkeit der Methode nachzuweisen und eine generelle Übertragbarkeit auf modular aufgebaute I&C-Systeme zu ermöglichen. Der Vorteil einer solchen automatisierbaren Komplexitätsmessung wäre zum einen die Erkennung bestehender Komplexitätspeaks in der fertigen Software und zum anderen die Identifikation erhöhter Komplexität bereits in der Entwurfsphase, um diese durch geeignete Strukturierung zu reduzieren /IST 10/.

Die Vorgehensweise des Konzepts wird in Anhang F detailliert beschrieben.

Bei der Bewertung des Projekts kommt die ISTec zu dem Schluss, dass die erzielten Ergebnisse dazu beitragen, ein vertieftes, detailliertes Verständnis der Komplexitätseigenschaften und von deren Auswertung zu erlangen. Die von der ISTec entwickelte Methode zur Komplexitätsmessung der Software digitaler Leittechnik bezeichnet sie hinsichtlich der Auswahl und der Definition der Metriken, wie beispielsweise die Verflechtung eines Funktionsplans, als sehr relevant und nützlich für die Beschreibung der Komplexität. Laut ISTec /IST 10/ ermöglicht die von ihnen entwickelte Komplexitätsanalyse das Aufdecken von besonders komplexen und damit fehleranfälligen Funktionsplänen bzw. Funktionsbausteinen. Das wiederum macht das gezielte Prüfen und Testen fehleranfälliger Softwarebestandteile im Hinblick auf deren spezifische Komplexität möglich und verringert so die Fehlerzahl.

Als Kritikpunkt sollte angemerkt werden, dass es keinerlei Möglichkeiten zu einer fundierten Validierung der Methode gibt. Vor allem die zahlreichen Annahmen und Abschätzungen bei der Konversion qualitativer Daten in quantitative Parameter sind mit großen Unsicherheiten behaftet. Die Auswahl der Metriken und der Aufbau des Bayesschen Netzes unterliegen in einem sehr hohen Maß der subjektiven Einschätzungen der Experten.

Der Ansatz der ISTec zur Validierung der Bayesschen-Netz-Analyse zur Zuverlässigkeit besteht in einem Vergleich der Ergebnisse für 21 Funktionspläne mit den Abschätzungen zweier unabhängiger Experten. Dabei zeigt sich eine Übereinstimmung von 93% bzw. 88% zwischen den beiden Experten und den Ergebnissen der Analyse. Die Abweichungen der Experten untereinander liegen dabei in derselben Größenordnung, wobei auffällig ist, dass bei den komplexesten und den am wenigsten komplexen Funktionsplänen die Übereinstimmung zwischen Analyse-Ergebnissen und den Experten am größten ist.

Eine Interpretation dieser Abweichungen ist schwierig, da dies sowohl bedeuten könnte, dass das Modell der Bayesschen Netze verbessert werden müsste, als auch dass die Experten in ihrer Bewertung schlecht abschneiden.

Aus Sicht der GRS ist eine objektive Vorhersage der Zuverlässigkeit eines digitalen Leittechniksystems mit dem von der ISTec entwickelten Verfahren aus den oben genannten Gründen nicht möglich. Auch die ISTec gibt an, dass „die quantitative Bestimmung der Zuverlässigkeit softwarebasierter Systeme anhand eines standardisierten, allgemein akzeptierten Verfahrens nach wie vor ein offenes Problem ist.“

Trotzdem können statistische Modelle, die in der Entwurfsphase zur Identifikation von Komplexitäts-Peaks herangezogen werden, dazu beitragen eine Software besser zu strukturieren und Fehler zu vermeiden.

6.3.2 Einsatz von Bayesschen Netzen zur Zuverlässigkeitsbewertung (STUK)

Im Rahmen der Studie der Finish Nuclear Safety Research Organization (STUK) zur Zuverlässigkeitsbewertung sicherheitskritischer, rechnerbasierter Systeme, wird Bayessche Inferenz unter Verwendung von Bayesschen Netzen verwendet. Ziel der Studie ist es, die Wahrscheinlichkeit für einen Ausfall bei Anforderung zu bestimmen /STUK 01/.

Dabei wird zunächst eine A-priori Wahrscheinlichkeitsverteilung für einen Ausfall bei Anforderung abgeschätzt, die auf quantitativen und qualitativen Charakteristika des Software-Entwicklungsprozess sowie ersten Testergebnissen beruht. Die A-posteriori Verteilung ergibt sich dann aus Werten betrieblicher Tests und der betrieblichen Erfahrung.

Um eine Ausfallwahrscheinlichkeit bei Anforderung in der Größenordnung von 10^{-5} zu erzielen, wie es Vorgabe für die Versagenswahrscheinlichkeit (der finnischen) Reaktorschnellabschaltung ist, wären für ein einziges System und Betriebsprofil mehrere hunderttausend fehlerfreie Tests nötig /STUK 01/, /YVL 97/.

Für ein unabhängiges, zweifach redundantes System reduziert sich die Ausfallwahrscheinlichkeit auf etwa $3x10^{-3}$ Versagensfälle pro Anforderung[5]. Anstelle der 1000 fehlerfreien Tests für ein einzelnes Betriebsprofil, wird die Anzahl der Tests reduziert und die Anzahl fehlerfreier Tests von unterschiedlichen Betriebsprofilen bestimmt. Die Studie zeigt damit, dass die Anwendbarkeit der Bayesschen Netze in der Zuverlässigkeitsbewertung rechnerbasierter Systeme auch auf Bayessche Netze für Systeme mit zwei oder mehreren Betriebsprofilen übertragen werden kann. Diese Untersuchungen sind auch deshalb sinnvoll, da in der betrieblichen Praxis mehrere Betriebsprofile verwendet werden und unentdeckte Fehler nur von Eingangsdaten bestimmter Profile zur Wirkung gebracht werden können. Die unterschiedlichen Betriebsprofile liefern folglich, aufgrund

[5] Die Studie geht nicht darauf ein, wie sich die Unabhängigkeit rechnerbasierter Systeme zeigen lässt.

ihrer unterschiedlichen Merkmale, auch unterschiedliche Ausfallwahrscheinlichkeiten für das gleiche System /STUK 01/.

Neben den betrieblichen Erfahrungswerten liefern auch Systemtests wichtige Erkenntnisse über Ausfallwahrscheinlichkeiten, auch wenn diese häufig mit Betriebsprofilen und unter Rahmenbedingungen durchgeführt werden, die in der betrieblichen Praxis nicht zu erwarten sind /STUK 01/.

Die Vorgehensweise des Konzepts wird in Anhang G detailliert beschrieben.

Die STUK kommt in der Bewertung ihrer Studie zu dem Schluss, dass Bayessche Netze ein flexibles und kompatibles Werkzeug bilden, um die sehr unterschiedlichen Software-Merkmale statistisch miteinander zu vereinen /STUK 01/. Es wird betont, dass sich in den statistischen Annahmen auch die subjektiven Ansichten der Analysten hinsichtlich des System-Verständnisses und der verschiedenen Betriebsumgebungen widerspiegeln und die Modell-Parameter der Bayesschen Netze daher nicht frei von subjektiven Beurteilungen sind. Den Bayesschen Netzen wird jedoch eine große Transparenz zugeschrieben, die es ermöglicht die Modell-Parameter jederzeit zu hinterfragen und gegebenenfalls zu verbessern /STUK 01/.

Zudem wird dem Modell zugutegehalten, dass die Subjektivität in der Bewertung der Zuverlässigkeit auch dadurch minimiert wird, dass der Einfluss qualitativer Merkmale durch die Zuhilfenahme einer Vielzahl von Daten reduziert wird /STUK 01/.

Die für die Zuverlässigkeitsbewertung sicherheitskritischer, rechnerbasierter Systeme erforderliche hohe Zuverlässigkeit bei vergleichsweise geringer statistischer Datenlage beschreibt die Studie als ein „von keiner statistischen Methode zu lösendes Dilemma“. Als beste Möglichkeit diesen Mangel an statistischen Daten zu kompensieren, wird die Kombination aller statistischen Merkmale erachtet. Darunter fallen Merkmale ähnlicher Systeme sowie qualitative Charakteristiken. Laut STUK bestätigen die Ergebnisse der Simulation, dass Bayessche Netzwerke unter Verwendung der Bayesschen Inferenz eine effiziente und konsistente Möglichkeit für die komplexe Zuverlässigkeitsbewertung sind /STUK 01/.

Damit greift die STUK in ihrer Bewertung zwei zentrale Fragenstellungen auf, die im Zusammenhang mit den Modellen zur Bewertung der Softwarezuverlässigkeit vielfach diskutiert werden:

1. **Abschätzung der Modellparameter:** Insbesondere bei geringer Datenlage ist die Konversion qualitativer Daten in quantitative Parameter oft mit großen Unsicherheiten behaftet. Dies macht den Nachweis kleiner Ausfallwahrscheinlichkeiten, wie sie bei sicherheitskritischen Systemen gefordert werden, teilweise schwierig /BNL 10/. Daher ist damit zu rechnen, dass alle statistischen Modelle ein hohes Maß an subjektiver Beurteilung durch den Analysten aufweisen.

2. **Statistische Datenlage:** Verglichen mit den hohen Anforderungen an die Zuverlässigkeit ist die Menge (Größe des Zustandsraums) an statistischen Daten aus Systemtests, betrieblichen Tests etc. sehr gering. Daher ist fraglich, ob die Unsicherheiten der zugrundeliegenden Annahmen klein genug werden können, um Aussagen über sehr hohe Zuverlässigkeitswerte zu ermöglichen /BNL 10/.

Diese beiden Punkte stellen ein Dilemma für alle statistischen Methoden dar. Sie machen eine Abschätzung der Validität eines statistischen Modells extrem schwierig und stellen somit die Anwendbarkeit aller statistischen Methoden zur quantitativen Zuverlässigkeitsbewertung in Frage.

6.4 Diskussion der Zuverlässigkeitsbewertung

Nach der Einführung der grundlegenden Methoden der Zuverlässigkeitsbewertung von Systemen werden im Rahmen dieses Abschnitts zunächst die konkreten Anforderungen an die Zuverlässigkeitsbewertung eines Systems beziehungsweise von Software zusammengefasst. Diese Anforderungen werden üblicherweise nicht für qualitative und quantitative Bewertungen unterschieden. Außerdem werden Empfehlungen zu konkreten Methoden und deren Anwendung zusammengefasst.

Anschließend wird eine Übertragbarkeit der Anforderungen sowie Empfehlungen zu den Methoden und Vorgehensweisen diskutiert.

6.4.1 Nicht-nuklear

Anforderungen an die Zuverlässigkeitsbewertung des Systems

In den betrachteten nicht-nuklearen Industriezweigen werden konkrete Anforderungen an die Durchführung einer Zuverlässigkeitsbewertung von Systemen formuliert. Die Anforderungen werden hauptsächlich in den Standards /IEEE 08/, /BSI 98/ und /ECSS 80/ festgelegt. Die drei wesentlichen Punkte sind im Folgenden kurz erläutert.

Als konkrete Anforderung an eine Zuverlässigkeitsbewertung fordert /BSI 98/ eine genaue Definition der Versagenskriterien, da die Bewertung der Zuverlässigkeit signifikant von dieser abhängt. /ECSS 80/ fordert weiterhin eine Definition des Grades an Toleranz der Software gegen Fehlfunktionen, eine Festlegung der Detektion, Isolation und Diagnose von Software-Versagen sowie die Aufstellung von Anforderungen zur Verhinderung einer Fehlerausbreitung.

/BSI 98/ fordert weiterhin, dass bei der Bewertung der Zuverlässigkeit grundsätzlich alle Komponenten, d.h. Software, Elektronik, Mechanik und Bediener, einbezogen werden. Denn nur die Bewertung aller Einzelkomponenten sowie deren Gesamtkonfiguration lässt eine repräsentative Zuverlässigkeitsbewertung des Gesamtsystems zu.

Ein weiterer wesentlicher Aspekt für die Durchführung einer belastbaren Zuverlässigkeitsbewertung sind die der Analyse zu Grunde gelegten Daten. An diese werden in /IEEE 08/ konkrete Anforderungen, wie Fehlerfreiheit, Genauigkeit und Sachdienlichkeit gestellt. Auch /BSI 98/ stellt konkrete Anforderungen an die Daten, die für die unterschiedlichen Modelle benötigt werden, sowie an die Methoden zum Sammeln der Daten:

- Sammeln der Daten stets mit einer festen Absicht und dem Wissen über die Methode, mit der sie analysiert werden
- Sicherstellung der Vollständigkeit der Daten
- Konsistente Definition der Daten
- Eingabe der Daten in eine Datenbank
- Regelmäßige Rücksprache mit dem Anbieter der Daten

Darüber hinaus wird empfohlen, die Daten, sofern möglich, automatisch zu erfasse. Auch an die Datenspeicherungen werden in /BSI 98/ einige Anforderungen formuliert.

Außerdem sollen nur die Daten erfasst werden, die relevant sind, und die Erfassung soll möglichst einfach gestaltet sein.

Anforderungen an die Zuverlässigkeitsbewertung der Software

Auch an eine Zuverlässigkeitsbewertung von Software werden konkrete Anforderungen gestellt. Die Anforderungen werden hauptsächlich in den Standards /DIN 50/, /BSI 98/ und /ECSS 80/ formuliert. Die wesentlichen Anforderungen sind im Folgenden kurz aufgeführt.

Eine Zuverlässigkeitsbewertung von Software muss nachweisen, dass die folgenden wesentlichen Anforderungen an die Software, die eine hohe Qualität und Zuverlässigkeit garantieren sollen, erfüllt sind. Solche Anforderungen formuliert zum Beispiel /ECSS 80/. Konkret sind dies die

- Übertragbarkeit,
- Wartbarkeit,
- Wiederverwendbarkeit und
- Korrektheit.

der Software. Außerdem präsentiert /ECSS 80/, wie weitere Anforderungen an eine Software aufzustellen sind, um eine hohe Systemzuverlässigkeit zu garantieren:

- Die Anforderungen sollen dokumentiert werden.
- Sie sollen vollständig und eindeutig sein.
- Sie sollen hinreichend präzise sein, um eine Verifizierung und Validierung zu ermöglichen.
- Für jede Anforderung soll die Methode der Verifizierung festgelegt werden.
- Jede Anforderung soll eine eindeutige Kennung besitzen.

/BSI 98/ erläutert darüber hinaus, dass die Software-Zuverlässigkeit signifikant von der äußeren Umgebung abhängig ist und folglich bei der Bewertung der Softwarezuverlässigkeit die Betriebsumgebung einbezogen werden muss.

Auch die Einbeziehung der konkreten Version der Software ist unerlässlich. Die Zuverlässigkeitsbewertung des Systems gilt jeweils nur für eine konkrete Version. Für die Bewertung kleiner Veränderungen empfiehlt /BSI 98/ die Definition einer Basis-Version und die Verwendung von Reliability Growth Methoden zur Bewertung der Zuverlässigkeit.

Laut /ECSS 80/ ist die Bewertung der Zuverlässigkeit von Software jedoch generell skeptisch zu bewerten. Auch laut /DIN 50/ ist der aktuelle Stand der Technik derart, dass weder die Anwendung von Qualitätssicherungsverfahren (fehlervermeidende und fehlererkennende Maßnahmen) noch die Anwendung fehlertoleranter Software-Verfahren die absolute Sicherheit der Software garantieren können. Es ist nach /DIN 50/ aktuell kein Weg bekannt, die Fehlerfreiheit bzw. eine ausreichende Zuverlässigkeit in einer vergleichsweise komplexen sicherheitsrelevanten Software zu beweisen. Dies gilt insbesondere für Spezifikations- und Entwurfsfehler.

Empfehlungen zu Methoden und Anwendung

Weitere Anforderungen, die an die Bewertung der Zuverlässigkeit gestellt werden, betreffen die Methoden der Analyse. Die entsprechenden Anforderungen werden hauptsächlich in den Standards /IEEE 08/, /IEEE 03/, /ECSS 80/ und /BSI 98/ formuliert.

So merkt /IEEE 08/ an, dass eine Übertragbarkeit der Analyse und des Modells nicht möglich ist, sobald eine Veränderung an den Versagenskriterien, dem Code oder der Computerumgebung durchgeführt wurde.

In /ECSS 80/ wird konkret angemerkt, dass eine Übertragbarkeit der quantitativen und qualitativen Modelle der Bewertung von Hardware auf Software schwer möglich ist. Dies ist unter anderem darin begründet, dass Softwarestrukturen oft deutlich komplexer sind und es sich bei Softwarefehlern um Fehler überwiegend systematischer Natur handelt. Darüber hinaus verweist /IEEE 03/ jedoch darauf, dass für eine quantitative Bewertung von Software sämtliche Hardware-Komponenten einbezogen werden sollen, da die Software-Zuverlässigkeit von der Umgebung abhängt. Somit wird deutlich, dass die Verwendung unterschiedlicher Methoden zur Bewertung der Zuverlässigkeit von Software und Hardware problematisch ist. Diese beiden Aspekte stehen in einem direkten Widerspruch und verlangen nach der Entwicklung geeigneter Methoden zur Zuverlässigkeitsbewertung von Software und vom Gesamtsystem.

/IEEE 08/ fordert, dass bei einer zeitabhängigen Bewertung der Zuverlässigkeit von Software eine Reduzierung der benötigten Zeit durch eine mehrfache Auslegung des Systems nur bei einer vollständigen Diversität des Systems möglich ist. /BSI 98/ bezweifelt die Möglichkeit der Erhöhung von Software-Zuverlässigkeit auf Basis der Betriebsdauer komplett. Dies wird auch in /REES 94/ diskutiert. Es wird folgendes Argument gegen die Verwendung von zeitabhängigen Methoden für die Bewertung der Zuverlässigkeit von Software identifiziert: „Ein Softwarefehler wird dann das erste Mal auftreten, wenn die entsprechende Sequenz des Programmes das erste Mal durchlaufen wird. Der Zeitfaktor ist somit kein statistischer Effekt und somit für die Bewertung der Zuverlässigkeit irrelevant."

Für Modelle konkret zur Bewertung der Zuverlässigkeit von Software fordert /BSI 98/, dass das gewählte Modell an den zu untersuchenden Datensatz angepasst ist. Auch fordert der Standard eine Optimierung der Zuverlässigkeitsanalyse durch die Anwendung mehrere Modelle und Methoden, da kein Modell durchgehend korrekte Annahmen machen kann. Letztendlich empfiehlt /BSI 98/ die Bevorzugung von quantitativen Methoden und Metriken. Diese Meinung spiegelt sich auch in Diskussionen über die quantitative Bewertung der Zuverlässigkeit von Software wieder.

/BSI 98/ gibt darüber hinaus einige Empfehlungen im Umgang mit konkreten Methoden. Neben der Anforderung, dass bei der Auswahl der verwendeten Methode den jeweiligen gesetzlichen Bestimmungen sowie zwingenden Auflagen zu folgen ist, stellt er die folgenden Anforderungen an die Methode:

- Exakte Vorhersage
- Nützlichkeit
- Qualität der Annahmen
- Anwendbarkeit bzw. Eignung
- Einfachheit

Schlussendlich fordert der Standard /IEEE 10/ die Zuverlässigkeitsbewertung anhand einer detaillierten Richtlinie, die in einer vollständigen und konsistenten Dokumentation resultiert. Die Richtlinie ermöglicht somit die Bedingung zu erfüllen, dass auch Außenstehende in der Lage sind die Analyse nachzuvollziehen, zu vergleichen und zu bewer-

ten. Auch /IEEE 08/ liefert ein konkretes 13-Schritte-Programm zur Bewertung der Zuverlässigkeit. Konkrete Anweisungen welche Methoden verwendet werden können, enthalten diese Richtlinien jedoch nicht.

6.4.2 Übertragbarkeit der Methoden und Anforderungen auf die Kerntechnik

Im Folgenden wird eine mögliche Übertragbarkeit der in den vorhergehenden Abschnitten beschriebenen Anforderungen an eine Zuverlässigkeitsbewertung sowie an die Methoden aus den nicht-nuklearen Industriezweigen auf die Kerntechnik diskutiert.

Anforderungen an die Zuverlässigkeitsbewertung des Systems

Auch in der Kerntechnik werden konkrete Anforderungen an die Entwicklung des Systems sowie an den vollständigen Lebenszyklus gestellt. Auch eine genaue Spezifikation von Ausfallursachen und eine Auslegung des Systems gegen diese werden in der Kerntechnik konkret gefordert (siehe auch Kapitel 4.1). Eine Übertragung der Anforderungen an eine Zuverlässigkeitsbewertung des Gesamtsystems auf die Kerntechnik ist somit nicht notwendig.

Die Anforderungen an die für die Methoden notwendigen Daten unterscheiden sich jedoch im nicht-nuklearen Bereich deutlich von den kerntechnischen Daten. Während im nicht-nuklearen Bereich konkrete Anforderungen an die Art der Daten, die Erfassung und auch die Speicherung der Daten gestellt werden, finden sich entsprechende Anforderungen im nuklearen Bereich nicht. Eine Übertragung der Anforderungen ist jedoch nur teilweise möglich. So können zum Beispiel die Anforderungen an die Speicherung der Daten und einige generelle Aspekte, wie eine wohlüberlegte Auswahl der Daten übernommen werden. Aspekte wie die Forderung nach der Sicherstellung der Vollständigkeit der Daten sind jedoch in der Kerntechnik kaum umsetzbar, da eine Erfassung sämtlicher Betriebszustände nicht möglich ist.

Anforderungen an die Zuverlässigkeitsbewertung der Software

Zur Erhöhung der Zuverlässigkeit eines Systems fordert /DIN 07/, dass die Selbstüberwachung des Systems Zufallsausfälle der Hardware, fehlerhaftes Verhalten der Software oder eine fehlerhafte Datenübertragung in praktikablem Ausmaß entdeckt. Die Software muss auf den entdeckten Fehler in geeigneter Weise zeitgerecht reagieren.

Die Reaktionsmaßnahmen müssen in der Spezifikation festgelegt sein. Darüber hinaus muss die Software laut /DIN 07/ regelmäßigen Prüfungen unterzogen werden. Hierbei wird gefordert, dass jede Sicherheitsfunktion regelmäßig geprüft werden muss und jedes Versagen einer Sicherheitsfunktion aufgedeckt werden muss. Darüber hinaus wird im Bereich der Kerntechnik in diversen Standards die Anwendung von Diversität zur Erhöhung der Zuverlässigkeit eines Systems und zur Verringerung des Potentials von Fehlern gemeinsamer Ursache empfohlen /DIN 07/, /DIN 09/, /IAEA 02/, /DIN 10b/.

Weiterhin fordert /DIN 07/ konkret die Verwendung geeigneter Software-Werkzeuge um das Risiko des Einbringens von Fehlern in den Entwicklungsprozess zu verringern. Hierzu wird jedoch ergänzend gefordert, dass diese Werkzeuge angemessen geprüft und verifiziert sein müssen und selbst eine ausreichende Qualität und Zuverlässigkeit aufweisen. Außerdem müssen laut /DIN 10a/ die Werkzeuge, die die Zuverlässigkeit der Software beeinflussen können, in der Qualitätssicherung festgelegt sein.

Auch die Verwendung bereits existierender Software und Systemkomponenten können für die Qualität und Zuverlässigkeit des Systems relevant sein. Insbesondere die Wiederverwendung validierter vorgefertigter Software kann laut /DIN 07/ das Vertrauen in die Zuverlässigkeit des Systems erhöhen. /DIN 10a/ fordert den Nachweis der Zuverlässigkeit und der funktionalen Eignung vorgefertigter Software, sowie die Ausrichtung des Konfigurationsmanagements auf die Gesichtspunkte der Softwarezuverlässigkeit. Außerdem wird eine anwendbare und vertrauenswürdige Betriebserfahrung gefordert.

Prinzipiell ist eine Übertragbarkeit der Anforderungen für eine zuverlässige Software denkbar. Obwohl in der Kerntechnik nur wenige konkrete Anforderungen an die Zuverlässigkeit von Software im Rahmen des Regelwerks und der Standards und Normen der Kerntechnik gestellt werden, gehen die Anforderungen aus dem nicht-nuklearen Bereich jedoch nicht signifikant über die des nuklearen Bereichs hinaus. Eine Übertragung ist somit hinfällig.

Wie entsprechende Anforderungen im Bereich der rechnerbasierten und programmierbaren Leittechnik konkret umgesetzt und erfüllt werden können, kann selbst den Regelwerken und Standards der nicht-nuklearen Bereichen nicht entnommen werden. Dementsprechend ist eine Übertragung auf den nuklearen Bereich nicht möglich.

Empfehlungen zu Methoden und Anwendung

Bereits im nicht-nuklearen Bereich ist es umstritten, ob die für konventionelle leittechnische Systeme eingesetzten Methoden zur Bewertung der Zuverlässigkeit auf Systeme mit rechnerbasierten oder programmierbaren Komponenten übertragen werden können.

Die Regelwerke und Normen aus dem Bereich der Kerntechnik stehen den verschiedenen Methoden der Zuverlässigkeitsbewertung ebenfalls sehr skeptisch gegenüber. /IAEA 00/ macht darauf aufmerksam, dass eine quantitative Einschätzung eines rechnerbasierten oder programmierbaren Systems heutzutage nicht vollständig möglich ist. Es kann lediglich auf qualitative Methoden zurückgegriffen werden. Der Standard selbst weist keine Lösungsvorschläge für diese Problematik auf.

/IAEA 94/ gibt zu bedenken, dass eine quantitative Zuverlässigkeitsangabe (wie z.B. Ausfallwahrscheinlichkeit 10^{-6}) nicht praktiikabel ist. Dieses Problem lässt sich nur über Redundanz der Systeme lösen. Hierbei setzt jedoch die Problematik der Diversität von Software Grenzen für die Zuverlässigkeit, da man bei Software davon ausgehen muss, dass Fehler systematischer Natur sind und keine Zufallsfehler. Somit lässt sich festhalten, dass der Einsatz der Vielzahl von Quantifizierungsmethoden, die im Laufe der Zeit für Software entwickelt wurden, stets fraglich bleibt. Dennoch stellt der Bericht einige der gängigen Methoden vor und weist darauf hin, dass eine Kombination verschiedener deterministischer und probabilistischer Methoden das bestmögliche Resultat erbringt.

Auch /NUR 08/ weist darauf hin, dass es zurzeit keine Vorgaben dazu gibt, wie eine ausreichende Diversität von computerbasierten Systemen zu spezifizieren ist. Ab wann eine Software zuverlässig als diversitär bezeichnet werden kann, ist häufig strittig. Darüber hinaus erläutert /DIN 07/, dass selbst wenn eine diversitäre Realisierung möglich ist, die durch Diversität erreichte Steigerung der Zuverlässigkeit nicht quantifiziert werden kann. Es ist eine Beurteilung erforderlich, die von der durch Software erreichbaren qualitativen Zuverlässigkeit ausgeht.

6.4.3 Fazit

Bezüglich der Anforderungen an eine zuverlässige Software sowie an das Gesamtsystem ist durchaus ein Vergleich des nuklearen und des nicht-nuklearen Bereichs möglich. In beiden Fällen werden konkrete Anforderungen an die Entwicklung der Software sowie

an den vollständigen Lebenszyklus gestellt. Auch eine genaue Spezifikation von Ausfallursachen und eine Auslegung des Systems gegen diese werden in beiden Bereichen konkret gefordert. Wie entsprechende Anforderungen im Bereich der rechnerbasierten und programmierbaren Leittechnik umgesetzt und konkret erfüllt werden können, kann auch den internationalen oder nicht-nuklearen Regelwerken und Standards nicht entnommen werden. Eine Übertragung von Anforderungen ist prinzipiell denkbar, zum aktuellen Zeitpunkt jedoch nicht notwendig, da die Anforderungen aus dem nuklearen Bereich denen aus dem nicht-nuklearen Bereich in keinerlei Hinsicht nachstehen.

Lediglich Anforderungen an die der Analyse zugrunde gelegten Daten werden in der Kerntechnik nicht formuliert. Hier ist es durchaus sinnvoll die Empfehlungen und Anforderungen aus /IEEE 08/ und /BSI 98/ teileweise, soweit möglich, zu übernehmen.

Zur Übertragbarkeit der Empfehlungen zu Methoden und deren Anwendung lässt sich zusammenfassend feststellen, dass eine Übertragbarkeit vor allem der quantitativen Methoden zum aktuellen Zeitpunkt kaum denkbar ist. Die Regelwerke und Standards des kerntechnischen Bereichs stehen aus diversen Gründen einer belastbaren Zuverlässigkeitsbewertung skeptisch gegenüber (/ECSS 80/ und /DIN 50/)

Der Versuch einer Übertragung einschlägiger Methoden zur Zuverlässigkeitsbewertung aus dem nicht-nuklearen Bereich ist zum aktuellen Zeitpunkt schwer möglich und würde die Skepsis gegenüber der Zuverlässigkeit programmierbarer oder softwarebasierter Systeme nicht auflösen. Insbesondere ein belastbarer Nachweis von Diversität kann nur schwer geführt werden.

Da der Bewertung der Zuverlässigkeit eines rechnerbasierten oder programmierbaren Systems in der Kerntechnik sehr skeptisch gegenüber gestanden wird, gibt /IAEA 02/ konkret die Empfehlung, dass für den Fall, dass der notwendige Beweis der Zuverlässigkeit eines Systems nicht erbracht werden kann, zusätzlich konventionelle, analoge Systeme zum Einsatz kommen sollen.

7 Fazit

Im Rahmen dieses Kapitels werden nun abschließend die wesentlichen Ergebnisse des Projekts noch einmal kurz zusammengefasst.

Vergleich und Übertragbarkeit des System-Sicherheitslebenszyklusses

Der System-Sicherheitslebenszyklus unterscheidet sich in den verschiedenen Industriezweigen nur geringfügig. Die Automobilindustrie unterscheidet sich jedoch dahingehend, dass der gesamte Lebenszyklus auf die Serienproduktion angepasst wurde.

Die Kerntechnik unterscheidet sich von den anderen Industriezweigen dadurch, dass der Lebenszyklus nach DIN EN 61513 erst später einsetzt, da die Bestimmung der Auslegungsgrundlagen nicht als Phase im Lebenszyklus der Software vorgesehen ist, sondern als Vorarbeit in die erste Phase des Lebenszyklus eingeht. Sie beginnt ihren System-Sicherheitslebenszyklus erst mit der Bestimmung und Zuordnung der Sicherheitsanforderungen des Systems. Die Phasen zur Konzeptentwicklung sowie der Gefährdungs- und Risikoanalyse werden nicht konkret aufgegriffen und DIN EN 61513 behandelt darüber hinaus die Phasen der Stilllegung und Entsorgung der Software nicht. Neue Erkenntnisse lassen sich aus diesen Unterschieden jedoch nicht ableiten, da die DIN EN 61513 die Erfüllung und Umsetzung der allgemeinen Norm DIN EN 61508 fordert und somit die entsprechenden Phasen des System-Sicherheitslebenszyklus sowie die entsprechenden Anforderungen an diesen trotzdem indirekt bereits enthält.

Eine Festlegung und Berücksichtigung der Fragestellung, ob es sich bei der Entwicklung um eine Neuentwicklung oder eine Modifikation der Software handelt wird lediglich in ISO 26262 als früher Bestandteil des System-Sicherheitslebenszyklus betrachtet. Dieser Aspekt findet sich in keinem anderen Industriezweig und auch nicht in der allgemeinen Norm oder der Kerntechnik. Eine frühe Betrachtung dieser Fragestellung ist jedoch sinnvoll und kann durchaus auf die Kerntechnik übertragen werden.

Vergleich und Übertragbarkeit des Software-Sicherheitslebenszyklus

Aktuell wird sowohl in der allgemeinen Norm DIN EN 61508-3, als auch in der Kerntechnik das V-Modell als Vorgehensmodell empfohlen. Laut Konferenzvortrag /MYK 15/ wird die Norm DIN EN 61508-3 jedoch so überarbeitet werden, dass als mögliche Methoden sowohl das V-Modell als auch das Scrum-Modell normenkonform verwendet werden

können. Da in dem Standard DIN EN 60880 angegeben wird, dass er mit DIN EN 61513 und den darin verwiesen Standards (u.a. DIN EN 61508) ein konsistentes Set an Dokumenten darstellt, ist die Kerntechnik von dieser Änderung ebenfalls betroffen.

Darüber hinaus ergeben sich keine neuen Erkenntnisse für die Kerntechnik aus der Betrachtung des Software-Lebenszyklus der anderen Industriezweige. Aktuell besteht keine Notwendigkeit aus anderen Industriezweigen Aspekte zum Thema Software-Sicherheitslebenszyklus auf die Kerntechnik zu übertragen.

Vergleich und Übertragbarkeit der Anforderungen an die Software

Es hat sich gezeigt, dass in einigen Themenbereichen durchaus eine Übertragung von konkreten Anforderungen aus anderen Industriezweigen sinnvoll ist. Konkret sind Anforderungen aus den Bereichen Personal und Management, Codierstandards, Dokumentation, Schnittstellen und Hardware, Werkzeuge, Handling und Speicherung sowie Begutachtung zu nennen. Darüber hinaus können Empfehlungen zum Aufbau und zur Strukturierung des Regelwerks übernommen werden.

In den Bereichen Selbstüberwachung und Diversität gehen die Anforderungen aus dem Kerntechnischen Regelwerk über die der anderen Bereiche hinaus. Eine Übertragung ist somit hinfällig.

Vergleich und Übertragbarkeit der Anforderungen an die Qualifizierung

Vergleicht man die an die Qualifizierung gestellten Anforderungen mit denen aus nationalen und internationalen Normen und Standards aus dem Bereich der Kerntechnik, so wird deutlich, dass bereits detaillierte Anforderungen enthalten sind. Andere Industriezweige sind bei diesem Thema größtenteils weniger detailliert als die vorhandenen Anforderungen in der Kerntechnik. Deswegen besteht keine Notwendigkeit Anforderungen zur Qualifizierung aus nicht-nuklearen Bereichen zu übertragen.

Für den Aspekt der Dokumentation und Berichterstattung gehen die Vorgaben und Empfehlungen insbesondere im Schienenverkehr jedoch über die Kerntechnik hinaus. Es ist durchaus sinnvoll entsprechend detaillierte Vorgaben an die Dokumentation und Berichterstattung auch für die Kerntechnik aufzustellen. Eine Übertragbarkeit aus dem Schienenverkehr ist hierbei möglich.

Vergleich und Übertragbarkeit der Anforderungen an die Zuverlässigkeitsbewertung

Eine Übertragung der Anforderungen an die Zuverlässigkeitsbewertung eines Systems auf die Kerntechnik ist in weiten Teilen nicht notwendig, da die Anforderungen in der Kerntechnik denen aus anderen Bereichen nicht nachstehen. Insbesondere zu Anforderungen bei der Erfassung und Speicherung von Daten ist jedoch eine Übertragung der entsprechenden Anforderungen, soweit möglich, sinnvoll.

Bei der Zuverlässigkeitsbewertung der Software gehen die Anforderungen aus dem nicht-nuklearen Bereich nicht signifikant über die des nuklearen Bereichs hinaus. Wie entsprechende Anforderungen im Bereich der rechnerbasierten und programmierbaren Leittechnik konkret umgesetzt und erfüllt werden können, kann selbst den Regelwerken und Standards der nicht-nuklearen Bereichen nicht entnommen werden. Dementsprechend ist eine Übertragung auf die Kerntechnik nicht möglich. Dennoch ist festzuhalten, dass eine entsprechende Ergänzung im kerntechnischen Regelwerk sinnvoll ist.

Vergleich und Übertragbarkeit der Methoden zur Zuverlässigkeitsbewertung

Es hat sich gezeigt, dass im Schienenverkehr und in der Automobilindustrie konkrete Vorgehen und Empfehlungen für Methoden der Qualifizierung gegeben werden. So wird zum Beispiel in der Automobilindustrie ein angemessener Satz der folgenden Methoden

- Wiederholbare Tests mit spezifizierten Testprozeduren, Testfällen und Erfüllungs-/Ausfallkriterien
- Analysen wie z. B. FMEA, Fehlerbaumanalyse, Ereignisbaumanalyse, Simulation
- Langzeittests
- Anwendertests unter realen Bedingungen
- Reviews

empfohlen. In der Kerntechnik werden lediglich statische und dynamische Tests im Validierungsplan gefordert. Konkretere Anforderungen an die Methoden werden nicht gestellt. Dementsprechend ist eine Übertragung der Anforderungen an die durchzuführenden Methoden sinnvoll.

Darüber hinaus ist darauf hinzuweisen, dass es keine universell geeignete, konkrete Methode zur Zuverlässigkeitsbewertung softwarebasierter Systeme gibt. Die Vorgehensweise für eine Zuverlässigkeitsbewertung hängt entscheidend von der verwendeten Software ab. Auch wird die Kombination mehrerer Methoden empfohlen.

Außerdem ist zur Übertragbarkeit der Empfehlungen zu Methoden und Anwendung zusammenfassend festzustellen, dass die Regelwerke und Standards des kerntechnischen Bereichs aus diversen Gründen einer belastbaren Zuverlässigkeitsbewertung skeptisch gegenüber stehen. Der Versuch einer Übertragung einschlägiger Methoden zur Zuverlässigkeitsbewertung aus dem nicht-nuklearen Bereich ist deshalb schwer möglich. In der Kerntechnik wird für den Fall, dass der notwendige Beweis der Zuverlässigkeit eines Systems nicht erbracht werden kann gefordert, dass, zusätzlich konventionelle, analoge Systeme zum Einsatz kommen.

Literaturverzeichnis

/AOP 52/ NATO AOP-52 „Guidance on Software Safety Design and Assessment of Munition-Related Computing Systems", März 2009

/BLE 14/ Bless, M.: Scrum und die IEC 62304, Medizinische Software mit agilen Methoden normenkonform entwickeln, Version 1.8, Januar 2014. Erreichbar unter http://agilecoach.de/wp-content/uploads/2013/07/Marc-Bless-Scrum-und-die-IEC-62304-eBook-v1.8.pdf, zitiert am 12.02.2016.

/BMU 12/ Bundesministerium für Umwelt, Naturschutz und Reaktorsicherheit: Sicherheitsanforderungen an Kernkraftwerke, November 2012.

/BMU 13/ Bundesministerium für Umwelt, Naturschutz und Reaktorsicherheit: Interpretationen zu den Sicherheitsanforderungen an Kernkraftwerke, November 2013.

/BNL 10/ Chu, T.-L., Yue, M., Martinez-Guridi, G., Lehner, J.: Review of quantitative software reliability methods, Brookhaven National Laboratory, Letter Report, Digital System Software PRA, JCN N-6725, September 2010.

/BSI 98/ British Standards Institution: Reliability of systems, equipment and components, BS 5760, Part 8, 1998.

/DEF 42/ Ministry of Defence: Reliability and Maintainability Assurance Guides, Defence Standard 00-42, Part 2, 01.09.1997.

/DEF 55/ Ministry of Defence: Requirements for safety related software in defence equipment. 01.08.1997

/DEF 95/ Ministry of Defence: System Requirements for the Design, Development, Supply and Maintenance of Software, 05-95/Issue 3, 23.06.1995

/DIN 03/ DIN EN 50129: Bahnanwendungen – Telekommunikationstechnik, Signaltechnik und Datenverarbeitungssysteme – Sicherheitsrelevante elektronische Systeme für Signaltechnik. Dezember 2003.

/DIN 07/ DIN EN 60880: Kernkraftwerke – Leittechnik für Systeme mit sicherheitstechnischer Bedeutung Softwareaspekte für rechnerbasierte Systeme zur Realisierung von Funktionen der Kategorie A, 2007.

/DIN 09/ DIN EN 61226: Kernkraftwerke – Leittechnische Systeme mit sicherheitstechnischer Bedeutung – Kategorisierung leittechnischer Funktionen, 2010.

/DIN 10a/ DIN EN 62138: Kernkraftwerke – Leittechnik für Systeme mit sicherheitstechnischer Bedeutung – Softwareaspekte für rechnerbasierte Systeme zur Realisierung von Funktionen der Kategorien B und C, März 2010.

/DIN 10b/ DIN EN 62340: Kernkraftwerke – Leittechnische Systeme mit sicherheitstechnischer Bedeutung – Anforderungen zur Beherrschung von Versagen aufgrund gemeinsamer Ursache, Dezember 2010.

/DIN 13/ DIN EN 61513: Kernkraftwerke – Leittechnik für Systeme mit sicherheitstechnischer Bedeutung – Allgemeine Systemanforderungen, September 2013.

/DIN 50/ DIN EN 50128: Bahnanwendungen – Telekommunikationstechnik, Signaltechnik und Datenverarbeitungssysteme – Software für Eisenbahnsteuerungs- und Überwachungssysteme, 2011.

/DIN 61a/ DIN EN 61508-0: Funktionale Sicherheit sicherheitsbezogener elektrischer/ elektronischer/ programmierbarer elektronischer Systeme. Funktionale Sicherheit und die IEC 61508, 03.05.2005.

/DIN 61b/ DIN EN 61508-1: Funktionale Sicherheit sicherheitsbezogener elektrischer/ elektronischer/ programmierbarer elektronischer Systeme, Teil 1: Allgemeine Anforderungen, Februar 2011.

/DIN 61c/ DIN EN 61508-2: Funktionale Sicherheit sicherheitsbezogener elektrischer/ elektronischer/ programmierbarer elektronischer Systeme, Teil 2: Anforderungen an sicherheitsbezogene elektrische/ elektronische/ programmierbare elektronische Systeme, Februar 2010.

/DIN 61d/ DIN EN 61508-3: Funktionale Sicherheit sicherheitsbezogener elektrischer/ elektronischer/ programmierbarer elektronischer Systeme, Teil 3: Anforderungen an Software, März 2010.

/DIN 61e/ DIN EN 61508-4: Funktionale Sicherheit sicherheitsbezogener elektrischer/ elektronischer/ programmierbarer elektronischer Systeme, Teil 4: Begriffe und Abkürzungen, Februar 2011.

/DIN 61f/ DIN EN 61508-5: Funktionale Sicherheit sicherheitsbezogener elektrischer/ elektronischer/ programmierbarer elektronischer Systeme, Teil 5: Beispiele zur Ermittlung der Stufe der Sicherheitsintegrität, Februar 2011.

/DIN 61g/ DIN EN 61508-6: Funktionale Sicherheit sicherheitsbezogener elektrischer/ elektronischer/ programmierbarer elektronischer Systeme, Teil 6: Anwendungsrichtlinie für IEC 61508-2 und IEC 61508-3, Februar 2011.

/DIN 61h/ DIN EN 61508-7: Funktionale Sicherheit sicherheitsbezogener elektrischer/ elektronischer/ programmierbarer elektronischer Systeme, Teil 7: Überblick über Verfahren und Maßnahmen, Februar 2011.

/DIN 81/ DIN 25424-1: Fehlerbaumanalyse, Teil 1: Methode und Bildzeichen, September 1981.

/DIN 99/ DIN EN 50126: Bahnanwendungen – Spezifikation und Nachweis der Zuverlässigkeit, Verfügbarkeit, Instandhaltbarkeit, Sicherheit (RAMS), März 2000.

/DOD 10/ Department of Defense: Joint Software Systems Safety Engineering Handbook, 27. August 2010.

/ECSS 80/ ECSS-Q-80: Software Product Assurance, 1996.

/ENR 13/ Licensing of safety critical software for nuclear reactors, Common position of seven European nuclear regulators and authorised technical support organisations, Bel V, BfS, CSN, ISTec, ONR, SSM, STUK, Revision 2013

/FAA 03/ FAA Order 8110.49, Software Approval Guidelines, AIR-120, Aviation Safety - Aircraft Certification Service, Aircraft Engineering Division, Juni 2003

/FAE 09/ Prof. Dr. Fähnrich, K.-P.: Vorlesung Softwaretechnik – Vorgehensmodelle, V-Modell XT, Universität Leipzig, erreichbar unter http://tinyurl.com/hbcblqj, zitiert am 12.02.2016.

/FUS 09/ Fusani, M.: Examining software engineering requirements in safety-related standards, РАДІОЕЛЕКТРОННІ І КОМП'ЮТЕРНІ СИСТЕМИ, 2009, № 7, 02.02.2009.

/GRS 04/ Piljugin, E., Märtz, J., Heinsohn, H., Frey, W.: Fachliche Unterstützung des BMU bei der Entwicklung probabilistischer Bewertungsmethoden - Anpassung und Erprobung von Methoden zur probabilistischen Bewertung digitaler Leittechnik, GRS-A-3258, Dezember 2004.

/GRS 15/ Dr. Gottschall, M., Dr. Lindner, F., Piljugin, E., Vogt, P.: Entwicklung eines Ansatzes zur Analyse der Netzwerktechnologien in sicherheitsrelevanten Leittechniksystemen hinsichtlich Verarbeitung und Auswirkung postulierter Fehler, GRS-377, Juni 2015.

/GRS 15a/ GRS Homepage: www.grs.de/Prognose-Freisetzungen-KKW-Unfaelle, 03.09.2015.

/GRS 15b/ R. Arians, S. Arnold, F. Lindner, H. Mbonjo, C. Quester, D. Sommer, Aufstellung von Kriterien und Kenngrößen zur deterministischen Prüfung der Eignung von Redesign-Komponenten für den Einsatz in der Sicherheitsleittechnik von Kernkraftwerken, GRS-395, März 2015.

/GRS 15c/ Arians, R., Arnold, S., Blum, S., Buchholz, M., Lochthofen, A., Quester, C., Sommer, D.: Entwicklung und Einsatz von Analysemethoden zur Beurteilung software-basierter leittechnischer Einrichtungen in deutschen Kernkraftwerken, GRS-355, März 2015.

/GRS 15d/ Jopen, M., Lindner, F., Piljugin, E., Vogt, P.: Entwicklung eines Ansatzes zur Analyse der Netzwerktechnologien in sicherheitsrelevanten Leittechniksystemen hinsichtlich Verbreitung und Auswirkung postulierter Fehler, GRS-377, Juni 2015.

/IAEA 00/ IAEA: Software for Computer Based Systems Important to Safety in Nuclear Power Plants, Safety Standard Series NS-G-1.1, 2000.

/IAEA 02/ IAEA: Instrumentation and Control Systems Important to Safety in Nuclear Power Plants, Safety Standard Series NS-G-1.3, 2002.

/IAEA 09/ IAEA Nuclear Energy Series, Protection against Common Cause Failures in Digital I&C Systems of Nuclear Power Plants, No. NP-T-1.5

/IAEA 94/ IAEA Technical Report Series No. 367, Software Important to Safety in Nuclear Power Plants, 1994

/IAEA 99/ IAEA Technical Report Series No. 384, Verification and Validation of Software Related to the Safety of Nuclear Power Plant Instrumentation and Control, 1999

/IEC 01/ IEC 60812: Analysetechnik für die Funktionsfähigkeit von Systemen – Verfahren für die Fehlerzustandsart- und -auswirkungsanalyse (FMEA), 2001.

/IEC 05/ IEC 62430: Nuclear Power Plants – Instrumentation and Control Systems important to Safety – Common Cause Failure, 2005.

/IEC 06/ IEC 61025: Fault tree analysis (FTA), 2006.

/IEEE 03/ IEEE: Standard Criteria for Digital Computers in Safety Systems of Nuclear Power Plants, Std. 7-4.3.2, 2003.

/IEEE 08/ IEEE: Recommended Practice on Software Reliability, Std 1633-2008, 2008.

/IEEE 09/ IEEE: Standard for a Software Quality Metrics Methodology, Std 1061-1998, 2009.

/IEEE 10/ IEEE: Standard Framework for Reliability Prediction of Hardware, Std. 1413-2010, 2010.

/IEEE 12/ IEEE: Standard for System and Software Verification and Validation, IEEE Computer Society, Std 1012-2012, 2012.

/IEEE 13/ IEEE: Standard Framework for Reliability Prediction of Hardware, IEEE Reliability Society, Std 1413-2010, 2010.

/IEEE 33/ IEEE: Recommended Practice on Software Reliability, IEEE Reliability Society, Std 1633-2008, 27.06.2008.

/IEEE 74/ IEEE: Standard for Developing Software Life Cycle Processes, IEEE Computer Society Std 1074-1997, 1997.

/ISO 08/ ISO/IEC 12207: Systems and software engineering – Software life cycle processes, 2008.

/ISO 11/ ISO 26262: Road vehicles – Functional Safety. 15. November 2011.

/ISO 13/ DIN EN ISO 13849-2: Sicherheit von Maschinen - Sicherheitsbezogene Teile von Steuerungen - Teil 2: Validierung, Februar 2013.

/ISO 15/ ISO 13849-1: Sicherheit von Maschinen - Sicherheitsbezogene Teile von Steuerungen, Teil 1: Allgemeine Gestaltungsleitsätze, 15.12.2015.

/IST 10/ ISTEC: Komplexitätsmessung der Software digitaler Leittechniksysteme, A-1569, Februar 2010.

/KOO 16/ Koord, Y., Krauter, V.: Überblick Vorgehensmodelle im Projektmanagement, FOM Hamburg, 2016, erreichbar unter http://tinyurl.com/he2jkhs/, zitiert am 12.02.2016.

/KOS 13/ Prof. Dr. Koschke R.: Softwaretechnik, Fachbereich Mathematik und Informatik, Universität Bremen, 2013, erreichbar unter https://www.informatik.uni-bremen.de/st/lehre/swt13/prozesse.pdf, zitiert am 12.02.2016.

/KTA 12/ Kerntechnischer Ausschuss: KTA 3902, Auslegung von Hebezeugen in Kernkraftwerken, November 2012.

/KTA 31/ Kerntechnischer Ausschuss: KTA 3501, Reaktorschutzsystem und Überwachungseinrichtung des Sicherheitssystems, Januar 2016.

/KTA 33/ Kerntechnischer Ausschuss: KTA 3503, Typprüfung von elektrischen Baugruppen der Sicherheitsleittechnik, Januar 2016.

/LDRA 05/ LDRA: Testbed Manual, Revision 23, 2005.

/LIT 74/ Littlewood, B., Verrall, J. L: A Bayesian Reliability Model with a Stochastically Monotone Failure Rate, IEEE Transactions on Reliability, Vol. R-23, No. 2, June 1974.

/LYU 96/ Lyu, M. R.: Handbook of Software Reliability Engineering, April 1996.

/MIL 80/ Department of Defense: Procedures for Performing a Failure Mode, Effects and Criticality Analysis, Military Standard, MIL-STD-1629A, November 1980.

/MIL 88/ Department of Defence: System Safety, MIL-STD-882E, May 2012

/MOD 07/ Ministry of Defence, Defence Standard 00-56, Safety Management Requirements of Defence Systems, Juni 2007

/MYK 15/ Myklebust, T.: Certification on Safety-Critical Software, 14th International Conference on Software QA & Testing on Embedded Systems, 2015.

/NASA 87/ NASA-STD 8719.13, Software Safety Standard, Revision C, Juli 2013

/NRC 11/ US NRC: Criteria for use of Computers in Safety Systems of Nuclear Power Plants, Regulatory guide 1.152, Revision 3, Juli 2011.

/NUR 01/ US NRC: Digital Systems Software Requirements Guidelines, NUREG/CR-6734, Juni 2001

/NUR 05/ US NRC: Current state of reliability modeling methodologies for digital systems and their acceptance criteria for nuclear power plant assessments, NUREG/CR-6901, Oktober 2005.

/NUR 07/ US NRC: Standard Review Plan for the Review of Safety analysis Reports for Nuclear Power Plants. Review Process for Digital Instrumentation and Control Systems, NUREG 800, Appendix 7.0-A, März.2007.

/NUR 08/ US-NRC: Diversity Strategies for Nuclear Power Plant Instrumentation and Control Systems, NUREG/CR-7007, Dezember 2008.

/NUR 11/ US NRC: Development of Quantitative Software Reliability Models for Digital Protection Systems of Nuclear Power Plants, NUREG/CR-7044, Juli 2011.

/NUR 87/ US NRC: Standard Review Plan for the Review of Safety analysis Reports for Nuclear Power Plants, NUREG 800, 1987.

/NUR 96/ US NRC: Development of Tools for Safety Analysis of Control Software in Advance Reactors, NUREG/CR-6465, 1996.

/ONT 99/ CE-1001-STD, Revision 2: Standard for Software Engineering of Safety Critical Software, Ontario Power Generation, Dezember 1999

/REES 94/ Rees, R.A.: Detectability of Software Failures, Reliability Review, Vol 14, No. 4, 1994.

/RSK 11/ RSK/ESK-Geschäftsstelle beim BfS, Rechnerbasierte Sicherheitsleittechnik für den Einsatz in der höchsten Sicherheitskategorie in deutschen Kernkraftwerken, Darstellung der Beratungsergebnisse der RSK-Arbeitsgruppe, Einsatz rechnerbasierte Leittechnik, 20.09.2011

/RTCA 11/ RTCA: Software Considerations in Airborne Systems and Equipment Certification, DO-178C, 13.12.2011.

/RTCA 92/ RTCA: Certification Standard for Avionik, DO-178B, 1992

/SAE 09/ SAE International: Potential Failure Mode and Effects Analysis in Design (Design FMEA) and Potential Failure Mode and Effects Analysis in Manufacturing and Assembly Processes (Process FMEA) and Effect Analysis for Machinery (Machinery FMEA), Standard J-1739, 02.08.2002.

/SAG 07/ Saglietti, F.: Computer Safety, Reliablitity, and Security, 27th International Conference, SAFECOMP 2007.

/SCH 01/ Scholz, F.: Bewertungsmöglichkeit der Softwarezuverlässigkeit digitaler Leittechnik, August 2001.

/STUK 01/ Helminen, A: Reliability Estimation of Saftey-Critical Software-Based Systems Using Bayesian Networks, STUK-YTO-TR 178, Juni 2001.

/STUK 02/ Haapanen, P., Helminen, A.: Failure Mode and Effects Analysis of Software-based Automation Systems. STUK-YTO-TR 190, August 2002.

/TEL 97/ Siemens AG, Schulung TELEPERM XS, 1997

/THU 04/ Thums, A.: Formale Fehlerbaumanalyse, Dissertation, 2004.

/VDI 07/ VDI: Zuverlässigkeitsmanagement, VDI 4003, März 2007.

/VDI 11/ VDI/VDE 3528: Anforderungen an Serienprodukte und Kriterien für deren Einsatz in der Sicherheitsleittechnik von Kernkraftwerken, August 2011.

/VDT 08/ VdTÜV: Stellungnahme zu den erforderlichen Vorsorgemaßnahmen gegen systematisches Versagen von digitalen leittechnischen Einrichtungen in kerntechnischen Anlagen, die Leittechnikfunktionen der Kategorie 1 ausführen, 2008.

/YVL 97/ Finnish Centre for Radiation and Nuclear Safety: Probabilistic Safety Analyses (PSA), Helsinki, YVL 2.8, pp. 1-10, 199

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

Die nachfolgende Tabelle gibt die in diesem Dokument verwendeten Abkürzungen mit ihrer Bedeutung wieder.

Abkürzung	Bedeutung	Erläuterung
AECL	Atomic Energy of Canada Limited	Firma nach kanadischem Recht im Besitz der Krone bzw. der Königin der Vereinigten Königreiche
AOP	Allied Ordnance Publication	NATO-Publikation zur militärischen Ausrüstung, Waffen und Munition
BBN	Bayesian Belief Networks	Ein mathematisches, probabilistisches, grafisches Modell in der Stochastik
BSI	British Standards Institution	Das britische Pendant zum Deutschen Institut für Normung e.V. heißt heute nur noch British Standards, wurde ehemals aber als British Standards Institution bezeichnet
CANDU-Reaktor	Canada Deuterium Uranium Reaktor	Schwerwasserreaktor, entwickelt von AECL
CCF	Common Cause Failure	Gemeinsam Verursachte Ausfälle, GVA
CFR	Code of Federal Regulations (USA)	Sammlung von Bundesrichtlinien der USA, bedeutende Quelle für das Bundesrecht der Vereinigten Staaten
COTS	Commercial-Off-The-Shelf	Seriengefertigte Produkte aus dem Elektronik- oder Softwaresektor
DFM	Dynamic Flowgraph Methodology	Eine rechnergestütze Annäherung an die Zuverlässigkeitsanalyse
DID	Design Input Document	Pflichtenheft
DIN	Deutsches Institut für Normung e.V.	Bedeutendste nationale Normungsorganisation in der Bundesrepublik Deutschland
E/E/PE (System)	Electrical/Electronic/Programmable Electronic (System)	Elektrische/Elektronische/Programmierbare Elektronische (-Systeme)
ECSS	European Cooperation for Space Standardization	Initiative der ESA, nationaler Raumfahrtagenturen und der Raumfahrtindustrie zur Ausarbeitung einheitlicher Normen
ESA	European Space Agency	Europäische Weltraumorganisation

Abkürzung	Bedeutung	Erläuterung
ESFAS	Engineered Safety Feature Actuation System	Leittechnische Funktion des Sicherheitssystem
FAA	Federal Aviation Administration	Bundesluftfahrtbehörde der Vereinigten Staaten
FPGA	Field Programmable Gate Array	Integrierter Schaltkreis der Digitaltechnik, in den eine logische Schaltung programmiert werden kann
FTA	Fault Tree Analysis	Fehlerbaumanalyse
GVA	Gemeinsam Verursachte Ausfälle	In der Risikoanalyse betrachtete Ausfälle von mehreren Komponenten als Folge einer einzelnen Fehlerursache
I&C	Information and Communication	Leittechnik
IAEA	International Atomic Energy Agency	Internationale Atomenergie-Organisation (IAEO), mit den Vereinten Nationen durch Abkommen verbundene Organisation
IEC	International Electrotechnical Commission	Internationale Elektrotechnische Kommission, eine internationale Normungsorganisation
IEEE	Institute of Electrical and Electronics Engineers	Weltweiter Berufsverband von Ingenieuren, hauptsächlich aus den Bereichen Elektrotechnik und Informationstechnik
ISTEC	Industrielle Software-Technik GmbH	Institut für Sicherheitstechnologie GmbH (ISTec), eine GRS-Tochter
KTA	Kerntechnischer Ausschuss	1972 per Erlass durch das Bundesministerium für Bildung und Wissenschaft nach dem Vorbild des "Deutschen Dampfkessel-Ausschusses" gebildeter Ausschuss
LAN	Local Area Network	Lokales Rechnernetz
LCSAJ	"Linear Code Sequence and Jump"-Metrik	Eine softwarebasierte Analysemethode zur Beurteilung eines Testumfangs
LOC	Lines of Code	Anzahl von Code-Zeilen eines Computerprogramms
NASA	National Aeronautics and Space Administration	Bundesbehörde für Raumfahrt und Flugwissenschaft der Vereinigten Staaten
NATO	North Atlantic Treaty Organization	Internationale Organisation, die den Nordatlantikvertrag, ein militärisches Bündnis von 28 Staaten, umsetzt

Abkürzung	Bedeutung	Erläuterung
NHPP	Non-Homogeneous Poisson Process	Ein stochastischer Prozess der Wahrscheinlichkeitstheorie, mit dem zahlreiche Zufallsphänomene beschrieben werden können
NRC	Nuclear Regulatory Commission	Für die Sicherheit, Genehmigung und Verlängerung von Betriebslizenzen von Kernkraftwerken zuständige Behörde der Vereinigten Staaten
NUREG	Nuclear Regulatory Guide	Publikationen der American Nuclear Society (ANS)
OP	Organisatorische Prozessposten	Organisatorische Prozessposten, mit deren Hilfe innerhalb einer projektspezifischen Beschreibung der Softwarelebenszyklus-Prozess beschrieben wird
PCA	Principal Components Analysis	Hauptkomponentenanalyse
RAM	Random-Access Memory	Direktzugriffsspeicher, der besonders bei Rechnern als Arbeitsspeicher Verwendung findet
ROM	Read-Only Memory	Festwertspeicher oder auch Nur-Lese-Speicher
RSK	Reaktor-Sicherheitskommission	Gremium von Experten aus dem Bereich Kerntechnik, die den Bundesminister für Umwelt, Naturschutz und Reaktorsicherheit in Fragen der Sicherheit von Kernkraftwerken berät
RTCA	Radio Technical Commission for Aeronautics	Eine nicht gewinnorientierte Vereinigung, die in den Vereinigten Staaten Empfehlungen für Kommunikation, Navigation und Überwachung des Flugverkehrsmanagement abgibt
SCCA	Software Common Cause Analysis	Analyse von Gemeinsam Verursachten Ausfällen in Software
SFMECA	Software Failure Modes, Effects and Criticality Analysis	Ausfalleffekt- und Ausfallkritizitätsanalyse der ECSS bezüglich Software
SFTA	Software Fault Tree Analysis	Fehlerbaumanalyse von Softwarefehlern
SIL	Sicherheits-Integritätslevel	In der internationalen Normung gemäß IEC 61508 definierter Begriff der Sicherheitsanforderungsstufe

Abkürzung	Bedeutung	Erläuterung
SLZ	Softwarelebenszyklus	Zeitablauf einer Softwarelösung, bestehend aus den Phasen Probelentstehung, Entwicklungsprozess, Implementierung und Nutzung bis zur Ablösung der Software durch ihren Nachfolger
SLZP	Softwarelebenszyklus-Prozess	Prozess, der den Softwarelebenszyklus beschreibt
SRG	Software Reliability Growth	Zuwachs an Softwarezuverlässigkeit
SSL	Softwaresicherheitslebenszyklus	Softwarelebenszyklus mit Fokus auf der Softwaresicherheit
STUK	Säteilyturvakeskus (Finish Nuclear Safety Research Organzation)	Finnische Strahlenschutzbehörde
SwCI	Software Criticality Index	Indizes innerhalb einer Matrix, die beschreiben, wie sicherheitskritisch eine Software ist
SWFMEA	Software Failure Modes, Effects and Analysis	Methode innerhalb SFMECA zum systematischen Vorgehen bei der Analyse eines Software-Systems, um mögliche Fehlerzustandsarten, ihre Ursachen und ihre Auswirkungen zu ermitteln
WENRA	Western European Nuclear Regulators' Association	Unabhängiges Netzwerk der europäischen Aufsichtsbehörden der kernenergiebetreibenden Länder der EU und der Schweiz

A Definition verwendeter Begriffe

In unterschiedlichen Anwendungsgebieten werden in Normen, Regelwerken und Standards Begriffe teils mit sehr unterschiedlicher Bedeutung verwendet. In Kapitel 2.1 wurden bereits die Definitionen angegeben, welche für die GRS maßgebend sind. Im Folgenden sind die jeweils originalen Definitionen aus den für dieses Dokument relevanten Normen und Standards aufgeführt.

- **Anforderungen an Software (engl.: requirements)**

Department of Defence (USA), Joint Software Systems Safety Engineering Handbook	
requirement	1. A condition or capability needed by a user to solve a problem or achieve an objective. 2. A condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed documents. 3. A documented representation of a condition or capability as in (1) or (2). [IEEE 610.12-1990]
ISO/IEC/IEEE 24765 Software Vocabulary	
requirement	1. A condition or capability needed by a user to solve a problem or achieve an objective. 2. A condition or capability that must be met or possessed by a system, system component, product, or service to satisfy an agreement, standard, specification, or other formally imposed documents. 3. A documented representation of a condition or capability as in (1) or (2). 4. A condition or capability that must be met or possessed by a system, product, service, result, or component to satisfy a contract, standard, specification, or other formally imposed document. Requirements include the quantified and documented needs, wants, and expectations of the sponsor, customer, and other stakeholders.
Ministry of Defence (UK), Reliability and Maintainability Assurance Guides, Defence Standard 00-42 Part 2: Software	
requirements engineering	The activities that lead to the production of the Purchaser's requirements, including requirements capture, definition, analysis and the development of derived requirements.

procurement specification	The most detailed specification of the system produced by the Purchaser's organisation, and the basis for the contract with the Contractor. Developed from the Cardinal Points Specification and the Staff Requirement, possibly with the aid of a feasibility study.
Ministry of Defence (UK), Requirements for Safety Related Software in Defence Equipment, Defence Standard 00-55 Part 1: Requirements	
requirements	Functional requirements express what the software must do. Nonfunctional requirements express properties of the final implementation, including accuracy, performance, reliability, user-interface etc.
software requirements	The system level document defining the requirements which, if met, will ensure that the SRS performs safely and according to operational need.
NASA Software Safety Guidebook, NASA-GB-8719.13	
requirement	1. Condition or capability needed by a user to solve a problem or achieve an objective. 2. Statements describing essential, necessary or desired attributes.
requirements, derived	1. Essential, necessary or desired attributes not explicitly documented, but logically implied by the documented requirements. 2. Condition or capability needed, e.g. due to a design or technology constraint, to fulfill the user's requirement(s).
requirement specification	Specification that sets forth the requirements for a system or system component.
software requirements specification (SRS)	Documentation of the essential requirements (functions, performance, design constraints, and attributes) of the software and its external interfaces. (IEEE Standard 610.12-1990)

- **Anwendungsfunktion (engl.: application function)**

IEC 60880, deutsche Übersetzung	
Anwendungsfunktion	Funktion eines leittechnischen Systems, die eine Aufgabe in Verbindung mit dem gesteuerten Prozess und nicht mit dem Funktionieren des Systems selbst erfüllt. [DIN EN 61513]

- **Anwendungssoftware (engl.: application software)**

DIN EN 61508-4	
application software	Part of the software of a programmable electronic system that specifies the functions that perform a task related to the EUC rather than the functioning of, and services provided by the programmable device itself.
DIN EN 62138	
Anwendungs-software	Teil der Software eines leittechnischen Systems, durch den Anwendungsfunktionen realisiert werden. [DIN EN 61513]

- **Betriebssoftware (engl.: operational system software)**

IEC 60880, deutsche Übersetzung	
Betriebs-software	Software, die auf dem Zielprozessor während dessen Betriebs läuft, wie z. B.: Eingangs-/Ausgangs-Treiber und Dienste, Interrupt-Management, Scheduler, Kommunikationstreiber, anwendungsorientierte Bibliotheken, online-Diagnostik, Redundanz-Management.
DIN EN 62138	
Betriebs-software	Teil der Systemsoftware, dessen ausführbarer Code während des Systembetriebs am Zielsystem läuft [DIN EN 61513]

- **Computerprogramm (engl.: computer program)**

Department of Defence (USA), Joint Software Systems Safety Engineering Handbook	
computer program	A combination of computer instructions and data definitions that enable computer hardware to perform computational or control functions. [IEEE 610.12-1990]
DIN EN 62138	
Programm	Von Spezialisten verfasstes Dokument, das durch automatische Werkzeuge in einen ablauffähigen Code umgesetzt wird. ANMERKUNG Dies schließt traditionelle Programme mit ein, die in einer allgemeinen Sprache geschrieben sind.

	Weiter sind Programme in anwendungsorientierten Sprachen eingeschlossen.
IEC 60880, deutsche Übersetzung	
Rechnerprogramm	Satz geordneter Instruktionen und Daten, in dem Rechenvorgänge in einer für den Rechner geeigneten Form spezifiziert werden.
ISO/IEC/IEEE Software Vocabulary	
computer program	1. A combination of computer instructions and data definitions that enable computer hardware to perform computational or control functions 2. A syntactic unit that conforms to the rules of a particular programming language and that is composed of declarations and statements or instructions needed for a certain function, task, or problem solution. [ISO/IEC 2382-1:1993]
code	1. In software engineering, computer instructions and data definitions expressed in a programming language or in a form output by an assembler, compiler, or other translator. 2. To express a computer program in a programming language. 3. A character or bit pattern that is assigned a particular meaning.
NASA Software Safety Guidebook, NASA-GB-8719.13	
computer program	A combination of computer instructions and data definitions that enable computer hardware to perform computational or control functions.

- **Computersystem (engl.: computer system)**

ISO/IEC/IEEE Software Vocabulary	
computer system	A system containing one or more computers and associated software

- **Daten (engl.: data)**

IEC 60880, deutsche Übersetzung	
Daten	Darstellung von Information oder Anweisungen in einer für die Übertragung, Auswertung oder Verarbeitung mittels Rechnern geeigneten Weise. [IEEE 610]

- **Diversität (engl.: diversity)**

DIN EN 61508-4	
diversity	Different means of performing a required function NOTE Diversity may be achieved by different physical methods or different design approaches.
IEC 60880, deutsche Übersetzung	
Diversität	Vorhandensein von zwei oder mehreren unterschiedlichen Verfahren oder Mitteln, um ein bestimmtes Ziel zu erreichen. Diversität wird insbesondere als Schutzmaßnahme gegen GVA-Versagen eingesetzt. Sie kann erreicht werden, indem physikalisch unterschiedliche Systeme eingesetzt werden oder durch funktionale Diversität, bei der gleichartige Systeme ein bestimmtes Ziel über unterschiedliche Verfahren erreichen. [IEC 60880]
ISO/IEC/IEEE Software Vocabulary	
diversity	In fault tolerance, realization of the same function by different means EXAMPLE Use of different processors, storage media, programming languages, algorithms, or development teams
Ministry of Defence (UK), Requirements for Safety Related Software in Defence Equipment, Defence Standard 00-55 Part 1: Requirements	
diversity	Existence of different means of performing a required function, e. g. other physical principles, other ways of solving the same problem.
Sicherheitsanforderungen an Kernkraftwerke	
Diversität	Vorhandensein von zwei oder mehr funktionsbereiten Einrichtungen zur Erfüllung der Funktion, die physikalisch oder technisch verschiedenartig ausgelegt sind.

- **E/E/PE-System (engl.: E/E/PE system für electrical/electronic/programmable electronic system)**

DIN EN 61508-4	
electrical/ electronic/ programmable electronic, E/E/PE	Based on electrical (E) and/or electronic (E) and/or programmable electronic (PE) technology. NOTE The term is intended to cover any and all devices or systems operating on electrical principles. EXAMPLE Electrical/electronic/programmable electronic devices include: – electro-mechanical devices (electrical); – solid-state non-programmable electronic devices (electronic); – electronic devices based on computer technology
electrical/ electronic/ programmable electronic system, E/E/PE system	System for control, protection or monitoring based on one or more electrical/electronic programmable electronic (E/E/PE) devices, including all elements of the system such as power supplies, sensors and other input devices, data highways and other communication paths, and actuators and other output devices NOTE The E/E/PE device is shown centrally located but such device(s) could exist at several places in the E/E/PE system.

- **Fehler (engl.: fault)**

IEC 60880, deutsche Übersetzung	
Fehler	Mangel an einer Hardware-, Software- oder Systemkomponente. (DIN EN 61513, 3.22)
Sicherheitsanforderungen an Kernkraftwerke	
Fehler	(1) Abweichung der Spezifikation von den tatsächlichen Erfordernissen (Spezifikationsfehler) (2) Abweichung der tatsächlichen Ausführung eines Anlagenteils von der für die Erfüllung der Spezifikation erforderlichen konstruktiven und fertigungstechnischen Ausführung des Anlagenteils (3) Abweichung zwischen dem berechneten, beobachteten oder gemessenen Wert und dem wahren, spezifizierten oder theoretisch richtigen Wert

- **Fehlertoleranz (engl.: fault tolerance)**

IEC 60880, deutsche Übersetzung	
Fehlertoleranz	Fähigkeit eines Systems, trotz des Vorhandenseins einer begrenzten Anzahl von Hardware- oder Softwarefehlern kontinuierlich korrekt weiterzuarbeiten.

- **Formale Methoden (engl.: formal methods)**

Department of Defence (USA), Joint Software Systems Safety Engineering Handbook	
formal methods	1. The use of formal logic, discrete mathematics, and machine-readable languages to specify and verify software. 2. The use of mathematical techniques in design and analysis of the system. [NASA-GB-8719-13]
Ministry of Defence (UK), Reliability and Maintainability Assurance Guides, Defence Standard 00-42 Part 2: Software	
formal methods	The application of a mathematical process for the verification of software design compliance with the specification.
Ministry of Defence (UK), Requirements for Safety Related Software in Defence Equipment, Defence Standard 00-55 Part 1: Requirements	
formal method	A software specification and development method, based on a mathematical system, that comprises: a collection of mathematical notations addressing the specification, design and development phases of software production; a well-founded logical inference system in which formal verification proofs and proofs of other properties can be formulated; and a methodological framework within which software may be developed from the specification in a formally verifiable manner.

- **Funktionale Diversität (engl.: functional diversity)**

IEC 60880, deutsche Übersetzung	
Funktionale Diversität	Anwendung der Diversität auf der Funktionsebene (z. B. Ableitung eines Abschaltkriteriums sowohl aus Druck- als auch Temperaturgrenzwerten).

- **Funktionale Validierung (engl.: functional validation)**

DIN EN 62138	
Funktionale Validierung	Prüfung der Übereinstimmung von Spezifikationen der Anwendungsfunktionen mit den originären funktionalen und Leistungsfähigkeitsanforderungen der Anlage. Die funktionale Validierung ist als Ergänzung zur Systemvalidierung zu sehen, bei der die Übereinstimmung des Systems mit den Spezifikationen der Anwendungsfunktionen geprüft wird. [DIN EN 61513]

- **Hardware (engl.: (computer) hardware)**

Department of Defence (USA), Joint Software Systems Safety Engineering Handbook	
hardware	Physical equipment used to process, store, or transmit computer programs or data. [IEEE 610.12-1990]
ISO/IEC/IEEE Software Vocabulary	
hardware	1. Physical equipment used to process, store, or transmit computer programs or data. 2. All or part of the physical components of an information system. [ISO/IEC 2382-1:1993]
NASA Software Safety Guidebook, NASA-GB-8719.13	
computer hardware	Devices capable of accepting and storing computer data, executing a systematic sequence of operations on computer data, or producing control outputs. Such devices can perform substantial interpretation, computation, communication, control or other logical functions.

- **Implementierung (engl.: implementation)**

ISO/IEC/IEEE Software Vocabulary	
implementation	1. The process of translating a design into hardware components, software components, or both. 2. The result of the process in (1). 3. A definition that provides the information needed to create an object and allow the object to participate in providing an appropriate set of services. [ISO/IEC 19500-2:2003] 4. The installation and customization of packaged software. 5. Construction.

	6. The system development phase at the end of which the hardware, software and procedures of the system considered become operational. [ISO/IEC 2382-20:1990] 7. A process of instantiation whose validity can be subject to test. [ISO/IEC 10746-3:1996] 8. Phase of development during which user documentation is created according to the design, tested, and revised. [ISO/IEC 26514] 9. Period of time in the software life implementation phase: cycle during which a software product is created from design documentation and debugged.

- **Initialisieren (engl.: initialise)**

IEC 60880, deutsche Übersetzung	
initialisieren	Setzen von Zählern, Schaltern, Adressen oder Inhalten von Speichern auf Null oder andere Startwerte, zu Beginn oder an bestimmten Punkten des Betriebs eines Rechnerprogramms.

- **Integration (engl.: integration)**

ISO/IEC/IEEE Software Vocabulary	
integrate	1. To combine software components, hardware components, or both into an overall system. 2. To pull in the changes from one child branch into its parent.
integration	The process of combining software components, hardware components, or both into an overall system.

- **Integrationstests (engl.: integration tests)**

IEC 60880, deutsche Übersetzung	
Integrationstests	Tests, die während des Integrationsvorgangs von Hardware/Software vor Validierung des Rechner-Systems durchgeführt werden, um die Kompatibilität der Software und Hardware des Rechners zu verifizieren.

- **Integrierter Schaltkreis (engl.: integrated circuit (IC))**

ISO/IEC/IEEE Software Vocabulary	
integrated circuit (IC), Syn: micro-chip, chip	A small piece of semiconductive material that contains interconnected electronic elements. [ISO/IEC 2382-1:1993]

- **Kategorie einer leittechnischen Funktion (engl.: category of an I&C function)**

DIN EN 62138	
Kategorie einer leittechnischen Funktion	Eine von drei möglichen Sicherheitszuordnungen (A, B, C) von leittechnischen Funktionen, die aus der Sicherheitsrelevanz der durchzuführenden Funktionen resultieren. Wenn die Funktion für die Sicherheit nicht signifikant ist, erfolgt keine Sicherheitszuordnung (Kategorisierung). [DIN EN 61513]
ISO/IEC/IEEE Software Vocabulary	
category	1. A specifically defined division or grouping of software based upon one or more attributes or characteristics. [ISO/IEC TR 12182:1998] 2. An attribute of an anomaly to which a group of classifications belongs. [IEEE Std 1044-1993 (R2002)]

- **Klassifizierung (engl.: classification)**

DIN EN 62138	
Klasse eines leittechnischen Systems	Eine von drei möglichen Zuordnungen (1, 2, 3) sicherheitstechnisch wichtiger leittechnischer Systeme, entsprechend der Anforderung, leittechnische Funktionen unterschiedlicher Sicherheitsrelevanz zu realisieren. Wenn das leittechnische System keine Funktion von sicherheitstechnischer Bedeutung realisiert, wird es als unklassifiziert eingestuft. [DIN EN 61513]
ISO/IEC/IEEE Software Vocabulary	
classification	1. A choice within a category. [IEEE Std 1044-1993 (R2002)] 2. The manner in which the assets are organized for ease of search and extraction within a reuse library.

	[IEEE Std 1517-1999 (R2004)]

- **Komplexität (engl.: complexity)**

DIN EN 62138	
Komplexität	Schwierigkeitsgrad der Verständlichkeit und Verifizierbarkeit eines Systems oder einer Komponente aufgrund von Auslegung, Realisierung oder Verhalten.
ISO/IEC/IEEE Software Vocabulary	
complexity	1. The degree to which a system's design or code is difficult to understand because of numerous components or relationships among components. 2. Pertaining to any of a set of structure-based metrics that measure the attribute in 1. 3. the degree to which a system or component has a design or implementation that is difficult to understand and verify
Department of Defence (USA), Joint Software Systems Safety Engineering Handbook	
complexity	An attribute of systems or items which makes their operation difficult to comprehend. Increased system complexity is often caused by sophisticated components and multiple interrelationships.
Ministry of Defence (UK), Requirements for Safety Related Software in Defence Equipment, Defence Standard 00-55 Part 1: Requirements	
complexity	The degree to which a system or component has a design or implementation that is difficult to understand and verify.

- **Konzept der gestaffelten Verteidigung (engl.: defence in depth)**

IEC 60880, deutsche Übersetzung	
Konzept der gestaffelten Verteidigung	Anwendung von mehr als einer Schutzmaßnahme zum Erreichen eines gegebenen Sicherheitsziels, sodass dieses erreicht wird, auch wenn eine der Schutzmaßnahmen versagt. [IAEA Safety Glossery]

- **Metrik (engl.: metric)**

ISO/IEC/IEEE Software Vocabulary	
metric	1. A combination of two or more measures or attributes. [ISO/IEC 20926:2003] 2. A quantitative measure of the degree to which a system, component, or process possesses a given attribute. 3. The defined measurement method and the measurement scale. [ISO/IEC 14598-1:1999]

- **PE-System (engl.: PE system für programmable electronic system)**

DIN EN 61508-4	
programmable electronic system, PE system	System for control, protection or monitoring based on one or more programmable electronic devices, including all elements of the system such as power supplies, sensors and other input devices, data highways and other communication paths, and actuators and other output devices.
programmable electronic, PE	Based on computer technology which may be comprised of hardware, software, and of input and/or output units. NOTE This term covers microelectronic devices based on one or more central processing units (CPUs) together with associated memories, etc. EXAMPLE The following are all programmable electronic devices: – microprocessors; – micro-controllers; – programmable controllers; – application specific integrated circuits (ASICs); – programmable logic controllers (PLCs); – other computer-based devices (for example smart sensors, transmitters, actuators).

- **Programmierbares, rechnerbasiertes System (engl.: computer-based system)**

IEC 60880, deutsche Übersetzung	
rechnerbasiertes System	Leittechnisches System, dessen Funktionen meistens von Mikroprozessoren, programmierten elektronischen Einheiten oder Rechnern abhängen oder durch die Verwendung derartiger Gerätschaften zur Gänze durchgeführt werden. ANMERKUNG Entspricht den Ausdrücken digitales System, softwarebasiertes System, programmierbares System. [DIN EN 61513]

- **Programmiersprache (engl.: programming language)**

ISO/IEC/IEEE Software Vocabulary	
computer language	A language designed to enable humans to communicate with computers.

- **Qualifizierung (engl.: qualification)**

ISO/IEC/IEEE Software Vocabulary	
qualification	1. Process of demonstrating whether an entity is capable of fulfilling specified requirements. [ISO/IEC 12207:2008] 2. The process of determining whether a system or component is suitable for operational use.
US Federal Aviation Administration - Order 8110.49	
tool qualification	Tool qualification is the process necessary to obtain certification credit for a software tool within the context of a specific airborne system (see RTCA/DO-178B, Section 12.2 and Glossary).
Department of Defence (USA), Joint Software Systems Safety Engineering Handbook	
qualification testing	Testing conducted to determine whether a system or component is suitable for operational test. [IEEE 610.12-1990] Quality: (1) The degree to which a system, component, or process meets specified requirements. (2) The degree to which a system, component, or process meets customer or user needs or expectations. [IEEE610.12-1990]

- **Qualität (engl.: quality)**

ISO/IEC/IEEE Software Vocabulary	
quality	1. The degree to which a system, component, or process meets specified requirements. [IEEE Std 829-2008.] 2. Ability of a product, service, system, component, or process to meet customer or user needs, expectations, or requirements. 3. The totality of characteristics of an entity that bear on its ability to satisfy stated and implied needs. [ISO/IEC 9126-1:2001] 4. Conformity to user expectations, conformity to user requirements, customer satisfaction, reliability, and level of defects present. [ISO/IEC 20926:2003]
Department of Defence (USA), Joint Software Systems Safety Engineering Handbook	
quality	1. The degree to which a system, component, or process meets specified requirements. 2. The degree to which a system, component, or process meets customer or user needs or expectations. [IEEE610.12-1990]

- **Qualitätssicherung (engl.: quality assurance)**

Department of Defence (USA), Joint Software Systems Safety Engineering Handbook	
quality assurance	1. A planned and systematic pattern of all actions necessary to provide adequate confidence that an item or product conforms to established technical requirements. 2. A set of activities designed to evaluate the process by which products are developed or manufactured. [IEEE610.12-1990]
ISO/IEC/IEEE Software Vocabulary	
quality management	Coordinated activities to direct and control an organization with regard to quality. [ISO/IEC TR 19759:2005]
quality management plan	[Output/Input] the quality management plan describes how the project management team will implement the performing organization's quality policy. The quality management plan is a component or a subsidiary plan of the project management plan.

NASA Software Safety Guidebook, NASA-GB-8719.13	
software assurance	The process of verifying that the software developed meets the quality, safety, reliability, security requirements as well as technical and performance requirements. Assurance looks at both the process used to develop the software and the analyses and tests performed to verify the software. Software Quality Assurance (SQA) and Software Product Assurance (SPA) are sometimes used interchangeably with Software Assurance.

- **Rechner (engl.: computer)**

IEC 60880, deutsche Übersetzung	
Rechner	Programmierbare Funktionseinheit, die aus einem oder mehreren Prozessoren und peripherem Gerät besteht, und die durch interne Programme gesteuert wird und wesentliche Rechnungen durchführen kann, einschließlich umfangreicher arithmetischer oder logischer Rechenvorgänge, ohne menschlichem Eingriff während eines Rechenlaufs. ANMERKUNG Ein Rechner kann sowohl ein einzelner Rechner sein als auch eine Zusammenschaltung verschiedener Einheiten. [ISO 238/1]

- **Redundanz (engl.: redundancy)**

DIN EN 61508-4	
redundancy	The existence of more than one means for performing a required function or for representing information. EXAMPLE Duplicated functional components and the addition of parity bits are both instances of redundancy. NOTE 1 Redundancy is used primarily to improve reliability (probability of functioning properly over a given period of time) or availability (probability of functioning at given instant). It may also be used in order to minimize spurious actions through architectures such as 2oo3. NOTE 2 The definition in IEV 191-15-01 is less complete. NOTE 3

	Redundancy may be "hot" or "active" (all redundant item running at the same time), "cold" or "stand-by" (only one of the redundant item working at the same time), "mixed" (one or several items running and one or several items in stand-by at the same time). [IEC 62059-11]
IEC 60880, deutsche Übersetzung	
Redundanz	Bereitstellung unterschiedlicher (identischer oder verschiedener) Strukturen, Systeme oder Komponenten, sodass jedes, unabhängig vom Betriebszustand oder Versagen irgendeines anderen, die geforderte Funktion ausüben kann. [IAEA NS-G-1.3, Glossary]
ISO/IEC/IEEE Software Vocabulary	
redundancy	In fault tolerance, the presence of auxiliary components in a system to perform the same or similar functions as other elements for the purpose of preventing or recovering from failures.
NASA Software Safety Guidebook, NASA-GB-8719.13	
redundancy	Provision of additional functional capability (hardware and associated software) to provide at least two means of performing the same task.
Sicherheitsanforderungen an Kernkraftwerke	
Redundanz	Vorhandensein von mehr funktionsbereiten Einrichtungen, als zur Erfüllung der vorgesehenen Funktion notwendig

- **Software (engl.: software)**

IEC 60880, deutsche Übersetzung	
Software	Programme (d. h. Sätze geordneter Instruktionen), Daten, Regeln und zugehörige Dokumentation betreffend den Betrieb eines rechnerbasierten leittechnischen Systems.
DIN EN 61508	
Software	Intellectual creation comprising the programs, procedures, data, rules and any associated documentation pertaining to the operation of a data processing system. NOTE 1 Software is independent of the medium on which it is recorded. NOTE 2 This definition without Note 1 differs from ISO/IEC 2382-1 by the addition of the word data.

DIN EN 62138	
Software	Programme (d. h. Sätze geordneter Instruktionen), Daten, Regeln einschließlich zugehöriger Dokumentation, die zum Betrieb eines rechnerbasierten leittechnischen Systems gehören.
ISO/IEC/IEEE Software Vocabulary	
Software	1. All or part of the programs, procedures, rules, and associated documentation of an information processing system. [ISO/IEC 2382-1:1993] 2. Computer programs, procedures, and possibly associated documentation and data pertaining to the operation of a computer system. [IEEE Std 829-2008] 3. Program or set of programs used to run a computer. [ISO/IEC 26514]
Department of Defence (USA), MIL-STD-882E	
Software	A combination of associated computer instructions and computer data that enable a computer to perform computational or control functions. Software includes computer programs, procedures, rules, and any associated documentation pertaining to the operation of a computer system. Software includes new development, complex programmable logic devices (firmware), NDI, COTS, GOTS, re-used, GFE, and Government-developed software used in the system.
Ministry of Defence (UK), Requirements for Safety Related Software in Defence Equipment, Defence Standard 00-55 Part 1: Requirements	
Software	Computer programs, procedures and associated documentation and data pertaining to the operation of a computer system.
NASA Software Safety Guidebook, NASA-GB-8719.13	
Software	1. Computer programs and computer databases. Note: although some definitions of software include documentation, MIL-STD-498 limits the definition to programs and computer databases in accordance with Defense Federal Acquisition Regulation Supplement 227.401 (MIL-STD-498). 2. Organized set of information capable of controlling the operation of a device.
US Federal Aviation Administration - Order 8110.49	
Software	Software are computer programs and, possibly, associated documentation and data pertaining to the operation of a computer system. [RTCA/DO-178B, Glossary]

- **Softwaredesign (engl.: software design)**

DIN EN 61508-4	
software design	The use of scientific principles, technical information, and imagination in the definition of a software system to perform pre-specified functions with maximum economy and efficiency.

- **Softwareentwicklung (engl.: software development)**

IEC 60880, deutsche Übersetzung	
Software-Entwicklung	Phase des Softwarelebenszyklus, die zur Erzeugung der Software für ein leittechnisches System oder eines anderen Softwareprodukts führt. Sie umfasst alle Tätigkeiten von der Anforderungsspezifikation bis zur Installation in der Anlage. [IEC 62138]

- **Softwaremodifizierung (engl.: software modification)**

IEC 60880, deutsche Übersetzung	
Softwaremodifizierung	Änderung eines bereits abgestimmten Dokuments, die zu einer Veränderung des ablauffähigen Codes führt. ANMERKUNG Softwaremodifizierungen können während der Entwicklung von Software auftreten (z. B. bei der Behebung von Fehlern, die in einem späteren Stadium der Entwicklung gefunden wurden), oder nachdem die Software bereits in Betrieb genommen wurde. [DIN IEC 60889]

- **Software-Version (engl.: software version)**

IEC 60880, deutsche Übersetzung	
Software-Version	Jeweilige Ausgabe eines Software-Produkts, die durch Änderung oder Korrektur eines vorhergehenden Software-Produkts erstellt wurde. [IEEE 610, modifiziert]

- **Softwarevalidierung (engl.: software validation)**

DIN EN 62138	
Software-Validierung	Test und Überprüfung der integrierten Software, um Übereinstimmung mit den in der Spezifikation des leittechnischen Systems vorgegebenen Anforderungen an Funktionalität, Leistungsfähigkeit und Schnittstellen sicherzustellen.
ISO/IEC/IEEE Software Vocabulary	
validation	The process of providing evidence that the software and its associated products satisfy system requirements allocated to software at the end of each life cycle activity, solve the right problem, and satisfy intended use and user needs. [IEEE Std 1012-2004]
Ministry of Defence (UK), Requirements for Safety Related Software in Defence Equipment, Defence Standard 00-55 Part 1: Requirements	
validation	The process of evaluating software at the end of the software development process to ensure compliance with the Software Requirement. See also preliminary validation.
preliminary validation	The tests that are performed on the Software Specification to ensure that the Software Specification correctly implements the Software Requirement.
NASA Software Safety Guidebook, NASA-GB-8719.13	
validation	1. An evaluation technique to support or corroborate safety requirements to ensure necessary functions are complete and traceable. 2. The process of evaluating software at the end of the software development process to ensure compliance with software requirements. 3. Confirmation by examination and provision of objective evidence that the particular requirements for a specific use are fulfilled (for software). The process of evaluating software to ensure compliance with specified. 4. The process of determining whether the system operates correctly and executes the correct functions.

- **Spezifikation (engl.: specification)**

IEC 60880, deutsche Übersetzung	
Spezifikation	Dokument, in dem die Anforderungen, Funktionsweisen oder andere Eigenschaften eines Systems oder einer Komponente, oft auch die Verfahren zur Überprüfung der Einhaltung dieser Angaben, in kompletter, präziser und verifizierbarer Weise angegeben sind. ANMERKUNG Es gibt verschiedene Arten von Spezifikationen, z. B. Anforderungs- oder Systemspezifikationen. [IEEE 610]

- **Systemsoftware (engl.: system software)**

DIN EN 61508-4	
system software	Part of the software of a PE system that relates to the functioning of, and services provided by, the programmable device itself, as opposed to the application software that specifies the functions that perform a task related to the safety of the EUC.
DIN EN 62138	
Systemsoftware	Teil der Software eines leittechnischen Systems, der für ein bestimmtes Rechnersystem oder eine Rechnerfamilie erstellt wurde, um Entwicklung, Betrieb und Modifikation des Systems und der zugehörigen Programme zu erleichtern. ANMERKUNG Die Systemsoftware der Gerätefamilien besteht üblicherweise aus der Betriebssoftware und der Software unterstützender Systeme (Softwarewerkzeuge). [DIN EN 61513]
IEC 60880, deutsche Übersetzung	
Systemsoftware	Teil der Software eines leittechnischen Systems, der für ein bestimmtes Rechnersystem oder eine Rechnerfamilie erstellt wurde, um Entwicklung, Betrieb und Modifizierung dieser Einrichtungen und der zugehörigen Programme zu erleichtern. [IEC 62138]

- **Systemvalidierung (engl.: system validation)**

DIN EN 61508-4	
validation	Confirmation by examination and provision of objective evidence that the particular requirements for a specific intended use are fulfilled. NOTE 1 In this standard there are three validation phases: – overall safety validation (see Figure 2 of IEC 61508-1); – E/E/PE system validation (see Figure 3 of IEC 61508-1); – software validation (see Figure 4 of IEC 61508-1). NOTE 2 Validation is the activity of demonstrating that the safety-related system under consideration, before or after installation, meets in all respects the safety requirements specification for that safety-related system. Therefore, for example, software validation means confirming by examination and provision of objective evidence that the software satisfies the software safety requirements specification. [ISO 8402, modified]
Department of Defence (USA), Joint Software Systems Safety Engineering Handbook	
validation	The determination that the requirements for a product are sufficiently correct and complete. [SAE ARP 4754]
IEC 60880, deutsche Übersetzung	
Systemvalidierung	Bestätigung durch Prüfung und Beibringen anderer Nachweise, dass ein System zur Gänze die Anforderungsspezifikation erfüllt (Funktionalität, Ansprechzeit, Fehlertoleranz, Robustheit).
ISO/IEC/IEEE Software Vocabulary	
validation	1. Confirmation, through the provision of objective evidence, that the requirements for a specific intended use or application have been fulfilled. [ISO/IEC 15288:2008] 2. In a life cycle context, the set of activities ensuring and gaining confidence that a system is able to accomplish its intended use, goals and objectives. [ISO/IEC 12207:2008]

- **Verifizierung (engl.: verification)**

Department of Defence (USA), Joint Software Systems Safety Engineering Handbook	
verification	The evaluation of an implementation of requirements to determine that they have been met. [SAE ARP 4754]
DIN EN 61508-4	
verification	Confirmation by examination and provision of objective evidence that the requirements have been fulfilled. NOTE In the context of this standard, verification is the activity of demonstrating for each phase of the relevant safety lifecycle (overall, E/E/PE system and software), by analysis, mathematical reasoning and/or tests, that, for the specific inputs, the outputs meet in all respects the objectives and requirements set for the specific phase. EXAMPLE Verification activities include – reviews on outputs (documents from all phases of the safety lifecycle) to ensure compliance with the objectives and requirements of the phase, taking into account the specific inputs to that phase; – design reviews; – tests performed on the designed products to ensure that they perform according to their specification; – integration tests performed where different parts of a system are put together in a step-by-step manner and by the performance of environmental tests to ensure that all the parts work together in the specified manner. [ISO 8402, modified]
DIN EN 62138	
Verifizierung	Durch Prüfen und Vorlegen objektiver Beweise sicherstellen, dass die Ergebnisse einer Tätigkeit den Zielen und Anforderungen entsprechen, die für diese Tätigkeit definiert wurden. [ISO 12207]
IEC 60880, deutsche Übersetzung	
Verifizierung	Durch Prüfen und Vorlegen objektiver Beweise bestätigen, dass die Ergebnisse einer Tätigkeit den Zielen und Anforderungen entsprechen, die für diese Tätigkeit definiert wurden. [IEC 62138]
ISO/IEC/IEEE Software Vocabulary	
verification	1. The process of evaluating a system or component to determine whether the products of a given development phase satisfy the conditions imposed at the start of that phase. [IEEE Std 1012-2004]

	2. Formal proof of program correctness. 3. Confirmation, through the provision of objective evidence, that specified requirements have been fulfilled. [ISO/IEC 12207:2008], [ISO/IEC 15288:2008], [IEEE Std 15288-2008], [ISO/IEC 25000:2005] 4. Confirmation by examination and provision of objective evidence that the particular requirements for a specific intended use are fulfilled. [ISO/IEC 9126-1:2001] 5. The evaluation of whether or not a product, service, or system complies with a regulation, requirement, specification, or imposed condition. It is often an internal process. 6. Process of providing objective evidence that software and its associated products comply with requirements (e.g., for correctness, completeness, consistency, and accuracy) for all life cycle activities during each life cycle process (acquisition, supply, development, operation, and maintenance), satisfy standards, practices, and conventions during life cycle processes, and successfully complete each life cycle activity and satisfy all the criteria for initiating succeeding life cycle activities (e.g., building the software correctly). [IEEE Std 829-2008] NOTE [ISO 9000:2005] "Verified" is used to designate the corresponding status. In design and development, verification concerns the process of examining the result of a given activity to determine conformity with the stated requirement for that activity. A system may be verified to meet the stated requirements, yet be unsuitable for operation by the actual users.
Ministry of Defence (UK), Requirements for Safety Related Software in Defence Equipment, Defence Standard 00-55 Part 1: Requirements	
verification	The process of evaluating a system or component to determine whether the products of a given development phase satisfy the conditions imposed at the start of that phase.
NASA Software Safety Guidebook, NASA-GB-8719.13	
software verification	1. The process of determining whether the products of a given phase of the software development cycle fulfill the requirements established during the previous phase(s) (see also validation). 2. Formal proof of program correctness. 3. The act of reviewing, inspecting, testing, checking, auditing, or otherwise establishing and documenting whether items, processes, services, or documents conform to specified requirements. 4. Confirmation by examination and provision of objective evidence that specified requirements have been fulfilled (for software).

	5. The process of evaluating the products of a given phase to determine the correctness and consistency of those products with respect to the products and standards provided as input to that phase.
formal verification	(For Software) The process of evaluating the products of a given phase using formal mathematical proofs to ensure correctness and consistency with respect to the products and standards provided as input to that phase.
Sicherheitsanforderungen an Kernkraftwerke	
Verifizierung	Bestätigung durch Bereitstellung eines objektiven Nachweises, dass festgelegte Kriterien erfüllt worden sind

- **Versagen (engl.: failure)**

IEC 60880, deutsche Übersetzung	
Versagen	Abweichen des vorliegenden Verhaltens von dem beabsichtigten Verhalten. [DIN EN 61513, modifiziert]
Sicherheitsanforderungen an Kernkraftwerke	
Versagen	Nicht- oder Fehlfunktion bei Anforderung aktiver Systeme bzw. Verlust der Integrität bzw. Funktionsfähigkeit bei passiven Systemen

- **Versagen gemeinsamer Ursache (engl.: common cause failure (CCF))**

IEC 60880, deutsche Übersetzung	
Versagen gemeinsamer Ursache	Versagen von zwei oder mehreren Strukturen, Systemen oder Komponenten infolge eines einzelnen spezifischen Ereignisses oder Grundes. [IAEA NS-G-1.3, Glossary]

- **Vorgefertigte Software (engl.: pre-developed software (PDS))**

IEC 60880, deutsche Übersetzung	
vorgefertigte Software	Software, die bereits vorgefertigt, als kommerzielles oder gesetzlich geschütztes Produkt verfügbar und für den Einsatz in einem rechnerbasierten System vorgesehen ist. [IEC 62138, modifiziert]

- **Zertifizierung (engl.: certification)**

ISO/IEC/IEEE Software Vocabulary	
certification	1. A written guarantee that a system or component complies with its specified requirements and is acceptable for operational use. 2. A formal demonstration that a system or component complies with its specified requirements and is acceptable for operational use. 3. The process of confirming that a system or component complies with its specified requirements and is acceptable for operational use. EXAMPLE A written authorization that a computer system is secure and is permitted to operate in a defined environment.
NASA Software Safety Guidebook, NASA-GB-8719.13	
certification	Legal recognition by the certification authority that a product, service, organization or person complies with the applicable requirements. Such certification comprises the activity of checking the product, service, organization or person and the formal recognition of compliance with the applicable requirements by issue of a certificate, license, approval or other document as required by national law or procedures. In particular, certification of a product involves: (a) the process of assuring the design of a product to ensure that it complies with a set of standards applicable to that type of product so as to demonstrate an acceptable level of safety; (b) the process of assessing an individual product to ensure that it conforms with the certified type design; (c) the issue of any certificate required by national laws to declare that compliance or conformity has been found with applicable standards in accordance with items (a) or (b) above.

- **Zuverlässigkeit (engl.: reliability)**

Defence Standard 00-55 part 1	
reliability	The ability of a system or component to perform its required functions under stated conditions for a specified period of time.
ISO/IEC/IEEE Software Vocabulary	
reliability	1. The ability of a system or component to perform its required functions under stated conditions for a specified period of time. 2. capability of the software product to maintain a specified level of performance when used under specified conditions.

	[ISO/IEC 9126-1:2001] NOTE Wear or aging does not occur in software. Limitations in reliability are due to faults in requirements, design, and implementation. Failures due to these faults depend on the way the software product is used and the program options selected rather than on elapsed time.
NASA Software Safety Guidebook, NASA-GB-8719.13	
reliability	The probability of a given system performing its mission adequately for a specified period of time under the expected operating conditions.

B Beschreibung von Standards und Normen

B.1 Allgemeine Standards und Normen

B.1.1 DIN EN 61508 „Funktionale Sicherheit sicherheitsbezogener elektrischer/ elektronischer/ programmierbarer elektronischer Systeme“

Bereits seit vielen Jahren werden in verschiedenen Anwendungsbereichen Systeme aus elektrischen, elektronischen oder programmierbaren elektronischen Komponenten (E/E/PE) zur Ausführung von Sicherheitsfunktionen eingesetzt. Rechnerbasierte Systeme werden vor allem in betrieblichen, zunehmend aber auch in sicherheitstechnisch wichtigen Anwendungsbereichen eingesetzt. Um diese Technologie wirksam und sicherheitsgerichtet nutzen zu können, beschreibt die DIN EN 61508 in mehreren Teilen einen allgemeinen Lösungsweg für alle Tätigkeiten während des Sicherheitslebenszyklus von Systemen, die aus elektrischen, elektronischen oder programmierbaren elek-tronischen Komponenten bestehen und die zur Ausführung von Sicherheitsfunktionen eingesetzt werden.

Die Normenreihe besteht insgesamt aus acht Teilen:

- Teil 0: Funktionale Sicherheit /DIN 61a/
- Teil 1: Allgemeine Anforderungen /DIN 61b/
- Teil 2: Anforderungen an sicherheitsbezogene E/E/PE-Systeme /DIN 61c/
- Teil 3: Anforderungen an Software /DIN 61d/
- Teil 4: Begriffe und Abkürzungen /DIN 61e/
- Teil 5: Beispiele zur Ermittlung der Stufe der Sicherheitsintegrität /DIN 61f/
- Teil 6: Anwendungsrichtlinie /DIN 61g/
- Teil 7: Anwendungshinweise über Verfahren und Maßnahmen. /DIN 61h/

Ein zentraler Begriff in dieser Normenreihe ist der Sicherheitslebenszyklus eines Systems, der die verschiedenen Lebensphasen des Systems unter dem Gesichtspunkt der funktionalen Sicherheit charakterisiert. Dieser Lebenszyklus gliedert sich nach DIN EN 61508 wie folgt in verschiedene Phasen:

1. Konzept
2. Definition des gesamten Umfangs

3. Gefahren- und Risikoanalyse
4. Gesamtsicherheitsanforderungen
5. Zuordnung der Gesamtsicherheitsanforderungen
6. Planung des gesamten Betriebs und der Instandhaltung
7. Planung der Sicherheitsvalidierung
8. Planung der Installation und Inbetriebnahme
9. Sicherheitsanforderungsspezifikation für E/E/PE-Systeme
10. Realisierung der E/E/PE-Systeme
11. Gesamtinstallation und Inbetriebnahme
12. Gesamtsicherheitsvalidierung
13. Gesamtbetrieb, Instandhaltung und Reparatur
14. Außerbetriebnahme oder Entsorgung

Nach der Erläuterung dieser Phasen werden in DIN EN 61508 Anforderungen an diese Phasen gestellt, die an Umfang und Inhalt der jeweiligen Phase ausgerichtet sind. Des Weiteren werden Methoden vorgestellt, mit der die Erfüllung der formulierten Anforderungen verifiziert werden können.

In Teil 1 der Normenreihen wird unter anderem das Sicherheitsintegritätslevel (SIL) eingeführt. Für jedes E/E/PE-System sollen die Sicherheitsfunktionen diesen SIL zugeordnet werden und die für dieses SIL spezifizierten Anforderungen für die Sicherheitsintegrität der Sicherheitsfunktion erfüllen. Dabei ist SIL 4 die höchste und SIL 1 die niedrigste Stufe der Sicherheitsintegrität.

Für jedes SIL wird eine erlaubte mittlere Ausfallwahrscheinlichkeit einer Sicherheitsfunktion bei ihrer Anforderung (Probability of Failure on Demand, PFD) und eine mittlere Ausfallrate einer Sicherheitsfunktion (Probability of Failure per Hour, PFH) festgelegt. Es gelten dabei folgende Werte:

- SIL 4: $PFD_{avg} \geq 10^{-5}$ bis $< 10^{-4}$, $PFH \geq 10^{-9}$ bis $< 10^{-8}\,h^{-1}$
- SIL 3: $PFD_{avg} \geq 10^{-4}$ bis $< 10^{-3}$, $PFH \geq 10^{-8}$ bis $< 10^{-7}\,h^{-1}$
- SIL 2: $PFD_{avg} \geq 10^{-3}$ bis $< 10^{-2}$, $PFH \geq 10^{-7}$ bis $< 10^{-6}\,h^{-1}$
- SIL 1: $PFD_{avg} \geq 10^{-2}$ bis $< 10^{-1}$, $PFH \geq 10^{-6}$ bis $< 10^{-5}\,h^{-1}$

In DIN EN 61508 werden weder Anforderungen der einzelnen SIL spezifiziert noch wird dargelegt, wie das SIL einer Sicherheitsfunktion zu bestimmen ist. Es werden jedoch in

Teil 5 mögliche Techniken und Methoden zur Bestimmung des SILs einer Sicherheitsfunktion vorgestellt.

Der im Rahmen dieses Vorhabens relevante Teil 3 der Normenreihe setzt sich insbesondere mit Anforderungen an den Sicherheitslebenszyklus der Softwarebestandteile von E/E/PE-Systemen auseinander. Dieser Software-Sicherheitslebenszyklus wird in DIN EN 61508-3 als Teilbereich des System-Sicherheitslebenszyklus betrachtet und bezieht sich auf jede Software, die in sicherheitsrelevanten Systemen eingesetzt wird. Hierzu zählen Betriebssysteme, Systemsoftware, Software in Kommunikationsnetzwerken, Funktionen der Mensch-Maschine-Schnittstelle, Firmware und Anwendungssoftware. Im Wesentlichen werden Entwurf, Entwicklung, Validierung und Modifikationen sicherheitsrelevanter Software behandelt. Außerdem werden Anforderungen an Werkzeuge gestellt, die zur Entwicklung und Konfiguration eines sicherheitsrelevanten Systems verwendet werden. Dazu zählen Werkzeuge zur Entwicklung und zum Entwurf, Übersetzer, Werkzeuge für Test und Debugging sowie Werkzeuge für das Konfigurationsmanagement. Es darf jedes Modell eines Software-Lebenszyklus eingesetzt werden, sofern es die Anforderungen der DIN EN 61508 erfüllt. Jede Phase des Software-Lebenszyklus muss darin in elementare Tätigkeiten aufgeteilt werden, wobei für jede Phase angemessene Verfahren und Maßnahmen verwendet werden müssen.

Im weiteren Verlauf werden Anforderungen an die Sicherheit der Software formuliert. Die wichtigsten Ziele sind hierbei, eine Softwarearchitektur zu schaffen, die die spezifizierten Anforderungen an die Software im Hinblick auf den geforderten Sicherheits-Integritätslevel (SIL) erfüllt, einen Satz von Werkzeugen auszuwählen, der über den gesamten Sicherheitslebenszyklus der Software die Verifizierung, Validierung, Beurteilung und Modifikation unterstützt, und zu verifizieren, dass die Anforderungen an die Software erreicht worden sind. Dabei wird betont, dass die Verantwortlichkeit für die Übereinstimmung mit den Anforderungen in Abhängigkeit von der Art der Softwareentwicklung beim Lieferanten, beim Anwender oder bei beiden liegt. Die Zuordnung muss entsprechend dokumentiert werden. Bei der Entwicklung von sicherheitsrelevanter Software müssen Programmiersprachen in Übereinstimmung mit angemessenen Programmierrichtlinien verwendet werden. Diese muss gute Programmiertechniken beschreiben, unsichere Sprachmerkmale verbieten, die Codeverständlichkeit fördern, Verifizierung und Test erleichtern und die Verfahren für die Dokumentation des Source-Codes beschreiben. Ein Konfigurationsmanagement muss darüber hinaus sicherstellen, dass nur zueinander

und zum sicherheitsrelevanten System verträgliche Werkzeuge eingesetzt werden. Dabei muss jede neue Version eines Offline-Werkzeugs qualifiziert werden. Während der Entwurfs- und Entwicklungsphase müssen außerdem Integrationstests spezifiziert werden, um die Verträglichkeit der Hard- und Software der sicherheitsrelevanten programmierbaren Elektronik sicherzustellen.

Nach der Software-Entwicklung wird in DIN EN 61508-3 auf den Betrieb und die Modifikation der Software sowie auf die Validierung der Sicherheit des Systems eingegangen. Dabei wird beispielsweise gefordert, dass die hauptsächliche Validierungsmethode für die Software der Test sein muss, der durch Analyse, Animation und Modellbildung unterstützt werden darf. Außerdem muss der Lieferant und/ oder Entwickler der Software dem Systementwickler alle Validierungsergebnisse bezüglich der Sicherheit und alle weiteren relevanten Dokumente verfügbar machen.

Abschließend werden Anforderungen an die Verifizierung der Software gestellt. Beispielweise wird gefordert, dass bei der Planung der Verifizierung auf die Bewertung der Anforderungen der Sicherheitsintegrität, die Auswahl und Dokumentation der Verifizierungsstrategien, Tätigkeiten und Verfahren und die Auswahl und Anwendung der Verifizierungswerkzeuge hingewiesen werden soll. Auch muss der Nachweis, dass die verifizierte Phase in jeder Hinsicht zufriedenstellend abgeschlossen wurde, dokumentiert werden.

B.1.2 DIN EN ISO 13849 „Sicherheit von Maschinen – Sicherheitsbezogene Teile von Steuerungen“

Die Norm beinhaltet Sicherheitsanforderungen und einen Leitfaden für die Gestaltung und Integration sicherheitsbezogener Teile von Steuerungen (SRP/CS) von Maschinen, einschließlich der Entwicklung von Software. SRP/CSs können dabei entweder aus Hardware und Software bestehen und separater oder integraler Bestandteil der Maschinensteuerung sein. Außerdem werden Validierungsverfahren für die Sicherheitsfunktionen der betreffenden Teile der Steuerungen festgelegt. Die Norm gilt für alle Arten von Maschinen, unabhängig von der verwendeten Technologie (elektrisch, hydraulisch, mechanisch) und besteht aus zwei Teilen:

- Teil 1: Allgemeine Gestaltungsleitsätze /ISO 15/
- Teil 2: Validierung /ISO 13/

Die Eigenschaft der SRP/CS eine Sicherheitsfunktion unter vorhersehbaren Bedingungen auszuführen, wird einer von fünf Performance Level (PL) zugeordnet. Diese PL werden als Wahrscheinlichkeit eines gefährlichen Ausfalls pro Stunde definiert (siehe Kap. 4.5). Diese Wahrscheinlichkeit hängt u.a. von der Hardware- und Softwarestruktur, dem Umfang der Fehlerdetektionsmechanismen, der Zuverlässigkeit von Bauteilen, dem Gestaltungsprozess, der Belastung im Betrieb, den Umgebungsbedingungen und den betrieblichen Einsatzbedingungen ab. /ISO 15/

Teil 1 der Norm stellt eine Methode auf Basis einer Kategorisierung von Strukturen nach speziellen Entwurfskriterien und spezifiziertem Verhalten bei Fehlerbedingungen bereit. Diese Kategorien werden einer von fünf Stufen zugeordnet (B, 1, 2, 3, 4). PL und diese Kategorien können auf sicherheitsbezogene Teile von Steuerungen angewandt werden. Hierzu zählen: /ISO 15/

- Nicht trennende, berührungslos wirkende oder druckempfindliche Schutzeinrichtungen,
- Steuerungsbaugruppen,
- Leistungsschalterelemente sowie
- Sicherheitsfunktionen ausführende Steuerungen in allen Arten von Maschinen.

Teil 1 der Norm geht nach Definition der PL auf ihre Bestimmung ein. Dabei wird gefordert, dass für jede Sicherheitsfunktion ein erforderlicher PL festgelegt und dokumentiert werden soll. Für die Vorgehensweise wird auf einen informativen Anhang verwiesen. Für die Bestimmung des PL wird gefordert, dass es sich um das Ergebnis einer Risikobeurteilung handelt und sich auf den Anteil der Risikominderung durch die sicherheitsbezogenen Teile der Steuerung bezieht. Je größer dieser Anteil ist, umso größer muss der erforderliche PL sein. /ISO 15/

Anschließend wird auf die Entwicklung von SRP/CS eingegangen. Eine Sicherheitsfunktion kann dabei durch ein oder mehrere SRP/CS realisiert sein oder mehrere Sicherheitsfunktionen können sich ein oder mehrere SRP/CS teilen. Darüber hinaus können SRP/CS sowohl Sicherheitsfunktionen als auch betriebliche Steuerungsfunktionen beinhalten. /ISO 15/

Im nächsten Schritt wird auf die Bewertung des erreichten PL und seine Beziehung zu SIL eingegangen (siehe Kap.4.5), bevor Software-Sicherheitsanforderungen formuliert werden. Hauptziel dieser Anforderungen ist es, lesbare, verständliche, testbare und

wartbare Software zu erhalten. Alle Tätigkeiten im Lebenszyklus von sicherheitsbezogener Embedded- oder Anwendungssoftware müssen hauptsächlich die Vermeidung von Fehlern berücksichtigen, die während des Software-Lebenszyklus eingebracht werden. /ISO 15/

Abschließend wird die Verifizierung gefordert, mit der nachgewiesen werden soll, dass für jede einzelne Sicherheitsfunktion der PL des zugehörigen SRP/CS dem erforderlichen PL entspricht. Die PLs verschiedener SRP/CS, die Teil einer Sicherheitsfunktion sind, müssen größer oder gleich dem erforderlichen PL dieser Funktion sein. /ISO 15/

Nachfolgend werden in Teil 1 der Norm Anforderungen an die Sicherheitsfunktionen formuliert. Hierzu werden zunächst eine Liste und Einzelheiten von Sicherheitsfunktionen zur Verfügung gestellt, die durch SRP/CS bereitgestellt werden können. Anschließend werden Anforderungen daran gestellt, was bei Identifikation und Spezifikation der Sicherheitsfunktionen beachtet werden muss. /ISO 15/

Anschließend werden Kategorien definiert, welche als Basisparameter dienen, um einen bestimmten PL zu erreichen. Sie legen das erforderliche Verhalten der SRP/CS bezüglich ihrer Widerstandsfähigkeit gegenüber Fehlern fest. Es werden für jede Kategorie (B, 1, 2, 3, 4) Anforderungen formuliert. /ISO 15/

Abschließend werden in Teil 1 der Norm Anforderungen an die Berücksichtigung von Fehlern und den Fehlerausschluss, an die Instandhaltung, die technische Dokumentation und die Benutzerinformationen formuliert. /ISO 15/

Teil 2 der Norm befasst sich ausschließlich mit dem Thema der Validierung. Dabei wird auf Validierungsverfahren, -leitsätze und -pläne eingegangen. Zweck des Validierungsverfahrens ist die Bestätigung, dass die Gestaltung der SRP/CS die Spezifikation der Sicherheitsanforderungen der Maschinen unterstützt. Dabei muss gezeigt werden, dass die in Teil 1 der Norm formulierten Anforderungen durch das SRP/CS erfüllt werden. Die Validierung sollte von Personen durchgeführt werden, die unabhängig von der Gestaltung der SRP/CS sind. Die Validierung besteht aus der Durchführung der Analyse und von Funktionsprüfungen unter vorhersehbaren Bedingungen in Übereinstimmung mit dem Validierungsplan. Sowohl an die Validierung durch Analyse als auch durch Prüfung werden in Teil 2 der Norm Anforderungen gestellt. Dabei muss die Validierung für das SRP/CS oder eine Kombination von SRP/CSs nachweisen, dass die in der Spezifikation

der Sicherheitsanforderungen geforderten PL und Kategorien erfüllt werden. Grundsätzlich erfordert dies eine Fehleranalyse an Hand von Schaltplänen und, falls die Fehleranalyse nicht schlüssig ist, Prüfungen durch Fehlersimulationen in den tatsächlich vorhandenen Steuerkreisen und Simulation des Verhaltens der Steuerung beim Auftreten von Fehlern. /ISO 15/

Außerdem werden Anforderungen an die Validierung der Spezifikation von Sicherheitsanforderungen an die Sicherheitsfunktion und der Sicherheitsfunktionen selber sowie der sicherheitsbezogenen Software formuliert. Weiterhin wird die Validierung der Umgebungsanforderungen, der Instandhaltungsanforderungen und der technischen Dokumentation und Benutzerinformation behandelt. Im informativen Anhang wird zudem auf Validierungswerkzeuge für mechanische, pneumatische, hydraulische und elektrische Systeme eingegangen. /ISO 15/

B.1.3 IEEE 1074 „Standard for Developing Software Life Cycle Processes“

Der Standard IEEE 1074 beschreibt die Vorgehensweise bei der Erstellung eines Software-Lebenszyklusprozesses. Hierbei handelt es sich um eine projektspezifische Beschreibung der Umsetzung eines Software-Lebenszyklus mit Hilfe der bereits vorhandenen organisatorischen Prozessabläufe. Nach IEEE 1074 sind dazu die folgenden Schritte notwendig:

- Auswahl eines Software-Lebenszyklusmodells für den Einsatz in einem spezifischen Projekt
- Erzeugung eines Software-Lebenszyklus unter Verwendung des ausgewählten Modells und der in IEEE 1074 vorgegebenen Tätigkeiten
- Umsetzung des Software-Lebenszyklus in einen Software-Lebenszyklusprozess mit Hilfe der organisatorischen Prozessabläufe.

Tätigkeiten sind hierbei definiert als auszuführende Arbeitspakete, welche jeweils spezifische Eingangs- und Ausgangsinformationen sowie eine Beschreibung der Aktivität enthalten. Sie bestimmen demnach den Übergang einer Eingangs- zu einer Ausgangsinformation.

IEEE 1074 beschäftigt sich im Wesentlichen mit der Struktur und den Anforderungen an die verschiedenen Tätigkeiten, die zur Abbildung eines generischen Software-Lebenszyklus notwendig sind. Dabei werden die verschiedenen Phasen von der Entwicklung bis zum Abbau der Software-Komponenten berücksichtigt. Dazu zählen

- Projektmanagement (z.B. Projektplanung, -kontrolle)
- Vorentwicklungsphase (z.B. Konzeptentwicklung, Softwareimport)
- Entwicklungsphase (z.B. Anfertigung der Anforderungen, Design)
- Nachentwicklungsphase (z.B. Installation, Wartung, Außerbetriebnahme)
- Umfassende Aktivitäten (z.B. Evaluation, Dokumentation).

B.1.4 IEEE 1012 „Standard for System and Software Verification and Validation"

Der IEEE Standard 1012 behandelt die Verifizierung und Validierung von Systemen. Hierin inbegriffen sind sowohl Systeme mit Hardware- als auch mit Software-Komponenten. Der Fokus liegt dabei auf der Durchführung des eigentlichen Verifizierungs- und Validierungsprozesses (V&V-Prozess) auf Basis des System-Lebenszyklus und den geforderten Eingangs- und Ausgangsdaten.

Ziel dieses Standards ist es, eine allgemeingültige Vorgehensweise für einen V&V-Prozess vorzustellen, der vom System-, Software- oder auch Hardware-Lebenszyklus unterstützt wird. Darüber hinaus soll der Inhalt für den Verifizierungs- und Validierungsplan definiert werden. Die Intensität und Rigorosität des V&V-Prozesses soll hierbei immer dem zugrunde liegenden SIL angepasst werden. Eine Übersicht, welche V&V-Teilprozesse für Systeme mit bestimmten SIL relevant sind, wird in dem Standard ebenfalls gegeben.

Zunächst werden die Begriffe Verifizierung und Validierung definiert. Sie dienen laut Standard dazu, die Qualität des Systems über den gesamten Lebenszyklus zu steigern.

Die Verifizierung ist definiert als Prozess zur Bewertung eines Systems (oder einer Komponente) hinsichtlich der Erfüllung der formalen Bedingungen an einen Entwicklungsschritt. Sie erbringt hierbei einen objektiven Nachweis über

- die Konformität mit den Anforderungen,
- die Erfüllung von Standards, üblicher Praxis und Konventionen,
- die erfolgreiche Vollendung jeder Tätigkeit des Lebenszyklus und Bestätigung der entsprechenden Kriterien.

Die Validierung ist definiert als Prozess zur Bewertung eines Systems (oder einer Komponente) hinsichtlich der Erfüllung der Spezifikationen an das System (die Komponente) am Ende des Entwicklungsprozesses. Sie erbringt den Nachweis darüber, dass

- die System-Anforderungen nach jeder Phase des Software-Lebenszyklus erfüllt sind,
- die richtigen Probleme gelöst werden,
- der Verwendungszweck in der endgültigen Umgebung erfüllt wird.

Der V&V-Prozess demonstriert somit, ob die Anforderungen korrekt, vollständig, akkurat, konsistent und prüfbar sind. Die Aufgabe umfasst hierbei die Bewertung, Analyse, Einschätzung, Kontrolle, Inspektion und die Überprüfung der Produkte und Prozesse. Wichtig ist laut Standard hierbei, dass der V&V-Prozess parallel zu allen Schritten des Lebenszyklus durchgeführt wird und nicht als deren Abschluss erfolgt.

Neben einer Erläuterung des Zusammenhangs des V&V-Prozesses und dem Lebenszyklus geht der Standard konkret auf die V&V-Prozesse für die folgenden Themengebiete ein:

- Management
- Akquisition
- Lieferplanung
- Projektierung
- Konfigurationsmanagement
- Anforderungsformulierung des Kunden
- Systembedarfserstellung
- Systemarchitekturerstellung, Software- oder Hardwarekonzept
- Systemimplementierung (Konzept, Design, Qualifizierung…)
- Systemintegration
- Software- und Hardwaremodifikation

- Betrieb
- Wartung
- Außerbetriebnahme.

Zusätzlich beinhaltet der Standard einige administrative Anforderungen, sowie Anforderungen an die Berichterstattung und Dokumentation.

B.1.5 IEEE 1413 “IEEE Standard Framework for Reliability Prediction of Hardware“

Dieser Standard widmet sich der Problematik, dass bis dato keine Vorgaben zur Erstellung einer Zuverlässigkeitsvorhersage existiert. Er weist darauf hin, dass die Vorhersage einer Zuverlässigkeit mit der gewählten Methode, der Qualität der Daten, aber auch dem Umfang der Analyse stark variieren kann. Die Dokumentation des Prozesses wird jedoch häufig nicht präsentiert. Somit ist einer Bewertung der Zuverlässigkeitsanalyse oder ein Vergleich kaum möglich. Dieser Standard hat daher eine Richtlinie entwickelt, die in einer vollständigen und konsistenten Dokumentation und Vorhersage (für Hardware) resultiert. Außenstehende sind dann in der Lage die Analyse nachzuvollziehen, ihre Vorteile und Schwächen abzuwägen oder auch verschiedene Analysen zu vergleichen.

Dieser Standard wird an dieser Stelle nur der Vollständigkeit halber erwähnt, da seine Richtlinie als Grundlage für die Umsetzung der verschiedenen Modelle und Methoden genutzt werden kann. Der Fokus des Standards liegt jedoch auf der Bewertung von Hardware.

B.1.6 IEEE 1633 “IEEE Recommended Practice on Software Reliability”

In diesem Standard werden verschiedene Modelle für die Bewertung der Software-Zuverlässigkeit und ihre jeweilige Anwendbarkeit auf bestimmte Gegebenheiten vorgestellt. Die Methoden sollen zumindest in erster Näherung die Vorhersage der Systemzuverlässigkeit ermöglichen. Zwar sind die vorgestellten Methoden in ihrer Charakteristik auf die Luft- und Raumfahrttechnik abgestimmt, sie können jedoch laut Standard auch für andere Industriezweige übernommen werden.

Im Rahmen dieses Standards werden die Grundlagen der Konzepte zur Zuverlässigkeitsbewertung mit ihren Vorteilen und Grenzen erläutert. Die Software-Zuverlässigkeit

bewertet die Abhängigkeit zwischen dem Versagen des Systems und den Faktoren, die es bedingen. Die wesentlichen Faktoren, die bei der Versagensanalyse berücksichtigt werden müssen, sind:

- Einschleppung des Fehlers
- Entfernung des Fehlers
- Anwendungsumgebung.

Üblicherweise nimmt die Versagensrate von Software mit der Identifikation und Beseitigung der Fehler mit der Zeit ab. Der Standard weist jedoch darauf hin, dass diese Annahme nicht gilt, wenn die Fehler bereits vor der Inbetriebnahme in der endgültigen Prozessumgebung entfernt werden, auch wenn einige Modelle dies implizit annehmen.

Der Standard erläutert, dass Software-Zuverlässigkeitsmodelle üblicherweise die Zuverlässigkeit von Software sowohl bewerten als auch vorhersagen sollen. Hierbei muss jedoch berücksichtigt werden, dass nicht die Vorhersage eines Zeitpunktes oder einer konkreten Art des Versagens gemeint ist, sondern die Vorhersage der Wahrscheinlichkeit für ein Versagen der Software.

Als Basis für eine erfolgreiche Zuverlässigkeitsanalyse werden laut Standard ausreichend gute Daten benötigt. Als Anforderung an die Daten werden die Fehlerfreiheit, die Genauigkeit und die Sachdienlichkeit aufgeführt.

Bei der Anfertigung einer Zuverlässigkeitsanalyse sollte laut Standard bedacht werden, dass es verschiedene Faktoren gibt, die eine Vorhersage der Zuverlässigkeit beeinflussen, wie z.B. eine Änderung der Versagens-Kriterien oder eine signifikante Veränderung des Codes oder der Computerumgebung. All diese Faktoren fordern eine neue Definition der Parameter des Modells, da ansonsten die Wirksamkeit des Modells beeinträchtigt sein kann.

Die Möglichkeiten der Quantifizierung von Software-Zuverlässigkeit werden laut Standard im Wesentlichen durch den Faktor Zeit beeinträchtigt. Viele Modelle nehmen als Grundlage für ihre Versagenswahrscheinlichkeit eine fehlerfreie Laufzeit an. Dies bedeutet jedoch, dass für eine Zuverlässigkeit von 10^{-9} mehrere Milliarden Stunden Laufzeit absolviert werden müssen. Das Erreichen einer Zuverlässigkeit nach der Hälfte der

Zeit und entsprechend doppelter Auslegung der Systeme ist lediglich bei der Verwendung funktioneller Diversität denkbar.

Zusätzlich zu dieser Diskussion von Software-Zuverlässigkeit wird im Standard eine allgemeine Methode zur Bewertung und Vorhersage von Software-Zuverlässigkeit diskutiert. Dabei wird ein 13-Schritte-Programm beschrieben, wobei jeder einzelne Schritt an das entsprechende Projekt und die Phase des Software-Lebenszyklus angepasst werden soll. Diese allgemeine Methode beinhaltet noch kein konkretes Modell zur Bewertung oder Vorhersage der Software-Zuverlässigkeit.

Im weiteren Verlauf stellt der Standard einige Modelle zur Bewertung der Zuverlässigkeit vor. Dazu zählen exponentielle nicht-homogene Modelle, nicht-exponentielle Modelle und Bayessche Wahrscheinlichkeitsmodelle.

B.1.7 BS 5760 “Reliability of systems, equipment and components, Part 8: Guide to assessment of reliability of systems containing software”

Dieser Teil der britischen Standardreihe BS 5760 beschreibt einige der verfügbaren Methoden zur Zuverlässigkeitsbewertung von Systemen im Hinblick auf Software-Fehler.

Die Zuverlässigkeit eines Systems wird in diesem Standard über die Versagenswahrscheinlichkeit unter Einbeziehung aller Komponenten (Software, Elektronik, Mechanik, Bediener) definiert. Fehler in der Software oder auch Hardware können zu einem Versagen des Systems führen. Der Standard unterscheidet hierbei zwischen physikalischem Versagen, einem Versagen des System-Designs oder einem Software-Versagen. Im Rahmen dessen wird auf die Eigenschaften von Software-Versagen vertiefend eingegangen. Es wird dabei geschlussfolgert, dass die Software-Zuverlässigkeit signifikant von der Umgebung, in der die Software arbeitet, abhängig ist. Außerdem wird erläutert, dass das Erreichen eines hohen Levels an Software-Zuverlässigkeit auf Basis der Betriebsdauer kaum möglich ist. Software-Versagen ist zwangsläufig komplex und überwiegt üblicherweise den Einfluss von Hardware-Fehlern. Letztere sind meist zufälliger Natur und somit stochastisch zu bewerten.

Im weiteren Verlauf werden einige Besonderheiten der Software-Zuverlässigkeit diskutiert. Der Standard weist hierbei darauf hin, dass das erste Problem bei der Bewertung der Zuverlässigkeit von Software die Definition des Software-Versagens ist. In unterschiedlichen Anwendungsbereichen können sehr verschiedene Arten von Fehlern und

auch verschiedene Grade von Ausfällen in ihrem Effekt als Versagen der Software interpretiert werden. Somit ist die Verwendung einer wohl definierten Struktur zur Erstellung einer Zuverlässigkeits-Bewertung unerlässlich. Im Folgenden werden verschiedene Ansätze zur Bewertung des Entwicklungsprozesses, der Produkteigenschaften und des Software-Versagens einer Zuverlässigkeits-Analyse vorgestellt und diskutiert.

Anschließend liefert der Standard Anforderungen an die Bewertung von Softwarezuverlässigkeit. Er beginnt mit der Anforderung, bei der Auswahl der Bewertungsmethode den gesetzlichen Bestimmungen oder anderen zwingenden Auflagen zu folgen. Außerdem fordert er, dass für die Bewertung des Entwicklungsprozesses quantitative Methoden bevorzugt werden sollen. Darüber hinaus gibt der Standard zu bedenken, dass das zu wählende Modell signifikant vom zu untersuchenden Datensatz abhängt und dass kein Modell vollständig korrekte Annahmen machen kann. Deshalb ist es in den meisten Fällen sinnvoll, mehrere Methoden und Modelle auf einen Datensatz anzuwenden, um repräsentative Resultate zu erhalten.

Weiterführend stellt der Standard die grundlegenden Prinzipien verschiedener Modelle zur Bewertung und Vorhersage von Softwarezuverlässigkeit vor. Auch geht der Standard konkret auf die Vor- und Nachteile der unterschiedlichen Methoden sowie ihre Grenzen in der Anwendung ein.

Abschließend liefert der Standard einen Leitfaden sowie grundlegende Empfehlungen für die Anwendung der Modelle. Die vier wesentlichen Aspekte, die im Rahmen dessen diskutiert werden, sind

- die Daten, die für die unterschiedlichen Methoden benötigt werden,
- die Methoden, mit denen die Daten gesammelt werden sollen,
- Empfehlungen für das Speichern und Archivieren der Daten und
- die Vorstellung von Prozeduren und Anforderungen für die Wartung der Software, sowie deren Effekt auf die Bewertung der Zuverlässigkeit von Software.

B.2 Nukleare Regelwerke, Standards und Normen

B.2.1 Sicherheitsanforderungen an Kernkraftwerke und zugehörige Interpretationen

Die Sicherheitsanforderungen an Kernkraftwerke (SiAnf) enthalten grundsätzliche und übergeordnete sicherheitstechnische Anforderungen, um einen sicheren Betrieb sowie die notwendige Vorsorge gegen Schäden bei anormalen Betriebszuständen oder bei Störfällen zu gewährleisten und damit das grundlegende Sicherheitsziel des Schutzes von Menschen und Umwelt vor ionisierender Strahlung jederzeit sicherzustellen.

Es werden neben Anforderungen an die Organisation und das technische Sicherheitskonzept auch Anforderungen an die Leittechnik formuliert. Demnach ist das Kernkraftwerk auf Sicherheitsebene 1 mit betrieblichen Steuer- und Regeleinrichtungen mit Leittechnikfunktionen auszurüsten. Auf Sicherheitsebene 2 sind Einrichtungen vorzusehen, die geeignet sind, bei Ereignissen der Sicherheitsebene 2 eine Anforderung von Schutzaktionen der Sicherheitsebene 3 zu vermeiden. Sicherheitsebene 3 ist mit einem Reaktorschutzsystem zu versehen, dessen Leittechnik-Funktionen bei Erreichen festgelegter Ansprechwerte Schutzaktionen auslösen und das nach folgenden Grund-sätzen auszulegen ist:

- Redundante Auslegung von Komponenten, Baugruppen und Teilsystemen
- Diversität
- Räumlich getrennte Installation entsprechend dem Wirkungsbereich möglicher versagensauslösender Ereignisse
- Selbsttätige Überwachung auf einen Ausfall hin
- Anpassung der Komponenten an die möglichen Umgebungsbedingungen
- Einfache Struktur der Software
- Begrenzung des Funktionsumfangs von Hard- und Software auf das sicherheitstechnisch notwendige Maß
- Einsatz fehlervermeidender, fehlerentdeckender und fehlerbeherrschender Maßnahmen und Einrichtungen.

Gemäß den SiAnf sind die Potentiale für und die Auswirkungen von systematischem Versagen der leittechnischen Einrichtungen auf die Ereignisabläufe der Sicherheitsebene 3 unter Berücksichtigung der verfahrenstechnischen Vorgaben zu analysieren. Es wird weiter gefordert, dass Vorkehrungen gegen systematisches Versagen zu Minderung von dessen Eintrittswahrscheinlichkeit zu treffen sind, dass es auf Sicherheitsebene 3 nicht mehr unterstellt werden muss. Diese Forderung gilt auch für rechnerbasierte und programmierbare Einrichtungen.

Speziell für den Einsatz von Software in Leittechniksystemen wird zudem gefordert, dass in den Betriebsphasen, in denen die Verfügbarkeit der Reaktorschnellabschaltung erforderlich ist, jederzeit eine Reaktorschnellabschaltung von Hand möglich sein muss, auch beim unterstellten systematischen Versagen rechnerbasierter und programmierbarer leittechnischer Einrichtungen einschließlich systematischen Softwareversagens.

Diese allgemein gehaltenen Anforderungen werden in den Interpretationen zu den SiAnf erläutert und konkretisiert. Unter anderem wird der Begriff der Dissimilarität eingeführt. Darauf aufbauend werden Anforderungen bezüglich der Auslegung im Hinblick auf systematische Ausfälle von rechnerbasierten oder programmierbaren leittechnischen Einrichtungen gestellt. Dort wird unter anderem gefordert, dass die Systeme, die Leittechnikfunktionen der Kategorie A ausführen, derart auszulegen sind, dass ein systematischer Ausfall auf der Sicherheitsebene 3 nicht mehr unterstellt werden muss. Kann für rechnerbasierte oder programmierbare leittechnische Einrichtungen diese Nachweisführung nach dem Stand von Wissenschaft und Technik nicht erfolgen, sind Vorkehrungen derart zu treffen, dass ein systematischer Ausfall von Hardware und Software auf der Sicherheitsebene 3 beherrscht wird. Hierzu wird definiert, bis zu welchem Grad an Diversität ein System auszulegen ist. Ist beispielsweise ein aktiver systematischer Ausfall sicherheitsgerichtet, bestehen keine Vorgaben zum Einsatz diversitärer Einrichtungen, während ansonsten eine zweifach oder dreifach diversitäre Auslegung gefordert wird.

Darüber hinaus enthalten die Interpretationen zu den SiAnf Anforderungen an die Qualifizierung von Hard- und Software der leittechnischen Einrichtungen. Aufgrund ihrer abgestuften Sicherheitsrelevanz wird dabei getrennt auf Leittechnik-Funktionen der Kategorien A, B und C eingegangen.

B.2.2 KTA 3501 „Reaktorschutzsystem und Überwachungseinrichtungen des Sicherheitssystems“

In den SiAnf werden Anforderungen an die Auslegung eines Reaktorschutzsystems auf Sicherheitsebene 3 sowie leittechnischen Einrichtungen auf den Sicherheitsebenen 1 und 2 gefordert, ohne jedoch konkrete Anforderungen zu stellen. Die Regel KTA 3501 beschäftigt sich daher vertiefend mit den Anforderungen an Aufbau, Ausführung, Gerätetechnik, Einbau und Prüfung der Sicherheitsleittechnik für Einrichtungen, die leittechnische Funktionen der Kategorie A und B ausführen. Sie enthält eine Zusammenstellung von Auslegungskriterien, Anforderungen an die Qualität und Qualitätssicherung sowie Anforderungen an die Funktionsweise der Sicherheitsleittechnik.

Nachdem zunächst grundlegende Anforderungen zur Ermittlung der Aufgabenstellung der entsprechenden Leittechniksysteme aufgeführt wurden, werden die Auslegungsanforderungen definiert. Dort wird beispielsweise gefordert, dass nachzuweisen ist, dass zusätzlich zu einem Störfall gleichzeitig ein Zufallsausfall, ein systematischer Ausfall und Folgeausfälle von der Gesamtheit der A-Funktions-Einrichtungen beherrscht werden.

Anschließend wird auf den Aufbau und die Ausführung der Leittechnik-Systeme eingegangen. Hierbei wird nach A- und B-Funktions-Einrichtungen unterschieden. Im Rahmen dessen werden auch Anforderungen an die Software-Qualität formuliert. Während für B-Funktions-Einrichtungen im Hinblick auf die Software-Qualität ausschließlich gefordert wird, dass die Programme robust und selbstüberwachend auszulegen sind und für die Entwicklung und Qualifizierung der Software Beschreibungen und rechnergestützte Testverfahren anzuwenden sind, die den Nachweis der korrekten Arbeitsweise unterstützen, werden bei A-Funktions-Einrichtungen deutlich höhere Anforderungen gestellt. Bei A-Funktions-Einrichtungen wird darüber hinaus beispielsweise gefordert, dass Funktionen der Anwender- und Systemsoftware in eigenständigen Softwareeinheiten zu realisieren sind, die Software in einem Phasenmodell in verifizierbaren Schritten zu entwickeln ist und dass die Software so auszulegen ist, dass keine unzulässigen Rückwirkungen von leittechnischen Einrichtungen der sicherheitstechnisch niederwertigen Kategorie auf die leittechnischen Einrichtungen der höherwertigen Kategorie auftreten. Darüber hinaus muss die Software in A-Funktions-Einrichtungen einfach aufgebaut und ihr Funktionsumfang auf das für die jeweilige Funktion notwendige Maß begrenzt sein.

B.2.3 KTA 3503 „Typprüfung von elektrischen Baugruppen der Sicherheitsleittechnik“

In der KTA 3503 wird festgelegt, für welche elektrischen Baugruppen der Leittechniksysteme, die Leittechnikfunktionen der Kategorie A und B ausführen, Typprüfungen durchzuführen sind. In einer solchen Typprüfung beurteilen Sachverständige, ob die Baugruppe den Datenblattangaben und den spezifizierten Eigenschaften entspricht.

Für rechnerbasierte oder programmierbare Baugruppen wird in KTA 3503 gefordert, dass eine Prüfung der Software und ihrer Qualitätsmerkmale im Rahmen von theoretischen Prüfungen und die Prüfung der Funktion im Rahmen von praktischen Prüfungen durchzuführen ist. Entsprechende Datenblätter sind inklusive der datentechnischen Anschlüsse vorzulegen.

Es werden zudem Anforderungen an die Software-Unterlagen formuliert. Beispielsweise muss der gesamte Entwicklungsprozess der Software oder der programmierbaren Bauelemente durch diese Unterlagen belegt werden. Auch sind der Aufbau, der Funktionsablauf und das Zeitverhalten von eingesetzten Programmen sowie implementierte Selbstüberwachungsmechanismen zu beschreiben. Die eingesetzten und vorhandenen Software-Werkzeuge sind anzugeben.

B.2.4 DIN EN 61513 „Kernkraftwerke – Leittechnik für Systeme mit sicherheitstechnischer Bedeutung: Allgemeine Systemanforderungen“

Die DIN EN 61513 enthält Anforderungen an leittechnische Systeme und Geräte, die für die Ausführung sicherheitstechnisch wichtiger leittechnischer Funktionen in Kernkraftwerken eingesetzt werden. Sie setzt die in DIN EN 61508 enthaltenen allgemeinen Anforderungen für die Anwendung im nuklearen Bereich um. Sie enthält Anforderungen und Empfehlungen für die gesamte leittechnische Architektur, die sowohl auf konventioneller fest verdrahteter als auch auf rechnerbasierter Technologie oder auf einer Kombination beider Technologien basieren kann. Sie legt fest, wie die Anforderungen an die Architektur der sicherheitstechnisch wichtigen leittechnischen Systeme aus der sicherheitstechnischen Auslegungsgrundlage des Kernkraftwerks und wie die Anforderungen an die einzelnen sicherheitstechnisch wichtigen Systeme von diesen Gesamtanforderungen abzuleiten sind.

B.2.5 DIN IEC 60880 „Kernkraftwerke – Leittechnische Systeme mit sicherheitstechnischer Bedeutung: Softwareaspekte für rechnerbasierte Systeme zur Realisierung von Funktionen der Kategorie A“

Qualität und Zuverlässigkeit der in Kernkraftwerken für sicherheitstechnische Zwecke eingesetzten Software muss während des gesamten Lebenszyklus von den grundsätzlichen Anforderungsspezifikationen über die verschiedenen Entwicklungsphasen bis hin zu Validierungs- und Verifizierungsprozeduren für Betrieb und Wartung gewährleistet sein. Ziel der DIN IEC 60880 ist es daher, die verschiedenen relevanten Sicherheitsaspekte zu identifizieren und die daraus resultierenden Anforderungen zu formulieren. Dabei konzentriert sich die DIN IEC 60880 auf leittechnische Systeme, die Leittechnik-Funktionen der Kategorie A ausführen.

Zur Herstellung von leittechnischen Systemen in Kernkraftwerken wird in DIN EN 61513 /DIN 13/ das Konzept des System-Sicherheitslebenszyklus eingeführt. Dieses Konzept wurde durch die Einführung eines Software-Sicherheitslebenszyklus für die Entwicklung rechnerbasierter Systeme erweitert. Da Software in der Regel wesentlich zur Erzeugung von Funktionen beiträgt, die von einem leittechnischen System ausgeübt werden, und auch zusätzliche Funktionen unterstützen kann, die über die Systemauslegung eingeführt wurden, ist der Software-Sicherheitslebenszyklus in den meisten Fällen voll in den System-Sicherheitslebenszyklus integriert. In DIN IEC 60880 wird daher die Software-Validierung und Integration als Teil der System-Validierung und Integration betrachtet. Beispielsweise wird klargestellt, dass ein Zugriffsschutz für Software zur Minimierung der Verwundbarkeit der Software gegenüber nichtautorisiertem Zugriff notwendig ist, die hauptsächlichen Schutzmaßnahmen üblicherweise jedoch auf Systemebene getroffen werden. Die DIN IEC 60880 umfasst daher Anforderungen und Empfehlungen hinsichtlich der allgemeinen Vorgehensweise zur Software-Entwicklung und Verifizierung, den Software-Aspekten der Systemintegration sowie den Software-Aspekten bei der Validierung eines rechnerbasierten Systems, bevor auf die Software-Aspekte bei Installation und Betrieb eingegangen wird.

Bevor die Auslegungs- und Implementierungsphase der Programmentwicklung beginnt, muss die Software-Anforderungsspezifikation verfügbar sein. Letztere umfasst Anforderungen an die Selbstüberwachung, die periodischen Prüfungen und die Erstellung der Dokumentation. Im Rahmen der Auslegung und Implementierung der Software unterscheidet die DIN IEC 60880 zwischen der Verwendung allgemeiner Sprachen bei der

Programmierung, der Verwendung anwendungsorientierter Sprachen mit zugehörigen Codegeneratoren und der Verwendung und Konfiguration von vorgefertigter Software. Die in DIN IEC 60880 beschriebenen grundlegenden Prinzipien zur Entwicklung eines hochqualitativen Codes sind davon unabhängig anwendbar, die detaillierten Anforderungen unterscheiden sich jedoch an vielen Stellen, je nachdem, ob allgemeine oder anwendungsorientierte Sprachen oder vorgefertigte Software zum Einsatz kommt. Anforderungen an Software-Werkzeuge für die Erstellung von Software sind ebenfalls enthalten. Auch auf die Qualifizierung vorgefertigter Software wird gesondert eingegangen.

Da sich eine Modifizierung der eingebauten Software normalerweise sowohl auf den Code als auch auf die Dokumentation auswirkt, werden in DIN IEC 60880 Prozeduren zur Änderungs- und Konfigurationskontrolle sowie für die Qualifizierung vorgefertigter Software dargelegt.

Weiterhin enthält die DIN IEC 60880 eine Reihe von Anforderungen zu Vorkehrungen gegen Gemeinsam Verursachte Ausfälle (GVA), die durch Softwarefehler ausgelöst werden, und der Realisierung von Diversität.

B.2.6 DIN EN 62138 „Kernkraftwerke – Leittechnik für Systeme mit sicherheitstechnischer Bedeutung: Softwareaspekte für rechnerbasierte Systeme zur Realisierung von Funktionen der Kategorien B oder C“

Zusammen mit der im letzten Abschnitt beschriebenen Norm DIN IEC 60880 deckt die DIN EN 62138 das Gebiet der Softwareaspekte rechnerbasierter Systeme ab, die in Kernkraftwerken für die Ausführung sicherheitstechnisch wichtiger Funktionen verwendet werden. Dabei konzentriert sich die DIN EN 62138 auf leittechnische Systeme, die Leittechnik-Funktionen der Kategorien B und C ausführen.

Entsprechend der abgestuften Klassifizierung der leittechnischen Systeme, die sich aus der abgestuften Sicherheitsrelevanz der Kategorien B und C von Leittechnik-Funktionen im Vergleich zu Funktionen der Kategorie A ergibt, sind auch die Anforderungen an die dort eingesetzte Software abgestuft. Um ein bestimmtes Basisniveau des Vertrauensgrades für ein sicherheitstechnisch wichtiges Leittechniksystem zu erreichen, werden zunächst für die Leittechniksysteme der Klasse 2 und 3 allgemeine Anforderungen an die Software formuliert.

Für Software in Leittechniksystemen der Klasse 3 muss demnach eine adäquate Qualitätssicherung und eine Dokumentation der Software-Anforderungsspezifikation, in der die sicherheitstechnisch wichtigen Randbedingungen festgelegt sind, der Spezifikation der Auslegung, Integration, Validierung und der Modifizierung erfolgen. Es muss nachgewiesen werden, dass die anzuwendende Software die sicherheitstechnisch wichtigen Funktionen im erforderlichen Umfang unterstützt und sie nicht negativ beeinflusst und dass die Software-Anforderungsspezifikation erfüllt wird. Außerdem muss sichergestellt sein, dass die Operateure des Leittechniksystems so früh wie möglich über Softwarefehler und Softwareversagen bezüglich sicherheitstechnisch wichtiger Funktionen informiert werden, um geeignete Maßnahmen ergreifen zu können.

Für Software der Leittechniksysteme der Klasse 2 gilt darüber hinaus, dass ein auf Prüfungen und Auslegung basierender Nachweis erbracht werden muss, dass die geforderte sicherheitsrelevante Leistungsfähigkeit unter allen spezifizierten Bedingungen gegeben ist. Außerdem ist eine Sicherheitsdokumentation für vorgefertigte Software und vorgefertigte Einrichtungen mit integrierter Software zu erstellen. Darüber hinaus werden strengere Anforderungen an Verifizierung, Konfigurationsmanagement, Auswahl und Verwendung von Softwarewerkzeugen und Sprachen, Zugriffsschutz und Fehlertoleranz und explizitere Anforderungen bezüglich Einfachheit, Klarheit, Präzision, Verifizierbarkeit, Prüfbarkeit und Modifizierbarkeit gestellt.

Detaillierte Anforderungen an leittechnische Systeme der Kategorie C und die zusätzlichen Anforderungen an leittechnische Systeme der Kategorie B werden in DIN EN 62138 dann im Folgenden getrennt behandelt. Inhaltlich sind die Anforderungen entsprechend der Anforderungen an leittechnische Systeme in DIN IEC 60880 strukturiert. Insbesondere wird auf die Verwendung vorgefertigter Software und von Geräten und Gerätefamilien, die nicht notwendigerweise für die speziellen Gegebenheiten der Nuklearenergie ausgelegt sind, die Verwendung spezieller „Black-Box“-Einrichtungen mit eingebetteter Software sowie auf die Verwendung anwendungsorientierter Sprachen eingegangen.

B.2.7 DIN EN 62340 „Kernkraftwerke - Leittechnische Systeme mit sicherheitstechnischer Bedeutung - Anforderungen zur Beherrschung von Versagen aufgrund gemeinsamer Ursache“

Um ein hohes Sicherheitsniveau zu erreichen, wird Redundanz als eines der Schlüsselmerkmale für die Auslegung sicherheitstechnisch wichtiger leittechnischer Systeme angewendet. Ein Versagen aufgrund gemeinsamer Ursache kann die Wirksamkeit von Redundanz aufheben. Daher müssen entsprechende Maßnahmen gegen das Eintreten von Gemeinsam Verursachten Ausfällen (GVA) getroffen werden. Die DIN EN 62340 beschäftigt sich mit Anforderungen und Empfehlungen zur Beherrschung von GVAs.

GVAs können eintreten, wenn ein latenter Fehler systematisch in einigen oder allen redundanten Strängen vorhanden ist, durch ein spezifisches Ereignis ausgelöst wird und damit einen koinzidenten Ausfall einiger oder aller Stränge verursacht. Solche latenten Fehler können in jeder Phase des Lebenszyklus eines leittechnischen Systems entstanden sein. Speziell für rechnerbasierte Leittechniksysteme sind darüber hinaus latente Fortpflanzungsmechanismen relevant, die beschädigte Daten von einem fehlerhaften System zu den entsprechenden Systemen anderer Redundanzen übertragen und damit als Folge weitere redundante Stränge ausfallen können.

Grundsätzlich bezieht sich die Norm sowohl auf leittechnische Systeme, die mit festverdrahteter analoger Technik realisiert sind, als auch auf rechnerbasierte Systeme oder Systeme mit Softwarekomponenten. Bedingungen und Anforderungen zur Vermeidung eines GVAs werden daher zunächst allgemein diskutiert. Hinsichtlich latenter Fehler wird dabei insbesondere auf die Auslegungsgrundlagen einschließlich der Anforderungsspezifikationen des leittechnischen Systems eingegangen. Dabei wird u.a. die Realisierung von unabhängigen leittechnischen Systemen zur Beherrschung eines GVAs gefordert.

Im Hinblick auf rechnerbasierte oder programmierbare Komponenten wird sowohl auf prinzipielle GVA-Mechanismen als auch auf Maßnahmen zur Vermeidung spezifischer Fehler und Schwachstellen in rechnerbasierten Systemen und Softwarekomponenten eingegangen. Auch die Toleranz gegen angenommene latente Softwarefehler wird angesprochen.

B.2.8 VdTÜV „Stellungnahme zu den erforderlichen Vorsorgemaßnahmen gegen systematisches Versagen von digitalen leittechnischen

Einrichtungen in kerntechnischen Anlagen, die Leittechnikfunktionen der Kategorie 1 ausführen“

Der VdTÜV hat eine Stellungnahme zu den erforderlichen Vorsorgemaßnahmen gegen systematisches Versagen von digitalen leittechnischen Einrichtungen in kerntechnischen Anlagen, die Leittechnik-Funktionen der Kategorie 1[6] ausführen, abgegeben. Dort werden mögliche fehlerbeherrschende Maßnahmen im Hinblick auf eine dissimilare Auslegung von redundanten Kanälen, Strängen oder Teilsystemen betrachtet. Folgende Aussagen werden getroffen:

Für Ereignisabläufe bei ungestörtem Leistungsbetrieb, die nicht zu Störfällen oder sicherheitstechnisch unzulässigen Komponentenschäden führen und durch Handmaßnahmen von der Warte aus normalisiert werden können, wird für die leittechnischen Einrichtungen der betroffenen Schutzaktionen keine dissimilare Auslegung gefordert.

Für Ereignisabläufe bei Störfällen mit ausschließlich eindeutig sicherheitsgerichteter Auslösung von Schutzaktionen ist eine zweifache dissimilare Auslegung der leittechnischen Einrichtungen, die Leittechnik-Funktionen der Kategorie 1 ausführen, dann erforderlich, wenn ein Störfall oder ein Ereignis bei passivem systematischem Versagen nicht in der Sicherheitsebene 3 beherrscht wird. Für Schutzaktionen, bei denen ein aktives systematisches Versagen von leittechnischen Einrichtungen, die Leittechnikfunktionen der Kategorie 1 ausführen, bei Störfällen und Ereignissen immer eine eindeutig sicherheitsgerichtete Auswirkung hat, darf auf eine dissimilare Auslegung verzichtet werden. Das gilt auch für Schutzaktionen, bei denen ein passives systematisches Versagen durch Handmaßnahmen beherrscht wird. Dabei ist jedoch das 30-Minuten-Konzept zu beachten.

Für Ereignisabläufe bei ungestörtem Leistungsbetrieb oder bei Störfällen, die nicht für jeden Anlagenzustand sicherheitsgerichtete Aktionen beinhalten, ist in den leittechnischen Einrichtungen, die Leittechnik-Funktionen der Kategorie 1 ausführen, eine zweifache oder dreifache dissimilare Auslegung vorzusehen. Eine zweifache dissimilare Auslegung ist ausreichend, wenn die verbleibenden Teilsysteme für die Störfallbeherrschung in der Sicherheitsebene 3 ausreichen oder in hinreichender Zeit

---

[6] Leittechnikfunktionen der Kategorie 1 entsprechen nach den SiAnf heute den Leittechnikfunktionen der Kategorie A.

durch Handmaßnahmen beherrscht werden. Eine dreifache dissimilare Auslegung ist erforderlich, wenn eine Mehrheitsentscheidung automatisch erfolgen muss.

B.2.9 VDI/ VDE 3528 „Anforderungen an Serienprodukte und Kriterien für deren Einsatz in der Sicherheitsleittechnik von Kernkraftwerken“

Die bisher in der Sicherheitsleittechnik eingesetzten Geräte müssen zunehmend durch neue Geräte ersetzt werden. Die am Markt verfügbaren funktional mindestens gleichwertigen Geräte sind in der Regel nicht nach kerntechnischen Gesichtspunkten qualifiziert. Da sie zudem oftmals aus technologisch weit fortgeschrittenen Bauelementen wie Microcontrollern, FPGAs usw. aufgebaut sind, ergeben sich vielfältige Fragestellungen hinsichtlich der Qualifizierung dieser Geräte. Aufgrund der relativ hohen Komplexität moderner Geräte und des relativ niedrigen Stückzahlbedarfs wird zusätzlich zu der etablierten Vorgehensweise, kerntechnisch spezifische Geräte zu entwickeln und zu qualifizieren, der Ansatz verfolgt, den Einsatz geeigneter Serienprodukte in der Kerntechnik zu ermöglichen. Die VDI/VDE-Richtlinie 3528 gibt daher Empfehlungen zu grundsätzlichen Anforderungen an Serienprodukte, die für Funktionen der Sicherheitsleittechnik eingesetzt werden. Dabei werden auch allgemeine Anforderungen an Serienprodukte mit hochintegrierten Bauelementen oder eingebetteter Software sowie zusätzlich zu ergreifende Maßnahmen und Randbedingungen für deren Einsatz in der Sicherheitsleittechnik genannt.

Zunächst wird allgemein auf die Funktionszuverlässigkeit bei der Auslegung von Leittechniksystemen eingegangen. Dazu sind Maßnahmen zur Fehlerbeherrschung und Fehlervermeidung notwendig, welche kurz dargestellt werden. Es wird erläutert, dass Maßnahmen zur Fehlervermeidung oftmals geräte- bzw. komponentenbezogen sind und ein hoher Anteil von Qualitätssicherungsmaßnahmen bereits bei der Entwicklung der Gerätesysteme erbracht werden muss. Maßnahmen zur Fehlerbeherrschung sind dagegen bei der anwendungsspezifischen Auslegung des Systems wesentlich. Beides zusammen muss dann zu einer anforderungsgerechten Funktionszuverlässigkeit des Systems führen.

Um Auslegungsziele leittechnischer Systeme unter Verwendung von Geräten und Gerätesystemen mit unterschiedlichen Qualifizierungstiefen zu erreichen, werden in VDI/ VDE 3528 verschiedene Designvarianten vorgestellt, mit denen durch unterschiedliche

Schwerpunkte bei der leittechnischen Systemkonfiguration eine vergleichbare Zuverlässigkeit für Funktionen einer bestimmten Sicherheitskategorie realisiert werden kann:

- Designvariante 1: Seriengeräte/ Gerätesysteme
- Designvariante 2: Typgeprüfte Geräte/Gerätesysteme
- Designvariante 3: Kerntechnisch typgeprüfte Geräte/ Gerätesysteme

Während in Variante 3 bereits nach KTA und DIN speziell für den Einsatz in einem Sicherheitssystem typgeprüfte Geräte und Gerätesysteme vorgesehen sind, muss bei Variante 2 abschließend eine Eignungsprüfung der typgeprüften Geräte zum Einsatz in einem Sicherheitssystem durchgeführt werden. Gegebenenfalls sind zur Erhöhung der Systemzuverlässigkeit Maßnahmen zur Fehlertoleranz, Unabhängigkeit und Funktionsverteilung notwendig. Beim Einsatz von Seriengeräten in Variante 1 sollen zunächst vom Hersteller belastbare Angaben zur Betriebserfahrung gemacht werden. Geringer belastbare Nachweise des Herstellers zur Fehlervermeidung können durch zusätzliche Maßnahmen bezüglich Fehlertoleranz und Funktionsverteilung auf Systemebene ausgeglichen werden. Darüber hinaus ist ein Qualitätspass für die Betriebszeit des Gerätesystems einzuführen, mit dem die Betriebserfahrung und die Änderungen der Gerätetechnik mitverfolgt und beurteilt werden. Variante 1 ist nach VDI/VDE 3528 nur zur Realisierung von Funktionen der Kategorie B und C zulässig.

Nachfolgend wird auf die Qualifizierung von Systemen eingegangen, die Funktionen der Kategorie A bis C ausführen sollen. Dabei wird u.a. festgestellt, dass dissimilare Komponenten mit einem niederen Qualifizierungslevel in einer höherwertigen Struktur eingesetzt werden können, falls Geräte oder Gerätesysteme nicht mit der erforderlichen Qualifizierung bereitgestellt werden können. Durch den alternativen Einsatz dissimilarer Komponenten niedrigerer Qualifizierung muss sich jedoch eine äquivalente Versagenssicherheit ergeben.

Abschließend wird eine Übersicht über Qualitäts- und Auslegungsmerkmale gegeben, wobei teilweise auf die verschiedenen Designvarianten getrennt eingegangen wird. Dabei wird unter anderem auf die Vorkehrungen gegen das systematische Versagen von Produkten mit integrierter Software eingegangen.

B.2.10 IAEA NS-G-1.3 “Instrumentation and Control Systems Important to Safety in Nuclear Power Plants”

Das Ziel dieses Standards ist die Bereitstellung von Richtlinien für die Auslegung sämtlicher Einrichtungen, die für sicherheitstechnisch wichtige Leittechnikfunktionen im Kernkraftwerk notwendig sind. Dies beinhaltet auch rechnerbasierte oder programmierbare leittechnische Einrichtungen.

Bevor Anforderungen an ein Leittechniksystem formuliert werden können, sind gemäß NS-G-1.3 erst eine genaue Charakterisierung der jeweiligen Aufgaben und Funktionen des Leittechniksystems sowie eine Klassifizierung des Systems notwendig. Jedes leittechnische System soll demnach so konzipiert sein, dass seine Spezifikation, Verifizierung, Validierung, Qualitätssicherung, Qualitätskontrolle und Zuverlässigkeit seiner Klassifizierung entspricht. Die Klassifizierung soll die sicherheitstechnische Relevanz des Systems kategorisieren und somit seine Anforderungen definieren. Bei der Charakterisierung ist darauf zu achten, dass nicht ausschließlich die Ursprungsaufgabe eines Teilsystems, sondern auch Korrelationsaspekte zu anderen Systemen, berücksichtigt werden.

Aspekte, die konkret bei der Klassifizierung von rechnerbasierter oder programmierbarer sicherheitsrelevanter Leittechnik berücksichtigt werden sollen, sind separat aufgeführt.

Im weiteren Verlauf werden konkrete Schlüsseleigenschaften der rechnerbasierten oder programmierbaren sicherheitsrelevanten Leittechnik identifiziert und entsprechende Richtlinien und Anforderungen formuliert. Dabei wird primär zwischen den Anforderungen an Sicherheitssysteme und Systeme lediglich mit Bezug zu Sicherheitsaspekten unterschieden. Insbesondere wird dabei die Toleranz gegen Zufallsfehler sowie gemeinsam verursachte Fehler hervorgehoben. Auch die Unabhängigkeit und Qualität der Einrichtungen sowie die Ermöglichung von Prüfungen und Wartungen wird diskutiert.

Wenn der notwendige Beweis der Zuverlässigkeit des Systems nicht durchführbar ist, sollten zusätzliche konservative Systeme zur Anwendung kommen. Bei dem Nachweis der Zuverlässigkeit eines rechnerbasierten oder programmierbaren Systems können beispielsweise besondere Probleme auftreten. Die Nutzung von Diversität ist eine Möglichkeit Konservatismus anzuwenden, um die Schwierigkeiten bei dem Nachweis des notwendigen Grades der Zuverlässigkeit zu kompensieren.

Gemäß NS-G-1.3 bietet Diversität Schutz gegen gemeinsam verursachte Fehler, ergänzt das Defence-in-Depth-Prinzip und erhöht die Wahrscheinlichkeit, dass sicherheitstechnisch wichtige Aufgaben bei Anforderung ausgeführt werden. Genannte Arten von Diversität sind menschliche Diversität, Design-Diversität, Software-Diversität, funktionale Diversität, Signal-Diversität, Diversität der Ausrüstung und System-Diversität. Es wird explizit darauf hingewiesen, dass auch eingesetzte Software der Diversität unterliegen soll. Ferner wird diskutiert, dass verschiedene Versionen von Software, die unter denselben Sicherheitsanforderungen und Spezifikationen entworfen wurden, nicht als diversitär bezeichnet werden können. Gemäß NS-G-1.3 können erst funktionelle Diversität sowie unabhängige Eingangssignale Diversität garantieren. Für die Einschätzung der Zuverlässigkeit von Software wird eine qualitative Bewertung anhand der Komplexität des Systems sowie der Verifizierung und Validierung vorgeschlagen.

Ergänzend dazu wird der Einsatz von Software im Reaktorschutz diskutiert. Es wird darauf hingewiesen, dass die verwendete Software grundsätzlich auf die Hardware abgestimmt sein soll. Des Weiteren soll beim Einsatz in mehreren Anwendungen der Sicherheitsstandard stets der höchsten Qualifizierung entsprechen. Die Entwicklung der eingesetzten Software soll einem kontrollierten Prozess unterliegen und ihr Einsatz vollständig dokumentiert werden.

Im weiteren Verlauf werden ergänzend systemspezifische Richtlinien vorgestellt. Hierbei liegt der Fokus im speziellen auf dem Reaktorschutz. Dabei wird auch die Interaktion zwischen Reaktorschutz und anderen Systemen aufgegriffen. Insbesondere soll bei der Interaktion mit Systemen eines geringeren Sicherheitsstandards auf eine zuverlässige Entkopplung geachtet werden. Auch die Interaktion zwischen Mensch und Technik wird in diesem Standard eingehend diskutiert.

Im letzten Teil des Standards werden die einzelnen Schritte der Entwicklung eines sicherheitsrelevanten leittechnischen Systems vorgestellt. Dabei gelten die Ausführungen des Standards gleichermaßen für konventionelle und softwarebasierte Leittechnik. Die Entwicklung des Systems soll auf einem kontrollierten, iterativen, kleinschrittigen Verfahren basieren. Nach jedem Teilschritt sind die Verifizierung des Systems gegenüber seinen Anforderungen sowie eine regelmäßige Validierung gefordert. Verifizierung und Validierung sollen hierbei nicht von den Entwicklern selbst vorgenommen werden. Aspekte der Instandhaltung und Aktualisierung sowie der Dokumentation werden ebenfalls aufgegriffen.

B.2.11 IAEA NS-G-1.1 “Software for Computer Based Systems Important to Safety in Nuclear Power Plants”

NS-G-1.1 befasst sich mit den Anforderungen an rechnerbasierte oder programmierbare sicherheitsrelevante Leittechnik. Dieser Standard soll einen Leitfaden für den Einsatz von Software in sicherheitstechnisch wichtigen, rechnerbasierten oder programmierbaren leittechnischen Einrichtungen im Kernkraftwerk bieten.

Zu Beginn wird darauf hingewiesen, dass Fehler in rechnerbasierten oder programmierbaren Systemen meist systematischer Natur sind und somit Redundanzen unter Verwendung der gleichen Software nicht als diversitär betrachtet werden können. Lediglich der Versuch, Diversität in der Methode, der Programmiersprache, den Entwicklungswerkzeugen und den programmierenden Personen zu fordern, kann eine Art von Diversität erfüllen. Ausschließlich funktionelle Diversität stellt eine zuverlässige Form der Diversität dar. Diversität soll möglichst früh in die verschiedenen Untersysteme eingebaut werden, um sie auf diesem Weg zu maximieren.

Der Standard weist ausdrücklich darauf hin, dass sich die Erbringung eines Nachweises der Systemzuverlässigkeit schwierig gestaltet, da eine quantitative Einschätzung eines rechnerbasierten oder programmierbaren Systems heutzutage nicht vollständig möglich ist. Es kann hierbei lediglich auf qualitative Methoden zurückgegriffen werden.

Es werden speziell Anforderungen und Richtlinien an Computersysteme formuliert, da diese auch die Validierung des Gesamtsystems beeinflussen. Dazu wird ausgeführt, dass insbesondere das Computersystem ein potentieller Auslöser für Fehler mit gemeinsamer Ursache ist. Es werden daher konkrete Anforderungen an das Interface sowie funktionelle und nicht-funktionelle Anforderungen formuliert. Dabei wird insbesondere darauf hingewiesen, dass bei der Entwicklung des Computersystems auf eine Kompatibilität zwischen Hardware, Software und Firmware geachtet werden muss.

Um eine wohl strukturierte und gut kontrollierte Entwicklung für rechnerbasierte oder programmierbare Leittechnik sicherzustellen, werden in NS-G-1.1 die einzelnen Entwicklungsschritte vorgestellt und entsprechende Anforderungen formuliert. Dabei wird u.a. nach jedem Entwicklungsschritt eine Verifizierung gegenüber den Anforderungen an das System sowie eine regelmäßige Validierung gefordert. Zum Teil können Anforderungen aus der konventionellen Leittechnik übernommen werden. Es müssen jedoch

auch einige neue Gesichtspunkte berücksichtigt werden. Es wird daher auf die Anforderungen an den Entwurf, die Entwicklung, die Qualitätssicherung und die Dokumentation eingegangen. Dabei wird beispielsweise für die Entwicklung und für die spätere Konfiguration eine eindeutige Versionskontrolle sowie eine lückenlose Dokumentation und Archivierung sämtlicher Versionen gefordert. Konkret wird zudem die Trennung von sicherheitstechnisch wichtigen und betrieblichen Aspekten genannt.

Im weiteren Verlauf werden Empfehlungen für den Prozess des Software-Entwurfs sowie der Umsetzung der Software gegeben. Es werden einige Begrifflichkeiten, die speziell die Entwicklung von Software betreffen, neu eingeführt. So wird zum Beispiel die Verwendung einer Programmiersprache mit gründlich definierter Syntax empfohlen und bei der Verwendung von Hilfsmitteln (Codegenerator, Übersetzer etc.) ist auf die Validierung eben dieser zu achten. Auch muss die Unabhängigkeit von sicherheitstechnisch wichtigen und betrieblichen Aspekten aufgrund von gemeinsamen ausführenden Systemen gewährleistet werden.

Anschließend werden Empfehlungen für die Verifizierung und Analyse der Software präsentiert. Es wird darauf hingewiesen, dass keine der zur Verifizierung und Analyse zur Verfügung stehenden Methoden eine vollständige Überprüfung leisten kann. Vollständige Sicherheit kann ausschließlich nach Verwendung mehrere Techniken garantiert werden. Die Verifizierung soll ausschließlich von entsprechend qualifiziertem Personal an der endgültigen Version durchgeführt werden.

Bei der Prüfung der Software sind sämtliche Programmbestandteile bis hin zur Hardware und der Ausführung der Software einzuschließen. Besonderes Augenmerk soll auf die Schnittstellen zwischen einzelnen Bestandteilen, die ausführenden Hilfsprogramme sowie die Eingangsvariablen gelegt werden. Die Software soll vollständig und nachvollziehbar getestet werden. Tests sollen wiederholbar und unter möglichst geringem menschlichem Einfluss durchgeführt werden. Die Testergebnisse sollen vollständig dokumentiert sein.

Im Anschluss beschäftigt sich der Standard mit der Integration der Software, die aus dem Hochladen der Software sowie der Überprüfung der Software-Hardware-Schnittstelle und Betrieb der Software besteht. Es wird darauf hingewiesen, dass eine Validierung des endgültigen Programms notwendig ist, um zu zeigen, dass das Programm die ihm zugedachte Funktion erfüllt. Von der Verwendung von Inter- oder Extrapolationen

zur Reduzierung der durchzuführenden Tests wird aufgrund der digitalen Natur des Programms abgeraten. Der Standard erlaubt für die Validierung jedoch stichprobenartige Tests, wenn die Anzahl der durchgeführten Tests ausreichend groß ist, den gesamten Einsatzbereich abdeckt und die Tests repräsentativ sind.

Abschließend werden die Installation und Übergabe, die Ausführung und die nachträgliche Modifikation des Programms behandelt. Der letzte Schritt der Verifizierung und Validierung ist die Überprüfung der Erfüllung der Sicherheitsanforderungen des Kraftwerks. Ist das System in Betrieb, soll für eine Instandhaltung und die Einhaltung der angenommenen Betriebsbedingungen Sorge getragen werden.

B.2.12 IAEA Technical Report No. 367 “Software Important to Safety in Nuclear Power Plants”

Die in NS-G-1.1 zunächst allgemein formulierten Anforderungen werden in diesem Bericht detaillierter spezifiziert. Hierzu beginnt der Bericht mit einer Beschreibung der aktuellen Praxis der Softwareentwicklung für Systeme der nuklearen Sicherheit.

Zu Beginn werden in diesem Bericht zunächst die Vor- und Nachteile von Software diskutiert und die Ziele und Methoden der Software-Entwicklung sowie deren Umsetzung vorgestellt. Hierbei wird kurz auf den Lebenszyklus von Software sowie das Management der Software-Entwicklung eingegangen.

Im weiteren Verlauf beschäftigt sich der Bericht mit den einzelnen Schritten der Softwareentwicklung, beginnend mit der Modellierung für die Analyse der Anforderungen und den Entwurf für die Software. Basierend auf den Modellierungen werden die Anforderungen an den Entwurf formuliert. Es wird dabei kurz vorgestellt, welche Aspekte simuliert werden sollten und welche Anforderungen die Modelle erfüllen sollten. Der Bericht gibt zu bedenken, dass die Entwicklung einer Überprüfungsmöglichkeit für die Modellierung und die gestellten Anforderungen bisher noch sehr theoretisch ist und es fraglich ist, wann dies sinnvoll und zuverlässig für große Systeme möglich ist.

Anschließend beschäftigt sich der Bericht mit den Schritten der Dokumentation, der Programmierung, der Verifizierung und Validierung sowie der Überprüfung der Software. Im Bereich der Dokumentation verweist der Bericht auf Erfahrungen aus Kanada /ONT 99/ und aus der Luftfahrtechnik. Es wird darauf hingewiesen, dass von der Präzision der

Dokumentation letztlich die Qualität der Software abhängt. In Bezug auf die Programmierung stellt der Bericht einige Anforderungen an die Programmiersprache und rät sogar konkret von einigen Programmiersprachen ab, da diese zu viele Unsicherheiten aufweisen. Als wesentlicher Aspekt wird die Programmierung eines möglichst simplen, wohl strukturierten und modular aufgebauten Codes gefordert.

Zu Verifizierung und Validierung werden konkrete Anforderungen formuliert und erläutert sowie verschiedene Methoden und ihre Überprüfung diskutiert. Vor allem zur Prüfung von Software wird klargestellt, dass die Tests die in der Software enthaltenen Fehler finden und nicht ihre Korrektheit zeigen sollen. Es werden verschiedene Typen, Ebenen, Näherungen sowie Absichten von Software-Tests vorgestellt und deren Begrifflichkeiten erläutert. Einige gängige Methoden werden im Detail vorgestellt. Dabei wird erläutert, dass es für ein erfolgreiches Prüfen von Software wichtig ist, dass Methoden kombiniert werden, die sich in ihren Nachteilen kompensieren. Darüber hinaus muss für eine erfolgreiche Prüfung eine Spezifikation der Software vorliegen. Letztendlich weist der Bericht darauf hin, dass das Prüfen der Software als Ergänzung zur Verifizierung und Validierung zu betrachten ist und diese nicht ersetzen kann.

Weitergehend wird die Einführung eines quantitativen Gütefaktors (Figure of Merit) vorgestellt. Der Gütefaktor soll eine objektive Bewertung der Qualität und Zuverlässigkeit der Software ermöglichen. Beim Betrieb von Hardwarekomponenten geht man üblicherweise davon aus, dass nach einer eingehenden Überprüfung die Hardware zuverlässig arbeitet und lediglich kleine Zufallsfehler auftreten. Auch erwartet man hierbei eine hohe Toleranz der Komponente gegen „kleinere“ Fehler. Diese Annahmen können für Softwarekomponenten nicht übertragen werden, da diese bereits aufgrund eines einzelnen, geringfügigen Designfehlers vollständig versagen kann.

Im weiteren Verlauf beschäftigt sich der Bericht mit der Wartung, Aktualisierung und nachträglichen Ergänzung von Software. Er verweist explizit darauf, dass diese Aspekte bereits im Entwurf der Software berücksichtigt werden sollten. Auch wird darauf hingewiesen, die erneute Überprüfung veränderter Systeme zu bedenken. Außerdem wird eine schrittweise Anleitung zur Aktualisierung von Software vorgestellt.

Es wird weiterhin diskutiert, dass unterstützende Programme und der Einsatz von Werkzeugen zur Softwareentwicklung zwar den Vorteil haben, dass sie einen strukturierteren, wohl dokumentierten Code hervorbringen und das Risiko von menschlichem Versagen

senken, die Nachteile jedoch nicht zu vernachlässigen sind. Das primäre Problem hierbei ist die Überprüfung und das Garantieren der Zuverlässigkeit dieser Programme. Der Bericht kommt zu dem Schluss, dass eine zuverlässige Verwendung solcher Programme fraglich bleibt.

Auch die Verwendung von bereits existierender Software wird kurz aufgegriffen. Es wird darauf hingewiesen, dass für die Verwendung bereits existierender Software eine vollständige Spezifikation, Analyse, Verifizierung und Überprüfung durchgeführt werden muss. Außerdem muss die Möglichkeit der vollständigen Einsicht der Entwicklungsunterlagen gegeben sein.

Im weiteren Verlauf beschäftigt sich der Bericht mit den Anforderungen an die Qualifikation der Entwickler. Dabei wird ein einheitliches Schulungs- und Qualifizierungsprogramm vorgeschlagen.

Abschließend wird die Balance von softwarebasierten Systemen diskutiert. Es wird darauf hingewiesen, dass nur eine gute Balance zwischen verschiedenen Aspekten wie z.B. Kosten, Zeitplan, Zuverlässigkeit, Komplexität und Sicherheit am Ende in der Entwicklung eines sicheren Systems resultieren kann, wobei letztlich sämtliche Probleme auf eine Balance zwischen Sicherheit und Kosten reduziert werden können. Hierbei ist insbesondere die Abwägung des Aufwandes für die Verifizierung, Validierung und Fehleranalyse von enormer Wichtigkeit, da hiervon entscheidend die Zuverlässigkeit der Software abhängt. Als eine Möglichkeit, dieses Problem zu beherrschen, wird die Kontrolle durch Gutachter angeführt.

B.2.13 IAEA Technical Report No. 384 “Verification and Validation of Software Related to the Safety of Nuclear Power Plant Instrumentation and Control”

Dieser Bericht beschäftigt sich mit der Verifizierung und Validierung von Software für die Anwendung in sicherheitstechnisch wichtigen Systemen in Kernkraftwerken. Dabei behandelt der Bericht die Frage nach möglichen Methoden der Verifizierung und Validierung und ihrer Effektivität.

Zu Beginn des Berichts wird die Notwendigkeit der Verifizierung und Validierung diskutiert. Dazu wird zunächst dargestellt, dass eine erfolgreiche Verifizierung und Validierung dann erreicht ist, wenn die Anzahl der Softwarefehler bis auf ein akzeptables Minimum

reduziert wurde. Hierbei sind zwei verschiedene Arten von Fehlern relevant: Fehler, die auf einer falsch formulierten Anforderung, jedoch mit korrekter Einbindung, und Fehler bei der Einbindung von Anforderungen auftreten.

Im weiteren Verlauf werden verschiedene Arten von Software sowie deren Klassifizierung vorgestellt. Für ein besseres Verständnis der verschiedenen Arten von Software erläutert der Bericht die Unterschiede von neu entwickelter Software, bestehender Software aus ähnlicher Anwendung, bestehender Software aus einer anderen Anwendung und konfigurierter Software.

Weiterhin präsentiert der Bericht die Aufgaben und Notwendigkeiten der verschiedenen Entwicklungsschritte. Hierzu zählen die Spezifikation der Systemanforderungen und des Computersystems, der Software-Entwurf sowie die Integration, Validierung, Abnahme, Betrieb, Wartung und Modifikation von Software. Anschließend wird die Verifizierung der einzelnen Entwicklungsphasen eingehend beschrieben. Jede Phase soll hinsichtlich ihrer geforderten Eingangsdaten, den spezifischen Ausführungen und den erwarteten Ausgangsdaten geprüft werden. Ziel der Verifizierung ist dann, für jede einzelne Phase zu bestätigen, dass das entwickelte Produkt den Anforderungen entspricht. Neben konkreten Empfehlungen für eine erfolgreiche Umsetzung werden auch konkrete Techniken für die Verifizierung und Validierung von Software vorgestellt. Auch wird kurz auf die Verifizierung von bereits bestehender Software sowie die abgeschwächten Anforderungen an die Verifizierung von Software in Leittechniksystemen, die Funktionen der Kategorie B und C ausführen, eingegangen.

Bei der Validierung der Software muss demonstriert werden, dass das System seine Funktion entsprechend den Spezifikationsanforderungen korrekt ausführt. Sind diese jedoch bereits fehlerhaft, verhindert auch eine erfolgreiche Validierung nicht das Versagen des Systems. Die Systemvalidierung wird nach der Integration der Software, jedoch vor deren Inbetriebnahme, durchgeführt und sollte alle Bestandteile des Gesamtsystems (z.B. auch Sensoren etc.) einschließen. Der Bericht weist weiterhin darauf hin, dass nicht alle Anforderungen, die an die Validierung für Software der Kategorie A gestellt werden, auch für Kategorie B und C erfüllt werden müssen. Dennoch sollte auch bei Software der Kategorie B und C demonstriert werden, dass die an das System gestellten Anforderungen erfüllt sind.

Abschließend beschäftigt sich der Bericht mit der Lizensierung von Software. Von den jeweiligen Mitgliedsländern wird demnach erwartet, dass sie eine Regelung für die Lizensierung finden, die mit den jeweiligen Gesetzen des Landes kompatibel ist. Hierbei kann der Umfang einer Lizensierung mit den unterschiedlichen Gesetzen und Vorgaben eines Landes sowie der Kategorisierung der Software stark variieren. Generell wird darauf verwiesen, dass für eine Lizensierung eine Reihe von Sicherheitsdokumenten angefertigt werden müssen, die die Zuverlässigkeit eines Systems rechtfertigen. Insbesondere muss der Nachweis einer erfolgreichen Verifizierung und Validierung geliefert werden. Auch weist der Bericht darauf hin, dass zur Durchführung einer Lizensierung die entsprechenden Kompetenzen vorhanden sein müssen.

B.2.14 IAEA NP-T-1.5 “Protection against Common Cause Failures in Digital I&C Systems of Nuclear Power Plants”

Da Software niemals als hundertprozentig fehlerfrei angesehen werden kann, besteht grundsätzlich immer ein Risiko von Ausfällen mit gemeinsamer Ursache. Aus diesem Grund diskutiert dieser Standard mögliche Quellen für GVAs in digitalen Leittechniksystemen und beschreibt Maßnahmen, mit denen das Auftreten von GVAs minimiert werden kann.

Die Minimierung von Fehlern erfolgt demnach durch die Vermeidung von Fehlern während der Planung und Entwicklung sowie durch die Fehlererkennung und Fehlerentfernung während der Verifizierung und Validierung im Entwicklungsprozess.

Trotz der getroffenen Maßnahmen zur Beseitigung von Fehlern in leittechnischen Systemen wird unterstellt, dass einige Fehler nicht entdeckt werden und verbleiben. Für angeblich voneinander unabhängige Systeme ist sicherzustellen, dass systematische Fehler nicht existieren oder nicht zur gleichen Zeit in allen Systemen ausgelöst werden können. Diversität ist das bevorzugte Mittel, mit dem dies erreicht wird. Dazu werden u.a. menschliche Diversität, funktionale Diversität und Design-Diversität als Diversitätsmerkmale genannt.

Zusammenfassend wird ausgesagt, dass eine einzelne Art von Diversität hilfreich ist, aber üblicherweise die Vermeidung systematischer Fehler nicht garantiert. Der Gebrauch mehrerer Arten von Diversität könnte laut Aussage des Standards die wichtigste Vorgehensweise sein, mit dieser Einschränkung umzugehen.

B.2.15 CE-1001-STD „Standard for Software Engineering of Safety Critical Software”

In Kanada wird die Software-Entwicklung für sicherheitstechnisch wichtige Software unter anderem mit Hilfe des Standards CE-1001-STD reglementiert. Dieser wurde vom kanadischen Staatsunternehmen AECL, das auch die CANDU®-Reaktoren entwickelt hat, ausgearbeitet. Die Auflagen werden als „weiche Forderungen“ (shall-requests) formuliert und ihre Kontrolle wird nicht explizit gefordert.

Der Standard dient der Festlegung von Anforderungen an die Entwicklung von Software, die in Leittechniksystemen, die Funktionen der Kategorie I[7] ausführen, eingesetzt wird. Dazu wird die Entwicklung ausschließlich für textbasierte Programmiersprachen und Neuentwicklungen für sicherheitstechnisch wichtige Anwendungen im Bereich des Echtzeitschutzes, der Kontrolle und des Monitorings in Kernkraftwerken behandelt.

Dieser Standard definiert zunächst einen minimalen Satz an Prozessen, die für die Entwicklung von sicherheitstechnisch wichtiger Software, welche in der Kategorie I zum Einsatz kommen soll, notwendig sind. Der wesentliche Schritt der Entwicklung ist nach CE-1001-STD die Festlegung der Anforderungen an die Software. Sie werden im „Design Input Document“ (DID), das die Basis für den Entwicklungsprozess darstellt, spezifiziert und gerechtfertigt. Es wird darauf hingewiesen, dass auch die Umgebungsbedingungen, in denen die Software nachher betrieben werden soll, festgelegt werden sollen. Außerdem sollen bereits in der Entwicklung potentielle Gefahren- und Fehlerquellen identifiziert und diskutiert sowie die Toleranz gegenüber fehlerhaften Eingangssignalen festgelegt werden.

Bei der Entwicklung des Designs wird insbesondere auf die Selbstkontrolle des Systems Wert gelegt. Darüber hinaus soll auf einen logischen und nachvollziehbaren Strukturaufbau sowie eine eindeutige Namensgebung geachtet werden. Die Einbeziehung verschiedener Personen in die einzelnen Entwicklungsstufen soll eine Form der Diversität erfüllen.

[7] Diese Kategorie entspricht der deutschen Definition der Kategorie A.

Um die Zuverlässigkeit des eingesetzten Codes zu garantieren, werden verschiedene Anforderungen formuliert. Dazu zählt u.a., dass keine Rekursion und keine Selbstmodifikation eingebaut werden darf und dass eine eindeutige Definition und Beschreibung des Codes im Klartext sowie eine Dokumentation und Archivierung aller Code-Änderungen vorliegen muss. Darüber hinaus ist auf die Erfüllung aller im DID definierten Anforderungen zu achten. Außerdem sollen schlecht zu verifizierende Programmteile von Anfang an vermieden werden.

Im weiteren Verlauf beschäftigt sich der Standard mit der Verifizierung und Validierung der Software. Zu den einzelnen Schritten gehören hier die Überprüfung, systematische Verifizierung, eine Testphase und Fehleranalyse sowie die Qualifizierung der Software im Hinblick auf die Zuverlässigkeit. Dabei ist insbesondere darauf zu achten, dass alle Komponenten der Software (Module, Programme, Datenstrukturen, Datenbanken etc.) in die Kontrolle eingeschlossen werden. Im Anschluss an die Verifizierung soll das Programm in einer Testphase auf mögliche Fehler oder Unstimmigkeiten in Bezug auf die Anforderungen getestet werden. In dieser Testphase soll zunächst jede Programmeinheit für sich auf Funktionalität, dann die Integration der Programmeinheiten und schließlich die Kompatibilität von Hardware- und anderen Software-Komponenten überprüft werden. Anschließend soll eine Validierung der Software durchgeführt werden, um zu überprüfen, ob die ausgeführte Software den Einsatzbedingungen und Einsatzmöglichkeiten des Anwenders entspricht. Dabei sollen verschiedene Szenarien mit dem Endverbraucher getestet werden. Die Software soll dazu auf der Hardware des Endverbrauchers unter den endgültigen Bedingungen validiert werden. Im letzten Teil der Verifizierung soll unter den endgültigen Bedingungen eine konkrete Fehleranalyse durchgeführt werden. In diesem Schritt soll insbesondere auf die Identifizierung und Überprüfung von Sicherheitssystemen innerhalb der Software geachtet werden. Ein besonderes Augenmerk soll hier auf sich-selbst-kontrollierende Software-Komponenten gelegt werden. Auch die Überprüfung der verschiedenen Eingänge (RAM, ROM etc.) soll im Fokus stehen.

Gemäß CE-1001-STD erhöht die genaue Definition von Zuständigkeiten und Verantwortlichkeiten bereits in der Planung der Software-Entwicklung die Sicherheit und Zuverlässigkeit des Endprodukts. Eine Festlegung von unterschiedlichen Zuständigkeiten für einzelne Bereiche soll die Qualitätssicherung darstellen. Simulationen und Modellie-

rungen sollen bereits vorab eine Einschätzung der Zuverlässigkeit, Funktionalität, Sicherheit etc. liefern. Wichtig ist, dass bereits zu diesem Zeitpunkt festgelegt wird, dass die Software über die gesamte Zeit ihrer Lebensdauer zuverlässig arbeiten muss.

Die wesentlichsten Anforderungen an die Umsetzung von Konfigurationen sind die Verifizierung sowie die Dokumentation und Archivierung einer jeden Konfiguration. Es wird jedoch darauf hingewiesen, dass insbesondere Wartungsarbeiten und Konfigurationen ein großes Gefahrenpotential für die Sicherheit von softwarebasierter Leittechnik darstellen. Deswegen wird gefordert, dass Konfigurationsanfragen periodisch geprüft, verifiziert und klassifiziert werden müssen, um zu entscheiden, ob sie gerechtfertigt und umsetzbar sind.

Abschließend werden die Anforderungen an die Vor- und Ausbildung des Endverbrauchers, d.h. des Nutzers oder Betreibers, formuliert. Es wird eine Vorqualifikation des Anwenders in Bezug auf die Software-Anwendung, wie auch den entsprechenden Einsatzbereich gefordert. Darüber hinaus sollen Schulungs- und Ausbildungsmaßnahmen definiert und angeboten werden. Dies gilt sowohl für die Einführung der Software, als auch für jegliche Konfiguration. Ausbildungserfolge und Qualifikationen sollen mit angemessenen Nachweisen überprüft werden.

B.2.16 NRC NUREG 800, Appendix 7.0-A “Review Process for Digital Instrumentation and Control Systems”

Anhang 7.0-A des NUREG 800 liefert eine Übersicht über den Prozess der Bewertung und Überprüfung von digitalen Leittechniksystemen. Als wesentliche Aspekte werden die Qualifizierung von digitalen Leittechniksystemen und der Schutz gegen Fehler mit gemeinsamer Ursache aufgegriffen.

Es wird darauf hingewiesen, dass eine Qualifizierung von digitalen Leittechniksystemen und Software deutlich schwieriger ist als die Qualifizierung von konventioneller Leittechnik. Inspektion und Überprüfung verifizieren zwar eine korrekte Umsetzung und validieren die korrekte Funktion, können jedoch ein diskontinuierliches Versagen nicht ausschließen. Somit ist für die Qualifizierung von digitalen Leittechniksystemen die Bestätigung, dass eine qualitativ hochwertige Prozessentwicklung sowie eine eingehende Spezifizierung und Umsetzung von Anforderungen an den Entwurf erfolgt ist, unerlässlich.

Es werden neben der Qualitätssicherung ebenfalls Diversität und ein gestaffeltes Sicherheitskonzept (Defence in depth) zur Vorbeugung von Fehlern gemeinsamer Ursache gefordert. Die Integration einer digitalen Komponente in ein konventionelles System ist demnach noch leicht auf Diversität und ein gestaffeltes Sicherheitskonzept zu überprüfen. Bei einem Austausch des gesamten Kontroll- und Sicherheitssystems gegen rechnerbasierte Lösungen wird dies jedoch problematisch.

Falls ein unterstellter systematischer Fehler eine Sicherheitsfunktion blockieren kann, wird im Hinblick auf Diversität gefordert, dass ein diversitäres Mittel verwendet werden sollte, welches entweder die gleiche (wie das für den systematischen Fehler anfällige Sicherheitssystem) oder eine andere Funktion (welche einen angemessenen Schutz liefert) ausführt. Dabei muss nachgewiesen werden, dass es unwahrscheinlich ist, dass das diversitäre Mittel von dem gleichen systematischen Fehler betroffen wird. Die diversitäre Funktion kann von einem betrieblichen System ausgeführt werden, falls dieses System eine ausreichende Güte hat, um die notwendigen Funktionen unter den bei dem Ereignis auftretenden Bedingungen auszuführen.

Es wird der formelle Ablauf einer Überprüfung von rechnerbasierten Leittechniksystemen schematisch dargestellt. Hierbei werden für die Überprüfung sowohl Anforderungen an die Funktion als auch an die Entwicklung der Software formuliert, wobei beide Arten von Anforderungen im Entwicklungsprozess zusammengeführt werden.

Wurde die rechnerbasierte Leittechnik entsprechend diesen Vorgaben überprüft und validiert, folgt im Anschluss die Integration von Software und Hardware. Letztendlich wird dann das Gesamtsystem gegen die Systemanforderungen validiert. Hierbei müssen die Anforderungen mit 10 CFR Teil 50 sowie dem Regulatory Guide 1.152 /NRC 11/ übereinstimmen. Es wird darauf hingewiesen, dass eine Dokumentation der Überprüfung notwendig ist. Die Ausführlichkeit der Dokumentation ist hierbei abhängig von der Komplexität und Sicherheitsrelevanz des Systems.

Es wird weiterführend darauf hingewiesen, dass auch Fehler in der Spezifikation der Anforderungen zu einem Versagen eines rechnerbasierten Systems führen können. Darüber hinaus wird zur Formulierung und Überprüfung der Anforderungen die Verwendung von Programmiersprachen oder Analyseprogrammen, die z.B. Blockdiagramme oder ähnliches erstellen, empfohlen, da dies die Vermeidung von Mehrdeutigkeiten, Unvollständigkeiten oder Inkonsistenzen in den Anforderungen vermeidet. Dabei können je-

doch nicht alle Sprachen und Programme alle Anforderungen erfassen. Somit muss darauf geachtet werden, dass alle Anforderungen vollständig abgedeckt werden. Eine Verwendung mehrerer Sprachen und Programme ist somit ratsam. Letztendlich muss noch bedacht werden, dass eine eventuelle Transformation von Daten für die Vorbereitung solcher Programme ebenfalls verifiziert werden soll.

Abschließend wird die Bewertung und Überprüfung von kommerzieller Software behandelt. Es wird jedoch darauf hingewiesen, dass für eine kommerzielle Zulassung eines Systems die Vorgaben aus 10 CFR Teil 50 berücksichtigt werden sollten.

B.2.17 NRC Guide 1.152 "Criteria for Use of Computers[8] in Safety Systems of Nuclear Power Plants"

Dieser Leitfaden beschreibt eine Methode zur Umsetzung der gesetzlichen Vorgaben des 10 CFR Teil 50 (und 52) für rechnerbasierte und programmierbare Leittechnik. Eine Abweichung von hier vorgeschlagenen Vorgehensweisen ist möglich, solange die Erfüllung der Anforderungen zuverlässig nachgewiesen wird.

Im ersten Teil des Leitfadens werden die Anforderungen aus den gesetzlichen Vorgaben des 10 CFR sowie der NRC und der IEEE zusammengefasst.

Anschließend wird die Problematik von Softwarefehlern, die in redundanten Systemen zu einem Fehler mit gemeinsamer Ursache führen können, diskutiert. Dabei wird erläutert, dass nur eine funktionelle und operative Diversität, Diversität im Entwurf der Software sowie Diversität innerhalb des gestaffelten Sicherheitskonzepts einem Fehler mit gemeinsamer Ursache vorbeugen können. Manuelle Betätigungen von sicherheitstechnisch wichtigen und betrieblichen Systemen sind dabei akzeptabel, sofern die notwendigen diversitären Bedienelemente und Anzeigen die benötigte Funktion unter den bei dem Ereignis auftretenden Bedingungen innerhalb einer akzeptablen Zeit ausführen können.

Ebenfalls gibt der Leitfaden zu bedenken, dass unterschiedliche Geräte benutzt werden sollen. Da viele Hersteller identische Geräte (z.B. Prozessoren) verwenden, muss für

[8] Der Begriff „Computer" umfasst hier ein System, dass Computer, Hardware, Software, Firmware und Benutzeroberfläche einschließt.

eine Garantie der Diversität das gesamte System hinterfragt werden. Diversität allein über verschiedene Entwickler zu erreichen, ist kritisch zu betrachten. Es wird ebenfalls darauf hingewiesen, dass verschiedene Softwareversionen, die aus den gleichen Anforderungen entwickelt wurden, ebenfalls nicht als diversitär betrachtet werden können. Erst eine Diversität in der Funktionsweise, den Eingangssignalen und den daraus resultierenden Anforderungen stellen eine gute Basis für eine diversitäre Software dar.

Im weiteren Verlauf geht der Standard auf den Lebenszyklus von Software-Entwicklung und -Betrieb ein, wobei der Fokus hier auf dem Aspekt der Software-Sicherung liegt. Für eine zuverlässige Sicherung der Software und der Hardware sollen zu Beginn der Software-Entwicklung sämtliche potentiellen Gefährdungen identifiziert und diskutiert werden. Darauf aufbauend sollen dann die Anforderungen für eine sichere Entwicklung sowie einen zuverlässigen Betrieb definiert werden. Als wesentliche Punkte der Software-Sicherung nennt der Leitfaden die Zugriffskontrolle und die Vermeidung einer Einbindung von ungewolltem Code. Darüber hinaus weist der Leitfaden darauf hin, dass eine Kommunikation (falls unbedingt notwendig) zwischen zwei Systemen nur von der höheren zur niedrigeren Sicherheitsstufe erlaubt ist. Die Systeme sollen außerdem mittels einer Hardware-Lösung entkoppelt sein.

B.2.18 NRC NUREG/CR-6734 “Digital Systems Software Requirements Guidelines”

Dieser Bericht bietet eine Richtlinie für die Bewertung von Integrität und Qualität von Anforderungen für den Einsatz von Software in Kernkraftwerken. Er soll auf Basis des NUREG 800 /NUR 87/ eine Richtlinie für die Aufstellung und die Überprüfung der Spezifikation von Anforderungen für die Software-Entwicklung liefern.

Der Bericht liefert verschiedene Checklisten sowie einen Satz von 45 Versagensfällen, die aufzeigen sollen, warum eine Überprüfung der Spezifikation der Anforderungen essentiell ist. Die vorgestellten Versagensfälle beschränken sich jedoch nicht auf Ausfälle in Kernkraftwerken.

Der Bericht beginnt mit der Übertragung der Anforderungen aus dem NUREG 800 /NUR 87/. Als wichtigste Anforderungen werden dabei die Begriffe Vollständigkeit, Konsistenz, Korrektheit, Nachvollziehbarkeit, Eindeutigkeit, Verifizierbarkeit und Klassifizierung genannt.

Bei der Formulierung konkreter Richtlinien wird zwischen den Kategorien Hoch und Mittel in Bezug auf die Wichtigkeit für die Sicherheit und Zuverlässigkeit des Systems unterschieden. Über die Richtlinien hinaus liefert der Bericht eine genauere Erläuterung dieser Richtlinien aus computertechnischer Sicht.

Im weiteren Verlauf liefert der Bericht eine Methode, mit der die Erfüllung der Anforderungen in der Umsetzung des Systems nachgewiesen werden kann. Der Bericht verweist darauf, dass im Wesentlichen die folgenden drei Aspekte überprüft werden müssen:

- Wurden alle Bereiche abgedeckt, die zur Entwicklung der Software aus den Anforderungen notwendig sind?
- Sind ausreichende und eindeutige Details enthalten, die eine Umsetzung aller Anforderungen garantieren?
- Können die aufgestellten Anforderungen verifiziert werden?

B.2.19 NRC NUREG/CR-7007 “Diversity Strategies for Nuclear Power Plant Instrumentation and Control Systems”

Viele Standards fordern für ein zuverlässiges rechnerbasiertes System den Nachweis von Diversität sowie eines gestaffelten Sicherheitskonzepts. Auch die gesetzlichen Vorgaben des 10 CFR 50 fordern allgemein für alle sicherheitstechnisch wichtigen Systeme die Berücksichtigung von systematischem, nicht zufälligem Versagen redundanter Elemente. Aus den gesetzlichen Anforderungen hat sich daher die Methodik der Diversität in konventionellen Systemen entwickelt. Speziell eine diversitäre Umsetzung von Softwarelösungen ist jedoch schwierig zu realisieren. Derzeit gibt es keine Vorgaben dazu, wie eine ausreichende Diversität von rechnerbasierten Systemen zu spezifizieren ist. Es ist zudem häufig strittig, ab wann eine Software zuverlässig als diversitär bezeichnet werden kann.

Um auf diesen Aspekt genauer einzugehen, werden in NUREG/CR-7007 die Diversität und das gestaffelte Sicherheitskonzept basierend auf Erfahrungen im nuklearen und nicht-nuklearen Bereich diskutiert. Darauf aufbauend werden verschiedene Kriterien für Diversität eingeführt. Ziel des Standards ist die Vorstellung verschiedener Konzepte für eine erfolgreiche Umsetzung von Diversität und einem gestaffelten Sicherheitskonzept sowie einer Methode zur Überprüfung, inwiefern eine Vermeidung von Fehlern mit gemeinsamer Ursache bedacht wurde.

Der Standard beginnt mit den Definitionen und Hintergründen der Diversität und des gestaffelten Sicherheitskonzepts sowie einem kurzen Auszug aus den gesetzlichen Bestimmungen. Darüber hinaus diskutiert er die Charakteristika von Fehlern mit gemeinsamer Ursache und stellt aktuelle Konzepte zur Bewertung von Diversität und des gestaffelten Sicherheitskonzepts vor. Auch die Auswirkungen von Diversität auf die Anfälligkeit gegen Fehler gemeinsamer Ursache werden erläutert. Im Anschluss werden die verschiedenen Formen von Diversität diskutiert.

Weiterführend werden in NUREG 7007 Strategien der Diversität aus aktuellen Anwendungen des nicht-nuklearen Bereichs vorgestellt. Hierbei werden Beispiele aus Industriezweigen aufgegriffen, in denen ebenfalls eine hohe Zuverlässigkeit der Systeme unabdingbar ist. Anschließend werden Beispiele zu Diversität und gestaffeltem Sicherheitskonzept aus der internationalen nuklearen Industrie vorgestellt. Hier werden neben konventionellen leittechnischen Systemen auch aktuelle rechnerbasierte Lösungen vorgestellt.

Abschließend werden drei verschiedene Strategien für eine erfolgreiche Umsetzung von Diversität in rechnerbasierten Systemen vorgestellt. Dazu zählt der Einsatz unterschiedlicher Technologie, unterschiedlichem Vorgehen und unterschiedlichem Aufbau. Es werden ihre Prinzipien und ihre Auswirkungen sowie die Erfüllung der verschiedenen Diversitätskriterien jeder Strategie im Einzelnen diskutiert. Die Strategien, die auf den konventionellen Methoden der Diversität basieren, wurden für die speziellen Anforderungen an rechnerbasierte Leittechnik erweitert.

Es wird darüber hinaus eine Analyse der Fragestellung durchgeführt, ob die Elemente der Diversität sämtliche Schwachstellen des Systems abdecken. Zusammenfassend beinhaltet diese Überprüfung die folgenden sechs Schritte:

1. Klassifizierung der Strategie
2. Bestätigung des inhärenten Unterschieds
3. Identifizierung der absichtlichen Diversität (bezogen auf einen möglichen Fehler gemeinsamer Ursache)
4. Kategorisierung der Strategie (A, B oder C)
5. Beurteilung der Strategie

6. Festlegung die Angemessenheit der Diversität

Ziel der Bewertung ist eine Bestätigung der erfolgreichen Vorbeugung eines Fehlers gemeinsamer Ursache.

B.2.20 Positionspapier europäischer Aufsichtsbehörden

Europaweit kommt immer mehr Software in Kernkraftwerken zum Einsatz, während eine Bewertung der Software im Hinblick auf ihre Zuverlässigkeit nach wie vor schwierig ist. Aus diesem Grund haben sich Aufsichtsbehörden verschiedener europäischer Länder (Belgien, Deutschland, Spanien, Großbritannien, Schweden und Finnland) dahingehend beraten, wie im jeweiligen Land Genehmigungen sicherheitskritischer Software in Kernkraftwerken durchgeführt werden. Gemeinsamkeiten in der Vorgehensweise sowie Unterschiede zwischen den einzelnen Ländern wurden schließlich in einem gemeinsamen Positionspapier festgehalten.

Das Papier referenziert hauptsächlich die DIN IEC 60880 als einzige in allen beteiligten Ländern geltende Norm und deren Inhalte sowie die ebenfalls in diesem Kapitel beschriebenen Normen und Standards. Darüber hinaus wird unter anderem auf den IAEA safety guide NS-G-1.1 (siehe Kap. B.2.11) verwiesen.

Das Positionspapier befasst sich mit dem gesamten Lebenszyklus der Software bzw. der softwarebasierten Komponenten beginnend mit Anforderungen, Architektur und Design von System und Software, über Implementierung, Verifizierung und Validierung der Software, bis hin zur Inbetriebnahme. Ebenfalls angesprochen werden Änderungen nach Inbetriebnahme und Konfigurationsmanagement der Software sowie die Durchführung von Tests.

Das Papier befasst sich nicht nur mit neu erstellter Software, sondern es werden auch Anforderungen an den Einsatz vorgefertigter Software zusammengefasst. Weiterhin beschäftigt sich das Dokument in allgemeiner Weise mit Anforderungen an Software-Werkzeuge, organisatorische Anforderungen und Anforderungen an die Sicherstellung der Softwarequalität. Aspekte der Sicherung werden ebenfalls angesprochen.

Ein weiteres Kapitel beinhaltet Empfehlungen und Anforderungen im Hinblick auf die Diversität von Systemen und Komponenten. Dabei wird auch auf die Unabhängigkeit

von Systemen und die Diversität speziell von Software eingegangen. Weiterhin beschäftigt sich das Positionspapier mit der Zuverlässigkeit der Software, deren Verifizierung und Validierung sowie der Verwendung von Daten zur Betriebserfahrung mit der Software. Einige Wesentliche Anforderungen zu dieser Thematik lassen sich wie folgt zusammenfassen:

- Um eine hohe Zuverlässigkeit zu erreichen, werden typischerweise redundante Systeme und Komponenten genutzt. Während Redundanz mit identischen Systemen und Komponenten einen effektiven Schutz gegen Zufallsausfälle von Hardware bietet, geht die Möglichkeit eines GVA aus systematischen Fehlern hervor, die beispielsweise in der Spezifikation, im Design, in der Implementierung und bei Fehlern in der Wartung liegen können. Die Einführung von Diversität kann einen Schutz gegen GVAs bieten.

- Eine Vorgehensweise, die typischerweise während des Architekturdesigns zum Schutz gegen GVAs angewendet wird, ist es, den Gebrauch von multiplen, möglicherweise diversitären Systemen in Betracht zu ziehen. Auch die Erwägung des Einsatzes von Defence-in-Depth, so dass ein Fehler in einer Ebene durch die übergreifende Systemarchitektur kompensiert wird, kann zur Notwendigkeit des Einsatzes von diversitären, möglicherweise rechnerbasierten und programmierbaren Systemen führen.

- Die Anzahl der benötigten Systeme, Komponenten oder Kanäle, der Grad an Diversität zwischen ihnen, die vorgesehenen Zuverlässigkeitsziele und die Auswahl der Technologie der benötigten Systeme, Komponenten oder Kanäle muss bestimmt werden. Eine Vorgehensweise für 1v2-Systeme, von denen eines ein rechnerbasiertes oder programmierbares System ist, ist es, zusätzlich ein einfaches nicht rechnerbasiertes oder programmierbares System einzusetzen. Wenn mehrere rechnerbasierte oder programmierbare Systeme, Komponenten oder Kanäle eingesetzt werden, muss der Einsatz von Software-Diversität in Betracht gezogen werden.

B.3 Nicht-Nukleare Standards und Normen

B.3.1 Schienenverkehr

DIN EN 50126 „Bahnanwendungen – Spezifikation und Nachweis der Zuverlässigkeit, Verfügbarkeit, Instandhaltbarkeit, Sicherheit (RAMS)“

DIN EN 50126 stellt den Bahnunternehmen und der Bahnindustrie sowie ihren Zulieferern ein Verfahren zur konsequenten Anwendung eines Managements für Zuverlässigkeit, Verfügbarkeit, Instandhaltbarkeit und Sicherheit (RAMS) zur Verfügung. Sie gilt für alle Bahnanwendungen und für alle Systemebenen, für vollständige Bahnstrecken bis hin zu allen zu den Bahnstrecken gehörenden Großsystemen sowie für die einzelnen und kombinierten Subsysteme und Komponenten innerhalb dieser Großsysteme, einschließlich derjenigen, die Software enthalten.

In der Norm werden zunächst die Begriffe Zuverlässigkeit, Verfügbarkeit, Instandhaltbarkeit, Sicherheit sowie ihre Wechselwirkungen und der Begriff RAMS, unter dem die Kombination dieser vier Begriffe verstanden wird, definiert. RAMS ist demnach eine Charakteristik für das Langzeitbetriebsverhalten eines Systems und wird durch die Anwendung anerkannter technischer Konzepte, Verfahren, Werkzeuge und Techniken während des gesamten Lebenszyklus des Systems erreicht. RAMS für ein System lässt sich als qualitative und quantitative Angabe des Grads beschreiben, bis zu welchem man sich darauf verlassen kann, dass das System, das Subsystem oder die Komponenten, aus denen das System besteht, spezifikationsgemäß funktionieren und ebenso verfügbar und sicher sind.

Zunächst werden in DIN EN 50126 RAMS beeinflussende Faktoren für Bahnen und ein Prozess zur Ermittlung dieser Faktoren betrachtet, die dann in die Spezifikation der RAMS-Systemanforderungen eingehen. Anschließend wird das Risikokonzept beschrieben. Hier muss im Rahmen einer Risikoanalyse zum einen die Wahrscheinlichkeit oder Häufigkeit eines Ereignisses oder einer Kombination von Ereignissen, die zu einem Gefahrenfall führen, und zum anderen die Folgen und Konsequenzen des Gefahrenfalls betrachtet werden, um eine Risikobewertung durchführen zu können. Anhand dessen können dann Anforderungen an die Sicherheitsintegrität abgeleitet werden. Hierbei wird explizit darauf hingewiesen, dass zwar eine grundsätzliche Korrelation zwischen Sicherheitsintegrität und Ausfallwahrscheinlichkeit in DIN EN 61508 gegeben ist, die DIN EN

50126 eine solche Korrelation für Bahnsysteme jedoch nicht vorschreibt. Die Festlegung dieser Korrelation liegt laut DIN EN 50126 in der Verantwortlichkeit der Bahnunternehmen.

Anschließend wird ein Prozess für ein RAMS-Management auf der Grundlage des System-Lebenszyklus (siehe Kap.4.4.1) und dessen Aktivitäten beschrieben. Im Rahmen dessen beschäftigt sich die Norm mit der Spezifikation von RAMS-Anforderungen und dem Nachweis der Erfüllung dieser Anforderungen. Dieses RAMS-Management soll die Zusammenarbeit zwischen den Bahnunternehmen und der Bahn-Zulieferindustrie zur Erzielung einer optimalen Kombination von RAMS und Kosten fördern.

DIN EN 50128 „Bahnanwendungen – Telekommunikationstechnik, Signaltechnik und Datenverarbeitungssysteme – Software für Eisenbahnsteuerungs- und Überwachungssysteme“

Die DIN EN 50128 konzentriert sich auf den Lebenszyklus sicherheitsrelevanter Software für Steuerungs- und Überwachungssysteme im Schienenverkehr. Sie beschreibt Anforderungen an die Entwicklung, Bereitstellung und Wartung solcher Software. Dazu zählen auch Anforderungen hinsichtlich der Organisationsstruktur und der Aufteilung von Verantwortlichkeiten der in Bereitstellung und Wartung der Software eingebundenen Personen. Zusätzlich beschreibt die Norm auch Kriterien für die Qualifikation und die Fachkenntnis des Personals. Die DIN EN 50128 basiert auf den in der DIN EN 61508-1 eingeführten Sicherheits-Integritätsleveln (SIL), unterscheidet dabei jedoch zwischen System-SIL und Software-SIL. Letzteres besteht statt aus 4 aus 5 Stufen von 0 bis 4. Es muss auf Grundlage des System-SIL und der Höhe des Risikos, das mit der Benutzung der Software im System verbunden ist, auf Systemebene festgelegt und begutachtet werden.

Die Vorgehensweise in Bezug auf sicherheitsrelevante Software, einschließlich Anwendungsprogrammierung, Betriebssysteme, unterstützende Werkzeuge und Firmware, wird in der Norm mit aufeinander folgenden funktionalen Schritten beschrieben. Die DIN EN 50128 befasst sich mit den Schritten

- Entwurf, Entwicklung und Test der Software
- Integration der Software auf der Ziel-Hardware und Verifizierung der Funktionalität

- Abnahme und Bereitstellung der Software
- Software-Wartung.

Aktivitäten, die sich über die gesamte Software-Entwicklung erstrecken, wie beispielsweise Software-Tests, Verifizierung und Validierung, Begutachtung, Qualitätssicherung und Änderungen der Software, sind ebenfalls Inhalt dieser Norm. Darüber hinaus befasst sich die Norm mit Anforderungen für Hilfswerkzeuge und durch Anwendungsdaten oder Algorithmen konfigurierbare Systeme. Zusätzlich werden Anforderungen hinsichtlich des an der Softwareentwicklung beteiligten Personals gestellt. Ebenfalls angesprochen wird vorgefertigte Software.

Abschließend wird eine Übersicht über Verfahren, die im Software-Lebenszyklus einsetzbar sind, gegeben. Dabei werden diese Verfahren jeweils kurz beschrieben und die Zielsetzung genannt.

Die Norm enthält eine Vielzahl von Anforderungen an die Bereiche Softwaremanagement und -organisation, Software-Sicherung, Entwicklung generischer Software, Entwicklung der Anwendungsdaten oder -algorithmen und Bereitstellung der Software. Für viele Aspekte dieser Bereiche sind die Anforderungen nach SIL 0, SIL1/ SIL2 und SIL3/ SIL4 gestaffelt.

Es sind Techniken und Maßnahmen für Software-SIL 0 bis 4 enthalten. Dabei sind die für Software-SIL 1 und SIL 2 sowie die für Software-SIL 3 und 4 geforderten Techniken identisch. Je höher das Risiko ist, das von einem Softwarefehler ausgeht, desto höher muss das Software-SIL gewählt werden. Die Festlegung des Software-SIL muss auf Grundlage des System-SIL und des mit Nutzung der Software im System verbundenen Risikos erfolgen. Die Norm gibt allerdings keinen Hinweis darauf, welcher Level für ein gegebenes Risiko angemessen ist.

Es werden Kriterien für die Auswahl der Techniken und Maßnahmen zu verschiedenen Aspekten wie Software-Architektur, Integration, Prüfung der Gesamt-Software, Analysetechniken, etc. in Tabellen zusammengefasst. Dabei enthält jede Tabelle neben einer Auflistung der Techniken und Maßnahmen Spalten für die verschiedenen Software-SIL, in denen jeweils vermerkt ist, ob die genannte Technik oder Maßnahme für dieses Level „verbindlich“, „dringend empfohlen“, „empfohlen“ oder „nicht empfohlen“ ist, oder ob es keine Empfehlung gibt. Damit macht die DIN EN 50128 nicht nur konkrete Aussagen zu

Maßnahmen, die zur Erreichung eines speziellen SIL ergriffen werden müssen, sondern auch Aussagen zu Techniken, die hierbei nicht eingesetzt werden dürfen. Auch ist aufgeführt, welche Kombinationen von Techniken für SIL3/ SIL4 bzw. für SIL1/ SIL2 genehmigt sind.

DIN EN 50129 „Bahnanwendungen – Telekommunikationstechnik, Signaltechnik und Datenverarbeitungssysteme – Sicherheitsrelevante elektronische Systeme für Signaltechnik“

DIN EN 50129 definiert Anforderungen für die Anerkennung und Zulassung von sicherheitsrelevanten elektronischen Systemen im Eisenbahnbereich. Sicherheitsrelevante elektronische Systeme für Signaltechnisch umfassen Hardware- und Softwareaspekte. DIN EN 50129 befasst sich mit den Anforderungen an sicherheitsrelevante Hardware und an das Gesamtsystem. Diese Anforderungen basieren dabei auf den Anforderungen der DIN EN 50126. Sie gilt für Spezifikation, Entwurf, Konstruktion, Installation, Abnahme, Betrieb, Instandhaltung und Änderung bzw. Erweiterung von kompletten Signalsystemen sowie Teilsystemen und Einrichtungen des Gesamtsystems.

DIN EN 50129 definiert die Bedingungen, die ein sicherheitsrelevantes elektronisches System, Teilsystem oder eine Betrachtungseinheit der Eisenbahn erfüllen muss, um für die vorgesehene Anwendung als angemessen sicher anerkannt zu werden. Hierzu werden Anforderungen an den Nachweis des Qualitätsmanagements, des Sicherheitsmanagements und der funktionalen und technischen Sicherheit formuliert.

DIN EN 50129 definiert zunächst, wie der Sicherheitsnachweis strukturiert werden soll. Anschließend werden dann für jede Phase des Nachweises Anforderungen formuliert. Für das Qualitätsmanagement wird beispielsweise gefordert, dass es über den gesamten Lebenszyklus hinweg anwendbar sein muss. Außerdem sollten die Tiefe der Darlegungen und der Umfang der begleitenden Dokumentation der zugeordneten Sicherheitsanforderungsstufe SIL angemessen sein. Der Sicherheitsmanagementprozess muss unter Kontrolle einer geeigneten Sicherheitsorganisation ablaufen. Beim Nachweis des korrekten funktionalen Verhaltens sind folgende Aspekte zu berücksichtigen:

- Beschreibung der Systemarchitektur
- Definition der Schnittstellen
- Erfüllung der Systemanforderungsspezifikation
- Erfüllung der Sicherheitsanforderungsspezifikation

- Nachweis der korrekten Hardware- und Softwarefunktionalität

Darüber hinaus werden Ausfallauswirkungen, der Betrieb mit externen Einflüssen und sicherheitsbezogene Anwendungsbedingungen sowie die Sicherheitserprobung betrachtet.

Abschließend wird der Prozess der Sicherheitsanerkennung und –zulassung beschrieben.

Im Anhang werden Informationen zur Zuordnung der Systeme zu Sicherheitsanforderungsstufen (SIL) formuliert. Dabei wird erläutert, wie eine solche Zuordnung der Systeme zu den SIL erfolgen kann und was dabei zu beachten ist. Abschließend erfolgt eine Zuordnung von SIL zu tolerierbaren Gefährdungsraten THR:

SIL 1: $10^{-6} \leq THR < 10^{-5}$
SIL 2: $10^{-7} \leq THR < 10^{-6}$
SIL 3: $10^{-8} \leq THR < 10^{-7}$
SIL 4: $10^{-9} \leq THR < 10^{-8}$

Diese Werte ergeben sich aus der Annahme, dass die Sicherheit sowohl von angemessenen Maßnahmen zur Vermeidung oder Tolerierung von Fehlern zum Schutz gegen systematische Fehler als auch von angemessenen Maßnahmen zur Beherrschung von zufälligen Fehlern abhängt. Maßnahmen gegen beide Ursachen sollten ausbalanciert sein, um ein Optimum an Sicherheit eines Systems zu erreichen. Nach DIN EN 50129 werden SIL als ein Mittel benutzt, um qualitative Verfahren zur Vermeidung systematischer Fehler an quantitative Verfahren zur Beherrschung zufälliger Fehler anzupassen, da es nicht möglich ist, Sicherheit gegen systematische Fehler zu quantifizieren.

B.3.2 Automobil

ISO 26262: „Road Vehicles – Functional Safety“

ISO 26262 ist eine Norm für sicherheitsrelevante elektrische und elektronische Systeme in Kraftfahrzeugen. Ihre Umsetzung soll die funktionale Sicherheit eines Systems mit elektrischen, elektronischen und softwarebasierten Komponenten im Kraftfahrzeug im gesamten Lebenszyklus gewährleisten. Es handelt sich hierbei um eine Anpassung der Norm IEC 61508 an die spezifischen Gegebenheiten der Automobilindustrie.

Die Normenreihe besteht aus mehreren Teilen:

- Teil 1: Vokabular
- Teil 2: Management der funktionalen Sicherheit
- Teil 3: Konzeptphase
- Teil 4: Produktentwicklung: Systemebene
- Teil 5: Produktentwicklung: Hardwareebene
- Teil 6: Produktentwicklung: Softwareebene
- Teil 7: Produktion, Betrieb und Außerbetriebnahme
- Teil 8: Unterstützende Prozesse
- Teil 9: ASIL- und sicherheitsorientierte Analysen
- Teil 10: Guideline

Die Normenreihe ist anzuwenden auf sicherheitsrelevante Systeme mit elektrischen oder elektronischen Komponenten, die in Serienproduktions-Personenkraftwagen mit einem Maximalgewicht von 3500 kg installiert sind. Sie behandelt die Gefährdung durch Fehlverhalten von sicherheitsrelevanten Systemen mit elektrischen oder elektronischen Komponenten und die Wechselwirkung dieser Systeme, nicht jedoch die Gefährdung durch Feuer, Rauch, Hitze, Strahlung, Korrosion, etc.

B.3.3 Luftfahrt

RTCA DO-178C „Software Considerations in Airborne Systems and Equipment Certification“

Der US-Standard DO-178C /RTCA 11/ der Radio Technical Commission for Aeronautics (RTCA) ist ein weltweit viel zitierter Standard bezüglich Software in sicherheitskritischen Infrastrukturen. Der Standard umfasst Anforderungen an den gesamten Lebenszyklus der Software.

Zu den Systemaspekten zählen beispielsweise Anforderungen an das System, der Informationsfluss zwischen den Lebenszyklen von System und Software bzw. von Hardware und Software, Anforderungen an die Architektur sowie Zuordnung zu einem Software-Level. Im Standard werden fünf dieser Software-Level definiert, die sich durch die Auswirkungen eines Softwareversagens oder eines unvorhergesehenen Verhaltens der Software auf das Flugzeug unterscheiden:

- Level A: katastrophale Fehlfunktion des Flugzeugs (viele Tote, Verlust des Flugzeugs)
- Level B: gefährliche Fehlfunktion des Flugzeugs (Deutliche Reduktion von Sicherheitsmargen, starke Belastung des Flugpersonals, Situation vom Flugpersonal u. U. nicht vollständig zu bewältigen, schwere Verletzungen einiger Passagiere bis hin zu einigen Toten)
- Level C: größere Fehlfunktion des Flugzeugs (Reduktion von Sicherheitsmargen, deutliche Belastung des Flugpersonals, körperliche Unannehmlichkeiten bis hin zu Verletzungen von Passagieren)
- Level D: kleinere Fehlfunktion des Flugzeugs (geringe Reduktion von Sicherheitsmargen, geringfügig erhöhte Belastung des Flugpersonals, körperliche Unannehmlichkeiten bei manchen Passagieren)
- Level E: Fehlfunktion des Flugzeugs ohne Auswirkungen auf die Sicherheit

Häufig werden abgestufte Anforderungen für diese Level gestellt. Grundsätzlich unterscheidet der Standard auch zwischen Anforderungen an die Software, die sich aus einer Analyse der Systemanforderungen oder der Sicherheitsanforderungen ergeben („low-level requirements“) und Anforderungen an die Software, die sich aus den High-Level-Anforderungen ableiten oder aus diesen entwickeln („high-level requirements“) lassen.

Des Weiteren beschäftigt sich der Standard mit dem Einsatz vorgefertigter Software, mit Software-Werkzeugen und mit weiteren Methoden zur Sicherstellung der Softwarequalität. Dabei liegt der Schwerpunkt auf der Zertifizierung von Software.

Für die im Standard genannten Prozesse werden in tabellarisch die jeweils durchzuführenden Schritte, die Anwendbarkeit der Anforderungen für die verschiedenen Software-Level, die vorzulegenden Nachweise sowie die Kategorie des anzuwendenden Konfigurationsmanagements aufgeführt.

Zusätzlich zu den Anforderungen im Bereich Softwareentwicklung enthält der Standard eine Beschreibung des Prozesses zur Vorbereitung der Genehmigung der Software. Die Hauptinhalte sind hierbei

- Herstellung von Kommunikation und Verständnis zwischen dem Antragsteller und der Genehmigungsbehörde während des Software-Lebenszyklus zur Unterstützung des Genehmigungsprozesses,
- Erreichen einer Übereinstimmung hinsichtlich der Genehmigung des vom Antragsteller vorzulegenden Plans zur Softwarezertifizierung und
- Nachweisführung über den Prozess des Software-Lebenszyklus.

Zusätzlich werden im Standard die zur Nachweisführung im Rahmen der Zertifizierung vorzulegenden Unterlagen und Daten beschrieben. Dazu zählen mindestens:

- Plan zu Softwareaspekten der Zertifizierung,
- Index der Konfiguration der Software sowie
- Zusammenfassung der Übereinstimmung zwischen erstellter Software und dem Plan zu Softwareaspekten der Zertifizierung.

Grundsätzlich werden in DO-178C /RCT 11/ unterschiedliche Begriffe in Bezug auf die Genehmigung eines Flugzeugs und der darin enthaltenen maschinellen Ausrüstung verwendet. Der Begriff „Zertifizierung“ bezieht sich dabei auf das Flugzeug, Motoren, Propeller und auch Notstromaggregate. Dagegen wird die Software als Teil des Gesamtsystems oder der an Bord des zertifizierten Flugzeugs installierten Ausrüstung angesehen und nicht einzeln zertifiziert. Systeme und Ausrüstungsteile einschließlich der enthaltenen Softwarebestandteile werden „genehmigt“ und damit als Bestandteile der Zertifizierung anerkannt. Damit gilt die Genehmigung einer Software nach dem momentanen Standards und Normen immer nur im Kontext einer speziellen Zertifizierung und nicht davon losgelöst. Zusätzlich ist eine Qualifizierung eines Softwarewerkzeugs oder einzelner Funktionen eines Softwarewerkzeugs immer dann notwendig, wenn einzelne im DO-178C beschriebene Prozesse durch den Einsatz dieses Softwarewerkzeugs automatisiert, reduziert oder gestrichen werden, ohne dass der Output des Werkzeugs so wie im Standard beschrieben verifiziert wird. Die Qualifizierung eines Softwarewerkzeugs bezieht sich immer nur auf die Nutzung in dem im Plan zu Softwareaspekten der Zertifizierung beschriebenen System. Für die Nutzung in einem anderen System muss eine erneute Qualifizierung erfolgen. Der DO-178C nennt fünf Qualifizierungsstufen. Welche davon ein Softwarewerkzeug erfüllen muss, richtet sich nach verschiedenen Kriterien wie beispielsweise der Frage, ob das Werkzeug selbst einen

Fehler einbringen könnte, oder der Frage, ob das Werkzeug bei der Erkennung eines Fehlers versagen könnte.

FAA 8110.49 „Software Approval Guidelines“

Diese Anordnung der US-amerikanischen Bundesflugverwaltung (Federal Aviation Administration, FAA) etabliert Richtlinien zur Zulassung von Software in Erfüllung der Anforderungen aus dem US-Standard RTCA DO-178C. Die Anordnung ist ergänzend zum DO-178C zu verstehen und verdeutlicht an vielen Stellen dessen Anforderungen. Sie wird zur Zertifizierung von in Flugzeugen verbauten Systemen und Bauteilen mit Softwarebestandteilen herangezogen.

Folgende Aspekte werden in der Anordnung angesprochen:

- Review-Prozess der Software während des Software-Lebenszyklus durch die Genehmigungsbehörde (Reviews während der Planung, der Entwicklung, der Verifizierung und der Zertifizierung)
- Grad der Beteiligung der FAA an Softwareprojekten (Zeitpunkte, Häufigkeit sowie Gebiete der Beteiligung),
- Überprüfung der Softwarekonformität (Übereinstimmung mit den US-Bundesluftfahrtregelungen),
- Genehmigung von „field-loadable“ Software (vor Ort ladbare Software, schließt auch Software von Servicerechnern ein),
- Genehmigung von durch den Nutzer veränderbarer Software,
- Vorgefertigte Software bei Level D,
- Qualifizierung von Software-Werkzeugen,
- Veränderungen von Software in Alt-Systemen (d. h. Systemen, die noch anhand von älteren Versionen des DO-178C genehmigt wurden),
- Analysen für Software-Änderungsvorhaben sowie
- Wiederverwendung von Daten eines Software-Lebenszyklus in einem weiteren Genehmigungsverfahren.

Insgesamt ist die Anordnung der FAA auf die schrittweise Abarbeitung des Zertifizierungsprozesses ausgerichtet und stark anwendungsorientiert aufgebaut. Beispielsweise steigt der Grad der Beteiligung der FAA an Softwareprojekten typischerweise mit dem angestrebten Level der Software. Im DO-178C /RTCA 11/ werden fünf Software-Levels definiert, die sich durch die Auswirkungen eines Softwareversagens oder eines unvorhergesehenen Verhaltens der Software auf das Flugzeug unterscheiden. Für jedes dieser Software-Level enthält der DO-178C eine Reihe von Anforderungen, die bei der Genehmigung der jeweiligen Software erfüllt sein müssen. Darüber hinaus gibt es für Software der Level A, B, C und D eine Checkliste für den Anwender, aus der sich anhand von Informationen über den Antragsteller, den Entwickler der Software und das derzeitige System eine Gesamtpunktzahl ermitteln lässt, welche wiederum einem Maß der Beteiligung der FAA am Softwareprojekt zugeordnet wird.

Die Anordnung beinhaltet im Hinblick auf field-loadable Software, aufbauend auf den im DO-178C gestellten Design-Anforderungen, zusätzliche Richtlinien für die Genehmigungsbehörde. Dabei werden die Aspekte der Genehmigung von field-loadable Software, deren Installation im Flugzeug sowie von Instandhaltung und Teil-Kennzeichnung angesprochen. Es wird gesondert auf die Genehmigung von field-loadable Software eingegangen, die identisch zu bereits genehmigter Software ist und auf typgeprüftem Gerät installiert werden soll.

Vorgefertigte Software ist nach DO-178C grundsätzlich ebenfalls genehmigungsfähig, allerdings gelten die dafür beschriebenen Anforderungen nur für das Software-Level D und somit für Software, deren Fehlfunktion höchstens zu einer kleineren Fehlfunktion des Flugzeugs führen kann. Eine Genehmigung von durch den Nutzer veränderbarer Software beinhaltet die Erlaubnis für die Airline bzw. den Operateur, innerhalb festgelegter Grenzen und unter Zuhilfenahme spezifizierter Prozeduren die Software ohne weitere Beteiligung der Genehmigungsbehörde zu verändern. Dabei steht im Vordergrund, dass eine Modifikation der Software die Sicherheitsmargen und den Betrieb des Flugzeugs sowie die Arbeitslast für das Flugpersonal nicht verändern darf. Zusätzlich darf eine solche Modifikation auch keine Auswirkungen auf nicht veränderliche Softwarebestandteile und Schutzmechanismen des Systems haben.

Im Hinblick auf die Qualifizierung eines Softwarewerkzeugs enthält die Anordnung der FAA weitere Erklärungen und Ergänzungen zu den Anforderungen des DO-178B /RTCA 92/. Vor allem werden die Intention dieser Anforderungen und deren Anwendung näher erläutert. Hierzu zählen beispielsweise die Fragen

- wann ein Software-Werkzeug qualifiziert werden sollte,
- welche Kriterien für Verifizierungs-Werkzeuge und welche Kriterien für Entwicklungs-Werkzeuge gelten, oder
- welche Daten zur Nachweisführung vorgelegt werden müssen.

Weiterhin beschäftigt sich die Anordnung der FAA mit der Genehmigung von Software-Änderungen in bereits in Betrieb genommenen Systemen bzw. „Alt-Systemen“. Der DO-178B, auf den sich die Anordnung 8110.49 bezieht, unterscheidet sich teilweise deutlich von seinen Vorgängern. Unterschiede bestehen beispielsweise in den Anforderungen zu durchzuführenden Software-Tests oder in der Klassifizierung von fünf statt zuvor nur drei verschiedener Software-Level. Die Anordnung der FAA stellt den Zusammenhang zwischen den älteren Versionen des Standards und dem DO-178B her und beschreibt, wie die Äquivalenz von bereits genehmigter, älterer Software zu den neuen Anforderungen genügender Software aufgezeigt werden kann. Sowohl für den Fall, dass eine Äquivalenz gezeigt werden kann, als auch für den Fall, dass dies nicht gelingt, beschreibt die Anordnung der FAA weitere Vorgehensweisen.

Ein weiterer Aspekt, der in der Anordnung der FAA ausführlich beschrieben wird, sind Analysen zur Auswirkung von Softwareänderungen. Bei der Veränderung von bereits in Betrieb genommener Software ist laut DO-178C eine erneute Verifizierung der Änderungen und der von ihr betroffenen Systemteile durchzuführen. Die Anordnung 8110.49 beschreibt eine Prozedur, um die Auswirkungen der Softwareänderungen zu bestimmen und sicherzustellen, dass die Änderungen keine negativen Auswirkungen auf die Sicherheit haben. Zusätzlich wird bei dieser Analyse festgestellt, in welchem Maß die Genehmigungsbehörde bei den Softwareänderungen involviert werden muss.

B.3.4 Raumfahrt

NASA-STD-8719.13C: „Software Safety Standard“

Der NASA Standard 8719.13C befasst sich mit Anforderungen an die Prozesse bei Erwerb, Entwicklung und Modifikation von Softwarebestandteilen für Systeme mit sicherheitstechnischer Relevanz hinsichtlich der Sicherheit der Software. Es wird betont, dass es notwendig ist, die Prozesse, die die Sicherheit der Software betreffen, in den Lebens-

zyklus des Gesamtsystems und damit in die Prozesse, die die Sicherheit des Gesamtsystems betreffen, einzubetten. Der Standard beschreibt verschiedene Aktivitäten, Indikatoren und die relevante Dokumentation der Prozesse und nennt organisatorische Vorkehrungen zur Umsetzung.

Bei der Erarbeitung der verschiedenen Anforderungen an den Erwerbs-, bzw. Entwicklungsprozess definiert dieser Standard Aktivitäten, die vor, während und nach dem Softwareentwicklungsprozess notwendig sind, und stellt Anforderungen an diese.

Ein wichtiger Bestandteil des Standards ist eine Methode zur Bestimmung der Kritikalität eines bestimmten Software-Bestandteils hinsichtlich der funktionalen Sicherheit des Systems. Hierfür wird ein zweiteiliges Verfahren vorgegeben:

- Software-Sicherheits-„Lackmustest“: Es wird eine Reihe von Kriterien vorgegeben, die, wenn erfüllt, Software als sicherheitsrelevant auszeichnet. Die Kriterien beziehen sich dabei auf die Ausführung von Sicherheitsfunktionen sowie auf das Herbeiführen von unsicheren Zuständen des Systems. Software, welche nicht als sicherheitsrelevant identifiziert wird, liegt nicht im Anwendungsbereich dieses Standards.
- Software-Risikobewertung: Die Risikobewertung sicherheitsrelevanter Software wird in mehreren Schritten durchgeführt. Zuerst wird der Automatisierungsgrad, bzw. die Zeitkritikalität der auszuführenden Funktion betrachtet und entsprechend gegebener Kategorien eingeordnet.

ECSS-Q-80, “Software Product Assurance”

Die aktuelle Version des Standards ECSS-Q-ST-80 C stand der GRS zu gegebenen Zeitpunkt nicht zur Verfügung, da für die Einsicht in den Standard eine Mitgliedschaft bei der ECSS notwendig ist. Daher wird im Folgenden auf die die ursprüngliche Version des Standards aus dem Jahr 1996 eingegangen (Version ECSS-Q-80A). Die Standard-Serie beinhaltet vier Handbücher zu speziellen Software-Bereichen. Sämtliche Handbücher liegen ebenfalls nur in der ersten Version (A) vor. Für die Software-Zuverlässigkeit sind insbesondere einige Kapitel des allgemeinen Standards ECSS-Q-80 sowie das Handbuch ECSS-Q-HB-80-03 von Interesse.

Der Standard ECSS-Q-HB-80 liefert eine Anleitung zur Erstellung von Anforderungen, einige konkrete Anforderungen an die Bewertung von Software sowie einen Leitfaden

zur Verifizierung diese Anforderungen. Der Standard beschäftigt sich allgemein mit allen Bereichen der Bewertung von Software, wie z.B. Verifizierung und Validierung und Zuverlässigkeit.

Zu Beginn des Standards werden die Anforderungen an die Leistung, die Verwaltung und die Rahmenbedingungen des Software-Entwurfs gestellt. Im Anschluss formuliert der Standard die Anforderungen an die Entwicklung der Software bezogen auf den konkreten Lebenszyklus. Abschließend werden Anforderungen an die Qualität von Software formuliert. Dabei wird unterschieden zwischen Anforderungen an die Zuverlässigkeit, die Sicherheit und die Instandhaltung der Software.

Die wesentlichen Anforderungen die der Standard an die Qualität von Software stellt, sind Übertragbarkeit, Wartbarkeit, Wiederverwendbarkeit und Korrektheit. Um dies zu erreichen, sollen die Entwickler Regeln für den Entwurf, die Entwicklung, den Code und die Umsetzung der Software festlegen. Auch sollen Metriken entwickelt werden, die eine Verifizierung dieser Anforderungen ermöglichen.

Im weiteren Verlauf stellt der Standard konkrete Anforderungen an die Software-Qualität und somit auch indirekt an die Zuverlässigkeit von Software. Dazu zählt, dass die Anforderungen dokumentiert werden sollen. Sie sollen vollständig, eindeutig und hinreichend präzise sein, um eine Verifizierung und Validierung zu ermöglichen. Für jede Anforderung soll die Methode der Verifizierung festgelegt werden. Jede Anforderung soll eine eindeutige Kennung besitzen.

Zusätzlich stellt der Standard weiterführende Anforderungen an den Entwurf von Software, die Dokumentation, die Hardware und die Firmware. Auch einige Anforderungen für die Verwendung bereits existierender Software werden vorgestellt.

Konkret mit der Zuverlässigkeit und Sicherheit von Software beschäftigt sich das Handbuch ECSS-Q-HB-80-03. Es liefert eine Richtlinie für die Anwendung der an die Zuverlässigkeit und Sicherheit gestellten Anforderungen aus ECSS-Q-80. Das Handbuch stellt konkrete Methoden und Techniken zur Anwendung in Bezug auf die Zuverlässigkeit und Sicherheit von Software vor. Hierbei beschränken sich die Methoden jedoch auf die Bewertung.

Zu Beginn des Handbuchs werden einige Begrifflichkeiten definiert. Insbesondere wird Softwarezuverlässigkeit als Eigenschaft der Software definiert, fehlerfrei zu sein. Das

Handbuch unterscheidet ganz gezielt zwischen Softwarezuverlässigkeit und Systemzuverlässigkeit. Während die Systemzuverlässigkeit durch quantitative und qualitative Modelle bewertet werden kann, ist diese Umsetzung für die Softwarezuverlässigkeit laut Handbuch in der Luft- und Raumfahrttechnik schwer zu übertragen. Hauptgrund hierfür ist die nicht vollständig gerechtfertigte Übertragung von Software-Eigenschaften auf die mathematischen Gleichungen der Modelle.

Im weiteren Verlauf beschreibt das Handbuch eine Methode zur Bewertung der umfassenden Verlässlichkeit und Sicherheit von Software sowie der entsprechenden Anforderungen. Hierzu werden zunächst die wesentlichen Vorgaben für die Definition der Anforderungen an die Klassifizierung der Software, die Handhabung der Software, die Integration der Software, aber auch an die Zuverlässigkeit von Software vorgestellt.

Im Anschluss präsentiert das Handbuch Hintergrundinformationen zu einigen Methoden und Modellen der Überprüfung der Software-Zuverlässigkeit und -Sicherheit. Hierzu zählen die SFMECA (Software Failure Modes, Effects and Criticality Analysis), die SFTA (Software Fault Tree Analysis) und die SCCA (Software Common Cause Analysis).

B.3.5 Wehrtechnik

MIL-STD-882E „Department of Defense Standard Practice - System Safety“

Dieser Standard des US-Verteidigungsministeriums beschreibt die Herangehensweise zur Gefahrenvermeidung und Risikominderung für Wehrtechnik-Systeme, Komponenten sowie Zubehör und Infrastruktur dieser Systeme einschließlich Hardware und Software. Dabei stehen Gefahren im Vordergrund, die zum Tod, zur Verletzung oder zur Berufsunfähigkeit von Personen, der Beschädigung oder dem Verlust von Gerät oder zum Schaden für die Umwelt führen könnten. Der Begriff Software schließt im Rahmen dieses Standards auch Firmware ein.

Der Standard beschreibt zunächst Sicherheitsanforderungen an Systeme über deren Lebenszyklus hinweg. In diesem Zusammenhang führt er einen Prozess für die Systemsicherheit ein. Einer der Prozessschritte befasst sich mit der Einschätzung und Dokumentation von Risiken. Dazu werden sowohl Kategorien für die Schwere einer Gefahr als auch für deren Eintrittswahrscheinlichkeit definiert, woraus sich eine Matrix zur Risi-

koeinschätzung ergibt. Darauf aufbauend wird festgestellt, dass diese Herangehensweise zur Risikoeinschätzung bei der Einschätzung des Beitrags von Software zum Gesamtrisiko des Systems nicht ausreicht. Daher führt der Standard für Software eine andere Vorgehensweise zur Bestimmung des Risikos ein als für Hardware. Hierzu werden zunächst „Software Control Categories“ eingeführt, die beschreiben, welchen Grad der Autonomie und der Kontrollbefugnisse die Software in Bezug auf das Systemverhalten hat. Aus diesen und den zuvor eingeführten Kategorien für die Schwere einer Gefahr ergibt sich eine Matrix, die Indices dafür enthält, wie sicherheitskritisch die Software ist (Software Criticality Index, SwCI). Je nach Kategorie unterscheiden sich auch die allgemein formulierten Anforderungen hinsichtlich ihrer Strenge (Level of Rigor, LOR). Der LOR spezifiziert, wie detailliert und tief die Analyse und Verifizierung der Software erfolgen muss, um ein ausreichendes Maß an Sicherheit bezüglich der korrekten Ausführung einer sicherheitskritischen oder sicherheitsrelevanten Softwarefunktion zu erreichen. Dabei werden für jeden LOR durchzuführende Aufgaben genannt. Für den Fall, dass diese Aufgaben zur Einschätzung des Beitrags der Software zum Gesamtrisiko nicht durchgeführt werden, führt der Standard die Risiko-Level „hoch“, „beträchtlich“, „mittel“ und „niedrig“ ein, die anhand des SwCI zugeordnet werden.

Department of Defense „Joint Software Systems Safety Engineering Handbook“

Dieses Handbuch beschäftigt sich mit Software-Aspekten in sicherheitskritischen Systemen in allen Phasen eines System-Lebenszyklus. Es beschreibt die Vorgehensweise bei der Entwicklung und Implementierung von effektiven Software-Prozessen in Sicherheitssystemen. Es soll dabei jedoch nicht die vorhandenen Standards ersetzen, sondern vielmehr die vorhandenen Anforderungen an Software in Sicherheitssystemen erläutern und die zur Erfüllung dieser Anforderungen notwendigen Aufgaben spezifizieren.

Zu Beginn werden zunächst verschiedene Modelle für Lebenszyklen von Hardware und Software vorgestellt. Dabei wird hervorgehoben, dass man zwar unterschiedliche Modelle beispielweise für Hardware- und Software-Entwicklung einsetzen kann, es jedoch beispielsweise für einen Software-Entwickler alleine nicht möglich ist, alle wesentlichen Aufgaben des Systems ausreichend detailliert zu betrachten. Alle Aspekte müssen demnach zu einer gemeinsamen Systementwicklung zusammengefügt werden, um die Zuverlässigkeit der sicherheitskritischen Funktion zu gewährleisten.

Im Hinblick auf den Einsatz vorgefertigter Software wird darauf hingewiesen, dass diese Software in der Regel nicht für den Einsatz in sicherheitsrelevanten Systemen entwickelt

wurde und demnach in den meisten Fällen nicht die entsprechenden Anforderungen erfüllen. Vorgefertigte Software bedarf demnach detaillierte Analysen und Prüfungen im Hinblick auf ihre Sicherheitsrelevanz.

Anschließend erfolgt eine Diskussion von möglichen Risiken beim Einsatz von Software in sicherheitsrelevanten Systemen. Dazu werden zunächst die verschiedenen Arten von Risiken definiert und erläutert. Um mit diesen Risiken umgehen zu können, ist ein Risikomanagement notwendig. Die wesentlichen Aufgaben sind hierbei die Durchführung einer Risikoplanung zur Eliminierung bzw. Minimierung von möglichen Auswirkungen und Konsequenzen, einer Risikobewertung zur Identifikation von möglichen Risikobereichen und einer Risikoanalyse zur Bestimmung der Ursachen, Auswirkungen und Stärke potentieller Risiken und Ermittlung möglicher Gegenmaßnahmen.

Abschließend wird auf die Planung und Durchführung der Software-Entwicklung eingegangen. Dabei liegt der Fokus weniger auf technischen Aufgaben, sondern vielmehr bei den notwendigen Führungsaufgaben.

NATO AOP-52 „Guidance on Software Safety Design and Assessment of Munition-Related Computing Systems“

Die AOP-52 der NATO ist eine Richtlinie, die den nationalen Regelwerken wie beispielsweise dem MIL-STD-882 der USA oder dem britischen Def-Stan 00-56 /MOD 07/ nachgeordnet ist.

Die AOP-52 präsentiert Leitlinien für Technik und Management zur Erreichung eines annehmbaren Zuverlässigkeitslevels sowie eines annehmbaren Risikos für die Gesamtsicherheit bei Software, die sicherheitskritische oder sicherheitstechnisch relevante Funktionen ausführt. AOP-52 verfolgt dabei einen Ansatz, der die Software nicht alleingestellt betrachtet, sondern im Kontext des Gesamtsystems, in dem sie eingesetzt wird, einschließlich aller interner und externer Schnittstellen. Dabei werden zunächst Prozesse und Techniken eingeführt, die für die Systemsicherheit und Softwareentwicklung relevant sind.

AOP-52 beschreibt eine Reihe von Dokumenten, die im Prozess der Softwareentwicklung gefordert werden, und gibt Hinweise für ihre Erstellung. Zu diesen Dokumenten zählen insbesondere die vom MIL-STD-882E geforderten Unterlagen zur Risikoeinschätzung.

Ein zentrales Dokument ist der Bericht zur Sicherheitseinschätzung („Safety Assessment Report“), der beschrieben wird als strukturierte Argumentation, die gemeinsam mit einer Gesamtheit von Sicherheitsnachweisen eine umfassende und valide Einschätzung darüber erlaubt, ob ein System für eine vorgegebene Anwendung in einer festgelegten Umgebung sicher betrieben werden kann. Dieser Bericht sollte nach AOP-52 auch eine Beschreibung des Beitrags der Software zum Restrisiko beinhalten. Dieser Bericht und vor allem der zugehörige Sicherheitsnachweis müssen periodisch überarbeitet werden.

Die AOP-52 beschreibt folgende Typen von Nachweisen, die für die Nachweisführung herangezogen werden können:

- Nachweise aus Analysen, beispielsweise um die Reaktion der Software auf bekannte Fehlertypen zu demonstrieren, mögliche failure modes aufzuzeigen, eine worst-case-Abschätzung der Laufzeiten zu ermöglichen oder die Erfüllung von Sicherheitsanforderungen zu belegen.
- Nachweise durch Demonstration wie Betriebserfahrung und Nachweise aus Verifizierung und Validierung.
- Quantitative Nachweise für quantitative Sicherheitsanforderungen.
- Nachweise aus Reviews hinsichtlich Nachvollziehbarkeit, Wartbarkeit, insbesondere auch in Bezug auf zukünftige Modifikation und Verbesserung, Einhaltung von good practices bei Design und Implementierung, Verifizierung der korrekten Implementierung der Sicherheitsanforderungen und Robustheit gegenüber Fehlern von innerhalb und außerhalb des betroffenen Bauteils.
- Qualitative Nachweise für die Umsetzung von good practices im Design

Des Weiteren werden Nachweise für den Prozess als Unterstützung für die oben genannten direkten Nachweise genannt. Hierzu zählen ebenfalls Fachkundenachweise für das beteiligte Personal. Darüber hinaus werden Nachweise darüber gefordert, dass der Entwicklungsprozess in Übereinstimmung mit geeigneten, etablierten Standards sowie unter Anwendung von „good practices“ bzgl. Methoden, Werkzeugen und Technologie durchgeführt wurde.

Die AOP-52 hebt ausdrücklich die Wichtigkeit von Personal für die Softwaresicherheit hervor. Daher werden auch die Identifizierung von Organisationen, Schlüsselrollen und

Verantwortlichkeiten im Rahmen des Prozesses für die Softwaresicherheit ebenso wie Fachkundenachweise der beteiligten Personen gefordert.

Ebenso wird darauf hingewiesen, dass das Systemdesign von vornherein mögliche Auswirkungen von Wartung, Upgrades oder Austausch von Teilen auf die Sicherheitsüberlegungen berücksichtigen sollte.

Die AOP-52 der NATO listet eine Reihe von konkreten Anforderungen an den Softwareentwicklungsprozess auf. Hierzu zählen

- Anforderungen an das Design, wie beispielsweise das Vier-Augen-Prinzip in Bezug auf Design, Coding, Tests und Betrieb von jedem Softwaremodul und die Verwendung von Standalone-Rechnern sofern möglich,
- Anforderungen an die Rechner-Umgebung der entwickelten Software, wie beispielsweise die Schnittstellen zwischen Hard- und Software,
- Anforderungen an die Selbstüberwachung der Softwarebestandteile, wie beispielsweise Watchdogs, Memorychecks und Fehlerdetektion,
- Anforderungen an den Schutz von sicherheitsrelevanten Softwarebestandteilen und Daten, wie beispielsweise die Verhinderung von unautorisiertem Zugriff und unautorisierter Interaktion mit anderen Systemteilen,
- Anforderungen an input/output-Schnittstellen,
- Anforderungen an die Mensch-Maschine-Schnittstelle,
- Anforderungen an Funktionen bzgl. Timing und Interrupt,
- Anforderungen an die Auswahl der Programmiersprache,
- Anforderungen an das Coding wie beispielsweise die Erstellung von modularem Code,
- Anforderungen an die Wartung der Software wie beispielsweise Änderungen an der Firmware und
- Anforderungen für die Analyse und das Testen der Software.

Darüber hinaus beschäftigt sich der AOP-52 mit vorgefertigter Software. Dieser Begriff umfasst hierbei nicht nur kommerzielle, handelsübliche Software („commercial off the

shelf software“, COTS), sondern auch ältere, bereits eingesetzte Software, die nun ein Upgrade erhalten soll sowie wiederverwendbare Software einschließlich ihrer Bibliotheken.

Die Anforderungen und Hinweise für die Analyse und das Testen der Software schließen Anforderungen an neu entwickelte Software ebenso ein wie Anforderungen an vorgefertigte Software.

Defence Standard 00-42 „Reliability and maintainability assurance guides – part 2: software“

Der Defence Standard 00-42 des britischen Verteidigungsministeriums zielt hauptsächlich auf die Zuverlässigkeit von Software ab. Er behandelt sowohl neu entwickelte als auch vorgefertigte und zugekaufte Softwareteile (Commercial-Off-The-Shelf Produkte, COTS), die ohne Modifikation eingesetzt und lediglich neu konfiguriert werden.

Der Standard behandelt vor allem zwei Dokumente, die als die Schlüsseldokumente für das Erreichen von Softwarezuverlässigkeit beschrieben werden, den Software-Zuverlässigkeitsplan (Software Reliability Plan), der vor allem das Software-spezifische Management beschreibt und technische Aufgaben nennt, die durchgeführt werden müssen, sowie den Software-Zuverlässigkeitsnachweis (Software Reliability Case), der parallel zum Verlauf des Projekts entsteht und die Erreichung der geforderten Softwarezuverlässigkeit dokumentiert.

Der Software-Zuverlässigkeitsplan muss Beschreibungen folgender Aspekte enthalten:

- Anforderungen an die Software-Zuverlässigkeit,
- Prozess der Softwareentwicklung,
- Techniken, Methoden und Werkzeuge für die Bewertung der Softwarezuverlässigkeit in jeder Phase des Software-Lebenszyklus,
- Projektspezifische Risikoanalyse für die Software,
- Organisationsstruktur mit Personen und Verantwortlichkeiten im Rahmen der Softwareentwicklung, einschließlich Unterauftragnehmer, und
- Prozeduren für Berichterstattung hinsichtlich der Fortschritte bei der Softwarezuverlässigkeit.

Der Software-Zuverlässigkeitsnachweis wird als lebendes Dokument beschrieben, dem während des Softwarelebenszyklus immer mehr Details hinzugefügt werden. Zu Beginn des Projekts soll er darstellen, dass für die entstehende Software nur ein minimales Risiko besteht, die Zuverlässigkeitsanforderungen nicht zu erfüllen. Während der Softwareentwicklung soll er zeigen, dass das entwickelte Produkt diese Anforderungen erfüllt. Während des späteren Betriebs der Software soll er zeigen, dass die Software zuverlässig ist und auch nach Wartungs- und Instandhaltungsarbeiten zuverlässig bleibt. Der Software-Zuverlässigkeits-Nachweis sollte über folgende drei Phasen des Lebenszyklus der Software weiterentwickelt werden:

- Ausschreibungsphase (Unterlagen zur Rechtfertigung von Designentscheidungen, die sich im Angebot widerspiegeln),
- Entwicklungsphase (Nachweise darüber, dass die entwickelte Software den Anforderungen genügt), und
- Betriebsphase (Unterlagen zur Betriebserfahrung, Analyse der Auswirkungen möglicher Softwarefehler und möglichen Softwareversagens).

Sowohl der Software-Zuverlässigkeitsplan als auch der Software-Zuverlässigkeitsnachweis werden im Defence Standard 00-42 ausführlich behandelt. Beispielsweise werden Hinweise zur Erstellung, zur Strukturierung und zur Einschätzung der Dokumente gegeben.

Defence Standard 00-55 „Requirements for safety related software in defence equipment – part 1: requirements; part 2: guidance“

Teil 1 und 2 des Standards beschäftigen sich mit Anforderungen an Prozeduren und technischen Umsetzungen bei der Entwicklung von sicherheitsrelevanter Software, wobei der Schwerpunkt bei Werkzeugen und Hilfs-Software zur Entwicklung, Prüfung, Zertifizierung und Wartung der sicherheitsrelevanten Software liegt. Teil 2 liefert dabei vertiefende Informationen und Erläuterungen zu den in Teil 1 definierten Anforderungen. Sie werden daher inhaltlich im Folgenden nicht getrennt behandelt.

Zunächst wird in dem Standard der Begriff der sicherheitsrelevanten Software definiert. Demnach handelt es sich um Software, welche in sicherheitsrelevanten Systemen eingesetzt wird und alle Sicherheitsintegritätslevel umfasst. Davon unterschieden wird sicherheitskritische Software, welche ausschließlich auf SIL 4 eingesetzt wird.

Es werden die Themen Sicherheitsmanagement, personelle Verantwortlichkeiten, Planungs- und Entwicklungsprozess von sicherheitsrelevanter Software sowie Zertifizierung und Einsatz im Betrieb behandelt. Abschließend wird noch auf die Unterschiede der Software-Anforderungen in den verschiedenen SIL-Stufen eingegangen.

Zunächst werden die verschiedenen Aufgaben des Sicherheitsmanagements aufgeführt. Hierzu zählen u.a. die Erstellung eines Software-Sicherheitskonzepts, eines Software-Sicherheitsnachweises, welcher auch ausreichend protokolliert und dokumentiert sein muss, sowie die Durchführung einer Sicherheitsanalyse der sicherheitsrelevanten Software und des entsprechenden Entwicklungsprozesses. Der Software-Sicherheitsnachweis sowie die Sicherheitsanalyse werden dabei als Teil des Sicherheitsnachweises des Gesamtsystems betrachtet.

Anschließend werden die Aufgaben und Verantwortlichkeiten der verschiedenen am Entwicklungsprozess beteiligten Personen und Gruppen definiert.

Um die Qualitätssicherung bei der Software-Entwicklung zu gewährleisten, soll ein zertifiziertes Qualitätssicherungssystem angewandt werden. Im Rahmen dessen sollen die verschiedenen Arbeitsschritte zur Software-Entwicklung sowie zur Verifizierung und Validierung detailliert geplant werden. Es werden verschiedene Anforderungen an die einzelnen Planungsschritte formuliert. Hierbei wird beispielsweise auf die Auswahl von Entwicklungsmethoden, der Programmiersprache, der eingesetzten Werkzeuge bei der Entwicklung sowie der vorgefertigten Software eingegangen.

Während des Entwicklungsprozesses sollen alle ausgeführten Aufgaben dokumentiert werden. Weiterhin werden Anforderungen an die Software-Spezifikation und im weiteren Verlauf an die Verifizierung sowie die Prüfung und Integration der Software gegeben.

C Modelle zur Software-Entwicklung

Bei der Entwicklung einer Software wird ein sogenanntes Vorgehensmodell verwendet. Dieses definiert einen Rahmen für den organisatorischen Prozess der Softwareerstellung. Darin werden die folgenden Punkte festgelegt /FAE 09/:

- durchzuführende Aktivitäten
- Abfolge der Arbeiten (Entwicklungsstufen, Phasen)
- Definition der Teilprodukte / Ergebnisse
- Definition der Kriterien für die Fertigstellung
- Verantwortlichkeiten und Kompetenzen
- Erforderliche Mitarbeiterqualifikation
- zu berücksichtigende Standards, Richtlinien, Methoden und Werkzeuge

Das Ziel aller Vorgehensmodelle ist es, die Softwareprojekte durch eine klar definierte, strukturierte und standardisierte Vorgehensweise zu planen. Dabei soll auch die Zeit, das Budget und die Qualität der Software kontrolliert werden. Auf diese Weise soll der gesamte Entwicklungsprozess optimiert werden /FAE 09/.

Vorgehensmodelle, die die Softwareentwicklung stark in Phasen einteilen, werden auch Phasenmodelle genannt. Im Folgenden werden die Vor- und Nachteile von drei unterschiedlichen, weit verbreiten Vorgehensmodellen beschrieben /FAE 09/.

C.1 Wasserfallmodell

Das Wasserfallmodell besteht aus vielen hintereinanderliegenden Phasen, die Top-down in der richtigen Reihenfolge durchlaufen werden müssen. Erst nach Abschluss einer Phase, die immer auch eine Dokumentation beinhaltet, beginnt die nächste Phase. Die erste Phase beinhaltet eine Analyse des Systems sowie die Definition der Systemanforderungen. In der nächsten Phase wird aus den Anforderungen ein Systementwurf erstellt, so dass in der darauffolgenden Phase das System implementiert werden kann. Schließlich folgt eine Testphase und im Anschluss daran der Einsatz und die Wartung des Systems /KOO 16/, /KOS 13/.

Das Wasserfallmodell ist ein einfaches und verständliches Modell, das eine leichte Fortschrittskontrolle ermöglicht. Es besitzt eine hohe Qualität der Dokumentation und benötigt wenig Managementaufwand /KOS 13/.

Ein Nachteil des Modells liegt jedoch darin, dass die vollständige oder sequentielle Bearbeitung der Softwareentwicklungsschritte nicht immer sinnvoll ist. Änderungen von Anforderungen lassen sich beispielsweise in einer späteren Phase nicht mehr integrieren, so dass die gesamte Entwicklung neu begonnen werden muss. Nachteilig am Wasserfallmodell ist auch das Fehlen eines durchgängigen Qualitätssicherungsprozesses /FAE 09/.

C.2 V-Modell

Das V-Modell stellt eine Erweiterung des Wasserfall-Modells durch Integration einer expliziten Qualitätssicherung dar. Wie im Wasserfallmodell werden die Produkte im V-Modell streng sequentiell erstellt. Hinzu kommen jedoch entsprechende Tests zur Verifizierung und Validierung. In /KOO 16/ sind die einzelnen Phasen des V-Modells sowie die Test- und Validierungsaktivitäten dargestellt.

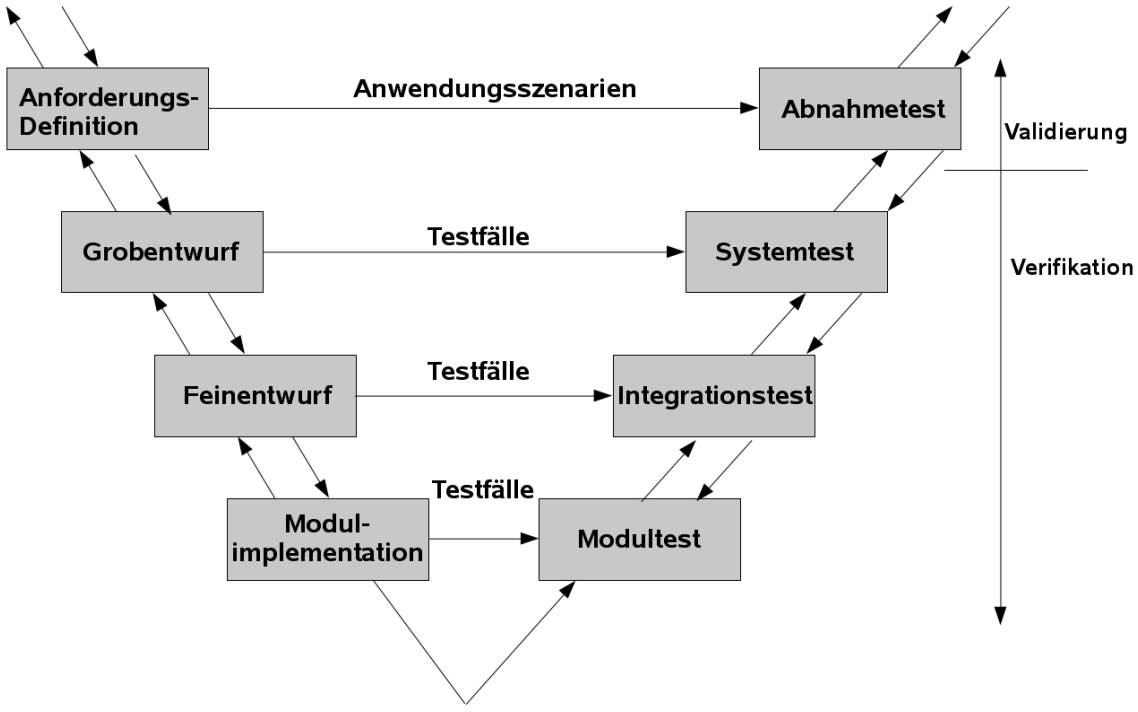

Abb. C.1 Phasen des V-Modells /KOO 16/

C.3 Agile Vorgehensweise, Scrum

Aufgrund der geltenden Standards und Normen sind die Rahmenbedingungen für die Entwicklung von Software eng gefasst. Um die Entwicklungszyklen trotz dieser strengen Vorgaben möglichst kurz zu halten und flexibel auf Anforderungsänderungen reagieren zu können wurde die sogenannte agile Softwareentwicklung eingeführt. Ein Beispiel für ein solches agiles Rahmenprogramm (Framework) für das Projekt- und Produktmanagement ist das sogenannte Scrum Vorgehensmodell. Dabei handelt es sich um eine Methode, bei der die Definition der Systemanforderungen, der Systementwurf, die Implementierung und das Testen iterativ und inkrementell erfolgen. Das führt dazu, dass die Produktentwicklung besonders flexibel und schnell bei der Realisation neuer Softwareinkremente ist. Als nachteilig für die Entwicklung sicherheitsrelevanter Software, wird bei Scrum die weniger umfangreiche Dokumentation angesehen.

Ob agile Prozesse und Methoden normenkonform angewendet werden können, wird in der Literatur vielfach diskutiert. Wie in Abschnitt 4.2 beschrieben wird in den Normen zwar nicht festgelegt, welche Vorgehensmodelle verwendet werden sollen. In manchen Normen, wie z.B. der DIN EN 50128, werden jedoch Dokumente gefordert, die sich aus bestimmten Vorgehensmodellen ergeben.

Erfolgreich angewendet wird Scrum bzw. ein mit weiteren Sicherheitsanforderungen erweitertes Scrum im Schienenverkehr und der Öl- und Gasindustrie /MYK 15/. Auch für medizinische Software ist laut /BLE 14/ eine normenkonforme Anwendung mit der IEC 62304 möglich.

Im Standard IEC 61508-3 /DIN 61d/ wird die Erfüllung der Anforderungen an den Softwarelebenszyklus gefordert, wobei das zu verwendende Vorgehensmodell nicht festgelegt wird. Die Anforderungen sind dabei jedoch so gestellt, das nur das V-Modell als Vorgehensmodell verwendet werden kann. Laut Konferenzvortrag /MYK 15/ wird die Norm Standard IEC 61508-3 /DIN 61d/ so überarbeitet werden, dass als mögliche Methoden sowohl das V-Modell als auch das Scrum Modell normenkonform verwendet werden können.

Im Schienenverkehr und der Öl- und Gasindustrie wird SafeScrum vom TÜV Nord und TÜV Rheinland akzeptiert. Auch andere Zertifizierungsstellen, wie beispielsweise TÜV Süd und Lloyds akzeptieren Agile Entwicklungsmethoden.

D Beschreibung von Methoden zur Zuverlässigkeitsbewertung

D.1 Software Failure Modes and Effects Analysis (SFMEA)

Die FMEA wird mit den folgenden Zielen definiert:

- Erkennen und Bewerten aller unerwünschten Auswirkungen der Ausfallarten und der resultierenden Folgeereignisse.
- Analyse der Ausfallarten hinsichtlich der Funktions- und Leistungsfähigkeit des gesamten Systems und Abschätzung der Dringlichkeit der Fehlerbehebung.
- Klassifizierung der entdeckten Ausfallarten nach relevanten Eigenschaften (z.B. Erkennbarkeit, Diagnosefähigkeit, Prüfbarkeit, Vorkehrungen für Einsatz, Reparatur, Instandhaltung etc.).
- Abschätzung der Ausfallschwere und -wahrscheinlichkeit nach der Identifizierung von Funktionsausfällen des Systems.
- Verringerung der Ausfallarten durch Aufstellen eines Entwurfsverbesserungsplans
- Verringerung der Ausfallwahrscheinlichkeit durch Entwicklung eines Instandhaltungsplans /IEC 01/.

Dabei gliedert sich die FMEA in folgende Schritte /IEC 01/:

- Festlegung der Systemgrenzen für die Analyse
- Verstehen der Forderungen an das System und seine Funktion
- Festlegung von Kriterien für Ausfall und Erfolg
- Ermitteln und Aufzeichnen der Ausfallarten für jede Einheit sowie deren Ausfallwirkung
- Zusammenfassung aller Versagensauswirkungen
- Bericht der Ergebnisse

Die FMEA wird in einem Aufgabenblatt dokumentiert.

Je nach FMEA-Anwendung kann auch noch eine Klassifizierung der Schwere *S,* d.h. die Klassifizierung der Auswirkungen einer Ausfallart erfolgen. Mögliche Faktoren für die Klassifikation der Schwere können Auswirkungen auf Benutzer und Umwelt, funktionales Verhalten des Systems/ Prozesses, allgemeine Sicherheitsforderungen etc. sein. Da die Schwere eines Ausfalls nur in Verbindung mit der Eintrittshäufigkeit oder Eintrittswahrscheinlichkeit *P* angemessen beurteilt werden kann, sollte auch die Eintrittshäufigkeit bestimmt werden. Die Eintrittswahrscheinlichkeit kann (bezogen auf ein bestimmtes Zeitintervall) aus den Daten der Lebensdauerprüfung der Komponenten, vorhandenen Ausfallratendatenbanken und anderen Ausfalldaten geschätzt werden.

D.2 Software Failure Modes, Effects and Criticality Analysis (SFMECA)

Bei der FMECA wird zudem die Kritizität (Bedeutung) als Merkmal für das Ausmaß einer Ausfallartauswirkung bestimmt. Dabei bestimmt sich die Ausfall-Kritizität aus der Kombination der Schwere einer Auswirkung und der Eintrittshäufigkeit oder anderer Eigenschaften eines Ausfalls. Die Kritizität ist eine nicht allgemeingültig festgelegte Metrik, um zu einer quantitativen Entscheidungsfindung bei der Fehlerbehandlung zu gelangen und Prioritäten bei der Milderung der Auswirkungen zu setzen. Das Risiko *R* wird berechnet aus der Schwere S und der Eintrittswahrscheinlichkeit P:

$$R = S \cdot P$$

Dabei wird der Schweregrad nach den oben beschriebenen Faktoren abgeschätzt, so dass es sich um einen dimensionslosen Schätzwert handelt.

Zudem wird die Risikoprioritätszahl *RPN* definiert, die noch einen dritten Aspekt der Fehleranalyse berücksichtigt: die Wahrscheinlichkeit der Fehlererkennung und Behebung, bevor der Fehler zu einem Systemausfall führt. Die Fehlererkennungszahl *D* fließt als Schätzwert in die Berechnung der Risikoprioritätszahl ein und ermöglicht es eine Reihenfolge für die Fehlerentschärfung zu bestimmen. Sie ist definiert als:

$$RPN = S \cdot O \cdot D$$

wobei *O* die Eintrittswahrscheinlichkeit für die Ausfallart innerhalb eines festgelegten Zeitraums ist. Je höher die Fehlerentdeckungszahl *D* ist, desto unwahrscheinlicher ist ihre Erkennung. In den unterschiedlichen Standards der FMECA werden den Werten

von *S*, *O* und *D* unterschiedliche Skalen zugeordnet (beispielsweise Schweregrad I-IV siehe Abb. D.1)

Neben dem beschriebenen Konzept der Risikoprioritätszahl, lässt sich die Kritizität auch in einer Matrix darstellen (siehe Abb. D.1). Mit zunehmender Schwere und Eintrittswahrscheinlichkeit steigt das Risiko für ein katastrophales Versagen. Die Entscheidung, ob Ausfallart 1 oder Ausfallart 2 bei einer Fehlerbehandlung priorisiert werden sollte, hängt von der Skalierung der Schwere- und Häufigkeitsklassen ab und der Priorisierung von Schwere und Häufigkeit.

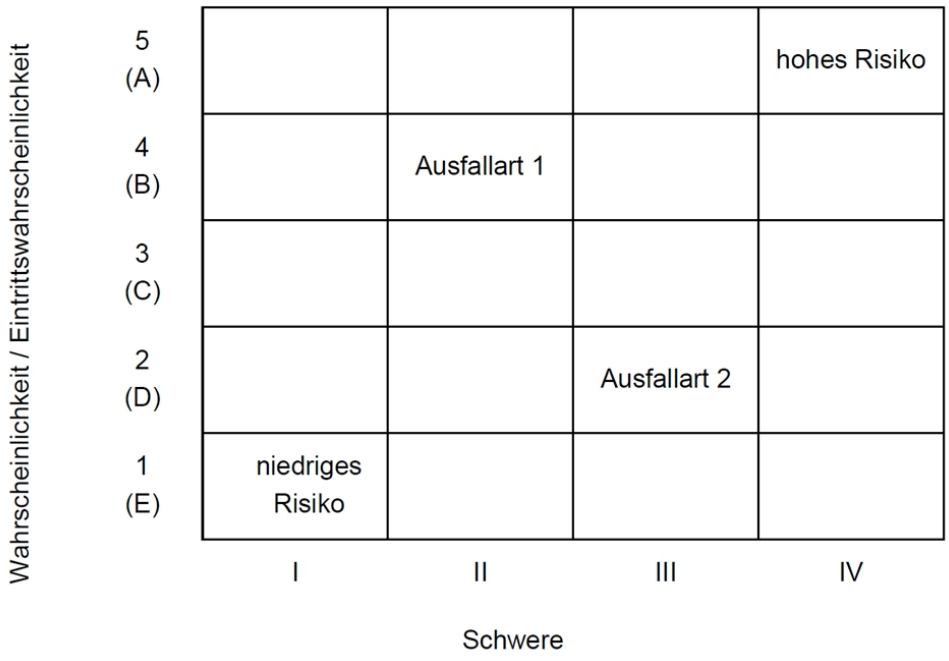

Abb. D.1 Kritizitäts-Matrix

Bei der Anwendung der FMECA auf Software nach dem Bottom-up Ansatz werden zunächst die Funktionen der Anwendungsschicht untersucht, da die tieferliegenden Ebenen in der Praxis zu kompliziert für eine Analyse sind. Nachdem eine geeignete Aufgliederung des Systems in seine Elemente erfolgt ist, werden die Ausfallarten bestimmt. Im Unterschied zur Hardware, bei der sich Ausfallarten und Ausfallwahrscheinlichkeiten bestimmter Komponenten auf Basis der Betriebserfahrung unkompliziert ermitteln lassen, sind die Ausfallarten für Softwarefehler nicht genau bekannt. Da Software jedoch kein probabilistisches Versagen aufgrund von Alterung besitzt, sondern ein deterministisches Verhalten verursacht durch latente Fehler, ist eine genaue Kenntnis der Ausfallart nicht gegeben. Eine genaue Kenntnis würde zu einer direkten Fehlerkorrektur führen.

Daher werden bei genauer Kenntnis der Software allgemeingültige, häufig eintretende Fehlerarten prognostiziert und untersucht, wie z. B.:

- Computerbasierte Fehler
- Logikfehler
- Datenverarbeitungsfehler
- Interface Fehler
- Fehler in der Datendefinition
- Fehler in der Datenbank

Diese Fehler können weiter analysiert werden, hinsichtlich ihrer möglichen Ursachen und ihrer Auswirkungen (z.B. Hardware- oder Softwareabsturz, Fehlermeldungen, lange Ansprechzeiten etc.)

Für die Bestimmung der Kritizität müssen die Schwere und die Eintrittshäufigkeit des Softwarefehlers bestimmt werden. Die Schwere eines Softwarefehlers in einem automatisierten Systems lässt sich über die Auswirkungen auf das Sicherheits- und Kontrollsystem bei Versagen bestimmen. Die Auftrittshäufigkeit eines Softwarefehlers ist jedoch normalerweise unbekannt, da das Auslösen eines Fehlers nicht nur von der Software abhängt, sondern auch von der Laufzeit und der Umgebung. Einige der Analysten verwenden daher zur Bestimmung der Fehlereintrittswahrscheinlichkeit Komplexitätsmetriken wie beispielsweise die McCabe Komplexität oder die Halstead Komplexität (siehe Abschnitt 6.2.1).

D.3 Software Fault Tree Analysis (SFTA)

Die Software Fault Tree Analysis (SFTA) ist ein Verfahren, das sich an der für Hardware entwickelten Fehlerbaumanalyse (Fault Tree Analysis, FTA) orientiert.

Die Fehlerbaumanalyse ist eine bewährte Art der Systemanalyse, die sich einer deduktiven logischen Vorgehensweise bedient. Ziel der Analyse ist es die Fehlerfortpflanzung innerhalb eines redundanten Systems zu analysieren. Ihre Methodik ist in DIN 25424 /DIN 81/ und IEC 61025 /IEC 06/ beschrieben.

Die Fehlerbaumanalyse ermöglicht es, das zu betrachtende System in einem Modell abzubilden, welches die logischen und quantitativen Wechselwirkungen und Zusammenhänge in einem komplexen technischen System deutlich macht.

Zunächst wird die Systemgrenze festgelegt (d.h. die Definition der Gefährdung, die zu berücksichtigenden Ereignisse sowie der initiale Systemzustand). Die eigentliche Analyse beginnt mit einem Ausfall-Szenario des gesamten Systems oder auch einer Systemfunktion (TOP-Event), ausgelöst durch ein unerwünschtes Ereignis (Top-down Analyse) /LYU 96/. Es wird dann untersucht, welche Ausfallarten der Komponenten zu dem unerwünschten Ereignis geführt haben können. Dabei ergibt sich zunächst die minimale Schnittmenge (minimal cut sets) an Komponenten, deren Versagen, den Ausfall des Gesamtsystems verschuldet haben könnte, wobei redundante Systeme nicht mehrfach berücksichtigt werden. Die Schnittmenge wird dann mit jedem niedrigeren Level vergrößert, bis die eigentlich ursächlichen Ereignisse gefunden werden. Bei der qualitativen Analyse werden die minimalen Schnittmengen untersucht, d.h. die kleinste Menge an Basisereignissen, die zusammen zu einem Ausfall führen. So lässt sich feststellen, ob es einzelne unerwünschte Ereignisse gibt, die zum Ausfall des gesamten Systems führen können /LYU 96/.

Wenn die Ausfallwahrscheinlichkeiten der Basisereignisse bekannt sind, kann anhand des Fehlerbaums die Wahrscheinlichkeit des TOP-Events ermittelt und so die Fehlerbaumanalyse auch zu einer quantitativen Sicherheitsbewertung eingesetzt werden.

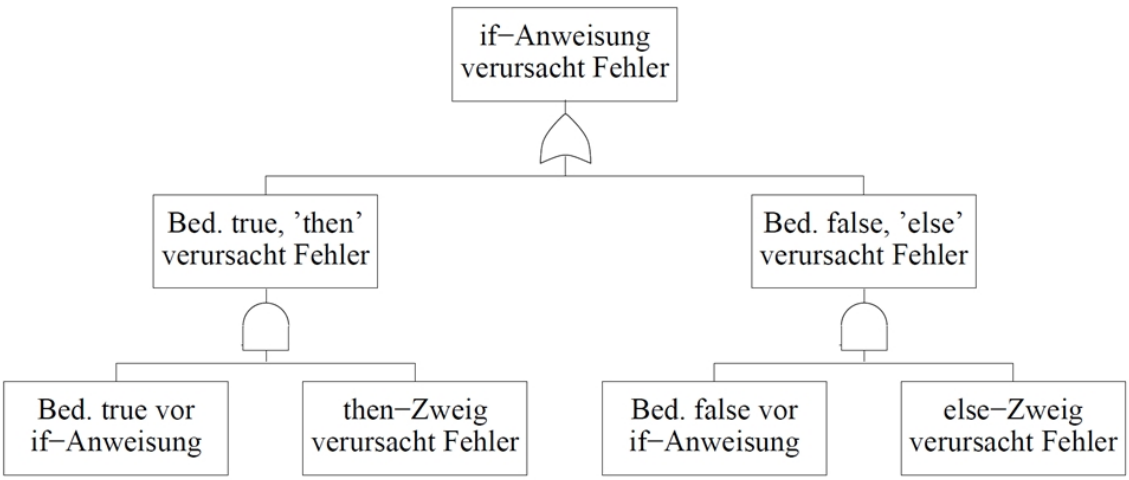

Abb. D.2 Muster eines Fehlerbaums für eine if-Anweisung. Die Ereignisse werden entsprechend ihrer Logik mit Und- oder Oder-Gattern verknüpft. Entnommen aus /THU 04/

D.4 Software Common Cause Analysis (SCCA)

Die Software Common Cause Analysis untersucht mögliche Ketten von Ereignissen und Bedingungen, die zu koinzidentem Softwareversagen mehrerer redundanter Teilsysteme führen. Dazu werden auslösende Ereignisse definiert, die möglicherweise zu GVA führen /SAG 07/.

Software, die durch solche Auslöser zu einem sicherheitskritischen Systemverhalten führt, muss drei Bedingungen erfüllen /SAG 07/:

- I – Impact (Auswirkung): Der potentielle Auslöser für GVA muss die Software in dem betrachteten Operationsmodus beeinflussen können.
- A – Avoidance (Vermeidung): Die Software darf während des Entwicklungsprozesses hinsichtlich des GVA-Auslösers nicht analysiert oder getestet worden sein.
- M – Mastering (Beherrschung): Die Software enthält keine Maßnahmen, die die negativen Auswirkungen des GVA-Verursachers noch während der Laufzeit kontrollieren oder entschärfen (wie beispielsweise logische Programmablaufüberwachung, Plausibilitätskontrolle und Daten-/Zeitredundanz).

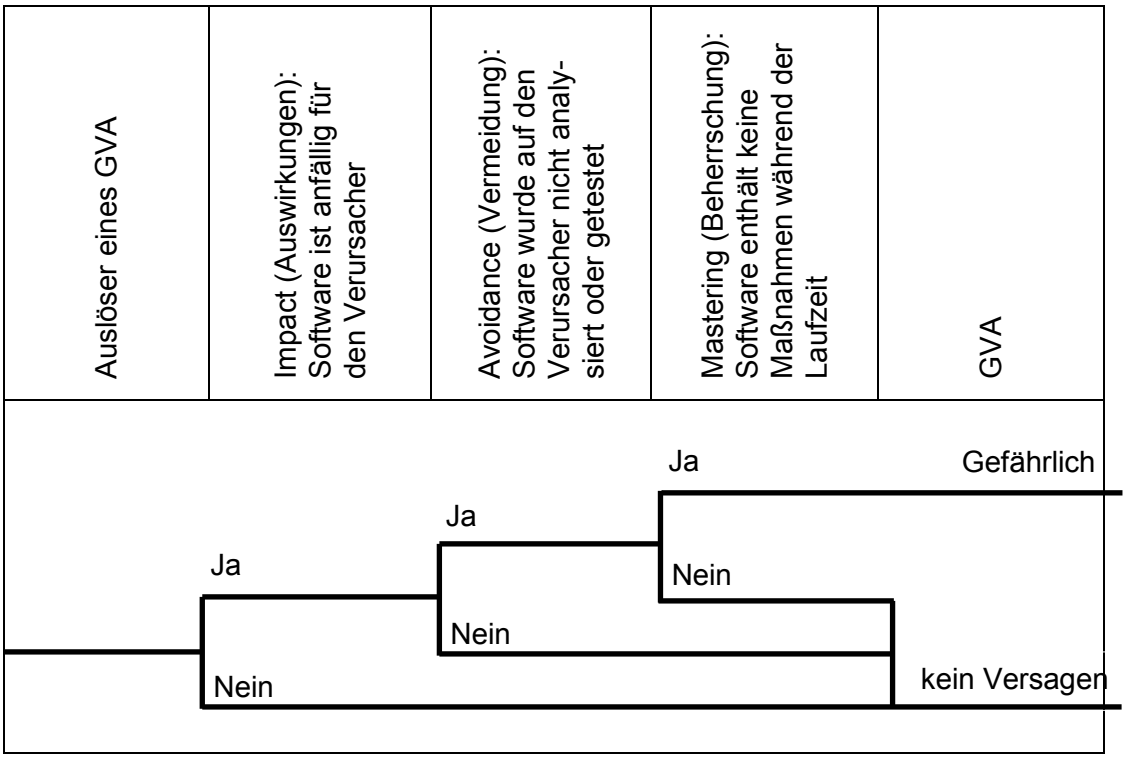

Abb. D.3 Ereignisfluss vom Auslöser bis zum Systemversagen nach /SAG 07/

Alternativ können in den Ereignisbaum auch Wahrscheinlichkeiten eingetragen werden.

Die Unabhängigkeit der Bedingungen ist bei der Modellierung des Ereignisbaums wichtig. Auf diese Weise werden unterschiedliche Aspekte der Software bzw. der Softwareentwicklung analysiert. Die Auswirkungen (Impact) werden durch die Eingangsgrößen der Software und durch die Wechselwirkung mit anderen Systemen bestimmt, die Vermeidung (Avoidance) von Softwarefehlern ist eine Frage der Softwareentwicklungszeit, die Fehler Beherrschung (Mastering) bezieht sich auf die Laufzeit des Programms. Eine weitere Verfeinerung der Bedingungen ist nur möglich, wenn diese Unabhängigkeit bewahrt wird /SAG 07/.

Um Ereignisse ausfindig zu machen, die zu Ausfällen aufgrund gemeinsamer Ursache führen können, wird nach dem Konzept „Einflussfaktoren versus Stärke“ (Stress versus Strength) vorgegangen. Nach diesem Konzept werden Komponenten vor allem vor den häufig Auftretenden externen Beanspruchungen geschützt. Wenn diese Faktoren stärker sind als angenommen oder wenn nicht berücksichtigte Beanspruchungen auftreten, kann es bei redundanten Komponenten zu GVA kommen /SAG 07/.

In Tab. D.1 werden Auslöser aufgeführt, welche die Software ungünstig beeinflussen können, jedoch selten in der Softwarebewertung und in Softwaretest untersucht werden. Dabei berücksichtigt die Tabelle Anforderungen des Standards IEC 62430 /IEC 05/ (Kapitel 8, 9.3 und 9.6). Zusätzlich werden nach IEC 61508 /DIN 61a-h/ Empfehlungen für die Prüfung der Auslöser für die unterschiedlichen SIL-Klassifizierungen gegeben /SAG 07/:

DE: Es wird dringend empfohlen den Auslöser eines GVA für das entsprechende Sicherheits-Integritätslevel (SIL) zu untersuchen.

E: Es wird empfohlen den Auslöser eines GVA für das entsprechende Sicherheits-Integritätslevel (SIL) zu berücksichtigen.

Mit der Software Common Cause Analysis lassen sich einerseits Auslöser identifizieren, deren Auftrittswahrscheinlichkeiten nicht vernachlässigbar gering sind und die nicht ausreichend abgesichert sind. Andererseits ermöglicht sie auch zu zeigen, dass die Wahrscheinlichkeit für einen Ausfall aufgrund gemeinsamer Ursache „ausreichend gering“ ist, verglichen mit einem Ausfall bei Anforderung.

Tab. D.1 Liste der auslösenden Ereignisse nach /SAG 07/

Auslöser **SIL**	**1**	**2**	**3**	**4**
Auslöser durch Signale und Daten				
Einfluss durch nicht getestete Kombinationen von Eingangssignalen	--	E	E	DE
Einfluss durch nicht getestete Signalformen von Eingangssignalen (Störsignale, ungetestete Frequenzspektren)	--	E	E	DE
Überschreiten von Grenzen	E	DE	DE	DE
Inkorrekte Kommunikationsdaten	E	DE	DE	DE
Auslöser durch mangelnde Unabhängigkeit				
Mangelnde Unabhängigkeit der Daten sicherheitsrelevanter und nicht-sicherheitsrelevanter Programmteile	--	DE	DE	DE
Mangelnde zeitliche Unabhängigkeit der sicherheitsrelevanten und nicht-sicherheitsrelevanten Programmteile	--	DE	DE	DE
Mangelnde Unabhängigkeit beim Verarbeiten unterschiedlicher Sicherheitsfunktionen	--	E	DE	DE
Unverfügbarkeit von mehreren Sicherheitsfunktionen bei fehlerhaften Eingangssignalen einer einzelnen Sicherheitsfunktion	--	E	DE	DE
Unerlaubte Zeiger	--	E	DE	DE
Auslöser aufgrund zeitlicher Einflussfaktoren				
Nicht-deterministisches oder nicht überwachtes zeitliches Verhalten				
Unterbrechungsgesteuerte statt periodische Verarbeitung				
Unterbrechungen in Abhängigkeit der Prozessdaten	--	E	DE	DE
Unterbrechungsgesteuerte Zeitplanung ohne logische und zeitliche Programmflussüberwachungen	E	DE	DE	DE
Ereignisorientierte Kommunikation von sicherheitsrelevanten Daten	--	E	DE	DE
Beeinflussung von nicht betrachteten Kombinationen von Eingangsdaten/ Kommunikationsereignisse/ Unterbrechungen	--	E	DE	DE

Auslöser **SIL**	**1**	**2**	**3**	**4**
Auswertung des zeitlichen Verhaltens des Aufbaus des Quellcodes (z.B. Programm Schleifen)	--	E	DE	DE
Verletzungen der ungünstigsten Zeit-/Frequenz-Betrachtungen (z.B. Abtasttheorem, Interrupt-Theorie)	E	DE	DE	DE
Asynchroner Zugriff auf gemeinsame Ressourcen	--	E	E	DE
Bearbeitung in Abhängigkeit des Kalenders	--	E	DE	DE
	--	E	E	DE
Auslöser aufgrund menschlichen Fehlverhaltens				
Falsche Parametrisierung aufgrund der Komplexität oder Mehrdeutigkeit der Parameter	DE	DE	DE	DE
Störung in der Parametrisierung	--	E	DE	DE
Unzulässige Änderung der Software oder der Parametrisierung	--	E	E	DE
Auslöser aufgrund der Programmierung				
Gefährlicher Aufbau des Quellcodes (Allgemein: zahlreiche Verwendung von globalen Variablen, Rekursionsformeln, plattformabhängige Datentypen etc.)	E	DE	DE	DE
Gefährlicher Aufbau des Quellcodes (Programmiersprachen spezifisch: z.B. Komplexe Zeiger, Unions, Macros etc.)	E	DE	DE	DE
Nicht abgefangene Laufzeitfehler (Überlauf, Teilen durch Null, ungültiger Zeigerzugriff)	E	E	DE	DE
Veränderte Speicherauslastung	E	DE	DE	DE
Dynamische Objekte explizit erzeugt während der Laufzeit der Anwendersoftware				
Dynamische Objekte implizit erzeugt während der Laufzeit (z.B. Bibliotheksfunktionen mit Speicherzuordnung, Bibliotheksfunktionen für Listen, Queue, Heap etc.)				

Laut IEC 61508-7 /DIN 61h/ kann die Wahrscheinlichkeit für ein auslösendes Ereignis, unter der Voraussetzung, dass N statistisch unabhängige Tests mit Eingangsdaten ähnlich dem Betriebsmodus durchgeführt wurden und dabei kein auslösendes Ereignis aufgetreten ist, abgeschätzt werden. Dabei wird das sogenannte Konfidenzniveau (confidence level) (1- β) bereits vorher festgelegt. Es gilt dann die Abschätzung

$$P_{Auslöser} \leq -\frac{\ln(1-\beta)}{N}$$

Laut IEC 61508 /DIN 61a-h/ sollte die Wahrscheinlichkeit, dass ein einzelner Auslöser einen gefährlichen GVA auslöst, nicht größer sein als 1% der Wahrscheinlichkeit eines Ausfalls bei Anforderung sein. Dabei werden auch die Methoden zur Vermeidung (Avoidance) und Beherrschung (Mastering) mitberücksichtigt und ihre Wirksamkeit mit einem Effektivitätsfaktor quantifiziert.

D.5 Dynamic Flowgraph Methodology (DFM)

Das Dynamic Flowgraph Methodology (DFM) Modell ist ein graphisches Netzwerk, das die zentralen Prozessparameter über kausale Zusammenhänge sowie deren zeitab-hängige Beziehungen miteinander verknüpft. Auf diese Weise lassen sich Ereignis-kombinationen ermitteln, um Fehlersituationen und Versagensereignisse des Systems zu modellieren. Die Methode, die sowohl auf rechnerbasierte und programmierbare Sys-teme als auch auf Hardware anwendbar ist, stellt eine mehrstufige und zeitabhängige Erweiterung zur Fault Tree Analysis (FTA) und zur Failure Modes and Effects Analysis (FMEA) dar. Der Vorteil der Methode liegt darin, dass ein DFM-Modell alle notwendigen Informationen enthält, so dass sich die Analyse des Systems automatisiert durchführen lässt /NUR 96/.

Im DFM-Modell werden die wichtigsten Parameter des Kontrollsystems, die Softwarevariablen, die das Verhalten der physikalischen Komponenten bestimmen und Software/Firmware-Funktionen als „Prozess-Variablen“ identifiziert und als Knotenpunkte dargestellt. Die Systemzustände werden dann als Folge zeitlich diskreter Übergänge modelliert. Das Netzwerk kann dann entweder induktiv oder deduktiv analysiert werden. Bei der induktiven Analyse werden ausgehend von einem Anfangszustand die mit fortschreitender Zeit sich ergebenden Zustände analysiert. Die bei dieser Analyse entste-

henden Netzwerke gleichen Ereignisbäumen, die jedoch nicht auf binäre Variablen beschränkt sind. Bei der deduktiven Analyse wird das System „rückwärts“ in der Zeit betrachtet, ausgehend von einem Top-Ereignis des Netzwerks (Fehlersituation) werden die Ereigniskombinationen ermittelt, die zu dem vorgegebenen Zustand geführt haben können. Es wird ein (zeitabhängiger) Fehlerbaum konstruiert, der zum Systemzustand geführt hat. Ähnlich den Minimaltermen einer Fehlerbaumanalyse, lassen sich dann sogenannte Primimplikanten (Prime Implicants) modellieren. Diese Primimplikanten sind Kombinationen von Systemzuständen, die zu einer Fehlersituation geführt haben können. Die Primimplikanten enthalten somit Informationen darüber, zu welchem Zeitpunkt sich ein Systemteil in einem bestimmten Zustand befunden haben muss /BNL 10/, /GRS 15d/.

Die Entwicklung der Fehlerbaumanalyse erfolgt Top-Down, d.h. dass die zu berücksichtigen Systeme ausgehend von einem groben Schema immer weiter verfeinert werden, bis der nötige Detaillierungsgrad erreicht wird /GRS 15d/.

Wie bei der SFMECA und der SFTA Methode wird auch bei der Anwendung der DFM vorausgesetzt, dass das System von einem Experten analysiert wird, der den Detaillierungsgrad und die Abhängigkeit der Subsysteme bzw. Systemkomponenten kennt.

D.6 Stochastische Modelle auf Grundlage von Ausfalldaten

D.6.1 Allgemein anwendbare statistische Methoden

Allgemein anwendbare statistische Methoden sind testbasierte Methoden. Diese Standardmethoden werden auf Softwaretests und soweit möglich auf Daten aus der Betriebserfahrung angewendet. Diese Vorgehensweise ist analog zur Analyse von Hardwaredaten. Allerdings können beim Testen von Hardwarekomponenten ähnliche Tests vielfach wiederholt und anschließend die Zahl der Ausfälle pro Anforderung oder pro Zeit ausgewertet werden. Dies ist beim Testen von Softwarekomponenten jedoch nicht möglich, da derselbe Test ohne Änderungen an der Software immer dasselbe Ergebnis liefert. Daher müssen die Testfälle beim Testen von Softwarekomponenten voneinander verschieden sein und möglichst den gesamten Input-Bereich der Software bzw. des rechnerbasierten oder programmierbaren Systems abdecken /BNL 10/.

Grundsätzlich können testbasierte Methoden für die Abschätzung der Wahrscheinlichkeit einer Fehlauslösung oder eines Ausfalls bei Anforderung eingesetzt werden. Die Anwendung einer testbasierten Methode umfasst die Generierung von Testfällen, die Durchführung von Tests und die Abschätzung der Ausfallrate /BNL 10/.

Die wichtigsten Typen von test-basierten Methoden sind das sogenannte White-Box Testing und das sogenannte Black-Box Testing. Der Unterschied zwischen diesen beiden Typen liegt darin, dass beim White-Box Testing Informationen über die interne Struktur der Software zur Ableitung der Testfälle verwendet werden. Beim Black-Box Testing wird der Output, den das System bei zufällig ausgewählten Input-Daten liefert, auf Richtigkeit überprüft und die Ergebnisse aller Tests mit statistischen Standardmethoden analysiert. Beim Black-Box Testing wird also das System als Ganzes betrachtet, während das White-Box Testing Informationen über einzelne Softwarekomponenten liefern kann /BNL 10/.

Aufgrund der großen Anzahl an Softwarekomponenten kann sich das White-Box Testing in der Praxis als zu aufwendig erweisen /NRC 11/.

Bei allen Methoden, die auf Testdaten zurückgreifen, stellt sich grundsätzlich die Frage, ob die Testumgebung, in der die Testdaten gesammelt werden, die tatsächliche Betriebsumgebung ausreichend genau abbildet. Eine weitere Schwierigkeit, die ebenfalls viele Methoden gemeinsam haben, besteht darin, dass das Testen der Software die eventuell vorhandenen Fehler in den Anforderungen an die Software bzw. deren Spezifikationen oft nicht zeigen kann /BNL 10/.

Testbasierte Methoden zur Zuverlässigkeitsbewertung müssen auf einer sehr großen Anzahl Testdaten basieren, um sehr kleine Ausfallraten nachzuweisen. Bei einer geforderten Ausfallwahrscheinlichkeit einer Software von 10^{-5} darf typischerweise in einer Größenordnung von 10^5 bis 10^6 (variiert je nach Testmethode) aufeinanderfolgenden Tests kein Ausfall auftreten /BNL 10/.

Testbasierte Methoden gehen nach Auftreten eines Fehlers von dessen sofortiger und vollständiger Behebung aus. Hier ist Vorsicht angebracht, da die Möglichkeit, dass ein Fehler nur teilweise beseitigt wird oder durch die Modifikation der Software weitere Fehler entstanden sind, in den meisten Fällen nicht berücksichtigt wird /BNL 10/.

D.6.2 Parametrische Black-Box-Modelle

Zu den parametrischen Black-Box-Modellen gehören unter anderem das „Fault activation"-Modell, das „Failure trend"-Modell und das „Environmental factors"-Modell, wobei die beiden ersten Modelle auch unter den „Reliability Growth"-Methoden einzuordnen sind (siehe Tab. D.2).

Tab. D.2 Übersicht über stochastische Black Box Modelle

Software Reliability Growth (SRG)		
Fault activation	**Failure trend**	**Environmental factors**
Generelle Annahmen: • das Versagen der Software wird durch einen unentdeckten Fehler verursacht • Fehler werden zufällig mit einer bestimmten Fehlerrate ausgelöst • Fehler sind voneinander unabhängig • Fehler werden sofort und vollständig behoben	**Generelle Annahmen:** • machen keine Annahmen über die Fehlerauslösung • Modellieren die Ausfallraten oder die Zeiten zwischen den Ausfällen • Fehler können auch verzögert und unvollständig korrigiert werden	**Generelle Annahmen:** • die Einsatzbedingungen beeinflussen die Systemzuverlässigkeit (z.B. Anzahl der Nutzer, Verwendung von Datenbanken etc.)

Im folgenden Abschnitt werden einige für das Projekt relevante „Reliability Growth"-Modelle detaillierter vorgestellt.

Reliability Growth Modelle

Reliability Growth Modelle prognostizieren ebenfalls auf der Grundlage von Testdaten die Zuverlässigkeit bzw. Ausfallwahrscheinlichkeit eines Systems oder einer Komponente unter der Annahme einer stetigen Korrektur der Softwarefehler. Diese Ausfallwahrscheinlichkeiten werden in der Zuverlässigkeitsbewertung eingesetzt, um zu zeigen, dass die Ausfallraten dem für die Software oder das rechnerbasierte oder programmierbare System geforderten Level entsprechen /BNL 10/.

Die in Tab. D.3 beschriebenen „Fault activation“ und „Failure trend“ gehören zur Gruppe der Reliability Growth Methoden. Als Input benötigen diese Zuverlässigkeitsmodelle entweder die zeitlichen Abstände zwischen aufeinanderfolgenden Ausfällen während eines Tests (Failure trend Methode) oder die Anzahl der Ausfälle in verschiedenen Zeitintervallen (Fault activation Methode). Um die Zunahme der Zuverlässigkeit bei stetiger Korrektur der Softwarefehler zu modellieren, werden in den Modellen Annahmen über die Behebung der Softwarefehler gemacht, die manchmal auch als „repair models“ bezeichnet werden.

Je nach Modell werden die folgenden Annahmen gemacht /BSI 98/:

1. Effekt der Fehlerkorrektur

    a) Vollständige Korrektur. Die Korrekturmaßnahmen waren zu 100% erfolgreich.
    b) Unvollständige Korrektur. Die Korrekturmaßnahmen waren teilweise (bis zu einer bestimmten Wahrscheinlichkeit) erfolgreich.
    c) Einführung neuer Fehler. Durch die Korrekturmaßnahmen wurden neue Fehler verursacht.

2. Zeitpunkt der Fehlerkorrektur

    a) Sofortige Korrektur nach der ersten Fehleraktivierung.
    b) Verzögerte Fehlerkorrektur nach mehrmaliger Fehleraktivierung.

„Failure trend“-Methoden können ohne zusätzlichen Aufwand auch die unvollständige Fehlerkorrektur, das Einbringen neuer Fehler und die verzögerte Fehlerkorrektur modellieren während „Fault activation“-Modelle durch solche Annahmen komplizierter werden.

Die meisten der in Tab. D.3 gelisteten Modelle gehen davon aus, dass ein Softwarefehler nach einem von ihm verursachen Ausfall sofort und vollständig behoben wird, können aber auf eine verzögerte und unvollständige Fehlerkorrektur erweitert werden. Nur in wenigen Modellen, wie beispielsweise im „Littlewood-Verrall“-Modell, wird die Annahme gemacht, dass die Korrektur von Fehlern auch zur Verschlechterung der Softwarezuverlässigkeit führen kann /BSI 98/.

Daher wächst die Zuverlässigkeit der Software bzw. des rechnerbasierten oder programmierbaren Systems mit der Zeit und die Ausfallraten werden kleiner. Anhand des (monoton) fallenden Verlaufs der Ausfallraten lässt sich abschätzen, um wie lange die Testphase verlängert werden muss, bevor der gewünschte Zuverlässigkeitslevel erreicht ist.

Die am Ende der Testphase erreichte Ausfallrate wird während des nachfolgenden Betriebs als konstant angesehen /BNL 10/.

Tab. D.3 Klassifikation stochastischer Zuverlässigkeitsmodelle, die einen „Black Box" Ansatz verwenden. Die aufgelisteten Methoden werden in den Standards /BSI 98/ und /IEEE 08/ als Methoden zur Zuverlässigkeitsbewertung genannt

<table>
<tr><th colspan="2">Software Reliability Growth (SRG)</th></tr>
<tr><th>Fault activation</th><th>Failure trend</th></tr>
<tr><td>Deterministic activation rate models (binominal models)
alle Fehler haben identische Fehlerauslöseraten oder werden durch eine mathematische Funktion beschrieben
in den meisten Modellen werden Fehler sofort und vollständig behoben
die Fehlerzahl ist eine feste Größe und in jedem Intervall binominal verteilt
Jelinski-Moranda model
Goel-Okumoto imperfect debugging model
(modelliert unvollständige Fehlerkorrektur)
Shooman model
Schick and Wolverton model
Generalized Poisson model
Binomial model
Brooks and Motley model</td><td>Failure rate trend models
Verlauf der Ausfalldaten des Systems wird durch Exponentialfunktionen beschrieben
Modelle benötigen Ausfalldaten des Systems
Duane model</td></tr>
<tr><td>Random activation rate models (binominal models)
Fehlerauslöseraten werden mit Zufallsvariablen beschrieben;
die Fehlerzahl ist eine feste Größe und ist in jedem Intervall binominal verteilt
Littlewood Stochastic Reliability Growth (LSRG) model
Weiss model
Ramamoorthy-Bastani Stochastic model</td><td>Random interfailure time models (Bayesian models)
Zeitspannen zwischen den Ausfällen werden durch Zufallsvariablen beschrieben
Littlewood-Verrall model (LV)
Keiller-Littlewood model
Ramamoorthy-Bastani Bayesian model</td></tr>
<tr><td colspan="2">Random number of faults models</td></tr>
</table>

Software Reliability Growth (SRG)	
Fault activation	**Failure trend**
(non-homogeneous Poisson process, NHPP) ▪ die Anzahl der Fehler wird als Zufallsvariable beschrieben ▪ die mathematische Beschreibung erfolgt durch einen exponentiellen oder nicht-exponentiellen inhomogenen Poisson Prozess (NHPP) **Exponentielle inhomogene Poisson Prozesse (Exponential NHPP)** • *Goel-Okumoto model* • *Musa model (modelliert verzögerte Fehlerkorrektur)* • *Rushforth, Staffanson and Crawford model* • *Schneidewind model* • *Langberg and Singapurwallah model* • *Generalized exponential model* • *Shooma's exponential model* **Nicht-Exponentielle inhomogene Poisson Prozesse (Non-exponential NHPP)** • *Duane model (unter Verwendung von NHPP)* • *Brooks and Motley's Binominal and Poisson model* • *Yamasa's Shape model* *Musa-Okumoto Logarithmic Poisson model*	

Binominale Modelle und Inhomogene Poisson Prozesse (NHPP)

Wie in Tab. D.3 gezeigt, existieren viele Software Reliability Growth Methoden, die sich unter anderem darin unterscheiden, welche empirischen Formeln für die Abnahme der Ausfallraten angesetzt werden. „Deterministic"- und „Random activation rate"-Modelle verwenden einen binominalen Ansatz für die Verteilung der Fehler, während „Random number of faults"-Modelle die Fehlerverteilung mit inhomogenen Poisson-Prozessen (Non-Homogeneous Poisson Process (NHPP)) beschreiben /BSI 98/.

Im Gegensatz zu homogenen Poisson-Prozessen ist die Rate des stochastischen Prozesses bei inhomogenen Poisson-Prozessen zeitabhängig. Der inhomogene Ansatz ist bei Software Reliability Growth Methoden notwendig, um die Abnahme der Ausfallrate über die Zeit abzubilden. Hierbei wird zusätzlich unterschieden in exponentielle und nicht-exponentielle NHPP, d. h. inhomogene Poisson-Prozesse, bei denen die Ausfallrate als exponentiell abnehmend bzw. nicht-exponentiell abnehmend angenommen wird. Die in den empirischen Formeln verwendeten Parameter werden bei den verschiedenen Methoden aus Testdaten geschätzt. Die Methoden zur Abschätzung der Parameter sind nicht direkter Bestandteil der Software Reliability Growth Methoden, vielmehr werden oftmals Standardmethoden wie beispielsweise Maximum-Likelihood-Schätzungen oder Least-Squares-Schätzungen verwendet /BNL 10/.

Die in Tab. D.3 aufgelisteten Modelle werden nach den Standards /IEEE 08/ und /BSI 98/ für die Zuverlässigkeitsbewertung im nicht-nuklearen Bereich empfohlen.

Random inter-failure time models

„Random inter-failure time models“ modellieren die Zeiten zwischen den Ausfällen mit unabhängigen Zufallsvariablen /BSI 98/. Während Modelle unter Verwendung inhomogener Poisson-Prozesse annehmen, dass die Ausfallwahrscheinlichkeit mit jedem korrigierten Fehler (exponentiell oder nicht-exponentiell) abnimmt, ermöglicht das Bayessche Modell nach Littlewood und Verrall /LIT 74/, /IEEE 08/ die Abnahme der Ausfallwahrscheinlichkeit stochastisch zu beschreiben /BNL 10/. Zwar geht das Modell von einer exponentiellen Abnahme der Ausfallwahrscheinlichkeit aus (exponentielles NHPP Model) durch das Einfügen der Maximum-Likelihood-Funktion wird jedoch auch die Möglichkeit modelliert, dass eine Fehlerbehebung zu neuen Fehlern führt, und somit die Ausfallwahrscheinlichkeit zunächst ansteigt /BNL 10/.

Markov-Prozesse

Das Ziel der Modellierung einer Software oder eines rechnerbasierten oder programmierbaren Systems als Markov-Modell besteht in seiner einfachsten Form in der Angabe von Wahrscheinlichkeiten für das Eintreten zukünftiger Ereignisse, d. h. dafür, dass das System einen speziellen Zustand annimmt. Hierzu wird die Software bzw. das System

durch seine Zustände und die zwischen ihnen möglichen Übergänge im Zustandsraum dargestellt /NUR 05/.[9]

Bei der Modellierung ist zu beachten, dass sich die Software bzw. das System zu jeder Zeit in genau einem der Zustände befindet. Die Besetzungswahrscheinlichkeit eines Zustandes gibt an, wie wahrscheinlich es ist, dass sich die Software bzw. das System zu einem beliebigen Zeitpunkt in diesem Zustand befindet. Sie ergibt sich für jeden Zustand aus den Übergangswahrscheinlichkeiten zwischen den einzelnen Zuständen /VDI 07/.

Der Einsatz dieser Methode scheitert in vielen Fällen daran, dass nicht alle möglichen Zustände der Software bzw. des Systems bekannt sind und bei der Modellierung berücksichtigt werden können. Auch wie genau sich die einzelnen Übergangswahrscheinlichkeiten zwischen den Zuständen angeben lassen ist entscheidend für die Anwendbarkeit der Methode /NUR 05/. Ein Nachteil, vor allem bei komplexeren Systemen, ist der mit der Zahl der möglichen Zustände schnell zunehmende Rechenaufwand /VDI 07/

D.6.3 Bayessche Netze

Bei Bayesschen Netzen handelt es sich um probabilistische, grafische Modelle, die einen Satz von Zufallsvariablen (dargestellt als Knoten) und deren kausale Abhängigkeiten (dargestellt als gerichtete Kanten bzw. Pfeile) in einem gerichteten azyklischen Graphen beschreiben /BNL 10/. Die gerichteten Kanten bzw. Pfeile stellen jeweils den kausalen Einfluss von dem Knoten, an dem die Kante bzw. der Pfeil beginnt („Elternknoten“, Ursache), auf den Knoten, auf den die Kante bzw. der Pfeil zeigt („Kindknoten“, Wirkung), dar. In einem zyklischen Graphen gibt es Knoten, die sich in Pfeilrichtung laufend mehr als einmal erreichen lassen. Bei einem azyklischen Graphen ist dies nicht der Fall. Die Zufallsvariablen können hier beobachtbare Größen, Unbekannte oder Hypothesen repräsentieren. Jedem „Kindknoten“ ist eine bedingte Wahrscheinlichkeitsverteilung der durch ihn dargestellten Zufallsvariablen zugeordnet, die von den durch seine „Elternknoten“ beschriebenen Zufallsvariablen abhängt.

[9] Weitere Modelle können “Hellmich, *Statistical inference of a software reliability model by linear filtering*, J. Stat. Manage. Syst. 19, 136-181 (2016)” entnommen werden.

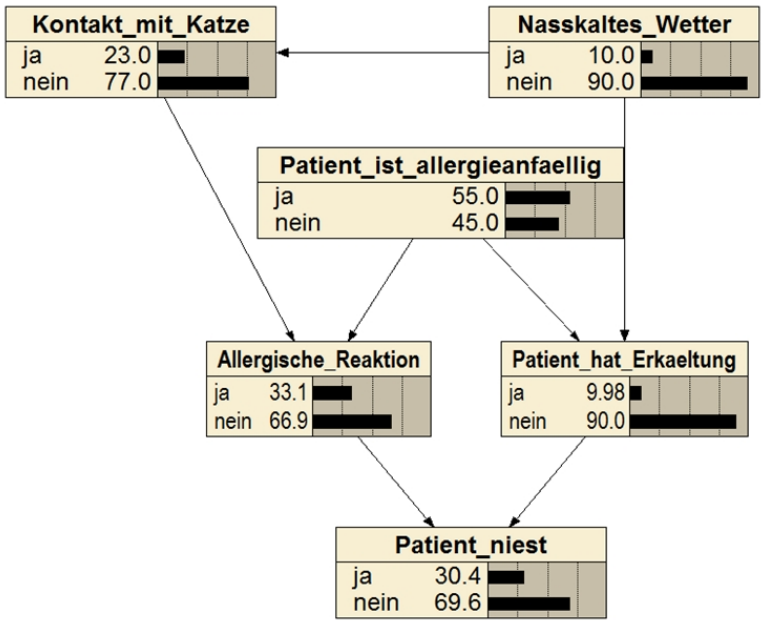

Abb. D.4 Beispiel eines Bayessches Netz /GRS 15a/

In einem Bayesschen Netz wird die gemeinsame Wahrscheinlichkeitsverteilung aller Variablen V_1, .., V_n definiert als

$$P(V_1, V_2, \dots, V_n) = \prod_{i=1}^{n} P(V_i \,|Eltern(V_i))$$

Zur Erstellung eines Bayesschen Netzes werden einerseits statistische Daten und andererseits Schätzungen von Experten verwendet. Bei der Zuverlässigkeitsbewertung eines sicherheitsrelevanten Systems können beispielsweise konstruktive Maßnahmen, Entwicklungsprozesse, Testergebnisse und Betriebserfahrung in die Erstellung des Bayesschen Netzes einfließen. Die Abbildung der kausalen Abhängigkeiten und Zusammenhänge zwischen den für die Zuverlässigkeitsbewertung relevanten Aspekten in einem Bayesschen Netz ist sowohl der schwierigste als auch der wichtigste Schritt in der Anwendung solcher Bayesschen Netze für rechnerbasierte oder programmierbare Systeme. Damit ist die Qualität der Zuverlässigkeitsbewertung mittels eines Bayesschen Netzes stark abhängig von dem in diesen Schritt eingehenden Fachwissen. Eine weitere Herausforderung stellt die Notwendigkeit dar, qualitative Belege in quantitative Wahrscheinlichkeitsaussagen umzusetzen. Bei nicht ausreichend vorhandenen Daten kann dies zu großen Unsicherheiten in den quantitativen Abschätzungen führen /BNL 10/.

Liegen auch messbare Daten **y** = (y_1,...,y_m) vor (auch *hard evidence* genannt), die dazu geeignet sind zu einer besseren Wahrscheinlichkeitsaussage für die Modellparameter zu kommen, so kann das Modell der Bayesschen Interferenz verwendet werden. Bei diesem Modell geht es darum aus den Daten zu lernen, indem die subjektiven Wahrscheinlichkeiten P für bestimmte Modellparameter mit Hilfe der Daten aufaddiert werden und so die Wahrscheinlichkeitsaussagen für die Modellparameter verbessert wird.

Bayessche Inferenz

Bei der Bayesschen Inferenz wird die Vorhersageunsicherheit der unbekannten Parameter des Modells durch Wahrscheinlichkeitsfunktionen beschrieben. Dabei haben die Funktionen neben den Zufallsparametern $\theta = (\theta_1,\ldots,\theta_n)$ auch messbare Variablen (*hard evidence*) $\boldsymbol{y} = (y_1,\ldots,y_m)$ wie statistische Beobachtungswerte. Zuerst wird eine A priori-Wahrscheinlichkeitsverteilung p(θ) definiert, die die Wahrscheinlichkeitsverteilung der Zufallsparameter aufgrund von Vorabinformation und ohne Berücksichtigung von Beobachtungen in Form der messbaren Variablen beschreibt. Die messbaren Daten ***y*** werden in einer gemeinsamen Wahrscheinlichkeitsverteilung, der sogenannten Likelihood-Funktion $p(\boldsymbol{y}|\theta)$ modelliert. Die Likelihood-Funktion $p(\boldsymbol{y}|\theta)$ beschreibt die Wahrscheinlichkeit die Daten ***y*** zu beobachten, falls die Parameter die Werte θ annehmen. Die A priori-Verteilung wird schließlich zu einer A posteriori-Verteilung $p(\theta|\boldsymbol{y})$ aktualisiert. Diese Funktion $p(\theta|\boldsymbol{y})$ beschreibt die Wahrscheinlichkeitsverteilung der Parameter θ unter der gemachten Annahme über θ (in Form der A priori-Verteilung) und bei gegebenen gemessenen Variablen ***y***. Die gemessenen Daten liefern so weitere Informationen über die zufälligen Variablen θ::

$$p(\theta|\boldsymbol{y}) = \textbf{Likelihood} \times \textbf{Priori} \times \textbf{Konstante}$$

$$p(\theta|\boldsymbol{y}) = \frac{p(\boldsymbol{y}|\theta)p(\theta)}{\int p(\boldsymbol{y}|\theta)p(\theta)d\theta}$$

Die Bayessche Inferenz hat den Vorteil, dass sie mit Wahrscheinlichkeitsverteilungen arbeitet und daher auch statistische Unsicherheiten der Parameter erfasst und zudem eine transparente Modellierung der Parameter erlaubt /STUK 01/.

Tab. D.4 Zusammenfassung der Quantitativen Methoden zur Zuverlässigkeitsbewertung von Software nach U.S. Nuclear Regulatory Commission /NUR 11/

	Dokumentation und Beschreibung der Methode	Annahmen sind vernünftig	Kontext des Betriebs	Lebens-Zyklus enthalten	Betriebserfahrungen enthalten	Unsicherheiten werden berücksichtigt	Verifizierung und Validierung	Großer Anwendungsbereich	Software CCF wird berücksichtigt
Reliability Growth Methoden	ja	?	nein	nein	ja	ja	ja	nein	nein
Bayessche Netze	ja	?	nein	ja	ja	ja	?	?	nein
Testbasierte Methoden (Black-Box)	ja	?	?	nein	ja	ja	?	?	nein
Testbasierte Methoden (White-Box)	ja	?	?	nein	ja	ja	nein	?	nein

ja Das Konzept der Quantitativen Methode erfüllt die gewünschte Charakteristik

? Das Konzept der Quantitativen Methode könnte die Charakteristik erfüllen und/oder seine Anwendung könnte zielführend sein

nein Das Konzept der Quantitativen Methode kann die gewünschte Charakteristik nicht erfüllt

E Hauptkomponentenanalyse

Die Hauptkomponentenanalyse kombiniert die oben beschriebenen Metriken auch mit der dynamischen Komplexität und ist ein Beispiel dafür, wie Metriken verwendete werden können, um die Qualität einer Software zu bestimmen. Die Schritte sind im Einzelnen:

1. Reduktion der ausgewählten in Wechselbeziehung zueinander stehenden Metriken zu einer kleineren Menge an orthogonalen, d.h. unabhängigen Metriken durch Hauptkomponentenanalyse. Weiter Reduktion der Metriken zu einer einzelnen Metrik.

2. Quantifizierung des Ausführungsprofils, das die Wahrscheinlichkeit p_i beschreibt, mit der die Ausführung einer Funktion in Modul m_i stattfindet.

3. Kombination der statischen Komplexitätsmetrik mit der dynamischen Komplexität des Ausführungsprofils

Reduktion der ausgewählten Metriken durch Hauptkomponentenanalyse (PCA)

Für die Bestimmung der Softwarezuverlässigkeit ist es vorteilhaft, wenn die Funktionalität des Systems bei der Entwicklung der Software in einzelne Module zerlegt wurde. Diese Module kommunizieren miteinander, können aber bei der Zuverlässigkeitsbewertung zunächst im Einzelnen untersucht werden. Je nach Einsatz und Komplexität der Module während des laufenden Programms, lässt sich die Zuverlässigkeit des Gesamtsystems dann bestimmen.

Für die Analyse werden die multivarianten Daten zunächst als Matrix dargestellt (*m* Software Metriken x *n* Programm Module) und dann die Hauptkomponenten durch die Berechnung der *m* Eigenvektoren und Eigenwerte λ_i der zugehörigen empirischen Kovarianzmatrix bestimmt. Um die Dimension so zu reduzieren, dass die wesentlichen Informationen erhalten bleiben, wird für jede Hauptkomponente bestimmt wie viel Prozent der gesamten Varianz der Daten durch sie abgedeckt wird. Es werden dann alle $p < m$ Hauptkomponenten ausgesucht, so dass ein großer Teil der Varianz abgedeckt wird. Typischerweise werden alle Hauptkomponenten mit Eigenwerten größer als 1 ausgewählt /LYU 96/.

Im folgenden Beispiel werden die Halstead (N, V, E) und die McCabe V(G) Metriken verwendet, um die Software Module zu bewerten. Die Analyse erfolgt dann in folgenden Schritten:

1. Aufstellen der (*m* x *n*) Matrix *X* mit den Software Metriken *m* und den Programm Modulen *n*

2. Umwandeln der Matrix *X* in eine normierte Matrix *Y*. Dazu werden die Elemente der Matrix so normiert, dass der Mittelwert der Daten im Nullpunkt liegt und die Standardabweichung den Wert 1 besitzt. Für jeden Vektor wird die Varianz, d.h. die Abweichung vom arithmetischen Mittel (hier der Nullpunkt) bestimmt. Anschaulich wird der Schwerpunkt der m-dimensionalen Daten in den Koordinatenursprung geschoben. Diese Normierung ist notwendig, um die sehr unterschiedlichen Werte der Metriken vergleichbar zu machen.

   Mittelwert: $\mu_j = \frac{1}{n}\sum_{i=1}^{n} x_{ij}$ Standardabweichung: $\sigma_j = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_{ij} - \mu_j)^2}$

   $\hat{y}_{ij} = \frac{x_{ij} - \mu_j}{\sigma_j}$

3. Schließlich wird die Kovarianzmatrix (auch Korrelationsmatrix genannt) der Matrix *Y* bestimmt, die alle paarweisen Varianzen der Elemente enthält, so dass gilt:

$$R = [r_{ij}]_{mxm} = \frac{Y^T Y}{(n-1)}$$

   Mit $r_{ij} = \sum_{k=1}^{n}(y_{ik} \cdot y_{kj})/(n-1)$ mit *i = 1, 2, …, m, j = 1, 2,…, m.*

4. Dann werden die Eigenwerte $\lambda_1, \lambda_2, \ldots, \lambda_i$ der Kovarianzmatrix R bestimmt. Die Eigenwerte sind ein Maß für die Varianz der Metriken (Streckungsfaktor des Eigenvektors). Wie groß die Varianz einer Metrik im Verhältnis zur Gesamtvarianz ist lässt sich bestimmen mit

$$r_l = \frac{\lambda_l}{\sum_{i=1}^{m} \lambda_i}$$

   Für die Eigenwerte $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_p$ werden mit sinkender Größe die Eigenvektoren $e_1, e_2, \ldots, e_p$ bestimmt, wobei Eigenvektoren zu Eigenwerten < 1 nicht berücksichtigt werden, da ihr Informationsgehalt vernachlässigbar gering ist, so dass man $p < m$ Eigenvektoren erhält.

5. Mit den berechneten Eigenvektoren kann die normierte Transformationsmatrix T* bestimmt werden. Für ihre Elemente gilt $t_j = e_j/\sqrt{\lambda_j}$ mit *j =1…p.* Die Spalten der Transformationsmatrix sind folglich die normierten Eigenvektoren zu den Eigenwerten. Die Elemente der Transformationsmatrix T* lassen sich so verstehen, dass das Element t_{ij} die Gewichtung der i-ten Komplexitätsmetrik, *i = 1, 2, …, m* für die j-te „neu definierte Metrik-Komponente“ darstellt. In Tabelle 3 ist eine Transformationsmatrix T* aus /LYU 96/ dargestellt.

6. Der letzte Schritt ist die Berechnung der neuen orthogonalen Vektoren für die neu definierten Metrik Komponenten aus der Transformationsmatrix T* mit $D_k = y_k T^*$, wobei y_k der Vektor der normierten Metrik für die Programm Module *k = 1, 2, …,n* ist. (Anschaulich wird die Messung ins neue Koordinatensystem transformiert.) Die *p* Komponenten des Vektors beschreiben die Streuung der neuen orthogonalen Komplexitätskomponenten. Aus den n Programm Modulen ergeben sich pro Komponente j = 1, 2, …, p genau n Werte, die zwischen 0 und 1 liegen /LYU 96/.

Tab. E.1 zeigt die Komponentenmatrix, die auch als Ladungsmatrix bezeichnet wird. Die Streuung in den Daten der Metriken lässt sich durch zwei Komponenten beschreiben. Es ist zu erkennen, dass die drei Halstead Metriken (N, V, E) stark mit der ersten Hauptkomponente korrelieren und die McCabe Metrik (V(G)) mit der zweiten Hauptkomponente. Die erste Komponente ist folglich ein Maß für die Komplexität aufgrund der Größe des Moduls, während die zweite Komponente sich auf die Komplexität des Kontrollflusses bezieht /LYU 96/.

Tab. E.1 Beispiel für eine Transformationsmatrix /LYU 96/

Metrik	Komponente 1	Komponente 2
N	0,364	-0,046
V	0,361	-0,061
Ê	0,334	-0,015
V(G)	-0,119	1,020

Durch die Hauptkomponentenanalyse lässt sich folglich bestimmen, welche Merkmale die Programm Module am besten beschreiben, d.h. den höchsten Informationsgehalt besitzen und somit bei der Zuverlässigkeitsbewertung am aussagekräftigsten sind.

Weitere Reduktion der Komplexität Metriken zu einem einzigen Wert

Um die Struktur der Software Komplexität weiter zu vereinfachen, wird für jedes Programm Modul eine lineare Funktion g(x) = ax + b definiert, die die Software Fehler mit der Komplexität des Programms x verknüpft. Dafür wird die relative Komplexitätsmetrik ρ bestimmt, die die gewichtete Summe einer Menge unkorrelierter Komponenten Metriken ist. Die relative Komplexität ist eng verknüpft mit der Anzahl der Softwarefehler und wird bestimmt zu:

$\rho_k = d_k (\Lambda^*)^T$,

wobei d_k der Vektor der neuen Metrik Komponenten für das Programm Modul *k* ist und $(\Lambda^*)^T$ der transponierte Vektor ($\lambda_1, \lambda_2, \ldots, \lambda_p$) mit den Eigenwerten der Metrik Komponenten ist. Die relative Komplexität wird schließlich noch skaliert, so dass für jedes Modul eine skalierte relative Komplexität bestimmt werden kann, die um einen Mittelwert von 50 und eine Standardabweichung von 10 verteilt ist:

$$\rho'_k = \frac{10\,\rho_k}{\sqrt{\sum_{j-1}^{p} \lambda_j^2}} + 50$$

Um auch die Änderungen der relativen Komplexität während des Software Entwicklungsprozesses zu berücksichtigen, die mit jeder neu entwickelten Version der i Module ansteigt, wird die relative Komplexität für jede Version eines Moduls ρ_i^1 ermittelt /LYU 96/. Die relative Komplexität des Gesamtsystems ist dann die Summe aller Modulkomplexitäten.

Quantifizierung der dynamischen Komplexität des Ausführungsprofils

Nachdem die „statischen" Eigenschaften des Programms wie Größe und Komplexität der Kontrollstruktur bestimmt wurden, muss für ein Programm mit einer großen Funktionsvielfalt, auch die „dynamische" Komplexität berücksichtigt werden. Sie wird bestimmt durch die Teilmenge an Modulen, die durchlaufen werden, wenn das System eine bestimmte Funktion ausführt. Dabei ist für die „dynamische" Komplexität die Einsatzumgebung des Systems entscheidend, da sie bestimmt welche Programm Module wirklich benötigt werden. In einer Einsatzumgebung in der die komplexen Programme des Systems mit hoher Wahrscheinlichkeit ausgeführt werden, ist die dynamische Komplexität

hoch. Es kann vorkommen, dass mögliche Anwendungsszenarien außergewöhnlich große dynamische Komplexität besitzen. Die Verteilung der Verarbeitungszeit auf die unterschiedlichen Module wird durch das Ausführungsprofil (Execution Profile) der Software bestimmt.

Da es für jede Funktion f_i ein Ausführungsprofil mit der Wahrscheinlichkeit $p^i_1, p^i_2, \ldots, p^i_n$ gibt, ermittelt sich die funktionale Komplexität Φ des Gesamtsystems für alle Funktionen und alle Module zu:

$$\phi_i = \sum_{j=1}^{n} p_j^i \, \rho_j$$

Da bei der Ausführung eines Programms nur bestimmte Funktionalitäten durchlaufen werden, wird immer nur eine Teilmenge des Gesamtsystems ausgeführt, so dass die betriebliche Komplexität ω geringer ist, als die funktionale Komplexität Φ.

F Einsatz von Software-Metriken und stochastischen Modellen zur Zuverlässigkeitsbewertung (ISTec)

Im Rahmen eines Forschungsvorhabens wurde vom Institut für Sicherheitstechnologie (ISTec) ein Konzept zur Vorhersage der Zuverlässigkeit eines softwarebasierten digitalen Leittechniksystems entwickelt. Die Methode, die Messungen der Komplexität mit statistischen Methoden zur Zuverlässigkeitsbewertung kombiniert, wurde speziell für softwarebasierte Leittechnik entwickelt, die anhand graphischer Spezifikationen mittels Code-Generatoren erzeugt wurde /IST 10/.

Für die bottom-up-Analyse wurden zunächst zwei unterschiedliche probabilistische Komplexitätsbewertungen der Funktionsbausteine entwickelt: Die Black-Box-Methode kann angewandt werden, wenn kein Zugriff auf den Programmtext der Funktionsbausteine möglich ist, während die White-Box Methode detaillierte Kenntnisse des Programms benötigt. Zusätzlich erfolgt eine probabilistische Komplexitätsbewertung der Funktionspläne, sowie weitere Komplexitätsmaße für Speicher und Parametrierbarkeit /IST 10/.

Für die Entwicklung des Konzepts wurde beispielhaft das digitale Leittechniksystem TELEPERM XS der Firma Siemens verwendet. Ziel des Projekts war es jedoch prototypisch die Automatisierbarkeit der Methode nachzuweisen und eine generelle Übertragbarkeit auf modular aufgebaute I&C-Systeme wie COMMON Q, TRICON oder KNICS zu ermöglichen. Der Vorteil einer solchen automatisierbaren Komplexitätsmessung wäre zum einen bestehende Komplexitätspeaks in der fertigen Software zu erkennen und zum anderen bereits in der Entwurfsphase erhöhte Komplexität zu identifizieren, um diese durch geeignete Strukturierung zu reduzieren /IST 10/.

F.1 TELEPERM XS

Mit TELEPERM XS können folgende leittechnische Funktionen realisiert werden:

- das Reaktorschutzsystem,
- das Engineered Safety Feature Actuation System (ESFAS)
- das Begrenzungssystem oder
- die Reaktorregelungen

Die Software Architektur ist in die Systemsoftware MICROS und die Anwendersoftware SPACE (Spezifikation and Coding Environment) eingeteilt und kann anlagenunabhängig eingesetzt werden. Das Echtzeitbetriebssystem MICROS übernimmt Initialisierung und Interface-Funktionen und umfasst die Protokollsoftware für die serielle Datenübertragung. Die Anwendersoftware kann an die leittechnischen Aufgaben angepasst werden und besteht aus /IST 10/, /TEL 97/:

- **Funktionsbaustein-Modulen**
  Elementare Grundkomponenten, die zur Realisierung leittechnischer Elementarfunktionen benötigt werden wie z.B. die logischen Funktionen AND, OR, 2von3. Neben den Funktionsbausteinen zur Realisierung elementarer Logik- und Arithmetik-Funktionen beinhaltet das Modul auch Funktionsbausteine zur Realisierung leittechnischer Grundfunktionen, wie Grenzwertgeber oder Integrierer sowie Spezialbausteine wie Formelbausteine, Sortierer, Rampengenerator etc. Insgesamt gibt es ein Grundrepertoire von ca. 100 Elementarfunktionen /IST 10/, /TEL 97/.

- **Funktionsplan- und Funktionsplangruppen-Modulen**
  Funktionspläne sind die graphische Repräsentation der leittechnischen Funktionen. Für jede leittechnische Aufgabe kann mittels eines Funktionsplans ein Funktionsplanmodul generiert werden, welches die für die Verschaltung benötigten Funktionsbausteine in algorithmisch korrekter Reihenfolge enthält. Alle Funktionsplan-Module, die auf einem Funktionsrechner mit der gleichen Zykluszeit ablaufen, werden zu einem Funktionsplangruppen-Modul mit richtiger Aufrufreihenfolge und dem Signalaustausch zwischen den Funktionsplänen implementiert /TEL 97/.

F.2 Quantitative Bestimmung der Komplexität der Funktionsbausteine

Für die Komplexitätsmessung der Funktionsbausteine wird eine Komplexitätsmatrix erstellt. Dabei werden zwei unterschiedliche Methoden beschrieben, der Black-Box View, bei dem Software-Dokumente und User Manuals als Grundlage für die Komplexitätsmessung dienen und der White-Box View bei dem der Quellcode der Funktionsbausteine als Grundlage dient /IST 10/.

Black-Box Methode

Im Falle der Black-Box Methode werden aus den Dokumenten die folgenden Informationen gewonnen /IST 10/:

- Zahl und Art der Parameter
    - unveränderliche Parameter, die nur durch eine neue Codegenerierung geändert werden können
    - veränderbare Parameter
    - abgeleitete Parameter
- Zustandsspeicher (aus Angaben in der Dokumentation)
- Zeitabhängigkeiten
    - allgemeine zeitabhängige Funktionsbausteine
    - Delay-Funktionsbausteine
- Statusverarbeitung der Signale
    - Aktive, passive Statusverarbeitung
    - Fehlerausbreitung und Ausbreitungssperre für fehlerhafte Signale (Maskierungen)

Tab. F.1 zeigt die Auswahl der Merkmale der Black-Box Methode. Diese wurden so gewählt, dass die Merkmale direkt mit der Komplexität der Funktionsbausteine zusammenhängen oder auch potentiell anfällig sind für Fehlereinführungen. Dabei wird davon ausgegangen, dass je komplexer ein Funktionsbaustein ist, umso wahrscheinlicher wird eine Fehlinterpretation der Spezifikation beim Design des Bausteins /IST 10/.

Tab. F.1 Komplexitätsmatrix der Funktionsbausteine nach der Black Box Methode. Nach /IST 10/

Merkmal	Beschreibung
Fan out	Korreliert mit der Zahl der Softwarefehler
Signale (Eingänge, Ausgänge, binär, analog, Meldung)	Zahl der Ein-/Ausgangssignale und Meldesignale beschreibt Komplexität der Kopplung der Funktionsbausteine untereinander
Parameter (nicht änderbar, änderbar, Bedingungen, abgeleitet)	Können durch Spezifikations- oder Übertragungsfehler zu einem Fehlverhalten der leittechnischen Funktion führen. Der Einfluss änderbarer Parameter auf die Entstehung von Softwarefehlern, hängt von der Häufigkeit der Parameteränderungen ab.
Interne Variable	Die Zahl der internen Variablen korreliert mit der Komplexität des Funktionsbausteins
Zustandsspeicher	Die Zahl der Zustandsspeicher korreliert direkt mit der Komplexität eines Funktionsbausteins
Speicherbedarf (Code, Stack)	Die Komplexität und der Speicherbedarf korrelieren auf komplexe Weise miteinander. Daher wird der absolute Speicherbedarf bestimmt.
Laufzeit (in µs) (Initialisieren, Parametrieren, Berechnen)	Die Laufzeit spiegelt den Umfang der Software wieder und ist daher ein Maß für die Komplexität. Es wird unterschieden in Laufzeiten für Initialisieren, Parametrieren und Berechnen.
Fehlercodes	Die Anzahl der unterschiedlichen Fehlerfälle ist ein möglicher Indikator für die Komplexität
Funktion (Text, Bild, Tabelle)	Es wird angenommen, dass die Beschreibungsweise eines Funktionsbausteins Rückschlüsse auf die Komplexität erlaubt

White-Box Methode

Bei der White-Box Methode werden durch das Software-Analyse-Werkezeuge „Testbed von LDR“ unter anderem die folgenden Metriken bestimmt /IST 10/:

- Komplexitätsmetriken
    - Knotenmetrik, zyklomatische Komplexität, wesentliche Knoten, wesentliche zyklomatische Komplexität

- Schleifen/Intervallanalyse
    - Anzahl der Schleifen, Schleifenschachtelungstiefe, Reduzierbarkeit (Intervalle)

- Linear Code Sequence and Jump LCSAJ
    - Anzahl der LCSAJs, Maximale LCSAJ Dichte
- Datenflussinformation
    - Fan Out

Die Metriken liefern eine „Größe“ für die Softwareattribute, die zunächst nicht bewertbar ist, d.h. nicht als „gut“ oder „schlecht“ bewertet werden kann. Erst im Gesamtkontext ergibt bei der Interpretation der statistischen Analyse der Metriken eine Bewertung.

Wie bereits erwähnt orientiert sich die probabilistische Komplexitätsbewertung am Aufbau der Anwendersoftware in Funktionsbausteine und Funktionspläne. Für die Zuverlässigkeitsbewertung wird die probabilistische Methode der Bayesschen Netze verwendet /IST 10/.

In einem aufwendigen Verfahren werden die Modellparameter als Komplexitätsfaktoren eines Komplexitätsvektors bestimmt. Dabei gliedert sich der Vektor in Komplexitätseigenschaften, die spezifisch für digitale Leittechniksysteme sind, wie einfache Umfangkriterien (K1-K5), die Komplexität der Verschaltung (K6), die Variabilitäts-Komplexität (K7-K8) und die Komplexität der Funktionsbausteine (K9):

Komplexität der Funktionspläne:
K1: Anzahl der Funktionsbausteine
K2: Anzahl der Eingangs-Signale
K3: Anzahl der Ausgabesignale
K4: Anzahl der vorgelagerten Funktionspläne
K5: Anzahl der nachfolgenden Funktionspläne
K6: Komplexität der Verschaltung
K7: Anzahl änderbarer Parameter
K8: Anzahl der internen Speicher

Komplexität Funktionsbausteine:
K9: Komplexität der Funktionsbausteine des Funktionsplans

Wie aus dem Komplexitätsvektor erkennbar ist, wird die Komplexität der Funktionspläne durch die Parameter K1-K8 und die Komplexität der Funktionsbausteine durch den Parameter K9 unabhängig voneinander bewertet /IST 10/.

Dieses Konzept soll der flexiblen Struktur von TELEPERM XS gerecht werden und eine breite Anwendbarkeit der Zuverlässigkeitsbewertung gewährleisten /IST 10/.

F.3 Bestimmung des Komplexitätsvektors

Die Erstellung des Komplexitätsvektors erfolgt durch eine automatisierte Komplexitätsmessung. Dies geschieht durch zwei Software-Werkzeuge /IST 10/:

- den systemspezifischen EXTRACTOR, der aus der digitalen Darstellung der Funktionspläne in der SPACE-Datenbank die komplexitätsrelevanten Charakteristika extrahiert und elementare Funktionsplandaten in der universellen Schnittstelle ablegt. Dabei befindet sich der Großteil der für die Komplexitätsmessung relevanten Daten in den Spezifikationstabellen von SPACE, aber auch andere Tabellen werden verwendet.
- den systemunabhängigen EVALUATOR, der auf der Grundlage der universellen Schnittstelle den Komplexitätsvektor ermittelt.

Der Komplexitätsvektor setzt sich aus unterschiedlichen Komplexitätsmaßen zusammen, die zusammengenommen ein Maß für die Systemkomplexität ergeben /IST 10/:

Elementare Umfangsmaße:
K1: Anzahl der Funktionsbausteine
K2: Anzahl der Eingangs-Signale
K3: Anzahl der Ausgabesignale
K4: Anzahl der vorgelagerten Funktionspläne
K5: Anzahl der nachfolgenden Funktionspläne
K6: Komplexität der Verschaltung

Variabilitäts-Komplexität:
K7: Anzahl änderbarer Parameter

K8: Anzahl der internen Speicher

K9: Komplexität der Funktionsbausteine des Funktionsplans

Elementare Umfangsmaße

Als elementare Charakteristika zur Beschreibung der Komplexität werden unter anderem die folgenden Werte ermittelt /IST 10/:

- Anzahl der verwendeten Funktionsbausteine (Umfang) (K1)
- Eingabe-/ Ausgabeverhalten (Umgebung) (K2 und K3)
- Anzahl der vorgelagerten/nachfolgenden Funktionspläne (K4 und K5)

Die Ermittlung dieser Komponenten ergibt sich durch Zählung auf Grundlage der graphischen Darstellung. Diese Werte können aus dem EXTRACTOR direkt übernommen werden /IST 10/.

Verschaltungs-Komplexität

Die Komplexitätsmessung der Verschaltung (K6) erfolgt mit dem Ziel die Komplexität der Funktionspläne möglichst unabhängig von den elementaren Funktionsbausteinen zu bewerten und eine allgemeine Anwendbarkeit auf graphisch-basierte, digitale Leittechnik-Systeme zu ermöglichen. Daher werden graphische Funktionspläne verwendet, um die Komplexität der Verschaltung zu bestimmen anstatt den daraus generierte Quellcode zu analysieren. Die aus dem Softwarebereich bekannten Komplexitätsmaße werden dazu auf die graphische Darstellung des Funktionsplans übertragen /IST 10/.

Diese Herangehensweise lässt sich auch damit begründen, dass sich die logische Komplexität eines Funktionsplans nicht im Kontrollfluss sondern in den Datenstrukturen widerspiegelt /IST 10/.

Unter der Verflechtung V(FP) eines Funktionsplans (FP) (K6) ist zu verstehen, dass einzelne Funktionsbausteine oder auch Gruppen von Funktionsbausteinen zur Berechnung mehrere Ausgangssignale verwendet werden können. Zudem wird die Verflechtung dadurch bestimmt, dass mehrere gemeinsamer Eingangssignale zur Berechnung des Ausgangssignals erforderlich sind. Die Verflechtung V(FP) eines Funktionsplans wird wie folgt bestimmt /IST 10/:

$$V(FP) = \sum_{S_i} \frac{|VB(S_i)| + |IN(S_i)|}{|BFP| + |SIN|}$$

Mit:

S_i	Ausgabesignale des Funktionsplans FP
$VB(S_i)$	Menge der Funktionsbausteine, die zur Berechnung des Ausgabe-Signals S_i beitragen. Sie werden als Vorbereich bezeichnet.
$IN(S_i)$	Menge der Eingang-Signale, die zur Berechnung des Ausgabe-Signals S beitragen
BFP	Gesamtmenge der Funktionsbausteine, aus denen der Funktionsplan besteht.
SIN	Gesamtmenge der Eingang- Signale des Funktionsplans
| |	Mächtigkeit (Anzahl der Elemente) einer Menge

Die Vorteile der so definierten Verflechtung sind ihre Einfachheit und ihr modularer Aufbau in Bezug auf Prozessoren, Funktionspläne und Einzelsignale, sowie die Unabhängigkeit von der Komplexität der Funktionsbausteine. Über die Menge der Funktionsbausteine, die zur Berechnung des Ausgabesignals verwendet werden, wird auch die Komplexität der Funktionsbausteine mitberücksichtigt. Das Verflechtungsmaß kann auf andere graphikbasierte Systeme übertragen werden und auf das Intervall [0, 1] normiert werden, wenn N die Anzahl der Ausgangssignale ist /IST 10/:

$$V_{Norm}(FP) = \frac{V(FP) - 1}{N - 1}$$

Die Bestimmung der Verschaltungs-Komplexität ist aufgrund des großen Umfangs ebenfalls automatisiert. Hierzu werden alle vom EXTRACTOR erzeugten Tabellen verwendet. Dadurch ergibt sich die Komponente 6 des Komplexitätsvektors /IST 10/.

Variabilitäts-Komplexität

Änderbare Parameter

TELEPERM XS besitzt auch änderbare Parameter (K7), die das digitale Leittechnik System sehr flexibel machen und es ermöglichen seine Funktionalität stark zu modifizieren. Durch die Möglichkeit der Parameterveränderung wird auch die Komplexität des Systems erhöht, da neue Varianten der implementierten Grundfunktionalität entstehen und eine zusätzliche Mensch-Maschine-Schnittstelle eingebracht wird. Zur Bewertung der Komplexität wird daher auch die Anzahl der änderbaren Parameter bestimmt /IST 10/:

$$Parameter\ (FP) = \sum_{FB_i} Parameter\ (FB_i)$$

Interne Speicher

Auch interne Zustände eines Systems, die für digitale Leittechniksysteme als interne Speicher der Funktionsbausteine realisiert werden, können das Verhalten eines Systems genau wie Eingabedaten oder Parameter beeinflussen. Daher sollte auch die Anzahl der internen Speicher bei der Bewertung der Komplexität berücksichtigt werden (K8) /IST 10/:

$$Interne\ Speicher\ (FP) = \sum_{FB_i} Interne\ Speicher\ (FB_i)$$

Mittels des EVALUATOR können durch Auswertung der Tabellen zu den Funktionsbausteinen diese Werte bestimmt werden:

- Änderbare Parameter (K7)
- Interne Speicher (K8)

Komplexität der Funktionsbausteine

Für den Komplexitätsvektor wird die Komplexität der Funktionsbausteine nach generischen Attributen, wie beispielsweise dessen Ein- und Ausgangsports (in-ports, out-ports), internen Speichern (memories) und Parametern beurteilt. Diese Merkmale, die auch Teil der Black-Box Methode sind, lassen sich aus den Benutzerdokumenten oder

durch Anwendung von Statistischen Software Analysatoren wie Testbed von LDRA bestimmen /IST 10/.

Für alle generischen Attribute eines Funktionsbausteins werden so viele Fehlermöglichkeiten zugeordnet, wie der Baustein Attribute besitzt. Als typische Fehler werden Fehlbeschaltung von Ein- und Ausgängen, Fehlinterpretation zwischengespeicherter Werte (interne Memories) oder fehlerhaft belegte Parameter angesehen. Die Verwendungshäufigkeit eines Funktionsbausteins (quantity) wirkt sich Fehler korrigierend auf die Quantifizierung der Bausteinkomplexität aus. Das lässt sich damit erklären, dass die Häufigkeit der Verwendung eines Funktionsbausteins aufgrund von „Trainings-Effekten" proportional zur korrekten Verwendung des Funktionsbausteins gesehen wird. Die Ermittlung der Bausteinkomplexität erfolgt nach diesen Überlegungen nach folgender Formel /IST 10/:

$$Bausteinkomplexität = \frac{in_{ports} + out_{ports} + parameter + memories}{\log(quantity)}$$

Die vier oben beschriebenen Charakteristika der Software bilden die Komponenten des Komplexitätsvektors /IST 10/:

- Elementare Umfangsmaße
- Verschaltungs-Komplexität
- Variabilitäts-Komplexität
- Komplexität der Funktionsbausteine

Dabei umfassen die Merkmale ein breites Spektrum von Komplexitätsmerkmalen. Um die Aussagekraft der einzelnen Faktoren der Komplexität zu bewahren, wird die Komplexität nicht zu einer einzigen Zahl zusammengefasst /IST 10/.

Durch die Verwendung von Bayesian Belief Networks (BBN) lässt sich der Komplexitätsvektor in eine Zuverlässigkeitsaussage umwandeln /IST 10/.

Die Bestimmung des Komplexitätsvektors wurde von der IsTec im Rahmen des Projektvorhabens „Komplexitätsmessung der Software digitaler Leittechniksysteme" für das digitale I&C-System TELEPERM XS der Firma AREVA NP automatisiert. Die entwickelte

Methode lässt sich generell auf Leittechniksysteme übertragen, deren Funktionalität durch Funktionspläne beschrieben werden kann /IST 10/.

Um die Ergebnisse der Komplexitätsmessung zu präsentieren, wurde zur Auswertung ein digitales I&C-System in TELEPERM XS Technologie zur Begrenzung und Regelung eines DWR gewählt (ca. 1000 Funktionspläne). Dabei konnten ca. 90% (888 Funktionspläne) des Test-Systems der Komplexitätsmessung unterzogen werden. Die restlichen 10% waren Funktionspläne, die für den spezifischen Anwendungsfall noch nicht fertig implementiert waren /IST 10/.

Die komponentenweise Auswertung der Komplexitätswerte (K1-K9) hat laut IsTec die folgenden Vorteile /IST 10/:

- Die Automatisierung der Komplexitätsanalyse ermöglicht eine schnelle Analyse der Komplexität des Systems.
- Die Auswertung der Komponenten des Komplexitätsvektors ermöglicht die Funktionsbausteine und ihrer Verschaltung in den Funktionsplänen bezüglich ihres Ein- und Ausgabeverhaltens quantitativ zu beschreiben und damit die Qualifizierung des Systems.
- Die Komplexitätsvektoren bilden eine geeignete Grundlage für eine Zuverlässigkeitsbewertung der Funktionspläne.

F.4 Bestimmung der Zuverlässigkeit auf Grundlage der Komplexität

Hochqualifizierte Software wird den Vorgaben und Normen entsprechend stringenten Verifizierungs- und Validierungsverfahren unterworfen. Da die Software jedoch trotz umfangreicher Funktionstests nicht abdeckend getestet werden kann, können versagensauslösende Eingangsdaten jeder Zeit in der Software vorhandene Fehler zur Wirkung bringen. Dass die Komplexität ein geeignetes Maß für die Analyse der Software-Zuverlässigkeit darstellt, wird von der IsTec /IST 10/ damit begründet, dass die Testabdeckung mit Zunahme der Komplexität der Software abnimmt.

Ähnlich verhält es sich mit der Anzahl der änderbaren Parameter: Je größer die Zahl der änderbaren Parameter, desto fehleranfälliger wird das System.

F.4.1 Bayesian Belief Netze

Um aus dem Komplexitätsvektor zu einem integralen Zuverlässigkeitswert zu gelangen, werden die Informationen (Beobachtungen) schrittweise durch wohldefinierte und weiter entwickelbare Regeln in Bayesian Belief Netze (BBNs) eingefügt. Das Ziel ist es mittels der Bayesian Belief Netze aus den Komponenten des Komplexitätsvektors eine Verteilung für die Zuverlässigkeit abzuleiten /IST 10/.

Die Knoten des Netzwerkes werden aus Software Gestaltungs-Aspekten (Shape-Aspekte), wie beispielsweise die Verschaltungskomplexität, und Software Umfangs-Aspekte (Size-Aspekte), wie beispielsweise die Anzahl der Funktionsblöcke zusammengesetzt. Die Abhängigkeit dieser Knoten wird erst später bestimmt, so dass sie zunächst als ungerichtete Kanten miteinander verknüpft werden /IST 10/.

Das BBN wird schrittweise konstruiert:

1. **Festlegung der Eingabe-Knoten und ihre Initialwerte**

   Für alle Komponenten des Komplexitätsvektors wird geprüft, ob sie direkt als Eingabe-Knoten im BBN verwendet werden oder indirekt, da sie sich aus Eingabe-Knoten berechnen lassen. Ein Beispiel für einen indirekten Eingabe-Knoten ist die Variabilität, die sich aus der Anzahl der Speicher und der Parameter zusammensetzt. Die Anzahl der aus den Komponenten des Komplexitätsvektors ermittelten Knoten lässt sich so auf sechs reduzieren.

   Die Abhängigkeit der Knoten muss im Folgenden noch ermittelt werden /IST 10/.

2. **Festlegung der Topologie, d.h. Wahl geeigneter Zwischenknoten und der Verbindung (Kanten) zwischen den Knoten des BBN**

   Die Entwicklung der Topologie wurde von Experten festgelegt, die sowohl Fachwissen über die Funktionspläne und ihre Komplexität haben, als auch über große Erfahrung mit BBNs verfügen.

   Bei der Ermittlung der Eingabe- und Zwischenknoten wird unterschieden zwischen der internen Komplexität des Funktionsplans (Komplexität der Funktionsbausteine und deren Verschaltung, Anzahl der Funktionsbausteine und der externen Komplexität (Menge aller Ein- und Ausgangssignale, Menge der ein- und ausgangsseitig verbundenen Funktionen). Für diese beiden Komplexitätsmetriken werden dann Zwischenknoten definiert, die die Eingangsknoten verknüpfen. Zuletzt werden die

Zwischenknoten der internen und externen Komplexität zum Endknoten „Zuverlässigkeit“ verbunden. Abb. F.1 zeigt das fertiggestellte BBN /IST 10/.

3. **Zuordnung von bedingten Wahrscheinlichkeitstabellen (Conditional Probability Tables (CPTs)) zu allen Knoten mit eingehenden Kanten**

   In den Wahrscheinlichkeitstabellen (CPT) werden die bedingten Wahrscheinlichkeiten der Zwischenknoten des BBNs beschrieben. Dabei müssen die Wahrscheinlichkeiten für alle qualitativen Komplexitätskategorien des Knotens (low, median, high, very high, extreme high) unter der Voraussetzung der entsprechenden Konstellationen der Vorgänger Knoten bestimmt werden. Die Kategorie Kombinationen wurden zunächst mit Farbcodes dargestellt, um festzulegen, welche Kategorie Kombinationen die gleichen Komplexitätswerte ergeben sollten. Ein Experte übersetzte die qualitativen Aussagen der Farbskalen dann in Zahlenwerte /IST 10/.

Nachdem das BBN vollständig erstellt ist, wird ein Beispiel einer BBN-Analyse gegeben. Dabei werden die Werte des Komplexitätsvektors zunächst in die Eingabeknoten des BBN umgerechnet und den Werten die entsprechenden Kategorien zugeordnet (von sehr hohe Zuverlässigkeit (grün), hohe Zuverlässigkeit (gelb), mittlere Zuverlässigkeit (orange), geringe Zuverlässigkeit (rot) bis sehr geringe Zuverlässigkeit (violett)). Das Ergebnis der Analyse liefert den End-Knoten, der der Zuverlässigkeit des Funktionsplans entspricht. Angegeben wird mit welcher Wahrscheinlichkeit der Funktionsplan den fünf unterschiedlichen Kategorien (grün, gelb, orange, rot und violett) angehört. Aus diesen Angaben wird schließlich ein Erwartungswert bestimmt /IST 10/.

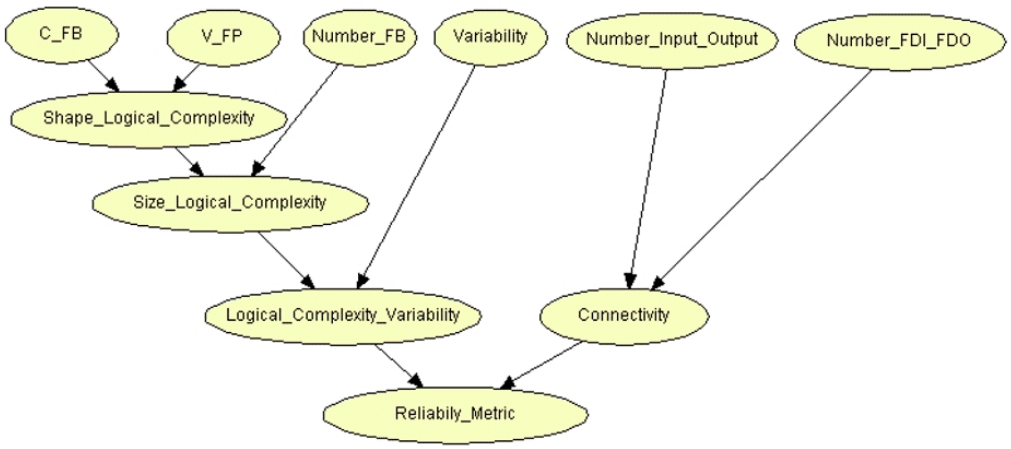

Abb. F.1 Bayessches Netz zur Zuverlässigkeitsbewertung /IST 10/

G Einsatz von Bayesschen Netzen zur Zuverlässigkeitsbewertung

Bei der Erstellung der Bayesschen Netze für die Zuverlässigkeitsbewertung sicherheitskritischer, rechnerbasierter Systeme durch die Finish Nuclear Safety Research Organization (STUK) wird der Fokus auf Erfahrungswerte aus dem betrieblichen Gebrauch und der Testphase (sog. *hard evidence*) gelegt. Diese Werte haben, im Gegensatz zu qualitativen Charakteristika wie Software-Design und Merkmale des Entwicklungsprozesses (sog. *soft evidence*), den Vorteil, dass sie keine umfangreiche Schätzung von Experten erfordern /STUK 01/.

Lediglich bei der ersten Abschätzung der Priori Wahrscheinlichkeitsverteilung werden quantitative und qualitative Charakteristika aus dem Software-Entwicklungsprozess, aus den ersten Tests und den Design Merkmalen während des Produktionsprozesses verwendet. Diese Abschätzung wird dann mit *hard-evidence*-Werten aus den betrieblichen Tests und der betrieblichen Erfahrung zur Posteriori Funktion aktualisiert. Die Verschiedenartigkeit der Charakteristika wird dann in der Strukturierung des Bayesschen Netzes mitberücksichtigt /STUK 01/.

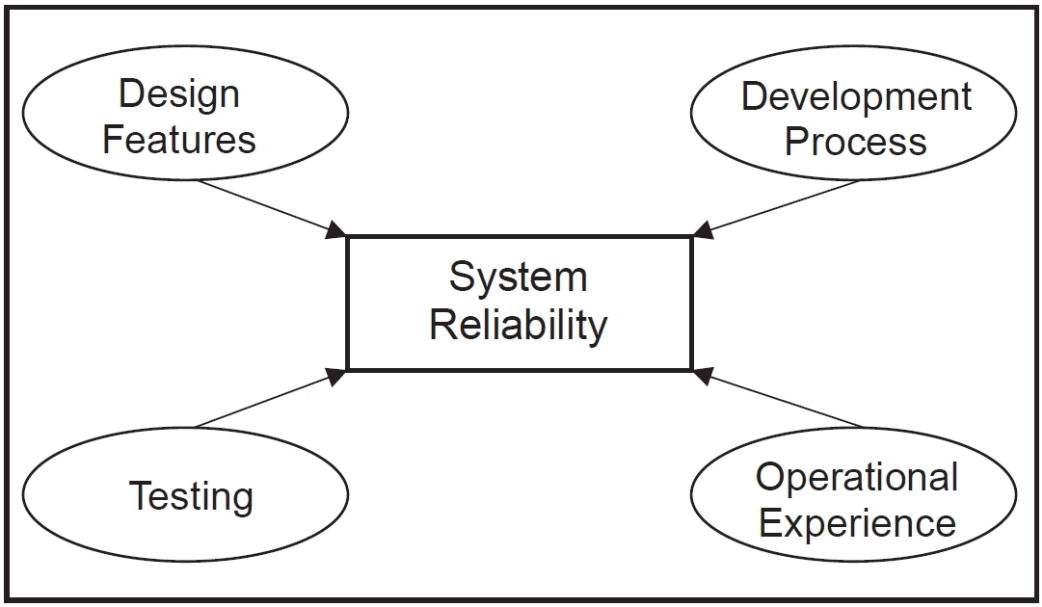

Abb. G.1 Hauptcharakteristika der Software, die für die Erstellung der Bayesschen Netze zur Zuverlässigkeitsbewertung eines sicherheitsrelevanten Systems verwendet werden. Entnommen aus /STUK 01/

Der erste Schritt der Methode ist das Aufstellen der Priori Ausfallwahrscheinlichkeit die auf den Zufallsparametern beruhen. Dabei richtet sich die Wahl der Priori Wahrscheilichkeitsverteilung auch nach der Likelihood Wahrscheinlichkeitsfunktion. Letztere ergibt sich aus der Maximum-Likelihood-Methode. Diese statistische Methode wird angewandt, um die Parameter zu ermitteln, die die beobachteten Daten am besten beschreiben /STUK 01/.

Häufig wird dabei eine konjugierte Priori Wahrscheinlichkeit gewählt, so dass sich die Berechnung vereinfacht und Priori- und Posteriori-Verteilungen den gleichen Verteilungstyp besitzen. Im Falle einer binominalverteilten Likelihood Wahrscheinlichkeitsfunktion würde die konjugierte Priori Wahrscheinlichkeit einer Beta Funktion entsprechen. Da aber Verteilungen von unterschiedlichen Betriebsprofilen flexibel kombiniert werden sollen, wird für die Priori Wahrscheinlichkeitsfunktion P der Parameter eine normalverteilte Logit Transformierte Versagenswahrscheinlichkeit θ definiert:

$$P = \frac{e^{\theta}}{1 + e^{\theta}}$$

Die Wahl der Priori Verteilung ist ein wichtiger Aspekt bei der Bayessche Inferenz. Die Abschätzung der Priori Wahrscheinlichkeitsverteilung erfolgt durch einen Gutachter auf Grundlage der Parameter, die zu Versagen führen können. Wenn es nur wenige Informationen über diese Parameter gibt, so sollte die Priori Wahrscheinlichkeitsverteilung flach gewählt werden. Die Berechnungen der Studie zeigen, dass der Einfluss der Priori Verteilung mit ansteigender Datenmenge geringer wird /STUK 01/.

Für die Likelihood Funktion, die die Ausfallwahrscheinlichkeit bei Anforderungen beschreibt wird eine Binominalverteilung verwendet, da diese die Anzahl der Ausfälle bei einer bestimmten Anzahl an Anforderungen statistisch definiert als:

$$f(y|\mathrm{p},\mathrm{n}) = \frac{n!}{y!(n-y)!} p^{y}(1-p)^{n-y} \qquad \mathrm{y} = 0,\ldots,\mathrm{n},$$

mit n die Anzahl der Anforderungen und p = die Ausfallwahrscheinlichkeit ist.

Wenn Ausfallwahrscheinlichkeiten für zwei Betriebsprofile berechnet werden, werden zuerst die Binominalverteilungen der Likelihood Funktionen aus den *hard evidence* Werten (statistische Messwerte aus dem Betrieb und der Testphase) der beiden Betriebsprofile berechnet. Die Parameter für die Versagenswahrscheinlichkeit der Priori Logit

Transformierten werden zusammengefasst und angeglichen. Zusätzlich wird eine normalverteilte Zufallsgröße Ω‘ (μ‘, σ‘) zur Priori Wahrscheinlichkeitsfunktion dazu addiert. Dieser zusätzliche Term berücksichtigt die Unterschiede der beiden Betriebsprofile und seine beiden Parameter μ‘ und σ‘ sind ein Maß für die Qualität der Softwareentwicklungstests. Da es zu zeitaufwändig ist, ein System mit den maximal möglichen Eingangsdaten zu testen, kann jeder Test nur eine begrenzte Anzahl an Eingangsdaten abdecken. Die Parameter μ‘ und σ‘ beschreiben wie gut die getesteten Eingangsdaten der unterschiedlichen Betriebsprofile das gesamte Spektrum an betrieblichen Eingangsdaten abdecken und bis zu welchem Maß auch die Wahrscheinlichkeit anderer Eingangsdaten mitberücksichtigt wurde /STUK 01/.

Bei negativem μ‘-Wert sind die Eingangsdaten des Betriebsprofils weniger umfangreich als die Eingangsdaten aus der betrieblichen Praxis. Bei positivem μ‘-Wert sind die Testdaten so komplex, dass die Menge von Testdaten anderer Betriebsprofile reduziert werden kann. Für den Fall, dass die Erfahrungswerte aus den Tests und dem betrieblichen Gebrauch gleich sind, hat Ω‘ (μ‘, σ‘) einen Mittelwert und eine Varianz von Null. Diese Vorgehensweise lässt sich auch auf die Berechnung mit mehreren Betriebsprofilen übertragen /STUK 01/.

Für die Berechnung der Ausfallwahrscheinlichkeit verschiedener Systeme, wenn beispielsweise die alte und neue, korrigierte Version einer Software untersucht werden sollen, wird eine ähnliche Vorgehensweise vorgeschlagen. Dabei wird zunächst jede Version einzeln modelliert und beide Systeme dann kombiniert, indem der Unterschied der Software Versionen durch eine normalverteilte Zufallsgröße Ω_1 mit dem Parameter ϑ_1 zur Logit Transformierten dazu addiert wird. Der Vorteil dieser Methode liegt darin, dass die Ausfallwahrscheinlichkeit nicht für jedes System einzeln bestimmt werden muss, sondern auch Informationen von älteren Versionen in die Zuverlässigkeitsbewertung eingehen können /STUK 01/.

Im Einzelnen wurden in der Studie die folgenden Modelle berechnet:

1. **Modell 1:** Zuverlässigkeitsbewertung für ein rechnerbasiertes System mit einem Betriebsprofil. Als Likelihood-Verteilung wird eine Binominalverteilung gewählt, die auf Testdaten basiert.

2. **Modell 2:** Zuverlässigkeitsbewertung für ein rechnerbasiertes System mit zwei unterschiedlichen Betriebsprofilen. Dazu wird das Bayesschen Netzwerk, wie beschrieben, um die normalverteilte Zufallsgröße Ω‘ (μ‘, σ‘) ergänzt.

3. **Modell 3:** Zuverlässigkeitsbewertung für ein rechnerbasiertes System mit mehreren Betriebsprofilen. Dabei werden auch Test und Betriebserfahrungen der (gleichen) rechnerbasierten Systeme aus unterschiedlichen Kernkraftwerken betrachtet, die unter unterschiedlichen Betriebs- und Umgebungsbedingungen eingesetzt werden. Dazu wird, ähnlich wie in Modell 2, eine normalverteilte Zufallsgröße Ω‘ definiert, die zur Logit Transformierten dazu addiert wird.

4. **Modell 4:** Zuverlässigkeitsbewertung für ein rechnerbasiertes System mit mehreren Betriebsprofilen und Informationen von unterschiedlichen Systemen, die jedoch eine gemeinsame Grundlage haben. Ein Beispiel dafür ist die Zuverlässigkeitsbewertung eines rechnerbasierten Systems unter Zuhilfenahme der Vorgängerversionen. Dabei wird für jede Version vorgegangen wie für Modell 3 und die einzelnen Systemversionen werden dann schließlich zusammengefügt /STUK 01/.

Gesellschaft für Anlagen- und Reaktorsicherheit (GRS) gGmbH

Schwertnergasse 1
50667 Köln
Telefon +49 221 2068-0
Telefax +49 221 2068-888

Forschungszentrum
Boltzmannstraße 14
85748 Garching b. München
Telefon +49 89 32004-0
Telefax +49 89 32004-300

Kurfürstendamm 200
10719 Berlin
Telefon +49 30 88589-0
Telefax +49 30 88589-111

Theodor-Heuss-Straße 4
38122 Braunschweig
Telefon +49 531 8012-0
Telefax +49 531 8012-200

www.grs.de

ISBN 978-3-946607-23-6