BERGISCHE UNIVERSITÄT WUPPERTAL

FAKULTÄT FÜR MASCHINENBAU UND SICHERHEITSTECHNIK

# Zuverlässigkeitsbewertung von fehlertoleranten Systemarchitekturen für hoch- und vollautomatisierte Fahrfunktionen unter Anforderungen der funktionalen Sicherheit

MASTER THESIS

ZUR ERLANGUNG DES AKADEMISCHEN GRADES
MASTER OF SCIENCE

**Autor:**
Tim M. Julitz

**Matrikelnummer:**
1540160

**Erster Prüfer:**
Jun.-Prof. Dr. Antoine Tordeux

**Zweiter Prüfer:**
Basma Khelfa

23. August 2021

# Inhaltsverzeichnis

# 1 Die Bedeutung von Fehlertoleranz für autonome Fahrzeugsysteme

Mit dem Gesetzesentwurf zur Änderung des Straßenverkehrsgesetzes und des Pflichtversicherungsgesetzes wird die Entwicklung von automatisierten Fahrzeugsystemen auf die nächste Stufe gehoben [1]. Durch das Gesetz werden die Voraussetzungen für den Einsatz von hochautomatisierten Fahrzeugen (SAE Stufe 4) im öffentlichen Straßenverkehr geschaffen. Bereits 2017 trat die achte Änderung des Straßenverkehrsgesetzes in Kraft, die den Betrieb von Stufe 3 Fahrzeugen ermöglichte [2].

Die Anfänge der Forschung zur Fahrzeugautomatisierung gehen auf das PROMETHEUS Pojekt zurück, welches im Jahr 1986 startete und die Grundsteine heutiger kommerzieller Fahrerassistenzsysteme bis zur Stufe 2 legte [3]. Einige Beispiele sind der Abstandstempomat Distronic Plus und der Notbremsassistent Pre-Safe von Daimler. Die nächsten Meilensteine in der Entwicklung von autonomen Fahrzeugen wurden mit dem Start der DARPA Challenges im Jahr 2004 erzielt. 2005 bahnten sich fahrerlose Fahrzeuge über 212 km ihren Weg durch die Mojave-Wüste, wobei das autonome navigieren im Vordergrund stand [4]. Im Folgeprojekt 2007 wurde städtischer Verkehr auf einem verlassenen Air-Force-Stützpunkt simuliert [5]. Diesmal stand die Beachtung von Verkehrsregeln im Mischverkehr im Fokus. Aus den DARPA Challenges gingen eine Vielzahl von Software Architekturen hervor, die eine wesentliche Gemeinsamkeit aufweisen: Die Architekturen sind in Module unterteilt, die verschiedene Funktionen erfüllen. Reke et al. fassen zusammen, dass die Module im Wesentlichen aus Lokalisierung, Wahrnehmung und Fahrzeugsteuerung bestehen [6]. Das Team des Siegerfahrzeuges von 2005 identifizierte einige erhebliche Probleme. Das entwickelte Fahrzeug konnte in der statischen Umgebung erfolgreich bestehen, eine Navigation durch den Straßenverkehr ist jedoch aufgrund der unzureichenden Zuverlässigkeit des Systems nicht möglich [7]. Die Hardware Architektur des Fahrzeugs bestand aus sechs Computern, die verschiedene Funktionen übernahmen. Watchdogs haben die Zustände von Software und Hardware überwacht, um im Fehlerfall das System neuzustarten. Während der 2007er DARPA Challenge konnten dann Erfolge im überwachten urbanen Umfeld erzielt werden. Einen wichtigen Beitrag erbrachte die gesteigerte Berücksichtigung von fehlerbehebenden Maßnahmen. Das erstplatzierte Fahrzeugsystem setzte auf verschiedene Betriebsmodi: Einem Normalzustand und einem Wiederherstellungszustand [8]. Der Wiederherstellungszustand wird ausgelöst, wenn Objekte den geplanten Pfad blockieren, Objekte zu spät detektiert werden oder Aktionen kinematisch nicht machbar sind. Zur Rückkehr in den Normalbetrieb sorgen vier Algorithmen, die im Wesentlichen

aus der Neuplanung von Pfaden und Erhöhung des Sicherheitsanstandes bestehen [8]. Es handelt sich also um softwarebasierte Lösungen zur Erhöhung der Robustheit. Die hardwareseitigen Maßnahmen bestanden aus dem Einsatz eines Dual-Core CPUs und der Kombination verschiedener Sensoren. Dennoch stellten Urmson et al. fest, dass die Zuverlässigkeit und Robustheit ihres Fahrzeugs nicht ausreicht, um im realen Straßenverkehr zu fahren, da dieser erheblich komplexer ist als das überwachte Umfeld [8]. Das Fahrzeug konnte sich zwar von vielen Fehlerzuständen erholen, die Zeit die es dazu brauchte war jedoch mit bis zu zehn Minuten erheblich, die im realen Umfeld nicht zur Verfügung steht, weswegen das entwickelte System auch nicht mit der SAE Stufe 4 klassifiziert werden kann.

In den folgenden Jahren gab es eine Vielzahl von ähnlichen Projekten aus dem industriellen und akademischen Umfeld, die den Stand der Technik weiter ausgebaut haben. Auf dem Weg zum fahrerlosen Betrieb der SAE Stufe 4 zeigen wiederholte Unfälle von automatisierten Fahrzeugsystemen, dass weitere Forschung zur Erhöhung der Sicherheit, Zuverlässigkeit und Robustheit der komplexen Systeme erforderlich ist [9, 10]. Durch den Wegfall der menschlichen Rückfallebene werden fehlertolerante Ansätze der Systemmodellierung erforderlich, die durch einfache Redundanz alleine nicht umgesetzt werden können. Die Prinzipien der Fehlertoleranz beruhen auf Selbstdiagnose, Zuverlässigkeit, Verfügbarkeit, Rekonstruktion und Fehlerbehebung. Die Zuverlässigkeit beschreibt die Fehlerfreiheit eines Systems über die Zeit. Sie wird in der Regel durch strukturelle Redundanz erhöht. Die Verfügbarkeit gibt an, ob ein System zu einem bestimmten Zeitpunkt funktioniert und kann durch diversitäre Redundanz oder Separation bzw. Unabhängigkeit beeinflusst werden, wobei auch von asymmetrischen Systemarchitekturen gesprochen wird. In traditionellen sicherheitskritischen Systemen, die u.a. in der Luftfahrt, im Schienenverkehr, in der Raumfahrt, im Militär oder in Atomkraftwerken zu finden sind, werden diese Prinzipien bereits angewendet. Aber auch in der automobilen Fachliteratur gewinnen sie zunehmend an Bedeutung, vgl. [11–16].

Die Entwicklung einer fehlertoleranten Systemarchitektur stellt eine der wichtigsten Herausforderungen für die Markteinführung von autonomen Fahrzeugen dar. Eine Vielzahl von Arbeiten beschäftigen sich mit der Modellierung von fehlertoleranten Ansätzen. Dabei betrachten sie jedoch vorwiegend isoliert einzelne Aspekte der Fehlertoleranz. Andere Aspekte werden wiederum kaum betrachtet. Die Arbeit von Dai et al. befasst sich beispielsweise als einer der wenigen Arbeiten mit der Zuverlässigkeit von fehlertoleranten Fahrzeugarchitekturen [17]. In vielen Veröffentlichungen geht es ausschließlich um Zuverlässigkeit ohne Bezug auf automotive Systeme zu nehmen. Die vorliegende Thesis wird unter Berücksichtigung von Selbstdiagnosefunktionen einen Schwerpunkt auf die Analyse der Zuverlässigkeit von fehlertoleranten Fahrzeugsystemen legen. Sie kombiniert damit zwei Aspekte der Fehlertoleranz. Am Ende der Thesis

wird anschließend anhand der analysierten Systeme eine Vorgehensweise aufgezeigt, wie alle Aspekte der Fehlertoleranz kombiniert werden können. Von besonderem Interesse ist die Frage, inwiefern fehlertolerante Architekturen die quantitativen Anforderungen der funktionalen Sicherheit der ISO 26262 erfüllen und welche Architekturen hierfür am besten geeignet sind. Die Thesis geht dieser Frage nach, indem sie folgender Struktur folgt (Abb. 1.1).

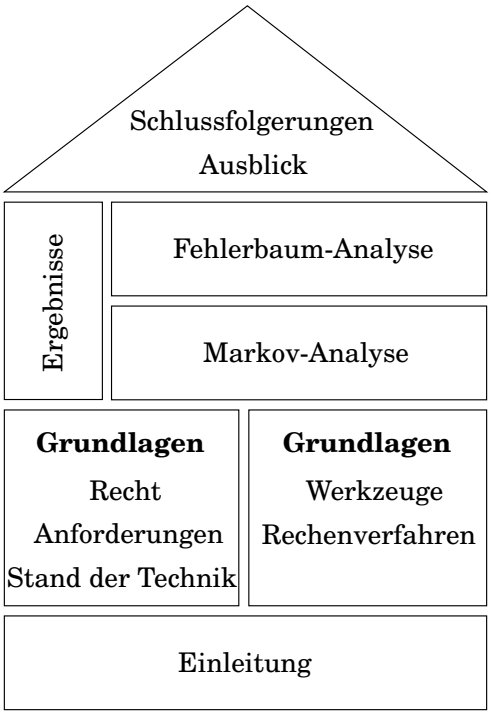


**Abb. 1.1:** Aufbau der Thesis

In Kapitel 2 werden rechtlichen Grundlagen erläutert und eine Einführung in die funktionale Sicherheit gegeben. Die Anforderungen der ISO 26262 stellen die Grundlage der Systembewertung dar. Anschließend werden Ansätze zur Modellierung von Fehlertoleranz und etablierte fehlertolerante Hardwarearchitekturen aufgezeigt. In Kapitel 3 folgt die Beschreibung von Werkzeugen, die für die Analyse des Ausfallverhaltens von Systemen verwendet werden können. Anhand von Minimalbeispielen werden exemplarisch Rechnungen vorgestellt. Die Systemanalyse findet in Kapitel 4 statt, welches in eine Markov- (Abs. 4.2) und in eine Fehlerbaumanalyse (Abs. 4.3) unterteilt ist. Mit Hilfe der Markovanalyse werden zur Erreichung der Anforderungen geeigneten Systemarchitekturen ermittelt. Die Komponenten der identifizierten Architekturen werden anschließend durch eine Fehlerbaumanalyse hinsichtlich ihrer Ausfallraten ausgelegt, um quantitative Aussagen zu den Systemeigenschaften treffen zu können. Die gewon-

nenen Erkenntnisse werden in Kapitel 5 diskutiert. Außerdem wird aufgezeigt, wie die Erkenntnisse in weiterführenden Forschungsprojekten verwendet werden können.

# 2 Anforderungen an fehlertolerante Systemarchitekturen sowie etablierte Konzepte

## 2.1 Rechtliche Grundlagen der Produktsicherheit

Die europäische Union mit ihren Mitgliedstaaten verfolgt die Strategie die Produktsicherheit zu harmonisieren, um die Handelshemmnisse im Binnenmarkt abzubauen. Der EU-Vertrag basiert dabei auf zwei wesentlichen Säulen: Die Binnenmarkt-Richtlinien und die Arbeitsschutz-Richtlinien (Abb. 2.1). Für die Maschinenhersteller sind v.a. die Binnenmarkt-Richtlinien mit der **Maschinenrichtlinie** von zentraler Bedeutung. Sie etabliert durch definierte Schutzziele und grundlegende Sicherheitsanforderungen ein einheitliches Schutzniveau zur Unfallverhütung von Maschinen beim Inverkehrbringen innerhalb des europäischen Wirtschaftsraums.

Das Inverkehrbringen ist dabei definiert als, die erstmalige entgeltliche oder unentgeltliche Bereitstellung eines Produktes auf dem europäischen Binnenmarkt im Rahmen einer Geschäftstätigkeit [18]. Demnach sind nicht nur Maschinenhersteller oder Bevollmächtigte mit dem Sitz in der EU betroffen, sondern auch Händler und Importeure die ihre Waren in die EU einführen.

Richtlinien die aufgrund des EU-Vertrags erlassen wurden entfalten erst nach der Umsetzung in nationales Recht ihre Wirkung. Für die Maschinenrichtlinie ist das in Deutschland durch das **Produktsicherheitsgesetz** geschehen, welches durch zahlreiche Verordnungen gestützt wird. Die Maschinenverordnung (9. ProdSV) ist die direkte Umsetzung der europäischen Richtlinie. Jedoch zitiert Anhang 1 der Maschinenverordnung die Maschinenrichtlinie direkt. Somit erfährt die Maschinenrichtlinie in Deutschland auch ihre direkte Anwendung. Wesentliche Anforderungen der Maschinenrichtlinie sind die Harmonisierung von Normen, die Durchführung von Risikoanalysen, eine technische Dokumentation, CE-Konformität, die Verfassung einer Einbauerklärung, die Integration einer Risikobeurteilung in den Herstellungsprozess und die Integration des CE-Prozesses im Unternehmen. Das CE-Zeichen dient der europaweiten Vereinheitlichung von Sicherheitsstandards für Maschinen.

Gesetze und EG-Richtlinien haben verbindlichen Charakter und müssen von jedem Inverkehrbringer angewendet werden. **Normen** dagegen stellen empfohlene Regeln oder Leitlinien für wiederkehrende Anwendungen dar. Das heißt, sie sind Standardlösungen für bekannte Probleme, um den Stand der Technik zu erfüllen. Trotzdem

ist eine blinde Anwendung nicht zu empfehlen, denn der durch die Normen abgebildete Stand der Technik entspricht nur dem Stand von der Veröffentlichung, welcher sich in der Zwischenzeit geändert haben könnte. Ihre Anwendung ist freiwillig.

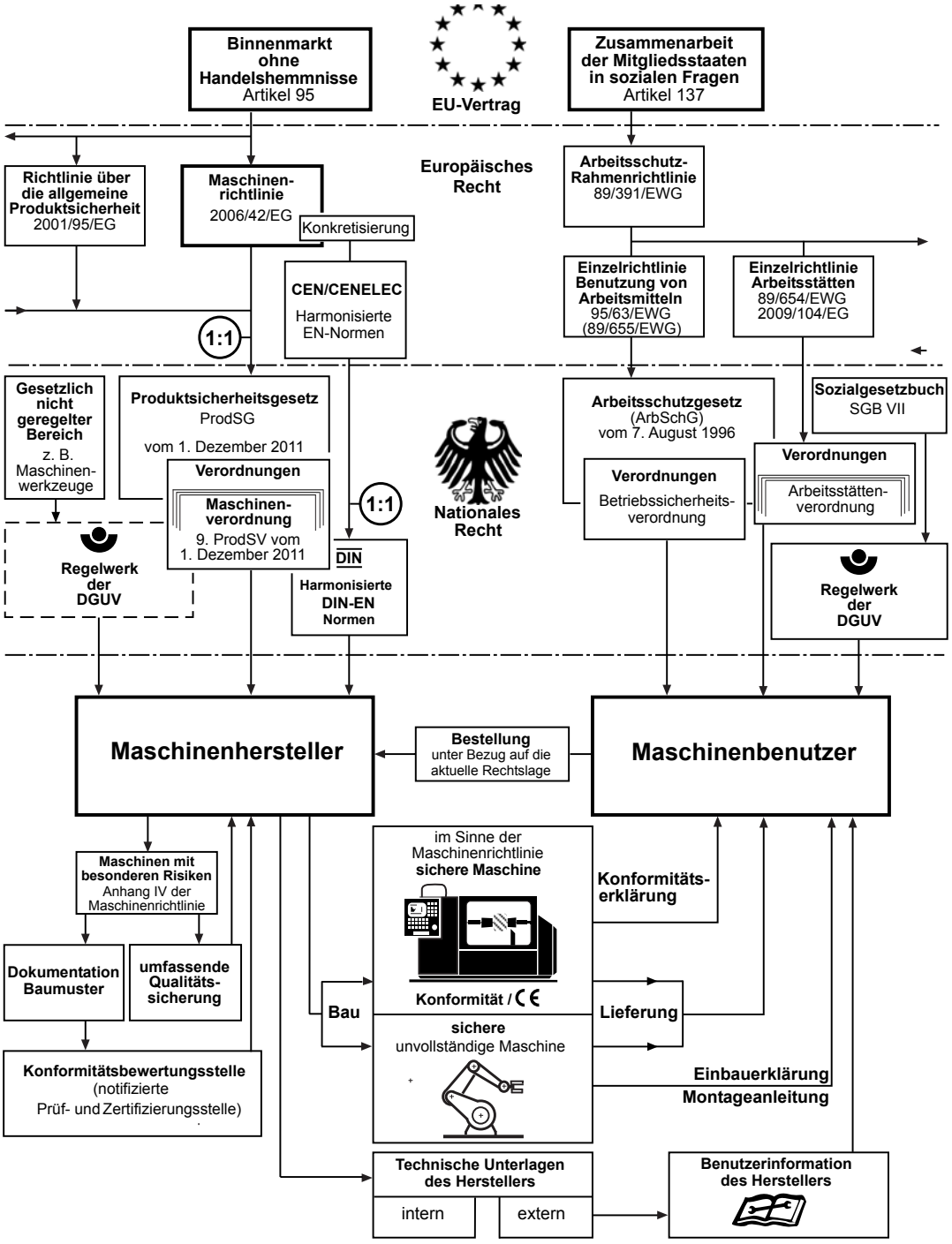


**Abb. 2.1:** Europäisches und nationales Produktsicherheitsrecht [19]

Sie konkretisieren vorhandene Gesetze und Verordnungen im Sinne von Anhang 1 der Maschinenrichtlinie. Mit ihrer Anwendung greift die sogenannte Konformitätsvermutung mit den zugrunde liegenden Gesetzestexten. Für den Bereich der Straßenfahrzeuge kann die ISO 26262 verwendet werden, um die Konformität mit der Maschinenrichtlinie darzulegen (Abschnitt 2.2). Auch wenn ihre Anwendung freiwillig ist, wird sie von vielen Erstausrüstern – englisch: Original Equipment Manufacturer (OEM) – vorausgesetzt.

## 2.2 Funktionale Sicherheit von Straßenfahrzeugen – ISO 26262

### 2.2.1 Einführung in die funktionale Sicherheit

Die ISO 26262 („Road vehicles – Functional safety“) ist eine Norm für sicherheitsrelevante elektrische, elektronische (E/E) Systeme in Kraftfahrzeugen. Sie ist die sektorspezifische Anwendung der Metanorm IEC 61508, die für die Entwicklung von elektrischen, elektronischen und programmierbar elektronischen (E/E/PE) Systemen mit Sicherheitsfunktion anwendbar ist (Abb. 2.2).

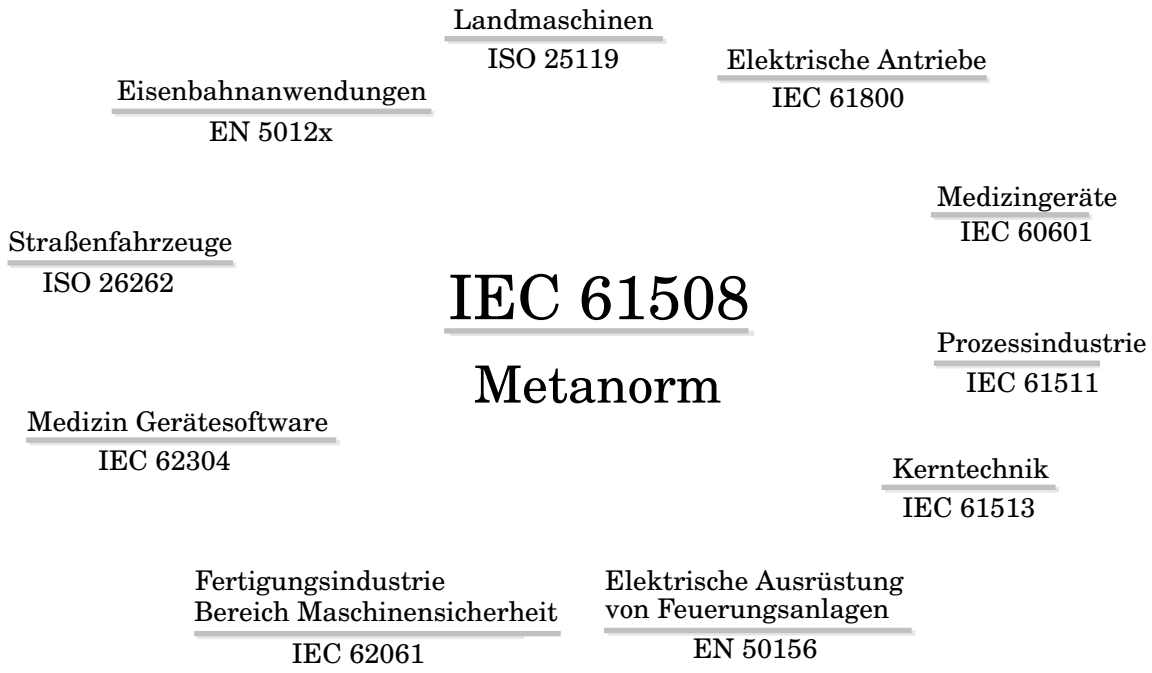


**Abb. 2.2:** Die Normenfamilie der funktionalen Sicherheit

Während der zunehmender Automatisierung und Elektrifizierung im Fahrzeugbau kamen viele verschiedene Methoden von software-basierten Sicherheitsmechanismen zum Einsatz, die den unterschiedlichsten Systementwicklungsansätzen unterlagen. Der wesentliche Zweck der ISO 26262 sollte es sein, sich auf ein Grundverständnis im Sys-

temengineering zu einigen [20]. Anfang des 21. Jahrhunderts ist das Thema der Funktionssicherheit in der Automobilbranche aufgekommen, worauf sich die ersten Arbeitskreise und Gremien bildeten. Bis zur Veröffentlichung der ersten Auflage im Jahr 2011 ist viel Wissen, Methodik und Lösungsansätze diskutiert worden, die in normative und informative Kapitel eingeflossen sind [20]. Die erste Auflage zielte auf alle Straßenfahrzeuge bis zu 3,5 t ab, die mit Sicherheitssystemen, bestehend aus mindestens einem E/E-System, in Serie produziert wurden. Die Neuauflage im Jahr 2018 erweiterte den Anwendungsbereich auch auf Straßenfahrzeuge jenseits der Gewichtsbegrenzung sowie auf Motorräder.

Die funktionale Sicherheit bezeichnet die korrekte Erbringung der Sicherheitsfunktion eines E/E-Systems. Die Funktionssicherheit ist gegeben, wenn keine unakzeptablen Risiken vorliegen, die auf Gefahren basieren, welche durch die Fehlfunktion von sicherheitsrelevanten E/E-Systemen verursacht werden. Nicht behandelt werden Gefahren, die im Zusammenhang mit Stromschlägen, Brand, Rauch, Wärme, Strahlung, Toxizität, Reaktivität, Korrosion, Energiefreisetzung oder ähnlichem stehen, sofern dies nicht direkt durch fehlerhaftes Verhalten sicherheitsrelevanter E/E-Systeme verursacht wurde [21]. Die ISO 26262 ist ein Managementsystem zur Reduzierung von Risiken, um die Funktionssicherheit zu erreichen. Dafür werden für den gesamten Sicherheitslebenszyklus des Fahrzeugs (d.h. während Entwicklung, Produktion, Service und Stilllegung) Anforderungen und Prozesse vorgegeben. Es wird ein spezifischer risikobasierter Ansatz zur Bestimmung von automotiven Sicherheitsintegritätsstufen – englisch: Automotive Safety Integrity Levels (ASILs) – zur Verfügung gestellt, mit denen angegeben wird, welche der Anforderungen der ISO 26262 eingehalten werden müssen. Die Anforderungen bestehen aus Spezifikationen für das funktionale Sicherheitsmanagement, für das Design und für Implementierungen am realen Fahrzeug sowie aus Verifizierungs-, Validierungs- und Bestätigungsmaßnahmen. Außerdem werden Anforderungen an die Kunden-Lieferanten-Beziehung gestellt [21]. Abbildung 2.3 zeigt die an dem V-Modell angelehnte Struktur der ISO 26262 Normenserie.

Das V-Modell beschreibt ein generisches Vorgehen bei der Entwicklung von mechatronischen Systemen. Vom V aus linksseitige Kapitel beschreiben den Systementwurf mit der Ausarbeitung von Spezifikationen, die auf die eingangs definierten Anforderungen basieren. Hierbei ist mit besonderer Sorgfalt vorzugehen, denn die Spezifikationen stellen gleichzeitig den Bewertungsmaßstab dar. Rechtsseitige Kapitel gehen auf die Systemintegration ein. Mit Verifikations- und Validierungsverfahren wird überprüft, ob die gestellten Anforderungen eingehalten werden. Hierfür ist die Verwendung von Prototypen üblich. Kapitel 1 und 2 sowie 8 bis 11 beinhalten allgemein gültige Konzepte und Definitionen die kontinuierlich anzuwenden sind. Die Kapitel 9 und 10 sind informative Abschnitte, die Erfahrungen über spezifische Anwendungsfälle, die bis zum

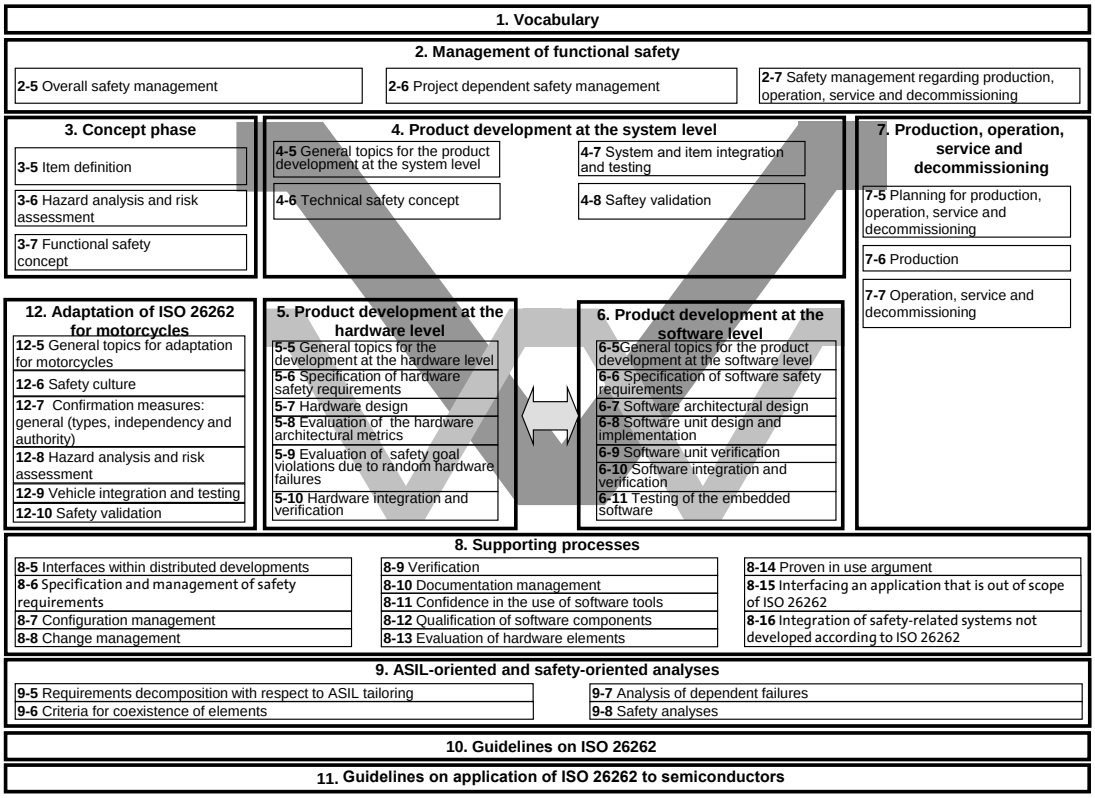


**Abb. 2.3:** Struktur der ISO 26262 Normenserie [21]

Zeitpunkt der Veröffentlichung der Norm gesammelt wurden für die Anwendung zur Verfügung stellen. Ein Abschnitt mit herausragender Bedeutung ist Kapitel 3 „Konzept Phase“, auf welches genauer in Abschnitt 2.2.2 eingegangen wird. Hier werden die ASILs festgelegt, woraus sich die zu erfüllenden Anforderungen ergeben, dessen Ermittlung wie zuvor dargestellt eine der wichtigsten Schritte, wenn nicht der wichtigste Schritt, während der Produktentwicklung ist.

### 2.2.2 Managementaktivitäten entlang des Sicherheitslebenszyklus

Die ISO 26262 legt Anforderungen in Bezug auf definierte Phasen des Sicherheitslebenszyklus fest (Abb. 2.4). Einige Anforderungen gelten auch für mehrere oder alle Phasen. Die wichtigsten Aufgaben des Sicherheitsmanagements sind die Planung, Koordinierung und Verfolgung der Aktivitäten bezüglich der funktionalen Sicherheit. Diese Aufgaben gelten für alle Phasen und werden nochmal unterteilt in das übergeordnete Sicherheitsmanagement, das projektbezogene Sicherheitsmanagement und das Sicherheitsmanagement in Bezug auf Produktion, Betrieb, Service und Stilllegung [22].

Die erste Handlung im Sicherheitslebenszyklus ist die Defintion des betrachteten Systems, welches in der ISO 26262 als „Item“ definiert ist. Es kann ein System oder

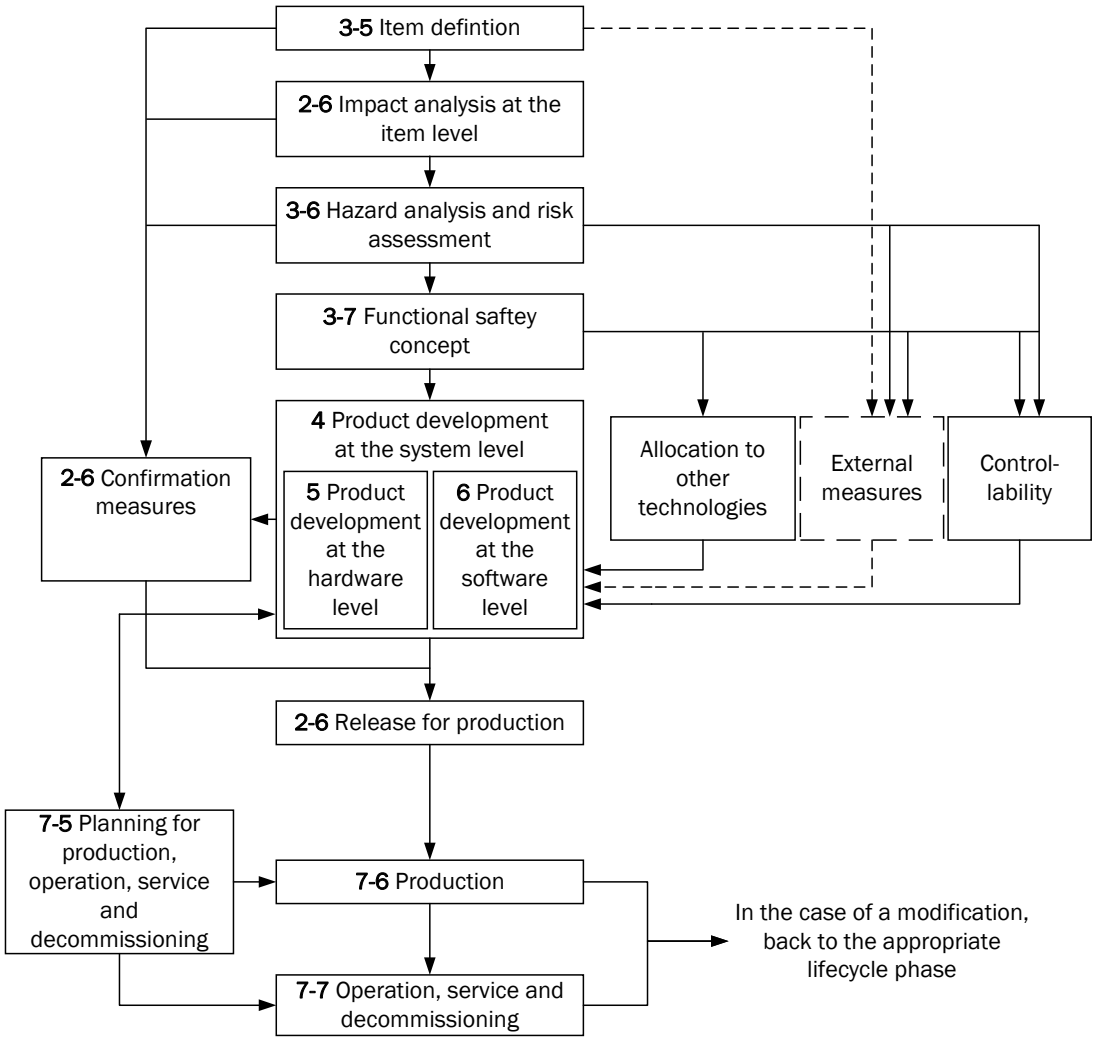


**Abb. 2.4:** Managementaktivitäten im Sicherheitslebenszyklus nach ISO 26262 [22]

eine Kombination von Systemen sein, die eine Funktion oder einen Teil davon auf der Fahrzeugebene ausführt [21]. Das Item wird in Bezug auf seine Funktionalität, Schnittstellen, Umgebungsbedingungen, gesetzlichen Anforderungen, bekannte Gefahren, seiner Grenzen und möglichen Wechselwirkungen beschrieben [23]. Mit der darauf folgenden „Impact analysis“ wird die Neuartigkeit des Items ermittelt. Es wird geprüft, ob es sich um ein neues System, um eine Modifizierung oder ein bestehendes System unter neuen Umgebungsbedingungen handelt. In der „Hazard analysis“ werden zunächst alle Szenarien und Betriebsmodi beschrieben, die zu gefährlichen Ereignissen führen können [23]. Die Gefahren inklusive ihrer Auswirkungen werden systematisch hinsichtlich der möglichen Fehlfunktionen auf der Fahrzeugebene ermittelt. Die gefährlichen Ereig-

nisse werden mit Hilfe der ASILs klassifiziert. Dies geschieht über Tabellen, die die Schadensschwere, die Eintrittswahrscheinlichkeit und die Kontrollierbarkeit genauer einordnen. Der Anhang von ISO 26262 Teil 3 stellt einen Leitfaden zur Verfügung, der quantifiziert beschreibt welche Ausprägungen der jeweiligen Attribute in speziellen Szenarien zu wählen sind [23]. Die ASILs werden über eine Kombination der Tabellenwerte bestimmt. Sie reichen von QM (Qualitätsmanagement) für die niedrigste Stufe, über ASIL A bis ASIL D für die höchste Stufe. Für jedes gefährliche Ereignis werden anschließend Sicherheitsziele definiert, die das jeweilige ASIL übernehmen. Sie stellen die übergeordneten (top-level) Sicherheitsanforderungen des Items dar und werden funktional beschrieben. Werden ähnliche Sicherheitsziele ermittelt, können diese kombiniert werden. Hierfür wird das höchste ASIL der einzelnen Ziele angesetzt. In späteren Phasen werden detaillierte Sicherheitsanforderungen von den Sicherheitszielen abgeleitet. Die entsprechenden ASILs werden übertragen oder durch Dekomposition angepasst.

Auf den Sicherheitszielen aufbauend wird ein funktionales Sicherheitskonzept entwickelt, indem funktionale Sicherheitsanforderungen von den Sicherheitszielen abgeleitet und auf die Elemente der Systemarchitektur bezogen werden. Ein Sicherheitsziel kann durch eine oder mehrere funktionale Sicherheitsanforderungen erfüllt werden. Über diese Anforderungen sollen Strategien für Fehlervermeidung, Fehlererkennung, den Umgang mit Fehlern, Fehlertoleranz, Fahrerwarnungen und zeitliche Koordinierung von Maßnahmen auf der Fahrzeugebene umgesetzt werden [23]. Jede funktionale Sicherheitsanforderung berücksichtigt Betriebsmodi, Zeitintervalle für Fehlertoleranz und Notbetrieb sowie funktionale Redundanzen [23]. Die hier definierten Anforderungen dienen als Referenz für spätere Verifikationen.

Nach der Anfertigung des funktionalen Sicherheitskonzepts wird das Item auf der Systemebene entwickelt. Der Prozess ist am V-Modell angelehnt (vgl. Abb. 2.3). Auf der linken Seite findet die Spezifikation der technischen Anforderungen, der Systemarchitektur und das Systemdesign ihren Platz. Verifikations- und Validierungsprozesse sind rechtsseitig angeordnet. Das Hardware-Software-Interface wird in dieser Phase entwickelt. Anschließend wird die Entwicklung in den Hardware- und den Softwarepfad unterteilt, die jeweils wieder dem V-Modell folgen und später zusammengefügt werden. Die folgenden Phasen beschreiben im Wesentlichen Prozesse im Rahmen des Produktlebenszyklus. Formalisiert werden Prozesse bezüglich der Produktion, des Betriebs, des Services und der Stilllegung.

### 2.2.3 Fehlerbegriffe nach ISO 26262

Fehlertoleranz beschreibt die Fähigkeit eine definierte Funktion bei Vorliegen eines oder mehreren spezifischen Fehlern bereitzustellen [21]. Bei der Analyse von fehlertoleranten Ansätzen ist zunächst einmal zu klären, welche Arten von Fehlern existieren gegen die

eine Toleranz errichtet werden soll. Die ISO 26262 definiert für den automobilen Bereich eine ganze Bandbreite von Begriffen, die es bei der Systemmodellierung zu beachten gilt. Die Norm spezifiziert den Begriff „Fehler“ in „error“ (Abweichung), „failure“(Ausfall) und „fault“ (Mangel, Defekt). Als error wird die Abweichung zwischen einem berechneten, beobachteten oder gemessenen Wert und einem wahren, spezifischen oder theoretisch korrekten Wert bezeichnet. Aus einem error kann ein fault resultieren, welcher einen nicht normalen Zustand darstellt, der die Ursache eines Fehlverhalten eines Systems sein kann. Ein failure beschreibt den Ausfall eines Elements. Die beabsichtigte Funktion wird nicht mehr erfüllt.

### Abhängige Ausfälle

Abhängige Ausfälle sind Ausfälle, die statistisch nicht unabhängig sind, d.h. die Wahrscheinlichkeit des kombinierten Auftretens der Ausfälle ist ungleich dem Produkt der Eintrittswahrscheinlichkeiten aller unabhängigen Ausfälle. Die Definition von unabhängigen Ausfällen folgt dem gleichen Schema. Der abhängige Ausfall ist ein Sammelbegriff einiger Fehlerarten, die im Folgenden vorgestellt werden (vgl. Abb. 2.5).

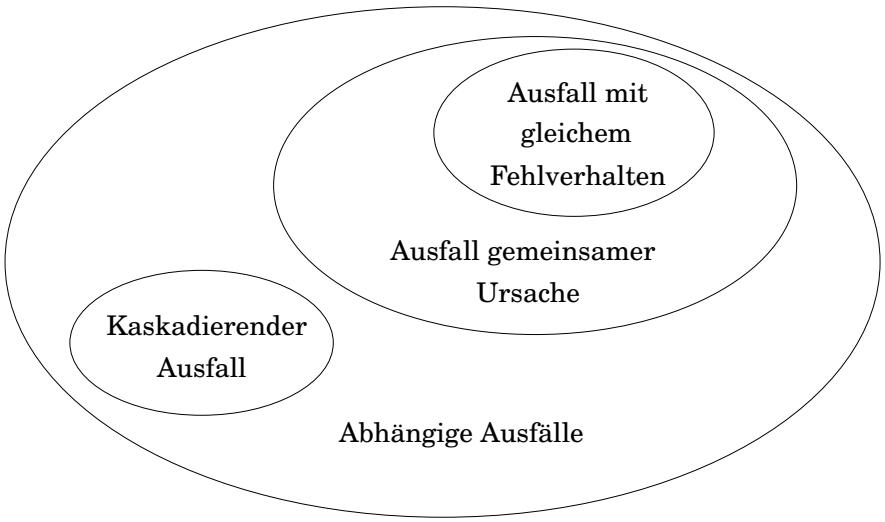


**Abb. 2.5:** Klassen abhängiger Ausfälle [20]

Abhängige Ausfälle bestehen beispielsweise aus zufälligen Hardwareausfällen aufgrund von gemeinsam genutzten Ressourcen oder Prüflogiken, Entwicklungsfehler aufgrund von Spezifikations-, Design- oder Implementierungsfehler sowie die Nutzung von Technologien, Produktionsfehler aufgrund des Produktionsprozesses, Anweisungen oder Ausbildung, Konstruktionsfehler, Reparaturfehler, Umgebungsfaktoren wie Temperatur, Vibration oder Druck, Verschleiß [20, 24].

### Ausfall gemeinsamer Ursache

Ausfälle gemeinsamer Ursachen werden in der ISO 26262 als Common Cause Failure (CCF) bezeichnet. CCF sind Ausfälle zweier oder mehrerer Elemente, die auf eine Ursache oder auf ein einzelnes spezifisches Event zurückzuführen sind, welche intern oder extern sein können. CCF sind abhängige Ausfälle, die keine kaskadierende Ausfälle sind [21].

### Ausfall mit gleichem Fehlverhalten

Ausfälle mit gleichem Fehlverhalten bzw. Common Mode Failure (CMF) sind eine Untermenge der CCF. Dabei handelt es sich um Elemente die in der gleichen Weise ausfallen bzw. das selbe Fehlverhalten zeigen. Das bedeutet jedoch nicht, dass die Ausfälle in der exakt selben Weise passieren müssen. Wie ähnlich das Fehlverhalten sein muss, um als CMF klassifiziert zu werden richtet sich nach dem Kontext [21]. Die ISO 26262 gibt zum Verständnis des CMF Begriffes folgende Beispiele: Wird bei zwei Temperatursensoren eine Abweichung von 5° festgestellt, handelt es sich um einen Mangel (fault). Ein CMF lässt die Temperatursensoren so ausfallen, dass die gemessene Abweichung weniger als 5° beträgt und der Fehler nicht erkannt wird. Die CPUs einer Locksteparchitektur müssen in exakt der selben Weise ausfallen, damit der Ausfall unbemerkt bleibt.

### Kaskadierender Ausfall

Ein kaskadierender Ausfall verursacht in seiner Folge weitere Ausfälle in anderen Elementen und ist kein CCF. Ein kaskadierender Ausfall kann die Ursache eines CCF sein. Fällt ein Teilsystem aus, werden die ausgefallenen Komponenten von anderen Teilsystemen kompensiert, was dazu führen kann, dass diese Komponenten ebenfalls ausfallen und eine Kettenreaktion ausgelöst wird. Kaskadierende Ausfälle treten häufig in Stromnetzen durch Überlastung ihrer Kapazität auf. Ein weiteres Beispiel ist das Versagen einer Hängebrücke, bei der eine einzelne Aufhängung ausfiel, wodurch die zusätzliche Last von den benachbarten Aufhängungen nicht getragen werden konnte.

### Einpunktausfälle

Führt ein einzelner Fehler oder eine einzelne Abweichung alleine zu einer Verletzung eines Sicherheitsziels, während kein Sicherheitsmechanismus vorliegt, dann handelt es sich um einen Einpunktausfall.

### Mehrpunktausfälle

Mehrpunktausfälle werden in der ISO 26262 als multiple-point failures bezeichnet. Sie beschreiben Ausfälle, die aus der Kombination mehrerer unabhängiger Hardwaremängel (faults) resultieren, die direkt zu einer Verletzung von Sicherheitszielen führen. Die

Kombination mehrerer unabhängiger Mängel wird multiple-point fault genannt. Zweipunktausfälle sind eine Untermenge der Mehrpunktausfälle.

#### Latente Fehler

Mehrpunktmängel (multiple-point faults), die von Sicherheitssystemen nicht erkannt oder vom Fahrer nicht wahrgenommen wurden. Während bei aktiven Fehlern die Fehlerfolgen sofort sichtbar werden, bleiben latente Fehler für längere Zeit unbekannt. Latente Fehler liegen häufig Entwicklungs- oder Reparatur- bzw. Wartungsfehlern zu Grunde. Ein Beispiel ist ein Programmierungsfehler für ein seltenes Ereignis.

#### Restfehler

In der Norm als „residual faults“ bezeichnete Restfehler sind zufällige Hardwarefehler auf Komponentenebene, die zur Verletzung eines Sicherheitsziels führen und durch kein Sicherheitskontrollsystem abgedeckt werden.

### 2.2.4 Dependent Failure Analysis

Ziel der Dependent Failure Analysis nach ISO 26262 ist es, eine ausreichende Unabhängigkeit sicherzustellen [24]. Eine vollständige Unabhängigkeit kann meist nicht erreicht werden [20]. Um dieses Ziel zu erreichen und die Auswirkungen abhängiger Ausfälle zu reduzieren, sind Sicherheitsmaßnahmen einzusetzen. Die Abwesenheit von abhängigen Ausfällen resultiert aus der Abwesenheit von kaskadierenden Ausfällen und Ausfällen gemeinsamer Ursachen (vgl. Abb. 2.5). Die Analyse soll die Unabhängigkeit von strukturell und diversitär Redundanten Elementen, von unterschiedlichen Funktionen die durch identische Hardware oder Software erbracht werden, von Funktionen inklusive ihrer Sicherheitsmaßnahmen, von Unterteilungen von Funktionen und Softwareelementen, von Hardware bezüglich ihrem räumlichen Abstand und von gemeinsam genutzten Ressourcen feststellen.

Eine Sicherheitsanalyse ist die Grundlage zur Bestimmung der Ursachen von potentiellen abhängigen Ausfällen. Für jede identifizierte Ursache ist die Plausibilität der Verletzung der Unabhängigkeit zwischen den betrachteten Elementen zu bewerten. Auf diese Weise sind individuelle Sicherheitsanforderungen und -ziele zu entwickeln. Mit Hilfe der Fehlerbaumanalyse und FMEAs können Informationen zu Abhängigkeiten gesammelt werden. Allerdings führt dieses Vorgehen dazu, dass viele Kaskaden unberücksichtigt bleiben [20]. Zur Unterstützung der Fehleranalysen hinsichtlich Fehlerkaskaden schlägt Ross die Verwendung von Simulationen basierend auf PSPICE vor [20]. Auf der Grundlage von injizierten Fehlern können systematische Fehlersimulationen Abhängigkeiten sichtbar gemacht werden. Weiter wird aufgeführt, dass das Erkennen von Kaskaden oft nur über langjährige Erfahrung möglich ist [20].

### 2.2.5 Quantifizierbare Anforderungen der ISO 26262

Die ISO 26262 Teil 5 stellt quantifizierbare Anforderungen an die Hardware Architektur auf der Ebene des Items hinsichtlich der Erkennung und Kontrolle von sicherheitsrelevanten Zufallsausfällen der Hardware. In Tabelle 2.1 sind die entsprechenden Grenzwerte unter Berücksichtigung der ASILs aufgelistet [25]. Diese Anforderungen gelten

**Tab. 2.1:** Mögliche Ableitung der zufälligen Hardwarefehler Zielwerte gemäß ISO 26262-5:2018 auf Item Ebene [25]

| ASIL | Zufällige Hardwarefehler Zielwerte |
|---|---|
| D | $< 10^{-8}\ h^{-1}$ |
| C | $< 10^{-7}\ h^{-1}$ |
| B | $< 10^{-7}\ h^{-1}$ |

für die Sicherheitsziele (saftey goals) des Items. Durch eine ASIL Dekomposition könnten die Werte auch noch reduziert werden, indem die Anforderung bezüglich eines ASILs in zwei Anforderungen eines geringeren ASILs aufgeteilt werden. Aufgrund der angestrebten Allgemeingültigkeit und Wiederverwendbarkeit des in dieser Arbeit entwickelten Ansatzes wird von einer Dekomposition jedoch abgesehen. Weiter benennt die Norm Anforderungen auf Komponentenebene. Tabelle 2.2 zeigt die Zielwerte der Ausfallraten für Einpunktfehler (single-point faults) [25]. Der Deckungsgrad von Diagnoseein-

**Tab. 2.2:** Ausfallratenzielwerte bezüglich Einpunktfehler gemäß ISO 26262-5:2018 auf Komponentenebene [25]

| ASIL des Sicherheitsziels | Ausfallratenklasse |
|---|---|
| D | Ausfallratenklasse 1 + gezielte Maßnahmen |
| C | Ausfallratenklasse 2 + gezielte Maßnahmen<br>oder<br>Ausfallratenklasse 1 |
| B | Ausfallratenklasse 2 |

richtungen unterliegt ebenfalls quantifizierten Anforderungen. Ein Zusammenhang von Diagnosedeckungsgrad und Restfehler, die nicht diagnostiziert werden ist in Tabelle 2.3 geggeben [25]. Um die Tabellen 2.2 und 2.3 lesen zu können, muss der Begriff der Ausfallratenklasse eingeführt werden, der ebenfalls in der Norm definiert ist [25].

- Fehlerraten der Klasse 1 müssen kleiner sein als der Zielwert für ASIL D (Tab. 2.1) geteilt durch 100.

**Tab. 2.3:** Fehlerratenklassen für gegebene Diagnosedeckungsgrade unter Berücksichtigung von Restfehlerngemäß ISO 26262-5:2018 auf Komponentenebene [25]

| **ASIL Sicherheitsziel** | **Diagnosedeckungsgrad bezüglich Restfehlern** | | | |
|---|---|---|---|---|
| | ≥ 99,9 % | ≥ 99 % | ≥ 90 % | < 90 % |
| D | Ausfallratenklasse 4 | Ausfallratenklasse 3 | Ausfallratenklasse 2 | Ausfallratenklasse 1 + gezielte Maßnahmen |
| C | Ausfallratenklasse 5 | Ausfallratenklasse 4 | Ausfallratenklasse 3 | Ausfallratenklasse 2 + gezielte Maßnahmen |
| B | Ausfallratenklasse 5 | Ausfallratenklasse 4 | Ausfallratenklasse 3 | Ausfallratenklasse 2 |

- Fehlerraten der Klasse 2 müssen kleiner oder gleich dem Zehnfachen der Ausfallrate der Klasse 1 sein.
- Fehlerraten der Klasse 3 müssen kleiner oder gleich dem 100-fachen der Ausfallrate der Klasse 1 sein.
- Fehlerraten der Klasse i müssen kleiner oder gleich dem $10^{i-1}$-fachen der Ausfallrate der Klasse 1 sein.

Die Ausfallratenklassen dienen der Einordnung der Fehlerhäufigkeiten.

## 2.3 Testkonzepte im Kontext der Automatisierungsstufen

Die Norm SAE J3016 definiert die Automatisierungsstufen für Kraftfahrzeuge im Straßenverkehr [26]. Neben der Stufe 0 existieren 5 Abstufungen des Automatisierungsgrades. Die Stufen 1 bis 3 sind durch die Rückfallebene des Fahrers charakterisiert. Für die letzten beiden Stufen entfällt diese Rückfallebene, was eine entscheidende Bedeutung für die erforderlichen Testkonzepte hat.

Testkonzepte für die Stufe 0, also Driver-Only Systeme ohne Automatisierung der Fahrfunktion, bestehen aus den bewährten Methoden des Sicherheitswissens. Die Freigabe erfolgt durch den Nachweis des Unterschreitens der Systemausfallwahrscheinlichkeit, die sich aus Einzelausfallwahrscheinlichkeiten der Komponenten zusammensetzt. Ein weit verbreitetes Verfahren für den quantitativen Nachweis hierbei ist die Fehlerbaumanalyse (FTA), welche im Abschnitt 3.2.1 genauer betrachtet wird. Außer-

dem muss sichergestellt werden, dass der Fahrer das Fahrzeug sicher im Straßenverkehr bewegen kann (Kontrollierbarkeit). Die Ergebnisse von Testfahrern werden auf zukünftige Nutzer übertragen [27]. Die Rückfallebene des Fahrers ist eine wichtige Konstante bei Testkonzepten dieser Automatisierungsstufe, die einschließlich bis Stufe 3 berücksichtigt wird.

Die Stufe 1 beschreibt assistierte Systeme, die entweder die Längsführung oder die Querführung übernehmen. Darunter fällt z.B. Adaptive-Cruise-Control (ACC) oder Spurhaltesysteme. Zusätzlich zu dem bestehenden Umfang werden Systeme ab Stufe 1 mit dem ADAS Code of Practice abgesichert [28]. Der ADAS Code of Practice geht davon aus, dass der Fahrer die Verantwortung für das Verhalten des Fahrzeugs behält. Die Kontrollierbarkeit und die Möglichkeit zur Übernahme müssen bei Systemen, die die Fahraufgabe unterstützen gewährleistet werden.

Teilautomatisierte Systeme, bestehend aus der Kombination von ACC und Spurhalteassistent entsprechen der Automatisierungsstufe 2 und wurden bereits für den Serieneinsatz freigegeben. Charakterisiert wird diese Stufe durch die Übernahme der Längs- und Querführung. Verantwortung des Fahrzeugverhaltens liegt auch in dieser Kategorie beim Fahrer. Daher steht auch hier die Kontrollierbarkeit und Möglichkeit zur Übernahme im Fokus und es gilt der gleiche Grundsatz wie zuvor: Zur Korrektur des Automatisierungsverhaltens wird auf die Fähigkeiten des Fahrers vertraut. Die besondere Herausforderungen bei dieser Stufe liegt im Konflikt zwischen der Entlastung des Fahrers und das notwendige Situationsbewusstsein des Überwachers für die Längs- und Querführung [27].

Die Stufe 3 soll dieses erforderliche Situationsbewusstsein minimieren, indem das System die Umgebungsbeobachtung übernimmt und der Fahrer nur nach Aufforderung zu reagieren hat. Echte Stufe 3 Systeme sind jedoch noch nicht eingeführt, da die derzeitige rechtliche Situation es erforderlich macht, die Hände am Lenkrad zu halten. Eine Novelle der Straßenverkehrsordnung, die sich in Arbeit befindet, wird dieses Problem lösen und den Weg für weitere Automatisierungsstufen frei machen.

Im Rahmen von dem elektronischen Stabilitätsprogramm und Notbremsassistenten konnten bereits Erfahrungen über die Herausforderungen der kommenden Stufen gesammelt werden. Diese Systeme stellen keine Stufe 4 Systeme dar. Trotzdem existiert eine Schnittstelle dieser Testkonzepte zu hochautomatisierten Systemen, denn die Rückfallebene des Fahrers existiert im betrachteten Funktionsbereich nicht. Sie setzen darauf den Kontrollverlust zu erkennen und die Kontrollierbarkeit wiederherzustellen. Beim Sicherheitsnachweis wird auf die Erhöhung der richtig-positiven Eingriffe und auf die Verringerung der falsch-positiven gesetzt [27].

## 2.4 Fehlertolerante Ansätze der Systemmodellierung für automatisierte Fahrfunktionen

### 2.4.1 Lockstepverfahren

Der Wegfall einer menschlichen Rückfallebene bei Fahrfunktionen der SAE Stufen vier und fünf macht fehlertolerante Ansätze der Systemmodellierung erforderlich. Für die Umsetzung von fehlertoleranten Systemarchitekturen werden in der Fachliteratur wiederholt Domain Electronic Control Units (ECUs) und Lockstep CPUs vorgeschlagen, die es erlauben die identische Operation unabhängig redundant auszuführen [12–14, 16]. CPUs im Lockstepverfahren bearbeiten mit der exakt selben Taktfrequenz den selben Input, um durch einen Vergleich den Output zu validieren. Im Dual-Core Lockstepbetrieb (vgl. Abb. 2.6) können Fehler detektiert werden, um darauf zu reagieren, z.B. durch eine Umschaltung in einen sicheren Zustand (fail safe).

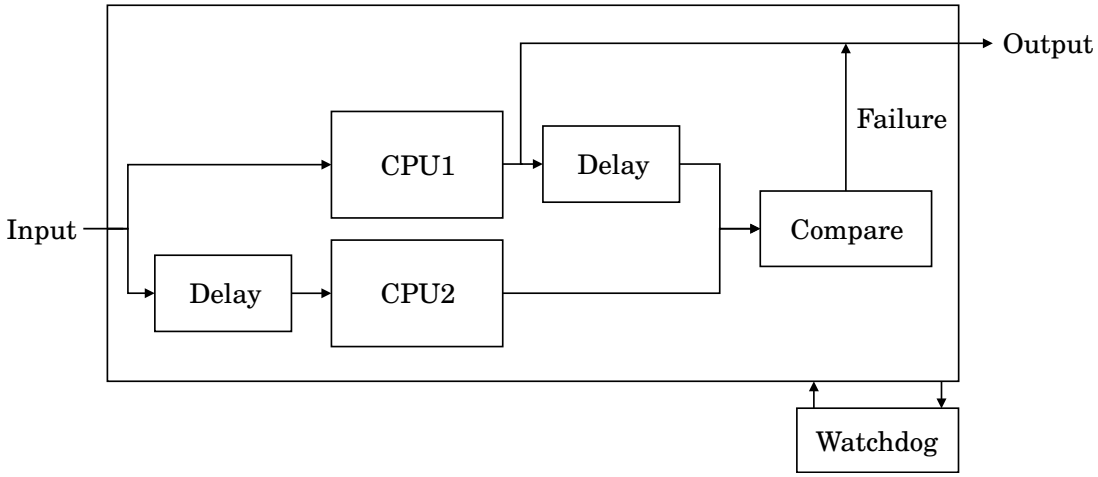


**Abb. 2.6:** Vereinfachte Dual-Core Lockstep Architektur

Führt der Vergleich der Ergebnisse der einzelnen CPUs zu einer Abweichung wird ein Fehler gemeldet. Das System kann in einen sicheren Zustand schalten. Eine Reduzierung der Anfälligkeit für Common-Cause- und Common-Mode-Fehler wird mit Verzögerungen der Rechenstränge und einem Watchdog bewirkt [29]. Die Verzögerung der CPUs verringert die Auswirkungen von Common-Cause-Fehlern, die durch Strom- oder Taktstörungen verursacht werden. Der externe Watchdog beinhaltet eine Ausfallerkennung für das System. Im vorliegenden Anwendungsfall werden über eine Timeout-Funktion Common-Mode-Fehler detektiert [29]. Komplexere Architekturen, die mit Hilfe von Mehrheitsentscheidern (MooN) umgesetzt werden, erlauben es über eine einfache Fehlererkennung hinauszugehen und Fehler ebenfalls zu tolerieren. Eine Tripel-Core Lockstep

Architektur im 2oo3 System kann bspw. zwei Fehler erkennung und einen tolerieren (fail operational).

### 2.4.2 Majoritätsredundanz

Die Majoritätsredundanz bzw. Mehrheitsentscheidungssysteme, kurz MooN Systeme, werden der aktiven Redundanz zugeordnet. Sie werden als Mittel zur Steigerung der Fehlertoleranz von Systemen, die ständig betriebsbereit sein müssen eingesetzt. Es gibt verschiedene Architekturen von MooN Systemen. Praktische Anwendungen finden sich in simplex, duplex, triplex und quadruplex Architekturen wieder. Einige Beispiele sind Pipelines (1oo1), Toröffnungsanlagen (1oo2), Gasturbinen (1oo2, 1oo4, 2oo2, 2oo3), zivile Luftfahrt (2oo3), Space Shuttle, Tornado, Atomkraftwerke (2oo4). Die Ergebnisse der MooN Systeme werden von einem Mehrheitsentscheider (Voter) verglichen, um das Ergebnis der Mehrheit weiterzugeben. Die Weitergabe des Ergebnisses erfolgt, wenn M der N Systeme funktionieren. Anderenfalls gilt das Gesamtsystem als ausgefallen. Die Überlebenswahrscheinlichkeit eines MooN Systems lässt sich bei gleicher Überlebenswahrscheinlichkeit $p$ der Komponenten mit Hilfe der Binominalverteilung berechnen (Gl. 2.1).

$$R_{MooN} = \sum_{k=m}^{n} \binom{n}{k} \cdot p^k \cdot (1-p)^{n-k} \tag{2.1}$$

Sind die Komponentenüberlebenswahrscheinlichkeiten ungleich erfolgt die Berechnung mit der Booleschen Modellbildung, welche für ein 2oo3 System in Gl. 2.2 dargestellt ist.

$$R_{2oo3} = p_1\ p_2 + p_1\ p_3 + p_2\ p_3 - 2\ p_1\ p_2\ p_3 \tag{2.2}$$

Basierend auf Gl. 2.1 sind nachfolgend einige Gleichungen verschiedener MooN Systeme unter Annahme einer Exponentialverteilung (konstante Ausfallrate) aufgeführt.

$$R_{1oo1} = e^{-\lambda t} \tag{2.3}$$

$$R_{1oo2} = 2\ e^{-\lambda t} - e^{-2\lambda t} \tag{2.4}$$

$$R_{2oo2} = e^{-2\lambda t} \tag{2.5}$$

$$R_{1oo3} = 3\ e^{-\lambda t} - 3\ e^{-2\lambda t} + e^{-3\lambda t} \tag{2.6}$$

$$R_{2oo3} = 3\ e^{-2\lambda t} - 2\ e^{-3\lambda t} \tag{2.7}$$

$$R_{3oo3} = e^{-3\lambda t} \tag{2.8}$$

Abbildung 2.7 zeigt für $\lambda = 10^{-3}\ 1/h$ die Überlebenswahrscheinlichkeiten der vorgestellten MooN Systeme über die Zeit.

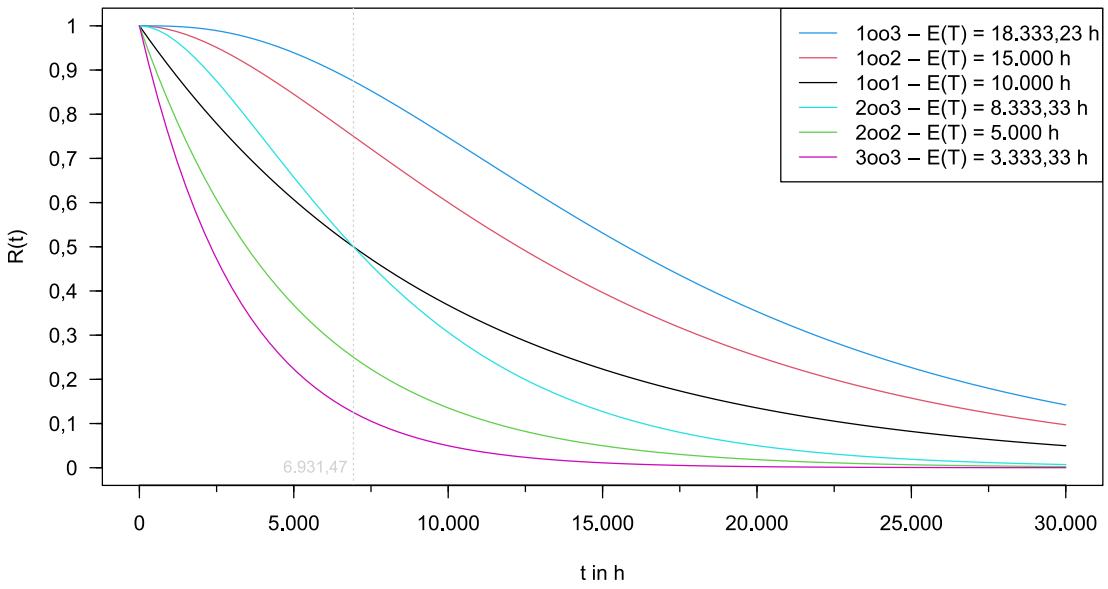


**Abb. 2.7:** Überlebenswahrscheinlichkeiten verschiedener einfacher MooN Systeme mit der konstanten Ausfallrate $\lambda = 10^{-3}\ 1/h$

Die NooN Systeme sind klassische Serienanordnungen. In Abb. 2.7 werden diese durch die schwarze, grüne und violette Kurve verbildlicht und zeigen, dass je mehr Komponenten ein Seriensystem besitzt, desto geringer fällt die Überlebenswahrscheinlichkeit aus. Die schwarze Kurve stellt gleichzeitig eine Referenz dar. Sie beschreibt eine einzelne Komponente, dessen Zuverlässigkeit gesteigert werden soll. Die rote und die dunkelblaue Kurve liegen in der gesamten Lebensdauer über der Referenz. Als 1ooN Systeme beschreiben sie Parallelanordnungen. Auch hier ist bekannt, dass die Überlebenswahrscheinlichkeit eines Parallelsystems sich mit jeder weiteren Komponente erhöht. Dabei ist jedoch zu berücksichtigen, dass heiße Redundanz unwirtschaftlich ist, was v.a. in komplexen Systemen an Bedeutung gewinnt.

Der Verlauf der hellblauen Kurve ist gesondert zu betrachten. Als 2oo3 System beschreibt sie eine Majoritätsredundanz abseits der bekannten Muster. Im ersten Abschnitt der Lebensdauer liegt die Kurve über der Referenz und fällt im weiteren Verlauf darunter. Der Erwartungswert des 2oo3 Systems liegt mit $8.333,33\ h$ unter dem der Referenz von $10^3\ h$. Im Folgenden werden weitere MooN Systeme analysiert, um dieses Verhalten zu untersuchen. Abb. 2.8 zeigt bis zu fünfgliedrige Majoritätsredundanzen mit M $\neq$ N mit der konstanten Ausfallrate $\lambda = 10^{-3}\ 1/h$ sowie die 1oo1 Referenz.

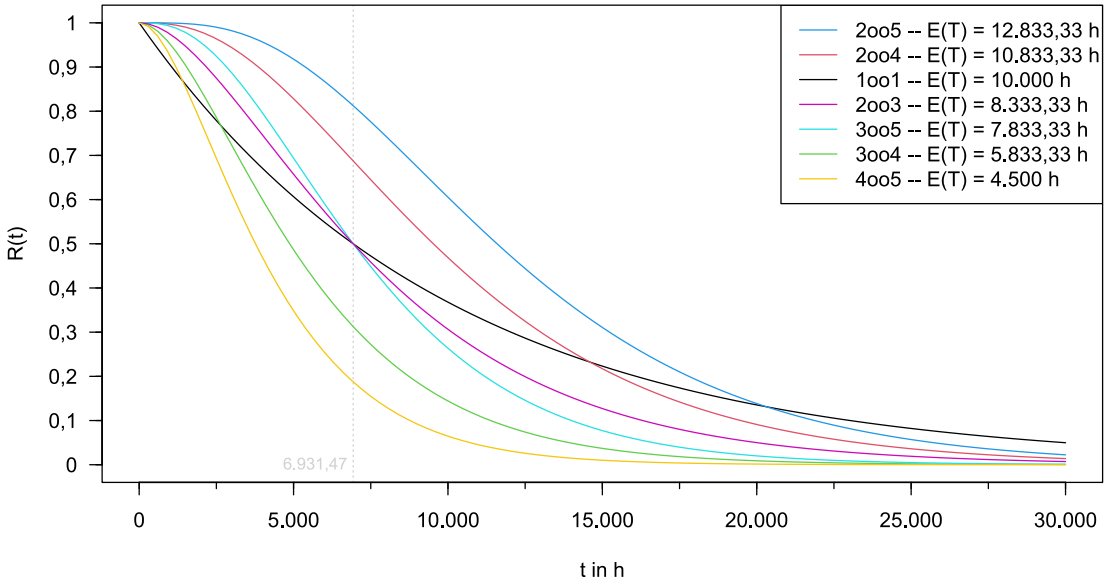


**Abb. 2.8:** Überlebenswahrscheinlichkeiten verschiedener MooN Systeme mit der konstanten Ausfallrate $\lambda = 10^{-3}\ 1/h$

Neben des Schnittpunkts des 2oo3 Systems kann noch ein weiterer Schnittpunkt identifiziert werden. Das 3oo5 System schneidet die Referenz genau an der selben Stelle bei $6.931,47\ h$. Zwei weitere Schnittpunkt im späteren Verlauf spielen eine geringere Rolle, da sie sich deutlich hinter dem Erwartungswert der Referenz von $10^3\ h$ befinden. Es kann beobachtet werden, dass das 3oo4 und das 4oo5 System unter den betrachteten Systemen den geringsten Erwartungswert der Überlebenswahrscheinlichkeit aufweist. Das 2oo5 und 2oo4 System liefern die höchsten Werte und liegen über den Großteil der betrachteten Zeitspanne über der Referenz. Abb. 2.8 bestätigt die in Abb-2.7 gemachte Beobachtung: Für konstante Ausfallraten liefern Majoritätsredundanzen nicht immer bessere Zuverlässigkeitskennwerte als eine einzelne Komponente (1oo1). Des Weiteren kann eine im Vergleich zur Referenz erhöhte Lebensdauer nur in frühen Phasen des konstanten Ausfallverhaltens festgestellt werden. Eine über den Erwartungswert der Referenz hinausgehende Erhöhung der Lebensdauer kann nur bei zwei komplexeren Systemanordnungen gesehen werden. Der Grad der Komplexität liefert in diesem Fall jedoch keinen Hinweis auf die Zuverlässigkeit. Die beiden Systeme mit der größten Komplexität liefern die geringsten Werte.

Die Erkenntnisse werden nun auch für nicht konstante Ausfallraten untersucht. Zur Modellierung wird eine zweiparametrige Weibullverteilung herangezogen (Gl. 2.9).

$$R(t) = \begin{cases} e^{-\left(\frac{t}{\eta}\right)^{\beta}} & \text{für} \quad t \geq 0 \\ 1 & \text{für} \quad t < 0 \end{cases} \tag{2.9}$$

Der Parameter $\beta$ ist ein Maß für die Ausfallsteilheit, die Werte zwischen $0,25 \leq \beta \leq 5$ annehmen kann. Die charakteristische Lebensdauer $\eta$ beschreibt die Zeit bei der 63,2 % der Komponenten ausgefallen sind. Mit $\eta = 10^3$ und $\beta = 1$ werden die Abbildungen 2.7 und 2.8 reproduziert. Dabei entspricht $\beta = 1$ der konstanten Ausfallrate. Im Bereich von $0 < \beta < 1$ nimmt die Ausfallrate mit wachsendem t monoton ab (Frühausfälle). Verschleißausfälle werden durch $\beta > 1$ charakterisiert. $\beta$ Werte $> 3$ beschreiben näherungsweise eine Normalverteilung. Für $\beta > 5$ wird die Verteilung rechtsschief.

Abb. 2.9 zeigt das Frühausfallverhalten der vorgestellten Majoritätsredundanzen mit variablen $\beta$ bei $\eta = 10^3$ $h$. Erneut kann ein Schnittpunkt zwischen dem 1oo1, dem 2oo3 und dem 3oo5 System identifiziert werden, der bei kleinen $\beta$ weiter vorne liegt. Bei größer werdenden $\beta$ verlagert sich der Schnittpunkt kontinuierlich in die hinteren Bereiche der Lebensdauer. Die Anordnung der Kurven des Frühausfallverhaltens ähnelt der Anordnung der Kurven des konstanten Ausfallverhaltens. 2oo5 und 2oo4 liegen über der Referenz, während 3oo4 und 4oo5 darunter liegen. Ein Unterschied kann jedoch in den Erwartungswerten festgestellt werden. In den Legenden sind die MooN Systeme nach ihren Erwartungswerten von groß nach klein sortiert. Die Unterschiede liegen in der Position der 1oo1 Referenz. Der größte Ausreißer ist bei $\beta = 0,25$ zu erkennen. Mit einem Erwartungswert von 240.000 h liegt das 1oo1 System deutlich über den anderen. Diese Größenordnung ist bei keinem der folgenden Betrachtungen wiederzufinden. Vermutlich strebt die 1oo1 Kurver signifikant langsamer gegen null als die übrigen Kurven. Da es sich hier um ein Frühausfallverhalten handelt erhält dieser Umstand keine weitere Bedeutung, da bevor diese späten Phasen der Lebensdauer erreicht werden, das Frühausfallverhalten in ein konstantes und dann in ein Verschleißausfallverhalten übergehen wird (vgl. Badewannenkurve). Der genaue Zeitpunkt dieser Phasenwechsel ist jedoch unbekannt und muss empirisch untersucht werden. Für $\beta = 0,5$ rutscht der Erwartungswert des 1oo1 Systems an Position zwei und für $\beta = 0,75$ an Position drei.

Das Verschleißausfallverhalten, dargestellt in Abb. 2.10 folgt einem ähnlichen Schema. Die Anordnungen der Kurven entspricht dem bekannten Bild. Bei $\beta = 5$ erzielt die 1oo1 Referenz nur den fünften Rang und das 3oo5 System liegt erstmal über dem 2oo3 System. Der untersuchte Schnittpunkt wandert auf der Lebensdauerachse weiter nach rechts. Während der Schnittpunkt beim Frühausfallverhalten noch in frühen Phasen der Lebensdauer liegt, liegt er beim Verschleißausfallverhalten weiter hinten in der nähe der Erwartungswerte.

Zusammengefasst kann gesagt werden, dass die Rangfolge der verschiedenen Architekturen von den Werten der Parameter und den betrachteten Zeitpunkt abhängt. Das 1oo1 System ist für kleine $\beta$ im Kontext der Erwartungswerte zuverlässiger. Für große $\beta$ sind komplexere Architekturen (2oo5, 2oo4) zuverlässiger. Zwischen den Anordnungen der Kurven bezüglich ihrer Überlebenswahrscheinlichkeit kann kein

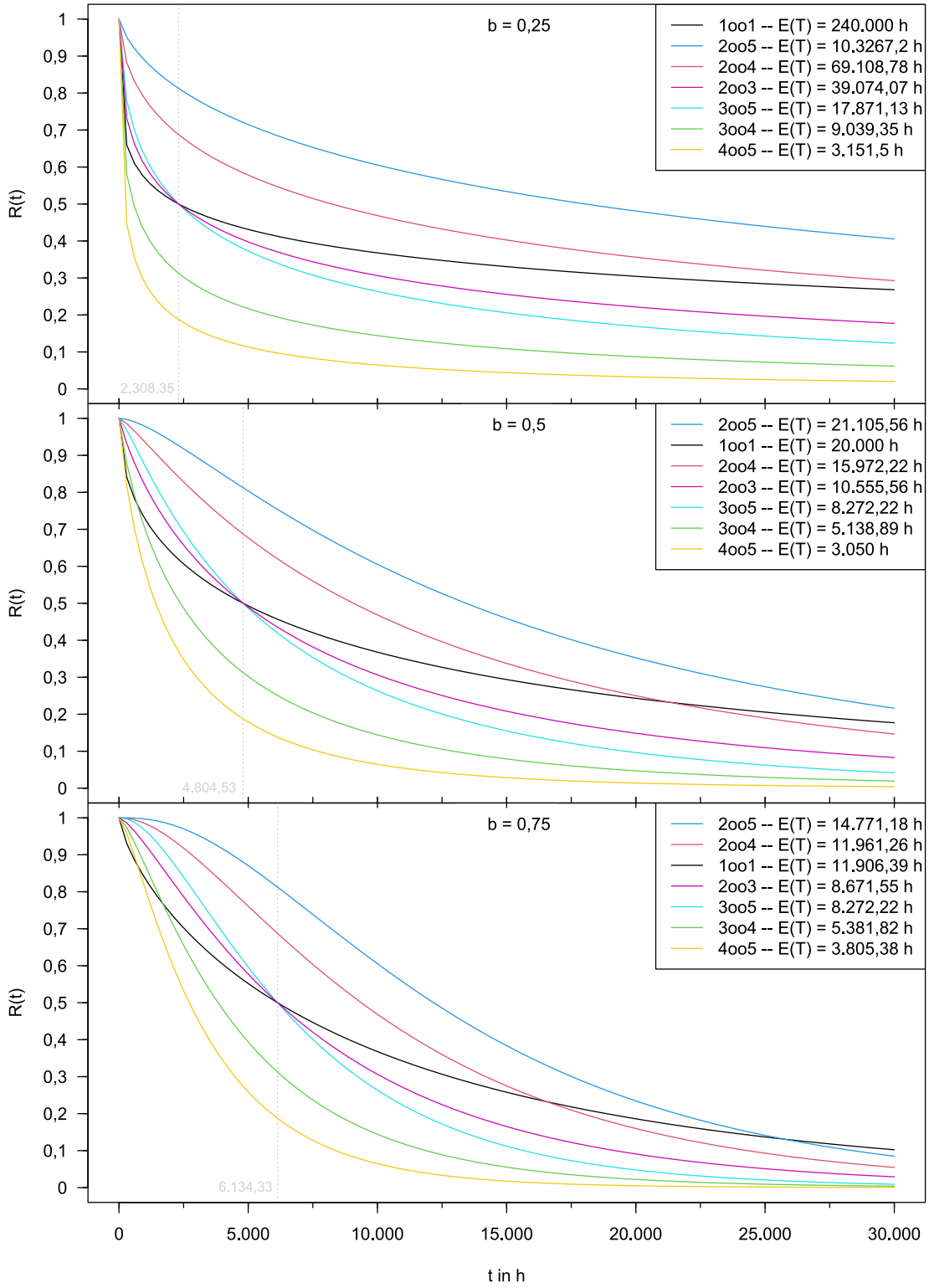


**Abb. 2.9:** Frühausfallverhalten verschiedener MooN Systeme mit variabler Ausfallsteilheit $\beta$ bei einer charakteristischen Lebensdauer von $\eta = 10^3\ h$

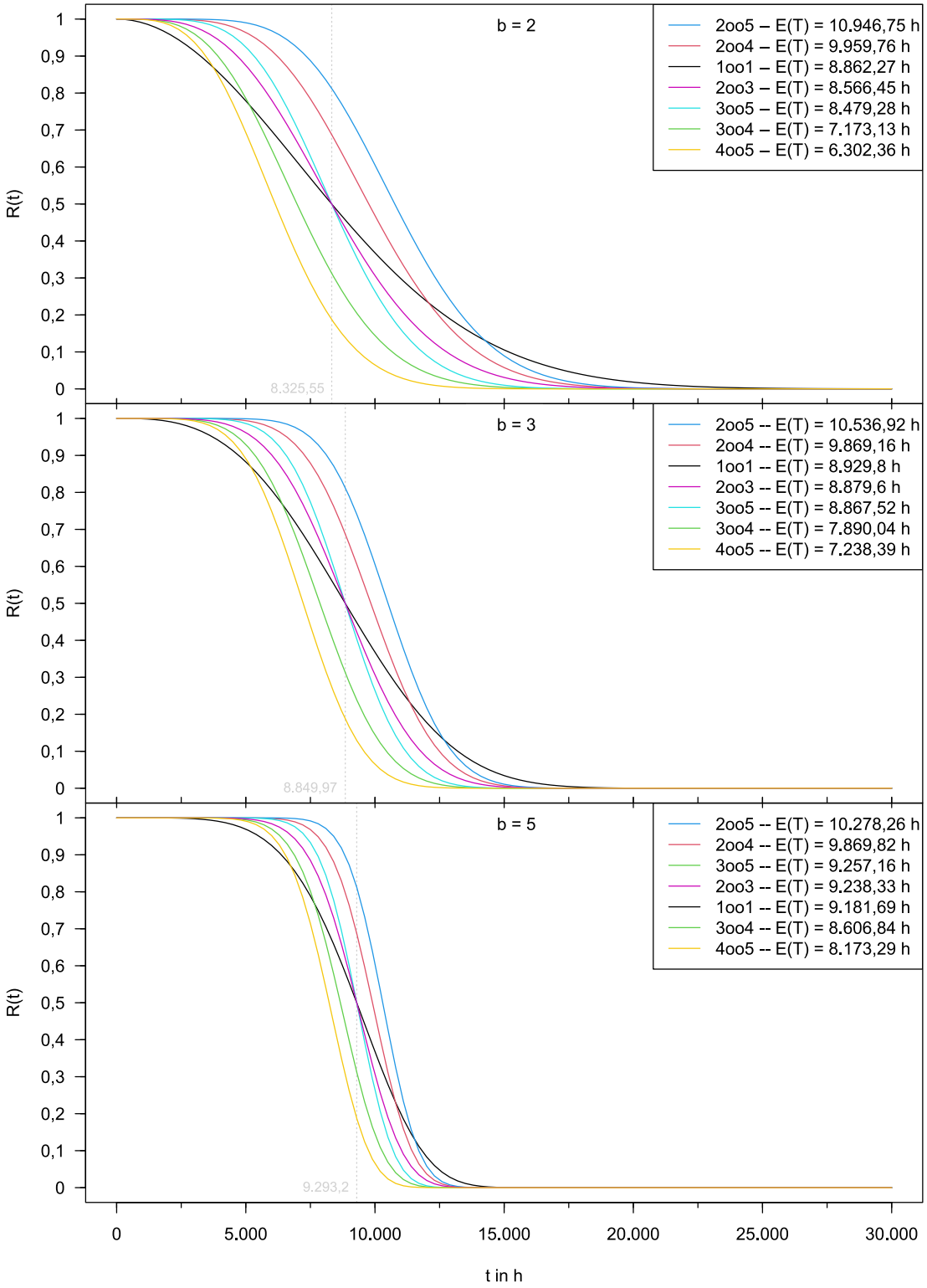


**Abb. 2.10:** Verschleißausfallverhalten verschiedener MooN Systeme mit variabler Ausfallsteilheit $\beta$ bei einer charakteristischen Lebensdauer von $\eta = 10^3\ h$

Unterschied festgestellt werden. Die Anordnungen bzw. die Rangfolgen als Funktion der Erwartungswerte sind jedoch unterschiedlich. Sie hängt insbesondere von der Ausfallrate ab. Zu berücksichtigen ist jedoch auch, dass es bei Majoritätsredundanzen nicht nur um eine Erhöhung der Zuverlässigkeit geht. Im Sinne der Fehlertoleranz ermöglichen sie Fehlerdiagnostik. Duplexsysteme (2oo2) ermöglichen das Erkennen eines Fehlers, wodurch ein sicheres Abschalten ermöglicht wird. Triplexsysteme (2oo3) können zwei Fehler erkennen und einen tolerieren (Fail-Operational), während Quadruplexsysteme (2oo4) drei Fehler erkennen und zwei tolerieren können. Im realen Anwendungsfall muss also auch die Fähigkeit zur Fehlerdiagnostik berücksichtigt werden. Eine Auswahl der geeigneten Architektur kann nicht alleine anhand der Zuverlässigkeitskennwerte erfolgen.

### 2.4.3 Etablierte Hardwarearchitekturen

Durch den Wegfall der menschlichen Rückfallebene werden fehlertolerante Ansätze der Systemmodellierung erforderlich, die durch einfache Redundanz alleine nicht umgesetzt werden können. Die Prinzipien der der Fehlertoleranz beruhen auf Selbstdiagnose, Zuverlässigkeit, Verfügbarkeit, Rekonstruktion und Fehlerbehebung. Die Zuverlässigkeit beschreibt die Fehlerfreiheit eines Systems über die Zeit. Sie wird in der Regel durch strukturelle Redundanz erhöht. Die Verfügbarkeit gibt an, ob ein System zu einem bestimmten Zeitpunkt funktioniert und kann durch diversitäre Redundanz oder Separation bzw. Unabhängigkeit beeinflusst werden, wobei auch von asymmetrischen Systemarchitekturen gesprochen wird. In traditionellen sicherheitskritischen Systemen, die u.a. in der Luftfahrt, im Schienenverkehr, in der Raumfahrt, im Militär oder in Atomkraftwerken zu finden sind, werden diese Prinzipien bereits angewendet. Aber auch in der automobilen Fachliteratur gewinnen sie zunehmend an Bedeutung, z.B. [11–14, 16]. Domain ECUs (Electronic Control Units) mit Lockstep CPUs erlauben es dieselbe Operationen mit einer ausreichenden Unabhängigkeit redundant auszuführen. CPUs im Lockstepverfahren bearbeiten mit der exakt selben Taktfrequenz den selben Input, um durch einen Vergleich den Output zu validieren. In der einfachen Dual-Core Lockstep Architektur wird derselbe Input von einem Main Core und einem Checker Core berechnet. Solange der Checker Core keine Abweichung feststellt, übermittelt der Main Core das korrekte Ergebnis. Wird jedoch eine Abweichung erkannt schaltet das System in einen sicheren Zustand (fail safe). Dabei werden die Taktzyklen der CPUs verzögert, um Common Cause Fehlern entgegenzuwirken. Komplexere Architekturen ermöglichen nicht nur strukturelle Redundanz, sondern ermöglichen auch Fehlertoleranz. Zwei etablierte Lockstep Architekturen sind in Abbildung 2.11 und 2.12 dargestellt.

Das 2-out-of-2 System aus Abb. 2.11 besitzt zwei unabhängige Pfade, die jeweils mit einer Dual-Core Lockstepparchitektur betrieben werden. Die beiden Pfade kommu-

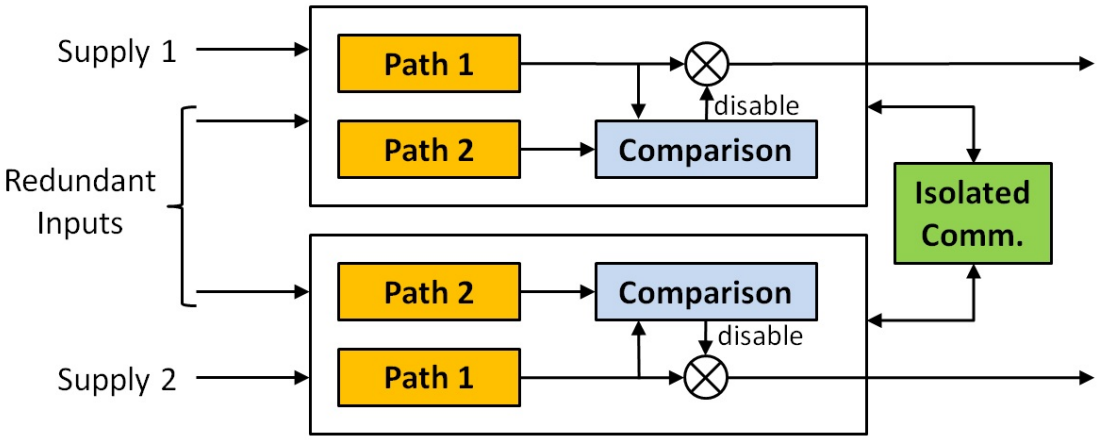


**Abb. 2.11:** 2oo2DFS autonomous vehicle architecture [12]

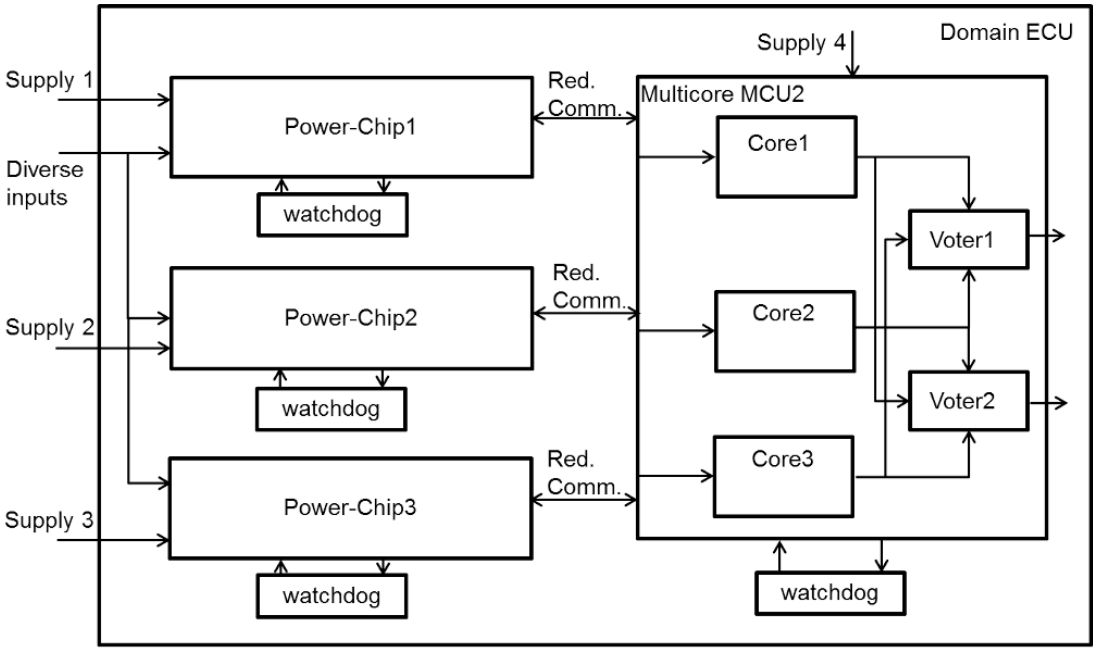


**Abb. 2.12:** 2oo3 Fail-Operational Systemarchitektur [16]

nizieren miteinander, um im Fehlerfall den Betrieb zu übernehmen. Es kann also ein Fehler diagnostiziert und toleriert werden. Befindet sich das System jedoch im einzelsträngigen Betrieb, existiert keine Fehlertoleranz mehr solange nicht beide Pfade wieder betriebsbereit sind. In Abb. 2.12 sind befinden sich drei CPUs im Lockstep Betrieb. Durch eine 2oo3 Majoritätsredundanz (triplex) können zwei Fehler erkannt und ein Fehler toleriert werden. Zusätzlich sind Watchdogs verbaut, die den Zustand der einzelnen Komponenten überwachen. Welche Architektur gewählt wird ist eine Abwägung zwischen Kosten, Leistung und Ausfallsicherheit. Die 2oo2DFS Architektur hat bspw. aufgrund des zusätzlichen Prozessors einen höheren Hardware-Overhead[1], kann aber Diversitätsanforderungen besser erfüllen, indem die beiden Stränge separiert oder durch unterschiedliche Hardware betrieben werden. Zusätzlich wird durch die Kommunikation zwischen den Strängen die Umschaltung in einen degenerierten Modus ermöglicht. Diese zusätzlichen Funktionen hat die Triplexarchitektur nicht. Eine Verbesserung der Rechenleistung im Vergleich zum Single-Core Betrieb können beide Architekturen nicht aufweisen [11].

Bei den bisher vorgestellten Architekturkonzepten handelt es sich natürlich nur um einen Ausschnitt. Eine Berücksichtigung von der Komponentenstruktur konnte damit noch nicht erbracht werden. Eine detaillierte Beschreibung von Komponenten wurde in Abbildung 2.13 von Sari vorgenommen [16]. Es handelt sich um eine Fallback Architektur, die in der Literatur immer mehr behandelt werden. Für diese Arbeit spielt dieses Konzept jedoch erstmal keine Rolle. Der Beobachtungsgegenstand der Abbildung 2.13 beinhaltet die Wahl der Komponenten. Mit der Benennung von Kamera, Radar und Lidar ist die Sensorik unmissverständlich beschrieben. Ebenso wird ein größeres Augenmerk auf die Modellierung des Datenverarbeitungsstrang gelegt. Monitoring, Data aquisition und Processing werden eigene Komponenten zugeordnet. Dem Prozess der Mehrheitsentscheidung werden ebenfalls Komponenten zugeordnet. Diese Komponentenstruktur wird im späteren Verlauf der Arbeit weiter verwendet.

Des Weiteren schlägt Kohn et al. in Abbildung 2.14 eine weitere Variante einer 2oo3 Architektur vor, die ebenfalls später wieder verwendet wird [12] Die Mehrheitsentscheidung erfolgt hierbei über zwei Stufen, um den Nachteil der fehlenden Diversität auszugleichen. Die drei Verarbeitungsstränge speisen unabhängig voneinander drei verschiedenen 2oo3 Mehrheitsentscheider, die wiederum einen weiteren Mehrheitsentscheider speisen, der den Systemoutput an die Aktorik weiterleitet.

[1] Daten, die nicht zu den Nutzdaten zählen, sondern als Zusatzinformation zur Übermittlung oder Speicherung benötigt werden.

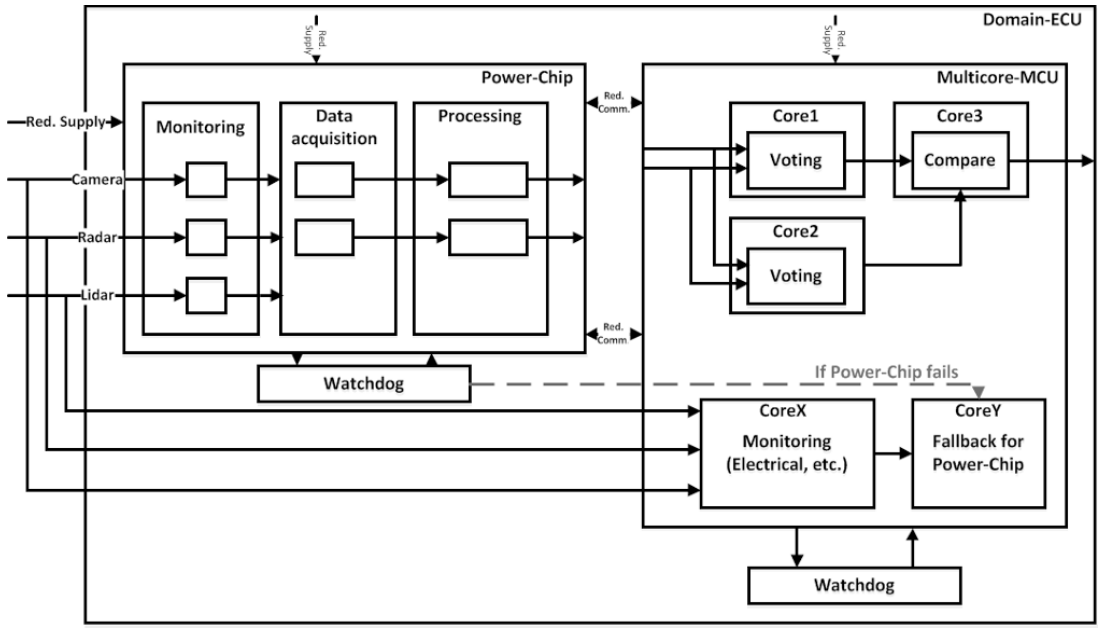


**Abb. 2.13:** Fail-Operational Systemarchitektur bis zum SAE Level 3 [16]

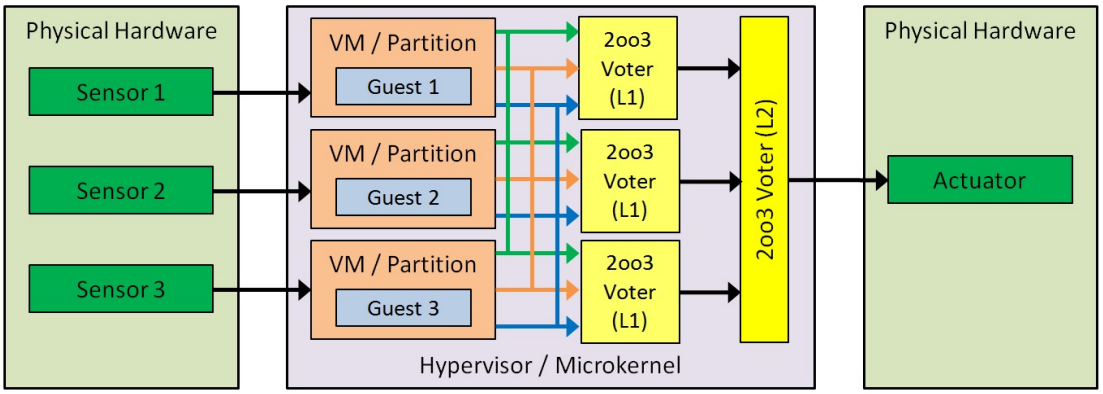


**Abb. 2.14:** Zweistufige 2oo3 Hardwarearchitektur nach Kohn et al. [12]

### Sensorarchitektur

Für alle Systemarchitekturen spielt der Input eine erhebliche Rolle, der im betrachteten Anwendungsfall im wesentlichen aus der Wahl der Sensorik liegt, die im Folgenden näher betrachtet wird. Im Allgemeinen ist die Literatur sich über die Wahl der richtigen Sensorarchitektur uneinig. Die bewährten Sensoren, die für automatisierte Fahrfunktionen zur Verfügung stehen belaufen sich auf Kamera, Radar, Lidar und Car2X Kommunikation. Eine Vielzahl von Konzepten sieht die Verwendung von Kamera, Radar und Lidar vor, wie auch der vorgestellte Ansatz von Sari (Abb. 2.13). Die 2oo2 Architektur von Kohn et al. (Abb. 2.14) sieht ebenfalls drei Sensoren vor, ohne diese genauer zu spezifzieren. Die Architekturen aus Abbildung 2.11 und 2.12 lassen die Wahl der Sensorarchitektur komplett offen. Die häufige Wahl dieser dreigliedrigen Architektur, bestehend aus Kamera, Radar und Lidar, beruht auf Überlegungen, die in [30] und [31] angestellt wurden. Es wurde ermittelt, welche Sensoren für welche Umgebungsbedingungen geeignet sind und welche Kombinationen notwendig sind, um eine ausreichende Sensorleistung zu erhalten (Tab. 2.4).

**Tab. 2.4:** Bewertung der Sensorperformance bezüglich verschiedener Fahraufgaben [30, 31]

| Perormance aspect | Human | Automated Vehicle (AV) | | | Connected vehicle (CV) | CAV AV+CV |
|---|---|---|---|---|---|---|
| | Eyes | Camera | Radar | Lidar | DSRC | |
| Object detection | Good | Fair | Good | Good | n/a | Good |
| Object classication | Good | Good | Poor | Fair | n/a | Good |
| Distance estimation | Fair | Fair | Good | Good | Good | Good |
| Edge detection | Good | Good | Poor | Good | n/a | Good |
| Lane tracking | Good | Good | Poor | Poor | n/a | Good |
| Visibility range | Good | Fair | Good | Fair | Good | Good |
| Poor weather | Fair | Poor | Good | Fair | Good | Good |
| Darkness | Poor | Fair | Good | Good | n/a | Good |
| Communication ability | Poor | n/a | n/a | n/a | Good | Good |

Bewertet wurde die Leistung verschiedener Sensoren für unterschiedliche Fahraufgaben. Die Klassifizierung erfolgt in Good, fair und poor. Zu sehen ist, dass der Einsatz von Kamera, Radar und Lidar erforderlich ist, um alle betrachteten Fahrsituationen gut bewältigen zu können. Daneben wurde die Notwendigkeit der Kommunikation zwischen den Fahrzeugen herausgestellt, die für diese Arbeit jedoch eine untergeordnete Rolle spielt. Hierbei handelt es sich jedoch nicht um eine allgemein anerkannte technische Regel, denn es beginnen immer mehr Konzepte auf einzelne Sensoren zu ver-

zichten. Tesla startete mit dem Verzicht von Lidar Sensoren und möchte nun auch den Radar einsparen, was als Vision-Only Ansatz bezeichnet wird [32]. Intel betreibt das Produkt Mobileye, welches auf einem adaptiven Ansatz beruht und auf den Radar verzichtet [33]. Grundlage ist die Kritik an der oftmals nicht redundanten Auslegung von Sensoren. Intels System kann autark durch Kameras oder Lidar betrieben werden. Der Normalzustand ist der kombinierte Betrieb. Im Fehlerfall wird kann auf einen Pfad verzichtet werden. Aus dem Projekt SafeAdapt (Safe Adaptive Software for Fully Electric Vehicles) ist eine für höhere Automatisierungsstufen taugliche Architektur hervorgegangen, die das Konzept von Intels Mobileye umsetzen kann [34]. Weiss et al. haben die Notwendigkeit von mindestens zwei Kommunikationspfaden erörtert, die unabhängige ECUs versorgen [35]. Diese ECUs können die Informationen der Mobileye Sensoren jeweils unabhängig verarbeiten. Es wurde eine Hybridlösung bestehend aus heißer und kalter Redundanz entwickelt (Abb. 2.15).

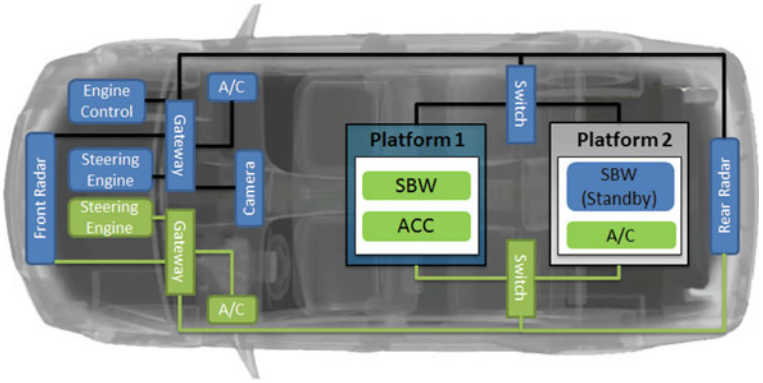


**Abb. 2.15:** Dynamische Rekonfiguration unabhängiger ECUs mittels heißer und kalter Redundanz [35]

Es handelt sich um eine Variante einer 2oo2 Architektur, die rekonfiguriert wird, sobald ein Fehler auftritt. Bei der Rekonfiguration der kalten Redundanz spielt die Rekonfigurationszeit eine Rolle, die zur Verfügung steht, um die Gefahrensituation rechtzeitig abzuwenden. Oszwald et al. haben die zur Verfügung stehende Rekonfigurationszeit nach dem Erkennen eines Fehlers bis zur Überführung des Systems in einen sicheren Zustand mit bis zu 200 ms geschätzt [36]. Unter Berücksichtigung verschiedener Kollisionsvermeidungsszenarien wurde eine präzise obere Grenze von 130 ms angegeben. Zu beachten ist jedoch, dass diese Ergebnisse auf Simulationen basieren und je nach Architektur abweichen können. Weiss et al. haben ermittelt, dass ihre hybride Architektur eine Kostenersparnis von 25 %, eine Gewichtsreduzierung von 16 %, eine Platzeinsparung von 12 % und eine Energieeinsparung von 22 % erbringt [35]. Ungeklärt ist jedoch wie die Standby-Komponenten sich im 2oo2-System auf die Zuverlässigkeit auswirken.

# 3 Werkzeuge zur Modellierung des Ausfallverhaltens von Systemen

## 3.1 Einordnung der Fehlerbaum- und Markovanalyse

**Fehlerbaumanalyse**

Für die zuverlässigkeitstechnische Analyse des Ausfallverhaltens von komplexen Systemen eignen sich Fehlerbaumdarstellungen. Basisereignisse (hier Komponentenzustände) werden mittels logischen Gattern verknüpft, um das Top-Ereignis, den Systemausfall zu analysieren. Fehlerbäume eigenen sich für die Generierung von Informationen für die Systemauslegung hinsichtlich Einsatzprofil, Architektur, Funktionsweise, Fehlerbeherrschung, etc. [37]. Fehlerbäume eignen sich besonders dann, wenn Kombinationen von Basisereignissen relevant sind. Werden Komponenten separat betrachtet ohne Berücksichtigung dessen Kombinationen sollte auf eine FMEA zurückgegriffen werden. Gegenüber anderen Methoden (z.B. Markov oder RBD) bietet die FTA den Vorteil Kausalitäten zu betrachten. Außerdem ist eine tabellarische Auswertung bezüglich Minimalschnitten, Importanzen und Sensitivitäten möglich.

**Markov-Ketten**

Markov-Ketten beschreiben Systemzustände aus einem endlichen Zustandsraum. Die Übergänge zwischen den Zuständen werden mittels Zustandsübergangswahrscheinlichkeiten untersucht. Die Berechnungen dieser Zustände können mit verschiedenen Verfahren erfolgen. Drei dieser Verfahren werden in Abschnitt 3.2.2 vorgestellt. Markovanalysen eignen sich besonders zur Untersuchung von sequentiellen Abfolgen von Ereignisse, beispielsweise bei der Betrachtung von Betriebsmodi, die an Ereignisse geknüpft eingenommen werden. Eine Kombination von Fehlerbäumen und Markovketten kann eingesetzt werden, um stand-by Redundanzen zu modellieren. Markov-Ketten sind außerdem eine Möglichkeit Instandsetzungsprobleme zu bearbeiten. Softwarelösungen für FTAs bieten ähnliche Funktionen.

## 3.2 Beispiele für analytische Lösungen

### 3.2.1 Fehlerbaumanalyse

#### 3.2.1.1 Seriensystem

Abbildung 3.1 zeigt das Blockschaltbild und den zugehörigen Fehlerbaum eines Seriensystems bestehend aus zwei Komponenten. Seriensysteme werden im Fehlerbaum durch ein Und-Gatter modelliert.

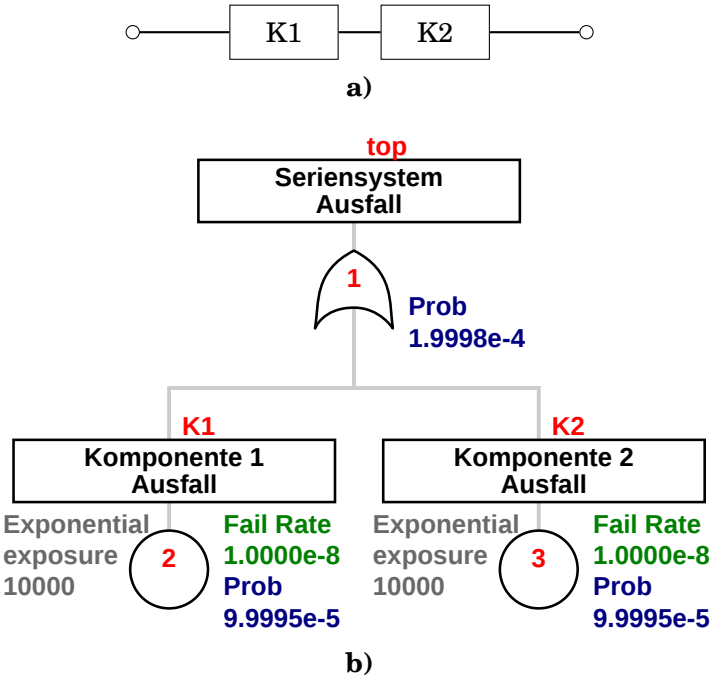


**Abb. 3.1:** Blockschaltbild a) und Fehlerbaum b) eines einfachen Seriensystems mit zwei Komponenten

Die Überlebenswahrscheinlichkeit eines Seriensystems wird durch Gleichung 3.1 beschrieben. Dabei stellt $p_i(t)$ die Überlebenswahrscheinlichkeit der Komponente $i$ in Abhängigkeit von der Zeit dar.

$$R(t) = \prod_{i=1}^{n} p_i(t) \tag{3.1}$$

Die Ausfallwahrscheinlichkeit ist die Gegenwahrscheinlichkeit zur Überlebenswahrscheinlichkeit und wird durch Gleichung 3.2 berechnet.

$$F(t) = 1 - R(t) = 1 - \prod_{i=1}^{n} p_i(t) = 1 - \prod_{i=1}^{n} (1 - q_i(t)) \tag{3.2}$$

Dabei steht $p_i(t)$ erneut für die Überlebenswahrscheinlichkeit der Komponente $i$. $q_i(t)$ beschreibt die Ausfallwahrscheinlichkeit der Komponente $i$. Die Exponentialverteilung wird in der technischen Zuverlässigkeit oft eingesetzt um Berechnungen zu vereinfachen. Sie erhält außerdem eine besondere Bedeutung, da die ISO 26262 das Exponentialmodell zu Grunde legt. Die Ausfallwahrscheinlichkeit einer exponentialverteilten Größe wird durch Gleichung 3.3 beschrieben.

$$F(t) = 1 - e^{-\lambda t} \tag{3.3}$$

Wird das Exponentialmodell auf die Gleichung zur Berrechnung des Seriensystems angewendet, ergibt sich folgendes.

$$F(t) = 1 - \prod_{i=1}^{2}(1 - (1 - e^{-\lambda t})) \tag{3.4}$$

$$= 1 - \prod_{i=1}^{2}(1 - 1 + e^{-\lambda t}) \tag{3.5}$$

$$= 1 - \prod_{i=1}^{2} e^{-\lambda t} \tag{3.6}$$

$$= 1 - e^{-2\lambda t} \tag{3.7}$$

Ein exponentialverteiltes Seriensystem, bestehend aus zwei Komponenten, wird dementsprechend nach Gleichung 3.7 berechnet.

#### 3.2.1.2 Parallelsystem

Abbildung 3.2 zeigt das Blockschaltbild und den zugehörigen Fehlerbaum eines Parallelsystems bestehend aus zwei Komponenten. Parallelsysteme werden im Fehlerbaum durch ein Oder-Gatter modelliert.

Die Überlebenswahrscheinlichkeit eines Parallelsystems ist gegeben durch

$$R(t) = 1 - \prod_{i=1}^{n}(1 - p_i(t)) \tag{3.8}$$

Die Gegenwahrscheinlichkeit entspricht der Ausfallwahrscheinlichkeit und wird durch

$$F(t) = \prod_{i=1}^{n} q_i(t) \tag{3.9}$$

beschrieben. Dabei ist $p_i(t)$ die Überlebenswahrscheinlichkeit und $q_i(t)$ die Ausfallwahrscheinlichkeit der Komponente $i$. Die Anwendung des Exponentialmodells auf die Glei-

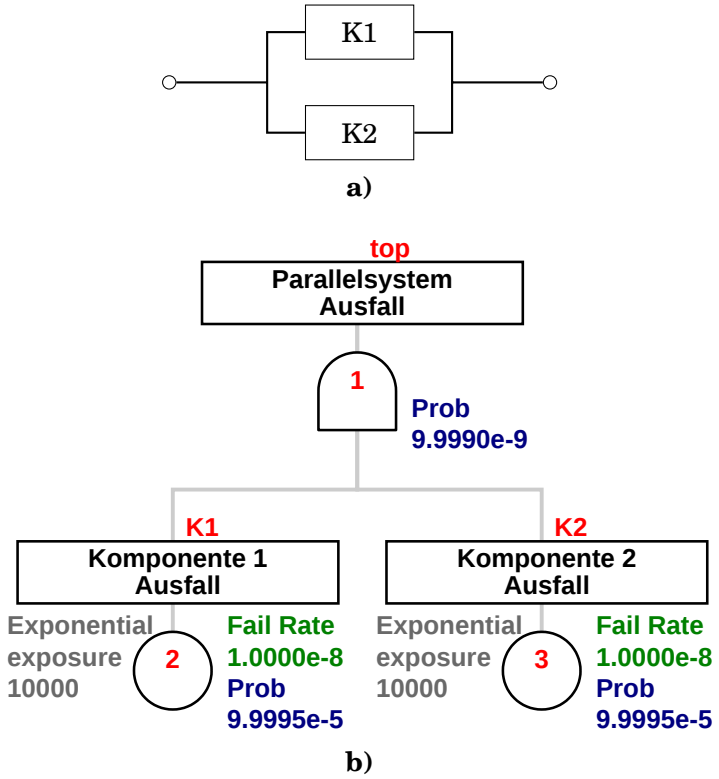


**Abb. 3.2:** Blockschaltbild a) und Fehlerbaum b) eines einfachen Parallelsystems mit zwei Komponenten

chuung zur Berechnung der Ausfallwahrscheinlichkeit des Parallelsystems ergibt folgendes.

$$F(t) = \prod_{i=1}^{2}(1 - e^{-\lambda t}) \tag{3.10}$$

$$= e^{-2\lambda t} - 2e^{-\lambda t} + 1 \tag{3.11}$$

Die Ausfallwahrscheinlichkeit eines Parallelsystems, bestehend aus zwei Komponenten, wird folglich durch Gleichung 3.11 beschrieben.

### 3.2.2 Markov-Ketten

Im Folgenden werden die Methoden Variation der Konstanten, Matrizen Diagonalisierung und Laplace Transformation zur Berechnung von Markovprozessen vorgestellt.

#### 3.2.2.1 Variation der Konstanten

Zur Veranschaulichung wird das in Abbildung 3.2 gezeigte nicht reparierbare Parallelsystem herangezogen. Der zugehörige Markovprozess ist in Abbildung 3.3 zu sehen.

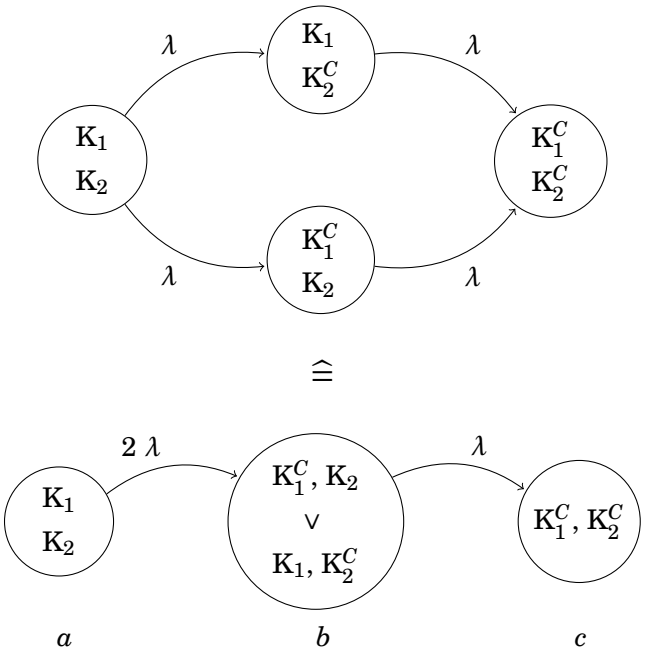


**Abb. 3.3:** Markov Kette eines nicht-reparierbaren Parallelsystems. Äquivalente Darstellungen.

Die Übergangsmatrix $A$ ist gegeben durch

$$A = \begin{array}{c} \\ a \\ b \\ c \end{array} \begin{array}{c} \begin{array}{ccc} a & b & c \end{array} \\ \begin{pmatrix} -2\lambda & 2\lambda & 0 \\ 0 & -\lambda & \lambda \\ 0 & 0 & 0 \end{pmatrix} \end{array} \tag{3.12}$$

Die Zustandswahrscheinlichkeiten werden dabei durch

$$P = \begin{pmatrix} P_a \\ P_b \\ P_c \end{pmatrix} \tag{3.13}$$

beschrieben. Es gilt

$$P_a + P_b + P_c = 1 \quad \forall\, t \tag{3.14}$$

und

$$P_a(0) = P_b(0) = P_c(0) = 0 \tag{3.15}$$

Mit Hilfe der Kolmogorov Rückwärtsgleichung (Gl. 3.16) wird die Berechnung der Zustandswahrscheinlichkeiten über ein Differentialgleichungssystem ermöglicht.

$$\dot{P} = A^T \, P \tag{3.16}$$

Das Einsetzen der Übergangsmatrix $A$ führt zu

$$\begin{pmatrix} \dot{P}_a \\ \dot{P}_b \\ \dot{P}_c \end{pmatrix} = \begin{pmatrix} -2\,\lambda & 0 & 0 \\ 2\,\lambda & -\lambda & 0 \\ 0 & \lambda & 0 \end{pmatrix} \begin{pmatrix} P_a \\ P_b \\ P_c \end{pmatrix} \tag{3.17}$$

Hieraus können die linearen Differentialgleichungen erster Ordnung abgeleitet werden.

$$\dot{P}_a = -2\,\lambda\,P_a \tag{3.18}$$

$$\dot{P}_b = 2\,\lambda\,P_a - \lambda\,P_b \tag{3.19}$$

$$\dot{P}_c = \lambda\,P_b \tag{3.20}$$

Die Lösung erfolgt durch

$$P_a = C\,e^{-2\lambda t}, \quad P_a(0) = 1 \quad \rightarrow \quad C = 1 \tag{3.21}$$

und lautet für $P_a$

$$\boxed{P_a = e^{-2\lambda t}} \tag{3.22}$$

Die DGL von Gl. 3.19 wird mittels der Methode der Variation der Konstanten gelöst
($\dot{y} = -\lambda y \quad \rightarrow \quad y = e^{-\lambda t}$)

$\dot{P}_b = 2\,\lambda\,P_a - \lambda\,P_b$
$P_b = P_{b,h} + P_{b,p}$

Die zu lösende DGL lautet

$$\dot{P}_b = -\lambda\,P_b \tag{3.23}$$

$$\dot{P}_b + \lambda\,P_b = 0 \tag{3.24}$$

$$\rightarrow P_b = k(t)\,e^{-\lambda t} \tag{3.25}$$

Die Variation der Konstanten führt zu

$$\dot{P}_b = k'(t)\ e^{-\lambda t} - k(t)\ \lambda\ e^{-\lambda t} \tag{3.26}$$

Gl. 3.25 wird eingesetzt.

$$\dot{P}_b = k'(t)\ e^{-\lambda t} - \lambda\ P_b \tag{3.27}$$

Der Vergleich mit Gl. 3.19 führt zu:

$$\left.\begin{aligned} \dot{P}_b &= 2\ \lambda\ P_a - \lambda\ P_b \\ \dot{P}_b &= k'(t)\ e^{-\lambda t} - \lambda\ P_b \end{aligned}\right\} \quad k'(t)\ e^{-\lambda t} = 2\ \lambda\ P_a \tag{3.28}$$

Gl. 3.22 wird eingesetzt.

$$k'(t)\ e^{-\lambda t} = 2\ \lambda\ e^{-2\lambda t} \tag{3.29}$$

Das führt zu

$$k'(t) = \frac{2\ \lambda\ e^{-2\lambda t}}{e^{-\lambda t}} = 2\ \lambda\ e^{-2\lambda t+\lambda t} = 2\ \lambda\ e^{-\lambda t} \tag{3.30}$$

$k(t)$ wird mittels Integration ermittelt.

$$\begin{aligned} k(t) &= \int_0^t 2\ \lambda\ e^{-\lambda u}\, du \qquad (3.31)\\ &= 2\int_0^t \lambda\ e^{-\lambda u}\, du \\ &= 2\ \left[-\frac{\lambda}{\lambda}\ e^{-\lambda u}\right]_0^t \\ &= -2\ \left[e^{-\lambda u}\right]_0^t \\ &= -2\ \left(e^{-\lambda t} - e^0\right) \\ &= -2\ \left(e^{-\lambda t} - 1\right) \\ &= 2\left(1 - e^{-\lambda t}\right) \qquad (3.32) \end{aligned}$$

Ein Einsetzen in Gl. 3.25 führt zu

$$P_b = k(t)\ e^{-\lambda t} = 2\left(1 - e^{-\lambda t}\right)\ e^{-\lambda t} = 2\left(e^{-\lambda t} - e^{-\lambda t}\ e^{-\lambda t}\right) \tag{3.33}$$

Durch Ausklammern folgt

$$\boxed{P_b = 2\left(e^{-\lambda t} - e^{-2\lambda t}\right)} \tag{3.34}$$

Unter Verwendung von Gl. 3.14 kann dann auch $P_c$ bestimmt werden.

$$P_c = 1 - P_a - P_b = 1 - e^{-2\lambda t} - 2\, e^{-\lambda t} + 2\, e^{-2\lambda t} \tag{3.35}$$

Die Ausklammerung ergibt

$$\boxed{P_c = 1 - 2\, e^{-\lambda t} + e^{-2\lambda t}} \tag{3.36}$$

Eine Probe ist durch den Vergleich mit Gl. 3.20 und Gl. 3.34 möglich.

$$\dot{P}_c = 2\,\lambda\, e^{-\lambda t} - 2\,\lambda\, e^{-2\lambda t} = 2\,\lambda \left( e^{-\lambda t} - e^{-2\lambda t} \right) = \lambda\, P_b \tag{3.37}$$

#### 3.2.2.2 Matrizen Diagonalisierung

Die diagonale Form einer Matrix wird durch

$$B^{-1} A\, B = D = \begin{pmatrix} \zeta_1 & 0 \\ 0 & \zeta_n \end{pmatrix} \tag{3.38}$$

beschrieben. Eine Umstellung nach der Übergangsmatrix $A$ führt zu

$$A = B\, D\, B^{-1} \tag{3.39}$$

Die Voraussetzungen für eine Diagonalisierung sind

- die Matrix A ist quadratisch
- die Matrix B ist invertierbar
- die n x n Matrix hat n Eigenwerte mit n Eigenvektoren

Wird Gl. 3.39 mit B multipliziert ergibt sich

$$A\, B = B\, D \tag{3.40}$$

Daraus folgt, dass die i-te Spalten von A B und B D identisch sind. Es folgt die Multiplikation mit dem Einheitsvektor $e_i$ und das Setzen von Klammern.

$$A\, B\, e_i = B\, (D\, e_i) = B\, \zeta_i\, e_i = \zeta_i\, B\, e_i \tag{3.41}$$

$D\, e_i$ beschreibt die i-te Spalte der Matrix $D$ und entspricht somit dem Wert der Hauptdiagonalen $\zeta_n$ mal dem Einheitsvektor $e_i$. Nach Umstellen folgt der Ausdruck $B\, e_i$, wel-

cher ebenfalls die i-te Spalte von $B$ beschreibt. Wobei jede dieser Spalten ein Vektor ist, weshalb $B\ e_i$ auch als $\vec{x}_i$ geschrieben werden kann:

$$\boxed{A\ \vec{x}_i = \zeta_i\ \vec{x}_i\ ,\quad \vec{x}_i \neq 0} \tag{3.42}$$

Ein Vektor, der mit einer Matrix multipliziert wird und dabei seine eigene Richtung beibehält wird Eigenvektor genannt, wobei $\zeta_i$ den Eigenwert darstellt. Hierbei handelt es sich um das Eigenwertproblem, wenn $\vec{x}_i$ kein Nullvektor ist. Da $\vec{x}_i$ per Definition ungleich null ist, da $B$ sonst nicht diagonalisierbar wäre, ist dies hier gegeben. Folglich werden für die Diagonalisierung einer Matrix ihre Eigenwerte mit den zugehörigen Eigenvektoren benötigt. Dabei stehen die Eigenwerte $\zeta_i$ auf der Hauptdiagonalen von $D$ und die Eigenvektoren $\vec{x}_i$ werden durch die Spalten von $B$ abgebildet.

$$\dot{P} = A^T\ P \tag{3.43}$$

Zur Lösung von Gleichung 3.43 muss die Übergangsmatrix $A$ transponiert werden.

$$A^T = \begin{pmatrix} -2\lambda & 0 & 0 \\ 2\lambda & -\lambda & 0 \\ 0 & \lambda & 0 \end{pmatrix} \tag{3.44}$$

Die Zustandswahrscheinlichkeiten $P$ werden dann ermittelt durch

$$P = e^{A^T t}\ P_0 \tag{3.45}$$

Die Berechnung von $e^{A^T t}$ erfolgt über $A^T = B\ D\ B^{-1} \rightarrow e^{A^T t} = B\ e^{Dt}\ B^{-1}$

Zunächst werden die Bedingungen für die Diagonalisierung von $A^T$ geprüft. Die 3 x 3 Matrix ist diagonalisierbar, wenn sie auch 3 Eigenvektoren besitzt. Um diese zu ermitteln, müssen zunächst die Eigenwerte $\zeta$ berechnet werden, indem sie von der Hauptdiagonalen abgezogen werden. Die Subtraktion der Einheitswerte $\zeta_i$ multipliziert mit dem Einheitsvektor $e_i$ entspricht dieser Rechenoperation.

$$A^T - \zeta_i\ e_i = \begin{pmatrix} -2\lambda - \zeta & 0 & 0 \\ 2\lambda & -\lambda - \zeta & 0 \\ 0 & \lambda & 0 - \zeta \end{pmatrix} \tag{3.46}$$

Das Abziehen von Eigenwerten auf der Hauptdiagonalen führt zum Rangverlust der Matrix. Folglich muss ihre Determinante null werden.

$$det\left[A^T - \zeta_i e_i\right] = (-2\lambda - \zeta)(-\lambda - \zeta)(-\zeta) \stackrel{!}{=} 0 \tag{3.47}$$

Hieraus ergeben sich die Nullstellen

$$\zeta_1 = -2\ \lambda, \quad \zeta_2 = -\lambda, \quad \zeta_3 = 0$$

die gleichzeitig die Eigenwerte darstellen. Die Diagonalisierbarkeit der Matrix kann festgestellt werden, da drei unterschiedliche Eigenwerte existieren, die auf drei unterschiedliche Eigenvektoren schließen.

Die Eigenvektoren werden berechnet, indem die identifizierten Eigenwerte jeweils von der Hauptdiagonalen abgezogen werden. Die Eigenvektoren der drei Eigenwerte werden im Folgenden berechnet.

Eigenvektor für $\zeta_1 = -2\lambda$:

$$A^T - \zeta_1 e_i = \begin{array}{c} \begin{matrix} x_1 & x_2 & x_3 \end{matrix} \\ \left(\begin{array}{ccc|c} -2\lambda + 2\lambda & 0 & 0 & 0 \\ 2\lambda & -\lambda + 2\lambda & 0 & 0 \\ 0 & \lambda & 2\lambda & 0 \end{array}\right) \end{array} \begin{matrix} \\ \text{I} \\ \text{II} \\ \text{III} \end{matrix} \tag{3.48}$$

$$= \left(\begin{array}{ccc|c} 0 & 0 & 0 & 0 \\ 2\lambda & \lambda & 0 & 0 \\ 0 & \lambda & 2\lambda & 0 \end{array}\right) \begin{matrix} \\ \cdot\frac{1}{2\lambda} \\ \cdot\frac{1}{\lambda} \end{matrix} \tag{3.49}$$

$$= \left(\begin{array}{ccc|c} 0 & 0 & 0 & 0 \\ 1 & \frac{1}{2} & 0 & 0 \\ 0 & 1 & 2 & 0 \end{array}\right) +(-\tfrac{1}{2} \cdot \text{III}) \tag{3.50}$$

$$= \left(\begin{array}{ccc|c} 0 & 0 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 2 & 0 \end{array}\right) \tag{3.51}$$

Das führt zu

$$x_1 - x_3 = 0 \tag{3.52}$$
$$x_2 + 2x_3 = 0 \tag{3.53}$$

$$x_1 = x_3 \tag{3.54}$$
$$x_2 = -2x_3 \tag{3.55}$$
$$x_3 = x_3 \tag{3.56}$$

Woraus sich folgende Lösung für den Eigenvektor $v_1$ ergibt:

$$\mathbb{L}_1 = \begin{pmatrix} x_3 \\ -2x_2 \\ x_3 \end{pmatrix} = x_3 \begin{pmatrix} 1 \\ -2 \\ 1 \end{pmatrix}, \quad x_3 \in \mathbb{R} \setminus \{0\} \tag{3.57}$$

$$\text{sei} \quad x_3 = 1, \quad v_1 = \begin{pmatrix} 1 \\ -2 \\ 1 \end{pmatrix} \tag{3.58}$$

Eigenvektor für $\zeta_2 = -\lambda$:

$$A^T - \zeta_2 e_i = \begin{array}{c} \begin{matrix} x_1 & x_2 & x_3 \end{matrix} \\ \left(\begin{array}{ccc|c} -2\lambda+\lambda & 0 & 0 & 0 \\ 2\lambda & -\lambda+\lambda & 0 & 0 \\ 0 & \lambda & \lambda & 0 \end{array}\right) \end{array} \begin{matrix} \\ \text{I} \\ \text{II} \\ \text{III} \end{matrix} \tag{3.59}$$

$$= \left(\begin{array}{ccc|c} -\lambda & 0 & 0 & 0 \\ 2\lambda & 0 & 0 & 0 \\ 0 & \lambda & \lambda & 0 \end{array}\right) \cdot\left(-\tfrac{1}{\lambda}\right) \tag{3.60}$$

$$= \left(\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 2\lambda & 0 & 0 & 0 \\ 0 & \lambda & \lambda & 0 \end{array}\right) \begin{matrix} \\ +(-2\lambda \cdot \text{I}) \\ \cdot\frac{1}{\lambda} \end{matrix} \tag{3.61}$$

$$= \left(\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{array}\right) \tag{3.62}$$

Das führt zu

$$x_1 = 0 \tag{3.63}$$

$$x_2 + x_3 = 0 \tag{3.64}$$

$$x_2 = -x_3 \tag{3.65}$$

$$x_3 = x_3 \tag{3.66}$$

Woraus sich folgende Lösung für den Eigenvektor $v_2$ ergibt:

$$\mathbb{L}_2 = \begin{pmatrix} 0 \\ -x_3 \\ x_3 \end{pmatrix} = x_3 \begin{pmatrix} 0 \\ -1 \\ 1 \end{pmatrix}, \quad x_3 \in \mathbb{R} \setminus \{0\} \tag{3.67}$$

$$\text{sei} \quad x_3 = 1, \quad v_2 = \begin{pmatrix} 0 \\ -1 \\ 1 \end{pmatrix} \tag{3.68}$$

Eigenvektor für $\zeta_3 = 0$ :

$$A^T - \zeta_3 e_i = \begin{array}{c} \begin{matrix} x_1 & x_2 & x_3 \end{matrix} \\ \left(\begin{array}{ccc|c} -2\lambda & 0 & 0 & 0 \\ 2\lambda & -\lambda & 0 & 0 \\ 0 & \lambda & 0 & 0 \end{array}\right) \end{array} \begin{array}{c} \\ \begin{matrix} \text{I} \\ \text{II} \\ \text{III} \end{matrix} \end{array} \quad \begin{array}{l} \\ \cdot\left(-\frac{1}{2\lambda}\right) \\ \\ \\ \end{array} \tag{3.69}$$

$$= \left(\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 2\lambda & -\lambda & 0 & 0 \\ 0 & \lambda & 0 & 0 \end{array}\right) \begin{array}{l} \\ \\ +(-2\lambda \cdot \text{I}) \end{array} \tag{3.70}$$

$$= \left(\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & -\lambda & 0 & 0 \\ 0 & \lambda & 0 & 0 \end{array}\right) \cdot\left(-\tfrac{1}{\lambda}\right) \tag{3.71}$$

$$= \left(\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & \lambda & 0 & 0 \end{array}\right) +(-\lambda \cdot \mathrm{II}) \tag{3.72}$$

$$= \left(\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right) \tag{3.73}$$

Das führt zu

$$x_1 = 0 \tag{3.74}$$

$$x_2 = 0 \tag{3.75}$$

$$x_3 = x_3 \tag{3.76}$$

Woraus sich folgende Lösung für den Eigenvektor $v_3$ ergibt:

$$\mathbb{L}_3 = \begin{pmatrix} 0 \\ 0 \\ x_3 \end{pmatrix} = x_3 \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \quad x_3 \in \mathbb{R} \setminus \{0\} \tag{3.77}$$

$$\text{sei} \quad x_3 = 1, \quad v_3 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \tag{3.78}$$

Die diagonale Form der Matrix ist zusammengesetzt durch

$$D = \begin{pmatrix} \zeta_1 & 0 & 0 \\ 0 & \zeta_2 & 0 \\ 0 & 0 & \zeta_3 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & -\lambda & 0 \\ 0 & 0 & -2\lambda \end{pmatrix} \tag{3.79}$$

$$B = (v_1,\ v_2,\ v_3) = \begin{pmatrix} 0 & 0 & 1 \\ 0 & -1 & -2 \\ 1 & 1 & 1 \end{pmatrix} \tag{3.80}$$

$$B^{-1} = \begin{pmatrix} 1 & 1 & 1 \\ -2 & -1 & 0 \\ 1 & 0 & 0 \end{pmatrix} \tag{3.81}$$

Es steht die Potenzierung zur Basis e aus.

$$A^T = B\ D\ B^{-1} \quad \rightarrow \quad e^{A^T t} = B\ e^{Dt}\ B^{-1} \tag{3.82}$$

Die Ergebnismatrix wird durch Ausmultiplizieren erhalten.

$$e^{A^T t} = \begin{pmatrix} 0 & 0 & 1 \\ 0 & -1 & -2 \\ 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & e^{-\lambda t} & 0 \\ 0 & 0 & e^{-2\lambda t} \end{pmatrix} \begin{pmatrix} 1 & 1 & 1 \\ -2 & -1 & 1 \\ 1 & 0 & 0 \end{pmatrix} \tag{3.83}$$

$$= \begin{pmatrix} 0 & 0 & e^{-2\lambda t} \\ 0 & -e^{-\lambda t} & -2e^{-2\lambda t} \\ 1 & e^{-\lambda t} & e^{-2\lambda t} \end{pmatrix} \begin{pmatrix} 1 & 1 & 1 \\ -2 & -1 & 1 \\ 1 & 0 & 0 \end{pmatrix} \tag{3.84}$$

$$= \begin{pmatrix} e^{-2\lambda t} & 0 & 0 \\ 2e^{-\lambda t} - 2e^{-2\lambda t} & e^{-\lambda t} & -e^{-\lambda t} \\ 1 - 2e^{-\lambda t} + e^{-2\lambda t} & 1 - e^{-\lambda t} & 1 + e^{-\lambda t} \end{pmatrix} \tag{3.85}$$

Die Multiplikation mit der Anfangsbedingung $P_0$ ergibt die Zustandswahrscheinlichkeit $P$.

$$P = e^{A^T t}\ P_0 \quad \text{mit} \quad P_0 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \tag{3.86}$$

$P_0$ steht für den Beginn im Zustand 1 von 3 (vgl. Abb. 3.3). Die Zustandswahrscheinlichkeiten für die Zustände $a$, $b$ und $c$ lauten

$$P = \begin{pmatrix} e^{-2\lambda t} \\ 2e^{-\lambda t} - 2e^{-2\lambda t} \\ 1 - 2e^{-\lambda t} + e^{-2\lambda t} \end{pmatrix} = \begin{pmatrix} P_a \\ P_b \\ P_c \end{pmatrix} \tag{3.87}$$

#### 3.2.2.3 Laplace Transformation

Die Laplace Transformation kann ebenfalls für die Lösung eines Markovprozesses verwendet werden. Sie ist definiert durch

$$F(s) = \int_0^\infty f(t)\, e^{-st} dt \tag{3.88}$$

$f(t)$: Originalfunktion

$F(s)$: Bildfunktion

Zunächst werden die Ableitungseigenschaften besprochen. Um die Laplace Transformierte abzuleiten, muss partiell integriert werden.

$$\begin{aligned} \mathscr{L}(\dot{f})(s) &= \int_0^\infty \dot{f}(t)\, e^{-st} dt \\ &= f(t)\, e^{-st}\big|_0^\infty + s \int_0^\infty f(t)\, e^{-st} dt \\ &= s\, F(s) - f(0) \end{aligned} \tag{3.89}$$

Daraus folgt die Lösung der Laplace Transformierten von $\dot{x} = A\, x$

$$s\, X(s) - x(0) = A\, X(s) \tag{3.90}$$

Der Ausdruck kann umgeschrieben werden als

$$(s\, I - A)\, X(s) = x(0) \tag{3.91}$$

$$X(s) = (s\, I - A)^{-1}\, x(0) \tag{3.92}$$

Die Rücktransformation führt zu

$$x(t) = \mathscr{L}^{-1}\left((s\, I - A)^{-1}\right)\, x(0) \tag{3.93}$$

$(s\, I - A)^{-1}$ wird Resolvente von A genannt. Sie beschreibt die Inverse einer mit der komplexen Zahl $z$ verschobenen Matrix. Die Resolvente ist definiert für $s \in \mathbb{C}$ ausgenommen der Eigenwerte von $A$, die durch $det\,(s\, I - A) = 0$ ermittelt werden.

$\Phi(t) = \mathscr{L}^{-1}\left((s\ I - A)^{-1}\right)$ wird Zustandsübergangsmatrix genannt. Sie bildet den Ausgangszustand zum Zeitpunkt $t$ ab:

$$x(t) = \Phi(t)\ x(0) \tag{3.94}$$

Der Zustand $x(t)$ ist eine lineare Funktion des Ausgangszustandes $x(0)$.

Die Anwendung auf das vorliegende Problem gelingt wie folgt:

$$A^T = \begin{pmatrix} -2\lambda & 0 & 0 \\ 2\lambda & -\lambda & 0 \\ 0 & \lambda & 0 \end{pmatrix} \tag{3.95}$$

$$\dot{P} = A^T\ P \tag{3.96}$$

$$s\ \mathscr{L}(P)(s) - P(0) = A^T\ \mathscr{L}(P)(s) \tag{3.97}$$

$$(s\ I - A^T)\ \mathscr{L}(P)(s) = P(0) \tag{3.98}$$

$$\mathscr{L}(P)(s) = (s\ I - A^T)^{-1}\ P(0) \tag{3.99}$$

$$B = (s\ I - A^T) = \begin{pmatrix} s+2\lambda & 0 & 0 \\ -2\lambda & s+\lambda & 0 \\ 0 & -\lambda & s \end{pmatrix} \tag{3.100}$$

$$det\ (B) = (s+2\lambda)(s+\lambda)(s) \tag{3.101}$$

$$B^{-1} = \frac{1}{(s+2\lambda)\ (s+\lambda)\ (s)} \begin{pmatrix} (s+\lambda)s & -2\lambda s & 2\lambda^2 \\ 0 & (s+2\lambda)s & (s+2\lambda)-\lambda \\ 0 & 0 & (s+2\lambda)(s+\lambda) \end{pmatrix} \tag{3.102}$$

$$= \begin{pmatrix} \frac{1}{s+2\lambda} & \frac{-2\lambda}{(s+2\lambda)(s+\lambda)} & \frac{2\lambda^2}{(s+2\lambda)(s+\lambda)s} \\ 0 & \frac{1}{s+\lambda} & \frac{-\lambda}{(s+\lambda)s} \\ 0 & 0 & \frac{1}{s} \end{pmatrix}^T \tag{3.103}$$

Eine Auswahl von Rücktransformationen, die für den vorliegenden Anwendungsfall benötigt werden, ist im Folgenden abgebildet.

$$F(s) \bullet\!\!-\!\!\circ f(t) \tag{3.104}$$

$$\frac{1}{s} \bullet\!\!-\!\!\circ 1 \tag{3.105}$$

$$\frac{1}{s-a} \bullet\!\!-\!\!\circ e^{at} \tag{3.106}$$

Die Rücktransformation von $\frac{1}{s+2\lambda}$ unter Verwendung von Gleichung 3.106 sieht wie folgt aus

$$\frac{1}{s+2\lambda} \bullet\!\!-\!\!\circ e^{-2\lambda t} \tag{3.107}$$

Die Rücktransformation von $\frac{-2\lambda}{(s+2\lambda)(s+\lambda)}$ unter Verwendung von Gleichung 3.106 lautet

$$\frac{-2\lambda}{(s+2\lambda)(s+\lambda)} = \frac{A}{(s+2\lambda)} + \frac{B}{(s+\lambda)} \tag{3.108}$$

$$= \frac{A(s+\lambda)}{(s+2\lambda)(s+\lambda)} + \frac{B(s+2\lambda)}{(s+\lambda)(s+2\lambda)} \tag{3.109}$$

$$= \frac{A(s+\lambda)+B(s+2\lambda)}{(s+2\lambda)(s+\lambda)} \tag{3.110}$$

$$= \frac{As+A\lambda+Bs+2B\lambda}{(s+2\lambda)(s+\lambda)} \tag{3.111}$$

$$= \frac{s(A+B)+\lambda(A+2B)}{(s+2\lambda)(s+\lambda)} \tag{3.112}$$

Der Koeffizientenvergleich führt zu

$$\left.\begin{aligned} A+B &= 0 \\ A+2B &= -2 \end{aligned}\right\} \quad A = 2\,, \quad B = -2 \tag{3.113}$$

$$\frac{2}{(s+2\lambda)} - \frac{2}{(s+\lambda)} \bullet\!\!-\!\!\circ 2\,e^{-2\lambda t} - 2\,e^{-\lambda t} \tag{3.114}$$

Die Rücktransformation von $\frac{2\lambda^2}{(s+2\lambda)(s+\lambda)s}$ unter Verwendung von Gl. 3.106 und Gl. 3.105 ergeben

$$\frac{2\lambda^2}{(s+2\lambda)(s+\lambda)s} = \frac{A}{(s+2\lambda)} + \frac{B}{(s+\lambda)} + \frac{C}{s} \tag{3.115}$$

$$= \frac{A(s+\lambda)s}{(s+2\lambda)(s+\lambda)s} + \frac{B(s+2\lambda)s}{(s+2\lambda)(s+\lambda)s} + \frac{C(s+2\lambda)(s+\lambda)}{(s+2\lambda)(s+\lambda)s} \tag{3.116}$$

$$= \frac{(As+A\lambda)s + (Bs+2B\lambda)s + (Cs+2C\lambda)(s+\lambda)}{(s+2\lambda)(s+\lambda)s} \tag{3.117}$$

$$= \frac{As^2 + A\lambda s + Bs^2 + 2B\lambda s + cs^2 + Cs\lambda + 2C\lambda s + 2C\lambda^2}{(s+2\lambda)(s+\lambda)s} \tag{3.118}$$

$$= \frac{s^2(A+B+C) + \lambda s(A+2B+3C) + \lambda^2(2C)}{(s+2\lambda)(s+\lambda)s} \tag{3.119}$$

der Koeffizientenvergleich führt zu

$$\left.\begin{array}{l} A+B+C=0 \\ A+2B+3C=0 \\ 2C=2 \end{array}\right\} \quad A=1\,, \quad B=-2\,, \quad C=1 \tag{3.120}$$

$$\frac{1}{(s+2\lambda)} - \frac{2}{(s+\lambda)} + \frac{1}{s} \bullet\!\!-\!\!\circ\; 1-2\,e^{-\lambda t} + e^{-2\lambda t} \tag{3.121}$$

## 3.3 Beispiele für numerische Lösungen

### 3.3.1 Simulation des Ausfallzeitpunktes einer exponentialverteilten Komponente

Bei komplexen Problemen, bei denen nicht jede Komponenten von Hand berechnet werden kann, können Monte-Carlo-Simulationen sinnvoll sein (vgl. Abb. 3.4). Simuliert wird eine exponentialverteilte Komponente mit der Ausfallrate $\lambda = 10^{-4}\ h^{-1}$. Der Erwartungswert (rote Linie) wird durch

$$E(T) = \frac{1}{\lambda} = MTBF = 10.000\ h \tag{3.122}$$

beschrieben.

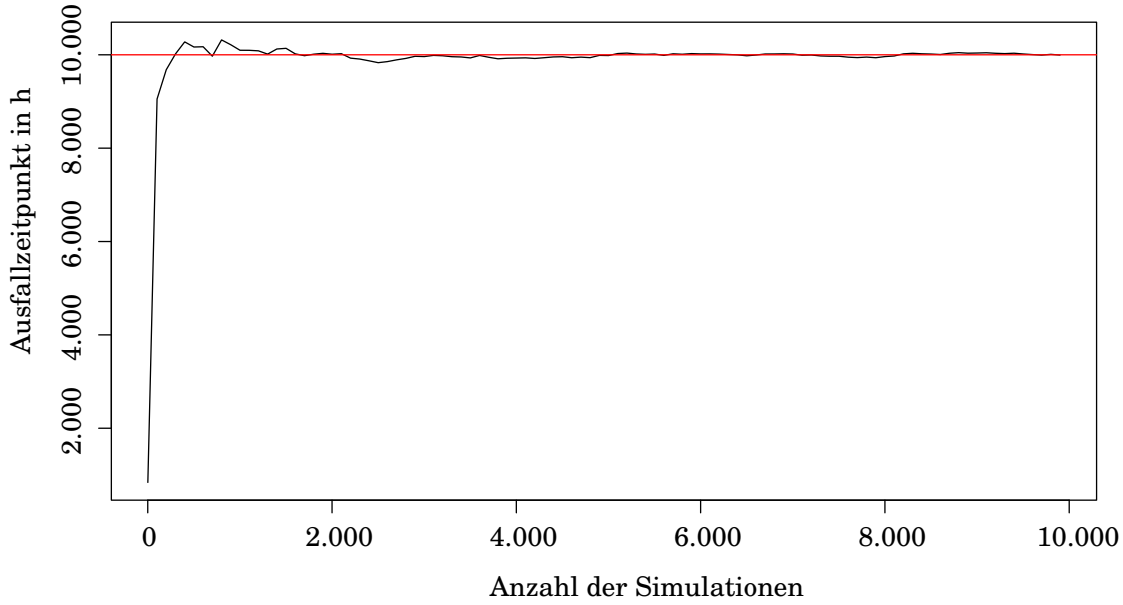


**Abb. 3.4:** Simulation des Ausfallzeitpunktes einer exponentialverteilten Komponente

Gemäß dem Gesetz der großen Zahlen kann beobachtet werden, dass mit zunehmender Simulationszahl das Ergebnis sich immer weiter der analytischen Lösung von Gleichung 3.122 annähert.

### 3.3.2 Simulation des Ausfallzeitpunktes einer weibullverteilten Komponente

Die Simulation kann unabhängig von der Verteilung durchgeführt werden. Die Wahl der folgenden Parameter erlauben es, die Exponentialverteilung durch eine Weibullverteilung anzunähern.

Lageparameter $\alpha = 10^{-4} \ \left(\frac{1}{h}\right)^{\beta}$
Ausfallsteilheit $\beta = 1$
charakteristische Lebensdauer $\eta = \alpha^{-1/\beta}$

Der Erwartungswert (rote Linie) wird beschrieben durch

$$E(T) = \int_0^{\infty} e^{-\alpha \ t^{\beta}} dt = 10.000 \ h \tag{3.123}$$

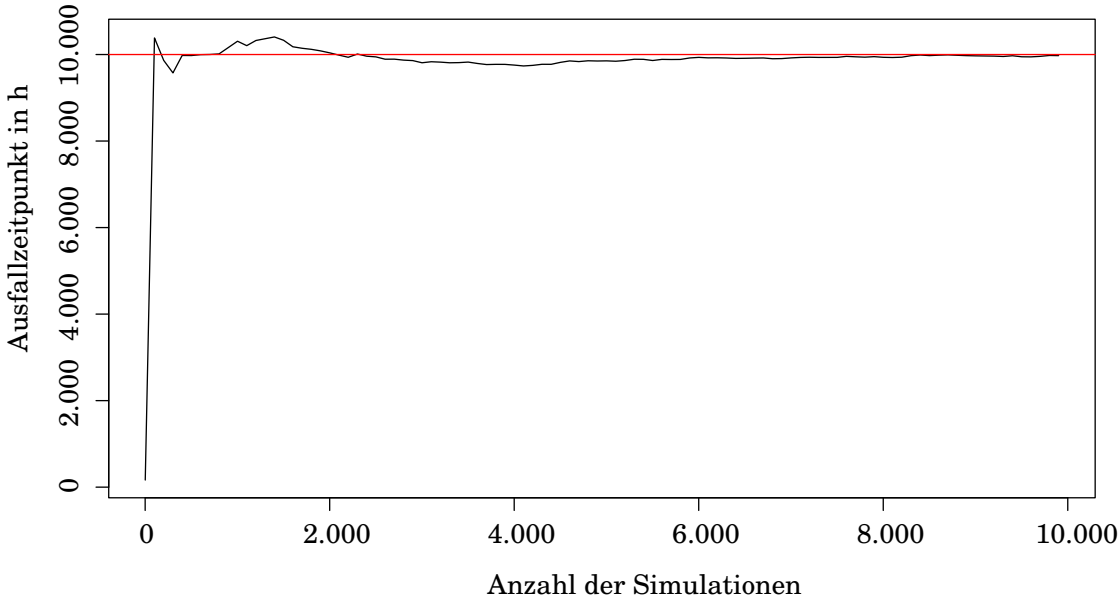


**Abb. 3.5:** Simulation des Ausfallzeitpunktes einer weibullverteilten Komponente

Es kann gesehen werden, dass durch die Wahl der Parameter das gleiche Ausfallverhalten wie im vorherigen Abschnitt simuliert wurde.

# 4 Quantitative Bewertung der Hardwarearchitekturen

## 4.1 System- und Zieldefinition

Für eine erfolgreiche Analyse der Hardwarearchitekturen werden die Systeme zunächst eingegrenzt sowie die Analyseziele näher definiert. Die in Abschnitt 2.4.3 vorgestellten Systemarchitekturen bilden die Grundlage für die folgenden Betrachtungen. Die Architektur in Abbildung 4.1 ist an Komponentenstruktur aus Abbildung 2.13 angelehnt.

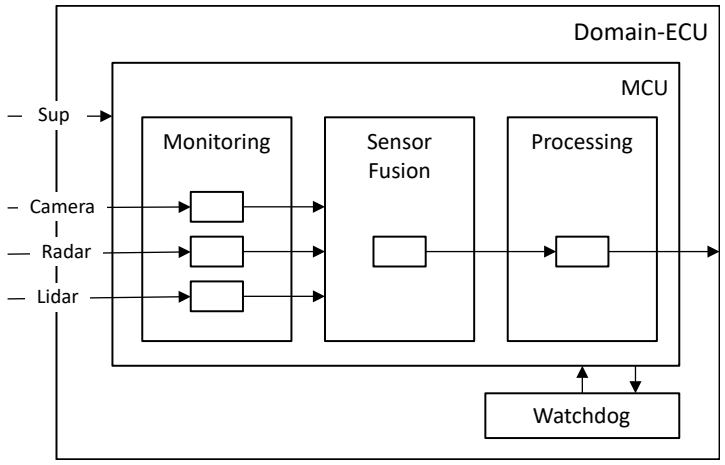


**Abb. 4.1:** 1oo1 Referenzarchitektur

Abbildung 4.1 zeigt ein 1oo1 System, das als Serienschaltung zu verstehen ist. Es dient als Referenz bei der Betrachtung der weiteren Systeme. Mit Hilfe einer Referenz kann die Effektivität der zuverlässigkeitserhöhenden Maßnahmen verifiziert werden. Unter allen Umständen ist eine Verringerung der Zuverlässigkeit im Vergleich zum 1oo1 System zu vermeiden. Das dargestellte Seriensystem beschreibt gleichzeitig einen Verarbeitungsstrang der Sensorinformationen, der in den folgenden Systemen redundant ausgelegt wird. Daneben besteht die Gemeinsamkeit der Systeme in den Systemgrenzen. Der Beobachtungsgegenstand liegt auf der ECU. Die berücksichtigen Schnittstellen liegen in den Sensorinformationen von Kamera, Radar und Lidar sowie der Energieversorgung, die gleichzeitig den Input darstellen. Obligatorisch ist daher auch die Betrachtung der Sensorzuverlässigkeit bzw. der Zuverlässigkeit der Sensorarchitektur. Bereits vor dem Beginn der Analyse ist ersichtlich, dass es sich hierbei um einen Flaschenhals handelt, da derselbe Input für verschiedene Teilsysteme verwendet wird. Das gleiche gilt für die isolierte Kommunikation des 2oo2 Systems. Im Weiteren Verlauf werden Zuverläs-

sigkeitswerte für diese kritischen Systemelemente festgelegt. Es ist denkbar, dass ihre Auslegung einen wirtschaftlichen Rahmen übersteigt. Ist das der Fall, können die Komponenten in Subsysteme mit niedrigerer Zuverlässigkeit aufgeteilt werden (vgl. ASIL Dekomposition). Eine solche Dekomposition ist jedoch nicht Gegenstand dieser Arbeit. Für identifizierte kritische Komponenten werden Zielwerte ermittelt, die als Top-Level Anforderung für ein mögliches Subsystem angesehen werden. Der Systemoutput besteht aus einer durch Majoritätsredundanz validierte Information, die an die Aktorik weitergeleitet wird, die selbst ebenfalls außerhalb der Betrachtung liegt.

Es werden verschiedene NooN und MooN Systeme untersucht. Die für die Analyse verwendete NooN Architektur ist in Abbildung 4.2 abgebildet. Es handelt sich um eine Variante der in Abbildung 2.11 vorgestellten 2oo2DFS Architektur unter Verwendung der Komponentenstruktur aus Abbildung 4.1. Das Schema kann auf jede beliebige NooN Architektur übertragen werden. Die zuvor gezeigte Serienanordnung ist vierfach redundant ausgelegt. Beide ECUs werden mittels 2oo2 Redundanz betrieben, wodurch die ECUs jeweils ein fail safe Verhalten aufweisen. Die isolierte Kommunikation zwischen den ECUs ermöglicht das fail operational Verhalten. Fällt ECU 1 aus, wird ECU 2 in Betrieb genommen. Es ist ersichtlich, dass eine solche Architektur eine Vielzahl von Komponenten besitzt, die zu einem großem Overhead führen würden. In der Praxis wird das durch kalte Redundanz vermieden. In dieser Betrachtung würde dies jedoch zu einer erheblichen Komplexitätssteigerung führen, weshalb das System als aktiv redundant angesehen wird. Die Validität des Ergebnisses erhält dadurch keine Einschränkungen. Es handelt sich eher um eine worst case Betrachtung. Die Zuverlässigkeit des aktiven Systems wird immer geringer als die des passiv unterstützten Systems sein. Zu untersuchen ist, ob die Auslegung der Komponenten des NooN Systems noch im wirtschaftlichen Rahmen liegt. Ist das nicht der Fall, kann die Standby Redundanz eine Maßnahme für weitere Analysen sein.

Das Konzept der beleuchteten MooN System ist in Abbildung 4.3 zu sehen. Dargestellt ist ein 2oo3 System, welches aus einer Kombination der 1oo1 Referenzarchitektur mit dem zweistufigen 2oo3 Vergleichs von Kohn et al. besteht [12]. Die 1oo1 Architektur ist dreifach redundant ausgelegt. Jeder MCU übermittelt seine Informationen an drei CPU Kerne von MCU 4. Jeder dieser Kerne führt einen 2oo3 Vergleichsprozess aus. Die Richtigkeit der Ergebnisse werden durch einen weiteren 2oo3 Vergleichsprozess in einem vierten Kern durchgeführt.

In Abschnitt 4.2 wird zunächst die Zuverlässigkeit der MooN bzw. NooN Architekturen mit Hilfe von Markov-Prozessen bewertet, um die inhärent ausfallsichersten

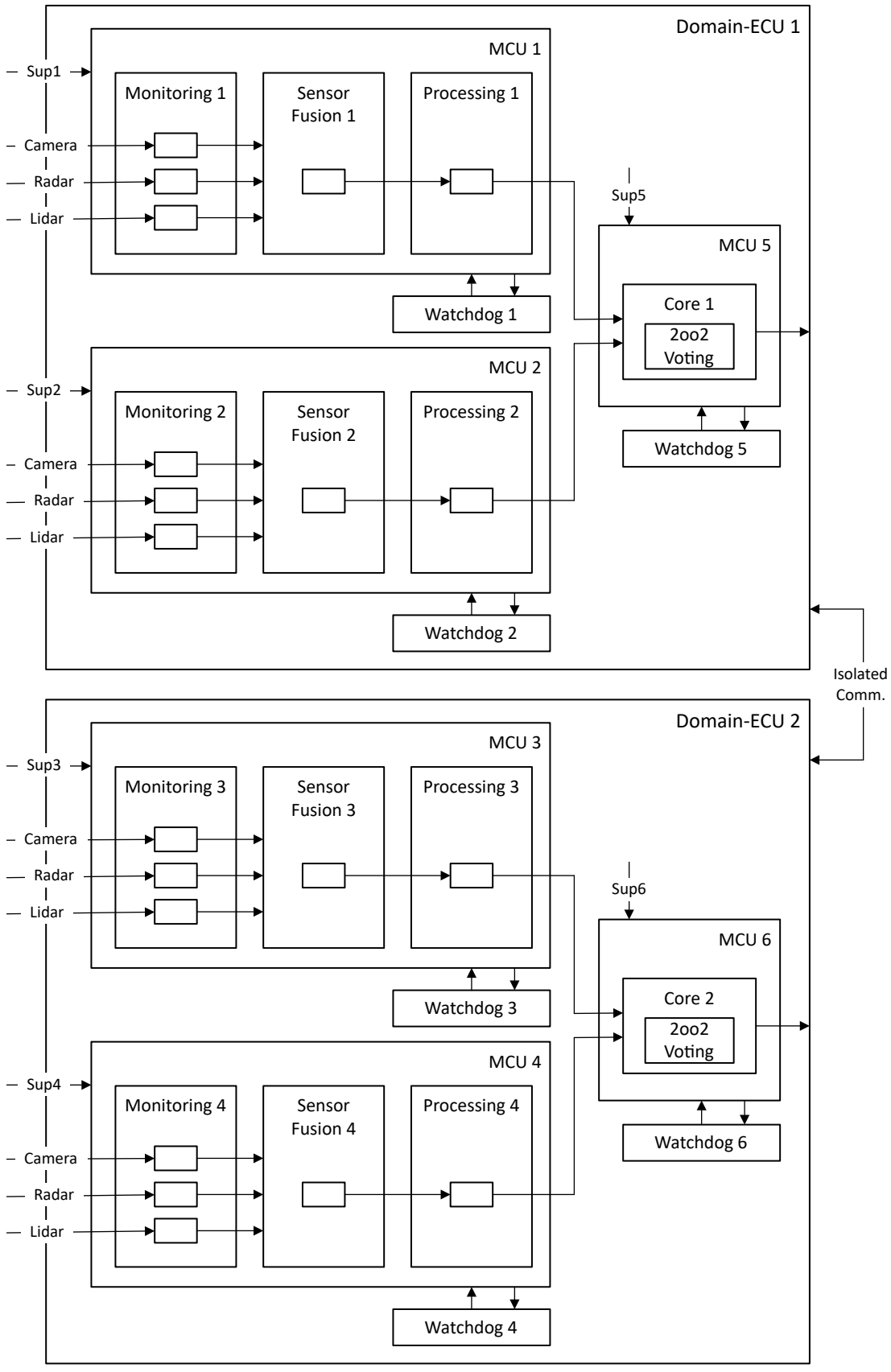


**Abb. 4.2:** 2oo2dfs Systemarchitektur für fehlertolerante Fahrfunktionen

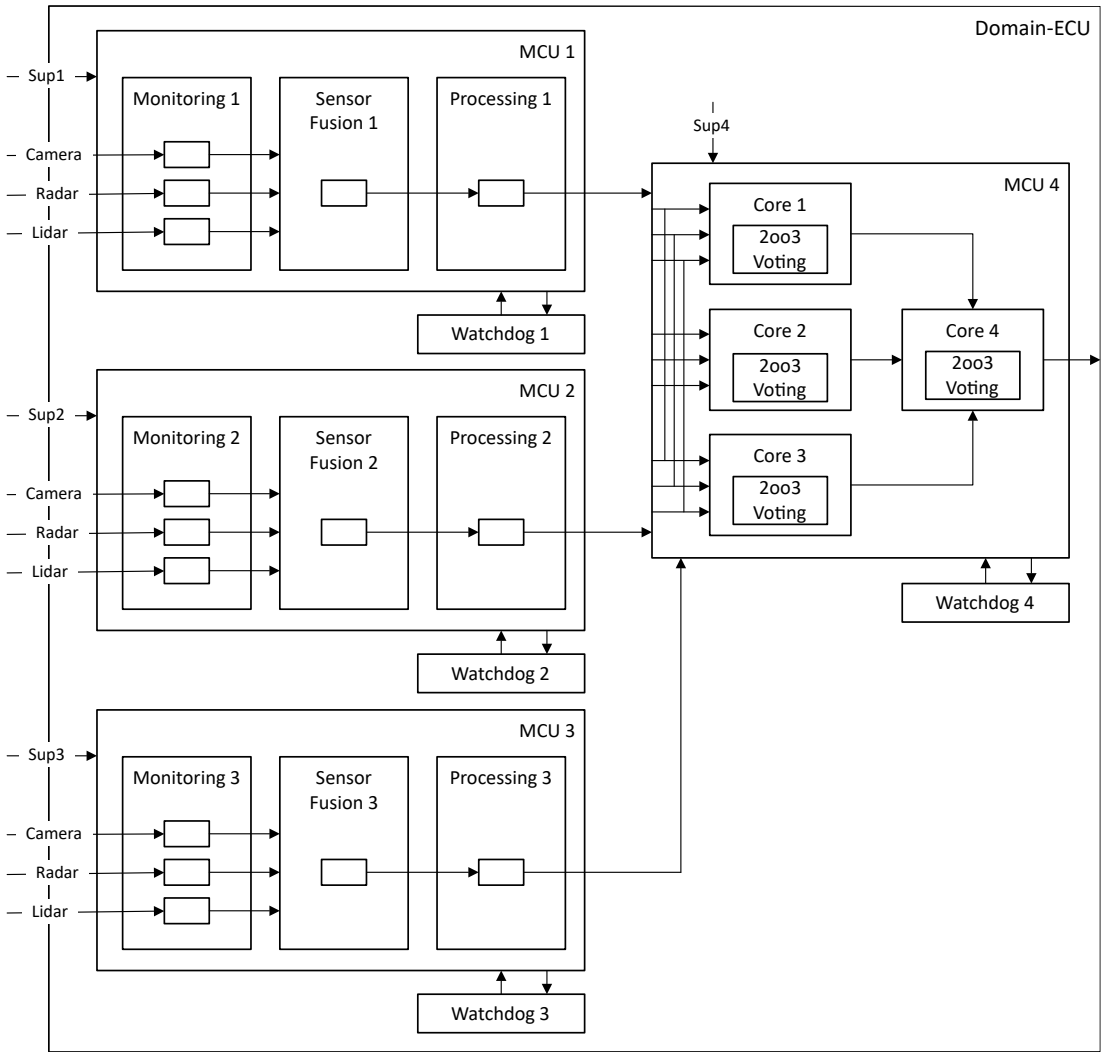


**Abb. 4.3:** 2oo3 Systemarchitektur für fehlertolerante Fahrfunktionen

Systeme zu ermitteln, die dann weiterverwendet werden. Dabei wird zunächst angenommen, das die Lebensdauern aller Komponenten identisch verteilt sind. In einem zweiten Schritt wird das System unter Berücksichtigung der in Abschnitt 2.2 ermittelten Anforderungen detaillierter modelliert. Hierfür werden Fehlerbäume herangezogen. Ziel ist es, die Zuverlässigkeitskennwerte der einzelnen Komponenten in Konformität mit der ISO 26262 individuell auszulegen. Die im Abschnitt 2.2.5 vorgestellten Zielwerte für Einpunktausfälle stellen Grundlage für die Fehlerbaumanalyse dar. Unter zur Hilfenahme von Importanzanalysen und Berücksichtigung der ISO 26262 Anforderungen werden diese einheitlichen Kennwerte weiter spezifiziert. Eine Cut Set Analyse ermöglicht die Validierung der Komponenten, die tatsächlich zu Einpunktausfällen führen. Für diese Komponenten bestehen dann keine Freiheitsgrade in der Auslegung mehr. Das gleiche

gilt für Zweipunktausfälle. Eine Reduzierung der Anforderung für diese Komponenten kann mittels ASIL Dekomposition erfolgen, die nicht Gegenstand dieser Arbeit ist. Für die restlichen Komponenten wird die Zuverlässigkeit individuell eingestellt. Am Ende der Analyse stehen spezifizierte Zuverlässigkeitsanforderungen der Komponenten und identifizierter Handlungsbedarf in der Architekturmodellierung.

## 4.2 Vergleich von Sensor- und MCU-Architekturen mittels Markovanalyse

### 4.2.1 Vorgehensweise der Markovanalyse

In Abschnitt 2.4.3 wurde gezeigt, dass etablierte fehlertolerante Systemarchitekturen aus einer Kombination von MooN Redundanzen bestehen. Auf der einen Seite steht die Sensorarchitektur. Es existieren Konzepte mit verschiedenen Sensorzahlen. Umstritten ist wie viele und welche Sensoren benötigt werden, um einen ausfallsicheren Betrieb zu ermöglichen. Auf der anderen Seite steht die Auslegung des Verarbeitungsstranges, der mit Hilfe von redundant ausgelegten MCUs modelliert wird (vgl. Abs. 4.1). Ganzheitlich betrachtet handelt es sich um eine Kombination zweier MooN Systeme, dessen Zuverlässigkeit mittels Markovanalyse berechnet wird. Zur Limitierung des Umfangs, werden maximal drei Sensoren und vier MCUs betrachtet.

Die Sensoren und MCUs können genau zwei Zustände einnehmen, ausgefallen und betriebsbereit. Es werden die Notationen $S_i$ und $M_j$ eingeführt, die besagen, dass $i$ Sensoren bzw. $j$ MCUs funktionieren. Die Ausfallrate der Sensoren wird mit $\lambda_S$ und die der MCUs wird mit $\lambda_M$ bezeichnet. Der Markovprozess startet in dem Zustand, in dem alle Komponenten betriebsbereit sind. Davon ausgehend ist jeder Zustand erreichbar, in dem alle Komponenten betriebsbereit oder ausgefallen sind. Dabei kann jedoch immer nur eine Komponente gleichzeitig ausfallen. Zur Veranschaulichung wird im Folgenden die Übergangsmatrix $A$ eines Systems bestehend aus zwei Sensoren und drei MCUs gezeigt. Über eine Diagonalisierung der Matrix (Abs. 3.2.2.2) wird die Zustandsübergangswahrscheinlichkeit für jeden Systemzustand ermittelt. Die Überlebenswahrscheinlichkeit des System ergibt sich anschließend aus der beabsichtigten Majoritätsredundanz, indem alle Wahrscheinlichkeiten der Zustände addiert werden, bei denen das System betriebsbereit ist. Die Überlebenswahrscheinlichkeit eines Systems, bestehend aus einer 1oo2 Sensor- und einer 2oo3 MCU Architektur, ergibt sich beispielsweise aus Gleichung 4.1.

$$
A = \begin{array}{c} \\ S_2M_3 \\ S_2M_2 \\ S_2M_1 \\ S_2M_0 \\ S_1M_3 \\ S_1M_2 \\ S_1M_1 \\ S_1M_0 \\ S_0M_3 \\ S_0M_2 \\ S_0M_1 \\ S_0M_0 \end{array}
\begin{array}{c}
\begin{array}{cccccccccccc} S_2M_3 & S_2M_2 & S_2M_1 & S_2M_0 & S_1M_3 & S_1M_2 & S_1M_1 & S_1M_0 & S_0M_3 & S_0M_2 & S_0M_1 & S_0M_0 \end{array} \\
\left(\begin{array}{cccccccccccc}
-2\lambda_S - 3\lambda_M & 3\lambda_M & 0 & 0 & 2\lambda_S & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -2\lambda_S - 2\lambda_M & 2\lambda_M & 0 & 2\lambda_S & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -2\lambda_S - \lambda_M & \lambda_M & 0 & 0 & 2\lambda_S & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -2\lambda_S & 0 & 0 & 0 & 2\lambda_S & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -\lambda_S - 3\lambda_M & 3\lambda_M & 0 & 0 & \lambda_S & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -\lambda_S - 2\lambda_M & 2\lambda_M & 0 & 0 & \lambda_S & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -\lambda_S - \lambda_M & -\lambda_M & 0 & 0 & \lambda_S & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -\lambda_S & 0 & 0 & 0 & \lambda_S \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -3\lambda_M & 3\lambda_M & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -2\lambda_M & 2\lambda_M & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\lambda_M & \lambda_M \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{array}\right)
\end{array}
$$

$$R_{1oo2/2oo3} = P(S_2M_3) + P(S_2M_2) + P(S_1M_3) + P(S_1M_2) \tag{4.1}$$

Diese Vorgehensweise ermöglicht für verschiedene Sensor- und MCU-Anzahlen die Überlebenswahrscheinlichkeit ihrer Architekturen hinsichtlich unterschiedlichen MooN Redundanzen gegeneinander aufzutragen. Das Ergebnis ist eine Rangfolge von MooN Architekturen für vorgegebene Sensor- und MCU-Zahlen. Die zuverlässigsten Systeme werden im folgenden Schritt weiterverwendet. Die Vorgehensweise ermöglicht ebenfalls eine Unterscheidung zwischen den Ausfallraten von Sensoren und MCUs. Zur Reduzierung der Komplexität wird zunächst eine fiktive Ausfallrate von $\lambda = 1$ angenommen, wodurch die Zeitachse überschaubar bleibt. Diese Annahme ist valide, da es in diesem Schritt lediglich um die Ermittlung einer Rangfolge geht. Das führt jedoch auch dazu, dass die Zeitachse einheitenlos bleibt. Sie wird in diesem Analyseschritt als Koeffizient für den Parameter Zeit betrachtet. Durch eine schrittweise Reduzierung der Ausfallrate $\lambda_S$ bzw. $\lambda_M$ können weitere Effekte beobachtet werden. Die Architekturen, bestehend aus Sensoren und MCUs, werden im Folgenden als $\text{MooN}_S/\text{MooN}_M$ notiert. Als Referenz wird ein 1oo1/1oo1 System verwendet.

Unter Berücksichtigung der in Abschnitt 2.4.3 vorgestellten etablierten Systemarchitekturen werden folgenden Beschränkungen getroffen: Aus wirtschaftlichen Gründen werden nicht mehr als drei Sensoren und vier MCUs eingesetzt. Die minimale MCU-Zahl wird mit drei angesetzt, um ausreichende Diagnosefähigkeiten sicherzustellen, die das Fail-Operational-Verhalten ermöglichen. Ein Betrieb mit zwei MCUs kann lediglich als fail-safe System betrieben werden. Die minimale Sensorzahl wird auf zwei festgesetzt. Die Literatur beschäftigt sich überwiegend mit der Frage, ob zwei oder drei verschiedene Sensoren notwendig sind. Teslas Vison-Only Projekt ist das einzige bekannte Konzept, dass auf die Verwendung von nur einem Sensor setzt [32]. Zudem steckt dieses Projekt noch in der Anfangsphase. Ohne wissenschaftliche Evidenz ist es nicht vernünftig weitere Arbeiten darauf basieren zu lassen.

### 4.2.2 Architekturbewertung von drei Sensoren und vier MCUs

Die Überlebenswahrscheinlichkeit der MooN-Systeme einer Hardwarearchitektur, bestehend aus drei Sensoren und vier MCUs, ist in Abbildung 4.4 dargestellt. Die Ausfallrate der Sensoren und MCUs ist zunächst identisch. Die Systeme dessen Zuverlässigkeitskurven unter der 1oo1 Referenz liegen können für eine Weiterverwendung kategorisch ausgeschlossen werden. Das beinhaltet das 3oo3/4oo4, das 2oo3/4oo4, das 3oo3/3oo4, das 1oo3/4oo4, das 3oo3/2oo4, das 3oo3/1oo4 und das 2oo3/3oo4 System. Einen Verlauf, der vergleichbar mit der Referenz ist, weist das 1oo3/3oo4 und das 2oo3/2oo4

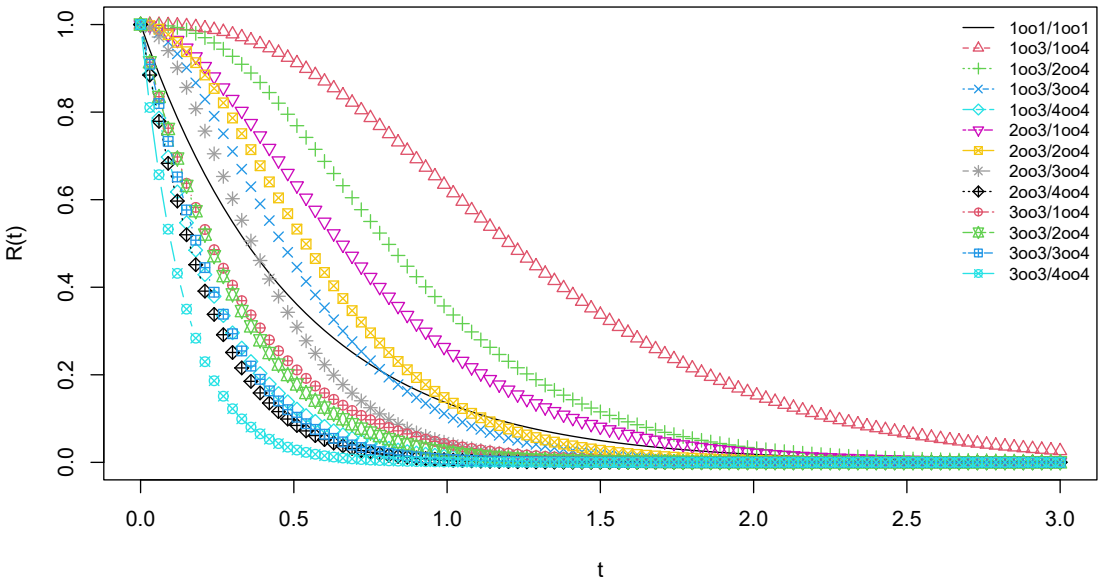


**Abb. 4.4:** Überlebenswahrscheinlichkeit R verschiedener Majoritätsredundanzen (MooN$_S$/MooN$_M$) mit 3 Sensoren und 4 MCUs mit $\lambda_S = 1$ und $\lambda_M = 1$

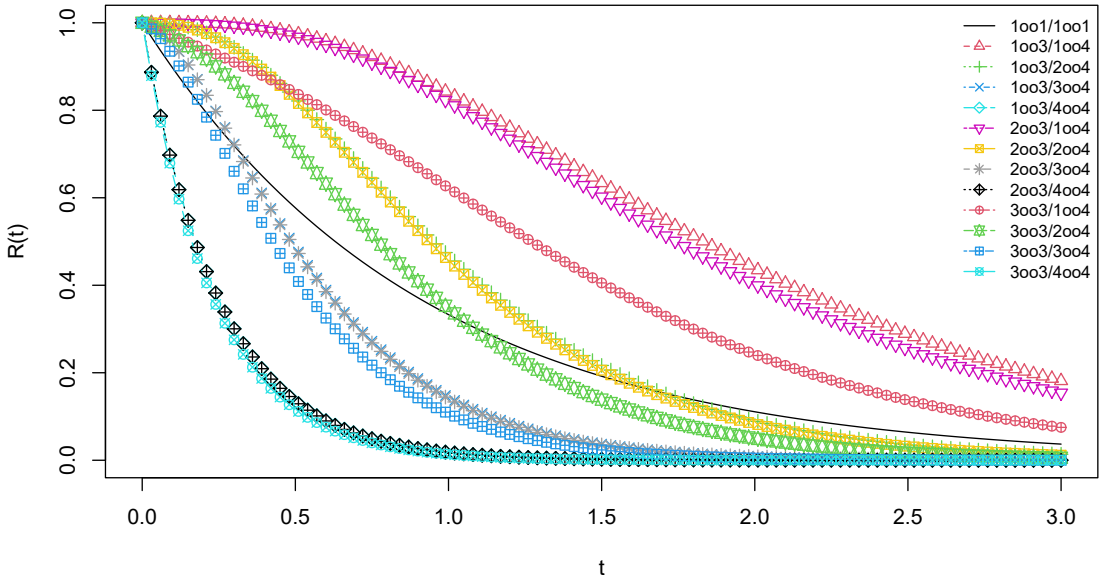


**Abb. 4.5:** Überlebenswahrscheinlichkeit R verschiedener Majoritätsredundanzen (MooN$_S$/MooN$_M$) mit 3 Sensoren und 4 MCUs mit$\lambda_S = 0.1$ und $\lambda_M = 1$

System auf. Deutlich besser ist das 2oo3/1oo4 und 1oo3/2oo4 System. Das 1oo3/1oo4 System liegt mit großem Abstand vorne. Es ist zu erkennen, dass Systeme die Serienanordnungen (NooN) beinhalten oder Systeme dessen Majoritätsredundanz viele übereinstimmende Komponenten (z.B. 3oo4) fordert eine niedrige Zuverlässigkeit aufweist. Dagegen bestehen die Systeme mit hoher Überlebenswahrscheinlichkeit aus Parallelanordnun-

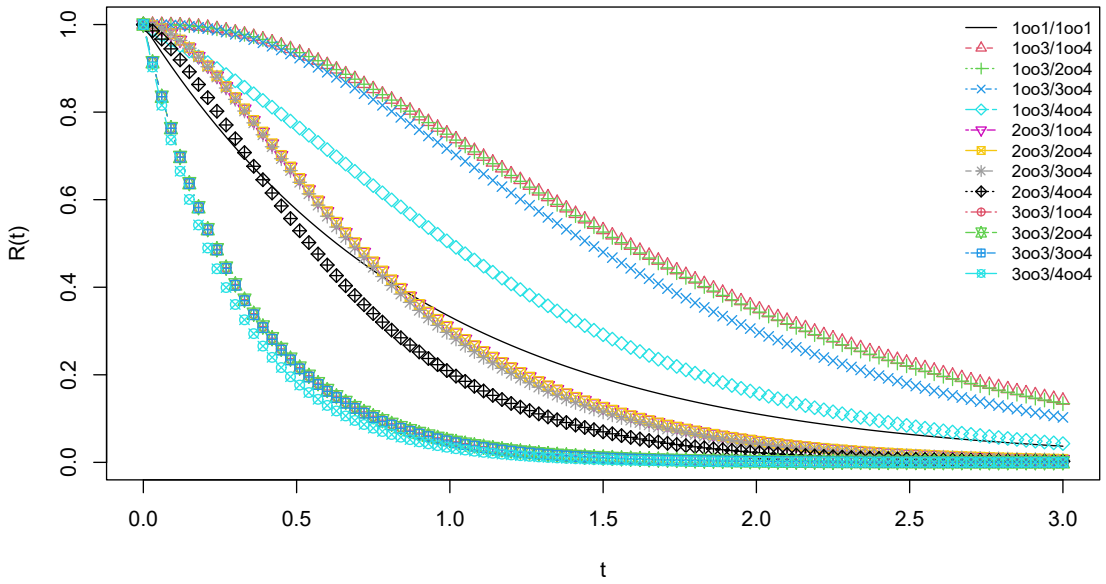


**Abb. 4.6:** Überlebenswahrscheinlichkeit R verschiedener Majoritätsredundanzen (MooN$_S$/MooN$_M$) mit 3 Sensoren und 4 MCUs mit $\lambda_S = 1$ und $\lambda_M = 0.1$

gen (1ooN). Jedoch wäre es falsch diese Parallelsysteme als geeignetste Architekturen anzusehen, da ihre Diagnosefunktion, die für ein Fail-Operational-Verhalten notwendig ist, eingeschränkt ist. Nach dem Schema aus Abbildung 4.2 aus Abschnitt 4.1 können Parallelanordnungen um Diagnosefunktionen erweitert werden. Dafür braucht es jedoch mindestens vier MCUs, die im duo 2oo2 System betrieben werden. Folglich sind Architekturen, die 1ooN System beinhalten ebenfalls nicht geeignet, um Fehlertoleranz zu gewährleisten. Sind die Ausfallraten von Sensoren und MCUs identisch (Abb. 4.4), dann ist das **2oo3/2oo4 System** die ausfallsicherste Variante.

Wird die Ausfallrate der Sensoren reduziert, steigen die Überlebenswahrscheinlichkeiten der System an, dessen Sensorik in einer Serienanordnung vorliegt, während die MCUs nicht in Serie geschaltet sind. In Abbildung 4.5 ist zu sehen, dass v.a. das 2oo3/1oo4, das 3oo3/1oo4 und das 2oo3/2oo4 System dazugewonnen haben. Die ersten beiden müssen aufgrund ihrer reinen Parallelanordnung unberücksichtigt bleiben. Das letzte System stellt jedoch die bereits als geeignete ermittelte 2oo3/2oo4 Architektur dar, wodurch das Ergebnis bestätigt wird. Außerdem ist die Bildung verschiedener Gruppen von Kurven zu erkennen. Bei weiterer Reduzierung der Ausfallrate würde sich dieser Effekt noch verstärken. Die bereits beschriebenen Systemeigenschaften können hierdurch abgeleitet werden. Serienanordnungen weisen geringere Zuverlässigkeiten auf, während Parallelanordnungen höhere zeigen. Interessant sind die Zwischengruppen, in denen die geeigneten Systeme liegen, da ihre Überlebenswahrscheinlichkeit nicht zu gering ist und ihre Struktur weniger reine Parallelanordnungen aufweisen. In Ab-

bildung 4.6 ist die Veränderung der Kurven bei einer Reduzierung der Ausfallrate der MCUs zu sehen. Es kann bestätigt werden, dass im Falle einer Reduzierung der Ausfallrate die Gruppen mit den entsprechenden Serienanordnungen sich nach oben verschieben, wenn die Sensorik dabei nicht auch in Serie geschaltet ist. Am auffälligsten ist, dass sich das 1oo3/3oo4 System stark verbessert hat. Aufgrund der reinen Sensor-Parallelanordnung kann dieses System jedoch nicht verwendet werden. Die Gruppen im oberen Zuverlässigkeitsbereich bestehen ausschließlich aus Architekturen, die reine Parallelsysteme beinhalten. Das führt zur These, dass die Reduzierung der MCU-Ausfallraten weniger sinnvoll ist als die der Sensoren. Die These wird in den folgenden Bewertungen überprüft.

### 4.2.3 Architekturbewertung von zwei Sensoren und vier MCUs

Besteht die Architektur aus zwei Sensoren und vier MCUs, wird ihre Überlebenswahrscheinlichkeit durch die in Abbildung 4.7 ersichtlichen Kurven beschrieben. Da in diesem Fall nur zwei Sensoren vorliegen, müssen die Hälfte der dargestellten Systeme unberücksichtigt bleiben, weil nur die 2oo2 Sensorik den Majoritätsvergleich erlaubt. Alle Kurven die über der 1oo1 Referernz liegen bestehen aus reinen Parallelsystemen, die nicht verwendet werden können. Eine Gruppe ähnelt dem Verlauf der Referenz, auch wenn sie leicht darunter liegt. Teil dieser Gruppe die 2oo2/2oo4 Architektur, die der aus dem vorherigen Abschnitt identifizierten Architektur ähnelt. Sie ist das einzige System in dieser Auswahl, das aufgrund der nicht vorhandenen Parallelität geeignet ist. Wie in

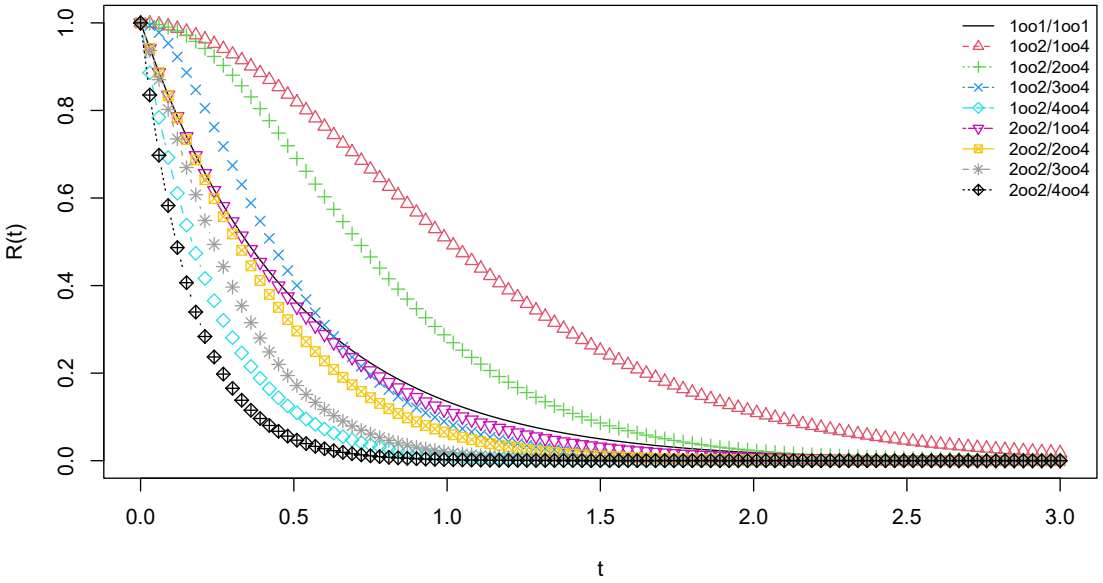


**Abb. 4.7:** Überlebenswahrscheinlichkeit R verschiedener Majoritätsredundanzen (MooN$_S$/MooN$_M$) mit 2 Sensoren und 4 MCUs mit $\lambda_S = 1$ und $\lambda_M = 1$

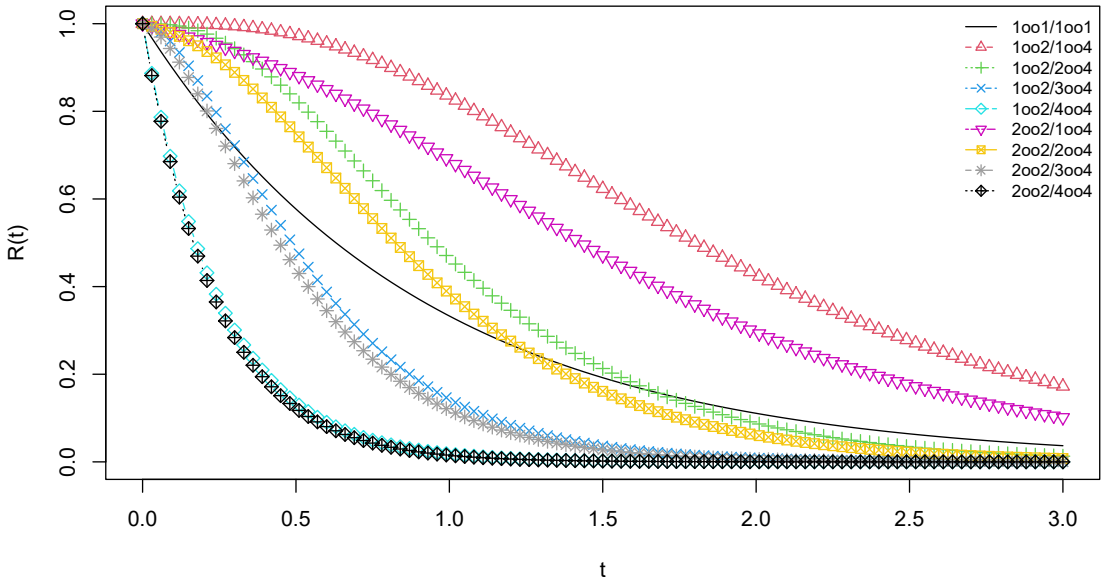


**Abb. 4.8:** Überlebenswahrscheinlichkeit R verschiedener Majoritätsredundanzen (MooN$_S$/MooN$_M$) mit 2 Sensoren und 4 MCUs mit $\lambda_S = 0.1$ und $\lambda_M = 1$

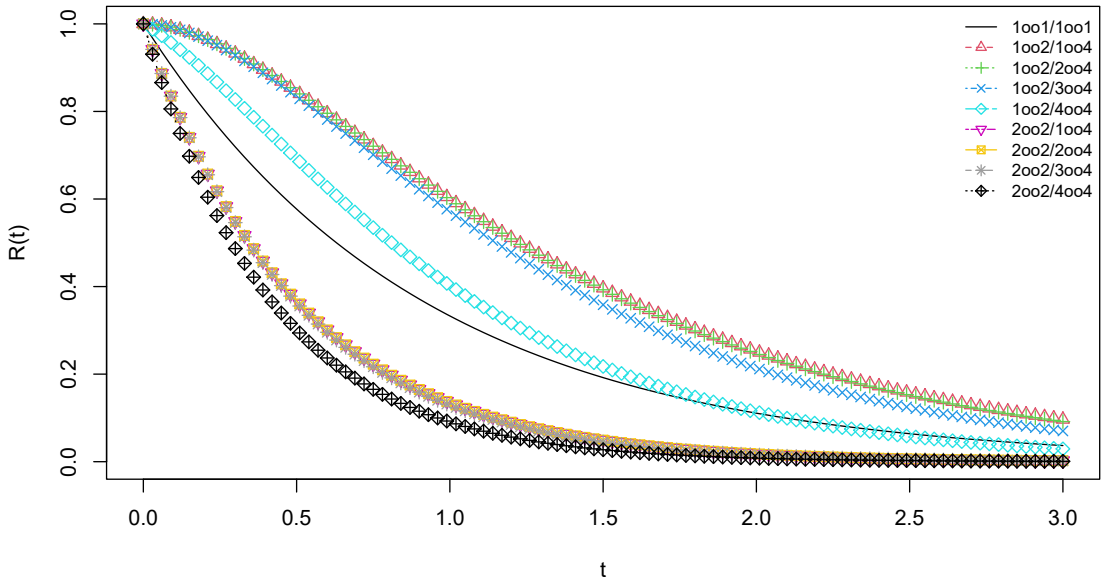


**Abb. 4.9:** Überlebenswahrscheinlichkeit R verschiedener Majoritätsredundanzen (MooN$_S$/MooN$_M$) mit 2 Sensoren und 4 MCUs mit $\lambda_S = 1$ und $\lambda_M = 0.1$

der Abbildung 4.7 zu sehen ist, kommen bei der Betrachtung von zwei Sensoren und vier MCUs generell nur wenige Alternativen in Frage. Es gibt nur ein System, das hier ebenfalls in Frage kommt, das 2oo2/3oo4 System. Die 2oo2/4oo4 Architektur fällt wegen ihrer starken Serienstruktur und damit ihrer geringen Zuverlässigkeit raus. Inwiefern sich die beiden Alternativen bei einer Veränderung der Ausfallraten verhalten, wird in

den folgenden beiden Abbildungen diskutiert. Die Reduzierung der Sensor-Ausfallrate ist in Abbildung 4.8 dargestellt. Zu erkennen ist, dass die Überlebenswahrscheinlichkeit einiger nicht geeigneter Architekturen gestiegen ist. Darunter befindet sich aber auch die geeignete **2oo2/2oo4 Architektur**, die nun über der Zuverlässigkeit der Referenz liegt. Sie kann also als gute Wahl angesehen werden, wenn die Ausfallrate der Sensoren geringer als die Ausfallrate der MCUs ist.

Abbildung 4.9 zeigt die betrachteten Architekturen bei einer Reduzierung der MCU-Ausfallrate. Im Wesentlichen haben sich zwei Gruppen gebildet: Eine Gruppe mit hoher Zuverlässigkeit und eine Gruppe mit niedriger Zuverlässigkeit. Dazwischen liegt das nicht geeignete 1002/4oo4 System. Alle Architekturen der oberen Gruppe bestehen aus reinen Parallelsystemen, wodurch die zuvor aufgestellte These bekräftigt wird, die besagt, dass eine Reduzierung der MCU Ausfallrate weniger sinnvoll ist.

### 4.2.4 Architekturbewertung von drei Sensoren und drei MCUs

Bei drei Sensoren und drei MCUs verhält sich die Zuverlässigkeit der Systeme wie in Abbildung 4.10 dargestellt. Es zeigt sich erneut das bekannte Schema: Seriensysteme liegen unten. Parallelsysteme liegen oben. Dazwischen befinden sich die brauchbaren Architekturen. In diesem Fall handelt es sich um die 2oo3/2oo3 Architektur. Sie entspricht im Durchschnitt der 1oo1 Referenz. Wird die Ausfallrate der Sensoren reduziert, zeigt Abbildung 4.8 erneut eine Verbesserung der Systeme mit starkem Sensor und geringerem MCU Einfluss. Interessant ist, dass das identifizierte 2oo3/2oo3 sich kaum

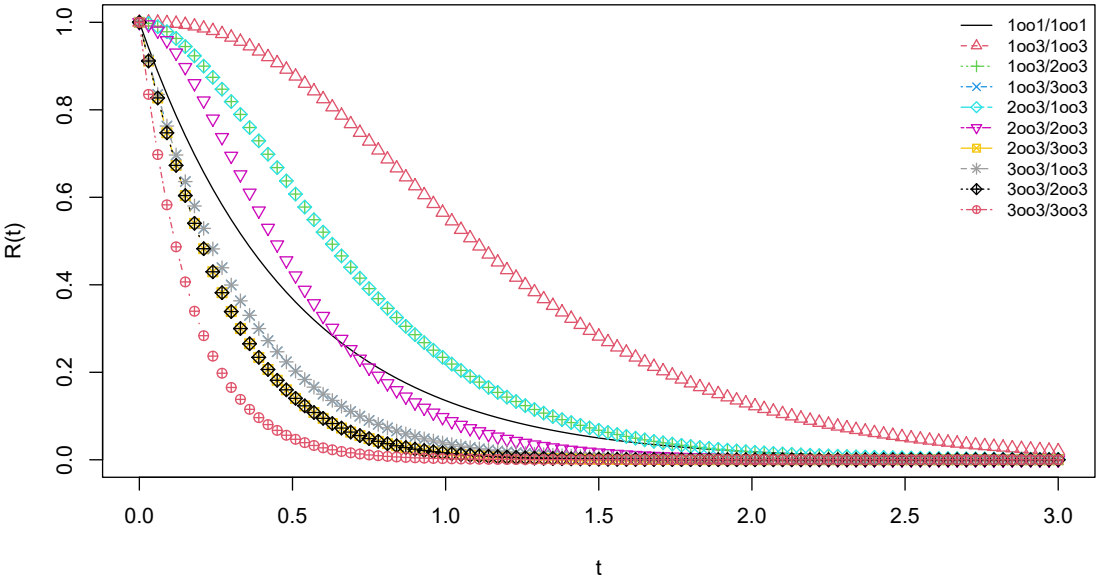


**Abb. 4.10:** Überlebenswahrscheinlichkeit R verschiedener Majoritätsredundanzen (MooN$_S$/MooN$_M$) mit 3 Sensoren und 3 MCUs mit $\lambda_S = 1$ und $\lambda_M = 1$

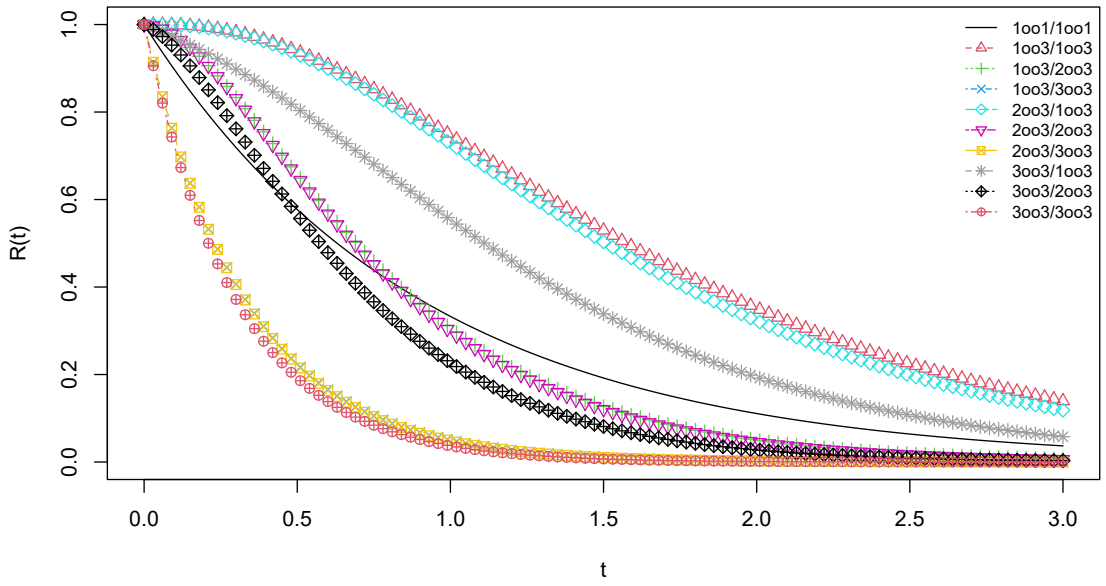


**Abb. 4.11:** Überlebenswahrscheinlichkeit R verschiedener Majoritätsredundanzen ($\text{MooN}_S/\text{MooN}_M$) mit 3 Sensoren und 3 MCUs mit $\lambda_S = 0.1$ und $\lambda_M = 1$

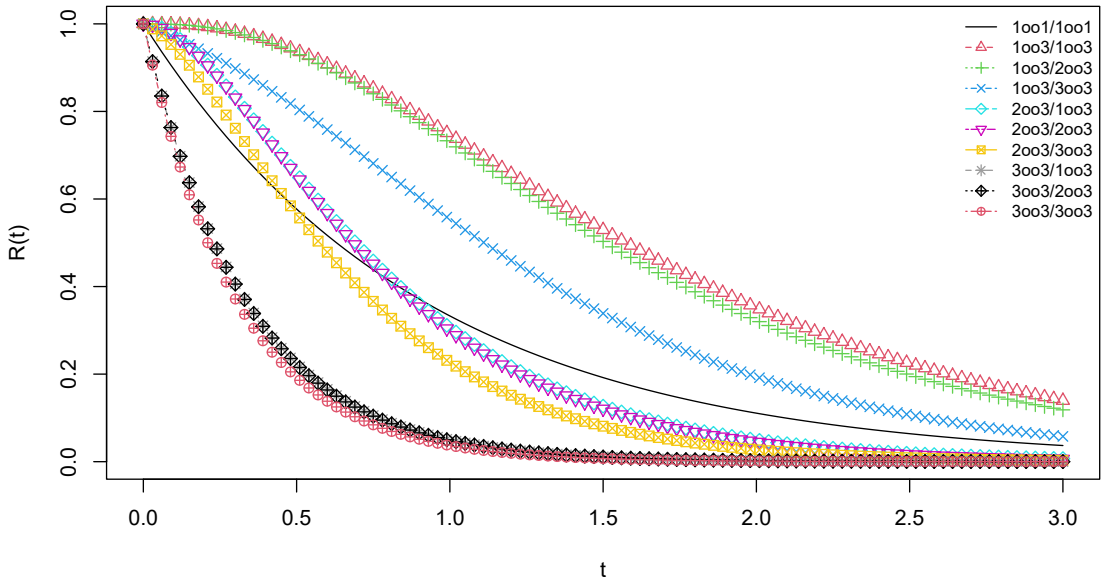


**Abb. 4.12:** Überlebenswahrscheinlichkeit R verschiedener Majoritätsredundanzen ($\text{MooN}_S/\text{MooN}_M$) mit 3 Sensoren und 3 MCUs mit $\lambda_S = 1$ und $\lambda_M = 0.1$

verändert. Die Referenzkurve wird lediglich etwas früher geschnitten. Durch die Symmetrie der Architektur ist dies nicht alleine zu erklären. Derselbe Effekt tritt in Abbildung 4.12 auf, in der die Erhöhung der MCU-Ausfallrate gezeigt wird. Die 2oo3/2oo3 Kurve entspricht hier exakt der Kurve aus der vorherigen Abbildung. Hierfür ist diesmal die Symmetrie verantwortlich zu machen. Für eine Systemarchitektur, bestehend

aus drei Sensoren und drei MCUs ist die 2oo3/2oo3 als am geeignetsten anzusehen. Die Ähnlichkeit des Verlaufs ihrer Überlebenswahrscheinlichkeit im Vergleich zur 1oo1 Referenz bei allen betrachteten Ausfallratenszenarien zeigt jedoch auch, dass die Wahl einer 3-Sensor/3-MCU Architektur weniger sinnvoll ist als die beiden zuvor analysierten Architekturen. Das **2oo3/2oo3 System** ist bezüglich ausfallsicherem Verhalten zwar geeignet, aus zuverlässigkeitstechnischer Sicht stellt es jedoch keinen Vorteil zum 1oo1 System dar. Die Diagnosefunktion über den Mehrheitsvergleich ist natürlich ein Vorteil, den die zuvor identifizierten Systeme jedoch auch hatten aber dafür einen Mehrwert in der Überlebenswahrscheinlichkeit aufweisen konnten.

### 4.2.5 Architekturbewertung von zwei Sensoren und drei MCUs

Werden zwei Sensoren und drei MCUs verwendet, lässt sich die Zuverlässigkeit ihrer Architekturen gemäß Abbildung 4.13 ausdrücken. Aufgrund der geringeren Zahl der möglichen Kombinationen ist der Freiheitsgrad für die Systemauswahl noch stärker eingeschränkt als zuvor. Die denkbaren Alternativen werden diesmal auf zwei Möglichkeiten beschränkt: 2oo2/2oo3 und 2oo2/3oo3, wobei das Letztere angesichts des reinen Seriensystems bezüglich der geringen Zuverlässigkeit ausscheidet. Folglich kann in diesem Schritt nur noch beobachtet werden, wie die 2oo2/2oo3 Architektur im Vergleich zur 1oo1 Referenz abschneidet. In Abbildung 4.13 ist zu erkennen, dass die Zuverlässigkeit der betrachteten Architektur im unteren Bereich und deutlich unter der Referenz liegt.

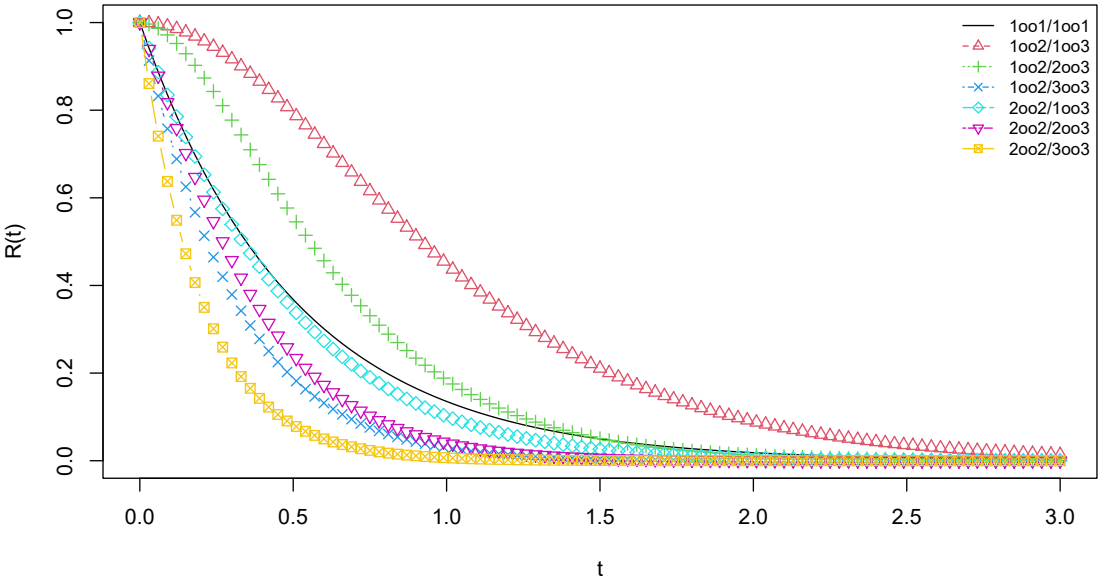


**Abb. 4.13:** Überlebenswahrscheinlichkeit R verschiedener Majoritätsredundanzen (MooN$_S$/MooN$_M$) mit 2 Sensoren und 3 MCUs mit $\lambda_S = 1$ und $\lambda_M = 1$

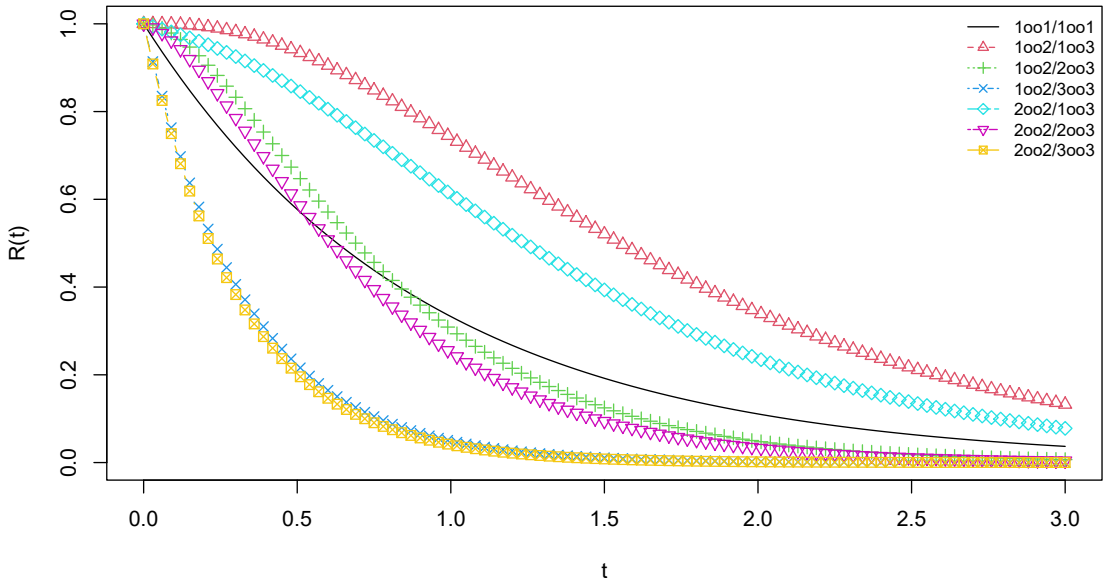


**Abb. 4.14:** Überlebenswahrscheinlichkeit R verschiedener Majoritätsredundanzen (MooN$_S$/MooN$_M$) mit 2 Sensoren und 3 MCUs mit $\lambda_S = 0.1$ und $\lambda_M = 1$

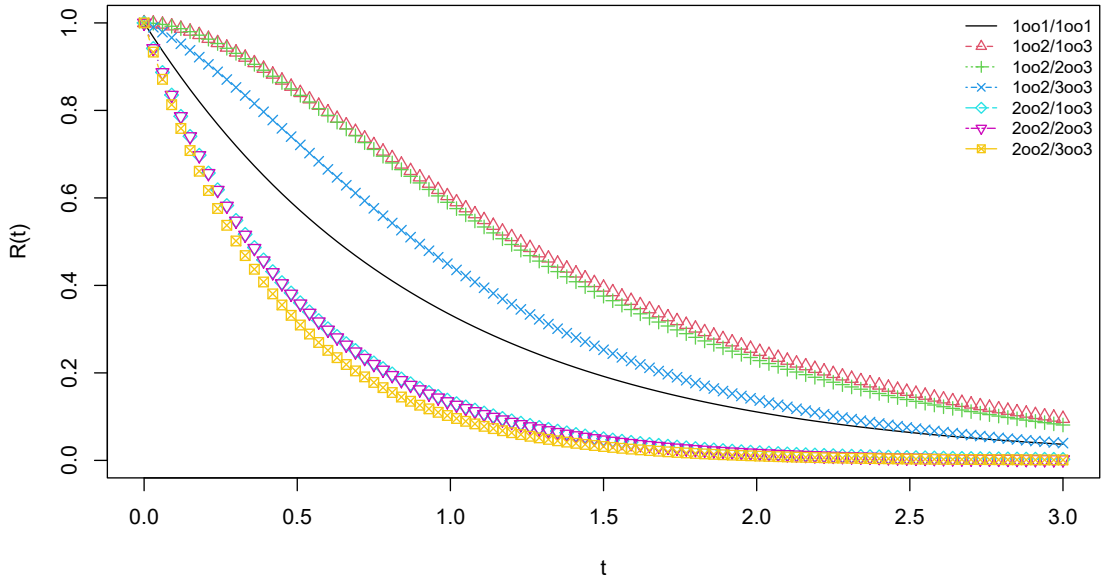


**Abb. 4.15:** Überlebenswahrscheinlichkeiten R verschiedener Majoritätsredundanzen (MooN$_S$/MooN$_M$) mit 2 Sensoren und 3 MCUs mit $\lambda_S = 1$ und $\lambda_M = 0.1$

Wird die Ausfallrate der Sensorik entsprechend Abbildung 4.14 reduziert, verbessert sich die Zuverlässigkeit des **2oo2/2oo3 Systems** zunächst und liegt anfänglich über der Referenzarchitektur. Jedoch wird diese schnell unterschritten, sodass das betrachtete MooN System insgesamt als weniger zuverlässig als die 1oo1 Architektur zu bewerten

ist. Die Reduzierung der Ausfallrate nach Abbildung 4.15 führt zu einem Absturz des 2oo2/2oo3 Systems. Aus zuverlässigkeitstechnischer Sicht ist eine Architektur, die aus zwei Sensoren und drei MCUs besteht nicht zu empfehlen.

### 4.2.6 Zusammenfassung der Ergebnisse der Markovanalyse

Unter den betrachteten Szenarien konnte festgestellt werden, dass eine Systemarchitektur, bestehend aus drei Sensoren und vier MCUs sowie aus zwei Sensoren und vier MCUs, die größten Überlebenswahrscheinlichkeiten aufweisen. Die $S_3M_4$ Variante liefert eine marginal höhere Zuverlässigkeit als das $S_2M_4$ System. Die Verbesserung der Überlebenswahrscheinlichkeit kann insbesondere mit einer 2oo3/2oo4 bzw. 2oo2/2oo4 Architektur erreicht werden, wenn die Ausfallrate der Sensoren geringer als die der MCUs gewählt werden. Bei der Verwendung einer 4-MCU-Architektur spielt die Sensorzahl (zwei oder drei Sensoren) aus zuverlässigkeitstechnischer Sicht also nur eine geringfügige Rolle. Dennoch bleibt die Frage ungeklärt, welche Sensoren für eine fehlertolerante Erfüllung aller potentiellen Fahrszenarien benötigt werden. Die Beantwortung ist nicht Gegenstand dieser Arbeit. Des Weiteren kann beobachtet werden, dass Architekturen mit reinen Parallel-Anordnungen besonders hohe Überlebenswahrscheinlichkeiten erzielen. Solche Architekturen können im fehlertoleranten Einsatz jedoch nicht verwendet werden, da mit ihnen Diagnosefunktionen, wie Lockstep Verfahren und Mehrheitsvergleich, nur schwierig umzusetzen sind. Mit Hilfe einer isolierten Kommunikation in Anlehnung an Abbildung 4.2 aus Abschnitt 4.1 könnten reine Parallelsystem mit einer redundanten Auslegung um eine Diagnose ergänzt werden, indem zwei failsafe Stränge zu einem fail-operational Strang parallel geschaltet werden. Der entstehende Overhead und Kosten sind dafür jedoch nicht zu vertreten. Solche DFS-Architekturen sind nur für wenig komplexe Systeme geeignet. Außerdem ist zu beachten, dass die Markovanalyse nur hinsichtlich der Zuverlässigkeit quantitativ durchgeführt wurde. Weitere Aspekte der Fehlertoleranz wurden lediglich qualitativ beurteilt. Dazu gehören beispielsweise die Eigenschaften der DSF-Architekturen, die mittels Separation und Diversität ein großes Potential zur Erhöhung der Fehlertoleranz ausweisen. Die Ergebnisse sind jedoch nur für die Auslegung der strukturellen Redundanz valide.

## 4.3 Ermittlung der Zuverlässigkeitszielwerte der Komponenten mittels Fehlerbaumanalyse

### 4.3.1 Vorgehensweise der Fehlerbaumanalyse

Mit Hilfe einer Fehlerbaumanalyse wird im Folgenden das Ausfallverhalten eines 2oo3/2oo4 Systems untersucht. Nach dem Vorliegen des Ergebnisses wird überprüft, inwiefern die

Reduzierung der Sensorzahl auf zwei das Ergebnis verändert. Die FTA wird das System detaillierter betrachten. Die Markovanalyse begrenzte die Komplexität auf MCU Ebene. Es werden sämtliche Komponenten modelliert, die eingangs in der Systemdefinition aufgelistet wurden (Abs. 4.1). Da der Vorteil der diversitären Redundanz der DFS Systeme (vgl. Abb. 4.2) gegenüber der Einzel-ECU Systeme (vgl. Abb. 4.3) mit der Markovanalyse nicht herausgestellt werden konnte, wird die FTA jeweils eines der Systemtypen mit einer 2oo3/2oo4 Architektur analysieren. Das 2oo3/2oo4DFS System wird aus den im letzten Abschnitt 4.2.6 erläuterten Gründen jedoch nicht komplett redundant ausgelegt. Verwendet wird ein 2oo3/2oo2DFS System, das dem System aus Abbildung 4.2 sehr ähnlich sein wird. Natürlich entspricht das Ausfallverhalten eines redundant ausgelegten 2oo2 Systems nicht dem Ausfallverhalten eines 2oo4 Systems, auch wenn beide Systeme vier MCUs aufweisen. Außerdem enthält das DFS Systeme eine zusätzliche MCU für die Mehrheitsentscheidung. Durch die Modellierung beider Systeme soll der Vorteil bezüglich diversitärer- und den Nachteil bezüglich struktureller Redundanz der DFS Architektur quantifiziert werden. Dafür wird eine Art Gewichtung in Form eines zusätzlichen Einpunktausfalles verwendet, der Ausfälle aufgrund gemeinsamer Ursachen charakterisiert, denen durch diversitäre Redundanz entgegengewirkt wird. Die Ausfallrate dieses zusätzlichen Events wird bei der Einzel-ECU Architektur größer gewählt als bei der DFS Architektur. Da hierfür keine quantifizierbar belastbaren Daten vorliegen, werden Annahmen getroffen, die im Laufe der Analyse variiert werden. Nach der Modellierung des Fehlerbaums wird im ersten Schritt eine Cut-Set Analyse durchgeführt, um alle Einpunktausfälle zu identifizieren. Die ISO 26262 legt für diese Komponenten gemäß Abschnitt 2.2.5 gesonderte Ausfallraten fest. Die Auslegung der restlichen Komponenten bleibt frei mit dem Ziel die Top-Level Anforderungen aus Tabelle 2.1 (Abs. 2.2.5) zu erfüllen. Auf der Grundlage der Erkenntnisse aus Abschnitt 4.2 der Markovanalyse wird angestrebt die Sensorausfallraten geringer als die MCU-Ausfallraten zu wählen. Den Watchdog-Ausfallraten bekommt eine besondere Bedeutung zuteil. Es handelt sich um Diagnoseeinrichtungen, dessen Diagnosedeckungsgrad berücksichtigt werden muss. Importanzanalysen unterstützen die Identifikation der kritischen Systemelemente.

### 4.3.2 Analyse der 2oo3/2oo4 Einzel-ECU Architektur

Im ersten Schritt wird der Fehlerbaum des betrachteten Systems modelliert. Das Blockschaltbild basiert auf Abbildung 4.3 (Abs. 4.1) mit einer zusätzlichen MCU inklusive des dazugehörigen Voting Cores. Die MCUs, die die Sensorinformationsverarbeitungsstränge beinhalten sind mit MCU1 bis MCU4 beziffert. Der MCU mit den CPU-Kernen für die Mehrheitsentscheidung wird Voting-MCU genannt. Die Kerne, dessen Input durch die MCUs 1 bis 4 geliefert wird, werden ebenfalls mit den Nummern 1 bis 4 versehen. Der CPU-Kern, der dieses Ergebnis durch eine erneute Mehrheitsentscheidung validiert

heißt Voting-Core. Der komplette Fehlerbaum kann aufgrund seiner Größe nicht dargestellt werden. Auf der mitgelieferten CD wird der Baum in seiner Gänze hinterlegt. Abbildung 4.16 zeigt die zweite Ebene, während die unteren Ebenen abgeschnitten sind.

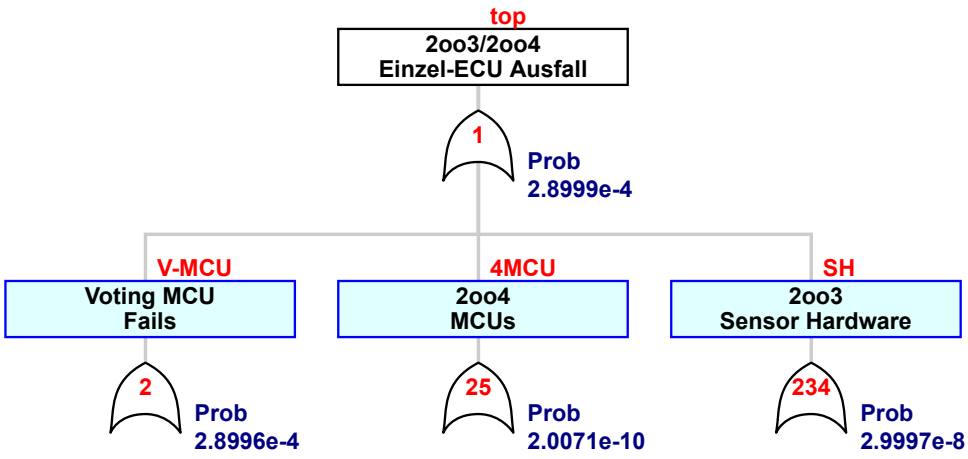


**Abb. 4.16:** Ebene zwei des 2oo3/2oo4 Einzel-ECU Fehlerbaums mit identischen konstanten Ausfallraten

Das System fällt aus, wenn entweder der Voting-MCU ausfällt oder die 2oo4 MCU Mehrheitsentscheidung bzw. die 2oo3 Sensor Hardware Mehrheitsentscheidung nicht erfolgreich ist. Die blau hinterlegten Felder zeigen, an dass die weitere Baumstruktur des entsprechenden Pfades eingeklappt wurde. Die ID der Ereignisse bzw. der Gatter werden durch die roten Nummern ausgedrückt. Für den ersten Schritt ist zunächst für jede Komponente dieselbe konstante Ausfallrate verwendet worden. Da es sich bei einer fehlertoleranten ECU für das fahrerlose Fahren eindeutig um eine ASIL D Klassifizierung handelt, wurde sich an dem Top-Level ASIL D Zielwert der ISO 26262 orientiert, der aus einer Fehlerraten von $10^{-8}$ besteht. Die eingesetzten Fehlerraten sind bei den Basisereignisse grün markiert. Da es sich um konstante Ausfallraten handelt, wird das Exponentialmodell verwendet (in der Abbildung grau hinterlegt). Die simulierte Betriebszeit wird als exposure bezeichnet. In diesem Fall ist sie mit 10.000 h angesetzt. Zusammen mit der Fehlerrate wird daraus die Ausfallwahrscheinlichkeit für den Zeitpunkt t berechnet, die über die Gatter in Richtung Top-Event entwickelt wird. Sie ist in blauer Farbe unter der Bezeichnung Prob markiert. Die Modellierung des Voting-MCUs ist in Abbildung 4.17 zu sehen.

Der MCU für den Voting-Prozess des Outputs der Verarbeitungs-MCUs 1 bis 4 fällt aus, wenn der 2oo4 Mehrheitsvergleich fehl schlägt, der Stufe zwei Voting Core ausfällt, die entsprechende Energieversorgung (Supply) oder der Watchdog ausfällt. Die hier dargestellte Variante der Berücksichtigung des Diagnosedeckungsgrades über den Watchdog ist eine stark vereinfachte Form. Auf Kosten von zunehmender Komplexität

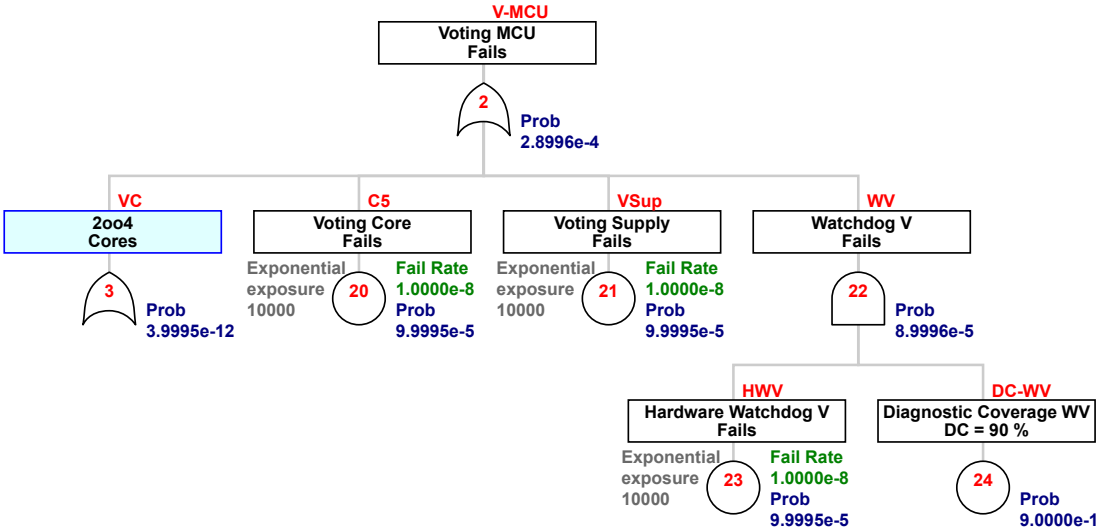


**Abb. 4.17:** Teilssystem Voting-MCU des 2oo3/2oo4 Einzel-ECU Fehlerbaums mit identischen konstanten Ausfallraten

kann, wenn benötigt, eine detailliertere Betrachtung, wie in [37] beschrieben, vorgenommen werden. Die Mehrheitsentscheidung, die hinter Gatter 3 verbogen ist wird im Weiteren näher beschrieben. Sie ist exemplarisch für alle weiteren eingesetzten Majoritätsredundanzen. Das für die Fehlerbaumanalyse verwendete R Framework „FaultTree“ inklusive der Erweiterung „FaultTree.SCRAM“ beinhaltet zwar die Verwendung von Voting-Gattern [38], ihre Berechnung kann die Software jedoch nicht vornehmen, weshalb eine Lösung über ein Parallel-Serien-System zur Modellierung der Mehrheitsentscheidungen implementiert wurde (Abb. 4.18).

Eine 2oo4 Mehrheitsentscheidung kann auch über vier parallel geschaltete Seriensysteme umgesetzt werden. Sobald 3 von 4 CPU Kerne ausfallen, ist das System ausgefallen. Mittels vier Seriensystemen können alle möglichen Kombinationen der Ausfälle modelliert werden. Dabei ist es wichtig, die sich wiederholenden Komponenten als Duplikate zu programmieren, die durch die Farbe magenta markiert sind. Dadurch werden nicht unterschiedliche Komponenten mit dem selben Ausfallverhalten eingesetzt, sondern exakt die selben Komponenten, die überall gleichzeitig und unabhängig vom Seed ausfallen. Die in Abbildung 4.18 zu sehenden S drücken aus, dass es sich um das ursprünglich programmierte Quellelement handelt (source). Die sich darauf beziehenden Duplikate werden mit einem R versehen (Replikant). Die Duplikate mit den entsprechenden R's verbergen sich hinter den blau eingefärbten eingeklappten Feldern.

Auf die in Abbidlung 4.16 angedeutete Modellierung des 2oo4 MCU und des 2oo3 Sensor Hardware Gatter wurde bisher noch nicht eingegangen. Ihr Aufbau folgt demselben zuletzt vorgestellten Parallel-Serien-System Schemas. Die MCUs sind äquivalent

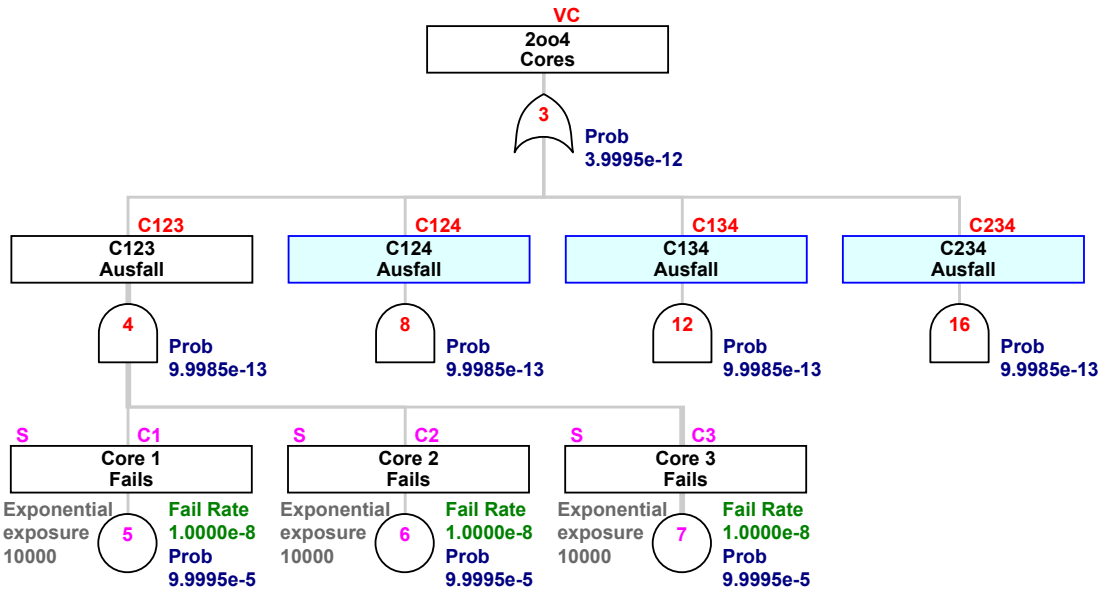


**Abb. 4.18:** Modellierung des Mehrheitsentscheidungsprozesses mittels Parallel-Serien-System

zur Voting MCU modelliert. Der Ausfall liegt vor, wenn das Sensor Fusion Modul, das Processing Modul, die Energieversorgung, der Watchdog oder Input ausfällt. Der Input besteht hierbei jedoch aus dem Monitoring der Sensorinformationen, das gemäß der 2oo3/2oo4 Architektur mittels 2oo3 Entscheidung überwacht wird. Für die 2oo3 Mehrheitsentscheidung werden drei Seriensysteme mit jeweils zwei Komponenten benötigt. Für die Sensorik werden Kamera, Radar und Lidar eingesetzt, was jedoch auch nur eine symbolische Zuordnung ist. Wesentlich ist, dass es sich um drei verschiedenen Sensoren handelt. Um eine 2oo2 Sensorarchitektur umsetzen, müssen an dieser Stelle Änderungen vorgenommen werden, indem ein Sensor entfernt wird. Das 2oo3 Sensor Hardware Gatter wird entsprechend modelliert. Hierbei handelt es sich dann aber um die Hardware der Sensoren und nicht um das Monitoring Modul. Die Modellierung des Fehlerbaums kann detailliert auf der beigelegten CD betrachtet werden.

### Cut-Set Analyse

Die Cut-Set Analyse wird in erster Linie eingesetzt, um alle Einpunktausfälle zu identifizieren, um deren gesonderten Anforderungen implementieren zu können. Für ein besseres Verständnis des Ausfallverhalten des Systems, sind neben diesen Minimalschnitten der ersten Ordnung auch Minimalschnitte der zweiten sowie dritten Ordnung von Bedeutung. Eine Übersicht der Cut-Sets der ersten Ordnung ist in Tabelle 4.1 zu finden. Das System besitzt genau zwei Einpunktausfälle, für die gemäß ISO 26262 die Ausfallrate $\lambda = 10^{-10}\ 1/h$ angesetzt werden muss. Dabei handelt es sich um den Voting-Core

**Tab. 4.1:** Einpunktausfälle (Minimalschnitte erster Ordnung) der Einzel-ECU Architektur

| | |
|---|---|
| **(1)** | C5 |
| **(2)** | VSup |

(C5) und die Voting Supply. Die Minimalschnitte zweiter Ordnung bzw. Zweipunktausfälle sind in Tabelle 4.2 zu sehen.

**Tab. 4.2:** Zweipunktausfälle (Minimalschnitte zweiter Ordnung) der Einzel-ECU Architektur

| | | |
|---|---|---|
| **(1)** | HWV | DC-WV |
| **(2)** | HK | HR |
| **(3)** | HK | HL |
| **(4)** | HR | HL |

Die Hardware der Sensoren (hier beispielhaft Kamera, Radar und Lidar) sowie die Watchdog-Architektur der Voting-MCU führen zu Zweipunktausfällen. Dies bestätigt die Erkenntnis, dass die Sensor-Ausfallrate von größerer Wichtigkeit ist. Im Zusammenhang mit den Cut-Sets der ersten Ordnung kann eine hohe Kritikalität der Voting-MCU erkannt werden. Die Ausfallraten sind entsprechend zu wählen. Tabelle 4.3 zeigt die Minimalschnitte der dritten Ordnung.

**Tab. 4.3:** Dreipunktausfälle (Minimalschnitte dritter Ordnung) der Einzel-ECU Architektur

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **(1)** | C1 | C2 | C3 | **(57)** | Sup1 | Sup2 | P4 |
| **(2)** | C1 | C2 | C4 | **(58)** | Sup1 | Sup2 | Sup4 |
| **(3)** | C1 | C3 | C4 | **(59)** | SF1 | SF3 | SF4 |
| **(4)** | C2 | C3 | C4 | **(60)** | SF1 | SF3 | P4 |
| **(5)** | SF1 | SF2 | SF3 | **(61)** | SF1 | SF3 | Sup4 |
| **(6)** | SF1 | SF2 | P3 | **(62)** | SF1 | P3 | SF4 |
| **(7)** | SF1 | SF2 | Sup3 | **(63)** | SF1 | P3 | P4 |
| **(8)** | SF1 | P2 | SF3 | **(64)** | SF1 | P3 | Sup4 |
| **(9)** | SF1 | P2 | P3 | **(65)** | SF1 | Sup3 | SF4 |
| **(10)** | SF1 | P2 | Sup3 | **(66)** | SF1 | Sup3 | P4 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **(11)** | SF1 | Sup2 | SF3 | **(67)** | SF1 | Sup3 | Sup4 |
| **(12)** | SF1 | Sup2 | P3 | **(68)** | P1 | SF3 | SF4 |
| **(13)** | SF1 | Sup2 | Sup3 | **(69)** | P1 | SF3 | P4 |
| **(14)** | P1 | SF2 | SF3 | **(70)** | P1 | SF3 | Sup4 |
| **(15)** | P1 | SF2 | P3 | **(71)** | P1 | P3 | SF4 |
| **(16)** | P1 | SF2 | Sup3 | **(72)** | P1 | P3 | P4 |
| **(17)** | P1 | P2 | SF3 | **(73)** | P1 | P3 | Sup4 |
| **(18)** | P1 | P2 | P3 | **(74)** | P1 | Sup3 | SF4 |
| **(19)** | P1 | P2 | Sup3 | **(75)** | P1 | Sup3 | P4 |
| **(20)** | P1 | Sup2 | SF3 | **(76)** | P1 | Sup3 | Sup4 |
| **(21)** | P1 | Sup2 | P3 | **(77)** | Sup1 | SF3 | SF4 |
| **(22)** | P1 | Sup2 | Sup3 | **(78)** | Sup1 | SF3 | P4 |
| **(23)** | Sup1 | SF2 | SF3 | **(79)** | Sup1 | SF3 | Sup4 |
| **(24)** | Sup1 | SF2 | P3 | **(80)** | Sup1 | P3 | SF4 |
| **(25)** | Sup1 | SF2 | Sup3 | **(81)** | Sup1 | P3 | P4 |
| **(26)** | Sup1 | P2 | SF3 | **(82)** | Sup1 | P3 | Sup4 |
| **(27)** | Sup1 | P2 | P3 | **(83)** | Sup1 | Sup3 | SF4 |
| **(28)** | Sup1 | P2 | Sup3 | **(84)** | Sup1 | Sup3 | P4 |
| **(29)** | Sup1 | Sup2 | SF3 | **(85)** | Sup1 | Sup3 | Sup4 |
| **(30)** | Sup1 | Sup2 | P3 | **(86)** | SF2 | SF3 | SF4 |
| **(31)** | Sup1 | Sup2 | Sup3 | **(87)** | SF2 | SF3 | P4 |
| **(32)** | SF1 | SF2 | SF4 | **(88)** | SF2 | SF3 | Sup4 |
| **(33)** | SF1 | SF2 | P4 | **(89)** | SF2 | P3 | SF4 |
| **(34)** | SF1 | SF2 | Sup4 | **(90)** | SF2 | P3 | P4 |
| **(35)** | SF1 | P2 | SF4 | **(91)** | SF2 | P3 | Sup4 |
| **(36)** | SF1 | P2 | P4 | **(92)** | SF2 | Sup3 | SF4 |
| **(37)** | SF1 | P2 | Sup4 | **(93)** | SF2 | Sup3 | P4 |
| **(38)** | SF1 | Sup2 | SF4 | **(94)** | SF2 | Sup3 | Sup4 |
| **(39)** | SF1 | Sup2 | P4 | **(95)** | P2 | SF3 | SF4 |
| **(40)** | SF1 | Sup2 | Sup4 | **(96)** | P2 | SF3 | P4 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **(41)** | P1 | SF2 | SF4 | **(97)** | P2 | SF3 | Sup4 |
| **(42)** | P1 | SF2 | P4 | **(98)** | P2 | P3 | SF4 |
| **(43)** | P1 | SF2 | Sup4 | **(99)** | P2 | P3 | P4 |
| **(44)** | P1 | P2 | SF4 | **(100)** | P2 | P3 | Sup4 |
| **(45)** | P1 | P2 | P4 | **(101)** | P2 | Sup3 | SF4 |
| **(46)** | P1 | P2 | Sup4 | **(102)** | P2 | Sup3 | P4 |
| **(47)** | P1 | Sup2 | SF4 | **(103)** | P2 | Sup3 | Sup4 |
| **(48)** | P1 | Sup2 | P4 | **(104)** | Sup2 | SF3 | SF4 |
| **(49)** | P1 | Sup2 | Sup4 | **(105)** | Sup2 | SF3 | P4 |
| **(50)** | Sup1 | SF2 | SF4 | **(106)** | Sup2 | SF3 | Sup4 |
| **(51)** | Sup1 | SF2 | P4 | **(107)** | Sup2 | P3 | SF4 |
| **(52)** | Sup1 | SF2 | Sup4 | **(108)** | Sup2 | P3 | P4 |
| **(53)** | Sup1 | P2 | SF4 | **(109)** | Sup2 | P3 | Sup4 |
| **(54)** | Sup1 | P2 | P4 | **(110)** | Sup2 | Sup3 | SF4 |
| **(55)** | Sup1 | P2 | Sup4 | **(111)** | Sup2 | Sup3 | P4 |
| **(56)** | Sup1 | Sup2 | SF4 | **(112)** | Sup2 | Sup3 | Sup4 |

Die Minimalschnitte höhere Ordnungen nehmen in ihrer Anzahl stark zu. Gleichzeitig nimmt ihr Einfluss auf das Gesamtsystem jedoch ab. Die Cut-Sets vierter Ordnung des Systems belaufen sich bereits auf eine Zahl von 432, weshalb sie hier nicht mehr aufgelistet werden. Tabelle 4.3 zeigt, dass es sich bei den Dreipunktausfällen, um Ausfälle der MCU-Komponenten handelt. Auch hierdurch kann bestätigt werden, dass die Annahme des geringeren Einflusses der MCUs gegenüber der Sensorik auf das Gesamtsystem korrekt ist. Zu den dritte Ordnung Cut-Sets gehören jedoch auch die CPU-Kerne des Voting-MCUs (Cut-Set 1 bis 4), dessen Ausfallrate entsprechend gewählt werden kann.

### Auslegung der Komponenten

Die Komponenten werden im Folgenden hinsichtlich ihrer Ausfallrate ausgelegt. Die Einpunktausfälle müssen mit einer Ausfallrate von $\lambda = 10^{-10}\ 1/h$ beauflagt werden. Für den Großtteil der Komponenten, die in den Dreipunktausfällen wiederzufinden sind, wird zunächst der Zielwert für das Gesamtsystem der ISO 26262 von $\lambda = 10^{-8}\ 1/h$ verwendet. Für die Sensorik und die anderen Zweipunktausfälle wird ein Lambda von $10^{-9}\ 1/h$ verwendet, um den bisherigen Erkenntnissen gerecht zu werden. Die Ausfall-

rate des Gesamtsystems ist für den angepassten Fehlerbaum in Abbildung 4.19 dargestellt.

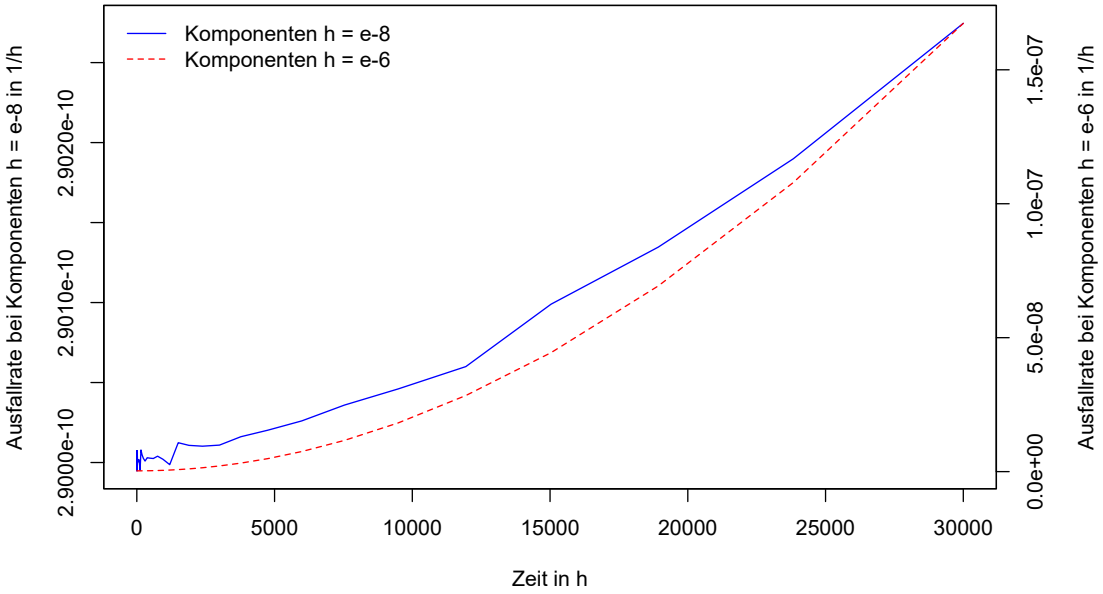


**Abb. 4.19:** Systemausfallrate bei einer Ausfallrate der Komponenten dritter Ordnung von $\lambda = 10^{-8}\ 1/h$ und $\lambda = 10^{-6}\ 1/h$

Die Ausfallrate des Gesamtsystems wird nach den Gleichungen 4.2 und 4.3 berechnet.

$$F(t) = 1 - e^{-\lambda t} \tag{4.2}$$

$$\lambda = -\frac{ln(1 - F(t))}{t} \tag{4.3}$$

Gleichung 4.2 beschreibt die exponentialverteilte Ausfallwahrscheinlichkeit. Im Fehlerbaum ist sie blau markiert unter „Prob“angeben. Wird die Gleichung nach Lambda umgestellt (Gl. 4.3) kann aus dem Top-Event des Fehlerbaums die Ausfallrate berechnet werden. Zunächst ist bei der Modellierung aufgefallen, dass die Hardware des Voting-Watchdogs einen weiteren Flaschenhals darstellt, weshalb sie mit einer Ausfallrate von $\lambda = 10^{-10}\ 1/h$ bemessen. Die blaue Kurve zeigt den Verlauf der Ausfallrate des Systems, wenn die Komponenten der dritten Ordnung mit $\lambda = 10^{-8}\ 1/h$ ausfallen für den Zeitraum bis 30000 Stunden. Zu erkennen ist, dass der Zielwert von $10^{-8}$ deutlich eingehalten wird. Wenn die linksseitige der blauen Kurve zugehörigen Achse betrachtet wird, ist zu sehen, dass die Ausfallrate nahezu konstant ist. Wird die Komponenten Ausfallrate jedoch erhöht (rote Kurve) verliert sie ihre Konstanz. Die Verwendung von einer Vielzahl von konstanten Ausfallrate ist im Ergebnis also nicht konstant. Die ISO 26262 gibt als Zielwert jedoch nur einen konstanten Wert an. Folglich ist die Frage zu klären, für welchen Zeitpunkt die Systemausfallrate betrachtet werden muss. Hierfür kann ei-

ne Statistik zu den gefahrenen Kilometern pro Jahr verwendet werden [39]. Auf dieser Grundlage wird eine jährliche gefahrene Kilometerzahl von 15.000 km angenommen. Wird dazu eine konservativ geschätzte Durchschnittsgeschwindigkeit von 80 km/h auf allen Straßentypen kombiniert verwendet, ist das Ergebnis eine Fahrzeit von jährlichen 187 Stunden, was bei einer Betriebsdauer von 10.000 Stunden 53,5 Jahre ergibt. Diese Stundenzahl wird im Folgenden als Referenzwert verwendet. Bei der Komponenten Ausfallrate von $\lambda = 10^{-6}$ $1/h$ entspricht die Systemausfallrate bei 10.000 Stunden $\lambda = 2{,}01 \cdot 10^{-8}$ $1/h$. Der Zielwert von $10^{-8}$ $1/h$ wird also noch nicht eingehalten. Bei einer Komponenten Ausfallrate von $\lambda = 10^{-6{,}2}$ $1/h$ wird der Zielwert eingehalten. Die Systemsausfallrate beträgt dann $5{,}33 \cdot 10^{-9}$ $1/h$. Der Fehlerbaum bis zur zweiten Ebene sieht mit den angepassten Werten wie folgt aus (Abb. 4.20).

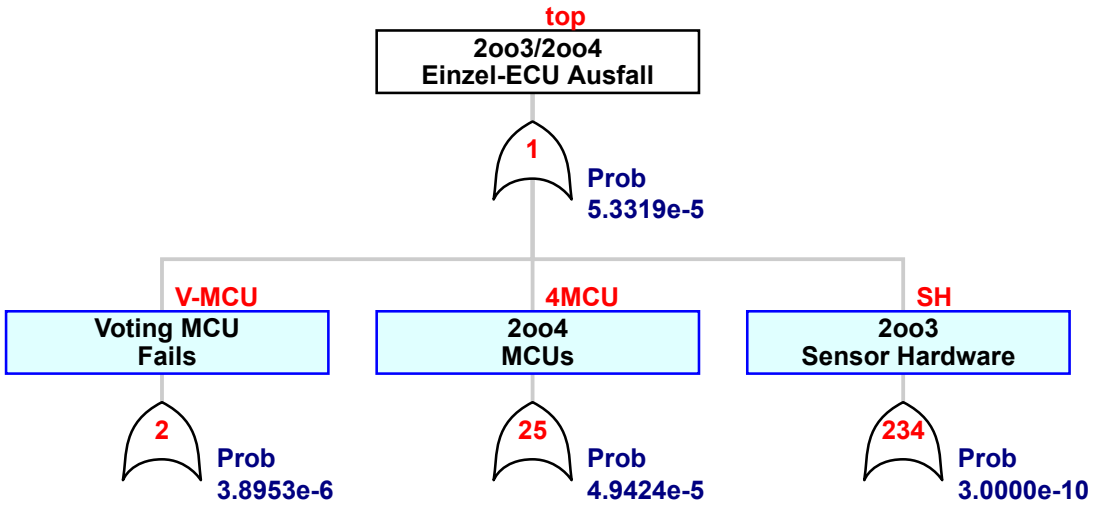


**Abb. 4.20:** Ebene zwei des 2oo3/2oo4 Einzel-ECU Fehlerbaums mit angepassten Ausfallraten

Zu sehen ist, dass der Teilbaum der Sensorhardware in seiner Auslegung noch angepasst werden, ohne die Top-Level Zuverlässigkeit zu reduzieren, da dessen Ausfallwahrscheinlichkeit im Vergleich zu den anderen Ästen viel geringer ist. Die Ausfallrate der Sensorhardware kann auf $10^{-7}$ $1/h$ erhöht werden, um die Ausfallwahrscheinlichkeiten anzugleichen. Der Teilbaum unter der ID 234 (siehe Abb. 4.20) fällt dann mit einer Wahrscheinlichkeit von $3 \cdot 10^{-6}$ % aus, wodurch die Ausfallrate des Gesamtsystems, die dann $5{,}6 \cdot 10^{-9}$ $1/h$ beträgt, unwesentlich verändert wird.

### 4.3.3 Analyse der 2oo3/2oo2 DFS Architektur

Der Aufbau des 2oo3/2oo2 DFS Fehlerbaus bedient sich derselben Modellierung der Teilsysteme, die im vorherigen Abschnitt erläutert wurden. Ihre Anordnung unterscheidet

sich lediglich. Die Top-Level Struktur der betrachteten DFS Architektur ist in Abbildung 4.21 dargestellt.

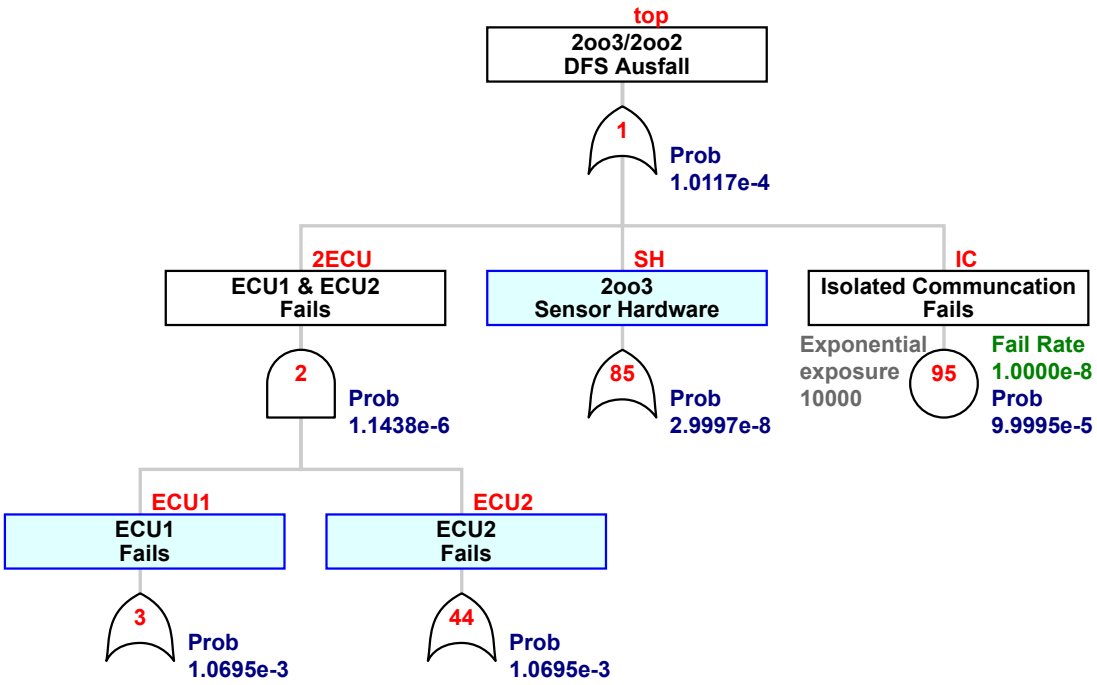


**Abb. 4.21:** Ebene zwei des 2oo3/2oo2 DFS Fehlerbaums mit konstanten identischen Ausfallraten

Die vier MCUs werden auf zwei ECUs verteilt, die parallel Redundant ausgelegt sind. Der 2oo3 Sensorhardware Strang entspricht exakt der Modellierung des Einzel-ECU Systems. Neu bei der DFS Architektur ist der Einsatz einer isolierten Kommunikation (ID 95), die bereits jetzt ersichtlich einen Schwachpunkt in der Architektur darstellt. Der Aufbau einer ECU ist in Abbildung 4.22 verdeutlicht.

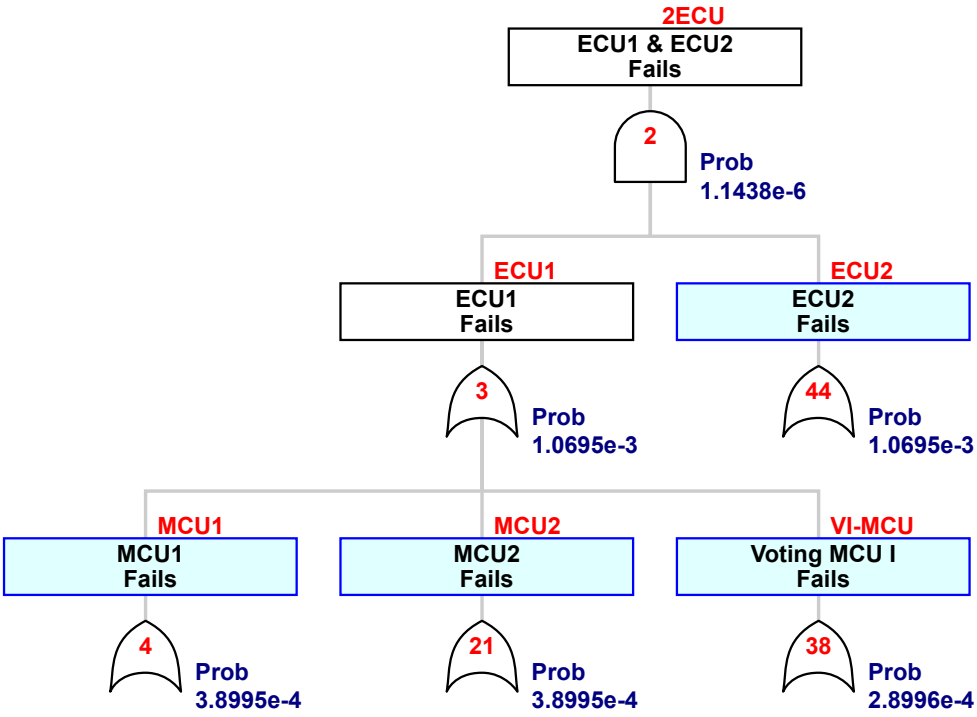


**Abb. 4.22:** Aufbau einer ECU der DFS Architektur

Die ECUs bestehen aus zwei MCUs im 2oo2 Mehrheitsvergleich, wodurch beide MCUs funktionieren müssen, damit das System betriebsbereit ist. Die MCUs haben denselben Aufbau wie die der Einzel-ECU Architektur. Eine Besonderheit der DFS Architektur besteht darin, dass zwei Voting-MCUs verwendet werden, jeweils eine in jeder ECU.

### Cut-Set Analyse

In Tabelle 4.4 ist zu sehen, dass das DFS System nur einen Minimalschnitt besitzt. Die Ursache dafür liegt in der Voting-MCU, die beim Einzel-ECU System die erste Ordnung Cut-Sets ausgelöst haben. Innerhalb der DFS Architektur existieren zwei redundant ausgelegte Voting-MCUs, jeweils in einer ECU.

**Tab. 4.4:** Einpunktausfälle (Minimalschnitte erster Ordnung) der DFS Architektur

| | |
|---|---|
| **(1)** | IC |

Das führt dazu, dass diese MCUs keine Einpunktausfälle verursachen können. Gleichzeitig führt das jedoch auch dazu, dass es eine große Anzahl von Cut-Sets der zweiten Ordnung gibt (Tab. 4.5).

**Tab. 4.5:** Zweipunktausfälle (Minimalschnitte zweiter Ordnung) der DFS Architektu

| | | |
|---|---|---|
| **(1)** | HK | HR |
| **(2)** | HK | HL |
| **(3)** | HR | HL |
| **(4)** | SF1 | SF3 |
| **(5)** | SF1 | P3 |
| **(6)** | SF1 | Sup3 |
| **(7)** | P1 | SF3 |
| **(8)** | P1 | P3 |
| **(9)** | P1 | Sup3 |
| **(10)** | Sup1 | SF3 |
| **(11)** | Sup1 | P3 |
| **(12)** | Sup1 | Sup3 |
| **(13)** | SF1 | SF4 |
| **(14)** | SF1 | P4 |
| **(15)** | SF1 | Sup4 |
| **(16)** | P1 | SF4 |
| **(17)** | P1 | P4 |
| **(18)** | P1 | Sup4 |
| **(19)** | Sup1 | SF4 |
| **(20)** | Sup1 | P4 |
| **(21)** | Sup1 | Sup4 |
| **(22)** | SF1 | Core II |
| **(23)** | SF1 | VII-Sup |
| **(24)** | P1 | Core II |
| **(25)** | P1 | VII-Sup |
| **(26)** | Sup1 | Core II |
| **(27)** | Sup1 | VII-Sup |
| **(28)** | SF2 | SF3 |
| **(29)** | SF2 | P3 |

| | | |
|---|---|---|
| **(30)** | SF2 | Sup3 |
| **(31)** | P2 | SF3 |
| **(32)** | P2 | P3 |
| **(33)** | P2 | Sup3 |
| **(34)** | Sup2 | SF3 |
| **(35)** | Sup2 | P3 |
| **(36)** | Sup2 | Sup3 |
| **(37)** | SF2 | SF4 |
| **(38)** | SF2 | P4 |
| **(39)** | SF2 | Sup4 |
| **(40)** | P2 | SF4 |
| **(41)** | P2 | P4 |
| **(42)** | P2 | Sup4 |
| **(43)** | Sup2 | SF4 |
| **(44)** | Sup2 | P4 |
| **(45)** | Sup2 | Sup4 |
| **(46)** | SF2 | Core II |
| **(47)** | SF2 | VII-Sup |
| **(48)** | P2 | Core II |
| **(49)** | P2 | VII-Sup |
| **(50)** | Sup2 | Core II |
| **(51)** | Sup2 | VII-Sup |
| **(52)** | Core I | SF3 |
| **(53)** | Core I | P3 |
| **(54)** | Core I | Sup3 |
| **(55)** | VI-Sup | SF3 |
| **(56)** | VI-Sup | P3 |
| **(57)** | I-Sup | Sup3 |
| **(58)** | Core I | SF4 |
| **(59)** | Core I | P4 |

| | | |
|---|---|---|
| **(60)** | Core I | Sup4 |
| **(61)** | VI-Sup | SF4 |
| **(62)** | VI-Sup | P4 |
| **(63)** | VI-Sup | Sup4 |
| **(64)** | Core I | Core II |
| **(65)** | Core I | VII-Sup |
| **(66)** | VI-Sup | Core II |
| **(67)** | VI-Sup | VII-Sup |

Hierunter fallen die meisten MCU-Komponenten sowie die Sensorhardware. Die Cut-Sets dritter Ordnung umfassen 144 Kombinationen, weshalb die hier nicht mehr aufgeführt werden. Sie umfassen die Watchdogs sowie die Monitoring Einrichtungen. Die in den Tabellen mit römisch I und II markierten Komponenten bezeichnen die Komponenten des ersten und zweiten Voting-MCUs.

### Auslegung der Komponenten

Die Eigenschaften der DFS Architektur führen bei der Auslegung der Komponenten dazu, dass die Erkenntnis aus Abschnitt 4.2 nicht mehr angewendet werden kann. Es wurde ermittelt, dass die Ausfallrate der Sensorhardware im Verhältnis zur MCU-Ausfallrate geringer zu wählen ist. In der DFS Architektur ist die Sensorik jedoch nicht mehr alleine für die Zweipunktausfälle verantwortlich. Jetzt ist eine Gruppe von zweite und dritte Ordnung Komponenten zu unterscheiden. Der einzige Einpunktausfall, bestehend aus der isolierten Kommunikation wird mit einer Ausfallrate von $\lambda = 10^{-10}\ 1/h$ gemäß ISO 26262 versehen. Die Zweipunktausfälle werden zunächst mit $\lambda = 10^{-9}\ 1/h$ und die Dreipunktausfälle mit $\lambda = 10^{-8}\ 1/h$ beaufschlagt. Die Ausfallrate für das Gesamtsystems ist in Abbildung 4.23 dargestellt.

Für das DFS System ist es nicht sinnvoll die Ausfälle dritter und zweiter Ordnung getrennt voneinander zu betrachten, da zum Ausfall jedes Teilsystems beide Ausfallarten hineinspielen. Bei der Einzel-ECU Architektur war es so, dass die Sensorhardware ausschließlich durch Zweipunktausfälle und die MCUs ausschließlich durch Dreipunktausfälle ausgefallen sind. Die Ausfallrate wird im Folgenden also kombiniert angepasst. Zuvor hat sich ein Delta zwischen den beiden Ordnungen um den Faktor ca. 10 bewährt. Außerdem wurde erkannt, dass die Hardware aller Watchdogs Systemschwachstellen darstellen, weshalb sie die Ausfallrate für Ausfalle der zweiten Ordnung erhalten, obwohl sie eigentlich der dritten Ordnung zuzuschreiben sind. Die Ursache ist der kon-

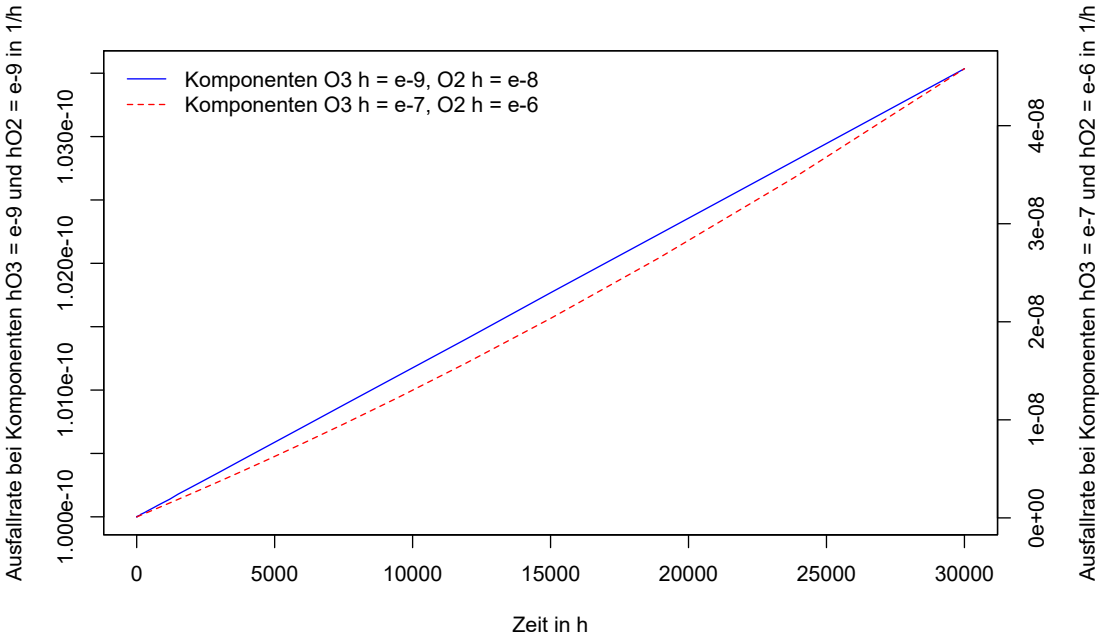


**Abb. 4.23:** Systemausfallrate DFS bei einer Ausfallrate der Komponenten dritter Ordnung von $\lambda = 10^{-9}\ 1/h$ und zweiter Ordnung von $\lambda = 10^{-8}\ 1/h$ sowie dritter Ordnung von $\lambda = 10^{-7}\ 1/h$ und zweiter Ordnung von $\lambda = 10^{-6}\ 1/h$

stante Diagnosedeckungsgrad. Die blaue Kurve aus Abbildung 4.23 zeigt die Ausfallrate des Gesamtsystems, wenn für die Komponenten dritter Ordnung ein Lambda von $10^{-9}\ 1/h$ und für die Komponenten zweiter Ordnung ein Lambda von $10^{-8}\ 1/h$ gewählt wird. Zu sehen ist, dass der Zielwert deutlich eingehalten wird und die rate in etwa konstant ist. Wird die Ausfallrate angepasst, sodass Ausfalle dritter Ordnung die Ausfallrate $10^{-7}\ 1/h$ und die der zweiten Ordnung die Ausfallrate $10^{-6}$ besitzen (rote Kurve mit rechtsseitiger Achse), dann verliert Die Gesamtausfallrate erneut ihre Konstanz. Bei $t = 10.000\ h$ erreicht die Systemausfallrate einen Wert von $1,3 \cdot 10^{-8}\ 1/h$ und erfüllt somit knapp nicht die Anforderung der ISO 26262. Das kann behoben werden, indem die Ausfallrate der zweiten Ordnung auf $\lambda = 10^{-7,2}\ 1/h$ reduziert wird. Das Gesamtsystem erreicht dann eine Ausfallrate von $5,6 \cdot 10^{-9}\ 1/h$. Die oberen beiden Ebenen des angepassten Fehlerbaums sind in Abbildung 4.24 zu sehen.

Es ist zu sehen, dass das Ausfallverhalten des Systems durch die Ausfallwahrscheinlichkeit der ECUs beschränkt wird. Hierbei ist aber auch zu berücksichtigen, dass die ECUs aus Komplexitätsgründen aktiv redundant ausgelegt wurden, wodurch auch eine worst-case Betrachtung umgesetzt wurde. DFS Systeme werden normalerweise über eine Standby Redundanz betrieben. Fällt ECU 1 aus, wird ECU 2 aktiviert. Davor befindet sie sich in einem Ruhemodus, wodurch die Lebensdauer im Vergleich zum aktiven Betrieb erhöht wird. Eine DFS Architektur, dessen ECUs passiv redun-

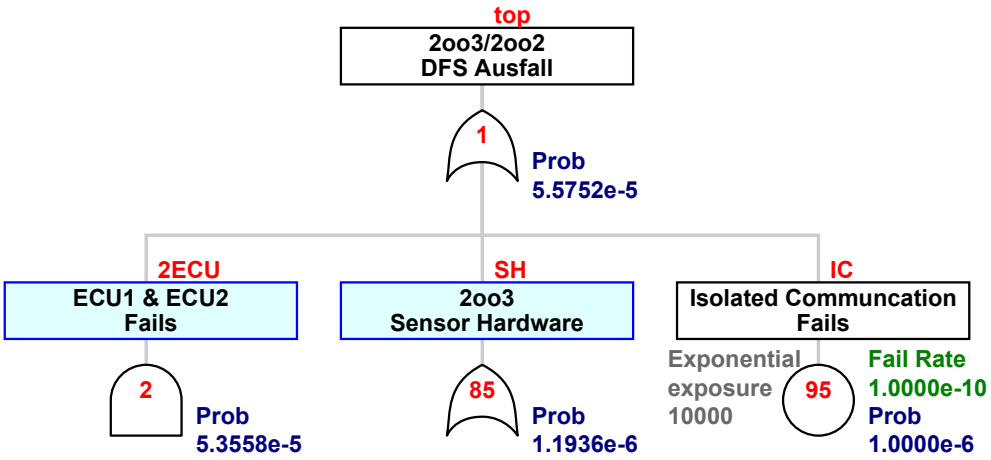


**Abb. 4.24:** Fehlerbaum der DFS Architektur bis zur Ebene zwei bei angepassten Komponenten Ausfallraten

dant ausgelegt sind, könnte nicht mehr durch die ECUs beschränkt sein. Hierbei handelt es sich jedoch nur um eine These, die in weiterführenden Arbeiten bestätigt werden muss. Die Ausfallrate der isolierten Kommunikation ist in Anbetracht ihrer Einpunktausfallseigenschaft sehr gering angesetzt. Durch eine redundante Auslegung und somit der Verhinderung eines Einpunktausfalles kann ihre Ausfallrate reduziert werden.

### 4.3.4 Zusammenfassung Ergebnisse der Fehlerbaumanalyse

Zur Erreichung des ASIL D Zielwerts von $\lambda = 10^{-8}\ 1/h$ wurden die Komponenten der Einzel-ECU und DFS Architektur hinsichtlich ihrer Ausfallrate ausgelegt. Blockschaltbilder die die Notation der Komponenten beschreiben sind in Abschnitt 4.1 zu finden. Die Komponenten wurden gemäß den Minimalschnitten in drei Ordnungen eingeteilt, für die verschiedene Ausfallraten festgelegt worden. Tabelle 4.6 zeigt die Auslegung der Komponenten für die Einzel-ECU Architektur.

Zu den Komponenten erster Ordnung, die Einpunktausfälle verursachen gehört der CPU Kern der Voting MCU, ihre Energieversorgung sowie die Hardware des Watchdogs, der die Voting MCU überwacht. Die ISO 26262 verlangt für Komponenten die zu Einpunktausfällen führen eine Ausfallrate von $10^{-10}\ 1/h$. Die Voting Watchdog Hardware gehört zwar nicht zu den Einpunktausfallkomponenten, dennoch wurde eine entsprechende Ausfallrate festgelegt, da identifiziert wurde, dass der Voting Watchdog das Gesamtsystem hinsichtlich des Ausfallverhaltens stark beschränkt. Die Ursache dafür ist der invariante Diagnosedeckungsgrad, der die zweite Komponente dieses Zweipunktausfalles darstellt. Die Ausfallrate des Watchdogs musste entsprechend gering gewählt werden, wenn der Diagnosedeckungsgrad nicht geändert wird. Die Komponenten der zweiten Ordnung bestehen aus der Sensorhardware. Die Markovanalyse zeigte bereits,

**Tab. 4.6:** Auslegung der Komponenten für die Einzel-ECU Architektur

| $\lambda = 10^{-6,2}\ \frac{1}{h}$ | $\lambda = 10^{-7}\ \frac{1}{h}$ | $\lambda = 10^{-10}\ \frac{1}{h}$ |
|---|---|---|
| MCU Core 1 bis 4 | Hardware Kamera | Voting Core |
| Monitoring Kamera 1 bis 4 | Hardware Radar | Voting Supply |
| Monitoring Radar 1 bis 4 | Hardware Lidar | Hardware Voting Watchdog |
| Monitoring Lidar 1 bis 4 | | |
| Sensor Fusion 1 bis 4 | | |
| Processing 1 bis 4 | | |
| Supply 1 bis 4 | | |
| Hardware Watchdog 1 bis 4 | | |

dass die Sensorik zu einer Gruppe von Komponenten gehört, dessen Ausfallrate geringer gewählt werden muss, in dem Fall geringer als die Ausfallrate der dritte Ordnung Komponenten. Die Sensorausfallrate konnte bis $\lambda = 10^{-7}\ 1/h$ reduziert werden. Zu den Komponenten der dritten Ordnung gehören die Komponenten der MCUs, exklusive der Voting MCU. Ihre Ausfallrate kann Werte bis zu $10^{-6,2}\ 1/h$ annehmen, um den Zielwert des Gesamtsystems von $\lambda = 10^{-8}\ 1/h$ noch zu erfüllen. Tabelle 4.7 zeigt die Auslegung der Komponenten für die DFS Architektur.

**Tab. 4.7:** Auslegung der Komponenten für die DFS Architektur

| $\lambda = 10^{-6}\ \frac{1}{h}$ | $\lambda = 10^{-7,2}\ \frac{1}{h}$ | $\lambda = 10^{-10}\ \frac{1}{h}$ |
|---|---|---|
| Monitoring Kamera 1 bis 4 | Hardware Kamera | Isolierte Komm. |
| Monitoring Radar 1 bis 4 | Hardware Radar | |
| Monitoring Lidar 1 bis 4 | Hardware Lidar | |
| | Sensor Fusion 1 bis 4 | |
| | Processing 1 bis 4 | |
| | Supply 1 bis 4 | |
| | Hardware Watchdog 1 bis 4 | |
| | Voting Core I & II | |
| | Voting Supply I & II | |
| | Hardware Voting Watchdog I & II | |

Der einzige Einpunktausfall der DFS Architektur kann durch die isolierte Kommunikation ausgelöst werden, weshalb nur sie eine Ausfallrate von $10^{-10}\ 1/h$ zugeordnet bekommen hat. Es wird empfohlen die Kommunikation redundant auszulegen, in dessen Folge das System frei von Einpunktausfällen wäre, wodurch ihre Ausfallrate entsprechend erhöht werden könnte, um Kosten einzusparen. Die Ausfallrate der Sensor- und

MCU-Architektur kann bei dem DFS System nicht mehr getrennt voneinander betrachtet werden, da die potentiellen Varianten der MCU-Ausfälle sich aus Minimalschnitten zweiter und dritter Ordnung zusammensetzen. Dennoch kann die Erkenntnis aus der Markovanalyse abgewandelt angewendet werden. Die identifizierte Gruppe der gesondert gestellten Sensorhardwarekomponenten bestand ausschließlich aus Komponenten der zweiten Ordnung. Bei der DFS Architektur werden alle Komponenten, die zu Zweipunktausfällen führen zu dieser kritischen Gruppe gezählt, die eine niedrigere Ausfallrate erhält. Zu dieser Gruppe gehören neben der Sensorhardware auch die MCU-Komponenten sowie die CPU-Kerne und Energieversorgungen beider Voting MCUs. Der Watchdog der Voting MCU wird aufgrund des konstanten Diagnosedeckungsgrades ebenfalls zu der zweiten Ordnung gezählt, obwohl er eigentlich der dritten Ordnung zuzuschreiben wäre. Die Ausfallrate der Komponentengruppe der zweiten Ordnung wurde auf $\lambda = 10^{-7,2}\ 1/h$ festgesetzt. Zur dritten Ordnung mit einer Ausfallrate von $10^{-6}\ 1/h$ gehören lediglich die Sensormonitoringeinheiten der MCUs.

Für einen Vergleich beider Architekturen ist es nicht zielführend für alle Komponenten dieselbe Ausfallrate anzusetzen, um die Gesamtausfallrate zu vergleichen, denn beide Systeme besitzen unterschiedliche Schwachstellen. Wir nur die Komponente einer Schwachstelle unterdimensioniert, liefert das System nur wegen dieser einen Komponente schlechtere Ergebnisse. Ein Vergleich der Architekturen kann über die Auslegung der Gesamtheit der Komponenten erfolgen. Mit einer Zahl von 67 Cut-Sets zweiter Ordnung müssen beim DFS System wesentlich mehr Komponenten zuverlässiger ausgelegt werden als beim Einzel-ECU System, das nur vier Cut-Sets zweiter Ordnung besitzt. Das führt im betrachteten Anwendungsfall dazu, dass 25 Komponenten der DFS Architektur in der Dimension $10^{-7}$ ausgelegt wurden, während bei der Einzel-ECU Architektur nur drei Komponenten in dieser Dimension ausgelegt werden müssen. Folglich ist die Auslegung der Einzel-ECU zur Erreichung des ASIL D Zielwerts wirtschaftlicher. Mit einer Zahl von 32 Komponenten wird der Großteil des Einzel-ECU Systems in der Dimension $10^{-6}$ ausgelegt. Werden Architekturen bestehend aus drei Sensoren und vier MCUs betrachtet, ist die Gesamtzahl der Komponenten mit (hier) 38 beim DFS und beim Einzel-ECU System identisch.

# 5 Erkenntnisse aus der Systemanalyse und dessen weiterführende Anwendung

Die Inverkehrbringing von automatisierten bzw. autonomen Fahrzeugsystemen der Automatisierungsstufen vier und fünf erfordert die Entwicklung von fehlertoleranten Systemarchitekturen, um auf die Rückfallebene des Fahrers verzichten zu können. In dieser Arbeit wurden verschiedene fehlertolerante Ansätze analysiert sowie Systemeigenschaften abgeleitet, die für weiterführende Entwicklungen verwendet werden können. Die Forschung hat sich bisher auf separate Aspekte der Fehlertoleranz konzentriert, die auf Selbstdiagnose, Zuverlässigkeit, Verfügbarkeit, Rekonstruktion und Fehlerbehebung beruht. Der vorliegende Ansatz kombiniert die Bereiche der Selbstdiagnose und der Zuverlässigkeit in ihrer Analyse. Dabei konnten folgende Erkenntnisse gewonnen werden.

**Erkenntnis 1** Die höchste Zuverlässigkeit erreichen reine Parallelsysteme (1ooN Redundanz). Für fehlertolerante Anwendungen sind sie aufgrund ihrer fehlenden Fähigkeit zur Selbstdiagnose jedoch nicht geeignet. MooN Majoritätsredundanzen mit M > 1 validieren ihren Input durch einen Mehrheitsvergleich, wodurch im Fehlerfall Teilsysteme abgeschaltet sowie verschiedene Betriebsmodi je nach Systemzustand verwendet werden können. Die Entwicklung von Diagnose- und Überwachungseinrichtungen für den fehlertoleranten Betrieb von 1ooN Systemen würde die Entwicklung von fehlertoleranten Systemarchitektur hinsichtlich der Reduzierung der Komplexität und Erhöhung der Zuverlässigkeit voranbringen.

**Erkenntnis 2** 2oo4 Architekturen sind erheblich zuverlässiger als 2oo3 Architekturen. 2oo4 Systeme sind v.a. im militärischen Bereich, in der Raumfahrt und in Atomkraftwerken etabliert. Für den automotiven Sektor werden jedoch hauptsächlich nur 2oo2 und 2oo3 Architekturen diskutiert. Es ist zu empfehlen, dass zukünftige Forschungsarbeiten vermehrt 2oo4 Systeme berücksichtigen. Eine wichtige zu klärende Frage ist, ob der Zuverlässigkeitsgewinn im Vergleich zu den gesteigerten Kosten durch die erhöhte Komponentenzahl und im Vergleich zu dem größeren Hardware-Overhead zu rechtfertigen ist.

**Erkenntnis 3** Wird ein System, welches auf einer einzelnen Domain ECU basiert verwendet, dann liefert eine 2oo3 Sensor-/ 2oo4 MCU-Architektur oder eine 2oo2 Sensor-/ 2oo4 MCU-Architektur die beste Eignung hinsichtlich ihrer Zuverlässigkeit unter Berücksichtigung ihrer Selbstdiagnosefähigkeit. Systeme mit einem großen parallelen Anteil liefern die höchsten Werte und Systeme mit einem hohen seriellen Anteil liefern die niedrigsten Werte. Die Zuverlässigkeit ändert sich nur marginal zwischen den Sensorarchitekturen 2oo2 und 2oo3. Aus zuverlässigkeitstechnischer Sicht spielt es also eine untergeordnete Rolle, ob zwei oder drei Sensoren verwendet werden.

**Erkenntnis 4** Die Komponenten der Systemarchitekturen können für ihre Auslegung zur Erreichung von Zuverlässigkeitszielwerten (z.B. der ISO 26262) in drei Gruppen eingeteilt werden, dessen Grundlage im Wesentlichen die Ordnung der den Komponenten zugehörigen Minimalschnitten darstellt. Im betrachteten Anwendungsfall hat sich ein Faktor von ca. 10 zwischen der Ausfallrate von Ordnung zwei und drei bewährt, während die Ausfallrate für die erste Ordnung normativ bedingt konstant ist. Diese Erkenntnis ist auf weiterführende Systemauslegungen übertragbar. Zu beachten ist, dass sich nicht blind an den Minimalschnitten orientiert werden kann, da verschiedene Effekte dazu führen können, dass eine Komponente hinsichtlich eines Systemausfalls kritischer ist, als die sonstigen Vertreter seiner Cut-Set Ordnung. Hierfür kann die Anwendung von Importanzanalysen empfohlen werden.

**Erkenntnis 5** Zur Erreichung von Zuverlässigkeitszielwerten (z.B. der ISO 26262) ist die Auslegung von Einzel-ECU Architekturen wirtschaftlicher als die Auslegung von DFS Architekturen. Die Ursache dafür ist die größere Zahl der Minimalschnitte zweiter Ordnung beim DFS System, wodurch mehr Komponenten zuverlässiger ausgelegt werden müssen. Dennoch kann auf der Grundlage der vorliegenden Analyse keine Aussage darüber getroffen werden, welche der beiden Architekturen für einen fehlertoleranten Einsatz besser geeignet ist. Es wurde ermittelt, dass das Einzel-ECU System zuverlässiger als das DFS System ist. Es ist aber auch anfälliger für Common-Cause-Ausfälle, denen durch Separation oder Diversität entgegengewirkt werden kann. Für eine Anwendung dieser Konzepte eignen sich DFS-Architekturen besonders gut. Es ist wahrscheinlich, dass die geringere Anfällig für Common-Cause-Ausfällen von DFS-Systemen ihre geringere Zuverlässigkeit bzw. ihr größerer wirtschaftlicher Aufwand aufwiegt. Es sind Methoden zu entwickeln, die das Ausfallverhalten von Systemen bezüglich Ausfällen gemeinsamer Ursache quantifizieren können. Dies könnte beispielsweise mit einem Diversitätindikator geschehen, der im Fehlerbaum als zusätzlichen Einpunktausfall modelliert wird. Der Indikator gibt durch die Wahl einer hohen oder geringen Ausfallrate an, wie Anfällig ein System für CCF ist. Hierfür werden jedoch Ausfalldaten benötigt,

um den Anteil der CCF beschreiben zu können. Der in dieser Arbeit vorgestellte Ansatz kann lediglich dazu dienen, den Bereich zu definieren, in dem sich der Diversitätsindikator bewegen kann, ohne das Ausfallverhalten des Gesamtsystems zu beschränken, indem sich an den Ausfallwahrscheinlichkeiten der Gatter der zweiten Ebene des Fehlerbaums orientiert wird.

**Erkenntnis 6** Die Verwendung von konstanten Fehlerraten führt im Fehlerbaum, sofern nicht ausschließlich serielle Gatter eingesetzt werden, zu nicht konstanten Ausfalltaten der Elemente der höheren Ebenen sowie des Top-Level Ereignis. Für eine weiterführende Systementwicklung wäre es von Nutzen, die Weibull-Parameter des Gesamtsystems zu schätzen, um dessen Ausfallverhalten noch detaillierter beschreiben zu können.

In dieser Arbeit wurde sich auf die Aspekte Selbstdiagnose und Zuverlässigkeit der Fehlertoleranz konzentriert. Für weiterführende Arbeiten ist es sinnvoll, sich auch mit den weiteren Aspekten der Verfügbarkeit, der Rekonstruktion und der Fehlerbehebung zu befassen. Die vorgestellte DFS Architektur eignet sich dafür bestens. Zur Reduzierung der Komplexität wurde die beiden ECUs des DFS Systems aktiv redundant ausgelegt. Eine echte Fehlertoleranz wird jedoch nur durch eine passiv redundante bzw. stand-by Redundanz erreicht. Über diesen Weg wird der Aspekt der Rekonstruktion berücksichtigt. Im Fehlerfall schaltet ECU 1 ab und ECU 2 übernimmt den Betrieb. Währenddessen setzen Fehlerbehebungsalgorithmen (z.B. Error Corecting Codes) die ausgefallen Komponenten von ECU 1 wieder in Stand, wodurch der Aspekt der Fehlerbehebung berücksichtigt wird. Die Zeit zur Instandsetzung einer Komponenten kann als Reparaturrate implementiert werden, die Verfügbarkeitsbetrachtungen ermöglicht. Durch diesen Ansatz sind alle Aspekte der Fehlertoleranz abgedeckt.

Hinsichtlich der vorliegenden Arbeit ist noch folgende Anmerkung zu machen. Zur Berechnung der Fehlerbäume wurde das R Framework FaultTree inklusive der Erweiterung FaultTree.Scram verwendet. Ursprünglich war es geplant zusätzlich Importanzanalysen und Fuzzyfizierungen der Komponenten Ausfallraten vorzunehmen, die das verwendete Framework eigenständig durchführen kann. Durch den Einsatz von Majoritätsredundanzen kam es jedoch zu Komplikationen. Da die Voting Gatter von der Software nicht berechnet werden können, musste die Modellierung über Parallel-Serien-Systeme erfolgen. Diese Methode erfordert den Einsatz von Duplikaten, d.h. den Einsatz von exakt denselben Komponenten an unterschiedlichen Positionen. Diese Duplikate bereiteten der Software jedoch erneut Probleme. Die Fuzzifizierungen und Importanzanalysen konnten unter Verwendung von Duplikaten nicht berechnet werden. Für weiterführende Arbeiten ist zu prüfen, ob die Verwendung von Duplikaten bei der

Modellierung von Majoritätsredundanzen mittels Parallel-Serien-Systemen tatsächlich essentiell ist oder ob die gleichen Ergebnisse auch mit unterschiedlichen Komponenten, die die selben Parameter besitzen auftreten. Gegebenenfalls ist zu prüfen, ob andere Softwarepakete für die Modellierung von MooN-Systemen besser geeignet sind.

# Literaturverzeichnis

[1] Bundesministerium für Verkehr und digitale Infrastruktur. Entwurf eines Gesetzes zur Änderung des Straßenverkehrsgesetzes und des Pflichtversicherungsgesetzes – Gesetz zum autonomen Fahren; 10.02.2021. Available from: https://www.bmvi.de/SharedDocs/DE/Anlage/Gesetze/Gesetze-19/gesetz-aenderung-strassenverkehrsgesetz-pflichtversicherungsgesetz-autonomes-fahren.pdf.

[2] Bundesgesetzblatt Jahrgang 2017 Teil I Nr 38, ausgegeben zu Bonn am 20 Juni 2017. Achtes Gesetz zur Änderung des Straßenverkehrsgesetzes; 16.06.2017. Available from: https://www.bgbl.de/xaver/bgbl/start.xav?start=//*[@attr_id='bgbl117s1648.pdf']#__bgbl__//*[@attr_id='bgbl117s1648.pdf']__1624132646983.

[3] Williams M. PROMETHEUS - The European research programme for optimising the road transport system in Europe. In: IEE Colloquium on Driver Information; 1988. p. 1/1–1/9. Available from: https://ieeexplore.ieee.org/document/209729.

[4] The 2005 DARPA Grand Challenge. Springer Berlin Heidelberg; 2007. Available from: https://doi.org/10.1007/978-3-540-73429-1.

[5] The DARPA Urban Challenge. Springer Berlin Heidelberg; 2009. Available from: https://doi.org/10.1007/978-3-642-03991-1.

[6] Reke M, Peter D, Schulte-Tigges J, Schiffer S, Ferrein A, Walter T, et al. A Self-Driving Car Architecture in ROS2. In: 2020 International SAUPEC/RobMech/PRASA Conference. IEEE; 2020. Available from: https://doi.org/10.1109/saupec/robmech/prasa48453.2020.9041020.

[7] Thrun S, Montemerlo M, Dahlkamp H, Stavens D, Aron A, et al. Stanley: The robot that won the DARPA Grand Challenge. vol. 23. Wiley; 2006. p. 661–692. Available from: https://doi.org/10.1002/rob.20147.

[8] Urmson C, Anhalt J, Bagnell D, Baker C, Bittner R, et al. Autonomous driving in urban environments: Boss and the Urban Challenge. vol. 25. Wiley; 2008. p. 425–466. Available from: https://doi.org/10.1002/rob.20255.

[9] Daily M, Medasani S, Behringer R, Trivedi M. Self-Driving Cars. vol. 50. Institute of Electrical and Electronics Engineers (IEEE); 2017. p. 18–23. Available from: https://doi.org/10.1109/mc.2017.4451204.

[10] Kopestinsky A. 25 Astonishing Self-Driving Car Statistics for 2021; 2021. Https://policyadvice.net/insurance/insights/self-driving-car-statistics/. Available from: https://policyadvice.net/insurance/insights/self-driving-car-statistics/.

[11] Baleani M, Ferrari A, Mangeruca L, Sangiovanni-Vincentelli A, Peri M, Pezzini S. Fault-tolerant platforms for automotive safety-critical applications. In: Proceedings of the international conference on Compilers, architectures and synthesis for embedded systems - CASES '03. ACM Press; 2003. Available from: https://doi.org/10.1145/951710.951734.

[12] Andre Kohn, Michael Käßmeyer, Rolf Schneider, Andre Roger, Claus Stellwag, Andreas Herkersdorf. Fail-operational in safety-related automotive multi-core systems. 10th IEEE International Symposium on Industrial Embedded Systems (SIES). IEEE; 2015. Available from: https://doi.org/10.1109/SIES.2015.7185051.

[13] Tasuku Ishigooka, Shinya Honda, Hiroaki Takada. Cost-Effective Redundancy Approach for Fail-Operational Autonomous Driving System. 2018 IEEE 21st International Symposium on Real-Time Distributed Computing (ISORC). IEEE; 2018. Available from: https://doi.org/10.1109/ISORC.2018.00023.

[14] Tobias Schmid, Stefanie Schraufstetter, Stefan Wagner, Dominik Hellhake. A Safety Argumentation for Fail-Operational Automotive Systems in Compliance with ISO 26262. 2019 4th International Conference on System Reliability and Safety (ICSRS). IEEE; 2019. Available from: https://doi.org/10.1109/ICSRS48664.2019.8987656.

[15] Lin SC, Zhang Y, Hsu CH, Skach M, Haque ME, Tang L, et al. The Architectural Implications of Autonomous Driving. In: Proceedings of the Twenty-Third International Conference on Architectural Support for Programming Languages and Operating Systems. ACM; 2018. Available from: https://doi.org/10.1145/3173162.3173191.

[16] Bülent Sari. Fail-operational Safety Architecture for ADAS/AD Systems and a Model-driven Approach for Dependent Failure Analysis. Wissenschaftl. Reihe Fahrzeugtech. Uni. Stuttgart. Springer Vieweg, Wiesbaden; 2020. Available from: https://doi.org/10.1007/978-3-658-29422-9.

[17] Dai X, Dong W, Sun X. Reliability and safety analysis of M out of N system based on Markov Process. In: 2016 IEEE Information Technology, Networking, Electronic and Automation Control Conference. IEEE; 2016. Available from: https://doi.org/10.1109/itnec.2016.7560348.

[18] Phil Salewski. Definition des Begriffs „Inverkehrbringen“: Ausmaß und Bedeutung. IT-Recht Kanzlei München; 2019. Accessed: 01.02.2021. Available from: https://www.it-recht-kanzlei.de/inverkehrbringen-definition.html#abschnitt_44.

[19] Alfred Neudörfer. Konstruieren sicherheitsgerechter Produkte. Methoden und systematische Lösungssammlungen zur EG-Maschinenrichtlinie. Berlin, Heidelberg: Springer; 2014. p. 15. ISBN: 978-3-642-45446-2. Available from: https://doi.org/10.1007/978-3-642-45447-9.

[20] Hans-Leo Ross. Funktionale Sicherheit im Automobil: ISO 26262, Systemengineering auf Basis eines Sicherheitslebenszyklus und bewährten Managementsystemen. München: Hanser; 2014. p. 1,7. ISBN: 978-3-446-43632-9. Available from: https://doi.org/10.3139/9783446438408.fm.

[21] ISO 26262-1:2018. Road vehicles – Functional safety – Part 1: Vocabulary; 2018. Available from: https://www.iso.org/standard/68383.html.

[22] ISO 26262-2:2018. Road vehicles – Functional safety – Part 2: Management of functional safety; 2018. Available from: https://www.beuth.de/de/norm/iso-26262-2/300423923.

[23] ISO 26262-3:2018. Road vehicles – Functional safety – Part 3: Concept phase; 2018. Available from: https://www.beuth.de/de/norm/iso-26262-3/300423934.

[24] ISO 26262-9:2018. Road vehicles — Functional safety — Part 9: Automotive safety integrity level (ASIL)-oriented and safety-oriented analyses; 2018. Available from: https://www.iso.org/standard/68391.html.

[25] ISO 26262-5:2018. Road vehicles — Functional safety — Part 5: Product development at the hardware level; 2018. Available from: https://www.iso.org/standard/68387.html.

[26] SAE J3016. Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles. SAE International; 2018. Available from: https://www.sae.org/standards/content/j3016_201806/.

[27] Walther Wachenfeld, Hermann Winner. Die Freigabe des autonomen Fahrens. In: Markus Maurer, J Christian Gerdes, Barbara Lenz, Hermann Winner, editors. Autonomes Fahren. Vieweg, Berlin, Heidelberg: Springer; 2015. p. 439–464. Available from: https://doi.org/10.1007/978-3-662-45854-9_21.

[28] Andreas Knapp, Markus Neumann, Martin Brockmann, Rainer Walz, Thomas Winkle. Code of Practice for the Design and Evaluation of ADAS. RESPONSE 3 a PReVENT Projekt. Inforamtion Society Technologies; 2009. Available from: https://www.acea.be/uploads/publications/20090831_Code_of_Practice_ADAS.pdf.

[29] Freescale Semiconductor, Inc. Safety Manual for MPC5744P. Document Number: MPC5744PSM Rev 3; 2014. p. 32, 47. Available from: https://www.nxp.com/files-static/microcontrollers/doc/ref_manual/MPC5744PSM.pdf.

[30] Brandon Schoettle. Sensor Fusion: A Comparison of Sensing Capabilities of Human Drivers and Highly Automated Vehicles. Technical Report SWT-2017-12. University of Michigan; 2017. Available from: http://www.umich.edu/%7Eumtriswt/PDF/SWT-2017-12.pdf.

[31] P Christ, T Voege. Safer Roads with Automated Vehicles? Corporate Partnership Board Report. ITF/OECD; 2018. Available from: https://www.itf-oecd.org/safer-roads-automated-vehicles-0.

[32] Raimund Schesswendter. Tesla präsentiert Supercomputer für Vision-Only-Ansatz beim autonomen Fahren. Digital Pioneers. t3n; 2021. Accessed: 06.08.2021. Available from: https://t3n.de/news/tesla-supercomputei-autonomes-fahren-radar-1386513/.

[33] Mobileye. True Redundancy™ The Realistic Path to Deploying AVs at Scale; 2021. Available from: https://www.mobileye.com/true-redundancy/.

[34] SafeAdapt. Safe adaptive software for fully electric vehicles;. Online. Accessed on 30 Juni 2021. Available from: https://www.safeadapt.eu/.

[35] Weiss G, Schleiss P, Drabek C, Ruiz A, Radermacher A. Safe Adaptation for Reliable and Energy-Efficient E/E Architectures. In: Comprehensive Energy Management - Safe Adaptation, Predictive Control and Thermal Management. Springer International Publishing; 2018. p. 1–18. Available from: https://doi.org/10.1007/978-3-319-57445-5_1.

[36] Oszwald F, Obergfell P, Traub M, Becker J. Using Simulation Techniques within the Design of a Reconfigurable Architecture for Fail-Operational Real-Time Automotive Embedded Systems. In: 2018 IEEE International Systems Engineering Symposium (ISSE). IEEE; 2018. Available from: https://doi.org/10.1109/syseng.2018.8544451.

[37] Edler F, Soden M, Hankammer R. Fehlerbaumanalyse in Theorie und Praxis. 1st ed. Springer Vieweg, Berlin, Heidelberg; 2015. p. 239 ff. Available from: https://doi.org/10.1007/978-3-662-48166-0.

[38] Silkworth D. Fault Tree Analysis on R; 2021. Available from: http://www.openreliability.org/fault-tree-analysis-on-r/.

[39] Statista. Autofahrer in Deutschland nach selbst gefahrenen Kilometern pro Jahr von 2016 bis 2020; 2021. Available from: https://de.statista.com/statistik/daten/studie/183003/umfrage/pkw---gefahrene-kilometer-pro-jahr/.

# Abkürzungsverzeichnis

| | |
|---|---|
| **ASIL** | Automotive Safety Integrity Level |
| **FTA** | Fehlerbaumanalyse |
| **ACC** | Adaptive-Cruise-Control |
| **OEM** | Original Equipment Manufacturer |
| **E/E** | elektrisch, elektronisch |
| **E/E/PE** | elektrisch, elektronisch, programmierbar elektronisch |
| **mvn** | m von n |
| **MooN** | M out of N |
| **ECU** | Electronic Control Unit |
| **CCF** | Common Cause Failure |
| **CMF** | Common Mode Failure |
| **DFS** | Dual-Fail-Safe |
| **FMEA** | Fehlermöglichkeits und -einflussanalyse |
| **RBD** | Reliability Block Diagram |
| **SF** | Sensor Fusion |
| **P** | Processing |
| **Sup** | Supply (Energieversorgung) |
| **HW** | Hardware Watchdog |
| **VSup** | Voting Supply |
| **HWV** | Hardware Voting Watchdog |
| **HK** | Hardware Kamera |
| **HR** | Hardware Radar |
| **HL** | Hardware Lidar |
| **MK** | Monitoring Kamera |
| **MR** | Monitoring Radar |
| **ML** | Monitoring Lidar |
| **IC** | Isolierte Kommunikation |

# Abbildungsverzeichnis

# Tabellenverzeichnis

# Eidesstattliche Erklärung

*Hiermit, erkläre ich, dass ich die von mir eingereichte Abschlussarbeit (Master-Thesis) selbstständig verfasst und keine andere als die angegebene Quelle und Hilfsmittel benutzt sowie Stellen der Abschlussarbeit, die anderen Werken dem Wortlaut oder Sinn nach entnommen wurden, in jedem Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.*

*Ich bin damit einverstanden, dass die Arbeit durch Dritte eingesehen und unter Wahrung urheberrechtlicher Grundsätze zitiert werden darf.*

Wuppertal, 23.08.2021
Ort, Datum

Tim M. Julitz