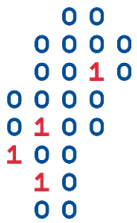
Hochschule Darmstadt – Frankfurt University of Applied Sciences
Hochschule Fulda – Hochschule RheinMain



Promotionszentrum
Angewandte Informatik
HAW Hessen

Model-Based Systems Engineering sicherheitskritischer Anwendungen im Maschinenbau mittelständischer Unternehmen

Dissertation zur Erlangung des akademischen Grades eines

Doktor der Naturwissenschaften (Dr. rer. nat.)

am hochschulübergreifenden Promotionszentrum Angewandte Informatik

der hessischen Hochschulen

von **Nick Berezowski, M. Sc.**

geboren am 06.12.1989 in Jena

Abgabe: 25.07.2024

Prüfung: 17.01.2025

Gustavsburg, Juli 2024

Erstbetreuung:	Prof. Dr. Reinhold Kröger (Hochschule RheinMain)
Zweitbetreuung:	Prof. Dr. Markus Haid (Hochschule Darmstadt)
Erstbegutachtung:	Prof. Dr. Bernhard Humm (Hochschule Darmstadt)
Zweitbegutachtung:	Prof. Dr. Matthias Hemmje (FernUniversität Hagen)

Model-Based Systems Engineering sicherheitskritischer Anwendungen im Maschinenbau mittelständischer Unternehmen © 2024 by Nick Berezowski is licensed under CC BY 4.0

Eidesstattliche Erklärung

Hiermit erkläre ich, Nick Berezowski an Eides statt, dass ich die Dissertation mit dem Titel „Model-Based Systems Engineering sicherheitskritischer Anwendungen im Maschinenbau mittelständischer Unternehmen“ selbstständig und keine anderen als die von mir angegebenen Hilfsmittel benutzt habe. Die den herangezogenen Werken wörtlich oder sinngemäß entnommenen Stellen sind als solche gekennzeichnet und die Grundsätze guter wissenschaftlicher Praxis wurden eingehalten. Ich versichere, dass ich keine kommerzielle Promotionsberatung in Anspruch genommen habe und die Arbeit nicht schon in einem früheren Promotionsverfahren im In- oder Ausland angenommen oder als ungenügend beurteilt worden ist.

Gustavsburg, 25. Juli 2024	_Nick Berezowski_____
Ort und Datum	Unterschrift

„*Das Verhüten von Unfällen darf nicht als eine Vorschrift des Gesetzes aufgefasst werden, sondern als ein Gebot menschlicher Verpflichtung und wirtschaftlicher Vernunft.*“ - Werner von Siemens

„*Der Computer ist das bemerkenswerteste Werkzeug, das wir je erfunden haben. Er kann eine totale Zeitverschwendung sein oder er kann uns helfen, Probleme zu lösen und unser Wissen zu erweitern.*“ - Ken Jennings

„*Modelle sind falsch, aber einige von ihnen sind nützlich.*“ - George E. P. Box

„*Die Modelle der Wissenschaft sind nicht wahr, und eben darum sind sie nützlich. Sie erzählen einfache Geschichten, die unser Geist erfassen kann. Es sind Lügen-für-Kinder, einfach Geschichten für den Unterricht und darum keinen Deut schlechter. Der Fortschritt der Wissenschaft liegt darin, dass immer klügere Kinder immer überzeugendere Lügen erzählen werden.*“ - Terry Pratchett

„*Das Entdecken des Unerwarteten ist wichtiger als die Bestätigung des Bekannten.*“ - George E. P. BoxBerezowski

„*Wenn du keine Fehler machst, dann sind die Probleme, an denen du arbeitest, nicht schwierig genug. Und das ist ein grosser Fehler.*“ - Frank Wilczek

Danksagung

Ohne die Unterstützung vieler Menschen wäre diese Arbeit nicht möglich gewesen. Zunächst möchte ich meinem Betreuer Reinhold Kröger für seine brillante Anleitung, seine Unterstützung, sein Wissen, seine Flexibilität, Offenheit beim Diskutieren neuer Ideen und seine Bereitschaft, Zeitfenster zu jedem möglichen Zeitpunkt anzubieten, danken. Ich kann Ihnen dafür nicht genug danken. Außerdem muss ich meinem zweiten Betreuer Markus Haid danken, dass er mir die Möglichkeit gegeben hat, in seinem Forschungsinstitut zusammenzuarbeiten und daran geglaubt hat, dass ich dazu fähig bin, solch ein Vorhaben durchzuführen. Ich habe viel mehr gelernt, als ich je erwartet hatte, zu der Zeit, als ich zum ersten Mal zu dir ins CCASS kam. Vielen Dank auch für deine Geduld und wertvolles Feedback zu zahlreichen Präsentationen, Entwürfen für Veröffentlichungen und dieser Arbeit. Es war definitiv eine großartige Lernerfahrung. Michael Kuhn danke ich sehr für die Betreuung und Unterstützung meines Projektes in den ersten eineinhalb Jahren und bedaure die Notwendigkeit des Betreuerwechsels sehr.

Auch ein großes Dankeschön geht an Bernhard Humm, der auf Seiten des Promotionszentrums beim Wechsel des Betreuers und Themas nach den ersten zwei Jahren der Promotion eine große Unterstützung bot und dabei unter anderem Reinhold Kröger vorgeschlagen hatte.

Ich bedanke mich bei meinen Freunden und Kollegen bei ProNES dafür, dass sie ihre Gedanken und Ideen geteilt und in Diskussionen offene Rückmeldungen gegeben haben. Ich habe dies sehr geschätzt. Danke an Jochen Weber, dass er mir die Unterstützung bei der Evaluation meines Promotionsvorhabens bot.

Zusätzlich bedanke ich mich bei meinen Freunden, die ebenso an ihren Promotionen arbeiteten. Es war wesentlich, die Unterstützung und das Verständnis von Menschen zu haben, die sich ähnlichen Herausforderungen während einer solch herausfordernden Zeit stellen.

Ein besonderer Dank geht an meine Familie und engen Freunde, die immer unglaubliche Unterstützung und Liebe angeboten haben. Danke an euch alle. Danke an Freunde, die gute Zeiten besser und schlechte Zeiten einfacher gemacht haben.

Abschließend und mit besonderem Nachdruck danke an meine wundervolle Frau Liana, die die harte Zeit des Schreibens an meiner Seite ausgehalten und dazu beigetragen hat, dass alles ein wenig leichter wurde. Ebenso möchte ich meiner ungeborenen Tochter Eliza danken, die mich durch ihre bloße Anwesenheit und die Vorfreude auf ihre Ankunft inspirierte und motivierte. Ohne euch hätte ich das wohl nie geschafft.

Kurzzusammenfassung

Nick Berezowski

Model-Based Systems Engineering sicherheitskritischer Anwendungen im Maschinenbau mittelständischer Unternehmen

Das Promotionsprojekt widmet sich der Anwendung des modellbasierten Systems Engineerings (MBSE) für die Entwicklung sicherheitskritischer Systeme im Maschinenbau, insbesondere für kleine und mittelständische Unternehmen (KMUs). Das Hauptziel ist die Steigerung von Qualität und Sicherheit der Systeme und gleichzeitig die Berücksichtigung der Ressourcen und Anforderungen von KMUs, um ihnen einen verbesserten Marktzugang zu ermöglichen.

Um dieses Ziel zu erreichen, werden SysML/UML durch spezifische Profile erweitert, um die Bedürfnisse der Stakeholder im Maschinenbau, insbesondere bei der Entwicklung von Prüfmaschinen, abzubilden. Zudem werden domänenspezifische Sicherheitsanalysemethoden, wie Risikographen zur Risikobeurteilung und dynamische Fehlerbaumanalyse sowie sicherheitsbezogene Blockdiagrammmethoden zur Beurteilung der Risikominderung, als Erweiterung von OMG Risk Analysis and Assessment Modeling Language (RAAML) integriert und in Papyrus implementiert.

Ein weiterer Schwerpunkt liegt auf der Entwicklung eines Konzepts zur losen Kopplung von Analysewerkzeugen an die Modellierungsplattform. Dieses Konzept erleichtert die Wiederverwendung von domänenspezifischen Werkzeugen, vereinfacht die Anwendbarkeit und reduziert den Integrationsaufwand für KMUs. Die Implementierung erfolgt durch die Anbindung der Analysewerkzeuge Stormchecker und SISTEMA mittels LabVIEW, das in der Maschinenbauindustrie weit verbreitet ist und Transformatoren für den Export von Artefakten, der Analysewerkzeugausführung und dem Reimport von Ergebnisdaten dient.

Die Ergebnisse werden anhand einer Fallstudie einer Batteriezellenprüfmaschine und durch Interviews mit den Stakeholdern des KMUs ProNES Automation GmbH evaluiert. Die Dissertation präsentiert einen anwendbaren Ansatz für technische Stakeholder in KMUs des Maschinenbaus, der eine direkte Annotation der maschinenbauspezifischen Risikobeurteilung und Sicherheitsanalysemethoden ermöglicht. Die vorgestellte Lösung bietet eine praxisnahe Möglichkeit, MBSE effizient in KMU-Prozesse zu integrieren und die Entwicklung sicherheitskritischer Maschinen zu verbessern.

Abstract

Nick Berezowski

Title translated: Model-Based Systems Engineering for Safety-Critical Applications in Mechanical Engineering of Small and Medium-Sized Enterprises

This project focuses on applying Model-Based Systems Engineering (MBSE) to develop safety-critical mechanical engineering systems, specifically targeting small and medium-sized enterprises (SMEs). The primary objective is to enhance the quality and safety of systems while considering the resources and requirements of SMEs, enabling them to achieve improved market access.

To achieve this goal, SysML/UML is extended through specific profiles to capture the needs of stakeholders in mechanical engineering, particularly in the development of testing machines. Additionally, domain-specific safety analysis methods such as risk graphs for risk assessment and dynamic fault trees and safety-related block diagrams for risk mitigation assessment are integrated as extensions to the OMG Risk Analysis and Assessment Modeling Language (RAAML) and implemented in Papyrus.

Another crucial aspect of this project involves the formulation of a concept for loosely coupling various analysis tools with the modeling platform. This concept facilitates the reuse of domain-specific tools, simplifies applicability, and reduces integration efforts for SMEs. The Implementation is realized by connecting the analysis tools Stormchecker and SISTEMA using LabVIEW, a widely used tool in the mechanical engineering industry, which serves as a transformer for exporting artifacts, executing analysis tools, and reimporting result data.

The outcomes are evaluated through a case study involving a battery cell testing machine and stakeholder interviews with ProNES Automation GmbH, an SME. The dissertation presents an applicable approach for technical stakeholders in mechanical engineering SMEs, enabling direct annotation of mechanical engineering-specific risk assessment and safety analysis methods. The proposed solution offers a tangible avenue for the efficient integration of MBSE practices within the operational processes of SMEs, thereby contributing to the enhanced development of safety-critical machinery.

Inhaltsverzeichnis

Eidesstattliche Erklärung ... **III**

Danksagung ... **V**

Kurzzusammenfassung ... **VI**

Abstract ... **VII**

Inhaltsverzeichnis ... **IX**

Abbildungsverzeichnis ... **XV**

Tabellenverzeichnis ... **XXI**

Formelverzeichnis ... **XXII**

Listingverzeichnis ... **XXIII**

Abkürzungsverzeichnis ... **XXIV**

1 Einleitung & Motivation ... **1**

1.1 Motivation ... **1**

1.2 Problemfelder ... **7**

1.2.1 Probleme bezogen auf Modellierung ... 8

1.2.2 Probleme bezogen auf Unterstützung in einzelnen Domänen ... 9

1.2.3 Probleme bezogen auf die Maschinenindustrie ... 9

1.2.4 Probleme bezogen auf die Toolkommunikation ... 10

1.2.5 Probleme bezogen auf KMUs ... 11

1.2.6 Problembeschreibung (PB) und zusammenfassende Herausforderung ... 13

1.3 Forschungsfragen, Methodologie, Zielsetzung und Aufgaben ... **15**

1.4 Ansatz und Aufbau der Arbeit ... **20**

2 Stand der Wissenschaft und Technologie ... **23**

2.1 Maschinenbauspezifische Modellierung ... **24**

2.1.1 Grundlagen des MBSE ... 24

2.1.2 Stakeholder in der Maschinenentwicklung ... 25

2.1.3 Sicherheitsbezogene Entwicklung im Maschinenbau ... 29

2.1.4 Gemeinsames Informationsmodell ... 38

2.1.5 Diskussion ... 42

2.2 Integration der Analysemethoden 44
2.2.1 Sicherheitsanalysemethoden im Maschinenbau 44
2.2.2 Sicherheitsanalyseerweiterungen in MBSE-Frameworks 54
2.2.3 Tool-Integration 63
2.2.4 Diskussion 66
2.3 MBSE-Prozesse für KMUs 67
2.3.1 Identifikation der Hindernisse zur Anwendbarkeit von MBSE in KMUs 67
2.3.2 Bewertung zur Eignung eines MBSE-Frameworks 69
2.3.3 Diskussion 73
2.4 Zusammenfassung & verbleibende Herausforderungen 74
3 Modellierung 77
3.1 Einführung 77
3.2 Maschinenbauspezifische Modellierung 78
3.2.1 Gemeinsame Sprache und Struktur für die Modellierung 78
3.2.2 Anwendungskontext für Stakeholder in der Maschinenentwicklung 79
3.2.3 Gemeinsames Informationsmodell 88
3.2.4 Construction Structure 92
3.2.5 Machine Assembly Structure 93
3.2.6 Testmachine Construction 96
3.2.7 Mechanical View 99
3.2.8 Electrical View 100
3.2.9 Hydraulic View und Pneumatic View 102
3.2.10 Informatic View 103
3.2.11 ISO 12100 & ISO 13849 104
3.2.12 Inhaltlicher Beitrag 109
3.3 Integration der Analysemethoden 111
3.3.1 Grundlagen zur Modellierung 111
3.3.2 Metamodellumsetzung für DFTs 112
3.3.3 Metamodellumsetzung für SBBM 115
3.3.4 Inhaltlicher Beitrag 119
3.4 MBSE-Prozesse für KMUs 120
3.4.1 Vorgehen zur vereinfachten Anwendbarkeit von MMBSEFE in KMUs 121
3.4.2 Lose Toolkopplung 123
3.4.3 LabVIEW zur Entwicklung von Modelltransformatoren 126
3.4.4 Inhaltlicher Beitrag 127
3.5 Zusammenfassung 127
4 Implementierung 131

4.1 Maschinenbauspezifische Modellierung ... 131
4.1.1 Modellierungswerkzeug ... 131
4.1.2 Maschinenbauspez. Bibliotheken und Profile (MMBSEFE) ... 133
4.2 Risikobeurteilung ... 139
4.2.1 Vorgehen ... 140
4.2.2 DFTs ... 143
4.2.3 SBBM ... 150
4.2.4 Risikograph ... 154
4.3 Zusammenfassung ... 157
5 Evaluation ... 163
5.1 Maschinenbauspezifisches Modellierungsbeispiel ... 164
5.1.1 Bestimmung der Grenzen der Maschine ... 166
5.1.2 Risikoeinschätzung ... 173
5.1.3 Identifizierung der Gefährdungen ... 174
5.1.4 Risikobewertung und Identifikation der risikomindernden Maßnahmen ... 178
5.1.5 Diskussion ... 183
5.2 Integration der Analysemethoden des Maschinenbaus ... 184
5.2.1 Gestaltung der Sicherheitsfunktionen ... 185
5.2.2 Evaluation des Einsatzes von DFTs ... 191
5.2.3 Evaluation des Einsatzes von SBBM ... 195
5.2.4 Vergleich DFTs und SBBM ... 199
5.2.5 Evaluation des Einsatzes von Risikographen ... 200
5.2.6 Diskussion ... 203
5.3 MBSE-Prozesse für KMUs ... 204
5.3.1 Schritt 1 – Evaluationsgegenstand und –ziele ... 204
5.3.2 Schritt 2a – Fragenkatalog zur Erwartungshaltung ... 204
5.3.3 Schritt 3 – Teilnehmer ... 205
5.3.4 Schritt 4a – Interviews zur Erwartungshaltung ... 205
5.3.5 Schritt 5a – Transkription, Analyse und Interpretation zur Erwartungshaltung ... 207
5.3.6 Schritt 2b - Fragenkatalog zur Zufriedenheit ... 208
5.3.7 Schritt 4b – Interviews zur Zufriedenheit ... 209
5.3.8 Schritt 5b – Transkription, Analyse und Interpretation zur Zufriedenheit ... 212
5.4 Zusammenfassung ... 215
6 Zusammenfassung der Arbeit und Ausblick ... 219
6.1 Zusammenfassung ... 219
6.2 Ausblick ... 224
Literatur ... 227

A Anhang 263

A.1 Publikationen & Vorträge 263

A.2 Statement zum Themen und Betreuerwechsel 264

A.3 „OMG-Umfeld“ 264

A.3.1 UML 264
A.3.2 SysML 266
A.3.3 MOF 268
A.3.4 Profile von Profilen 269
A.3.5 OCL 270
A.3.6 CSS 270
A.3.7 XMI 271
A.3.8 EMF 271
A.3.9 MBSE-Plattformen 272

A.4 Bestimmung der Grenzen der Maschine 273

A.4.1 Verwendungsgrenzen 273
A.4.2 Räumliche Grenzen 273
A.4.3 Zeitliche Grenzen 273
A.4.4 Weitere Grenzen 273
A.4.5 Information 274

A.5 Gefährdungsgruppen 274

A.5.1 Mechanische Gefährdungen 274
A.5.2 Elektrische Gefährdungen 274
A.5.3 Thermische Gefährdungen 274
A.5.4 Gefährdungen durch Lärm 275
A.5.5 Gefährdungen durch Vibration 275
A.5.6 Gefährdungen durch Strahlungen 275
A.5.7 Gefährdungen durch Materialien und Substanzen 275
A.5.8 Ergonomische Gefährdungen 276
A.5.9 Gefährdungen im Zusammenhand mit der Einsatzumgebung 276
A.5.10 Kombination von Gefährdungen 277
A.5.11 Gefährdungsgruppen mit Beispielen 277

A.6 Analysemethoden 279

A.6.1 FMEA 279
A.6.2 ETA & UWD 280
A.6.3 GSN 281
A.6.4 STPA 281
A.6.5 Risikomatrix und Vergleich zum Risikographen 282
A.6.6 SBBM 285

A.7 OMG RAAML 289

A.7.1 Domainmodel 289
A.7.2 Paketstruktur 291
A.7.3 FMEA 291
A.7.4 FTA 292

A.8 Erweiterte externe Use-Cases der Stakeholder 294
A.8.1 Maschinenbauingenieur 294
A.8.2 Elektroingenieur 295
A.8.3 Informatiker 296

A.9 MMBSEFE-Funktionsbeschreibungen 297
A.9.1 MachineConstructionConcept 297
A.9.2 ConstructionStructure 298
A.9.3 MachineAssemblyStructure 301
A.9.4 TestmachineConstruction 309
A.9.5 MechanicalView 322
A.9.6 ElectricalView 325
A.9.7 HydraulicView 328
A.9.8 PneumaticView 330
A.9.9 InformaticView 333
A.9.10 ISO 12100 & ISO 13849 335
A.9.11 DFT 344
A.9.12 SBBM 352

A.10 Modulbeschreibung der Transformatoren 362
A.10.1 SBBM in SISTEMA 362
A.10.2 Risikographen in LabVIEW 369

A.11 Modelle der Evaluation 372
A.11.1 Weitere Risikoitems 372
A.11.2 Sicherheitsbezogenes Eingangs-Ausgangs-Interface 377
A.11.3 Entwicklungen zur Sicherheitsfunktion „Nothalt“ 377
A.11.4 Weiterentwicklung zur Sicherheitsfunktion Prüflingsüberhitzungserkennung 384
A.11.5 Weitere Modelle 388

A.12 Interviews zur Evaluation 398
A.12.1 Fragenkatalog zur Erwartungshaltung der Stakeholder 398
A.12.2 Protokoll zur Erwartungshaltung des Maschinenbauers 398
A.12.3 Protokoll zur Erwartungshaltung des Elektroingenieurs 400
A.12.4 Protokoll zur Erwartungshaltung des Informatikers 402
A.12.5 Protokoll zur Erwartungshaltung des Sicherheitsingenieurs 404
A.12.6 Fragenkatalog zur Zufriedenheit der Stakeholder 406
A.12.7 Protokoll zur Zufriedenheit des Maschinenbauers 406
A.12.8 Protokoll zur Zufriedenheit des Elektroingenieurs 408
A.12.9 Protokoll zur Zufriedenheit des Informatikers 410

A.12.10 Protokolle zur Zufriedenheit des Sicherheitsingenieurs 412

Abbildungsverzeichnis

Abbildung 1.1: Problemlösungsphasen nach Nunamaker [112] 16
Abbildung 1.2: Struktur der Arbeit und Problemlösungsansatz 20
Abbildung 2.1: Überblick über Standards der funktionalen Sicherheit 30
Abbildung 2.2: Normen der Maschinenindustrie 31
Abbildung 2.3: Ablauf einer Risikobeurteilung nach ISO 12100 [163, 171] 35
Abbildung 2.4: Iterativer Prozess zur Gestaltung der SRP/CS [171, 34] 37
Abbildung 2.5: SysML-Profile für mechanische Baugruppen nach [32] 40
Abbildung 2.6: Unterscheidung induktive und deduktive Analysen 45
Abbildung 2.7: Einfaches FTA Beispiel 46
Abbildung 2.8: Gate-Darstellungen nach Aslansefat et al. [218] 48
Abbildung 2.9: Klassifizierung eines Risikographen nach IEC 61508 [9] 50
Abbildung 2.10: Risikograph nach ISO 13849 [34] 51
Abbildung 2.11: RAAML Core Profile [237] 59
Abbildung 2.12: RAAML Core Bibliothek [237] 59
Abbildung 2.13: RAAML General Concepts Profil [237] 59
Abbildung 2.14: RAAML General Concepts Bibliothek [237] 60
Abbildung 2.15: RAAML FTA Bibliothek [237] 61
Abbildung 3.1: Anwendungskontext - Modellierung der Anwendungsfälle des Frameworks 80
Abbildung 3.2: Anwendungskontext - Anwendungsfälle Maschinenbauer 81
Abbildung 3.3: Anwendungskontext - Anwendungsfälle Elektroingenieur 83
Abbildung 3.4: Anwendungskontext - Anwendungsfälle Informatiker 84
Abbildung 3.5: Anwendungskontext - Anwendungsfälle Sicherheitsingenieur 86
Abbildung 3.6: Mögliche Schichtenmodelle Metaarchitektur MMBSEFE 89
Abbildung 3.7: Grobe MMBSEFE-Struktur für die Anwendungsdomäne 91
Abbildung 3.8: ConstructionStructure-Bibliothek 93
Abbildung 3.9: ConstructionStructure-Profil 93
Abbildung 3.10: MachineAssemblyStructure-Bibliothek 95
Abbildung 3.11: TestmachineConstruction-Bibliothek 97
Abbildung 3.12: MechanicalView-Bibliothek 100
Abbildung 3.13: ElectricalView-Bibliothek 101

Abbildung 3.14: HydraulicView-Bibliothek 102
Abbildung 3.15: PneumaticView-Bibliothek 103
Abbildung 3.16: InformaticView-Bibliothek 104
Abbildung 3.17: Bestimmung eines Risikoitems 105
Abbildung 3.18: ISO12100&ISO13849-Bibliothek 107
Abbildung 3.19: ISO12100&ISO13849-Profil 109
Abbildung 3.20: DFT-Bibliothek 114
Abbildung 3.21: DFT-Profil 115
Abbildung 3.22: SBBM-Bibliothek 117
Abbildung 3.23: SBBM-Profil 118
Abbildung 3.24: Anwendungsprofil SBBM 122
Abbildung 3.25: Anwendungsprofil einfaches Beispiel Stereotyp SafetyFunction 123
Abbildung 3.26: Vereinfachtes Architekturkonzept zur Toolkopplung 125
Abbildung 4.1: Beispiel Sicherheitsfunktion BDD 134
Abbildung 4.2: Beispiel Sicherheitsfunktion IBD 134
Abbildung 4.3 Attribut zur farblichen Kennzeichnung der Rückkopplungsergebnisse 139
Abbildung 4.4: MMBSEFE-Ablauf zur Risikobeurteilung 140
Abbildung 4.5: MMBSEFE-Ablauf Bestimmung softwaregesteuerte Risikominderung 141
Abbildung 4.6: Implementierungskonzept zur Toolkopplung von DFTs 144
Abbildung 4.7: XMI-Cluster in LabVIEW 144
Abbildung 4.8: XMI-Export in LabVIEW - G-Code 145
Abbildung 4.9: Fehlerbaum bestimmen und auslesen - G-Code 145
Abbildung 4.10: Beispiel Fehlerbaum 146
Abbildung 4.11: Fehlerbaumtransformation in Galileo-Format - G-Code 146
Abbildung 4.12: Basic-Events in Galileo-Format - G-Code 147
Abbildung 4.13: Ausführen eines Fehlerbaums Storm Docker Container - G-Code 147
Abbildung 4.14: Beispiel-Ergebnisdaten Stormchecker 148
Abbildung 4.15: Ergebnisdaten-Parser Stormchecker - G-Code 148
Abbildung 4.16: Ergebnisdaten-Papyrus-Import DFTs - G-Code 149
Abbildung 4.17: Ergebnisdaten-Papyrus-DFTree-Parser - G-Code 150
Abbildung 4.18: Ergebnisdaten-Papyrus-Requirement-Parser - G-Code 150
Abbildung 4.19: Implementierung SBBM 151
Abbildung 4.20: SISTEMA-Input-Transformation - G-Code 153
Abbildung 4.21: SISTEMA-Input-Transformation Subsystem - G-Code 153
Abbildung 4.22: Implementierung Risikograph 155

Abbildung 4.23: Analyseausführung aller Risikoitems - G-Code 155
Abbildung 4.24: Analyseausführung eines Risikoitems für PL_r und RPZ - G-Code 156
Abbildung 4.25: Analyseausführung Risikoberechnung - G-Code 157
Abbildung 5.1: Batteriezellenprüfmaschine 165
Abbildung 5.2: Grenzen der Maschine - Anforderungen 166
Abbildung 5.3: Grenzen der Maschine - Anwendungskontext 167
Abbildung 5.4: Grenzen der Maschine - Prüfmaschinenteile 168
Abbildung 5.5: Grenzen der Maschine – Anwendungsfälle und Betriebsarten 168
Abbildung 5.6: Portalsystem – BDD zur Stakeholder-Aufteilung 169
Abbildung 5.7: Portalsystem – BDD des Maschinenbauers 170
Abbildung 5.8: Portalsystem – IBD des Maschinenbauers 171
Abbildung 5.9: Portalsystem – BDD des Elektroingenieurs 171
Abbildung 5.10: Portalsystem – IBD des Elektroingenieurs 172
Abbildung 5.11: Bestimmung der Folgen von Gefährdungssituationen 173
Abbildung 5.12: Bestimmung der Gefährdungen BDD 175
Abbildung 5.13: Bestimmung der relevanten Maschinenteile von Gefährdungen 176
Abbildung 5.14: Bestimmung der elektrischen Gefährdungssituationen 176
Abbildung 5.15: Bestimmung der elektrischen Gefährdungsereignisse 177
Abbildung 5.16: Bestimmung der elektrischen Ursachen für Gefährdungsereignisse 177
Abbildung 5.17: Zusammenhang zwischen Gefährdung, Gefährdungssituation und Folgen anhand direkten elektrischen Kontakt 178
Abbildung 5.18: Bewertung des Risikos eines direkten elektrischen Schlags 179
Abbildung 5.19: Identifikation der Maßnahmen gegen direkten elektrischen Schlag 180
Abbildung 5.20: Identifikation der Maßnahmen gegen thermische Strahlung 181
Abbildung 5.21: Identifikation der Maßnahmen gegen Verbrennungen 182
Abbildung 5.22: Identifikation der Maßnahmen gegen einen unerwarteten Anlauf 183
Abbildung 5.23: Grenzen der Maschine – Prüfmaschinenteile nach Risikobeurteilung 186
Abbildung 5.24: Grenzen der Maschine – Anwendungsfälle und Betriebsarten nach Risikobeurteilung 187
Abbildung 5.25: Prüflingsüberhitzungserkennung – Stakeholder-Aufteilung 187
Abbildung 5.26: Prüflingsüberhitzungserkennung – BDD des Maschinenbauers 188
Abbildung 5.27: Prüflingsüberhitzungserkennung – IBD des Maschinenbauers 188
Abbildung 5.28: Prüflingsüberhitzungserkennung – BDD des Elektroingenieurs 189
Abbildung 5.29: Prüflingsüberhitzungserkennung – IBD des Elektroingenieurs 189
Abbildung 5.30: Prüflingsüberhitzungserkennung – BDD des Informatikers 190

Abbildung 5.31: Prüflingsüberhitzungserkennung – Zustandsdiagramm 190
Abbildung 5.32: Prüflingsüberhitzungserkennung – BDD des DFTs 193
Abbildung 5.33: Prüflingsüberhitzungserkennung – IBD des nicht erfüllten DFTs 194
Abbildung 5.34: Prüflingsüberhitzungserkennung – IBD des erfüllten DFTs 194
Abbildung 5.35: Prüflingsüberhitzungserkennung – BDD der SBBM 196
Abbildung 5.36: Prüflingsüberhitzungserkennung – IBD des Blockdiagramms 198
Abbildung A.1: SysML Erweiterung von UML [26] 266
Abbildung A.2: SysML Diagrammtypen [22] 266
Abbildung A.3: 4-Schichten-Metadaten-Architektur abgeleitet von [318] 269
Abbildung A.4: Risikomatrix nach EN 62061 [170] 283
Abbildung A.5:Risikomatrix nach Nohl et al. [332, 333] 284
Abbildung A.6: Risikomatrix (links) und Risikograph (rechts) nach ISO 14121-2 [334] 284
Abbildung A.7: Ablaufdiagramm einer Strukturanalyse [223] 285
Abbildung A.8: Kernkonzept Domain Model 290
Abbildung A.9: RAAML Packet Struktur [16] 291
Abbildung A.10: RAAML FMEA Bibliothek [177] 292
Abbildung A.11: RAAML FTA Profil [177] 292
Abbildung A.12: Erweiterter Anwendungskontext - Anwendungsfälle Maschinenbauer 294
Abbildung A.13: Erweiterter Anwendungskontext - Anwendungsfälle Elektroingenieur 295
Abbildung A.14: Erweiterter Anwendungskontext - Anwendungsfälle Informatiker 296
Abbildung A.15: Detaillierte MMBSEFE-Struktur für Entwicklungsingenieure 298
Abbildung A.16: MachineAssemblyStructure-Profil 307
Abbildung A.17: TestmachineConstruction-Profil 321
Abbildung A.18: MechanicalView-Profil 323
Abbildung A.19: ElectricalView-Profil 326
Abbildung A.20: HydraulicView-Profil 329
Abbildung A.21: PneumaticView-Profil 331
Abbildung A.22: InformaticView-Profil 334
Abbildung A.23: XMI-Input-Transformation aller Elemente - G-Code 363
Abbildung A.24: XMI-Input-Transformation Violated-Abhängigkeiten - G-Code 363
Abbildung A.25: XMI-Input-Transformation Funktionen bestimmen und lesen - G-Code .. 364
Abbildung A.26: Typen von SISTEMA-Blöcken bestimmen & auslesen - G-Code 364
Abbildung A.27: SISTEMA-Input SSM-Dateizusammensetzung - G-Code 365
Abbildung A.28: SISTEMA-Input SSM-Datei erstellen - G-Code 365
Abbildung A.29: SISTEMA-Input SSM-Datei erstellen - G-Code 365

Abbildung A.30: SISTEMA-Output SSM-Datei lesen - G-Code 366
Abbildung A.31: SISTEMA-Output-Transformation - G-Code 366
Abbildung A.32: SISTEMA-Output-Transformation Blöcke - G-Code 367
Abbildung A.33: Ergebnisdaten-Papyrus-Import SBBM - G-Code 368
Abbildung A.34: Ergebnisdaten-Papyrus-SafetyFunction-Parser - G-Code 368
Abbildung A.35: XMI-Input-Transformation Risikodaten lesen - G-Code 369
Abbildung A.36: XMI-Input-Transformation Risikoelement lesen - G-Code 369
Abbildung A.37: XMI-Input-Transformation Risikoitem Korrelation - G-Code 370
Abbildung A.38: Ergebnisdaten-Papyrus-Import Risikograph - G-Code 370
Abbildung A.39: Ergebnisdaten-Papyrus-Import Attribute in Zwischenformat - G-Code 371
Abbildung A.40: Ergebnisdaten-Papyrus-Risikoitem-Parser - G-Code 371
Abbildung A.41: Risikoitem Schneiden 372
Abbildung A.42: Risikoitem statische Aufladung 373
Abbildung A.43: Bewertung des Risikos von thermischer Strahlung 374
Abbildung A.44: Bewertung des Risikos von Verbrennungen 375
Abbildung A.45: Bewertung des Risikos eines unerwarteten Anlaufs 376
Abbildung A.46: Eingangs- und Ausgangsinterface elektrische Signale 377
Abbildung A.47: Modellierung des Nothalt-Systems – Stakeholder-Aufteilung 377
Abbildung A.48: Modellierung des Nothalt-Systems – BDD des Maschinenbauers 377
Abbildung A.49: Modellierung des Nothalt-Systems – IBD des Maschinenbauers 378
Abbildung A.50: Modellierung des Nothalt-Systems – BDD des Elektroingenieurs 378
Abbildung A.51: Modellierung des Nothalt-Systems – IBD des Elektroingenieurs 379
Abbildung A.52: Modellierung des Nothalt-Systems – BDD des Informatikers 379
Abbildung A.53: Modellierung des Nothalt-Systems – Zustandsdiagramm 379
Abbildung A.54: Bewertung des Nothalt-Systems – BDD des DFTs 380
Abbildung A.55: Bewertung des Nothalt-Systems – IBD des DFTs 381
Abbildung A.56: Bewertung des Nothalt-Systems – BDD der SBBM 382
Abbildung A.57: Bewertung des Nothalt-Systems – IBD der SBBM 383
Abbildung A.58: Ext. Prüflingsüberhitzungserkennung – BDD des Maschinenbauers 384
Abbildung A.59: Ext. Prüflingsüberhitzungserkennung – IBD des Maschinenbauers 384
Abbildung A.60: Ext. Prüflingsüberhitzungserkennung – BDD des Elektroingenieurs 384
Abbildung A.61: Ext. Prüflingsüberhitzungserkennung – IBD des Elektroingenieurs 385
Abbildung A.62: Ext. Prüflingsüberhitzungserkennung – Zustandsdiagramm 385
Abbildung A.63: Ext. Prüflingsüberhitzungserkennung – BDD des DFTs 386
Abbildung A.64: Ext. Prüflingsüberhitzungserkennung – BDD der SBBM 387

Abbildung A.65: Modellierung des übergeordneten Zustandsdiagramms 388
Abbildung A.66: Zuhaltung – Zustandsdiagramm 388
Abbildung A.67: Zuhaltung – BDD des DFTs 389
Abbildung A.68: Zuhaltung – IBD des DFTs 390
Abbildung A.69: Zuhaltung – BDD der SBBM 391
Abbildung A.70: Zuhaltung – IBD der SBBM 392
Abbildung A.71: Modellierung des Wartungslichtgitters – Zustandsdiagramm 393
Abbildung A.72: Bewertung des Wartungslichtgitters – BDD des DFTs 394
Abbildung A.73: Bewertung des Wartungslichtgitters – IBD des DFTs 395
Abbildung A.74: Bewertung des Wartungslichtgitters – BDD der SBBM 396
Abbildung A.75: Bewertung des Wartungslichtgitters – IBD der SBBM 397

Tabellenverzeichnis

Tabelle 1.1: Forschungsaufgaben 17
Tabelle 2.1: Ausfallgrenzwerte für die vier Sicherheitsintegritätslevel [5, 34] 34
Tabelle 2.2: Verbleibende Herausforderung bzgl. maschinenbauspezifischer Modellierung .. 43
Tabelle 2.3: Analysen zugeordnet zu Domänen 45
Tabelle 2.4: Merkmale und Darstellung der Kategorien [34] 53
Tabelle 2.5: Zusammenfassung der Veröffentlichungen zur Modellkopplung 62
Tabelle 2.6: Verbleibende Herausforderungen bzgl. Integration der Analysen 67
Tabelle 2.7: Verbleibende Herausforderungen bzgl. MBSE-Prozesse für KMUs 74
Tabelle 2.8: Forschungsaufgaben nach Beobachtung 75
Tabelle 3.1: Verwandte Arbeiten typischen Sicherheitsanalysemethode im Maschinenbau . 111
Tabelle 3.2: Zusammenfassung der Veröffentlichungen zur Toolkopplung 124
Tabelle 3.3: Forschungsaufgaben nach Modellierung 129
Tabelle 4.1: Wertetabelle PL und Risikoprioritätszahl 156
Tabelle 4.2: Implementierte Profil- und Bibliothekselemente 158
Tabelle 4.3: Codeumfang in LabVIEW-Knoten 159
Tabelle 4.4: Forschungsaufgaben nach Implementierung 161
Tabelle 5.1: Berechnung der Ausfallkriterien der Prüflingsüberhitzungserkennung 192
Tabelle 5.2: Erweiterung der Ausfallkriterien der Prüflingsüberhitzungserkennung 196
Tabelle 5.3: Ergebnisse DFT und SBBM 200
Tabelle 5.4: Zusammenfassung zur Zufriedenheit der Stakeholder 213
Tabelle 5.5: Forschungsaufgaben nach Evaluation 216
Tabelle A.1: Beispiele für Gefährdungen [163] 279
Tabelle A.2: Risikoklassen-Matrix zur Klassifizierung von Unfällen [9] 283
Tabelle A.3: Gegenüberstellung Risikoeinschätzung ISO 13849 [34] und EN 62061 [170]. 283
Tabelle A.4: Abschätzung des DC nach ISO 13849 [34] 287
Tabelle A.5: Maßnahmen gegen CCF nach ISO 13849 [34] 288
Tabelle A.6: Werte der Kategorien [34] 289

Formelverzeichnis

Formel 2.1: Exponentialverteilung von Bauteilen 33
Formel 2.2: Berechnung $MTTF_D$ eines Elements [34] 53
Formel 4.1: Berechnung der Risikoprioritätszahl 156
Formel 4.2: Parameterwerte zur Berechnung der Risikoprioritätszahl 156
Formel 5.1: Berechnung λ_D eines Elements [34] 191
Formel A.1: Berechnung $MTTF_D$ Kanäle [34] 289

Listingverzeichnis

Listing 4.1: OCL-Constraint Function von SBBM-Profil 135
Listing 4.2: OCL-Constraint Trigger von SBBM-Profil 135
Listing 4.3: OCL-Constraint Actuator von SBBM-Profil 135
Listing 4.4: OCL-Constraint Subsystem von SBBM-Profil 136
Listing 4.5: OCL-Constraint Input von SBBM-Profil 136
Listing 4.6: OCL-Constraint Logic von SBBM-Profil 136
Listing 4.7: OCL-Constraint Channel von SBBM-Profil 136
Listing 4.8: OCL-Constraint Connector von SBBM-Profil 136
Listing 4.9: OCL-Constraint SafetyFunction von SBBM-Bibliothek 136
Listing 4.10: OCL-Constraint Subsystem von SBBM-Bibliothek 136
Listing 4.11: OCL-Constraint Logic-SB von SBBM-Bibliothek 137
Listing 4.12: OCL-Constraint Channel von SBBM-Bibliothek 137
Listing 4.13: OCL-Constraint Input-B von SBBM-Bibliothek 137
Listing 4.14: CSS-Eigenschaften für UnrepairableBE 138
Listing 4.15: CSS -Eigenschaften zur farbliche Kennzeichnung der Ergebnisse 139
Listing 4.16: XMI-Format Stereotypes in Papyrus 144
Listing 4.17: Fehlerbaum in Galileoformat 146
Listing 4.18: XML-Format SISTEMA für eine Sicherheitsfunktion 152
Listing 5.1: OCL-Constraint für Risikoitem 201

Abkürzungsverzeichnis

AADL	Architecture Analysis and Design Language
BDD	Blockdefinitionsdiagramm
BPMN	Business Process Model and Notation
CAD	Computer Aided Design
CCF	Common Cause Failure (Ausfall aufgrund gemeinsamer Ursachen)
CSS	Cascading Style Sheets
CTMC	Continuous-Time-Markov-Chains
DC	Diagnosedeckungsgrad
DC_{avg}	Mittlerer Diagnosedeckungsgrad
DFT	Dynamic Fault Tree Analysis (dynamische Fehlerbaumanalyse)
DIN	Deutsches Institut für Normung
E/E/PE	elektrische, elektronische und programmierbare elektronische (Systeme)
EMF	Eclipse Modeling Framework
ETA	Event-Tree-Analyse
FMEA	Failure Mode and Effects Analysis
FMECA	Failure Mode, Effect and Criticality Analysis
FMEDA	Failure Mode, Effect and Diagnostic Analysis
FTA	Fault Tree Analysis
FVL	Full Variability Language
GSN	Goal Structuring Notation
HAZOP	Hazard and Operability Study
IBD	Internal-Blockdiagramm
IDs	Identifier
IEC	International Electrotechnical Commission

INCOSE	International Council on Systems Engineering
ISO	Internationale Organisation für Normung
KMUs	Kleine und mittelständische Unternehmen
LML	Lifecycle Modeling Language
LVL	Limited Variability Language
MBSE	Model-Based Systems Engineering
MDA	Model-Driven Architecture
MDSD	Model-Driven Software Development
MgCCF	Maßnahmen gegen Common Cause Failures
MMBSEFE	maschinenbauspezifische MBSE-Framework-Erweiterung
MOF	Meta Object Facility
MTBF	Mean Time Between Failure
MTTF	Mean Time Between Failure
MTTFD	Mean Time to Dangerous Failure
MTTR	Mean-Time-To-Repair
OCL	Object Constraint Language
OMG	Object Management Group
PFD	Probability of Failure
PFDD	Average Probability of Dangerous Failure on Demand
PFHD	Average Probability of Dangerous Failure per Hour
PL	Performance Level
PLr	Performance Level required
RAAML	Risk Analysis & Assessment Modeling Language Specification
RBD	Reliability Blockdiagram
RCM	Reliability Configuration Modelle
RDF	Ratio of Dangerous Failure
SBBM	sicherheitsbezogene Blockdiagrammmethode

SFF	Safe Failure Fraction
SIL	Safety Integrity Level
SLOC	Source Lines of Code
SPSen	speicherprogrammierbare Steuerungen
SRASW	Safety Related Application Software
SRESW	Safety Related Embedded Software
SRP/CS	Safety Related Part of Control System
STPA	Systems Theoretic Process Analysis
SVG	Scalable Vector Graphics
SysML	Systems Modeling Language
UC	UseCase
UML	Unified Modeling Language
UUIDs	Universally Unique Identifier
UWD	Ursache-Wirkungsdiagramm
XMI	XML Metadata Interchange
XML	Extensible Markup Language

1 Einleitung & Motivation

Am 10. Juli 1976 ereignete sich im Norden von Mailand ein schwerwiegender Chemieunfall, der bekannt als Seveso-Katastrophe in die Geschichte einging [1]. Bedingt durch eine unkontrollierte Reaktion überhitzte die Anlage. Zu dieser Zeit waren Anlagen nicht mit automatischen Kühlsystemen oder Warnanlagen ausgestattet, wodurch durch ein Sicherheitsventil hochgiftiges Dioxin austrat [2]. Es entstanden katastrophale Folgen für die Menschen, die Tierwelt und die Natur [2]. Hunderte Hektar Land in den umliegenden Gemeinden wurden vergiftet und tausende Tiere starben, während hunderte Menschen an schwerer Chlorakne erkrankten [1]. Diese Katastrophe gilt als Auslöser für die Verschärfung und Vereinheitlichung von Gesetzen und Vorschriften zum Schutz von Menschen, Anlagen und Umwelt in allen Industriebereichen [1]. Diese Gesetze bilden die Grundlage für die Entwicklung von sicherheitskritischen Systemen und Anwendungen in der funktionalen Sicherheit, insbesondere in Bereichen wie der Automobilindustrie und Medizintechnik, die heute ein solches Design nach Sicherheitskriterien unerlässlich machen.

Die zunehmende Digitalisierung und die dadurch entstehende Verwendung softwarebasierter Funktionen in immer komplexeren Systemen mit wachsenden Softwareanteilen führte nebenher in den 1960er-Jahren zur sogenannten Softwarekrise [3]. Fehlerbehaftete Software verlangte auch im Maschinenbau nach Systementwicklungsprozessen, basierend auf Modellen, um beim Entwurf und der Entwicklung von Systemen die Entstehung von Entwurfsfehlern und Codierungsfehlern zu minimieren und sich auf die Vermeidung systematischer Fehler zu konzentrieren [4].

1.1 Motivation

Die funktionale Sicherheit dient der Erfüllung der notwendigen Eigenschaften von sicherheitskritischen Systemen. Sie befasst sich dabei mit potenziellen Fehler- und Gefahrenquellen und nutzt Designstrategien, um diese aufzudecken, Auswirkungen zu vermeiden oder zu minimieren und die Sicherheit über den gesamten Lebenszyklus einer Maschine zu gewährleisten. Ein sicherheitskritisches System besteht aus einem überwachenden und einem überwachten Teil, wobei der überwachende Teil das Sicherheitssystem darstellt [5, 6, 7, 8, 9, 10, 11]. Die Einführung von Software in das Sicherheitssystem hat die Welt des Ingenieurwesens dabei stark

Model-Based Systems Engineering sicherheitskritischer Anwendungen im Maschinenbau mittelständischer Unternehmen © 2024 by Nick Berezowski is licensed under CC BY 4.0

verändert. Traditionelle Zertifizierungsansätze, die ursprünglich für elektromechanische Systeme entwickelt wurden, sind oft unzureichend, um die Komplexität moderner softwareintensiver Systeme zu bewältigen [12]. Die Software eines sicherheitskritischen Systems wird hierbei als sicherheitskritische Anwendung bezeichnet. Die Betrachtungen dieser Arbeit gelten in erster Linie der funktionalen Sicherheit.

Sicherheitskritische Anwendungen finden sich seit langem in der Avionik, vermehrt auch im Bereich Automotive durch autonome Fahrzeuge, oder der Medizintechnik durch immer komplexere lebenserhaltende Systeme. Die IEC 61508 legt die Basis zur funktionalen Sicherheit jedes elektrischen, elektronischen und programmierbaren elektronischen (E/E/PE) System. Zusätzlich existieren spezialisierte und harmonisierte Standards für verschiedene Domänen, wie dem Maschinenbau, der Prozessindustrie, der Fahrzeugindustrie, der Medizintechnik und der Avionik [13, 14, 15, 16, 17]. Diese dienen der Auslegung sicherheitstechnischer Grundbegriffe und Verfahren für die einzelnen Domänen, indem Synonyme und Äquivalenzen gebildet und domänenspezifische System- und Softwareentwicklungsansätze mit einbezogen werden. Der Anspruch dieser Normensammlung stellt einen eindeutigen Entwicklungsprozess dar, um Entwicklern die Möglichkeit zu geben, Systeme so sicher und anwendungsspezifisch wie möglich zu gestalten. Die Kritikalität eines Systems wird durch das Sicherheitsintegritätslevel, englisch Safety Integrity Level (SIL), definiert und typischerweise in 4 Grundebenen unterteilt [5], abhängig der einzuhaltenden Versagenswahrscheinlichkeiten in der Domäne [14]. Je höher die SIL, desto strenger müssen formale und semiformale Methoden sowie Methoden zur Beherrschung zufälliger und systematischer Fehler Anwendung finden [7]. Je nach SIL werden verschiedene Analysemethoden und -verfahren empfohlen, gefordert oder aus dem Entwicklungsprozess ausgeschlossen. Hierbei müssen sich Ingenieure immer mit dem aktuellen Stand der Technik vertraut machen [5], um Fahrlässigkeit bis hin zum Vorsatz nach Strafgesetzbuch (vgl. bspw. § 26, § 221 oder § 826) auszuschließen.

Während der Entwicklung von Systemen und Anwendungen stellt die Kommunikation zwischen verschiedenen Stakeholdern eine wesentliche Herausforderung dar [18]. Ein dokumentenzentrierter Ansatz erfordert beispielsweise eine manuelle Übersetzung zwischen den Anforderungen in Textform und der Softwaremodelle. Stakeholder eines Bereiches erfassen somit oft nicht die Fähigkeiten und Kompetenzen anderer Stakeholder, da sie lediglich ihre eigenen Entwicklungsmodelle im Blick haben. Dies hat zur Folge, dass Artefakte zwischen den Beteiligten oft nur mündlich ausgetauscht werden und diese von anderen Stakeholdern nicht genutzt oder verstanden werden können [18]. Daraus können Verständnis- und Kommunikationsprobleme entstehen, die wiederum zu überlappenden Umsetzungen mit widersprüchlichen Eigenschaften

führen [18]. Diese gegensätzlichen Standpunkte entstehen durch die unterschiedlichen Interessen und Motivationen der beteiligten Stakeholder. Um diese Probleme zu umgehen und einen stark regulierten Entwicklungsprozess zu schaffen, rücken gemeinsame Modelle, die die Entwicklungsmodelle der einzelnen Stakeholder zusammenführen und als Kommunikationsbasis zwischen Stakeholdern dienen, zunehmend in den Fokus. Hinzu kommen die steigenden Softwareanteile und Komponenten, die Fehler aufwerfen und die Sicherheit der Software beeinträchtigen. Modelle geben dabei die Möglichkeit, rechtliche Risiken zu senken. Für die Entwicklung sicherheitskritischer Anwendungen hat sich der Schwerpunkt daher, vor allem bei Anforderungen zu höheren SIL, von einem dokumentenbasierten Ansatz zu einem modellbasierten Ansatz verschoben [19].

2001 veröffentlichte die Object Management Group (OMG) „Model-Driven Architecture: Vision, Standards and Emerging Technologies“ [20], zur Begegnung der Softwarekrise. Die Model-Driven Architecture (MDA) ist ein Ansatz für Model-Driven Software Development (MDSD), definiert durch die OMG, mit dem Ziel, die Entwicklung von Software möglichst sicher zu gestalten. Über ihre eigenen Standardisierungen zu Unified Modeling Language (UML), Meta Object Facility (MOF), XML Metadata Interchange (XMI) und weiteren Modellen bauten sie ein modellgetriebenes Framework auf [20], das die Grundlage für weitere Forschungen und Entwicklungen legen sollte. Die Studie „Survey of Model-Based Systems Engineering Methodologies“ [21] vom International Council on Systems Engineering (INCOSE) aus dem Jahr 2008 beschreibt bereits einen Model-Based Systems Engineering (MBSE)-Ansatz für die Systems Modeling Language (SysML) auf Basis der MDA Initiative der OMG. SysML wurde in der ersten Version 2007 von der OMG herausgegeben und bildet ein speziell für das Systemdesign technischer Systeme geeignetes erweitertes Subset von UML, das Anforderungsanalysen, Strukturbeschreibungen und Verhaltensbeschreibungen vereinfachen soll [22]. SysML-Modelle können mit UML-Profilen zusätzlich angereichert werden, um weitere Informationen zu integrieren. Solche Profile können unter anderem zur Annotation und Beurteilung der funktionalen Sicherheit dienen.

Zur Implementierung von MBSE wurden im Laufe der Jahre mehrere Modellierungssprachen entwickelt, darunter Architecture Analysis and Design Language (AADL) [23], Lifecycle Modeling Language (LML) [24], UML [25] und SysML [26]. Abhängig der Einsatzgebiete werden verschiedene Eigenschaften dieser Sprachen benötigt [27]. Nach Veröffentlichungen wie D’Ambrosio et al. [28] kann SysML heute jedoch als die bevorzugte Sprache zur Systemmodellierung angesehen werden. In der Vergangenheit fanden hierfür beispielsweise die AADL

[23], Modell Checking Language NuSMV [29], die formale Spezifikationssprache AltaRica [30] oder HiP-HOPS [31] Einsatz.

Die Zusammenführung von Anforderungen, Modellierung, Simulation, Implementierung und Test über MBSE kann das Vertrauen im Vergleich zum traditionellen dokumentenbasierten Ansatz erhöhen und koppelt den Entwurfsprozess mit allen Stakeholdern, deren Modellen und Sichtweisen. Zur effektiven Gestaltung müssen die gemeinsamen Modelle so ausgelegt werden, dass sie für die Domäne eindeutig und für Stakeholder verständlich sind, um die Umsetzung der Entwürfe der Systemmodelle zu unterstützen. Klassisch spricht man bei Entwicklungsmodellen in der Maschinenindustrie von Computer Aided Design (CAD) Zeichnungen für die mechanische und konstruktive Gestaltung, von Schaltplänen für die elektrische, elektronische, pneumatische und hydraulische Gestaltung und von UML-Diagrammen für die informationstechnische Gestaltung. Zur Unterstützung können UML-Profile dienen, die die domänenspezifischen Frameworks bilden und SysML in entsprechenden Modellierungsumgebungen erweitern [32, 33].

Die sicherheitsbezogene Anwendungssoftware, englisch Safety Related Application Software, (SRASW) bildet maschinenspezifische Funktionalitäten durch die Implementierung von Sicherheitsfunktionen in Programmiersprachen mit eingeschränktem Sprachumfang, englisch Limited Variability Language (LVL), und Programmiersprachen mit nicht eingeschränktem Sprachumfang, englisch Full Variability Language (FVL) ab [34]. Ausgeführt werden die Funktionen meist auf speicherprogrammierbaren Steuerungen (SPSen) [34, 35, 36]. IEC 61131-3 gibt typische Beispiele für LVL an, wie Funktionsblockdiagramme und sequentielle Funktionstabellen [35], die ähnlich endlichen Automaten aufgebaut werden können. Zusätzlich kann mit FVL und einer sicheren Sprachenteilmenge, wie beispielsweise MISRA-C oder -C++ ebenso eine Konformität erreicht werden [11, 37, 38].

Zur Bewertung der funktionalen Sicherheit eines Entwurfs finden probabilistische Sicherheitsanalysemethoden, wie die Fehlerbaumanalyse, englisch Fault Tree Analysis (FTA), die Ausfallarten- und Auswirkungsanalyse, englisch Failure Mode and Effects Analysis (FMEA), und andere komplexere und dynamischere Verfahren Einsatz [11, 39, 40]. Neben FTA und FMEA existieren Sicherheitsanalysemethoden, wie die FTA-Erweiterung für dynamische Fehlerbäume, englisch Dynamic Fault Tree Analysis (DFT), die Event-Tree-Analyse (ETA), die Hazard and Operability Study (HAZOP), die Systems Theoretic Process Analysis (STPA) und bezogen auf die Maschinenindustrie sicherheitsbezogene Blockdiagrammmethode (SBBM) oder die Risikographmethode [7, 34]. Die für die Bewertung erforderlichen Daten ergeben sich

aus den mechanischen und elektronischen Systemmodellen der Stakeholder, wodurch diese detailliert in der gemeinsamen Modellebene integriert werden müssen. Im Zusammenhang mit MBSE ist somit ein weiterer Stakeholder, der Sicherheitsingenieur, in die gemeinsame Modellebene mit einzubinden, um Risikobeurteilungen durchzuführen und so die Systemmodelle zu überprüfen und zu bewerten.

Typischerweise findet die Bewertung der Systemmodelle in separaten Analysemodellen statt. In Verbindung mit MBSE sind Abhängigkeiten zu den Systemmodellen aufzubauen. Bei der Untersuchung modellbasierter Engineering-Methoden für die funktionale Sicherheit zeigt sich allerdings, dass der Einsatz von UML-Profilen zur Verbesserung der Systemmodellierung beiträgt [41, 42, 43]. Die Integration bestehender Sicherheitsanalysen in SysML, wie von Clegg et al. in [44], [45] und [46] diskutiert, ermöglicht dabei eine nahtlose Verbindung zwischen System- und Sicherheitsmodellierung. Ein Ansatz aus der Eisenbahnindustrie, beschrieben in Cancila et al. [41] aus 2009, illustriert den Einsatz von UML und Modeling and Analysis of Real-Time and Embedded Systems (MARTE) zur Förderung der Wiederverwendung bestehender Analyseprofile. Ähnliche Ansätze existieren ebenso in anderen Domänen wie der Avionik. Jiang et al. [42] verwendet hierbei MBSE für eine funktionale Gefahrenanalyse des Bremssystems ziviler Flugzeuge. Allerdings finden sich auch andere Ansätze wie Lui et al. [47], der SysML und MARTE als zu schwach für die formale semantische Betrachtung in der Avionik ansieht und daher eine eigene Sprache, Safe State Machines, für das Safety Behavior entwickelte. Obwohl diese Ansätze hauptsächlich die theoretischen Grundlagen ohne direkte Implementierung der Analysemethoden beleuchten und die UML-Profile oft keine Wiederverwendbarkeit für andere Analysemethoden zeigen [43], bieten sie wertvolle Einblicke in die Nutzung von modellbasierten Techniken zur Risikoanalyse und wie stringende Verfahren über spezifische Sprachen und Werkzeuge entwickelt werden können, um die Genauigkeit und Sicherheit in der Modellierung sicherheitskritischer Systeme zu verbessern. Die Übertragung solcher modellbasierten Methoden auf die spezifischen Anforderungen der Maschinenindustrie könnte daher einen bedeutsamen Fortschritt darstellen. Die OMG Risk Analysis & Assessment Modeling Language Specification (RAAML) [48] bietet bereits einen möglichen zentralen Lösungsansatz für mehrere Domänen an. Ihre erste Beta-Version 1.0 wurde am 15. September 2021 offiziell veröffentlicht [49]. Sie befasst sich mit Profilerweiterungen für SysML, die grundlegende allgemeine Begriffe für die Definition des Informationsmodells und grundlegende Analysemethoden für MBSE für die funktionale Sicherheit einführen. Die grundlegenden Analysemethoden, wie FMEA und FTA beschränken sich jedoch auf statische probabilistische Berechnungen [49],

die für den praktischen Einsatz in anspruchsvollen Bereichen nur begrenzt geeignet sind. Aktuell gibt es nur wenige weiterführende Arbeiten wie Diatte et al. [50]. Auf Domänen zugeschnittene Erweiterungen finden sich unter anderem für Avionik [51], Cyber-Security [52], Medizintechnik [53] und Prozessindustrie [54], nicht jedoch in der Maschinenindustrie. Dies zeigt einen wesentlichen Vorteil des RAAML-Ansatzes; die einfache Erweiterbarkeit mittels zentral angelegter Profile.

Kleine und mittelständische Unternehmen (KMUs) sind seit Jahren ein wichtiges Thema in der Politik und der öffentlichen Diskussion. Ein Grund dafür ist ihre große wirtschaftliche und gesellschaftliche Bedeutung, insbesondere in Europa. So beschäftigt der Mittelstand die große Mehrheit der Beschäftigten in Deutschland und 67 Prozent aller Beschäftigten in Europa [55]. 2013 fielen über 99 Prozent aller in der EU tätigen Unternehmen unter die Definition von KMUs [56]. Somit verwundert es auch nicht, dass KMUs im Kontext der Entwicklung von Maschinensteuerungen in der Maschinenindustrie, speziell im Sondermaschinenbau, ebenso eine zentrale Rolle spielen. Durch ihre flexiblen Strukturen, Innovationsfähigkeit und Erfahrung bieten sie eine wichtige Stütze der Wirtschaft [57]. Im Kontext des digitalen Wandels finden sich diese Unternehmen jedoch im Spannungsfeld zwischen der langfristigen strategischen Ausrichtung und tiefgreifenden Veränderungen durch eine zunehmende und vor allem rasch voranschreitende Digitalisierung wieder [57]. Die rasante Entwicklung und Integration digitaler Technologien bringt eine Fülle neuer Möglichkeiten in der Projekt- und Produktionsplanung mit sich, für die viele KMUs jedoch nicht ausreichend gerüstet sind. Häufig fehlt es an essenziellem Knowhow sowie an den finanziellen und technologischen Ressourcen, um fortschrittliche Methoden wie MBSE effektiv zu implementieren [58]. Es existieren wenige Forschungsarbeiten, die sich auf die Bedürfnisse von KMUs in der Maschinensicherheit beziehen. Diese Lücke in der Forschung ist problematisch, da KMUs in diesem Fall auf allgemeine, weniger spezifische Methoden angewiesen sind, die nicht vollständig auf ihre Bedürfnisse zugeschnitten sind [59]. Zudem sind die in hochregulierten Bereichen wie der Luft- und Raumfahrt etablierten Verfahren für die Entwicklung sicherheitskritischer Systeme für KMUs häufig weniger gut geeignet, hauptsächlich aufgrund der prohibitiv hohen Kosten und der Notwendigkeit spezialisierten Knowhows, das in KMUs oft nicht vorhanden ist [60]. Dies zwingt Entwickler dazu, sich auf allgemeinere, weniger spezialisierte Vorgehensweisen zu stützen, die große Interpretationsspielräume bieten und somit zu unzureichenden oder fehleranfälligen Lösungen führen können.

Darüber hinaus erkennen KMUs die Notwendigkeit einer digitalen Transformation an, um wettbewerbsfähig zu bleiben, doch die Umsetzung dieser Transformation ist oft durch unzureichende interne Kompetenzen und externe Unterstützung limitiert [61]. Dies unterstreicht die Dringlichkeit, maßgeschneiderte Lösungen und Unterstützungsmechanismen zu entwickeln, die es KMUs ermöglichen, die Vorteile der Digitalisierung voll auszuschöpfen und ihre Systeme und Prozesse sicher und effektiv zu gestalten. Arbeiten wie Parente et al. [62] heben dabei die Rolle der Zusammenarbeit zwischen Universitäten und KMUs bei der digitalen Transformation hervor, um Technologien wie MBSE effektiv zu nutzen. Somit entsteht die Motivation, praxisnahe und kosteneffiziente Ansätze für die Implementierung von MBSE in KMUs zu entwickeln, die sowohl die technologischen als auch die ökonomischen Herausforderungen adressieren und den Unternehmen ermöglichen, ihre Wettbewerbsposition zu stärken und die Sicherheit ihrer Systeme zu erhöhen.

Eine weitere Rolle für die Motivation dieser Arbeit spielt das anwendungsspezifische berufliche Umfeld. Während des Forschungsprojektes wurde einer Anstellung als Projektleiter bei dem Unternehmen ProNES Automation GmbH (ProNES) [63] aus Landau nachgegangen. Das KMU ist in der Maschinenindustrie tätig, in der es sich auf den Entwurf und die Entwicklung von speziellen Prüfmaschinen für Batteriesysteme der nächsten Generation spezialisiert hat [64]. Dabei liegt ein besonderer Fokus des Unternehmens auf Sondermaschinen, die auf die individuellen Anforderungen der Kunden zugeschnitten werden müssen. Um den Anforderungen gerecht zu werden, greift das Unternehmen auf bewährte praxisnahe Vorgehensweisen zurück und nutzt spezifische Sicherheitsanalysemethoden, wie Risikographen oder sicherheitsbezogene Blockdiagramme zur Beurteilung der Lösungen. Hierbei kommt jedoch lediglich ein dokumentenbasierter Ansatz zu tragen, der sich negativ auf die Qualität der Batterieprüfmaschinen auswirken kann. Diese Arbeit kann somit zur Weiterentwicklung des Vorgehens und zur Verbesserung der Wettbewerbsfähigkeit des Unternehmens beitragen.

1.2 Problemfelder

Der vorangegangene Abschnitt zeigt, wie komplex sich die Entwicklung sicherheitskritischer Anwendungen gestalten kann. In diesem Abschnitt werden die Themen beschrieben, die im Rahmen der Entwicklung sicherheitskritischer Systeme im Anwendungsbereich Maschinensicherheit derzeit mit Defiziten behaftet sind, was zur Definition der Problemfelder dieser Arbeit führt.

1.2.1 Probleme bezogen auf Modellierung

In Anbetracht der Entwicklung aktueller Maschinen stehen Digitalisierung, Kommunikation und Flexibilität einzelner Komponenten im Vordergrund. Statische Prozesse ohne anpassungsfähige Fertigungsabläufe, Betriebsmittel und Teilnehmer sollen bald der Vergangenheit angehören [65]. Technologisch spricht die Industrie von der vierten industriellen Revolution (Industrie 4.0). Zur Vernetzung der gesamten Betriebsstruktur von Unternehmen sollen Automatisierungssysteme nicht mehr nur einzelne Prozesse steuern, sondern eigenständig Informationen zwischen Betriebsbereichen austauschen und ganze Arbeitsabläufe koordinieren [66]. Nicht nur durch diese steigende Digitalisierung und Automatisierung weitet sich das Feld der funktionalen Sicherheit immer weiter aus. Systeme, wie automatische Ofensteuerungen oder Fahrerassistenzsysteme, finden sich längst in unserem Alltag wieder. Sicherheitskritische Systeme werden zwangsweise komplexer und besitzen mehr Teilsysteme mit zunehmenden Softwareanteilen, die miteinander über Sensoren, Aktoren und Verarbeitungseinheiten interagieren.

Um dieser Herausforderung gewachsen zu sein, wird ein stark reglementierter Entwicklungsprozess benötigt, der die Grundlage für eine fehlersichere Softwareentwicklung im Sinne von MDSD und Systementwicklung im Sinne von MBSE liefert. Mit der bis heute üblichen dokumentenzentrierten Entwicklung halten Unternehmen jedoch aufgrund mangelnder Innovationsfähigkeiten an einer schlechten Engineering-Praxis fest [67]. Folgen sind teure Nacharbeiten bei inkonsistentem Design, da die Entwicklungsmodelle nur schwer verglichen werden können, und ein erhöhter Testaufwand aufgrund eines lückenhaften Entwicklungsprozesses.

Nur durch den Einsatz einer modellbasierten Entwicklung allein und beispielsweise einer manuellen Verbindung zwischen verschiedenen Modellen verschiedener Stakeholder lassen sich die Probleme allerdings nicht umgehen. Hier findet nur eine lose Verbindung zwischen verschiedenen Systemmodellen und Analysemodellen statt, indem unterschiedliche Modellebenen oder sogar verschiedene Modellierungssprachen zum Einsatz kommen [68, 69, 70, 71]. Jeder Stakeholder arbeitet in seiner eigenen Modellwelt. Die entstehende lose verbundene Datenbasis kann kostspielige Folgen haben, wenn die Stakeholder Änderungen an den Systemmodellen und an den Analysemodellen nicht direkt synchronisieren [72]. Dieses Problem verschärft sich zusätzlich, existiert kein eindeutiges Framework zur Darstellung der Stakeholder-spezifischen Artefakte und deren Korrelation zu Modellen der anderen Stakeholder. Dies ist nicht nur zeitaufwändig und fehleranfällig, sondern führt auch zu einer mangelnden Rückverfolgbarkeit zwischen dem Systemmodell und dem Sicherheitsmodell. Wie schwierig es ist, funktionale und nicht-funktionale Eigenschaften gemeinsam zu betrachten, zeigt das EU-Projekt AQUAS als Beispiel [73]. Hier wurden während des Entwurfsprozesses einzelne Synchronisationspunkte

eingeführt, um Safety, Security und Performance zu integrieren. Darüber hinaus findet eine Harmonisierung von Terminologien und Standards für verschiedene Domänen statt. Eine vollständige Integration von Methoden und Werkzeugen ist jedoch nicht erreicht worden [73].

Einige Ansätze beziehen sich auf eine enge Verbindung der Modelle, indem einzelne Sicherheitsanalysemethoden Bestandteil einer gemeinsamen Modellebene sind [74, 75, 76]. Dies fördert zwar die Konsistenz der Modelle, doch entsteht ein Mangel an einfacher Erweiterbarkeit, sofern der Entwicklungsansatz nicht auf grundlegenden Begriffen beruht [77], wodurch nur mit deutlichem Aufwand weitere Analysemethoden und deren Artefakte in Korrelation gebracht werden können.

1.2.2 Probleme bezogen auf Unterstützung in einzelnen Domänen

Entwicklungsschritte sind oft stark angepasst auf die jeweilige Domäne [78]. Zum einen kommen hier oft sehr unterschiedliche Hardwarekomponenten und Softwarelösungen zum Einsatz, und zum anderen müssen Entwicklungsprozesse anders ausgelegt werden. Hierbei können beispielsweise Lösungen aus der Automobilindustrie, allein aufgrund des Einsatzes unterschiedlicher Programmiersprachen, nicht eins zu eins für die Maschinenindustrie eingesetzt werden [78]. Aspekte wie Betriebssituationen und Reaktionen auf Fehlverhalten variieren stark [14, 34]. Nicht nur deshalb legt die IEC 61508 lediglich grundlegende Aspekte fest, die durch die einzelnen domänenspezifischen Standards konkretisiert und erweitert werden müssen [5]. Dies hat einen erheblichen Einfluss auf den Einsatz und die Auslegung von Sicherheitsanalysemethoden. Beispielsweise kommen angepasste SIL, unterschiedliche Verteilungsfunktionen zur Berechnung von probabilistischen Größen und Berechnungen zur Risikobeurteilung zum Einsatz. So findet beispielsweise STPA und HAZOP hauptsächlich in der Automobilindustrie Einsatz oder SBBM und Risikographen in der Maschinen- und Prozessindustrie.

1.2.3 Probleme bezogen auf die Maschinenindustrie

Die Innovation der Entwicklung sicherheitskritischer Anwendungen in der Maschinenindustrie ist historisch betrachtet lange vernachlässigt worden [79]. Hierbei lag der Fokus oft auf komplexeren Domänen, wie die Luft- & Raumfahrtindustrie oder die Automobilindustrie, die eine Vorreiterstellung einnehmen, da es sich um Systementwicklungen handelt, die bereits bei geringer Komplexität hohen Sicherheitsmaßstäben gerecht werden müssen. Als in der Maschinenindustrie noch rein mechanische Systeme entwickelt wurden, kamen in der Luft- und Raumfahrt längst elektrische/elektronische Komponenten zum Einsatz [80]. Allerdings wurden über die letzten Jahrzehnte auch sicherheitskritische Anwendungen in der Maschinenindustrie im-

mer komplexer. Von rein mechanischen Komponenten ging die Entwicklung über elektromechanische, pneumatische und hydraulische Komponenten zu heute eingebetteten programmierbaren elektronischen Systemen über. Mit jeder Stufe wurden weitere Stakeholder involviert, und es haben sich neue Herangehensweisen und Lösungsstrategien entwickelt.

Bis heute existieren wenige MBSE-Ansätze für den Maschinenbau und deren Sicherheitsstandards, wie ISO 12100 oder ISO 13849-1, doch entwickelt sich eine verstärkte Relevanz aufgrund der zunehmend komplexeren Systeme und vermehrten Softwareanteile. Ebenso kümmert sich der oben erwähnte RAAML-Ansatz weniger um die Maschinenindustrie, sondern konzentriert sich auf Sicherheitsanalysemethoden tragvollerer Domänen [48]. So finden sich in RAAML keine Begriffsklärungen oder Synonyme für die Maschinensicherheit wieder, sondern lediglich grundlegende domänenübergreifende Begrifflichkeiten, die als Grundlage für einzelne Domänen verwendet werden können.

Ein wichtiger Faktor bei der Systementwicklung ist die Entwicklungszeit. Die Kenntnisse von National Instruments (NI) LabVIEW spielen dabei eine zentrale Rolle in der Maschinenindustrie und den dort etablierten Testumgebungen [81, 82]. NI LabVIEW wird häufig als Rapid-Prototyping-Werkzeug [83, 84] im Maschinenbau eingesetzt. Doch ist LabVIEW eher ungeeignet für Softwareentwicklungen in der funktionalen Sicherheit und erfordert daher einen extrem hohen Testaufwand, der schnellen Entwicklungsmethoden entgegenwirkt [85]. Meist finden daher eher LVL wie Kontaktplan und Funktionsblockdiagramme Einsatz [86, 87]. Doch auch hier besteht oft Unklarheit über Einsatzmöglichkeiten mit MDSD. Aufgrund von Unsicherheiten über die Anwendbarkeit und den erforderlichen Aufwand zur Umsetzung von MDSD-Ansätzen wird der klassische dokumentenzentrierte Entwicklungsansatz bevorzugt, statt UML- oder SysML-basierte Notationen zu nutzen [36]. Dabei zeigen Übersichtsarbeiten zu MBSE wie Cederbladh et al. [88] im Vergleich zu traditionellen, dokumentzentrierten Workflows signifikante Vorteile in der Validierung und Verifizierung, insbesondere durch die Verfügbarkeit von Modellen bereits in den frühen Entwicklungsphasen.

1.2.4 Probleme bezogen auf die Toolkommunikation

Mit der Annahme einer SysML-basierten Notation für ein integriertes Modell aller Stakeholder entsteht ein weiteres Problem in Bezug zur Kommunikation zu externen Werkzeugen der Stakeholder [89]. Die Unterstützung und Verbindung dieser Werkzeuge benötigt ein abstraktes gemeinsames Komponenten- und Subsystemmodell, welches die Grundlage für die Sicherheitsanalysemethoden bilden kann und gleichzeitig die Stakeholder-spezifischen Strukturen, Modelle und Entwicklungswerkzeuge der Mechanik und Elektrotechnik unterstützt [89]. Die

Struktur der entstehenden annotierten Modelle bedarf einer so feinen Darstellbarkeit, damit die spezifischen Entwicklungswerkzeuge der Fachgebiete diese weiter entwickeln können und, bezogen auf die zu entwickelnde Software, die Komponentenmodelle innerhalb von SysML weiter verfeinert werden können [90], was zur Verhaltensspezifikation in Form von Zustandsautomaten beitragen kann. SysML-basierte MBSE-Werkzeuge finden sich in NoMagic MagicDraw [91], Modelio SysML Architect [92] oder Eclipse Papyrus [93] und weisen zudem Import- und Exportfunktionen mit einfach zu interpretierenden XML-Formaten auf. Findet eine unzureichende Integration der Modelle und Werkzeuge einzelner Stakeholder statt, führt dies bei der Weiterentwicklung der Modelle in den einzelnen Werkzeugen der Entwickler zu Inkonsistenzen in Bezug zu den gemeinsamen Modellen im MBSE-Werkzeug [89, 90].

Eine solche lose Verbindung zu Werkzeugen, in der einzelne Entwicklungswerkzeuge über Transformatoren an die gemeinsamen Modelle angebunden werden, lässt sich ebenso auf die Anbindung von Analysewerkzeugen übertragen [94, 95, 96]. Allerdings entsteht hier eine kumulative Wirkung dieser Einschränkungen [96]. Sicherheitsbewertungen eines Systems erfordern erhebliche Nacharbeit, sofern sich Spezifikationen oder Umsetzungen in den Werkzeugen ändern und diese Änderungen nicht synchronisiert werden, was Zeitaufwand und Kosten erhöht [97]. Darüber hinaus hat sich gezeigt, dass die Kosten für die Behebung von Entwurfsfehlern drastisch ansteigen, wenn ein System die Lebenszyklusphasen durchläuft [98]. Die frühzeitige Erkennung von Konstruktionsfehlern kann die Auswirkungen auf die Projektzeitpläne drastisch reduzieren [99].

Ein anderes Problem kann bei der engen Verbindung der Entwicklungs- und Analysewerkzeuge einzelner Stakeholder entstehen, indem diese direkt in die MBSE-Umgebung integriert werden. CAD-Modelle können in UML nur abstrakt, und nicht in den für die Stakeholder üblichen Darstellungsformen dargestellt werden, da es sich bei UML um kein Modell zur Darstellung von mehrdimensionalen Konstruktionen handelt [32]. Sicherheitsanalysewerkzeuge im Gegenzug müssen einen Nachweis zur Validierung erbringen, da es sich um Werkzeuge handelt, die die Fehlersicherheit eines Designs nachweisen [11]. Abhängig von der Komplexität von abzubildenden Analysewerkzeugen und deren Algorithmen kann sich eine solche Validierung oder Zertifizierung durchaus aufwendig gestalten.

1.2.5 Probleme bezogen auf KMUs

Kleine und mittelständische Unternehmen (KMUs) stehen gegenwärtig dem Druck beider, inländischer und globaler Wettbewerber gegenüber [100]. Ihre Fähigkeiten im Umgang mit engen

Kundenbeziehungen [101] oder ihre Fähigkeit, Nachfragen schnell vorherzusehen [102], ermöglichen es ihnen, durch die Einführung von Innovationen neue Wettbewerbsvorteile zu gewinnen und in neue Geschäftsfelder vorzustoßen [103]. Die eigenverantwortliche Entwicklung sicherheitskritischer Systeme und die Durchführung von Analysen zur funktionalen Sicherheit stellen eine große Herausforderung dar. Die hohen Anforderungen moderner Systeme erfordern semi-formale Methoden, die die Basis für ein modellbasiertes Design bilden, um die zunehmende Umsetzung in Software zu unterstützen und den hohen Fehlerpotentialen entgegenzuwirken [11]. Um sich an die schnell fortschreitende Welt und steigende Komplexität in Bezug auf funktionale Sicherheit anzupassen und gleichzeitig wettbewerbsfähig zu bleiben, scheint es für KMUs somit fast schon notwendig basierend auf Modellen in UML und SysML zu entwickeln. MDSD-Ansätze bilden hierfür einen möglichen Lösungsansatz. Der Einsatz verstärkt allerdings die Notwendigkeit für KMUs entsprechende Entwicklungsumgebungen einzusetzen, die auf ihre Bedürfnisse zugeschnitten sind [58].

Ein großes Problem stellen jedoch ihre begrenzten Ressourcen dar [101, 104, 105], wobei Untersuchungen häufig auf einen Mangel an zeitlichen, finanziellen und personellen Ressourcen hinweisen [103, 106], die vor allem im Tagesgeschäft Innovationen wie MBSE begrenzen. Dies wird besonders in Bezug auf die Kosten für die Anschaffung neuer Werkzeuge und das erforderliche Training deutlich. Grundsätzlich stehen große Unternehmen vor vergleichbaren technologischen Herausforderungen. Die strategischen Herausforderungen können sich aber signifikant unterscheiden, weil große Unternehmen eine breitere Marktabdeckung erzielen können und häufig mehr spezialisierte Ressourcen für Strategie- und Organisationsentwicklung haben [107]. Dadurch fällt es KMUs oft schwer, die gleichen Praktiken anzuwenden, die in größeren Unternehmen etabliert sind, was die Schwierigkeiten erhöht effizient zu arbeiten [108]. Studien wie Parrott [109] zeigen auf, wie durch die Anwendung von MBSE die Entwicklungszeiten verkürzt und die Datenqualität verbessert werden kann, allerdings erfordert dies eine robuste Konfigurationsverwaltung und eine Anpassung der vorhandenen Werkzeuge.

Aus diesen Gründen beruhen Engineering-Prozesse und etablierte Vorgehensweisen in KMUs meist auf Lösungsstrategien, die schnell Lösungsansätze bereitstellen und implementieren. Darüber hinaus sind standardisierte MBSE-Prozesse und -Werkzeuge oft sehr komplex und schwer in einem KMU-Umfeld anzuwenden [58]. KMUs können sich die Anschaffung neuer Werkzeuge und Training zur Einarbeitung in neue Werkzeuge finanziell oft nicht leisten [110]. Daher finden betriebsbewährte Werkzeuge zur Modellierung der Entwicklungsmodelle Einsatz, die nicht oder nur schwer im Kontext der MBSE verwendet werden können.

Ein weiterer Faktor ist das oft fehlende Knowhow im Hinblick auf die Analyse der funktionalen Sicherheit und die dabei verwendeten Methoden und Werkzeuge [111]. Dies führt erfahrungsbedingt in der Praxis dazu, dass teure, vorzertifizierte Hardware-Lösungen zugekauft werden, die bereits hinsichtlich ihrer Risiken und Gefährdungen bewertet wurden, jedoch negative Auswirkungen auf die Gewinnspanne haben können.

1.2.6 Problembeschreibung (PB) und zusammenfassende Herausforderung

Unter den Problemfeldern im Bereich des MBSE sicherheitskritischer Anwendungen im Maschinenbau mittelständischer Unternehmen ist ein Hauptproblem gemeinsam: eine einfache Anwendung domänenspezifischer Entwicklungsmodelle und Sicherheitsanalysemethoden in einem MBSE-Framework. Dies führt zu Herausforderungen in Bezug auf Modellierung, domänenspezifischer Anwendbarkeit, Toolkommunikation und Effektivität für KMUs.

Eine lose Verbindung zwischen Modellen und Werkzeugen oder sogar eine dokumentenbasierte Entwicklung, wie im vorgestellten KMU ProNES, kann leicht zu inkonsistenten Datenmodellen der einzelnen Stakeholder führen. Jedoch durch den Einsatz eines nicht auf die Domäne zugeschnittenen MBSE-Frameworks entsteht ein schwer integrierbares Datenmodell, das sich ebenso negativ auf die Entwicklungsergebnisse und auf die Effektivität eines ressourcenknappen KMU auswirkt.

Die Motivation für diese Arbeit basiert daher auf der Verbesserung der Anwendbarkeit von MBSE sicherheitskritischer Anwendungen im Maschinenbau in KMUs. Um dies zu erreichen, können die folgenden Probleme identifiziert werden:

PB1: Fehlende maschinenbauspezifische Integration von System-, Software- und sicherheitsbezogenen Modellen

Unter den Herausforderungen bei der Modellierung von Stakeholder-spezifischen Artefakten in der Entwicklung von Maschinen, speziell Prüfmaschinen im Sondermaschinenbau, ist die Integration verschiedener domänenspezifischer Begriffe in einem gemeinsamen Framework innerhalb von SysML ein bedeutendes Problem. Bestehende Ansätze sind oft unzureichend auf die Maschinenindustrie und deren spezielle Anforderungen wie Analysemethoden abgestimmt und beinhalten keine grundlegenden Begriffe zur Annotation maschinenbauspezifischer Aspekte. Derzeit ist es nicht möglich, Synonyme, Äquivalenzen, Verallgemeinerungen, Spezialisierungen einzelner Stakeholder sowie Gemeinsamkeiten und die Kommunikation unter Stakeholdern innerhalb der Domäne der Maschinenindustrie zu nutzen.

PB2: Fehlende Integration der Analysemethoden aus dem Maschinenbau

Aktuell besteht eine große Herausforderung bei der Integration verschiedener Funktionen und Eigenschaften relevanter Sicherheitsanalysemethoden in ein gemeinsames MBSE-Framework. Die vorhandenen Frameworks und Modelle berücksichtigen maschinenbauspezifische Analysemethoden nur unzureichend und bieten oft keine robusten Lösungen für die Integration solcher spezialisierten Methoden. Es fehlt an einer allgemeingültigen, wiederverwendbaren Erweiterung für domänenspezifische Analysemethoden der Maschinenindustrie.

PB3: Fehlende MBSE-Prozesse für KMUs

MBSE-Werkzeuge und -Modelle sind aktuell nur schwer in Prozesse von KMUs integrierbar. Die besonderen Bedürfnisse von KMUs, wie begrenzte Ressourcen und spezifische Anforderungen an die Flexibilität und Skalierbarkeit der Prozesse, werden oft nicht ausreichend berücksichtigt. Es fehlt an einem speziell auf die Bedürfnisse von KMUs zugeschnittenen anwendbaren Konzept, das die Integration etablierter Werkzeuge in ein MBSE-Framework sowie eine ressourcenschonende Anpassung der Entwicklungsprozesse ermöglicht.

Zusammenfassung des Gesamtziels und der Herausforderung

Fasst man diese Problemstellungen zusammen, so wird das übergeordnete Ziel sichtbar, das allen skizzierten Problembereichen gemeinsam ist.

Aktuelle MBSE-Frameworks müssen erweitert werden, um ein einfaches, verständliches, effektives und integriertes MBSE-Framework für die Maschinensicherheit und den im Maschinenbau agierenden KMU zu schaffen. Als Kernaufgabe dazu müssen generelle Begrifflichkeiten und domänenspezifische Sicherheitsaspekte wie Sicherheitsanalysemethoden in die Ebene der Systemmodelle integriert werden.

Die Herausforderung besteht somit in der Verbesserung der Domänenunterstützung über für Stakeholder selbsterklärende Modelle innerhalb eines für die jeweilige Domäne zuschneidbaren Frameworks. Im Kontext dieser Arbeit ist die Domäne des Maschinenbaus, insbesondere die Entwicklung von Prüfmaschinen, von besonderer Bedeutung. Dabei sollen existierende Forschungsergebnisse, Praxisanforderungen und Maschinenbaugrundlagen sowie die Integration der spezifischen Sicherheitsanalysemethoden möglichst durch Erweiterung bestehender Ansätze berücksichtigt werden.

1.3 Forschungsfragen, Methodologie, Zielsetzung und Aufgaben

Zur Konkretisierung der Problemstellungen des vorangegangenen Unterabschnitts sollen die folgenden Forschungsfragen (FF) definiert werden:

- **FF1: Frage zur fehlenden maschinenbauspezifischen Integration von System-, Software- und sicherheitsbezogenen Modellen für Prüfmaschinen**
  Wie lässt sich ein gemeinsames maschinenbauspezifisches Framework für die System- und Softwaremodellierung und für die sicherheitsbezogene Modellierung strukturell entwickeln, um die Abbildung von Stakeholder-spezifischen Begriffen, Gemeinsamkeiten, Verallgemeinerungen, Spezialisierungen und die Kommunikation unter Stakeholdern zu ermöglichen?
- **FF2: Frage zur Integration der Analysemethoden des Maschinenbaus**
  Wie können verschiedene domänenspezifische Sicherheitsanalysemethoden in bestehende MBSE-Konzepte integriert werden, um eine möglichst allgemeingültige, wiederverwendbare und auf Projekte spezialisierbare Erweiterung zu erreichen und dabei ein konsistentes Datenmodell zu fördern?
- **FF3: Frage zu MBSE-Prozesse für KMUs**
  Wie kann die Einsetzbarkeit von MBSE im Kontext von Sicherheitsbewertungen im Maschinenbau für KMUs verbessert werden und wie sind entsprechende Prozesse zur vereinfachten Anwendbarkeit zu strukturieren?

Um Antworten auf diese Forschungsfragen und die Problemstellungen aus Abschnitt 1.2.6 zu erarbeiten, baut diese Arbeit auf der etablierten Forschungsmethodik von Nunamaker [112] auf. Abbildung 1.1 gibt einen Überblick über die dabei eingesetzten vier Phasen zur Problemlösung durch Forschung und Entwicklung. Diese vier Phasen sind miteinander verbunden und können neue Forschungsziele zur Problemlösung beitragen.

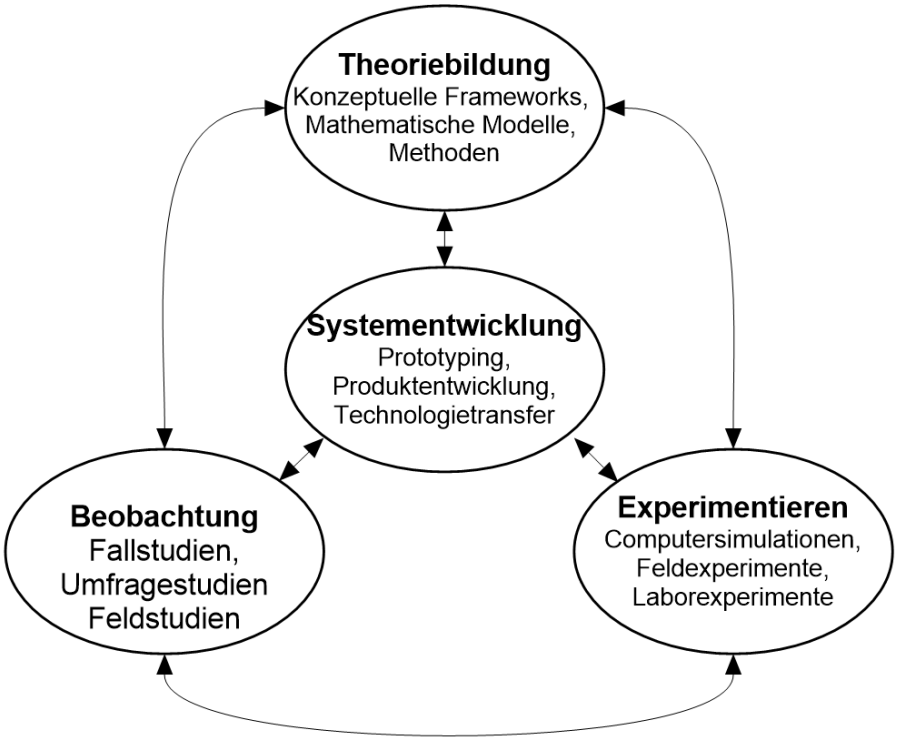


Abbildung 1.1: Problemlösungsphasen nach Nunamaker [112]

In der Beobachtungsphase werden Hintergrundinformationen gesammelt, Studien durchgeführt und das übergeordnete Ziel der Beobachtung von bestehendem Verhalten verfolgt. Diese Beobachtungen führen in der Regel zu Forschungsfragen anderer Phasen. Beispielsweise kann sich eine Theorie auf einigen gemachten Beobachtungen aufbauen oder kann auch die Ergebnisse anderer Phasen unterstützen, indem ein Experiment beobachtet wird. Die Phase Theoriebildung beinhaltet typischerweise die Modellierung von Lösungen in entsprechenden Frameworks, mathematischen Berechnungen oder Beweisen. Die Phase Systementwicklung oder Systemimplementierung kann Lösungen liefern, die auf den Theorien oder Beobachtungen beruhen, die dann in der Phase Experimentieren bewertet werden. Nach Nunamaker können diese Phasen miteinander verbunden sein und Ergebnisse (d.h. Forschungsziele) zueinander beitragen. In dieser Arbeit wird der Problemlösungsansatz von Nunamaker verwendet, um sicherzustellen, dass sich jedes Forschungsziel auf eine der Forschungsfragen auswirkt. Ausgehend von den Forschungsfragen FF1, FF2 und FF3 sollen die folgenden Forschungsziele (FZ) identifiziert werden.

- **Forschungsziel zur fehlenden maschinenbauspezifischen Integration und Modellierung von System-, Software- und sicherheitsbezogenen Modellen für Prüfmaschinen (FF1)**

- FZ1: eine Architektur für eine gemeinsame Modellebene zur Modularisierung eines MBSE-Frameworks für die verschiedenen Stakeholder im Hinblick auf Erweiterbarkeit und maschinenbauspezifische Spezialisierung, speziell für die Klasse der Prüfmaschinen, um die Anforderungen an die funktionale Sicherheit darin formulieren und bewerten zu können.

- **Forschungsziel zur Integration der Analysemethoden des Maschinenbaus (FF2)**
  - FZ2: typische Analysemethoden aus der Maschinensicherheit identifizieren, in einem MBSE-Framework implementieren, anwenden und miteinander vergleichen.
- **Forschungsziel zu MBSE-Prozessen für KMUs (FF3)**
  - FZ3: Zuschnitt eines geeigneten MBSE-Frameworks für den Einsatz in KMUs.

Unter Anwendung der Methodik von Nunamaker werden für jedes dieser Forschungsziele vier Forschungsaufgaben (FA) definiert, die in dieser Arbeit behandelt werden. Die initiale Planung dieser Forschungsziele ist in folgender Tabelle 1.1 dargestellt. Die FA sind entsprechend ihren Forschungszielen gekennzeichnet und mit einem Buchstabenkürzel versehen, die die Art der FA angibt. „B“ steht für Beobachtung, „TB“ für Theoriebildung, „I“ für Implementierung und „E“ für Experiment. Zusätzlich werden entstehende offene Herausforderungen mit der Folge „OH“ gekennzeichnet und am Ende jedes Kapitels zu den verbleibenden FA integriert.

Forschungsaufgaben (FA)	
FA	**Beschreibung**
Maschinenbauspezifische Modellierung	
FA1-B	Recherche einer gemeinsamen Modellebene und Architektur für MBSE im Maschinenbau, im Speziellen für Prüfmaschinen, und Grundlage zur Modularisierung
FA1-TB	Design einer modularisierten gemeinsamen Modellebene und Architektur für MBSE im Maschinenbau, im Speziellen für Prüfmaschinen
FA1-I	Umsetzung einer modularisierten gemeinsamen Modellebene und Architektur für MBSE im Maschinenbau, im Speziellen für Prüfmaschinen
FA1-E	Evaluation einer modularisierten gemeinsamen Modellebene und Architektur für MBSE im Maschinenbau, im Speziellen für Prüfmaschinen
Integration der Analysen des Maschinenbaus	
FA2-B	Identifikation maschinenbauspezifische Analysemethoden und passender Frameworks
FA2-TB	Modellierung zur Integration maschinenbauspezifische Analysemethoden in einem passenden Framework
FA2-I	Entwurf und Implementierung zur Integration maschinenbauspezifische Analysemethoden
FA2-E	Evaluation typischer Analysemethoden
MBSE-Prozesse für KMUs	
FA3-B	Identifikation der Hindernisse zur Anwendbarkeit von MBSE in KMUs
FA3-TB	Vorgehen zur Anwendbarkeit von MBSE in KMUs
FA3-I	Implementierung eines Konzeptes für KMUs in der Maschinenindustrie
FA3-E	Evaluation mit KMU

Tabelle 1.1: Forschungsaufgaben

Zur Abgrenzung der Forschungsziele und -aufgaben sollen nicht verfolgte Forschungsziele skizziert werden, die vielfach mit der Entwicklung sicherer Systeme verbunden werden.

- **Keine Überprüfung des mechanischen oder elektrotechnischen Modells**
  Die Arbeit befasst sich nicht mit der Überprüfung des mechanischen oder elektrotechnischen Modells einer Maschine und inwieweit dieses von den Stakeholdern in CAD oder Schaltplänen umgesetzt wird. Transformatoren für CAD oder Schaltplan werden nicht betrachtet. Der erreichte Detaillierungsgrad der verfolgten Systemmodellierung dient der Kommunikation der Stakeholder untereinander in einer gemeinsamen SysML/UML-basierten Modellwelt.

- **Keine Verifikation von Analysewerkzeug-Transformatoren**
  Die implementierten Transformatoren werden nicht formal verifiziert oder durch eine Testsuite in ihrer korrekten Funktionsweise abgesichert, da es sich um eine prototypische Implementierung handelt, die keine kommerzielle oder validierbare Qualität der implementierten Lösung beansprucht.

- **Keine Erstellung von Softwarecode auf Zielsystem**
  Die Arbeit beschäftigt sich nicht mit Betrachtungen zur Erstellung korrekter Software für sicherheitskritische Zielsysteme. Dabei werden beispielsweise spezifische Aspekte wie die Auswahl geeigneter Programmiersprachen, einschränkende Styleguides und zugehörige Prüfwerkzeuge, formale Korrektheitsbeweise für entwickelte Programme, Multiversion-Software oder der Nachweis der Fehlerfreiheit der genutzten Toolchain nicht untersucht.

- **Keine Betrachtungen zur Hardware-Systemarchitektur**
  Die Arbeit geht nicht auf Betrachtungen zur Systemarchitektur sicherheitskritischer Hardware-Systeme ein, etwa im Hinblick auf weitgehende Diagnosemöglichkeiten, fehlerkorrigierende Speicher, Master-Checker-Konfigurationen von CPUs oder Fehlermaskierung wie zum Beispiel Triple Modular Redunduncy.

- **Keine Security-Aspekte**
  Die gleichermaßen wichtigen Betrachtungen von Sicherheit im Sinne von Security (IT-Sicherheit) sind sicherheitskritischen Anwendungen untergeordnet. Um die Interdependenz zwischen Safety und Security in sicherheitskritischen Anwendungen zu verdeutlichen, ist es somit wichtig zu erkennen, dass beide Aspekte für einen robusten Systembetrieb grundlegend notwendig sind [113]. Die funktionale Sicherheit legt den Schwer-

punkt auf die formale Analyse in frühen Entwurfsphasen, um systematisch sowohl Sicherheitsrisiken als auch Sicherheitsschwachstellen zu bewältigen und sicherzustellen, dass die Systeme alle kritischen Anforderungen erfüllen [114]. Darüber hinaus erweisen sich diese formalen Analysen bei industriellen sicherheitskritischen Anwendungen, bei denen Safety- und Securityanforderungen häufig im Widerspruch zueinanderstehen, als wirksam bei der Identifizierung und Lösung dieser Konflikte, wodurch Safety gewährleistet wird, ohne die Security zu untergraben [115]. Um sich in dieser Arbeit auf ein Themenfeld zu konzentrieren sollen daher Aspekte zur Security nicht behandelt werden, da dies eigenständige Betrachtungen mit sich führen würde. Es ist jedoch wichtig zu betonen, dass sich in der Gesamtbetrachtung, vor allem im Bezug zur steigenden Automatisierung, Safety und Security gegenseitig bedingen und unterstützen (vgl. Jung [116] Ohne Security kein Safety).

Die weitere Gliederung der Arbeit folgt den verschiedenen Typen der FA und wird im Abschnitt 1.4 skizziert.

1.4 Ansatz und Aufbau der Arbeit

Der Aufbau dieser Arbeit orientiert sich an dem Ansatz von Nunamaker [112] und folgt dessen Modellierung. Eine Übersicht über den Aufbau der Arbeit und den Ansatz von Nunamaker ist in Abbildung 1.2 auf Basis der Problemlösungsphasen in Abbildung 1.1 dargestellt. Je nach Phase des Nunamaker-Modells werden die entsprechenden Forschungsaufgaben neu geordnet und in den jeweiligen Kapiteln dieser Arbeit platziert:

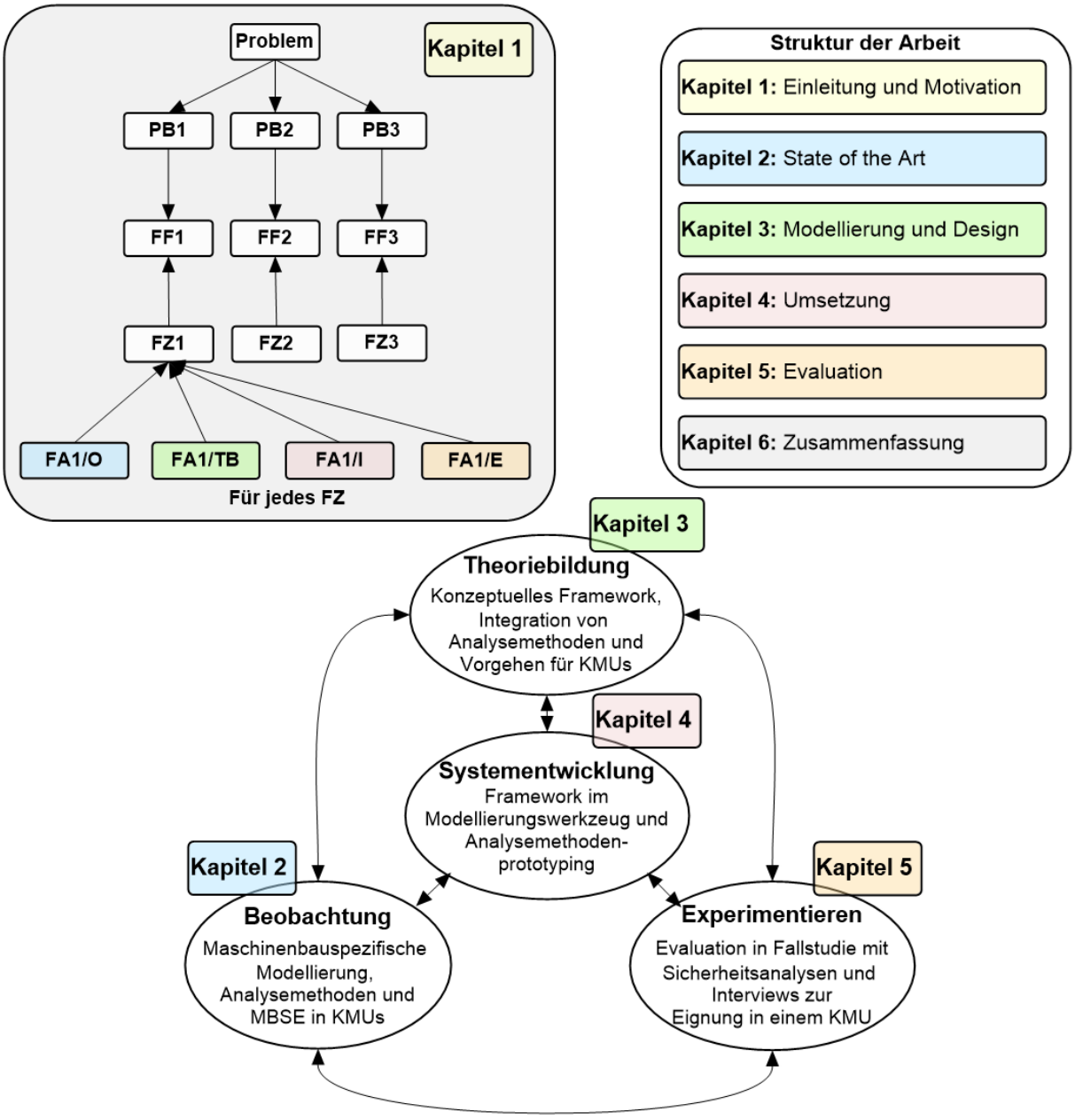


Abbildung 1.2: Struktur der Arbeit und Problemlösungsansatz

- Kapitel 2 enthält anfängliche Forschungsaufgaben des Typs Beobachtung, um den aktuellen Stand in Wissenschaft und Technologie einzuführen und zusammenzufassen. Hierbei liegt der Fokus in Abschnitt 2.1 in der maschinenbauspezifischen Modellierung,

in Abschnitt 2.2 in Analysemethoden und in Abschnitt 2.3 in der Möglichkeiten für MBSE in KMUs sowie geeigneten Evaluationsmethoden.

- Kapitel 3 enthält Forschungsaufgaben des Typs Theoriebildung und beschreibt daher in Abschnitt 3.1 eine kurze Einführung, in Abschnitt 3.2 und 3.3 die Modellierung und den Entwurf eines für den Maschinenbau spezialisierten Frameworks zur Integration von Analysemethoden und in Abschnitt 3.4 ein Vorgehen zur vereinfachten Anwendbarkeit in KMUs.
- Kapitel 4 fasst die Forschungsaufgaben des Typs Implementierung zusammen. Es beschreibt hierbei in Abschnitt 4.1 das konkrete Design und Begründungen zum einzusetzenden Modellierungswerkzeug sowie in Abschnitt 4.2 die Implementierung der Konzepte und daraus folgender Transformatoren.
- Kapitel 5 enthält in Abschnitt 5.1 und 5.2 eine ausführliche Anwendung der Konzepte der Arbeit im Rahmen einer Fallstudie. Abschnitt 5.3 nutzt darauf folgend die Fallstudie und evaluiert die Ansätze über den in Abschnitt 2.3 herausgearbeiteten Prozess einer geeigneten qualitativen Evaluationsmethode im Rahmen von Interviews. Das Kapitel umfasst somit alle Aufgaben des Typs Experiment.

Das Nunamaker-Modell wurde entwickelt, um einen formalen wissenschaftlichen Prozess zur Problemlösung anzuwenden. Durch das Einhalten dieses Modells wird eine klare Beziehung zwischen den Forschungsaufgaben und dem Forschungsziel hergestellt, was sicherstellt, dass jedes Ergebnis, das in dieser Arbeit präsentiert wird, direkt mit der entsprechenden Forschungsfrage und Problemstellung verknüpft werden kann. Umgekehrt kann die Lösung für jede Forschungsfrage direkt durch die entsprechenden Forschungsziele dargestellt werden. Darüber hinaus folgt das Modellierungskapitel dieser Arbeit dem User Centered System Design-Ansatz von Draper [117], der den Benutzer in den Mittelpunkt jeder Art von Modellierung oder Design stellt. Die Modellierung und das Design sowie der Entwurf und die Umsetzung folgen den in OMG UML [118] und OMG SysML [26] beschriebenen Grundlagen.

Während des Promotionsstudiums und Erstellung dieser Arbeit wurden verschiedene Publikationen erstellt und zwei Masterarbeiten begleitet, die ebenfalls zu diesem Dokument beigetragen haben. Eine vollständige Liste der Veröffentlichungen findet sich unter Anhang A.1. Ein Statement zum Themen und Betreuerwechsel 2020 findet sich in Anhang A.2.

In diesem ersten Kapitel wurde die allgemeine Motivation und Einordnung des Themas vorgestellt. Dabei hat sich gezeigt, dass Model-Based Systems Engineering sicherheitskritischer Anwendungen im Maschinenbau mittelständischer Unternehmen hochrelevant ist. Es wurden drei

große Problembereiche identifiziert, die im Rahmen dieser Arbeit behandelt werden. Die Definition der Problemstellung, der Forschungsfragen und der Forschungsziele dient als formale Grundlage für die Struktur der Arbeit. Basierend auf dem Nunamaker-Modell folgt diese Arbeit einem wissenschaftlich akzeptierten Ansatz bei der Darstellung der Forschungsergebnisse. Daher können die vorgestellten Ergebnisse auch als formale Grundlage für andere Arbeiten dienen.

Im nächsten Kapitel findet sich ein detaillierter Überblick über den Stand der Wissenschaft und Technologien, der die Ergebnisse der Forschungsaufgaben vom Typ Beobachtung darstellt und dazu beiträgt, die Ergebnisse dieser Arbeit vom aktuellen Stand der Wissenschaft und Technologie abzugrenzen.

2 Stand der Wissenschaft und Technologie

Im Maschinenbau kleiner und mittelständischer Unternehmen (KMUs) spielen sicherheitskritische Anwendungen eine entscheidende Rolle. Die Zuverlässigkeit in Bezug auf die Sicherheit von Maschinen ist nicht nur für die Produktion selbst, sondern auch für die Sicherheit von Mitarbeitern und der Umwelt, von größter Bedeutung. Model-Based Systems Engineering (MBSE) bietet hierfür eine mögliche Methode, um diese Herausforderungen zu bewältigen. Dabei werden verschiedene Aspekte wie die Anforderungsdefinition, Interaktion von Stakeholdern, Modellierung der System- und Analysemodelle und die Analyse der Artefakte betrachtet. Ziel ist es, die wichtigsten Ansätze zu identifizieren, um KMUs im Maschinenbau in die Lage zu versetzen, sichere und zuverlässige Produkte zu entwickeln und gleichzeitig wettbewerbsfähig zu bleiben.

Gemäß dem Ansatz von Nunamaker [112] repräsentiert dieses Kapitel den aktuellen Stand der Wissenschaft und Technologie im Bereich MBSE sicherheitskritischer Anwendungen im Maschinenbau mittelständischer Unternehmen und damit Forschungsaufgaben vom Typ Beobachtung. Es folgt der in Abschnitt 1.2.6 definierten Struktur. Konzepte, die für diese Arbeit relevant sind, werden detailliert eingeführt und bilden eine solide Grundlage für die Modellierungs-, Implementierungs- und Evaluierungsziele der nachfolgenden Kapitel. Daher beschreibt Abschnitt 2.1 den Stand der Forschung im Bereich der maschinenbauspezifischen System-, Software- und sicherheitsbezogenen Modellierung, während Abschnitt 2.2 Ergebnisse aus Recherchen zu Sicherheitsanalysemethoden im Maschinenbau enthält, um die sicherheitsbezogene Integration fortzusetzen. Schließlich werden in Abschnitt 2.3 die relevanten Herausforderungen beschrieben, die bei der Anwendung von MBSE in KMUs entstehen, um die Frage der vereinfachten Anwendbarkeit anzusprechen. Jede Forschungsaufgabe wird in einem eigenen Abschnitt behandelt und enthält die relevanten Informationen zur Bearbeitung der Aufgabe. Am Ende jedes Abschnitts wird eine Zusammenfassung der offenen Herausforderungen und verbleibenden Fragen gegeben.

Model-Based Systems Engineering sicherheitskritischer Anwendungen im Maschinenbau mittelständischer Unternehmen © 2024 by Nick Berezowski is licensed under CC BY 4.0

2.1 Maschinenbauspezifische Modellierung

In diesem Abschnitt werden die grundlegenden Konzepte und ausgewählte Literaturergebnisse im Bereich der modellbasierten Entwicklung und der sicherheitsbezogenen Integration präsentiert, um verschiedene Aspekte der Problembeschreibung PB1 zu behandeln. Die Struktur dieses Abschnitts entspricht Rechercheaufgaben. Unterabschnitt 2.1.1 zeigt die strukturellen Voraussetzungen für MBSE. Unterabschnitt 2.1.2 verdeutlicht die Stakeholder-spezifischen Sichten zur Veranschaulichung von Integrationsmöglichkeiten. Aufbauend auf der Sicht des Sicherheitsingenieurs, geht Unterabschnitt 2.1.3 auf die notwendigen Grundlagen der Maschinensicherheit ein. Und zuletzt legt Unterabschnitt 2.1.4 die Grundlagen zu einem gemeinsamen Informationsmodell fest. Schließlich werden die beobachteten Literaturinformationen, einschließlich verbleibender Herausforderungen, zusammengefasst, diskutiert und mit den Forschungszielen in Abschnitt 2.4 abgeglichen.

2.1.1 Grundlagen des MBSE

Im Allgemeinen haben modellbasierte Ansätze ihre Wurzeln in der Mathematik und basieren auf der mathematischen Logik, um die Handhabbarkeit von Modellen zu verbessern. Dies führte zur Entwicklung maschinenlesbarer Sprachen durch die Object Management Group (OMG), die heute ein kommerziell unterstütztes Standardisierungskonsortium für modellbasierte Ansätze in der Software- und Systementwicklung darstellt [119]. Die Geschichte von Model-Based Systems Engineering (MBSE) geht dabei zurück in die 1990er Jahre, als die Bedürfnisse nach besseren Methoden zur Modellierung von Systemen wuchsen [120]. Erste Versuche, Systeme modellbasiert zu entwickeln, waren auf die Verwendung von Unified Modeling Language (UML) [25] beschränkt. Doch erwies sich UML als unzureichend für die spezifischen Anforderungen [121], weshalb 2005 die Systems Modeling Language (SysML) [26] entwickelt wurde, die speziell MBSE technischer Systeme dient und auf Profilen innerhalb der Meta Object Facility (MOF) [122] basiert [121]. Als Ergänzung findet oft die Object Constraint Language (OCL) [123] Einsatz, um Einschränkungen von Modellen zu definieren, und Cascading Style Sheets (CSS) [124], um die Darstellbarkeit und Assoziation zu verbessern.

Graphische Modelle stellen in der Regel die Entitäten eines Systems als Knoten in einem Graphen dar und die Beziehungen als Verbindungen zwischen den Knoten. Die Syntax und Semantik, die von graphischen Modellen bereitgestellt werden, hilft, die Bedeutung von natürlichen Sprachsätzen zu erfassen. Um jedoch die vollständige Bedeutung der Sätze zu erfassen, ist eine Interpretation in eine maschinenlesbare Sprache wie XML Metadata Interchange (XMI [125]) erforderlich. Zusätzlich zeigt sich die Methode zur modellgetriebenen Softwareentwicklung,

englisch Model-Driven Software Development (MDSD), inklusive Codegenerierung als vielversprechender systematischer Ansatz [36, 87, 126, 127]. MDSD hat sich in Bereichen wie der Automobilbranche [20]und in der Kernsicherheitsforschung [21] über einen längeren Zeitraum als erfolgreich etabliert und zeigt auch in der Maschinenindustrie eine zunehmende Verbreitung und Evaluation [36].

MBSE ist als eine formale Anwendung von Modellierung definiert, die darauf abzielt, die Systemanforderungen, Design, Analyse, Überprüfung und Validierung von Prozessen zu unterstützen [128]. Im Gegensatz zum traditionellen Systems Engineering-Ansatz fokussiert sich MBSE auf die Entwicklung eines kohärenten Systemmodells [128]. Hierbei dient das Modell als zentraler Beweisgegenstand und stellt das primäre Artefakt dar. Studien zeigen, dass MBSE im Vergleich zu dem traditionellen Ansatz eine rigorosere Methode zur Aufrechterhaltung und Integration der Ergebnisse von Systems Engineering-Prozessen bereitstellt [129]. Hierdurch entsteht eine verbesserte Kommunikation und dadurch verringerte Entwicklungsrisiken [128], die sich in einer erhöhten Agilität widerspiegeln [130]. Ein detaillierter Einblick in die Grundbegriffe UML, SysML, Profile, OCL, CSS und XMI findet sich in Anhang A.3.

2.1.2 Stakeholder in der Maschinenentwicklung

Stakeholder sind Personen oder Gruppen, die ein Interesse an einer Organisation haben und von den Entscheidungen und Handlungen der Organisation betroffen sein können [131]. Beispiele für Stakeholder sind Kunden, Mitarbeiter, Aktionäre, Lieferanten, Behörden und lokale Gemeinschaften, doch ebenso verschiedene Entwicklergruppen und Akteure für die Entwicklung von Systemen. In interdisziplinären Entwicklungsprozessen, unabhängig der Anwendungsdomäne, arbeiten technische Stakeholder unterschiedlicher Fachrichtungen zusammen, um ein gemeinsames Ziel zu erreichen. Dabei ist der Informationsaustausch zwischen den Stakeholdern in Anbetracht steigender technologischer Entwicklung und Komplexität von großer Bedeutung, um ein effektives und erfolgreiches Zusammenarbeiten zu ermöglichen [132]. Der Austausch von Informationen und Wissen zwischen den Stakeholdern aus verschiedenen Bereichen ermöglicht es, ein umfassendes Verständnis des Problems zu erlangen und ganzheitliche Lösungsansätze zu entwickeln [133]. Eine effektive Kommunikation und Zusammenarbeit zwischen den Experten aus verschiedenen Disziplinen ist daher von entscheidender Bedeutung für den Erfolg [134]. Speziell im Maschinenbau können vier technische Stakeholder mit unterschiedlichen Interessen bei der Entwicklung einer Prüfmaschine identifiziert werden.

Maschinenbauingenieur

Ein Maschinenbauingenieur in der Maschinenindustrie kann an der Konstruktion, Entwicklung und Prüfung von mechanischen Systemen, Komponenten und Produkten beteiligt sein, die bei der Herstellung von Maschinen und Anlagen verwendet werden [135]. Sie wenden Prinzipien der Mechanik, Thermodynamik und Werkstoffkunde an, um Produkte zu entwickeln, die sicher, effizient und zuverlässig arbeiten [136]. Zu den Aufgaben eines Maschinenbauingenieurs in der Maschinenindustrie gehören der Entwurf und die Entwicklung mechanischer Systeme und Komponenten, die Durchführung von Belastungs- und Leistungsanalysen, die Beaufsichtigung des Fertigungs- und Montageprozesses, die Prüfung und Bewertung von Prototypen, die Instandhaltung bestehender mechanischer Systeme, die Durchführung von Forschungs- und Entwicklungsarbeiten, die Erstellung von technischen Zeichnungen und Spezifikationen und die Fehlersuche und Behebung mechanischer Probleme [135].

Das Design von Maschinen im Maschinenbau basiert hauptsächlich auf technischen Zeichnungen und 3D-Modellen für Konstruktion und Fertigung [135, 137], aber mathematische Simulationsmodelle werden auch genutzt, um das Maschinenverhalten zu beschreiben [135]. Computer Aided Design (CAD) ist ein wichtiges Werkzeug für Maschinenbauingenieure, um komplexe Teile und Baugruppen in einer digitalen Umgebung wie SolidWorks [138], AutoCAD [139] oder FreeCAD [140] zu entwerfen und zu simulieren [141]. CAD spart Zeit und Kosten und erhöht die Genauigkeit und Effizienz in der Fertigung [142, 143, 144].

Das mechanische Design umfasst den Zusammenbau von Baugruppen und Bauteilen [145], was die Grundlage für die elektrotechnische Entwicklung und Bewertung zur funktionalen Sicherheit legt. Es gibt bereits kommerzielle Plattformen, wie Syndeia, für ein modellbasiertes Design [146], die den Informationsaustausch durch Middleware-Lösungen zwischen Modellen und Werkzeugen ermöglichen und die Effizienz bei der Verwendung erhöhen können [147]. Syndeia bietet eine einheitliche Benutzeroberfläche und Methodik für die Arbeit mit verschiedenen Modellen und Werkzeugen [146]. Es gibt jedoch nur Anbindungen zu kommerziellen MBSE-Plattformen, wie MagicDraw [91] oder Rational Rhapsody [148].

Elektroingenieur

Ein Elektroingenieur entwirft und entwickelt E/E/PE-Systeme und Geräte und arbeitet in verschiedenen Domänen in Forschung, Entwicklung und Projektmanagement [149]. Zu den Aufgaben eines Elektroingenieurs in der Maschinenindustrie gehören der Entwurf und die Entwicklung von E/E/PE-Systemen und Komponenten, die Beaufsichtigung des Fertigungs- und Montageprozesses, die Prüfung und Bewertung von Prototypen, die Wartung und Aktualisierung

bestehender Systeme, die Durchführung von Forschungen, um elektrische Technologien voranzutreiben, die Erstellung von Schalt- und Stromlaufplänen und die Fehlersuche und Behebung elektrischer und elektronischer Probleme [149].

Im Maschinenbau werden Modelle in Form von Schaltplänen, Verdrahtungsplänen und Konstruktionszeichnungen entworfen. Diese enthalten Details wie Maße, Materialangaben, Anschlüsse und andere technische Spezifikationen [150]. Schaltpläne werden als Leitfaden für die Herstellung und Installation von E/E/PE-Systemen und -Komponenten genutzt [151]. Sie dienen im Weiteren der Überprüfung und Zertifizierung der funktionalen Sicherheit [5] und bieten Grundlage für Simulationsmodelle [152]. Elektroingenieure nutzen oft computergestützte Schaltplanerstellungs-Software wie EPLAN [153] oder QElectroTech [154] zur Erstellung und Verwaltung von elektrischen Schaltplänen, Verdrahtungsplänen und Dokumentationen [150].

Bauteile und Komponenten lassen sich in die Gruppen elektromechanische, passive und aktive Bauelemente, Sensoren, Aktoren, Controller, Spannungsversorgungen und elektrische Verbindungen einteilen [155]. Diese können je nach Bedarf kombiniert werden [156]. Die elektrotechnische Konstruktion nutzt hierfür die Prinzipien der mechanischen Konstruktion einer Maschine, um Schaltpläne zur Steuerung und Regelung zu entwerfen und zu entwickeln. Dies bildet die Grundlage für die elektrotechnische Ausstattung mit Sensoren, Aktoren und Steuereinheiten wie SPSen, die für die Entwicklung von Softwarefunktionalitäten und die Bewertung der funktionalen Sicherheit von Bauteilen und Funktionen einer Maschine erforderlich sind.

Informatiker

Informatiker in der Maschinenindustrie können an der Entwicklung und Gestaltung von Software und Computersystemen für Maschinen, Werkzeuge und Fertigungsprozesse beteiligt sein [157]. Sie nutzen Prinzipien der Informatik und Ingenieurwissenschaften, um Automatisierungs- und Steuerungssysteme zu verbessern, Datenanalysen- und Überwachungssysteme zu erstellen und vorbeugende Wartungswerkzeuge zu entwickeln. Informatiker arbeiten zusammen mit Maschinenbau- und Elektroingenieuren, um Software- und Hardware-Systeme nahtlos integrieren zu können und eine optimale Funktion zu gewährleisten [158].

Die Aufgaben eines Informatikers in der Maschinenindustrie umfassen unter anderem die Entwicklung von Software für die Überwachung, Steuerung und Regelung, die Integration von Sensoren, Aktoren und Steuereinheiten in die Maschinenkontrolle, die Entwicklung von Benutzeroberflächen für Maschinenbediener, die Überwachung der Datenübertragung und -analyse in der Fertigung, die Entwicklung von Prozessen und Methoden zur Optimierung der Leistung und Effizienz und die Durchführung von Tests und Debugging von Software-Systemen [158].

Der Entwurf und die Entwicklung von Modellen beziehen sich meist auf Modellierungssprachen, die die Struktur und das Verhalten eines Software-Systems beschreiben. UML-Diagramme dienen als Grundlage für den zu produzierenden Softwarecode einer Maschine, zumeist beschrieben in Funktionsblockdiagrammen [87]. Beispielsweise kann über XMI-Transformatoren automatisch Softwarecode aus UML erzeugt werden [36, 87, 159, 160], der in SPS-Automatisierungswerkzeugen wie TwinCAT [161] Einsatz findet. Dies fördert die Konsistenz und das Vertrauen sowie bietet Grundlage für die Überprüfung und Zertifizierung, beispielsweise über modellbasierte Testfallerstellung und Simulationen gemäß geltenden Normen der funktionalen Sicherheit [7]. Ein typisches Werkzeug stellt Eclipse Papyrus [93] dar. Es unterstützt das Eclipse Modeling Framework ((EMF) vgl. Anhang A.3.8) sowie Profile und basiert im Inneren auf der Eclipse Plugin-Architektur. Ein Vergleich zu anderen Werkzeugen findet sich in Anhang A.3.9.

Das informationstechnische Design nutzt somit die Rahmenbedingungen der mechanischen und elektrotechnischen Konstruktion einer Maschine, um Softwarealgorithmen zu entwerfen und umzusetzen. Sie nutzt im Speziellen die elektrotechnische Ausstattung mit Sensoren, Aktoren und Steuereinheiten wie zumeist SPSen [36], um Softwarefunktionalitäten hierauf abzubilden.

Sicherheitsingenieur

Ein Sicherheitsingenieur ist verantwortlich für die Einhaltung der funktionalen Sicherheit [162]. Er verwendet dabei verschiedene Analysemethoden, um die Sicherheit zu überprüfen und nachzuweisen [6]. Sein Expertenwissen helfen ihm dabei, potenzielle Gefahren zu bewerten und zu mindern [163]. Sicherheitsingenieure sind dabei mitverantwortlich für die Einhaltung von Sicherheitsvorschriften und -standards, die Durchführung von Sicherheitstrainings, die Untersuchung von Vorfällen und die Bewertung und Empfehlung von Sicherheitsmaßnahmen [164]. Die Aufgaben bei der Entwicklung einer Maschine beinhalten die Bestimmung der Grenzen der Maschine gemeinsam mit anderen Stakeholdern, die Identifikation von Gefährdungen und Gefährdungssituationen, die Risikoabschätzung mit Bestimmung der Effekte für Mensch, Maschine und Umwelt, die Risikobewertung durch Sicherheitsanalysen, die Entwicklung von konstruktiven und technischen risikomindernden Schutzmaßnahmen und Benutzerinformationen, die Zusammenarbeit mit anderen Stakeholdern, um sicherzustellen, dass geforderte Maßnahmen ausreichend in die Gestaltung und Entwicklung der Maschine einfließen, die Durchführung von Sicherheitsanalysen zum Nachweis der Risikominderung und die Erstellung und Aufrechterhaltung von sicherheitsbezogenen Dokumentationen und Aufzeichnungen [17, 34, 162, 163, 164].

Der Entwurf und die Entwicklung von Modellen beziehen sich auf die Sicherheitsanalysen zur Risikobeurteilung und der probabilistischen Analyse der Sicherheit der Maschine. Die sicherheitstechnische Umsetzung nutzt somit die Rahmenbedingungen der mechanischen und elektrotechnischen Konstruktion, sowie der informationstechnischen Umsetzung einer Maschine. Da die Sicherheit von Maschinen das Hauptaugenmerk dieser Arbeit darstellt, soll im folgenden Abschnitt detailliert auf die Rahmenbedingungen des Sicherheitsingenieurs eingegangen werden.

Natürlich gibt es noch eine Vielzahl an anderen Blickwinkeln und Arbeiten zu den Grundlagen von MBSE, die im Rahmen dieser Arbeit betrachtet werden können, doch soll sich der weitere Rahmen auf den genannten Sichtweisen beschränken. Auf Basis dieser Grundlagen wird bereits eine Abhängigkeit der verschiedenen Stakeholder zueinander ersichtlich und es bildet sich ein Rahmenwerk zur Integration dieser Sichten in ein gemeinsames Modell. Allerdings fehlt ein Rahmenwerk für UML und SysML, das die verschiedenen Sichten in Korrelation zur funktionalen Sicherheit integrieren kann. Es finden sich bisher keine generalisierten Begrifflichkeiten, um den Maschinenbau und dessen Sicherheitsbetrachtungen zu integrieren.

2.1.3 Sicherheitsbezogene Entwicklung im Maschinenbau

Zunächst soll ein Überblick über Normen und Standards gegeben werden, die die Rahmenbedingungen für sicherheitsbezogene Entwicklung im Maschinenbau und die Grundlagen für darauffolgenden Begriffsdefinitionen und die systematische Vorgehensweise liefern.

Standards

Um die Risiken bei der Entwicklung von Maschinen zu minimieren, gibt es Normen, die die Sicherheitsanforderungen definieren und die sicherheitskritische Entwicklung von Maschinen regeln. In diesem Abschnitt wird die Entwicklung in der Maschinenindustrie ausführlich behandelt und Grundlagen für eine Integration in MBSE gelegt.

Die Entwicklung und Forschung basiert in der Regel auf der, oft als Grundnorm der funktionalen Sicherheit bezeichneten, IEC 61508 [165]. Diese in Abbildung 2.1 mittig dargestellte Norm definiert Grundanforderungen an den gesamten Sicherheitslebenszyklus, von der Konzeption bis zur Außerbetriebnahme von Produkten [5].

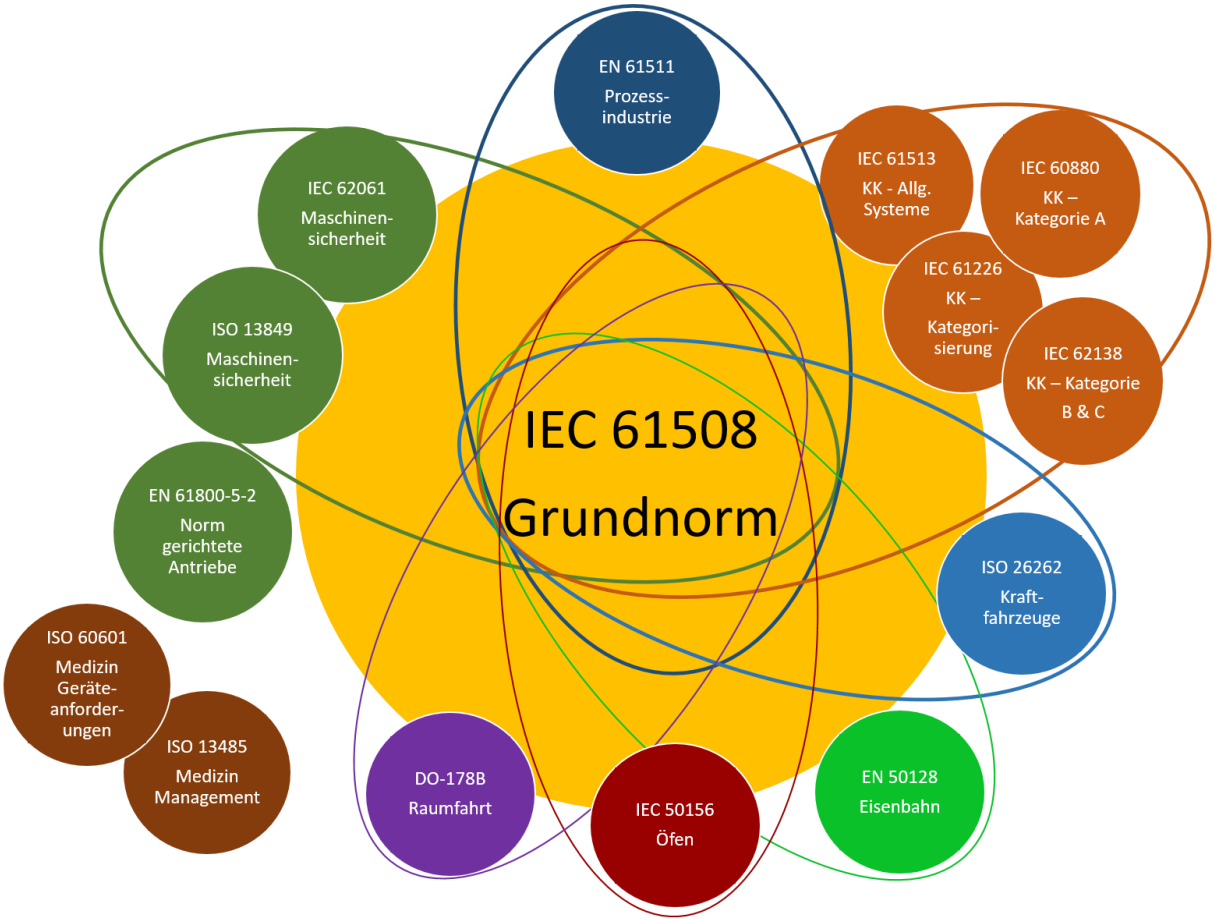


Abbildung 2.1: Überblick über Standards der funktionalen Sicherheit

Domänenspezifische Normen sind dieser untergeordnet und verfeinern deren Ansätze für eine erleichterte Anwendbarkeit [166]. Dabei variieren oft Gefährdungseinstufungen, Kategorisierungen oder der Einsatz spezieller Verfahren. Für medizinische Einrichtungen darf die Grundnorm IEC 61508 keine Anwendung finden [6]. Einen sehr interessanten und lesenswerten Überblick über die Grundzüge der funktionalen Sicherheit, eingesetzter Normen und Strukturen sowie Berechnungsmethoden gibt Börcsök [167].

In der Normierung können Normen neben der Aufteilung nach Domänen auch in Typ A, Typ B und Typ C Normen unterteilt werden, wie in Abbildung 2.2 beschrieben.

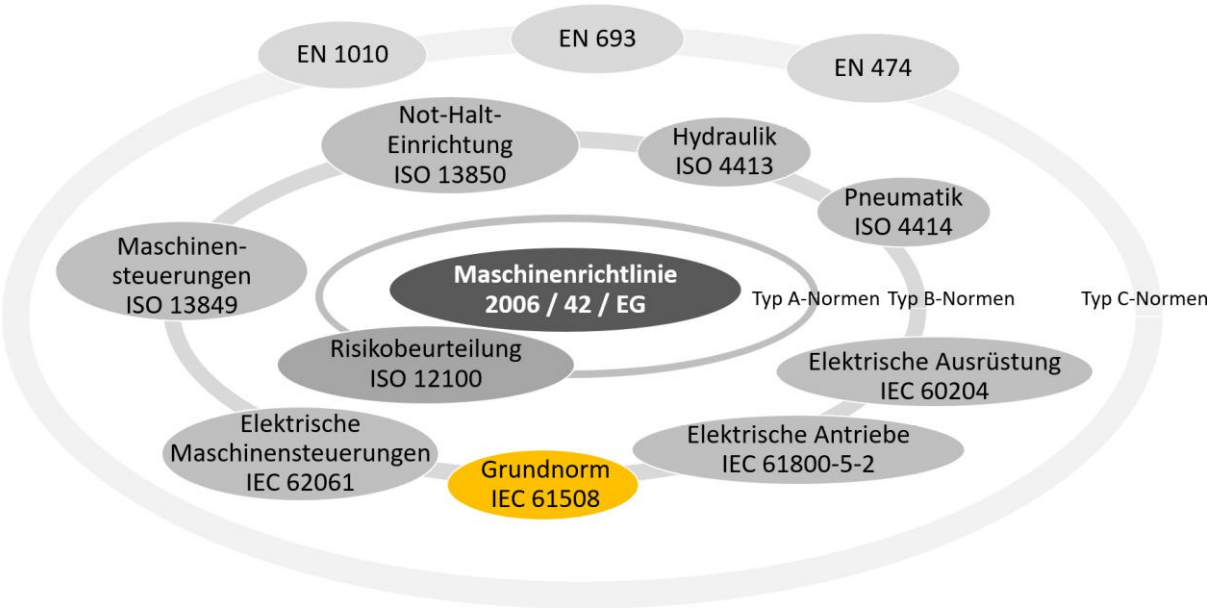


Abbildung 2.2: Normen der Maschinenindustrie

Typ A Normen geben allgemeine Grundsätze und Leitlinien vor, wie z.B. ISO 9001 für Qualitätsmanagement, ISO 14001 für Umweltmanagement oder speziell in der Maschinenindustrie ISO 12100 für Gestaltungsleitsätze für die Sicherheit von Maschinen. Typ B Normen legen spezifische Anforderungen fest und werden in B1 Normen für bestimmte Sicherheitsaspekte, wie z.B. die ISO 13849-1 für sicherheitsbezogene Teile von Steuerungen oder die EN 62061 für sicherheitsbezogene elektrische, elektronische Steuerungssysteme, und B2 Normen für bestimmte Schutzeinrichtungen, wie z.B. die ISO 13850 für Nothalt-Funktionen, unterteilt. Typ C Normen hingegen sind spezifische Normen für die Maschinensicherheit, die detaillierte Sicherheitsanforderungen an eine bestimmte Maschine oder eine Gruppe von Maschinen wiedergeben, wie z.B. die ISO 16092-3 für hydraulische Pressen oder die EN1010-1 für Druck- und Papierverarbeitungsmaschinen. Dabei haben die Festlegungen der Typ-C-Norm Vorrang, falls sie von denen abweichen, die in Typ-A- oder Typ-B-Normen definiert sind.

Bei den in Abbildung 2.2 grau hinterlegten Normen handelt es sich um auf Grundlage der Maschinenrichtlinie 2006/42/EG harmonisierte Normen. Die 2006/42/EG, verabschiedet vom Europäischen Parlament und Rat am 17. Mai 2006 und zuletzt 2019 konsolidiert, zielt auf die Standardisierung des Schutzniveaus zur Vermeidung von Unfällen im Zusammenhang mit Maschinen und Teilmaschinen ab, die innerhalb des Europäischen Wirtschaftsraums vertrieben werden dürfen [168]. Es handelt sich bei ihr um eine Gesetzesvorlage, die nicht oder nur schwer zur konkreten Entwicklung von Maschinen Einsatz finden kann. Das Hauptaugenmerk soll daher Typ-A und Typ-B1 Normen gelten, die die technischen Grundlagen für die Entwicklung von sicherheitskritischen Maschinen legen. Wie die Namen von ISO 13849 und EN 62061 bereits vermuten lassen, handelt es sich um redundante Normen, von denen lediglich eine für die

Entwicklung von sicherheitsbezogenen Steuerungen Verwendung finden muss. Aufgrund der zusätzlichen Harmonisierung in Bezug zur elektromagnetischen Verträglichkeitsrichtlinie 2014/30/EU und vereinfachter Verfahren für die Entwicklung bietet sich die ISO 13849-1 für das geplante Vorhaben an.

Um Grundlagen für eine Modellierung der funktionalen Sicherheit zu schaffen, sollen im Folgenden grundlegende Begriffe der Maschinensicherheit eingeführt werden.

Begriffe aus der Maschinensicherheit

Unter einem Ausfall, englisch Failure, wird die Beendigung der Fähigkeit einer Einheit, also eines Systems oder einer Komponente, verstanden, eine geforderte Funktion auszuführen [163]. Wird die Funktion auf einem Steuerungssystem ausgeführt, spricht man vom sicherheitsbezogenen Teil eines Steuerungssystems, englisch Safety Related Part of Control System (SRP/CS). Die Funktion selbst wird als Sicherheitsfunktion bezeichnet. Sicherheitsfunktionen repräsentieren Maschinenfunktionen, deren Ausfall direkt zur unmittelbaren Erhöhung von Risiken führen. Das Steuerungssystem besteht meist aus einer sicherheitsbezogenen Embedded-Software, englisch Safety Related Embedded Software (SRESW), also der Firmware, die als Bestandteil des Systems vom Hersteller mitgeliefert wird, und der sicherheitsbezogenen Anwendersoftware, englisch Safety Related Application Software (SRASW) [34].

Ein Fehlerzustand oder Fehlverhalten, englisch Error, kann als Abweichung von einem gewünschten Zustand verstanden werden [8, 163], beispielsweise aufgrund von Konstruktionsfehlern [163] oder Softwarefehlern (Soft-Errors) [6].

Die Störung, englisch Fault, ist die Ursache, die ein System in einen unerwünschten Zustand versetzt und damit den Verlust der Fähigkeit einer Funktionseinheit verursacht, eine geforderte Funktion auszuführen [8]. Sie entsteht oft als Folge eines Versagens einer betroffenen Einheit selbst, kann jedoch auch ohne vorheriges Versagen existieren [34].

Bei der Ausfallart, englisch Failure Mode, spricht die Normierung von der Art, in der ein Element ausfallen kann [5], wie beispielsweise das Verkleben eines Relais in einer Endposition oder Änderung von Schaltzeiten [34]. Dies legt zum einen Grundlagen für Failure Mode and Effects Analysis (FMEA) [6], wie der Name der Analyse bereits andeutet, und bietet zum anderen die Grundlage für probabilistische Berechnungen und Abschätzungen zu Ausfallraten und -zeiten [34]. Damit entsteht eine Kette von Ereignissen, in denen eine Störung zu einem Fehlverhalten, folglich zu einem Fehlerzustand, und schließlich zum Ausfall bzw. zu Abweichungen von beabsichtigen Verhalten führen kann.

Bei der Ausfallrate λ, englisch Failure Rate, spricht man von der Häufigkeit, mit der ein System ausfällt [34]. Hierbei ist die für die funktionale Sicherheit relevante Ausfallrate, die gefahrbringende Ausfallrate eines Bauteils λ_D. Sie berechnet sich aus der Ausfallrate multipliziert mit der gefährlichen Ausfallrate, englisch Ratio of Dangerous Failures (RDF). Einige Hersteller geben hier alternativ den Anteil sicherer Ausfälle, englisch Safe Failure Fraction (SFF), an [6]. Im Fall von Einheiten mit exponentieller Verteilung der Betriebsdauern bis zum gefahrbringenden Ausfall, d.h. einer konstanten Ausfallrate, ist der numerische Wert von λ_D gleich dem Kehrwert der mittleren Zeit bis zu einem gefährlichen Ausfall, englisch Mean Time to Dangerous Failure ($MTTF_D$) [34]. Somit ergibt sich für die Wahrscheinlichkeit eines gefährlichen Ausfalls über die Zeit, englisch Probability of Failure (PF_D)

$$PF_D(t) = 1 - e^{-\lambda_D t}.$$

Formel 2.1: Exponentialverteilung von Bauteilen [34]

Spricht man nicht von einem Bauteil, sondern von einem aus Komponenten bestehenden System, handelt es sich um PFH_D, also der Wahrscheinlichkeit eines gefahrbringenden Ausfalls pro Stunde, englisch Average Probability of Dangerous Failure per Hour [34], oder bei niedriger Anforderungsrate um die Wahrscheinlichkeit eines gefahrbringenden Ausfalls bei Anforderung, englisch Average Probability of Dangerous Failure on Demand (PFD_D) [169].

Der PFH_D hängt von zusätzlichen Mitteln zur Analyse von Maschinen und somit geplanter oder umgesetzter Sicherheitsfunktionen, wie der Kategorie, dem Diagnosedeckungsgrad, dem Ausfall infolge gemeinsamer Ursachen und dem Performance Level ab [34]. Die Kategorie beschreibt, wie ein Teil des SRP/CS in Bezug auf seine Widerstandsfähigkeit gegen Fehler und sein Verhalten bei einem Fehler eingestuft wird [34]. Die Einstufung erfolgt aufgrund der Struktur der Komponenten, der Fehlererkennung und/oder deren Zuverlässigkeit [34]. Der Diagnosedeckungsgrad (DC) ist ein Maß für die Wirksamkeit von Diagnosemaßnahmen. Er wird als Verhältnis der Ausfallrate der bemerkten gefährlichen Ausfälle zur Ausfallrate aller gefährlichen Ausfälle definiert [34]. Ein Ausfall aufgrund gemeinsamer Ursachen, englisch Common Cause Failure (CCF), bezieht sich auf Ausfälle mehrerer Einheiten aufgrund eines einzelnen Ereignisses, ohne dass die Ausfälle durch direkte Wechselwirkungen zwischen den Einheiten verursacht werden [34]. Das Performance Level (PL) gibt an, wie gut eine Sicherheitsfunktion unter vorhersehbaren Bedingungen arbeitet. Der PL kann in bekannte Sicherheitsintegritätslevel (SIL) umgewandelt werden, wie in Tabelle 2.1 dargestellt. SIL 4 wird nicht berücksichtigt, da SIL 4-Systeme gemäß den Anforderungen im Maschinenbau nicht bekannt sind [170].

SIL	PL	PFH_D
-	a	10^{-5} bis 10^{-4}
1	b	$3*10^{-6}$ bis 10^{-5}
1	c	10^{-6} bis $3*10^{-6}$
2	d	10^{-7} bis $< 10^{-6}$
3	e	10^{-8} bis $< 10^{-7}$
4	-	$\geq 10^{-9}$ bis $< 10^{-8}$

Tabelle 2.1: Ausfallgrenzwerte für die vier Sicherheitsintegritätslevel [5, 34]

Die aus Tabelle 2.1 hervorgehenden PL legen somit die Vorgaben für einzuhaltende PFH_D und erforderliche Systemstruktur. Ausgehend von der Risikobeurteilung wird das erforderliche PL, englisch PL required (PL_r), als Attribut einer umzusetzenden Sicherheitsfunktion festgelegt [34]. Es ist quasi ein Maß für die erforderliche Risikominderung je Sicherheitsfunktion bzw. erforderliche Integrität, welche erreicht werden muss im Hinblick auf die Ausfallwahrscheinlichkeit und Vermeidung systematischer Fehler in der Entwicklung [34]. Eine ausführliche Definition zum Zusammenhang und der Berechnungsgrundlagen finden sich in Abschnitt 2.2.1 unter der sicherheitsbezogenen Blockdiagrammmethode (SBBM).

Die Risikobeurteilung stellt nach der ISO 12100 das Gesamtverfahren bestehend aus der Risikoanalyse, Risikobewertung und Risikominimierung in der Maschinenindustrie dar [163]. Grundlage für eine Risikobeurteilung stellen die Gefährdungen, englisch Hazards, die bei der bestimmungsgemäßen Verwendung einer Maschine auftreten und zu Verletzungen oder Sachschäden führen [163]. Aufbauend auf den Gefährdungen entstehen Situationen, also Sachlagen, bei denen mindestens eine Person einer Gefährdung ausgesetzt ist [163]. Aufbauend auf den Gefährdungssituationen entstehen Risiken (R), die als Funktion R=WxS mit der Wahrscheinlichkeit des Eintritts eines Schadens (W) und seines Ausmaßes (S) definiert sind [163]. Beispiele für das Schadensausmaß können leichte, schwere oder tödliche Verletzung einer oder mehrerer Personen sein. Die Eintrittswahrscheinlichkeit definiert sich wiederum aus einer Funktion aus Gefährdungsexposition, Eintritt von Gefährdungsereignissen und der Möglichkeit zur Vermeidung oder Begrenzung [163]. Mit anderen Worten sprechen wir von der Häufigkeit des Gefährdungsereignisses, der Wahrscheinlichkeit der Ursachen, die zu den Auswirkungen führen, und der Vermeidbarkeit der Gefährdungssituation. Ein etwaiges Restrisiko nach der Risikoanalyse definiert das verbleibende bzw. tolerierbare Risiko, nachdem Schutzmaßnahmen ergriffen wurden. Eine Risikobewertung findet somit vor und nach der Anwendung von Sicherheitsfunktionen Einsatz. Um dieses Prozedere zu verdeutlichen, findet sich im folgenden Unterabschnitt in Abbildung 2.3 eine schematische Übersicht, die das methodische Vorgehen zur Risikobeurteilung und Risikominderung beschreibt.

Methodisches Vorgehen und Arbeitsweise des Sicherheitsingenieurs

Eine Visualisierung des bereits grob beschriebenen Risikobeurteilungsprozesses nach ISO 12100 kann über das folgende in Abbildung 2.3 beschriebene Aktivitätsdiagramm gegeben werden. Im Gegensatz zur ISO 12100 fließt hier, wie bereits in Hauke et al. [170] beschrieben, der iterative Prozess zur Gestaltung der sicherheitsbezogenen Teile der Steuerung nach ISO 13849-1 mit ein. Eine ausführliche Beschreibung dieses Prozesses findet sich in Abbildung 2.4.

Zu Beginn wird die Maschine genauer betrachtet, um ihre Grenzen zu bestimmen. Dabei werden Verwendungsgrenzen, räumliche Grenzen, zeitliche Grenzen und weitere Faktoren berücksichtigt [163]. Dieser Schritt der Risikoanalyse ist besonders wichtig, da er den Rahmen für die gesamte Risikobeurteilung setzt. Anhaltspunkte zum Verständnis finden sich in Anhang A.4.

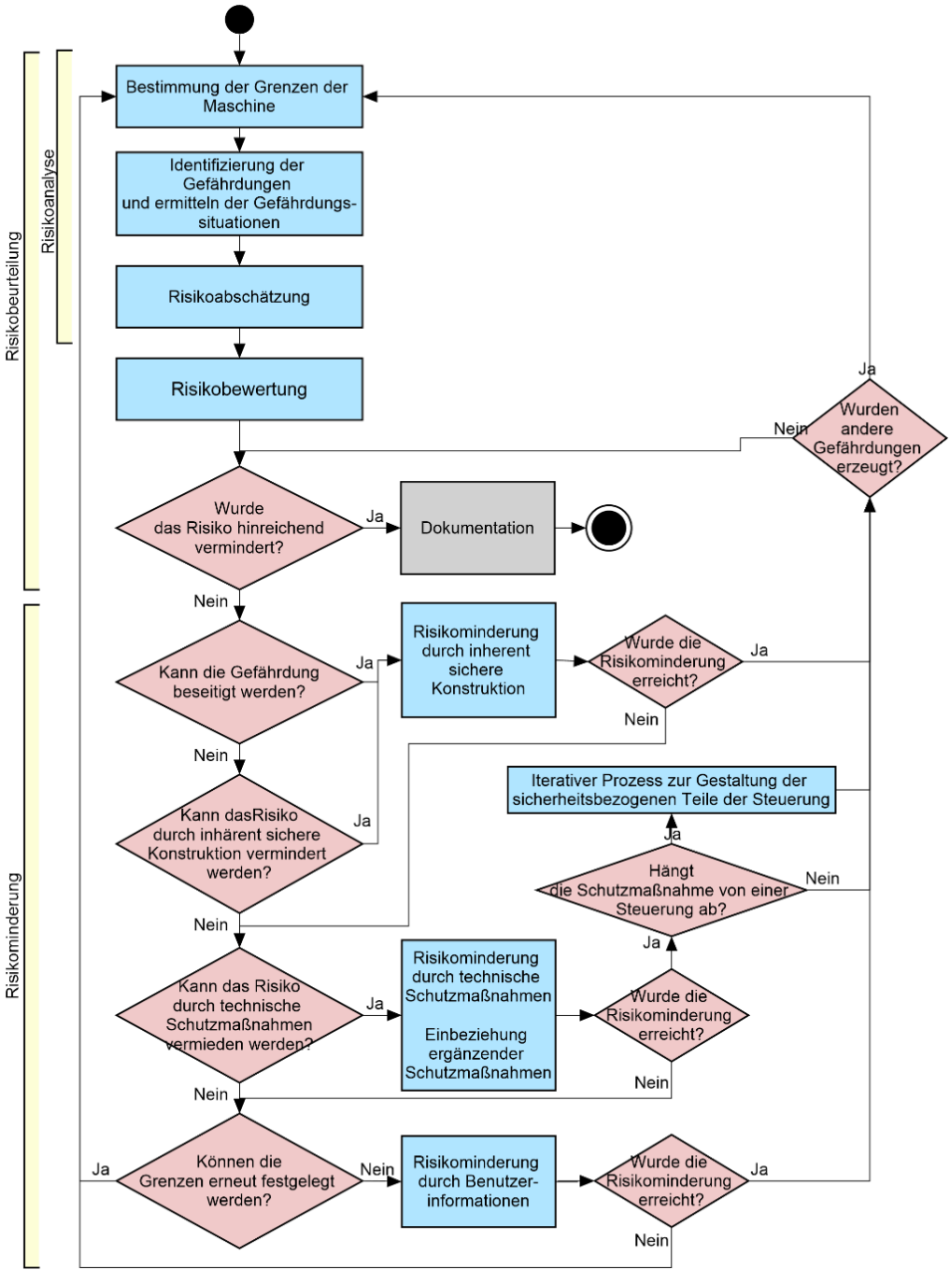


Abbildung 2.3: Ablauf einer Risikobeurteilung nach ISO 12100 [163, 170]

Im nächsten Schritt der Risikoanalyse werden mögliche Gefährdungen identifiziert, die im Zusammenhang mit der Maschine auftreten können. Hierbei werden relevante Gefahren berücksichtigt, die beispielsweise durch bestimmte Funktionen oder Betriebsarten entstehen können [163]. Eine Analyse erfolgt auch in Bezug auf die Lebensphase der Maschine, in der eine Gefährdung auftreten kann [163]. Nach ISO 12100 sind die Lebensphasen außerbetrieblicher Transport, innerbetrieblicher Transport, Montage, Installation, in Betrieb nehmen, Betrieb, Einlernen, Instandhaltung, Fehlersuche und Beseitigung, Reinigung, Wartung, außer Betrieb nehmen, Demontage und Entsorgung zu berücksichtigen [163]. Zur Einordnung von Gefährdungen gibt sie Gruppen von Gefährdungen an. Es wird unterscheiden zwischen mechanischen, elektrischen und thermischen Gefährdungen, sowie Gefährdungen durch oder im Zusammenhang mit Lärm, Vibration, Strahlungen, Material und Substanzen, Ergonomie, der Einsatzumgebung und Kombinationen [163]. Eine ausführliche Auflistung mit Beispielen findet sich in Anhang A.5.

Abschließend erfolgt in der Risikoanalyse die Risikoabschätzung anhand des Schadensausmaßes und der Eintrittswahrscheinlichkeit, wie bereits zuvor beschrieben. Im vierten Arbeitspaket der Risikobeurteilung, der Risikobewertung, werden die Ergebnisse der Risikoabschätzung zusammengeführt und bewertet. Basierend auf den Ergebnissen der Risikobewertung wird entschieden, ob Maßnahmen erforderlich sind. Eine Risikominderung ist notwendig, wenn das vorhandene Risiko größer als das akzeptable Grenzrisiko ist. Wenn Risikominderungsmaßnahmen durchgeführt wurden, muss der Prozess der Risikobeurteilung erneut gestartet werden.

Risikomindernde Maßnahmen umfassen konstruktive Maßnahmen, technische Maßnahmen und Benutzerinformationen. Letztere dürfen nur Einsatz finden, wenn eine weitere Minderung durch konstruktive oder technische Schutzmaßnahmen nicht oder nur mit nicht zumutbaren Aufwand, der die Funktion der Maschine beeinträchtigt, umgesetzt werden können [163]. Konstruktive Schutzmaßnahmen umfassen beispielsweise Entgraten von Blechen, um Schneidstellen oder gefährliche Stoßstellen zu vermeiden. Technische Schutzmaßnahmen umfassen Sicherheitsfunktionen, unabhängig, ob diese über eine SRASW gesteuert werden oder es sich um rein mechanische, elektrische oder elektronische Lösungen zur Risikominderung handelt.

Sollte es sich um eine über SRASW umgesetzte Sicherheitsfunktion handeln, ist das in Abbildung 2.4 dargestellte Vorgehen zur Risikominderung durch Gestaltungsleitsätze zur Entwicklung von SRP/CS umzusetzen [34]. Gemäß den Anforderungen müssen sämtliche Aktivitäten darauf abzielen, Fehler zu vermeiden, die während des Software-Lebenszyklus auftreten können [34].

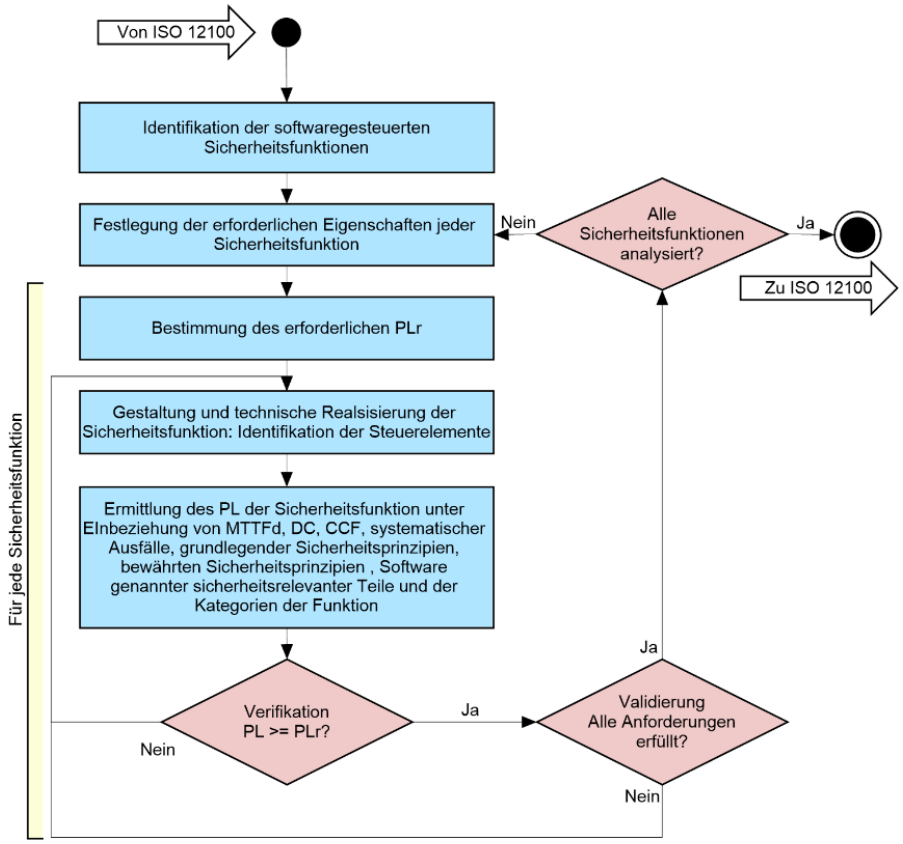


Abbildung 2.4: Iterativer Prozess zur Gestaltung der SRP/CS [34, 170]

Das primäre Ziel dieser Anforderungen ist es, gemäß Vorgehensmodellen, wie dem V-Modell, eine lesbare, verständliche, testbare und wartungsfähige Software zu erstellen. Ein vereinfachter Software-Lebenszyklus kann angewendet werden, wenn vorbewertete sicherheitsrelevante Hardware- und Softwaremodule in Kombination mit einer LVL Einsatz finden [171]. Anhand des bereits definierten Risikos lässt sich aus einer Funktion von Schadensausmaß und Eintrittswahrscheinlichkeit ebenso das PL_r berechnen [34]. Details zur Berechnung bzw. Abschätzung finden sich in Abschnitt 2.2.1.

Aufbauend auf PL_r ist die Schutzmaßnahme so zu gestalten, dass diese die erforderliche Sicherheitsfunktion realisieren kann. Anhand der Parameter zu $MTTF_d$, DC, CCF und systematischer Ausfälle einzelner Bauteile und Baugruppen und deren Struktur lässt sich das erreichte PL der Sicherheitsfunktion berechnen und feststellen, inwieweit dieses das PL_r erreicht [34]. Sofern Sicherheitsfunktionen überprüft wurden und die Realisierung den Vorgaben entspricht, kann zurück in Abbildung 2.3 gesprungen werden und die Risikobeurteilung und Risikominderung entsprechend dem vollständig durchlaufenen Plan abgeschlossen werden.

2.1.4 Gemeinsames Informationsmodell

Der deutsche Maschinenbau ist einer der bedeutendsten Wirtschaftsfaktoren Europas. Laut dem Verband Deutscher Maschinen- und Anlagenbauer (VDMA) hatte er im 2021 einen Anteil von 35 % am europäischen Maschinenbau-Umsatz [172]. Aufgrund der ständigen Einführung innovativer Fertigungstechnologien ist Deutschland ein äußerst wettbewerbsfähiger Standort [173]. Dies verdeutlichen auch die Innovationsausgaben von rund 17 % im Jahr 2021 [174].

Maschinen

Unter Maschine zählt jede mechanische Vorrichtung, die Kraft oder Energie überträgt und menschliche Arbeit einspart [175]. Dies beinhaltet unter anderem Kraftmaschinen, Arbeitsmaschinen, Transportmaschinen und Informationsmaschinen. Die 2006/42/EG gilt für eine breite Palette von Maschinen und Geräten, jedoch nicht für Hebezeuge und Aufzüge [168]. Beispiele für Maschinen aus Praxis und Alltag sind Werkzeugmaschinen, landwirtschaftliche Maschinen, Baumaschinen, Transportmaschinen, Produktionsmaschinen und Prüfmaschinen [176]. Allerdings können diese Maschinenarten weiter untergliedert werden, wie Dreh-, Fräs- und Bohr- aus Werkzeug- [177] oder Zugprüf-, Druckprüf- und Batterieprüf- aus Prüfmaschinen.

Aufgrund dieser umfassenden Definition soll im Folgenden der Fokus auf dem Sondermaschinenbau liegen. Diese Disziplin wird als eine der anspruchsvollsten im Maschinenbau angesehen, da sie kontinuierlich neue planerische und handwerkliche Herausforderungen an die Kreativität stellt [178]. Eine Sondermaschine ist eine Maschine, die für eine spezielle Aufgabe oder Anforderung entwickelt und hergestellt wird [179], wie beispielsweise für eine spezielle Fertigungsaufgabe, die Automatisierung von Produktions- und Arbeitsprozessen, die Verarbeitung von Materialien oder den Test von herstellerspezifischen Produkten. Im Vergleich zum Serienmaschinenbau wird diesem jedoch weniger Beachtung geschenkt, was in Forschungslücken in Planungs-, Entwicklungs- und Herstellungsprozessen der Sondermaschinen resultiert [178]. Oft werden Sondermaschinen komplett neu entwickelt oder aus bestehenden Maschinen und Maschinenteilen modifiziert. Jede Sondermaschine kann in Maschinenteile oder Module unterteilt werden, die je nach Anwendungsfall unterschiedlich strukturiert sind. Typischerweise besteht jede Maschine aus Eingabeeinheiten, Kontrolleinheiten, Energieversorgungen, einer mechanischen Konstruktion und einer Umgebung, und umfasst oft bewegende Teile, Messeinheiten oder Schutzeinrichtungen zum Schutz vor Gefahren. Je nach Maschinentyp kann diese Unterteilung weiter spezialisiert werden. Beispielsweise besitzen Prüfmaschinen spezielle bewegende Teile wie Portalsysteme, Roboterarme oder Translationsrotations-Systeme, die wiederum Mehrachssysteme repräsentieren [176]. Durch die Erstellung einer solchen Repräsentation über

Modelle entsteht die Grundlage für ein gesamtheitliches Systemmodell interdisziplinärer Entwicklungsprozesse, das einen Zusammenhang zwischen der realen Umgebung, der Systemmodelle und der Analysemodelle schafft. Dabei werden sicherheitskritische Systeminformationen in den Systemmodellen analysiert und durch die Analysemodelle auf Konformität überprüft.

Informationsaustausch in der Anwendungsdomäne

Die Aktualisierung und Pflege von Anforderungen über den gesamten Entwicklungsprozess hinweg legt die Grundlage für eine Nachverfolgbarkeit zu den fachspezifischen Modellen [180, 181]. Insbesondere bei Änderungen von Anforderungen können Mehraufwände entstehen, da Arbeitsschritte mehrfach ausgeführt werden müssen. Ein generelles Problem bei der Kommunikation entsteht durch die unterschiedlichen Sichtweisen und Modelle, die erkannt und verstanden werden müssen, um relevante Inhalte extrahieren zu können [134]. So entstehen schnell redundante Entwicklungs-Inhalte, die zu Verwirrungen bis hin zu kostenaufwändigen Fehlern führen. Klassische Beschreibungsmittel bei der Systementwicklung umfassen dabei keine Möglichkeiten zur Eingabe spezieller Informationen, wie Kollisionsmöglichkeiten oder Fehlerszenarien, die für Fehlermeldungen oder beim Service von Interesse sind [134]. Gesetzliche Vorgaben, wie [34, 163, 168] verlangen dabei eine ausführliche Dokumentation. Mit steigender Komplexität der Entwicklungsaufgabe wird aber auch die Dokumentation der Entwicklung komplexer, was zu einem erhöhten Aufwand führt [134]. In vielen Fällen erfolgt die Dokumentation zudem manuell, wodurch sie oft sehr minimalistisch ausfällt [182]. Dies führt wiederum zu nicht dokumentierten Änderungen und unvollständiger oder fehlerhafter Dokumentation.

Vor dem Hintergrund der steigenden Komplexität suchen Forscher nach Lösungen, um diese zu vereinfachen und zu optimieren, auch weil Systems Engineering für eine vollständige Abbildung in SysML zu umfangreich ist [183]. Einige schlagen vor, das Problem in einfachere Teilprobleme aufzuteilen oder kollaborative Designprozesse zu verfolgen. Es gibt Beispiele wie Constraint-basierte Methoden [184] und integrierte Plattformen [185], doch die Komplexität entsteht durch die Interaktion und die Schwierigkeiten bei der Konstruktion. Hierbei kann beispielsweise eine Hierarchie von Modellen helfen, um die Modellierung von Maschinen übersichtlicher zu gestalten [183]. Dies beinhaltet unter anderem die Modellierung des Verhaltens jeder Baugruppe und jedes Elements, aber auch die Verwaltung der Hierarchie von Modellen [183].

Obwohl sich viele Arbeiten wie Mhenni et al. [186] zur Black-Box und White-Box-Analyse oder Barbadienne et al. [187] zur Modellierung thermischer Effekte mit neuen SysML-Profilen für domänenspezifische Zwecke mechatronischer Systeme beschäftigen, konzentrieren sich

diese meist auf allgemeines Systemdesigns und nicht auf bestimmte Ingenieuraufgaben [32]. Unter Forschungsarbeiten zum CAD-Design und -Analyse findet sich Ben Hadj et al. [188], die eine Methode zur Generierung eines Montageplans auf Grundlage der Vereinfachung des CAD-Modells vorschlagen, indem Verbindungsteile des CAD-Modells entfernt werden. Eine weiterführende Arbeit zur möglichen Standardisierung einer Kommunikationsmethode zwischen MBSE und CAD findet sich in Brahmi et al. [32]. Diese schlagen ein SysML-Profil für das mechanische Design vor. Es besteht aus zwei Hauptphasen: der Struktur in einem Blockdefinitionsdiagramm (BDD), um die Zusammensetzung darzustellen, und der Generierung ihrer Architektur in einem Internal-Blockdiagramm (IBD), zur Identifikation der internen Architektur. Das in Abbildung 2.5 dargestellte Profil ermöglicht den Datenaustausch, um die Kommunikation zwischen Stakeholdern bei der Überprüfung und Validierung der mechanischen Entwicklung zu erleichtern [32]. Demnach setzt sich eine mechanische Baugruppe `Assembly` aus verschiedenen Teilbaugruppen `Subassembly` zusammen, welche wiederum aus Teilbaugruppen und mechanischen Komponenten `Part` bestehen [32]. Der `Assembly`-Stereotyp definiert Gesamtteilenummer, Anzahl der Teilbaugruppen, Gewicht, Volumen und Fläche, der `Subassembly`-Stereotyp Informationen über Anzahl bestehender Teile und der `Part`-Stereotyp mechanische Komponenten mit Name, Volumen, Fläche, Gewicht, Größe und Dicke jedes Teils.

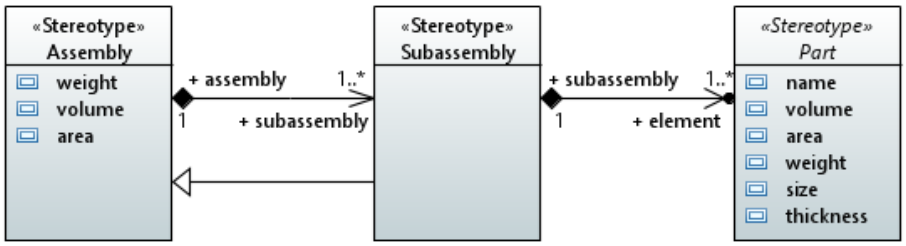


Abbildung 2.5: SysML-Profile für mechanische Baugruppen nach [32]

Zusätzlich beschreibt Brahmi et al. drei weitere zu modellierende Artefakte für die Baugruppenarchitektur: Kontaktfläche, Art der Verbindung und Justierung [32], welche aufgrund der Auslegung auf rein mechanische Prozesse nicht weiter betrachtet werden sollen. Eine weitere interessante Arbeit findet sich jedoch in Drave et al. [33]. Sie formalisieren und erweitern, basierend auf Koller et al. [189] und Pahl et al. [190], ein Konzept namens SysML4FMArch für den Maschinenbau und nutzen dieses in einem interdisziplinären Projekt. Obwohl SysML demnach seine Nachteile in Bezug auf Formalität und Intuitivität hat [33], weisen die Ergebnisse darauf hin, Modellierungssprachen zur Erklärung der funktionalen Architektur und zur Assimilation von Wissen aus Designkatalogen für einen ganzheitlichen modellbasierten Ansatz zu nutzen, was die Kluft zwischen Stakeholdern durch Abstraktion verringert. Die entwickelten

Metamodelle fokussieren sich allerdings eher auf die detailgetreue Nachbildung, als auf die Unterteilung zur Kommunikation und des gegenseitigen Verständnisses. Mhenni et al. [191] hingegen unterteilt über Stereotypen bereits in mechanische, elektrische, elektronische hydraulische, pneumatische und Softwarekomponenten, die über Ports miteinander verbunden werden können.

Um Designkataloge zu entwickeln, müssen also Generalisierungen von Baugruppen und Unterbaugruppen in der Maschinenindustrie abhängig des Maschinentyps, des Einsatzzwecks und der Sichtweise getroffen werden. Es lassen sich grundlegende gemeinsame Baugruppen und Abhängigkeiten zwischen den Stakeholdern erkennen.

Mechanische Baugruppen

Die mechanischen Baugruppen sind unerlässlich für den Maschinenbetrieb und übertragen und transformieren Energie zwischen verschiedenen Komponenten [145]. Diese Baugruppen müssen von anderen Stakeholdern strukturell und funktional verstanden werden [134], um Abhängigkeiten zu E/E/PE-Baugruppen zu erkennen und Grundlagen für die Analyse der funktionalen Sicherheit zu bieten [163]. Strukturelle Baugruppen sind den mechanischen Baugruppen untergeordnet und bieten die Grundlage für die Konstruktion der Maschine sowie schützen vor äußeren Einflüssen und Schäden [145]. Beispiele hierfür sind Rahmen und Abdeckungen.

Elektrische und elektronische Baugruppen

Die elektrischen und elektronischen Baugruppen sind für die Übertragung und Kontrolle von elektrischer Energie in der Maschine verantwortlich [156]. Programmierbare elektronische Baugruppen bilden die Grundlage für die Entwicklung von Softwarealgorithmen, um die verschiedenen Aspekte der Maschine zu messen und zu bewerten und die Bewegung der Maschine zu kontrollieren [151]. Die resultierenden Schaltpläne bilden die Grundlage für die Bewertung der Risikominderung [34].

Hydraulische und pneumatische Baugruppen

Die hydraulischen und pneumatischen Baugruppen sind für die Übertragung und Kontrolle von Energie mittels Flüssigkeiten und Gasen innerhalb der Maschine verantwortlich [192, 193]. Die Dimensionierung, Auslegung und Berechnung der Baugruppen findet in der mechanischen Konstruktion statt [193], während die Struktur in einem Schaltplan nach DIN ISO 1219-2 [194] dargestellt wird. Diese Baugruppen sind oft nicht auf mechanische Systeme beschränkt, sondern hängen von Sensoren und elektronischen Ventilen ab, was sie Teil des Schaltplans macht.

Software

Die Software besteht aus vernetzten Klassen, Komponenten oder Blöcken, zur Steuerung und Überwachung der Maschine [25, 26]. Dies umfasst eine benutzerfreundliche Schnittstelle für die Programmierung und Bedienung der Maschine sowie softwaregesteuerter Sicherheitsfunktionen. In der Maschinenindustrie finden Entwicklungen zumeist auf Basis von SPSen oder Sicherheits-SPSen statt, die mit einer anerkannten Teilmenge der Programmiersprachen in IEC 61131 [35] programmiert werden [34]. Es existieren hierfür bereits Arbeiten, die versuchen, die Notationen von IEC 61131-3 mit UML oder SysML zu integrieren [36, 87, 159, 160, 183, 195, 196]. Forschungsergebnisse weisen darauf hin, dass mehr Diagrammtypen verwirrend sein können [183]. Stattdessen sind, wie in [195] gezeigt, angepasste Teilbereiche von UML, wie Klassen- und Zustandsdiagramme zu verwenden. Basierend auf den Ergebnissen in [195] wurde plcML entwickelt, ein domänenspezifisches Profil für SPSen von Witsch [197]. Es reduziert die Anzahl der Notationen auf drei, bestehend aus Klassen, Aktivitäts- und Zustandsdiagrammen. Diese werden in [197] und [198] mit ihren Modelltransformationen und Modellformalisierungen in OCL beschrieben, um Codegenerierung und bidirektionale Abbildung für IEC 61131-3-Entwicklungsumgebungen zu ermöglichen. Leider eignet sich das plcML-Framework nicht für den Einsatz in einer gemeinsamen Modellierungsebene, da es sich auf die Softwareentwicklung und weniger auf andere genannte Entwicklermodelle konzertiert. Allerdings kann nachgewiesen werden, dass Klassen- und Zustandsdiagramme zur Softwaremodellierung eingesetzt werden können, um diese im Anschluss automatisch in Funktionsblockdiagramme zu transformieren [36, 87, 159, 160]. Die Entwicklung von diversen Codegeneratoren für endliche Automaten ist somit als gelöst anzusehen und wird in dieser Arbeit nicht weiter betrachtet. Stattdessen soll der Fokus auf der Entwicklung notwendiger gemeinsamer Modelle liegen, die als Basis für Generatoren dienen.

2.1.5 Diskussion

In diesem Abschnitt wurde das Forschungsziel des Typs Beobachtung im Bereich der maschinenbauspezifischen System-, Software- und Sicherheitsmodellierung vorgestellt. In diesem Kontext wurden verschiedene Grundlagen und Ansätze für die Entwicklung eines MBSE-Frameworks geschaffen, um das Ziel zu erreichen. Bestehende Frameworks wurden gezeigt, die Informationen zum Gesamtprozess eines MBSE-Frameworks liefern und auf die Unterstützung von SysML und XMI-Exportformaten abzielen. Verschiedene MBSE-Ansätze wurden identifiziert, die für die Entwicklung genutzt werden können. Eine Einführung mit Standards und Begrifflichkeiten der funktionalen Sicherheit des Maschinenbaus wurde ebenfalls gegeben.

Das dargelegte Problemfeld führt zur Notwendigkeit einer besseren Unterstützung des Entwicklungsprozesses durch Unterteilung einer Maschine in gemeinsame Teile und Module. Informatiker müssen Wege finden, um die Zusammenarbeit dieser Disziplinen effizienter zu gestalten und einen Bruch bei der Modellierung und dem Datenaustausch zu vermeiden. Die Integration der Maschinenbauingenieure, Elektroingenieure, Informatiker und Sicherheitsingenieure, die direkt am Entwicklungsprozess beteiligt sind, ist von besonderer Bedeutung. Im weiteren Verlauf dieser Arbeit wird daher ein generischer Ansatz vorgestellt, der jeden benannten Bereich der Systementwicklung und der sicherheitsbezogenen Entwicklung unterstützt.

Es kann festgestellt werden, dass MBSE im Generellen ein bekanntes und gut verstandenes Thema ist. Trotzdem gibt es keinen einheitlichen Standard oder Metadatenmodell für MBSE, was eine weitere Herausforderung darstellt. Insbesondere die Abbildung der Stakeholder-spezifischen Artefakte und die Korrelation zwischen den Stakeholdern sind problematisch, was im Modellierungsteil dieser Arbeit ausführlich behandelt wird. Jeder Stakeholder benötigt typischerweise eine zentrale Datenstruktur, um seine Sichtweise umzusetzen. Die Modellierung dieser Sichtweisen wird durch die Forschungsaufgabe FA1-TB angegangen und durch die Implementierung FA1-I sowie die entsprechende Bewertung FA1-E bestätigt.

In diesem Abschnitt wurde eine offene Herausforderung (OH) identifiziert, die die Entwicklung eines modularen Informationsmodells und Frameworks mit einer Unterteilung von Maschinenteilen und Stakeholder-spezifischen Baugruppen und Elementen für eine vereinfachte Anwendbarkeit zum Ziel hat. Dies wird in Tabelle 2.2 dargestellt.

OH1.1: Ein modulares Informationsmodell und Framework, das es ermöglicht, die verschiedenen Stakeholder in einer Modellebene zu integrieren.

Tabelle 2.2: Verbleibende Herausforderung bzgl. maschinenbauspezifischer Modellierung

Zusammenfassend kann man sagen, dass die Forschungsaufgabe vom Typ Beobachtung im Bereich einer gemeinsamen Modellebene für den Maschinenbau beantwortet wurde und noch bestehende Herausforderungen identifiziert wurden. Ein ähnlicher Ansatz wird auch zur Beantwortung der Forschungsaufgabe vom Typ Beobachtung im Bereich der Integration von Sicherheitsanalysemethoden des Maschinenbaus verfolgt, die im nächsten Abschnitt folgt.

2.2 Integration der Analysemethoden

Wie bereits während des zuvor beschriebenen Kapitels verdeutlicht, sind Sicherheitsanalysemethoden ein wichtiges Glied der sicherheitsbezogenen Entwicklung, von der Risikobeurteilung bis hin zum Nachweis der Risikominderung [199]. In diesem Abschnitt werden daher übliche und spezifische Analysemethoden des Maschinenbaus ausführlich beleuchtet und ausgewählte Literaturergebnisse zur Integration in ein Informationsmodell präsentiert, um die Aspekte der Problembeschreibung PB2 zu behandeln. Die Struktur dieses Abschnitts entspricht der Literaturrechercheaufgabe in diesem Bereich. Unterabschnitt 2.2.1 bietet einen Überblick über relevante Analysemethoden und deren Anwendung. Im Unterabschnitt 2.2.2 finden sich bestehende Forschungen zur Integration von Analysemethoden in MBSE-Plattformen und der Unterabschnitt 2.2.3 geht auf die Werkzeugintegration zur Analyse ein. Wie bereits zuvor findet sich zum Schluss des Abschnitts eine kurze Diskussion.

2.2.1 Sicherheitsanalysemethoden im Maschinenbau

Die Grundnorm [7] und domänenspezifische Standards, wie ISO 13849 [34], unterscheiden zwischen verschiedenen Arten von Sicherheitsanalysemethoden. Hierbei finden sich Unterscheidungen in induktive und deduktive Ansätze sowie in qualitative und quantitative Ansätze. Induktive Analysemethoden beginnen mit spezifischen Beobachtungen und ziehen daraus allgemeine Schlussfolgerungen, wie beispielsweise Failure Mode and Effects Analysis (FMEA) [17, 200]. Sie sind in der Regel qualitativ, aber sie können auch quantitative Anteile aufweisen, wenn sie zum Testen und Bewerten von Hypothesen verwendet werden [11]. Deduktive Analyse beginnen mit allgemeinen Theorien oder Annahmen und testen diese durch spezifische Beobachtungen und Daten, wie beispielsweise die statische Fault Tree Analysis (FTA) oder die dynamische Fault Tree Analysis (DFT) [17, 200]. Deduktive Analysen können ebenso qualitativ, beispielsweise zur Identifizierung von Ausfallszenarien oder quantitativ für probabilistische Berechnungen dienen [11]. Unabhängig der spezifischen Analysen sind diese so früh wie möglich durchzuführen, um im Fall negativer Ergebnisse Zeit- und Finanzierungsaufwand zu minimieren. Für gültige Aussagen sollten sie jedoch erst nach Abschluss von Teilen der Architektur oder Module verwendet werden. Im Zusammenhang mit MBSE nutzt man sie also nach Abschluss der ersten Modelle und vor jeglicher Implementierung. Je detaillierter die Analysen auf das System und die Spezifikationen angewendet werden, desto größer das Vertrauen in das entwickelte System.

Eine Übersicht relevanter Analysen findet sich in Abbildung 2.6. Sie erhebt keinen Anspruch auf Vollständigkeit. Neben den dort beschriebenen Analysen finden sich ebenso die Erstellung

von Monte-Carlo-Simulationen, generalisierte stochastische Petri-Netze (GSPN) oder Markov-Automaten [11], die im Weiteren allerdings nur nebenbei als formale Berechnungsmodelle erwähnt und nicht tiefergehend betrachtet werden sollen.

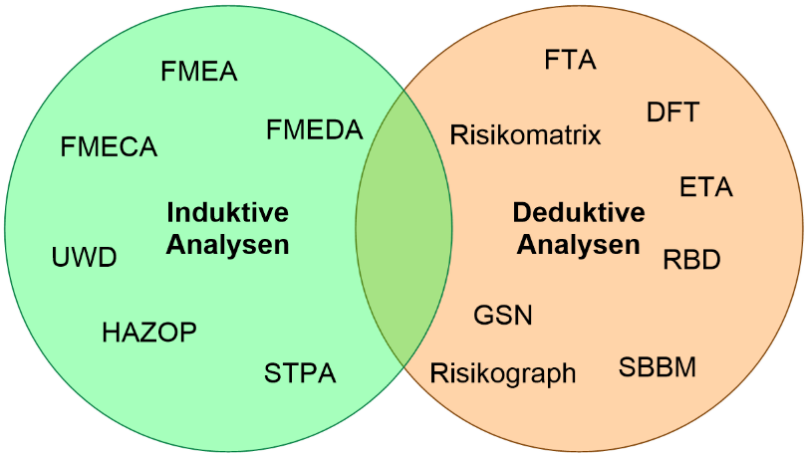


Abbildung 2.6: Unterscheidung induktive und deduktive Analysen
FME(C/D)A = Failure Mode, Effect and (Criticality/Diagnostic) Analysis; FTA = Fault Tree Analysis; DFT = Dynamic Fault Tree Analysis; ETA = Event Tree Analysis; RBD = Reliability Blockdiagram; UWD = Ursache-Wirkungsdiagramm; GSN = Goal Structuring Notation; HAZOP = Hazard & Operability Study; STPA = Systems Theoretic Process Analysis; SBBM = sicherheitsbezogene Blockdiagrammmethode

Eine Übersicht mit Zuordnung der Analysen zu den spezifischen Standards und Domänen findet sich in Tabelle 2.3, grau hinterlegt, die typischen Analysemethoden der Maschinenindustrie.

Analysen	Grundnorm IEC 61508 [5, 6, 7, 8, 9, 10, 11]	Maschinen ISO 13849 [17] IEC 62061 [169]	Automobil ISO 26262 [14]	Prozess IEC 61511 [15]	Nuklear IEC 62671 [201]	Luft- & Raumfahrt DO-178 [13]
FMEA [202]	X	X [17, 169]	X	X	X	X
FMECA [202]		X [17]	X	X	X	X
FMEDA [202]			X		X	X
FTA [203]	X	X [17, 169]	X	X	X	X
DFT [203]		X [17, 169]	X	X	X	X
ETA [204]	X	X [17]	X	X		
RBD [205]	X	X [169]	X	X		
HAZOP [14]	X		X	X	X	X
UWD [204]	X		X			
STPA [206]			X			
GSN [207]			X			
Risikograph [9, 34]	X	X [17, 34]				
Risikomatrix [9]	X	X [169]				
SBBM[34]		X [34]				

Tabelle 2.3: Analysen zugeordnet zu Domänen
Legende unter Abbildung 2.6

Basierend auf den identifizierten Analysen von Tabelle 2.3 sollen für die Maschinenindustrie relevante Analysen beschrieben werden, um das bestehende Wissen zu vertiefen und Herausforderungen herauszuarbeiten. Im Weiteren werden nur die Analysemethoden FTA, DFT als Erweiterung von FTA, Risikographen und die sicherheitsbezogene Blockdiagrammmethode

(SBBM) ausführlich erläutert, da sie wesentliche Bestandteile des zu entwickelnden gemeinsamen Informationsmodells darstellen sollen. Die Methoden FMEA, FMECA, ETA, RBD und Risikomatrix werden aufgrund ihrer geringeren Relevanz für die spezifischen Anforderungen der ISO 13849 oder bereits hinreichend bekannten Modellierungstechniken in Anhang A.6 behandelt. Weitere in Tabelle 2.3 aufgezählte Methoden, wie Goal Structuring Notation (GSN) und Systems Theoretic Process Analysis (STPA) können ebenso dort nachgelesen werden.

FTA

Die Fehlerbaumanalyse (FTA) ist ein systematischer Ansatz zur Identifikation und Bewertung potenzieller Fehler oder Ausfälle in einem technischen System. Sie wird häufig verwendet, um Risiken und Schwachstellen in Systemen aufzudecken, geeignete Maßnahmen zur Fehlervermeidung oder -minderung zu entwickeln und die Ausfallraten sicherheitsrelevanter Systemfehler zu bestimmen [200]. Die FTA verwendet eine graphische Notation, den Fehlerbaum, der, wie in Abbildung 2.7 dargestellt, aus logischen Verknüpfungen, wie AND-, OR- und XOR-Gates sowie Ereignissen bestehen kann [203]. Jedes Ereignis im Baum ist entweder ein Basisereignis `BasicEvent`, das direkt zum Fehler führt, oder ein logisches Ereignis wie `IntermediateEvent` und `TopEvent`, das aus anderen Ereignissen abgeleitet wird [11].

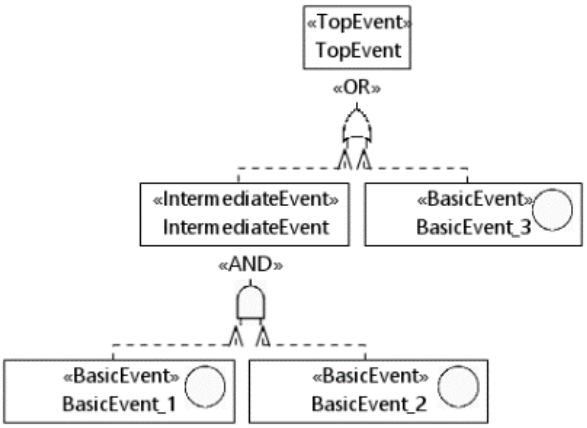


Abbildung 2.7: Einfaches FTA Beispiel

Die FTA besteht aus vier Schritten: Systemdefinition, Fehlerbaumkonstruktion, qualitative Bewertung und quantitative Bewertung [203]. Die Systemdefinition beschreibt das System, seine Funktionen und mögliche Systemausfälle [49]. Basierend darauf entsteht die Fehlerbaumkonstruktion, also die Struktur des Fehlerbaums, durch Identifikation der Ereignisse, die zu unerwünschtem Systemverhalten führen können [49]. Dadurch können zum einen qualitativ Ausfallszenarien identifiziert, semi-quantitativ Szenarien gemäß ihren Wahrscheinlichkeiten eingeordnet und quantitativ probabilistische Berechnungen des resultierenden Systemausfalls

durchgeführt werden [11, 49]. Für letzteres müssen die Wahrscheinlichkeiten der Basisereignisse bekannt sein, beispielsweise aus empirischen Daten oder Expertenbewertungen. Die IEC 61025 [203] beschreibt einen tieferen Einblick und eine detaillierte Vorgehensweise.

FTA wird häufig in fast allen sicherheitsrelevanten Bereichen wie der Eisenbahnindustrie, der Luft- und Raumfahrtindustrie oder der Kernkraftindustrie eingesetzt [208]. Es existieren zahlreiche Veröffentlichungen wie [48, 49, 70, 77, 186, 209, 210, 211, 212, 213, 214], die bereits dazu beitragen FTA in ein MBSE-Framework zu integrieren. Allerdings können statische Fehlerbäume das Verhalten in praktischen Systemen, vor allem für probabilistische Berechnungen, aufgrund ihrer zu simplen Annahmen oft nicht ausreichend detailliert beschreiben. Darüber hinaus fehlen gemeinsame Abhängigkeitsmuster wie die Verwaltung von Ersatzteilen, Redundanzen, die Modellierung von zeitlichen Sequenzen von Fehlern und zeitliche Abhängigkeiten [71, 208].

Vergleich zu FMEA

FTA und FMEA werden in der Maschinenindustrie oft in Kombination verwendet, um die Sicherheit und Zuverlässigkeit von Maschinen und Systemen zu gewährleisten. Obwohl sich die beiden Methoden in einigen Aspekten ähneln, gibt es Unterschiede in ihrem Einsatzbereich, der Herangehensweise und den Ergebnissen.

Der größte Unterschied zwischen FTA und FMEA besteht in ihrer Herangehensweise. FTA folgt einem Fehlerereignis bis zu allen Ursachen, um potenzielle Fehlerquellen zu identifizieren, und FMEA betrachtet systematisch mögliche Fehlerquellen, um deren Auswirkungen auf das System zu analysiert. FTA eignet sich daher eher für die Analyse komplexer Maschinen, während FMEA besser für die Analyse von Einzelkomponenten geeignet ist. Das Ergebnis von FTA liefert potenzielle Ursachen nach Wahrscheinlichkeit für einen Fehler und deren Auswirkungen, während FMEA eine Liste von potenziellen Fehlern, geordnet nach Priorität und basierend auf ihren Wahrscheinlichkeiten und Auswirkungen, liefert. Durch die Kombination von FMEA und FTA können Ingenieure potenzielle Fehler in jedem Aspekt eines Systems oder einer Maschine identifizieren und Prioritäten setzen, um effektive Lösungen zu entwickeln.

DFT

Dynamische Fehlerbäume sind die am weitesten verbreitete Erweiterung von statischen Fehlerbäumen und werden in verschiedenen Bereichen eingesetzt. Ihr Ursprung ist in Dugan et al. [215] zu finden, in dem eine große Anzahl von Varianten von DFTs vorgeschlagen wurde. Eine Übersicht über die Erweiterungen in Bezug zur eben beschriebenen FTA findet sich in Ruijters

et al. [216]. Im Vergleich zu statischen Fehlerbäumen und um dessen Einschränkungen zu überwinden, werden die in Abbildung 2.8 dargestellten Gates für DFTs eingeführt [217].

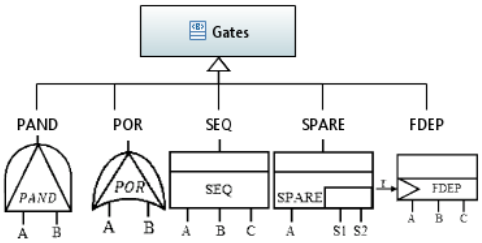


Abbildung 2.8: Gate-Darstellungen nach Aslansefat et al. [217]

Das Priority-AND Gate (PAND) definiert eine bestimmte dynamische Version des AND Gates. Das Ergebnis ist wahr, wenn beide Eingänge auftreten, aber das Ereignis A vor dem Ereignis B auftritt [217]. Das Priority-OR Gate (POR) verhält sich ähnlich, ist jedoch eine Disjunktion anstelle einer Konjunktion. Ereignis A muss zuerst auftreten, damit das Ergebnis wahr ist, jedoch müssen nicht beide Ereignisse gleichzeitig auftreten [217]. Das Sequence-Enforcing Gate (SEQ) definiert ein sequentielles Fehlerverhalten. Ereignis A muss vor Ereignis B auftreten und Ereignis C muss nach Ereignis B auftreten, damit das Ergebnis wahr ist [217]. Das SPARE Gate wird verwendet, um Redundanz im Systemdesign zu modellieren, wobei die Eingänge immer Basic Events sind [217]. Es besteht aus einer primären Komponente und mehreren Ersatzkomponenten bzw. Sparekomponenten. Wenn die primäre Komponente ausfällt, werden die Sparekomponenten nacheinander aktiviert [217]. Schließlich wird das Ergebnis des Gates wahr, wenn alle Eingänge wahr sind. Ein SPARE Gate kann drei Arten von Redundanz darstellen: Cold Standby Spare (CSP), Warm Standby Spare (WSP) und Hot Standby Spare (HSP). CSP bedeutet, dass ein Ersatzteil erst aktiviert werden muss, um die primäre Komponente im Falle eines Ausfalls zu ersetzen [217]. Zum Beispiel wäre das Gate nicht bestanden, wenn der Sensor zur Registrierung des Ausfalls vor der primären Komponente ausfällt. HSP stellt das Gegenteil dar. Das Ersatzteil läuft parallel zur primären Komponente und kann jederzeit ersetzt werden [217]. WSP stellt eine Zwischenlösung dar, bei der ein Ersatzteil teilweise parallel zur primären Komponente betrieben wird und bei Bedarf ersetzt werden kann. Die Ersatzkomponenten werden in einem reduzierten Bereitschaftszustand gehalten, bis sie benötigt werden [217]. Zuletzt, das Functional Dependency Gate (FDEP) zeigt die funktionale Abhängigkeit verschiedener Ereignisse von einem Trigger-Ereignis an [217]. Es wird verwendet, wenn die Funktionalität einer Komponente von dem Betrieb anderer Komponenten in einem System abhängt. Ein triviales Beispiel ist in der Regel die Versorgungsspannung, ohne die die angeschlossenen Komponenten ausfallen würden. Der Ausfall eines Ereignisses A, B oder C hat jedoch keinen Einfluss auf das Trigger-Ereignis.

Neben diesen dynamischen Gates umfassen DFTs kontinuierliche Zeitberechnungen, die die Entwicklung von Systemfehlern im Laufe der Zeit berücksichtigen. Fehlertolerante Systeme verfügen dabei oft über integrierte Wiederherstellungsmechanismen. Faktoren wie Abdeckung, Wiederherstellung und Einzelpunktfehler, die mit der grundlegenden Ereignisspezifikation verbunden sind, helfen in einigen Fällen, solche Wiederherstellungsmechanismen zu modellieren [218]. Das Fehlerverhalten von Komponenten wird in der Regel durch eine Wahrscheinlichkeitsfunktion spezifiziert, die für jeden Zeitpunkt die Wahrscheinlichkeit angibt, dass die Komponente noch nicht ausgefallen ist [216]. Wenn Komponenten ohne Auswirkungen auf andere Komponenten repariert werden können, haben die Fehlerbäume eine zusätzliche Reparaturzeitverteilung [216]. Wie Ausfallverteilungen werden Reparaturzeitverteilungen oft als exponentiell verteilt angenommen und durch eine Reparaturrate charakterisiert [217]. Ein weiterer wichtiger Faktor, insbesondere für SPARE-Gates, ist eine Ruhephase, in der eine Komponente nicht genutzt wird oder nur eine Redundanz darstellt [208]. Der Abdeckungsfaktor ist die Wahrscheinlichkeit, dass der Ausfall eines Bauteils nicht zum Ausfall des Systems führt, es sei denn, dies ist durch den Fehlerbaum angegeben [219]. Ausfallverteilungen können oft angemessen durch inverse Exponentialverteilungen approximiert werden, wie in der Grundnorm oder Ruijters et al. [216] beschrieben. Es gibt jedoch auch andere Verteilungen, wie die Weibull- oder LogNormal-Verteilung, die an dieser Stelle jedoch nicht vertieft behandelt werden sollen. Eine kurze Erläuterung zu Berechnungsmethoden und zum Einsatz findet sich in Basavalingappa et al. [220].

Es existieren bereits Arbeiten wie [210, 212, 214, 221] zur Integration in MBSE. Einen interessanten Ansatz bietet Baklouti et al. [221]. Hier wurde ein eigenes DFT-Profil entwickelt und ein Redundanzprofil genutzt, um die Redundanzinformationen in das Systemmodell zu integrieren, um DFTs automatisch aus internen Blockdiagrammen zu generieren. Der Algorithmus nutzt dabei eine Tiefensuche durch das IBD des Systems, um die relevanten Komponenten und deren Verbindungen zu identifizieren und die entsprechenden DFTs zu erstellen. Aktuell existiert allerdings keine Integration von DFTs in ein MBSE-Framework, das den Austausch von Artefakten über Analysen hinweg erlaubt und die Möglichkeit bietet, es domänenspezifisch zu erweitern.

Vergleich zu FTA

Da DFTs eine Erweiterung der FTAs darstellen, erlauben sie die Analyse viel komplexerer Systeme. Im Gegensatz zu FTA, die statisch ist und einen festen Zustand des Systems zur Analyse verwendet, berücksichtigt DFT die dynamische Veränderung des Systems im Laufe der Zeit. Dadurch wird es möglich, mögliche Fehlerzustände und deren Auswirkungen dynamisch

während des Betriebs zu analysieren und zu bewerten und Ereignisse zu analysieren, die zu einem bestimmten Zeitpunkt unter bestimmten Bedingungen auftreten, und nicht nur für Ereignisse, die jederzeit auftreten. Dies ermöglicht es, Ausfallwahrscheinlichkeiten auf Basis gezielter Vorhersagen potenzieller Fehler und mindernden Maßnahmen zu berechnen, um normative Vorgaben einzuhalten.

Risikograph

Die Risikographmethode ist ein qualitatives Verfahren zur Risikoeinschätzung, um die Auswirkungen von potenziellen Gefahren zu quantifizieren. Sie benutzt dabei die bekannten Größen der Eintrittswahrscheinlichkeit und des Schadensausmaßes, um die Beurteilung sicherheitsbezogener Teile der Steuerung durchzuführen [169]. Jedem Risiko wird eine Kombination von Kategorien zugeordnet, die normalerweise durch Zahlen oder Farben dargestellt werden, um die Risikostufe und Sicherheitsanforderungen zu kennzeichnen. Dadurch soll sichergestellt werden, dass die gefahrenmindernden Maßnahmen wie Sicherheitsfunktionen die Maschine effektiv absichern und den Anforderungen der geltenden Normen entsprechen. Für die Maschinensicherheit wird der Risikograph in ISO 13849 [34] beschrieben.

Ein Risikograph ermöglicht nach der Grundnorm die Bestimmung des SIL durch Kenntnis der Risikofaktoren [9]. Mit der Einführung der Parameter Häufigkeit des Auftretens einer Gefährdung und Möglichkeit zur Vermeidung oder Begrenzung einer Gefährdung kann die Art einer Gefährdungssituation besser beschrieben werden [9]. Durch Auswahl und Kombination jeweils eines Wertes, aus den ebenso in Abbildung 2.9 dargestellten vier verschiedenen Sätzen von Parametern, kann somit das SIL zugehörig zu jeder Sicherheitsfunktion bestimmt werden [9].

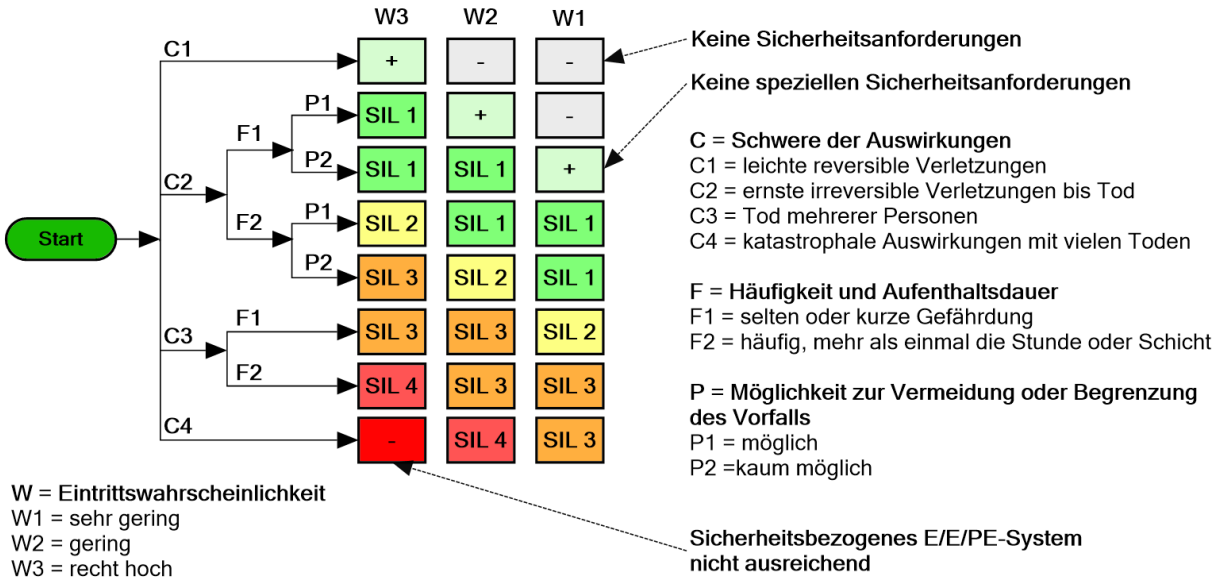


Abbildung 2.9: Klassifizierung eines Risikographen nach IEC 61508 [9]

Der Risikograph nach ISO 13849, in Abbildung 2.10 dargestellt, vereinfacht das Konzept aus der Grundnorm. Statt vier existieren nur noch zwei Schweregrade und auch Eintrittswahrscheinlichkeit eines Gefährdungsereignisses kann nur als niedrig bewertet werden, wenn bereits Zuverlässigkeitsdaten und Unfallgeschichten von vergleichbaren Maschinen existieren [34].

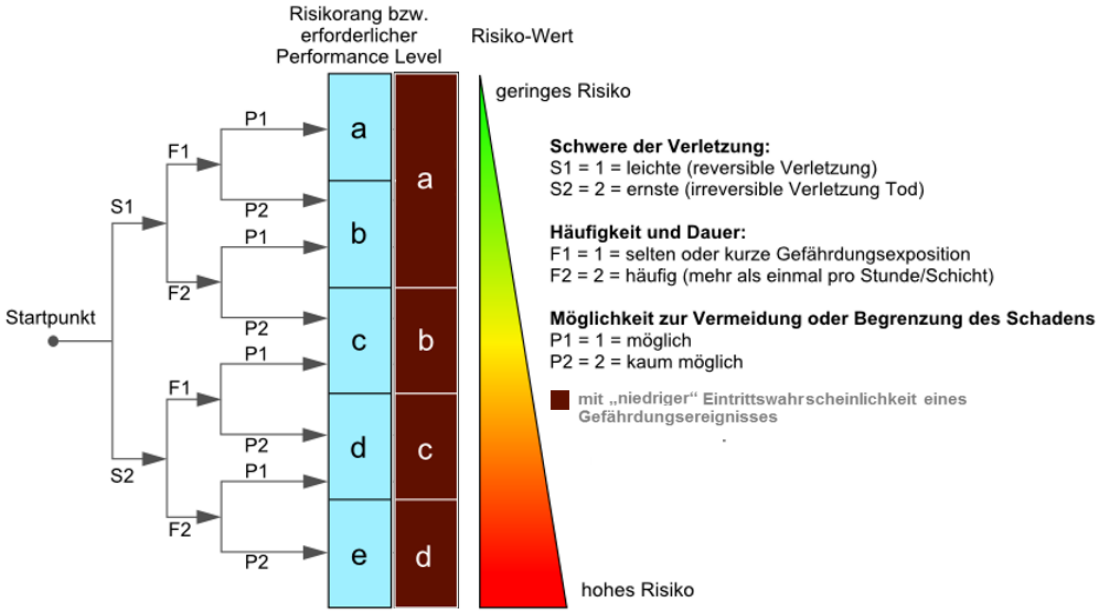


Abbildung 2.10: Risikograph nach ISO 13849 [34]

Zusätzlich zur direkten Bestimmung von SIL und PL_r kann zum Ziel der Risikoabschätzung eine Risikoprioritätszahl, im Folgenden auch Risikoindex genannt, erzeugt werden, die abhängig der bekannten Funktion zur Risikoberechnung für Stakeholder ein Feedback gibt, inwieweit die risikomindernden Maßnahmen ausreichend das Risiko mindern.

Laut den betrachteten Maschinennormen gibt es keine eindeutige Festlegung der Parameter und der Funktion zur Bestimmung der Risikoprioritätszahl. Somit kann beispielsweise über das Äquivalent zum SIL und PL_r eine Berechnung stattfinden, die eine Abschätzung des Risikos darlegt, in der ein SIL1 oder PL_r a keine risikomindernden Maßnahmen zur Folge hat.

Eine Erläuterung und ein Vergleich zur Risikomatrixmethode, die das Äquivalent zur Risikographmethode aus IEC 61061 [169] darstellt, findet sich in Anhang A.6.5.

Vergleich zu FMEA

In der Maschinenindustrie sind die Risikographmethode und FMEA weit verbreitete Methoden zur Risikoanalyse. Obwohl sie ähnliche Ziele haben, gibt es Unterschiede in ihren Ansätzen. Beide helfen dabei, potenzielle Risiken und Fehlerquellen in einem Prozess oder System zu identifizieren und zu bewerten, um präventive Maßnahmen zur Risikominderung zu ergreifen. Es finden jedoch unterschiedliche Bewertungen statt. Ein Risikograph basiert im Wesentlichen

auf einer zweidimensionalen Skala über Eintrittswahrscheinlichkeit und Auswirkungen zur Berechnung des Risikos auf Basis von Expertenwissen, während FMEA auf einer Bewertung von Risiken durch Identifizierung von potenziellen Fehlern, ihren Auswirkungen und ihrer Wahrscheinlichkeit sowie deren mathematischen Berechnungen und Datenanalysen beruht. Die Risikographmethode ist somit besser geeignet für die Beurteilung von Risiken, die auf Erfahrungswissen und subjektiven Einschätzungen basieren, während FMEA besser geeignet ist für die Analyse von Risiken, die auf verfügbaren Daten und Fakten basieren. In der Maschinenindustrie sind beide Methoden relevant. Die Risikographmethode hilft bei der Bewertung im Rahmen der Risikobeurteilung und die FMEA kann helfen, potenzielle Fehler in den Maschinenkomponenten und -prozessen zu identifizieren und die Risiken von Fehlfunktionen zu minimieren.

SBBM

Bei der sicherheitsbezogenen Blockdiagrammmethode (SBBM) handelt es sich um eine vereinfachte Methode in Bezug zum Reliability Blockdiagramm (RBD), um die Zuverlässigkeit von Sicherheitsfunktionen auf SRP/CS in der Maschinenindustrie zu überprüfen [34]. Sie abstrahiert dabei die physikalischen Verbindungen zwischen Bauteilen, um das Modell auf die logischen Zusammenhänge zu beschränken [222]. Hierbei ist jedes Bauteil in einer Sicherheitsfunktion Bestandteil einer bestimmten Struktur, die nach ISO 13849 [34] als Kategorie bezeichnet wird. Im aktuellen Entwurf der ISO 13849-1 [171] von 2021 wird dies bereits als Subsystem, also einer Teilfunktion, zusammengefasst. Ein Subsystem kann dabei aus einem oder mehreren Blöcken bestehen, die Eingabe-(I), Logik-(L) oder Ausgabeeinheiten (O) repräsentieren [171]. Es kann abhängig der vier Kategorien aus einem oder zwei Kanälen bestehen, wobei in der Kategorie 2 zusätzlich Testeinrichtungen (TE) und Ausgänge von Testeinrichtungen (OTE) vorkommen. Eine Kurzbeschreibung zur Struktur und der Abstufungen findet sich in Tabelle 2.4.

Vereinfacht kann eine direkte Korrelation zu PL_r gezogen werden, wobei mindestens Kategorie B oder 1 für PL_r b, Kategorie 2 für PL_r c, Kategorie 3 für PL_r d und Kategorie 4 für PL_r e vorgesehen werden sollte [170]. Es ist natürlich ebenso möglich, mit Kategorie 2 PL d zu erreichen, doch ist dieses Vorgehen nicht empfehlenswert. Ein Ablaufdiagramm findet sich in Anhang A.6.6 unter Strukturanalyse.

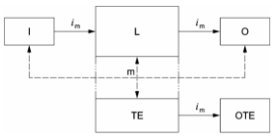
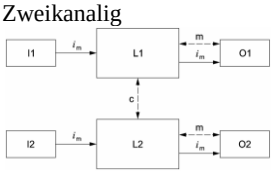
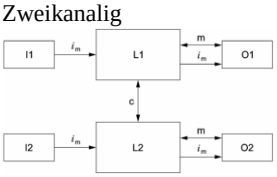
Kat.	Struktur	Anforderungen	Systemverhalten
B	Einkanalig	SRP/CS und/oder ihre Schutzeinrichtungen sowie ihre Bauteile müssen in Übereinstimmung mit den Normen so gestaltet, gebaut, ausgewählt, zusammengebaut und kombiniert werden, dass sie den zu erwartenden Einflüssen standhalten, und dabei grundlegender Sicherheitsprinzipien angewendet werden	Das Auftreten eines Fehlers kann zum Verlust der Sicherheitsfunktion führen.
1	Einkanalig	Die Anforderungen von B müssen erfüllt sein. Bewährte Bauteile und bewährte Sicherheitsprinzipien müssen angewendet werden.	Wie Kategorie B, aber die Wahrscheinlichkeit des Auftretens ist geringer als in Kategorie B.
2	Einkanalig, getestet 	Die Anforderungen von B und die Verwendung bewährter Sicherheitsprinzipien müssen erfüllt sein. Die Sicherheitsfunktion muss in geeigneten Zeitabständen durch die Maschinensteuerung getestet werden	Das Auftreten eines Fehlers kann zum Verlust der Sicherheitsfunktion zwischen den Tests führen. Der Verlust der Sicherheitsfunktion wird durch den Test erkannt.
3	Zweikanalig 	Die Anforderungen von B und die Verwendung bewährter Sicherheitsprinzipien müssen erfüllt sein. Sicherheitsbezogene Teile müssen so gestaltet werden, dass • ein einzelner Fehler jedes Teils nicht zum Verlust der Sicherheitsfunktion führt, und • wenn in angemessener Weise durchführbar, der einzelne Fehler erkannt wird.	Wenn ein einzelner Fehler auftritt, bleibt die Sicherheitsfunktion immer erhalten. Einige, aber nicht Fehler werden erkannt. Eine Anhäufung von unerkannten Fehlern kann zum Verlust der Sicherheitsfunktion führen.
4	Zweikanalig 	Die Anforderung von B und die Verwendung bewährter Sicherheitsprinzipien müssen erfüllt sein. Sicherheitsbezogene Teile müssen so gestaltet werden, dass • ein einzelner Fehler in jedem dieser Teile nicht zum Verlust der Sicherheitsfunktion führt, und • der einzelne Fehler bei oder vor der nächsten Anforderung der Sicherheitsfunktion erkannt wird. Wenn diese Erkennung nicht möglich ist, darf eine Anhäufung von unerkannten Fehlern nicht zum Verlust der Sicherheitsfunktion führen.	Wenn ein einzelner Fehler auftritt, bleibt die Sicherheitsfunktion immer erhalten. Die Erkennung von Fehleranhäufungen reduziert die Wahrscheinlichkeit des Verlustes der Sicherheitsfunktion (hoher DC_{avg}). Die Fehler werden rechtzeitig erkannt, um einen Verlust der Sicherheitsfunktion zu verhindern.

Tabelle 2.4: Merkmale und Darstellung der Kategorien [34]

Nachdem die Struktur der Sicherheitsfunktion steht, ist für jedes Element der $MTTF_D$- und DC-Wert anzugeben [34]. Um den $MTTF_D$-Wert eines Elements berechnen zu können, wird oft der $B10_D$-Wert von Herstellern mechanischer oder elektromechanischer Komponenten angegeben. Der $B10_D$ Wert gibt die Anzahl der Zyklen an, nachdem ein Element gefährlich ausfällt [34]. In Abhängigkeit von n_{op}, der Zyklen pro Jahr, lässt sich somit $MTTF_D$ berechnen. Weitere Möglichkeiten zur Berechnung bieten die mittlere Zeit zwischen zwei Fehlern, im Englischen Mean Time Between Failure (MTBF), MTTF oder λ im Zusammenhang mit RDF. MTTF und MTBF können gleichgesetzt werden, sofern Mean Time To Repair (MTTR), also die mittlere Zeit bis zur Wiederherstellung mit nahe Null angenommen wird [171]. Formel 2.2 fasst diese Zusammenhänge zusammen. Sollte kein RDF-Wert vom Herstellen angegeben werden, kann ein Anteil von 50 % aller Ausfälle als gefährliche Ausfälle angenommen werden.

$$MTTF_D = \frac{B_{10}}{0{,}1 * n_{op} * RDF} = \frac{MTTF}{RDF} = \frac{MTBF - MTTR}{RDF} = \frac{1}{8760h * \lambda * RDF}$$

Formel 2.2: Berechnung $MTTF_D$ eines Elements [34]

Die Abschätzung des DC eines Elements bezieht sich auf die Diagnosemaßnahmen, die von Eingabe-, Logik- oder Ausgabeeinheiten umgesetzt werden. Jedem Element ist eine Maßnahme und somit ein prozentualer DC-Wert zuzuordnen. Eine ausführliche aus der ISO 13849 exportierte Liste dieser Maßnahmen mit zugeordneten DC findet sich in Anhang A.6.6 unter Abschätzung des DC. Eine ähnliche Vorgehensweise, wie für den DC kann für CCF und das gesamte Subsystem umgesetzt werden, wobei sich die Maßnahmen gegen CCF additiv verhalten können. Eine ausführliche Auflistung zur Punktevergabe und Quantifizierung von Verfahren findet sich in Anhang A.6.6 unter Verfahren zur Punktevergabe & Quantifizierung für Maßnahmen gegen CCF. Die Berechnungsgrundlage, die SBBM-Werkzeuge nutzen, um den PFH_D-Wert jeder Subfunktion aus $MTTF_D$, DC und CCF sowie den PFH_D-Gesamtwert einer Sicherheitsfunktion zu berechnen, findet sich ebenso in Anhang A.6.6 unter Berechnungsgrundlage für SBBM. Eine Besonderheit stellen gekapselte Subsysteme dar [171]. Wenn der Hersteller eines Bauteils bereits PL, PFH_D und Kategorie angibt, kann das Element als Subsystem betrachtet werden und benötigt keinen zusätzlichen $MTTF_D$, DC- oder CCF-Wert.

Aktuell finden sich keine Veröffentlichungen zur SBBM in Bezug zu MBSE-Frameworks.

Vergleich zu DFT

Obwohl DFT und SBBM das gleiche Ziel haben, Sicherheitsrisiken zu bewerten, unterscheiden sie sich in ihrem Ansatz. DFT ist eine allgemeinere Methode zur Analyse von Ausfallereignissen, während SBBM speziell der Bewertung der Sicherheitsfunktionen von Steuerungen dient. Das heißt, während DFTs verschiedene Verteilungsfunktionen auf Einzelereignisse und ihre Abhängigkeitsmuster anwenden können, analysiert SBBM die Gesamtwirkung von Sicherheitsfunktionen auf die Gesamtsicherheit des Systems über eine detailreiche Erweiterung zu $MTTF_D$ und PFH_D mittels DC und CCF und der Schaltplan-ähnlichen Anordnung über die Kategorien. Beide Methoden sind jedoch proaktive Ansätze zur Risikobewertung und Sicherheitsanalyse und anerkannte Standards in der Maschinenindustrie. Sie berücksichtigen die Wirkung von Sicherheitsfunktionen auf die Gesamtsicherheit des Systems und helfen bei der Entwicklung und Umsetzung geeigneter Sicherheitsmaßnahmen. Je nach Anwendung und Zielsetzung können beide Methoden eingesetzt werden, um die Sicherheit von Maschinen und Anlagen zu verbessern und potenzielle Unfälle und Verletzungen zu vermeiden.

2.2.2 Sicherheitsanalyseerweiterungen in MBSE-Frameworks

Wie bereits beschrieben, existiert ein wachsendes Interesse am Einsatz semiformaler Methoden und MBSE, um Entwicklungsprozesse zu erleichtern, was sogar zur gesetzlichen Anforderung wird, wenn es sich um sicherheitskritische Systeme handelt [7, 11]. Die Grundnorm empfiehlt

ebenso den Einsatz formaler Verifikationsmethoden, jedoch ohne zu definieren, wie genau sie anzuwenden sind [223]. Dabei scheinen semiformale Ansätze zur Erfassung und Strukturierung der Anforderungen und der Systemarchitekturen den Vorgaben eher zu entsprechen [223].
In den letzten Jahren haben Forscher verschiedene Ansätze entwickelt, die MBSE nutzen, um Artefakte für Sicherheitsanalysemethoden zu integrieren und zu automatisieren. Dabei muss zwar nicht zwangsläufig SysML zur Modellierung der Systemmodelle eingesetzt werden, doch sollen aufgrund von Ambrosio et al. [28] nur SysML-Ansätze berücksichtigt werden. Im Bereich der Sicherheitsmodelle gibt es bislang keinen einheitlichen Standard, weshalb zwischen zentralen Veröffentlichungen zur losen und engen Kopplung unterschieden werden soll.

Lose Modellkopplung

Zu Beginn waren System- und Sicherheitsmodelle oft nur lose gekoppelt. Bei der losen Kopplung auf Modellebene werden unterschiedliche Sprachen zur Modellierung der Architektur und der Sicherheitsartefakte verwendet, was die Notwendigkeit von Zwischenmodelltransformationen zwischen den Darstellungen schafft. In Joshi et al. [209], Xiang et al. [210] und Hecht et al. [69] zeigen die Autoren die Fähigkeit, einen einzigen Typ von Sicherheitsartefakt aus SysML-Modellen zu generieren, allerdings basierend auf AADL-, RCM- und AltaRica-Modellen. Die Anwendung weiterer Sicherheitsanalysen ist kostspielig und schwer zu integrieren. Yakymets et al. [70] unterstützen hierbei bereits die Generierung der erforderlichen Artefakte, von der Spezifikation bis zu den Aktivitäten zur Bewertung der funktionalen Sicherheit, unter Einbeziehung der Systemarchitektur in SysML, bzw. der Erweiterung RobotML [224]. Die Forscher erzeugen für jede Gefahr eine FMECA und wandeln das SysML-Modell in ein AltaRica-Modell um, das den Fehlerbaum für jede Gefährdung erzeugt [70]. Diese lose Modellkopplung erschwert die Synchronisierung zwischen den Beteiligten aufgrund der unterschiedlichen Modellierungssprachen. Außerdem kann es aufgrund der fehlenden Re-Importe von Ergebnisdaten bei Nichterreichen der Ziele oder bei Aktualisierungen der Systemdaten schnell zu Inkonsistenzen zwischen den Sicherheitsartefakten und dem Systementwurf kommen [48].

Enge Modellkopplung

Um die Einschränkungen der zuvor beschriebenen Ansätze zu umgehen und Datenverluste zu vermeiden, wurden integrative Ansätze entwickelt. Durch die Erstellung und Verwaltung von Modellen in derselben Ebene können potenzielle Datenverluste vermieden und eine engere Kopplung aufgebaut werden. Diese Methode ermöglicht es, Sicherheitsartefakte über Profile zu integrieren, was eine effektive Zusammenarbeit zwischen Stakeholdern erleichtert. Um die scheinbar gegensätzlichen Denkweisen von System- und Sicherheitsingenieuren miteinander in

Einklang zu bringen, haben Pétin et al. [223] 2010 einen Systemmodellierungsansatz vorgeschlagen, der nicht-formale Methoden auf der Grundlage von SysML-Anforderungen und BDD mit formalen Methoden wie der Modellprüfung kombiniert. Obwohl dieser interdisziplinäre Austausch zwischen den Ansätzen der Informatik und der Systemtechnik zeigt, die Sicherheitsintegrität auf höchstem Niveau zu verifizieren, machen laut Pétin et al. [223] Fallstudien deutlich, dass für einen praxisrelevanten technischen Rahmen noch Anstrengungen unternommen werden müssen [223]. In Thramboulidis et al. [225] wird ein Rahmenwerk zur Automatisierung des Verifikationsprozesses von IEC61131-basierten industriellen Sicherheitsanwendungen vorgestellt. Unter Verwendung einer detaillierten Beschreibung des Systems in SysML können Gefahren, ihre Ursachen und das resultierende Risiko der Gefährdung sowie die entsprechenden Sicherheitsmaßnahmen zur Minderung ermittelt werden, was einen gemeinsamen Prozess zur Risikobeurteilung ergänzt und zur Definition der Anforderungen an das Sicherheitssystem führt [225]. Es handelt sich bei [223] und [225] zwar nur um Ansätze, die keine Analysen durchführen, doch bieten sie einen ersten Ansatz für Risikographen.

Einen wichtigen Aspekt und Vorteil einer engen Modellkopplung stellt Müller et al. [226] bereits 2016 mit der Sicherheitsanalyse und SysML-Modellierung in den frühen Phasen des Systemdesigns vor. Ein "Hazard Analysis"-Profil wird eingeführt, das die systematische Identifizierung sicherheitskritischer Funktionen und Komponenten ermöglicht, um Fehler zu vermeiden und die Qualität zu verbessern [226]. Der Ansatz umfasst ein dreiphasiges Verfahren: Anforderungen spezifizieren, funktionale Architektur erstellen und diese in reale Komponenten umsetzen. Das Profil ist flexibel und kann an verschiedene Aufgaben der Sicherheitsanalyse angepasst werden [226]. Die Veröffentlichung baut auf Arbeiten wie SafeML von Biggs et al. [77] auf und zielt darauf ab, eine Lücke in der bisherigen Forschung zu schließen, indem es funktionale und sicherheitskritische Aspekte integriert. Eine ähnliche Forschungsarbeit aus 2016 von Mhenni et al. [227] stell die ebenso auf SafeML basierende SafeSysE-Methodik vor. Ziel ist es, die Komplexität moderner technischer Systeme zu bewältigen und gleichzeitig Sicherheitsaspekte frühzeitig im Designprozess zu berücksichtigen [227]. Dies soll späte und kostspielige Änderungen vermeiden. SafeSysE kombiniert Methoden wie FMEA und FTA mit der SysML-Modellierung. Dies ermöglicht es Sicherheitsexperten, Sicherheitsartefakte direkt aus den Systemmodellen zu generieren, was die Konsistenz zwischen den Systemanforderungen und den Sicherheitsanalysen enorm verbessert. Choley et al. [228] erweitern den Ansatz der SafeSysE-Methodik, indem sie sich auf die sicherheitskritischen Cyber-Physical Systems (CPS) konzentrieren.

Um die Relevanz solcher integrativen Ansätze im Sicherheitsmanagement darzustellen, kann Käßmeyer et al. [229] dienen. Diese Studie beleuchtet die Notwendigkeit konsistenter und vollständiger Spezifikationen in sicherheitskritischen Systemen, die durch den Einsatz integrativer Ansätze wie I-SafE unterstützt werden [229]. I-SafE von Fraunhofer IESE, beschrieben in Antonino et al. [230], stellt eine bereits überarbeitete Lösung dar, die die Spezifikation und Analyse von Sicherheitsanforderungen sowie die Rückverfolgbarkeit bis hin zu Architektur- und Fehlermodellen unterstützt, bietet allerdings eine umfassende Unterstützung für allgemeine Sicherheitsanalysen in der Automobilindustrie. Durch diese Spezialisierung auf den ISO 26262 Standard fehlen jedoch flexible Mechanismen, um die Integration in andere Domänen zu erleichtern. Um diese und andere Einschränkungen zu umgehen, haben die Forscher ihren Ansatz 2020 verbessert. In Moncada et al. [231] besteht das zentrale Element der sicherheitsbezogenen Modellierung aus „Hazard“, das für jede Sicherheitsanalyse von zentraler Bedeutung ist. Umgesetzte Analysemethoden bestehen allerdings lediglich aus Hazard Analysis and Risk Assessment (HARA), der Gefährdungsanalyse und Risikobewertung in der Automobilindustrie, und Component Fault Trees (CFT), einer Erweiterung von FTA für Komponenten [231]. Zudem nutzen die Forscher des Fraunhofer IESE die kommerzielle Plattform Sparx Enterprise Architect für ihren Ansatz und berücksichtigen keine Einbindung von technischen Stakeholder-Modellen.

Weitere Forschungsarbeiten, wie die von Huang et al. [232, 233] aus den Jahren 2017 und 2018, zeigen ähnliche Einschränkungen bei der Erweiterbarkeit für andere Domänen auf. Hierhin diskutieren die Autoren die Integration von FMEA in MBSE, wobei sie betonen, dass die Evolution des FMEA-Prozesses zu einem modellbasierten Ansatz viele der traditionellen, manuellen Methoden ersetzt [232]. So kann die Modellierung von Ausfallmodi und -wirkungen effizienter und konsistenter gestaltet werden, jedoch können die spezifischen Anforderungen der Zieldomäne den Einsatz in anderen Domänen einschränken [232, 233]. Der Ansatz ist hierbei tiefgehend in IBM Werkzeuge, wie Rational Rhapsody und DOORS integriert, um eine starke Integration und einheitliche Dokumentation zu schaffen. Diese Implementierung berücksichtigt aktuelle Verfahren und Praktiken, wie sie von NASA und der Air Force gefordert werden, besitzt allerdings eine so stark integrierte Methodik, so dass auch dies die Anwendbarkeit in anderen Domänen, wie der Maschinenindustrie erschweren kann. Im Weiteren exisiteren weitere Ansätze, wie das Eclipse Polarsys Safety Framework, dass ein Open-Source-Software-Framework zur Entwicklung sicherheitskritischer Systeme bereitstellt, einschließlich Anforderungen, Modellierung und Verifikation, dessen Entwicklung allerdings bereits eingestellt ist [234].

Zusammengefasst bieten diese Veröffentlichungen wertvolle Beiträge zur Verbesserung der Konsistenz und Kommunikation zwischen Entwicklungs- und Sicherheitsingenieuren durch kontinuierliche SysML-Modellierung. Die Studien und Ansätze betonen die Wichtigkeit der Anpassung und Erweiterung bestehender Methoden, um eine breitere Anwendbarkeit und Flexibilität zu erreichen, ohne die grundlegenden Prinzipien der Sicherheitsanalyse zu vernachlässigen. Basierend auf diesen Ansätzen der engen Modellkopplung von System- und Sicherheitsmodellen in der gleichen Modellebene und Ansätzen wie Berres et al. [213, 235], wurde daher der im Folgenden vorgestellte Ansatz der OMG entwickelt, der sich an den Kernideen orientiert und einen allgemeinen domänenunabhängigen zentralen Ansatz schafft.

OMG RAAML

Die Risk Analysis and Assessment Modeling Language (RAAML) der OMG erweitert die Fähigkeiten der MBSE-Sprache SysML auf den Bereich der Sicherheitstechnik durch zentrale Elemente in Kernprofilen und methodenspezifischen Profilen [48, 49]. RAAML ist hierbei eine strukturierte Sammlung von Profilen und Bibliotheken, die das in Anhang A.7.1 dargestellte auf Situationen beruhende Kernkonzept aufgreifen und in einer vernetzten Paketstruktur organisieren. Damit handelt es sich um ein feineres Kernkonzept als das des Fraunhofer IESE. Die Profile und Bibliotheken sind, in die drei Pakete Core, General und Methods unterteilt. Das Core-Paket definiert grundlegende Konzepte und Beziehungen der Situation, das General-Paket mit General Concepts-Profil und -Bibliothek erweitert das Core-Paket zur Vorbereitung für methoden- oder domainspezifische Pakete und das Methods-Paket beschreibt die verschiedenen Analysen. In der aktuellen Fassung vom März 2022 [236] umfasst RAAML FTA nach ISO 61025 [203], FMEA nach IEC 60812 [202], STPA nach [206], GSN nach [207] und ein domänenspezifisches Paket für die Automobildomäne nach [14] mit Hazard & Operability Study (HAZOP). Jedes in Methods definierte Paket besteht dabei ebenso aus einem Profil und einer Bibliothek. Ausnahme bildet GSN. Für eine kondensierte Darstellung sollen im Folgenden lediglich Core, General und FTA näher betrachtet werden. FMEA kann zum näheren Verständnis in Anhang A.7.3 nachgelesen werden.

Gemäß RAAML [236] besteht das in Abbildung 2.11 dargestellte Core-Profil aus einer Situation und verschiedenen Abhängigkeiten. Die Metaklasse `Class` und der SysML Stereotyp `Block` werden durch die Situation erweitert [236]. Der Stereotyp `Situation` dient zur Unterscheidung, zu anderen Arten von Blöcken. Um einen Kontext zu Systemmodellen zu schaffen, wird die Beziehung `RelevantTo` verwendet [236]. Ein `ControllingMeasure` repräsentiert eine Maßnahme, die ergriffen wird, um potenzielle oder reale Situationen abzuwehren, beispielsweise durch Abschwächung der Schwere, Verringerung der

Wahrscheinlichkeit des Auftretens oder Erhöhung der Wahrscheinlichkeit der Entdeckung [236]. Zuletzt wird die `Violate` Beziehung genutzt, um Abhängigkeiten zu Situationen festzulegen, die gegen Anforderungen verstoßen. Das Element `IDCarrier` dient lediglich einer menschenlesbaren Identifikation von Elementen [236].

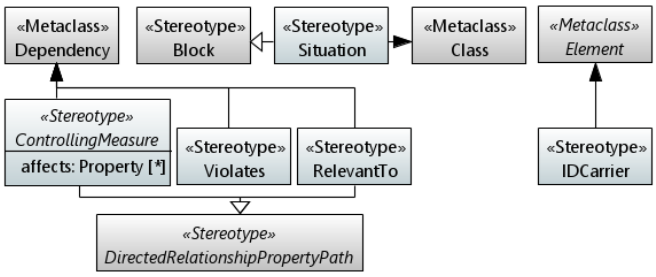


Abbildung 2.11: RAAML Core Profile [236]

Die in Abbildung 2.12 dargestellte Core-Bibliothek nutzt das Core-Profil, wobei Situationen von `AnySituation` erben. `Causality` ist die zentrale Beziehung zwischen Situationen, die Ursache-Wirkungs-Beziehungen herstellt und die Grundlage für weiteren Beziehungen bildet [236]. Diese Kausalität kann direkt, bedingt oder probabilistisch sein oder auf einer anderen Art von Situationsbeziehung beruhen, die vom Benutzer definiert wird [236].

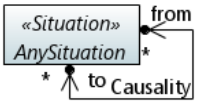


Abbildung 2.12: RAAML Core Bibliothek [236]

Das General Concepts-Profil in Abbildung 2.13 erweitert Situationen, um zwischen Ausfallart, Fehlerzustand und -verhalten sowie Störung zu unterscheiden [48]. Darüber hinaus wird der Stereotyp `ControllingAction` für die Erkennung `Detection`, die Vorbeugung `Prevention`, die Schadensbegrenzung `Mitigation`, und Empfehlungen `Recommendation` spezialisiert [48]. Das General Concepts Profil und die Bibliothek in Abbildung 2.14 bilden damit eine Basis für gemeinsame Konzepte von Sicherheitsanalysen, um Dopplungen zu vermeiden, übergreifende Ansätze umzusetzen und eine Erweiterbarkeit für weitere Analysen zu garantieren [48, 49].

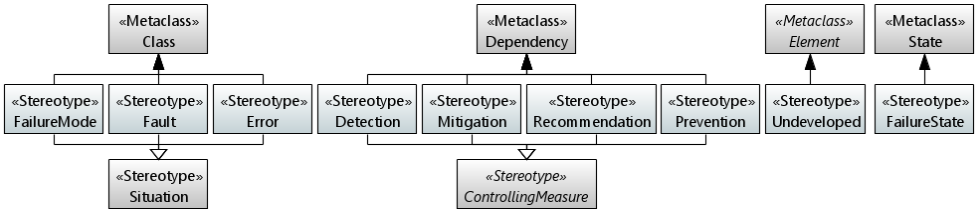


Abbildung 2.13: RAAML General Concepts Profil [236]

`AnySituation` wird für ein Ereignis `AbstractEvent`, Schadenspotenzial `HarmPotential` und Szenario `Scenario` erweitert [236]. Ein Ereignis beschreibt jede Veränderung im analysierten System. Somit spezialisiert die Bibliothek `AbstractEvent` für Ursache `Cause`, die andere Ereignisse auslöst, und dysfunktionales Ereignis `DysfunctionalEvent`, dessen Auftreten ein dysfunktionales Verhalten eines Systems oder eines Teilsystems beschreibt [236]. Eine Ursache besitzt eine Eintrittswahrscheinlichkeit `Occurence`, wohingegen sich dysfunktionale Ereignisse weiter in unerwünschte Zustände `UndesiredState`, die die Ausfallart näher beschreiben, und Effekte `Effect`, die die Auswirkungen anderer Situationen darstellen, spezialisieren [236]. Ausfallarten besitzen eine Aufdeckungswahrscheinlichkeit `Detectability` und Effekte einen Schweregrad `Severity` des Effekts. Ein Risiko `Risk` bildet eine Spezialisierung des Szenarios und baut dabei die Korrelation zwischen Ursache, Wirkung und Ausmaß auf.

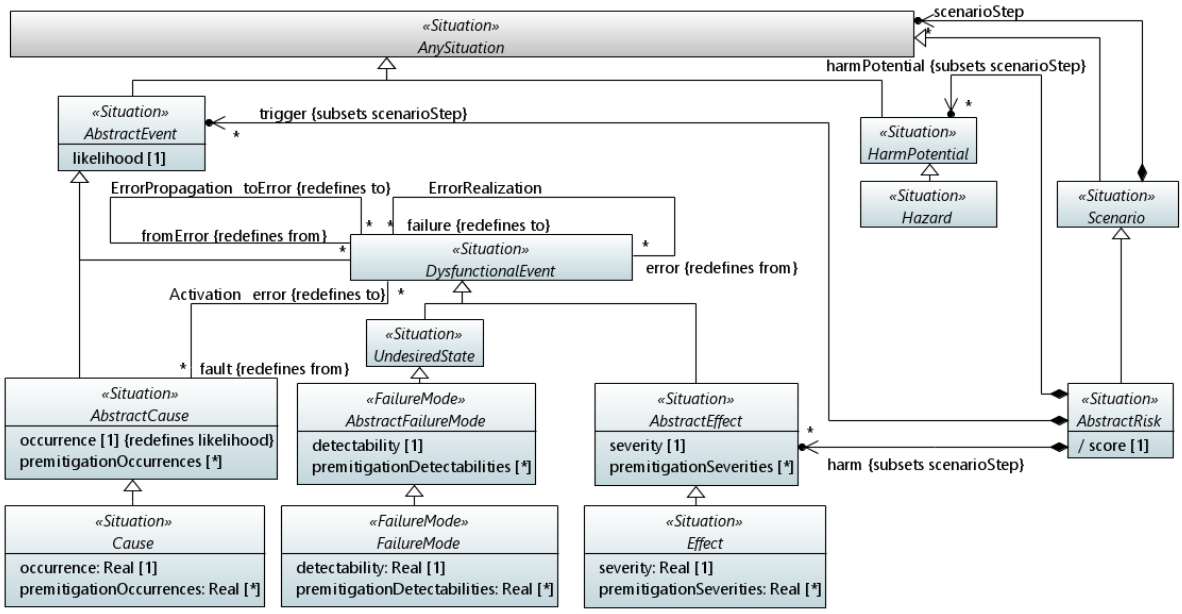


Abbildung 2.14: RAAML General Concepts Bibliothek [236]

Die FTA Bibliothek, dargestellt in Abbildung 2.15, beschreibt die verschiedenen Teile eines Fehlerbaums. Das zentrale Element stellt das `FTAElement` dar, das eine Spezialisierung aus dysfunktionales Ereignis ist und die Eigenschaft `likelihood` durch die Wahrscheinlichkeit `probability` ersetzt. Ein Fehlerbaum bildet somit ein Szenario, dass in einem dysfunktionalen Ereignis endet. Jedes FTA Ereignis enthält zusätzlich zur Wahrscheinlichkeit eine Priorität `priority`, wobei die Wahrscheinlichkeit nur bei `BasicEvent`, `ConditionalEvent`, `UndevelopedEvent` und `DormantEvent` angepasst werden kann. Die Wahrscheinlichkeiten der übrigen Ereignistypen werden bei der Analyse berechnet. Die Wahrscheinlichkeit in `FTATree` dient als Vergleichswert für das Ergebnis. Die Priorität bildet

die Grundlage für SEQ-Gates und zeigt die Möglichkeit der Erweiterbarkeit für Komponentenfehlerbäume und DFTs [49]. Ausführliche Erläuterungen zu den verschiedenen Event- und Gate-Typen finden sich in Anhang A.7.4.

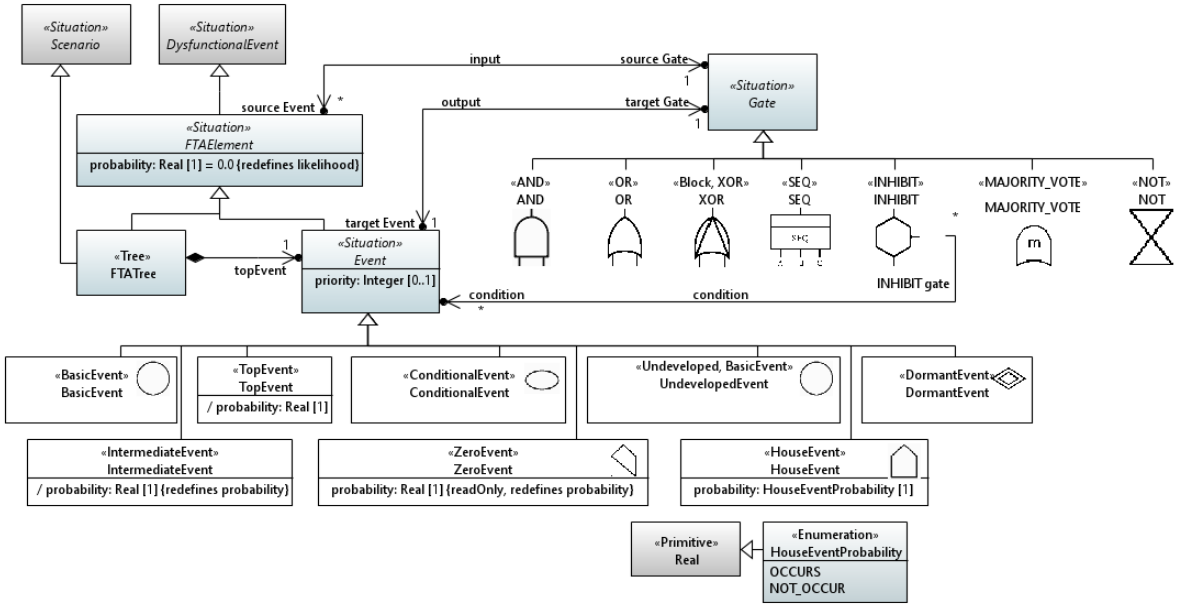


Abbildung 2.15: RAAML FTA Bibliothek [236]

Die Berechnungen zur Analyse können direkt in der SysML-Umgebung stattfinden, indem über Constraint-Blöcke und OCL Berechnungsalgorithmen integriert werden [49]. Dieses Vorgehen wird im Folgenden unter enge Toolkopplung aufgeführt, da nicht nur Modelle in der gleichen Modellebene inkludiert, sondern ebenso die Berechnungen und Analysen durchgeführt werden können. Der Vorteil liegt klar in der Unabhängigkeit von externen Werkzeugen und einer tiefergehenden Nachverfolgbarkeit der erstellten Artefakte in Bezug auf Berechnungsart und Teilproblemerfassung [49], doch entsteht das Problem der zusätzlichen Validierung und Zertifizierung der Algorithmen im realen Einsatz, die eine entscheidende Rolle in der funktionalen Sicherheit spielen [11]. Speziell bei komplexeren Algorithmen muss ein Beweis erbracht werden, dass die Algorithmen gleichwertig bisheriger Werkzeuge arbeiten.

Biggs et al. [48] zeigt allgemeine Anwendungsbeispiele für FMEA und FTA und gibt weitere Details zu domänenspezifischen Erweiterungen anhand des Beispiels der Automobilindustrie. In Berres et al. [49] wird die Anwendung von FTA genauer beschrieben, um den Beitrag von RAAML für MBSE besser zu verstehen. In den Beispielen zur RAAML Spezifikation [236] findet sich erweiternd ein Beispiel zur Anwendung von HAZOP in der Automobilindustrie. Weder [49] noch [236] zeigen Anwendung von SEQ-Gates in FTA und berücksichtigen auch keine dynamischen Gates, die ebenso durch die Norm IEC 61025 [203] als Teil von Fehlerbaumanalysen abgedeckt werden. Die beschriebenen Profile, Bibliotheken und Beispiele bieten

allerdings einen ersten Ausgangspunkt und eine einfache Einführung in die aktuellen Ansätze zur engen Modellkopplung in der funktionalen Sicherheit. Es fehlt jedoch noch immer an einer gemeinsamen Modellebene für den Maschinenbau, in der Stakeholder zusammen dargestellt werden können. Die gezeigten Ansätze zur Integration der funktionalen Sicherheit im Maschinenbau zeigen, dass es bisher an einer generalisierten Vorgehensweise wie OMG RAAML scheiterte. Mit dem RAAML Ansatz scheint sich dies nun jedoch zu ändern, doch fehlt es in diesem noch an einer maschinenbauspezifischen Integration auf Basis der Normierung IEC 12100 und ISO 13849.

Vergleich bekannter Ansätze

Einen Überblick der wichtigsten vorgestellten Veröffentlichungen zur losen und engen Modellkopplung findet sich in Tabelle 2.5.

Veröffentlichungen	Funktionale Sicherheit realisiert über	Art der Modellkopplung	Einfache Erweiterbarkeit	Bereits domänenspezifische Umsetzung	Basis OMG RAAML
Joshi et al. '07 [209]	AADL	Lose			
Xiang et al. '11 [210]	RCM	Lose		X	
David et al. '10 [211]	Alta-Rica	Manuell		X	
Hecht et al. '15 [69]	Alta-Rica	Lose			
Yakymets et al. '18 [70]	Alta-Rica	Lose		X	
Helle '12 [75]	SysML	Eng			
Thramboulidis et al. '10 [76]	Profile	Eng			
Mhenni et al. '14 [68]	Profile	Eng			
Biggs et al. '16 [77]	Profile	Eng		X	
Baklouti et al. '19 [212]	Profile	Eng		X	
Biggs et al. '19 [48]	Profile	Eng	X		X
Berres et al. '21 [49]	Profile	Eng	X	X	X

Tabelle 2.5: Zusammenfassung der Veröffentlichungen zur Modellkopplung

Eine lose Kopplung der Modelle über Export in andere Sprachen, wie AADL, RCM oder Alta-Rica scheint hierbei wenig sinnvoll, da beim Übersetzen in anderen Modellierungssprachen Fehlerpotenziale durch fehlerhafte Transformation oder fehlende Rücktransformation entstehen, sowie die Nachverfolgbarkeit darunter leiden kann. Hierbei legt die Auswahl eine enge Modellkopplung über den RAAML-Ansatz nahe. Zum einen besitzt dieser bereits grundlegende domänenspezifische Modelle, die zur Annotation und Weiterentwicklung eines eigenen Ansatzes genutzt werden können und zum anderen lässt sich der Ansatz über grundlegende sicherheitsbezogene Begriffe relativ einfach erweitern. So kann ein praktischer Mehrwert sichergestellt werden. Für die Erweiterung von UML und SysML zur maschinenbauspezifischen Modellierung können daher, wie bereits durch SysML implementiert, Profile und Bibliotheken

modelliert werden. Dadurch kann die Erweiterung von UML so gestalten werden, dass sie wiederverwendbar und wartbar bleibt. Das bedeutet, dass die Erweiterung für verschiedene Projekte nutzbar und an die Bedürfnisse einzelner Stakeholder anpassbar oder ebenso für andere Domänen abstrahierbar ist. Zudem kann beispielsweise relativ einfach über die Formulierung von OCL-Constraints sichergestellt werden, dass Anwender konsistent zur Erweiterung entwickeln, was die Verwendung vereinfacht und das Verständnis verbessert.

Eine enge Modellkopplung in der Maschinenindustrie kann zu einer verbesserten Integration der Stakeholder-spezifischen Modelle beitragen. Wenn Modelle eng miteinander gekoppelt sind, können sie effektiver zusammenarbeiten und das Gesamtergebnis verbessern. Außerdem können Engpässe und Überlastungen vermieden werden, da die Datenflüsse und Korrelationen zwischen den Modellen genauer kontrolliert werden können. Eine weitere wichtige Überlegung bei der Verwendung ist die Effektivität und Effizienz. Durch eine enge Modellkopplung können die Modelle schneller und effizienter arbeiten, da sie einfacher aufeinander zugreifen können. Dies kann besonders wichtig sein, wenn große und komplexe Maschinen entwickelt werden sollen, an denen mehrere Maschinenbauer, Elektroingenieure und Informatiker zusammenarbeiten. Darüber hinaus kann eine enge Modellkopplung zu einer besseren Fehlererkennung und Fehlerbehebung beitragen. Wenn Modelle eng miteinander gekoppelt sind, können Fehler in einem Modell schnell erkannt und behoben werden, bevor sie sich auf andere Modelle auswirken, was die Zuverlässigkeit der Anwendung erhöht. Schließlich kann eine enge Modellkopplung durch höhere Genauigkeit und schnellere Verarbeitung von Anforderungen und Anforderungsänderungen zu einer besseren Leistung und schnelleren Entwicklung der Gesamtmaschine führen. Dies wird besonders wichtig, wenn die Maschine sicherheitskritische Aufgaben ausführt.

2.2.3 Tool-Integration

Werden die Modelle in denselben SysML-Modellen umgesetzt, spielt es zusätzlich eine Rolle, ob die Sicherheitsanalysen direkt in der Modellierungsumgebung oder in externen validierten Werkzeugen ausgeführt werden. Aus diesem Grund unterscheidet die Arbeit im Folgenden zwischen loser und enger Toolkopplung bezüglich der Realisierung der Analysen.

Enge Toolkopplung

Hier soll wieder der RAAML Ansatz als zentrales Beispiel dienen. Er bietet eine enge Toolkopplung, indem Berechnungen in Constraints gespeichert und die Berechnungen für Analysen direkt in SysML durchgeführt werden [49]. Aktuell basieren Analysen auf statischen Berechnungsparametern und einfachen Berechnungsmethoden [49]. Hier kann beispielsweise

die statische FTA in einer SysML-Umgebung und unter Einbeziehung anderer Analysen durchgeführt werden.

Eine lose Modellkopplung mit einer engen Toolkopplung wird kaum genutzt. Es macht wenig Sinn, Berechnungen im Systemmodell zu speichern, obwohl die für die Analyse relevanten Daten in andere spezifischere Modelle übertragen und nicht in SysML berechnet werden.

Lose Toolkopplung

Im Gegensatz zum OMG-Ansatz tendieren viele anwendungsbezogene Ansätze eher zu einer losen Toolkopplung. Zum Beispiel entwickelte Helle [75] bereits 2012 eine Methode zur Generierung von RBD. Er integrierte Sicherheitsartefakte wie Ausfallraten von Komponenten mit sogenannten Tagged Values. Obwohl die Methode auf die Analysen der Zuverlässigkeit beschränkt ist, zeigte sie dennoch, dass eine Sicherheitsanalysemethode automatisch von SysML aus durchgeführt werden kann, indem die erforderlichen Sicherheitsdaten direkt in SysML eingebettet werden, ohne dass eine notwendige Transformation erforderlich ist. Allerdings beinhaltete die Methode von Helle keine weiteren vertiefenden Analysen. Die Autoren von Nordmann et al. [237] und der Nachfolgeveröffentlichung von Munk et al. [238] schlagen eine Methodik vor, bei der CFTs direkt in IBD und SysML-Aktivitätsdiagramme integriert werden, um sowohl statische als auch dynamische Aspekte der Systemarchitektur zu berücksichtigen. Dies ermöglicht eine konsistente und kontinuierliche Sicherheitsanalyse während des gesamten Systementwicklungsprozesses [237, 238]. Die Autoren von [68], [76] und [77] haben bereits die Integration der Risikoanalyse mithilfe von SysML-Profilen behandelt. Thramboulidis und Scholz [76] entwickelten 2010 ein allgemeines Modell, das bis zu 20 verschiedene Daten-Elemente einer Risikoanalyse erfasst. Mhenni et al. [68] nutzten im Jahr 2014 SysML-Profile, um sicherheitsrelevante Informationen für eine Analysemethode zu erfassen, die Sicherheitsartefakte in jeder Phase des Sicherheitslebenszyklus generieren kann. Biggs et al. [77] entwickelten im Jahr 2016 ein SysML-Profil namens SafeML für die Analyse der Zuverlässigkeit. Baklouti et al. [212] aus dem Jahr 2019 ist eine Erweiterung von Mhenni et al. [68]. [212] erfasst zusätzlich Redundanzen und erzeugt DFTs. Die Arbeit von Clegg et al. [44] zeigt, wie eine lose Toolkopplung durch die Verwendung von Skripten zur Datenextraktion und -integration bestehender Sicherheitsanalysen in die SysML-Modelle mit minimaler Unterbrechnung effektiv realisiert werden kann. Sie demonstrieren in dieser und weiteren Veröffentlichungen, wie FTA und FMEA in SysML integriert und weiter analysiert werden können, um eine konsistente und kontinuierliche Sicherheitsanalyse während des gesamten Systementwicklungsprozesses zu gewährleisten [44, 45, 46].

Im Rahmen der losen Toolkopplung konnten folgende Werkzeuge für die Analysemethoden FTA, DFT, Risikograph und SBBM herausgearbeitet werden.

FTA

Es existieren verschiedene Werkzeuge wie CAFTA [239] vom Electric Power Research Institute, EMFTA [240] in Eclipse, oder RAM Commander [241] und Fault Tree Analyzer [242] von ALD, die FTA abbilden. Einen Gesamtüberblick geben die Veröffentlichung Baklouti et al. [71] und Ruijters et al. [216].

DFT

Das Werkzeug RAM Commander [241] unterstützt in seinem Fault Tree Analyser ebenso dynamische Gates sowie exponentielle Verteilungsfunktionen für Ausfallraten und eignet sich aus diesem Standpunkt ebenso für die Berechnung von DFTs. Es handelt sich zwar um ein kommerzielles Werkzeug, doch findet sich ebenso eine kostenlose Testversion. Zusätzlich können DFTCalc [243] und Stormchecker [244] als geeignete Open-Source-Werkzeugumgebungen für DFTs identifiziert werden. Beide unterstützen eine effiziente Fehlerbaummodellierung durch kompakte Darstellungen, praktische Analysen über ein breites Spektrum von Zuverlässigkeitseigenschaften, effiziente Analysen mittels modernster stochastischer Techniken und Continuous-Time-Markov-Chains (CTMC). Darüber hinaus zeigen sie einen flexiblen und erweiterbaren Rahmen, in dem Gates leicht geändert oder hinzugefügt werden können [243, 244]. Es finden sich allerdings auch andere Ansätze wie Durga et al. [245], die eine Monte-Carlo-Simulation zur Berechnung einsetzen. Die Anwendbarkeit und Genauigkeit stochastischen Techniken wurden unter anderem in Veröffentlichungen wie Volk et al. [214] analysiert und verglichen. Hieraus ging in Volk et al. [214] das erweiterte Werkzeug SAFEST hervor, das auf Stormchecker basiert.

Risikograph

Es finden sich ebenso Werkzeuge, die den Risikographen unterstützen. Beispielsweise kann in der WEKA-Software [246] eine gesamte Risikobeurteilung nach ISO 12100 durchgeführt werden, wobei ein Risikograph oder eine Risikomatrix nach ISO 14121-2 und anderen Auslegungen umgesetzt werden kann. Aufgrund des Aufwandes und einfacher Berechnungsfaktoren bietet sich jedoch ein einfacheres Architekturkonzept, als die Anbindung der WEKA-Software an. Es besteht allerdings für diese recht fundamentale Analyse aktuell kein Rahmen, der den Risikographen im Zusammenhang mit der Risikobeurteilung adäquat integriert.

SBBM

Die SBBM wird wie bisher genannte Analysen ebenso von Analysewerkzeugen unterstützt, dass Sicherheitsingenieuren die Entwicklung und Überprüfung von Sicherheitsfunktionen auf SRP/CS erleichtern soll. An erster Stelle findet sich das Open-Source Analysewerkzeug SISTEMA [222]. Es umfasst die beschriebenen Anforderungen und ebenso verschiedene Hilfestellungen zur Bewertung der Sicherheit im Rahmen der ISO 13849. Das vom Institut für Arbeitssicherheit herausgegebene Tool ermöglicht es Benutzern, die Struktur auf der Basis der vorgesehenen Architekturen nachzubilden und bietet eine automatisierte Berechnung der Wahrscheinlichkeitswerte auf verschiedenen Detailebenen einschließlich des erreichten PLs [222]. Hierbei können Blöcke eines Kanals zusätzlich in Elemente zerlegt werden [222], was einen größeren Gestaltungsfreiraum gibt, allerdings von der ISO 13849 nicht direkt vorgesehen war. Ein Element verhält sich dabei gleich eines Blocks. Viele Hersteller stellen ebenso Bibliotheken mit validierten Werten für ihre Bauteile bereit, die über einen XML- oder SISTEMA-Bibliothek-Import in die Projekte integriert werden können. Zusätzlich können die Ergebnisse in einem druckbaren Report zusammengefasst werden. Eine kostenpflichtige Alternative stellt der PAScal Safety Calculator [247] von PILZ dar.

2.2.4 Diskussion

Im vorliegenden Abschnitt wurde das Forschungsziel des Typs Beobachtung aus dem Bereich der Integration maschinenbauspezifischer Analysemethoden vorgestellt. Im Rahmen dieser Forschungsaufgabe wurde ein Überblick über gängige Sicherheitsanalysemethoden und ihre Anwendungsbereiche im Maschinenbau gegeben. Es wurden Arbeitsweisen beschrieben und Analysewerkzeuge, im Speziellen in Bezug auf MBSE und Abbildbarkeit der Analysen, aufgezeigt, um Informationen zum Gesamtprozess in ein MBSE-Framework zu sammeln. Dabei wurde mit RAAML ein auf grundlegenden Elementen und Annotationen zum Systemmodell aufgebautes Framework identifiziert, das jedoch für dynamische und domänenspezifische Analysen erweitert werden muss, da bereits integrierte Analysen für die Maschinenindustrie nicht ausreichend sind. Eine Möglichkeit sollen DFTs als Erweiterung zur FTA bieten, wodurch eine über den Maschinenbau hinaus anwendbare dynamische Analysemethode integriert werden kann. Für domänenspezifische Analysemethoden wie die Risikographmethode und die SBBM existiert ebenso noch keine Einbindung in ein MBSE-Framework. Mit dem Fokus auf der Risikographmethode zur Risikoeinschätzung und der SBBM zum Nachweis ausreichender Risikominderung können zwei wesentliche Analysen in RAAML integriert werden. Somit können zum einen mit DFTs Ausfallwahrscheinlichkeiten für Komplett- und Teilsysteme berechnet werden, was sich ebenso auf Sicherheitsfunktionen abbilden lässt, und zum anderen bietet

SBBM die Ausfallraten einzelner Sicherheitsfunktionen, wodurch die Ergebnisse beider Analysen für den Maschinenbau in Bezug gesetzt werden können. Es kann angenommen werden, dass keine weiteren Standards erforderlich sind. Jedoch ist die Implementierung bestehender Integrationsmuster und -formate zu berücksichtigen, um Begriffe an den bisherigen Ansatz von RAAML anzupassen, indem Synonyme, Äquivalenzen, Verallgemeinerung oder Spezialisierung zu bilden sind. Dies führt zu Herausforderungen bei der Integration der Analysen, die im Modellierungsteil dieser Arbeit behandelt werden. Jede Analyse muss in die zentrale Struktur eingebunden werden, um den Artefakt-Austausch zwischen den Analysen weiterhin zu garantieren. Die Modellierung der Analysen wird von der Forschungsaufgabe FA2-TB übernommen und durch die Implementierung FA2-I sowie die entsprechende Evaluation FA2-E bestätigt.

Es wurden vier in Tabelle 2.6 zusammengefasste offene Herausforderungen (OH) identifiziert.

OH2.1: Eine Erweiterung des RAAML Ansatzes für domänenspezifische Merkmale, Begriffe und Sicherheitsanalysemethoden wird benötigt.
OH2.2: Es sollte möglich sein, eine Risikoeinschätzung mit einem auf SysML abgebildeten Risikographen durchzuführen.
OH2.3: Es sollte möglich sein, eine Risikominderung im Maschinenbau mit einem auf SysML abgebildeten DFT zu überprüfen.
OH2.4: Es sollte möglich sein, eine Risikominderung im Maschinenbau mit einer auf SysML abgebildeten SBBM zu überprüfen.

Tabelle 2.6: Verbleibende Herausforderungen bzgl. Integration der Analysen

2.3 MBSE-Prozesse für KMUs

Neben dem generellen Aufbau einer MBSE-Basis für den Maschinenbau und der Integration von Sicherheitsanalysemethoden für die Maschinensicherheit ist ein weiterer wesentlicher Faktor die Anwendbarkeit und Integration in bestehende Prozesse. Durch den geplanten Einsatz in einem KMU muss ein einfach anwendbarer Prozess herausgearbeitet werden. In diesem Abschnitt werden daher verschiedene Literaturergebnisse und Usability-Studien präsentiert, um die Aspekte der Problembeschreibung PB3 zu behandeln. Unterabschnitt 2.3.1 bietet einen Überblick über die Hindernisse zur Anwendbarkeit von MBSE in KMUs und Unterabschnitt 2.3.2 geht auf die Bewertung der Eignung eines MBSE-Frameworks und gängige Evaluationsmethoden ein sowie beschreibt einen Prozess zur qualitativen Evaluation.

2.3.1 Identifikation der Hindernisse zur Anwendbarkeit von MBSE in KMUs

MBSE hat sich in den letzten Jahren als eine bewährte Methode zur Entwicklung komplexer Systeme etabliert [58]. Vor allem in größeren Unternehmen wird MBSE aufgrund seiner zahl-

reichen Vorteile wie verbesserte Zusammenarbeit, Fehlerreduktion und Zeit- und Kostenersparnis zunehmend eingesetzt. In KMUs stellt sich jedoch eine andere Herausforderung dar. KMUs haben in der Regel begrenzte Ressourcen und Expertise, um komplexe Technologien wie MBSE zu implementieren und zu nutzen [101, 103, 104, 105, 106, 110], und sind durch eine nicht so breite Marktabdeckung anfälliger für Fehlinvestitionen [107]. Trotz dieser Herausforderungen müssen KMUs in der Lage sein, wettbewerbsfähige Produkte zu entwickeln, die den Marktanforderungen entsprechen. Es ist daher entscheidend, die spezifischen Hindernisse und Probleme, die bei der Anwendung von MBSE in KMUs entstehen, zu identifizieren und zu verstehen.

Die grundsätzliche Herausforderung für Unternehmen besteht darin, den optimalen MBSE-Ansatz zu identifizieren, welcher den spezifischen Anforderungen des Unternehmens entspricht. Hierbei müssen laut Vogel-Heuser [36] verschiedene Faktoren wie beispielsweise Marktanforderungen, Kundenbeziehung, Workflow-Anforderungen, Qualifikation des Personals und Budget berücksichtigt werden. Doch oft spielt auch das Vertrauen in den Mehrwert der Innovation eine entscheidende Rolle. Tests zum Einsatz von MBSE-Ansätzen scheitern oft an höher priorisierten „realen“ Projekten [36], auf Basis fehlender personeller Ressourcen. Häufig führen eine schlechte Benutzerfreundlichkeit der im MBSE-Framework eingesetzten Werkzeuge, fehlende Funktionen oder fehlende modulare und flexible Schulungseinheiten zur Ablehnung eines Ansatzes [248]. Zusätzlich wird das Risiko, zu viel Zeit erfahrener Ingenieure für die Bewertung einer „wahrscheinlich ungeeigneten“ Notation und eines Werkzeugs aufzuwenden, oft als zu hoch eingeschätzt, um eine systematische und umfassende Bewertung zu ermöglichen [36]. Dies bedeutet für ein geplantes Vorhaben, dass es lediglich einen beschränkten Lernaufwand „kosten“ darf und möglichst einfach anwendbar sein sollte, um Entwickler nicht vom Tagesgeschäft abzuhalten und Stress zu reduzieren.

Zur Steigerung der Effizienz und Qualität im Entwicklungsprozess eines Maschinenbauunternehmens sind bereits Studien wie Vogel-Heuser [36] durchgeführt worden. Hierbei konnte ein quantitativer Nutzen für modellbasierte Ansätze unter Verwendung von Notationen wie UML und SysML nachgewiesen werden [36]. Dennoch wird die Einführung in einem Industrieunternehmen als schwierig aufgefasst, da neben Schulungen und Regeln zur Anwendung auch geeignete Workflows zu bestehenden Werkzeugen existieren müssen [36], um finanzielle und zeitliche Ressourcen zum Erwerb und zur Einarbeitung zu schonen [103]. Eine Möglichkeit zur Begrenzung des finanziellen Aufwands findet sich in der Nutzung Lizenz-freier Werkzeuge, wie Papyrus.

Kießling et al. [110] entwickelten verschiedene Konzepte, die sich speziell an die Herausforderungen von KMUs im Sondermaschinenbau richten. Sie entwickelten dabei einen Engineering-Ansatz, der sich an einen Informationsaustausch in jedem Prozessschritt der mechanischen, elektronischen und softwaretechnischen Entwicklung orientiert und versucht bestehende Entwicklungswerkzeuge beizubehalten, da sich KMUs den Austausch ihrer Werkzeuge und Training zur Einarbeitung in neue Werkzeuge finanziell oft nicht leisten können [110]. Hierbei sind standardisierte MBSE-Prozesse und -Werkzeuge oft sehr komplex und schwer in einem KMU-Umfeld anzuwenden [58], da oft das notwendige Knowhow fehlt. Chapurlat et al. [58] schlägt einen angepassten Ansatz für die Einführung von MBSE in KMUs vor, der die spezifischen Bedürfnisse und Einschränkungen dieser Unternehmen berücksichtigt. Sie identifizieren die Prinzipien eines erfolgreichen Einsatzes von MBSE in KMUs wie folgt:

- Agilität: Das Unternehmen muss die Struktur besitzen, um ein modellbasiertes Entwicklungsframework in kollaborativen multidisziplinären Projekten umzusetzen.
- Vertrauen: Die Stakeholder müssen Innovationsbereitschaft besitzen und den Mehrwert der Darstellung in einem MBSE-Framework verstehen.
- Zusammenarbeit: Ein MBSE-Framework sollte die Möglichkeit besitzen, Stakeholdern einen einfachen Zugriff auf Artefakte hinsichtlich Qualität, Kosten, Dauerhaftigkeit, Stabilität und Effizienz zu geben, und eine einfache Kommunikationsbasis bieten.
- Innovation: Ein MBSE-Framework sollte den Grad der Innovation verbessern, um die Entwicklung zukünftiger Systeme und Zufriedenheit der Stakeholder zu unterstützen.

Darüber hinaus haben die Organisationsstruktur und -kultur einen erheblichen Einfluss auf die Einführung von MBSE [58, 249]. Durch die Unterschiede zu traditionellen Standards entsteht eine Herausforderung, einen maßgeschneiderten Ansatz zu entwerfen, der die Einführung erleichtert und beschleunigt [58]. Faktoren wie Flexibilität und Vernetzung innerhalb einer Organisation korrelieren stark mit einer erfolgreichen MBSE-Implementierung [249].

Um eine methodisch fundierte Bewertung zur Eignung eines MBSE-Frameworks für KMUs in der Maschinenindustrie durchzuführen, wird ein spezifischer Ansatz und eine abgestimmte Evaluationsmethodik benötigt, welche im folgenden Abschnitt näher erläutert werden.

2.3.2 Bewertung zur Eignung eines MBSE-Frameworks

Die Wahl eines methodischen Ansatzes hängt von den spezifischen Zielen und Möglichkeiten der Evaluation ab. Als Grundlage kann der 11. Teil der Norm ISO 9241 [250] Ergonomie der Mensch-System-Interaktion zur Gebrauchstauglichkeit dienen. Diese Norm enthält Studien zur

Nutzungseffizienz und zur Zufriedenheit der Nutzer, die die Messung der Gebrauchstauglichkeit von Produkten in ihrem Nutzungskontext vorschlagen. Hiernach liegen die Anforderungen hauptsächlich in den Aspekten Usability anhand von Effektivität (Vollständigkeit und Korrektheit der Lösungen), Effizienz (Verhältnis von Effektivität zu Zeitaufwand) und Benutzerzufriedenheit (Grad, zu dem Benutzerbedürfnisse erfüllt werden) [250]. Die generelle Methodik lässt sich unter Usability-Evaluierung [251] zusammenfassen.

Inwieweit sich ein MBSE-Ansatz und ein entsprechend entwickeltes Framework für den Einsatz in einem KMU eignen, lässt sich durch eine empirische Evaluation beurteilen [252]. Sie zielt darauf an, Daten durch direkte Beobachtung oder Experimente zu sammeln, um konkrete und verallgemeinerbare Erkenntnisse zu gewinnen [253, 254, 255]. Hierunter finden sich quantitative und qualitative Evaluationsmethoden. Eine quantitative Evaluationsmethodik fokussiert sich auf die Messung und Analyse numerisch erfassbarer Kenngrößen [255, 256]. Ziel ist es, über Umfragen, Fragebögen und Experimente sowie deren statistische Analyse verallgemeinerbare objektive Ergebnisse zu erzielen [255, 256], die oft einen breiteren Kontext oder größere Stichproben betreffen [254, 255]. Dies lässt sich unter dem Begriff der experimentellen Evaluation [257] zusammenfassen. Eine qualitative Evaluationsmethodik hingegen zielt darauf ab, tiefergehende Einsichten und Verständnisse über komplexe Phänomene zu gewinnen [258, 259], oft durch detaillierte Beobachtungen, Interviews und Fokusgruppen im Rahmen von Fallstudien [259]. Im Case Studie Research [260] werden zunehmend subjektive Erfahrungen und Perspektiven der beteiligten Personen berücksichtigt, um ein umfassenderes Bild der untersuchten Thematik zu erhalten.

Das gängige Verfahren zur Überprüfung der Eignung verschiedener Modellierungsmethoden in der Automatisierungstechnik besteht laut Siau et al. [261] darin, Stakeholder aus den entsprechenden Anwendungsdomänen zu konsultieren. Qualitative Methoden sind hierbei besonders nützlich, um die Perspektiven und Erfahrungen der Nutzer zu verstehen [259]. Alternativ zur Konsultation von Stakeholdern können auch Endanwender befragt werden, um praktische Erfahrungen zu sammeln und eine Fallstudie zu erstellen [262]. Um wertvolle Einsichten in die Anwendbarkeit und Effizienz von MBSE zu gewinnen und aufgrund begrenzter Kapazitäten in KMUs, bietet sich nach der Typologie von Evaluationsmodellen aus Beywl [263] für diese Arbeit eine qualitative Stakeholder-interessengesteuerte Evaluation an. Sie zeichnet sich durch die gleichberechtigte Einbeziehung und Berücksichtigung der Interessen aller relevanten Stakeholder aus, wodurch die erhobenen Daten maximal relevant und nützlich für die kontinuierliche Verbesserung der Systementwicklungspraktiken sein sollen. Sie wird in der Kategorie der

wertepriorisierenden Modelle eingeordnet, da sie explizit soziale Werte in den Evaluationsprozess integriert [263]. Um die im ersten Paragraphen dieses Abschnittes angesprochenen quantitativen Anforderungen zu erfüllen, können, soweit möglich, zusätzliche quantitative Untersuchungen durchgeführt werden, die auf die Messung und Analyse numerisch erfassbarer Daten abzielen. Beispielsweise kann der Implementierungsaufwand für die Erstellung der Software in Source Lines of Code (SLOC) der verwendeten Programmiersprache gemessen werden. Der Ressourcenverbrauch hingegen bezieht sich auf das Programm während der Ausführung und wird in Speicherverbrauch des Arbeitsspeichers bzw. des Hintergrundspeichers und des CPU-Verbrauchs gemessen. Der CPU-Verbrauch selbst misst die Rechenzeit abhängig von der CPU-Geschwindigkeit oder die Anzahl ausgeführter Maschineninstruktionen. Weiterhin können die Genauigkeit von Berechnungsergebnissen der Analysemethoden durch Vergleichsanalyse bestimmt und die Konsistenz sowie Abdeckung von Modellrepräsentationen durch logische Beweisführung und Fallunterscheidungen beurteilt werden [264]. Diese sind in dieser Arbeit allerdings von untergeordneter Bedeutung gegenüber der qualitativen Bewertung.

Methodisch betrachtet können Konsultationen, im Konkreten Interviews, von Experten zu Fallstudien entweder individuell in Form von Interviews oder in interaktiven Gruppen innerhalb von Workshops erfolgen [36]. Beide Ansätze erheben subjektive Einschätzungen der Befragten, die ihre Einstellungen, Meinungen und Kenntnisse aufnehmen. Fokusgruppen und Fallstudien arbeiten mit einer begrenzten Anzahl von Experten [261, 262]. Die gewonnenen Erkenntnisse können als Inspiration für weitere, detailliertere und tiefere ebenso quantitative Studien mit expliziteren Hypothesen dienen. Fokusgruppen haben den Vorteil einer natürlicheren Atmosphäre, die im Idealfall zu einer höheren Gesprächsbereitschaft und Offenheit der Teilnehmenden führt [265, 266]. Einzelinterviews hingegen sind einfacher durchzuführen, da die Personen zu unterschiedlichen Zeiten und an unterschiedlichen Orten befragt werden können [36]. Auch bei Fallstudien ist die Teilnehmergruppe oft sehr klein, wie beispielsweise in [267] und [268], weshalb beobachtete Aspekte nur als Anhaltspunkte dienen können [36].

Diese Art von Experimenten erfordert, dass die Testpersonen mit der neuen Notation oder dem Paradigma vertraut gemacht werden, was sehr zeitaufwändig ist [36]. Eine große Anzahl von Experten, mindestens 15-20 pro verglichener Notation, ist erforderlich, um aussagekräftige, objektive Ergebnisse zu erzielen [36]. Zudem ist es schwierig, die Vergleichbarkeit von Ergebnissen verschiedener Experten mit unterschiedlichem Vorwissen zu gewährleisten [36]. Eine Möglichkeit, um den Mehrwert der Evaluation einer Notation trotz einer geringen Expertenzahl hoch zu halten, stellen Evaluationsinstrumente wie Fragenkataloge dar [259, 269], zu denen die

Aussagen der Experten möglichst konkret für zukünftige Fallstudien und Interviews festgehalten werden.

Nach Vorgaben wie [259, 263, 269, 270], aber auch Studien wie [36, 267, 268] lässt sich der Prozess, wie er in Forschung und Praxis angewendet wird, durch die Prozessschritte Evaluationsgegenstand festlegen, Fragenkataloge entwickeln, Teilnehmer auswählen, Interviews durchführen sowie Ergebnisse analysieren und interpretieren identifizieren. Blöbaum et al. [270] legt die Schritte zur qualitativen Evaluation durch Interviews über Fragenkataloge genauer fest. Hieraus gehen folgende, den Schritten untergeordnete Erläuterungen hervor:

1. **Evaluationsgegenstand und -ziele festlegen:**
   - Bestimmung des Fokus der Evaluation sowie des übergeordneten Ziels
   - Dokumentation der Ziele und des Evaluationsumfangs

2. **Fragenkataloge entwickeln:**
   - Entwicklung spezifischer Fragen, die auf die festgelegten Evaluationsziele abzielen und alle relevanten Aspekte abdecken
   - Validierung des Fragenkatalogs

3. **Teilnehmer auswählen:**
   - Auswahl von geeigneten Interviewpartnern basierend auf ihrer Rolle und Expertise (z.B. Erfahrung, Fachwissen, Rollen innerhalb des Projekts)
   - Kontaktaufnahme und Einholung der Teilnahmebestätigungen

4. **Durchführung der Interviews:**
   - Terminierung und Organisation der Interviews zur Aufnahme der Ausgangssituation (Erwartungshaltung) und der Interviews zur Aufnahme der Endsituation (Zufriedenheit)
   - Durchführung der Interviews gemäß des entwickelten Leitfadens, entweder als Einzelinterviews oder in Gruppen
   - Sicherstellung einer neutralen und offenen Gesprächsatmosphäre zur Förderung von Offenheit und Ehrlichkeit der Teilnehmer

5. **Transkription, Analyse und Interpretation:**
   - Transkription der Interviewaufzeichnungen zur Erstellung einer schriftlichen Basis für die Analyse

- Qualitative Datenanalyse zur Identifikation von Mustern, Themen und relevanten Erkenntnissen (z.B. durch Kodierung und Kategorisierung)
- Interpretation der Ergebnisse im Kontext der festgelegten Evaluationsziele
- Empfehlungen basierend auf den gewonnenen Erkenntnissen

Um einen Vorher-Nachher-Vergleich zu ermöglichen, wird der Prozess wie in den Studien Abdinnour et al. [271] und Savelberg et al. [272] umgesetzt. Für den Schritt 2 sind zunächst Fragenkataloge zur Erwartungshaltung und nachfolgend Fragenkataloge zur Zufriedenheit zu entwickeln. Schritt 4 und Schritt 5 sind entsprechend doppelt zu durchlaufen; einmal anfangs zur Aufnahme und Analyse der Ausgangssituation, um die Forschungen auf die Bedürfnisse abzustimmen, und einmal nach Umsetzung der Fallstudie, um die tatsächliche Zufriedenheit darstellen zu können. Diese Schritte ermöglichen somit eine gründliche und strukturierte qualitative Evaluation, die auf die spezifischen Anforderungen zur Bewertung der Eignung eines MBSE-Frameworks abgestimmt ist.

2.3.3 Diskussion

Im vorliegenden Abschnitt wurde das Forschungsziel des Typs Beobachtung aus dem Bereich MBSE-Prozesse für KMUs vorgestellt. Dazu wurde für die Forschungsaufgabe eine fundierte Untersuchung verschiedener Studien durchgeführt, die sich mit den Herausforderungen für KMUs befassen. Diese Analyse dient als solide Basis und leitet die weiteren Kapitel dieser Arbeit ein.

Es wurden verschiedenen Hindernisse zur Anwendung von MBSE in KMUs diskutiert, ein Überblick über die Herausforderungen im Maschinenbau gegeben und Bewertungsverfahren angesprochen, die wesentlich zur Beurteilung eines anwendbaren Frameworks beitragen können. Die gesammelten Informationen bieten einen weiteren Schritt zur Unterstützung eines Gesamtprozesses und zur Integration der Stakeholder in ein KMU-freundliches Framework.

Zunächst wird ein möglichst vollständiges MBSE-Framework und benutzerfreundliches Tool mit kurzer Einarbeitungszeit benötigt, dass durch eine vereinfachte Anwendbarkeit, beispielsweise über Profile, das Vertrauen in die gemeinsame Datenbasis stärkt, aber dennoch die einzelnen Sichten unterstützt. Die Modelle dürfen sich dabei für einen effektiven Arbeitsprozess nicht zu sehr von den Entwicklungsmodellen der Stakeholder unterscheiden. Hierbei sind ressourcenschonende Prozesse zum Import und Export von Modellen über Transformatoren aus der MBSE-Umgebung in externe Entwicklerwerkzeuge darzustellen, um etablierte teuer erworbene zertifizierte Werkzeuge weiterzuverwenden und Effizienz sowie Agilität zu wahren.

Hieraus entstehen Herausforderungen, die bei der Entwicklung eines MBSE-Prozesses für KMUs zu beachten sind und im Modellierungsteil dieser Arbeit ausführlich behandelt werden. Die Modellierung einer Methode zur Anwendbarkeit von MBSE in KMU wird von der Forschungsaufgabe FA3-TB übernommen und durch die Implementierung FA3-I sowie die entsprechende Evaluation FA3-E beurteilt.

In diesem Abschnitt wurde die in Tabelle 2.7 dargestellte offene Herausforderung identifiziert.

OH3.1: Modellierung eines Architekturkonzeptes für KMUs in der Maschinenindustrie unter Berücksichtigung der Aspekte Agilität, Vertrauen, Zusammenarbeit und Effizienz

Tabelle 2.7: Verbleibende Herausforderungen bzgl. MBSE-Prozesse für KMUs

2.4 Zusammenfassung & verbleibende Herausforderungen

Dieses Kapitel stellte die anfänglichen Forschungsaufgaben vom Typ Beobachtung vor und beschreibt den aktuellen Stand der Wissenschaft und Technologie. Darüber hinaus wurden verwandte Arbeiten zu dieser Arbeit skizziert und diskutiert, sodass der Fokus dieser Arbeit definiert und die Einzigartigkeit dieser Forschung veranschaulicht wird. Es wurde dargelegt, dass im Bereich von MBSE sicherheitskritischer Anwendungen im Maschinenbau mittelständischer Unternehmen durchaus Verbesserungsbedarf für ein anwendbares Konzept besteht, das die verschiedenen Sichtweisen der Stakeholder, die Modelldarstellung und den Austausch sowie die Durchführung von Analysen unterstützt. Die Forschungsaufgaben vom Typ Beobachtung aus Abschnitt 1.3 Tabelle 1.1 dienen als Grundlage für weitere Forschung, Modellierung, Implementierung oder Experimente, d.h. für die entsprechenden Forschungsaufgaben der Typen Theoriebildung, Implementierung und Experiment. Die Titel dieser Forschungsaufgaben können nun präziser definiert werden. Dies führt zu einem vollständigen Satz präzisierter Forschungsaufgaben und den entsprechenden Forschungszielen. Tabelle 2.8 zeigt die Forschungsaufgaben. Aufgaben, die bereits in diesem Kapitel behandelt wurden, sind grau hinterlegt. Falls Titel präzisiert wurden, wird dies über Klammern angezeigt. Die Forschungsaufgaben, die weiß hinterlegt sind, stellen direkt die verbleibenden Herausforderungen nach diesem Kapitel dar und werden im verbleibenden Teil dieser Arbeit behandelt. Darüber hinaus enthält Tabelle 2.8 die Zuordnung dieser Forschungsaufgaben zu den identifizierten offenen Herausforderungen.

Alle relevanten Ergebnisse, die auf der Durchführung von Forschungsaufgaben vom Typ Beobachtung basieren, wurden in diesem Kapitel vorgestellt.

Forschungsaufgaben (FA)		
FA	**Beschreibung**	**Abschnitt**
Maschinenbauspezifische Modellierung		
FA1-B	Recherche einer gemeinsamen Modellebene und Architektur für MBSE im Maschinenbau, im Speziellen für Prüfmaschinen, und Grundlage zur Modularisierung	2.1
FA1-TB OH1.1 OH2.1 OH2.2	Design eines modularen Informationsmodells und einer Architektur für MBSE im Maschinenbau, im Speziellen für Prüfmaschinen (um einerseits Stakeholder-spezifische Baugruppen und Elemente modellieren und andererseits domänenspezifische Analysen, wie Risikographen durchführen zu können)	
FA1-I	Umsetzung einer modularisierten gemeinsamen Modellebene und Architektur für MBSE im Maschinenbau, im Speziellen für Prüfmaschinen	
FA1-E	Evaluation einer modularisierten gemeinsamen Modellebene und Architektur für MBSE im Maschinenbau, im Speziellen für Prüfmaschinen	
Integration der Analysen des Maschinenbaus		
FA2-B	Identifikation maschinenbauspezifische Analysemethoden und passender Frameworks	2.2
FA2-TB OH2.3 OH2.4	Modellierung zur Integration der maschinenbauspezifische Analysemethoden dynamische Fehlerbaumanalyse (DFT) und sicherheitsbezogene Blockdiagrammmethode (SBBM) zum Nachweise der Risikominderung	
FA2-I	Entwurf und Implementierung zur Integration maschinenbauspezifische Analysemethoden	
FA2-E	Evaluation typischer Analysemethoden	
MBSE-Prozesse für KMUs		
FA3-B	Identifikation der Hindernisse zur Anwendbarkeit von MBSE in KMUs	2.3
FA3-TB OH3.1	Vorgehen zur Anwendbarkeit von MBSE in KMUs unter Berücksichtigung der Aspekte Agilität, Vertrauen, Zusammenarbeit und Effizienz	
FA3-I	Implementierung eines Konzeptes für KMUs in der Maschinenindustrie	
FA3-E	Evaluation mit KMU	

Tabelle 2.8: Forschungsaufgaben nach Beobachtung

Es wurde dargelegt, dass bestehende Lösungen auf verschiedenen Standards beruhen, doch dass für den Einsatz im Maschinenbau eine Integration der Stakeholder und Analysen fehlt, um den Anforderungen der gesetzlichen Vorgaben zu entsprechen. Daher wird im nächsten Kapitel die Theoriebildung durch Modellierung und Design vorgestellt, die darauf abzielt, die in diesem Abschnitt dargestellten verbleibenden offenen Herausforderungen zu lösen. In dieser Modellierung wird die Methodik aus dem aktuellen Stand der Technik aufgegriffen, um neue Profile und Artefakte für ein gemeinsames Informationsmodell einzuführen und die gemeinsame Anwendbarkeit zu erleichtern.

3 Modellierung

In diesem Kapitel werden die Forschungsziele vom Typ Theoriebildung nach dem Ansatz von Nunamaker [112] aus Abschnitt 1.2.6 beschrieben. Die Struktur des Kapitels folgt der logischen Abfolge der Forschungsaufgaben. Nach einer kurzen Einführung zur konzeptuellen Modellierung in Abschnitt 3.1 beschreibt die Modellierung im Bereich maschinenbauspezifische Integration in Abschnitt 3.2, welche grundlegenden Aspekte bereitgestellt werden müssen, um ein MBSE-Framework für den Maschinenbau aufzubauen, und entwirft eine gemeinsame Modellebene für den Maschinenbau, im Speziellen für Prüfmaschinen und dem Einsatz von Risikographen zur Risikobeurteilung. In Abschnitt 3.3 wird der Bereich der Sicherheitsanalysemethoden näher behandelt und es werden die Modelle für dynamische Fehlerbaumanalyse (DFTs) und sicherheitsbezogene Blockdiagrammmethode (SBBM) eingeführt. Der vierte Abschnitt 3.4 beschäftigt sich mit der Modellierung der Anwendbarkeit eines MBSE-Frameworks in KMUs, im Speziellen in ProNES. Abschließend bietet dieses Kapitel eine Zusammenfassung in Abschnitt 3.5, in welcher die Ergebnisse des Theoriebildungsabschnitts zusammengefasst werden.

3.1 Einführung

In diesem Abschnitt folgt das konzeptionelle Design und die Modellierung dem Ansatz des User Centered System Design von Norman & Draper [117], der darauf abzielt, die Modellierung aus Sicht des Benutzers oder eines Anwendungsfalls zu starten. Zur Strukturierung und Beschreibung dieses konzeptionellen Designs werden der Rational Unified Process (RUP) [273] einschließlich Unified Modeling Language (UML) [25] und zur Erweiterung Systems Modeling Language (SysML) [26] verwendet. RUP definiert sechs Phasen: Business Modeling, Requirements, Analysis & Design, Implementation, Test und Deployment. Die Ergebnisse aus den Phasen Business Modeling, Requirements, Analysis und Design sind Teil dieses Kapitels und implementieren die Methodologie für die Nunamaker-Forschungsaufgaben zur Theoriebildung. Die Implementierungsphase des RUP entspricht den Implementierungsforschungsaufgaben von Nunamaker und wird unter Kapitel 4 beschrieben. Bei der Testphase des RUP handelt es sich um die Forschungsaufgaben vom Typ Experiment im Vergleich zu Nunamaker. Diese wird in Kapitel 5 behandelt. Die Deployment-Phase des RUP ist nicht besonders auf den Nunamaker-Ansatz ausgerichtet, wird jedoch in den einzelnen Kapiteln berücksichtigt. Somit wird

Model-Based Systems Engineering sicherheitskritischer Anwendungen im Maschinenbau mittelständischer Unternehmen © 2024 by Nick Berezowski is licensed under CC BY 4.0

jedes konzeptionelle Design und jede Modellierung, die von der Benutzerperspektive ausgeht, auf der Grundlage der Phasen des RUP entwickelt und unter Verwendung von UML und SysML formalisiert. An einigen Stellen dieses Kapitels werden auch Blockdiagramme zu Illustrationszwecken [274] eingesetzt. Diese könnten auch direkt in Form von UML-Diagrammen formuliert werden, aber einfache Blockdiagramme helfen, sich auf bestimmte Aspekte der Modellierung zu konzentrieren. Eine vollständige SysML-Modellierung wird jedoch immer bis zum Ende jedes Unterabschnitts vorgestellt. Teile der Modellierung wurden auf einer Konferenz veröffentlicht. Eine vollständige Liste aller Veröffentlichungen während der Themenbearbeitung findet sich in Anhang A.1.

3.2 Maschinenbauspezifische Modellierung

In diesem Abschnitt wird das Fundament für die nachfolgenden Modellierungen in Form eines MBSE eingeführt, das die verschiedenen Stakeholder des Maschinenbaus und deren Sichten unterstützt. Unterabschnitt 3.2.1 bildet die Grundlage für eine gemeinsame Sprache und die zu entwickelnde Struktur der Modellierung. Das Architekturkonzept für Forschungsaufgabe FA1-TB wird darauf folgend in Unterabschnitt 3.2.2 behandelt. Die Gründe dafür liegen in den zugrundeliegenden Methoden. Nach Norman & Draper wird die Modellierung aus der Perspektive der Benutzer initiiert, die insbesondere durch Anwendungsfälle definiert wird und somit der logische Ausgangspunkt dieses Abschnitts ist. Auch die Phasen des Geschäftsmodellierungs- und Anforderungsprozesses des RUP sind auf Anwendungsfälle aufgebaut. In Unterabschnitt 3.2.3 bis 3.2.11 folgt die Konzeptionierung und der Entwurf einer gemeinsamen Modellebene für den Maschinenbau, im speziellen Prüfmaschinen, um Stakeholder-spezifische Teilsysteme und Elemente modellieren und domänenspezifische Merkmale und Analysen durchführen zu können. Unterabschnitt 3.2.12 fasst die Ergebnisse kurz zusammen.

3.2.1 Gemeinsame Sprache und Struktur für die Modellierung

In Bezug zur Umsetzung eines verwendbaren MBSE-Frameworks legt die Recherche aus Abschnitt 2.1.1 nahe, Metamodelle zu nutzen, die den bestehenden UML-Ansatz erweitern und auf die domänenspezifischen Notwendigkeiten anpassen. Der Einsatz von Meta Object Facility (MOF) und UML bieten hierbei wesentliche Vorteile. Zum einen handelt es sich um von der OMG standardisierte Sprachen, die bereits in vielen Modellierungswerkzeugen umgesetzt wurden. Dies bedeutet, dass sie bereits einer großen Gemeinschaft von Entwicklern zur Verfügung stehen und dass es viele Ressourcen gibt, die bei der Verwendung der Werkzeuge helfen können. MOF wurde speziell für die Modellierung von Sprachen entwickelt, weshalb sie bereits

eine Vielzahl von Sprachelementen und Konstrukten besitzt, die bei der Erweiterung genutzt werden können. Gleichzeitig bietet die 4-Schichten-Architektur hohe Abstraktionsebenen, die es ermöglichen, Erweiterungen adäquat zu beschreiben. Dies kann die Modellierung für den Anwender einfacher und schneller machen, da man sich auf die Konzepte und Abstraktionen der Stakeholder konzentrieren kann, anstatt auf Details. Der Ansatz über UML kann sicherstellen, dass die Modellierung konsistent und standardisiert ist. Dies erleichtert es anderen Entwicklern, die Modelle zu verstehen und zu verwenden. Zuletzt bietet der Austausch und die Verarbeitung von Modellinformationen über XMI die Möglichkeit, automatisch Daten und Artefakte zu exportieren und zu importieren oder Code aus den Modellen zu generieren, was Zeit und Ressourcen sparen und die Fehleranfälligkeit reduzieren kann.

Um ein Framework zu entwickeln, dass keine unnötigen neuen Artefakte und Erweiterungen einführt und das vergleichbar mit bestehenden Systementwicklungen ist, soll SysML Einsatz finden. Hierbei sollen so viele Konzepte wie möglich von SysML wiederverwendet und nur solche hinzugefügt werden, die in SysML fehlen, um eine maschinenbauspezifische Stakeholder-abhängige Modellierung von Systemen zu ermöglichen. Dadurch wird Doppelarbeit vermieden und die Erweiterung kann Einsatz finden, ohne neue Werkzeugfunktionen zu benötigen, vorausgesetzt, SysML wird unterstützt. Dies führt dazu, dass eine relativ kleine Erweiterung auf der Grundlage von UML und SysML ausreicht, um Kernkonzepte abzudecken.

Die in diesem Kapitel präsentierten Modellierungsergebnisse basieren auf einer systematischen Analyse von Veröffentlichungen und wurden durch die Erkenntnisse und Perspektiven des Autors erweitert und kombiniert. Obwohl die Modellierung auf diesen Quellen aufbaut, ist sie weder zwingend noch endgültig, da alternative Perspektiven und Informationen zu unterschiedlichen Modellierungsansätzen führen können. Insgesamt stützt sich die Modellierung auf ein umfassendes Verständnis des Themas sowie auf die Erfahrungen und Meinungen des Autors.

3.2.2 Anwendungskontext für Stakeholder in der Maschinenentwicklung

Die in diesem Abschnitt beschriebenen Ergebnisse bauen auf den in Abschnitt 2.1.2 und 2.1.3 dargelegten Recherchen auf und beschreiben die Grundlagen für ein gemeinsames maschinenbauspezifisches Entwicklungsframework. Ein solches Framework ist hierbei für eine Reihe von Anwendungsfällen auszulegen, welche die Sichten der Stakeholder beschreiben und Grundlage für eine gemeinsame Modellierung bieten. Basierend auf diesen Anwendungsfällen werden in diesem Abschnitt die Zusammenhänge in der sicherheitsbezogenen Systementwicklung bezogen auf die systematische Vorgehensweise vorgestellt. In Abbildung 3.1 wird eine Übersicht

der Anwendungsfälle der Gesamtmodellierung-Lösungskontexte gegeben. Abbildung 3.2 bis Abbildung 3.5 definieren die Kontexte der betroffenen Stakeholder. Die Anwendungsfälle sind hierfür in fünf Anwendungskontexte gruppiert:

- Gemeinsame Anwendungsfälle in Grau: Dies sind Anwendungsfälle, bei denen verschiedene Stakeholder mit den Modellen interagieren.
- Anwendungsfälle des Maschinenbauers in Rot: Diese Anwendungsfälle gehören zur Modellierung und zum Entwurf der mechanischen Konstruktion.
- Anwendungsfälle des Elektroingenieurs in Blau: Diese Anwendungsfälle gehören zur Modellierung und zum Entwurf des elektrotechnischen Designs.
- Anwendungsfälle des Informatikers in Grün: Diese Anwendungsfälle gehören zur Modellierung und zum Entwurf der Software.
- Anwendungsfälle des Sicherheitsingenieurs in Orange: Diese Anwendungsfälle gehören zur Umsetzung der Vorgaben zur funktionalen Sicherheit.

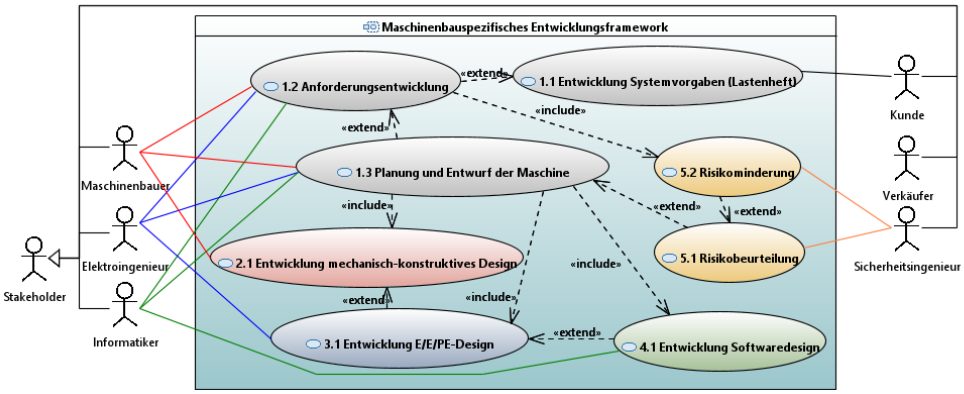


Abbildung 3.1: Anwendungskontext - Modellierung der Anwendungsfälle des Frameworks

Zusätzlich zu dieser Kategorisierung wird eine Nummerierung der Anwendungsfälle der Gesamtnutzungskontexte eingeführt, die anzeigt, in welchen Nutzungskontext ein bestimmter Anwendungsfall (UC) gehört. In den folgenden Absätzen findet sich für jeden UC eine kurze Beschreibung, beteiligte Aktoren und Vor- oder Nachbedingungen.

UC1.1 – Entwicklung Systemvorgaben (Lastenheft)

Akteure: Kunde, Verkäufer
Beschreibung: Kunde und Verkäufer erstellen gemeinsam eine Anforderungsspezifikation, mit den funktionalen und nichtfunktionalen Anforderungen, technischen Randbedingungen und Rahmenbedingungen aus Sicht des Kunden.
Nachbedingungen: Abnahme durch den Kunden

UC1.2 – Anforderungsentwicklung (Pflichtenheft)

Akteure: Maschinenbauer, Elektroingenieur, Informatiker
Beschreibung: Maschinenbauer, Elektroingenieur und Informatiker identifizieren gemeinsam die technischen Anforderungen an das zu entwickelnde System aus Sicht der Entwickler. Der Sicherheitsingenieur trägt durch risikomindernde Maßnahmen zur Anforderungsentwicklung in Bezug zur funktionalen Sicherheit bei (vgl. UC5.2).
Vorbedingungen: Lastenheft vorhanden.

UC1.3 – Planung und Entwurf der Maschine

Akteure: Maschinenbauer, Elektroingenieur, Informatiker
Beschreibung: Maschinenbauer, Elektroingenieur und Informatiker planen die Maschine in den für sie gewohnten Designstrategien.

Maschinenbauingenieur

Es ist wichtig, dass der Maschinenbauer als einer der ersten technischen Stakeholder in das Entwickler-Team eingebunden wird und eng mit den anderen Stakeholdern zusammenarbeitet. Dadurch kann sichergestellt werden, dass Anforderungen und Spezifikationen des mechanisch-konstruktiven Designs in Abhängigkeit zum elektrotechnischen Design in das Framework integriert werden und das Gesamtmodell ordnungsgemäß funktioniert. Abbildung 3.2 definiert die Kontexte des Maschinenbauers in Bezug zur Gesamtmodellierung.

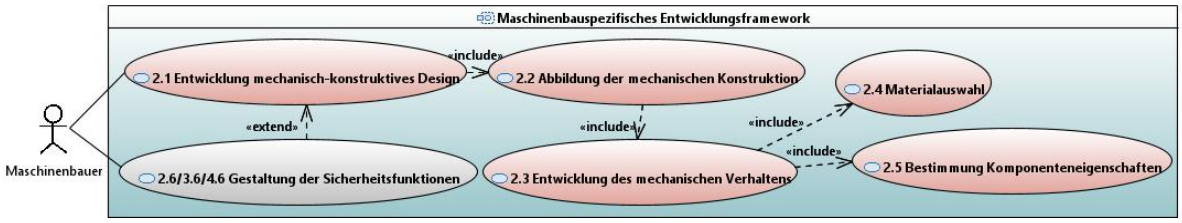


Abbildung 3.2: Anwendungskontext - Anwendungsfälle Maschinenbauer

UC2.1 – Entwicklung mechanisch-konstruktives Design

Akteure: Maschinenbauer
Beschreibung: Als Teil der Planung und des Entwurfs der Maschine entwirft der Maschinenbauer ein Designkonzept zur Erfüllung der mechanisch-konstruktiven Anforderungen an die Maschine.

UC2.2 – Abbildung der mechanischen Konstruktion

Akteure: Maschinenbauer
Beschreibung: Der Maschinenbauer entwirft die mechanische, hydraulische und pneumatische Konstruktion und den Aufbau der räumlichen Struktur. Hierbei nutzt er das E/E/PE-Design (vgl. UC3.2) zur Darstellung der mechanisch-konstruktiven Eigenschaften von E/E/PE-Komponenten. Erweiternde Anwendungsfälle zur Stakeholder-spezifischen Umsetzung und Analyse, die nicht Teil des Frameworks sind, finden sich unter Anhang A.8.1.

UC2.3 – Entwicklung des mechanischen Verhaltens

Akteure: Maschinenbauer
Beschreibung: Der Maschinenbauer entwickelt einzelne Komponenten, um das mechanische Verhalten zu beschreiben.
Vorbedingungen: Tatsächlich eingesetzte Komponenten und Materialien müssen bekannt sein.

UC2.4 – Materialauswahl

Akteure: Maschinenbauer
Beschreibung: Als Teil des mechanischen Verhaltens sind Materialeigenschaften, wie Steifigkeit und Festigkeit von Komponenten und Art der Schnittstellen vom Maschinenbauer anzugeben.
Vorbedingungen: Tatsächlich eingesetzte Komponenten und Materialien müssen bekannt sein.

UC2.5 – Bestimmung Komponenteneigenschaften

Akteure: Maschinenbauer
Beschreibung: Als Teil des mechanischen Verhaltens sind Komponenteneigenschaften, Gewicht, Volumen, Position, Dimensionen und Orientierung vom Maschinenbauer anzugeben.
Vorbedingungen: Tatsächlich eingesetzte Komponenten und Materialien müssen bekannt sein.

UC2.6 – Gestaltung der Sicherheitsfunktionen

Akteure: Maschinenbauer, Elektroingenieur, Informatiker
Beschreibung: Der Maschinenbauer erweitert das mechanisch-konstruktive Design zur Erfüllung der vom Sicherheitsingenieur identifizierten Sicherheitsfunktionen.
Vorbedingungen: Identifizierte Sicherheitsfunktionen.

Elektroingenieur

Es ist wichtig, dass der Elektroingenieur in enger Zusammenarbeit mit anderen technischen Stakeholdern in das Entwickler-Team eingebunden wird und eng mit dem Maschinenbauer und dem Informatiker zusammenarbeitet. Dadurch kann sichergestellt werden, dass alle relevanten Spezifikationen des elektrotechnischen Designs in Abhängigkeit zum mechanisch-konstruktiven Design in das Framework integriert werden. Ein Elektroingenieur kann auch bei der Erstellung von Verhaltensmodellen helfen, die die Funktionsweise der Maschine näher beschreiben und die Zusammenarbeit mit Informatikern erleichtern. Abbildung 3.3 definiert die Kontexte des Elektroingenieurs.

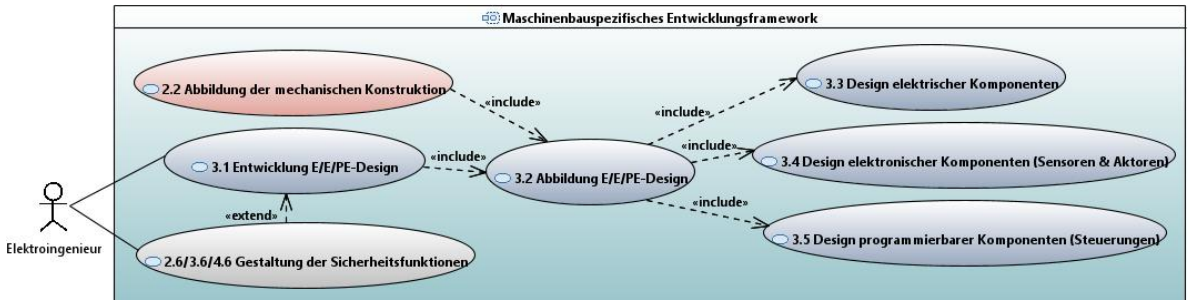


Abbildung 3.3: Anwendungskontext - Anwendungsfälle Elektroingenieur

UC3.1 – Entwicklung E/E/PE-Design

Akteure: Elektroingenieur
Beschreibung: Als Teil der Planung und des Entwurfs der Maschine entwirft der Elektroingenieur ein Designkonzept zur Erfüllung der elektrotechnischen Anforderungen an die Maschine.

UC3.2 – Abbildung E/E/PE-Design

Akteure: Elektroingenieur
Beschreibung: Der Elektroingenieur entwirft die elektrotechnische Struktur der Maschine mit Fokus auf den Energiefluss und die Verknüpfung, um den Zusammenhang elektrischer, elektronischer und programmierbarer elektronischer Komponenten darzustellen. Erweiternde Anwendungsfälle zur Stakeholder-spezifischen Umsetzung und Analyse, die nicht Teil des Frameworks sind, finden sich unter Anhang A.8.2.

UC3.3 – Design elektrischer Komponenten

Akteure: Elektroingenieur
Beschreibung: Der Elektroingenieur plant das Verhalten des elektrotechnischen Aufbaus und entwirft dabei die Struktur elektrischer Komponenten und deren Abhängigkeiten in den Modellen.
Vorbedingungen: Tatsächlich eingesetzte Komponenten und Materialien müssen bekannt sein.

UC3.4 – Design elektronischer Komponenten (Sensoren & Aktoren)

Akteure: Elektroingenieur
Beschreibung: Der Elektroingenieur plant das Verhalten des elektrotechnischen Aufbaus und entwirft dabei die Struktur, elektronische Komponenten und deren Abhängigkeiten in den Modellen, einschließlich elektronisch gesteuerter Pneumatik- und Hydraulikkomponenten.
Vorbedingungen: Tatsächlich eingesetzter Komponenten und Materialien müssen bekannt sein.

UC3.5 – Design programmierbarer Komponenten (Steuerungen)

Akteure: Elektroingenieur, Informatiker
Beschreibung: Der Elektroingenieur plant das Verhalten des elektrotechnischen Aufbaus und entwirft dabei die Struktur programmierbarer Komponenten gemeinsam mit dem Informatiker und deren Abhängigkeiten in den Modellen. Zusätzlich kann der Informatiker für Rückfragen zu Notwendigkeiten für die Softwaregestaltung eingebunden werden.

Vorbedingungen: Tatsächlich eingesetzte Komponenten und Materialien müssen bekannt sein.

UC3.6 – Gestaltung der Sicherheitsfunktionen

Akteure: Maschinenbauer, Elektroingenieur, Informatiker
Beschreibung: Der Elektroingenieur erweitert das E/E/PE-Design zur Erfüllung der vom Sicherheitsingenieur identifizierten Sicherheitsfunktionen. Dabei achtet er auf Ausfallsicherheitsparameter, wie MTTF und PFH, um die identifizierten Sicherheitsfunktionen umzusetzen.
Vorbedingungen: Identifizierte Sicherheitsfunktionen.

Informatiker

Es ist wichtig, dass der Informatiker in enger Zusammenarbeit mit anderen technischen Stakeholdern in das Entwickler-Team eingebunden ist und eng mit dem Maschinenbauer und dem Elektroingenieur zusammenarbeitet. Dadurch kann sichergestellt werden, dass die Anforderungen und relevanten Spezifikationen des Software-Designs in Abhängigkeit zu den elektrotechnischen Komponenten in das Framework integriert werden und das Gesamtmodell ordnungsgemäß funktioniert. Darüber hinaus kann der Informatiker bei der Integration von Modellen in das gesamte MBSE-Framework unterstützen, indem er seine Sichtweisen korrekt mit den Modellen und Daten der anderen Stakeholder verknüpft und sicherstellt, dass alle relevanten Informationen enthalten sind. Abbildung 3.4 definiert die Kontexte des Informatikers.

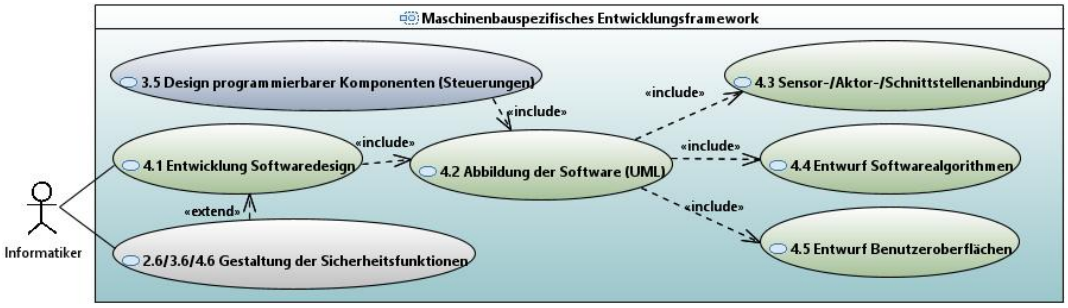


Abbildung 3.4: Anwendungskontext - Anwendungsfälle Informatiker

UC4.1 – Entwicklung Software-Design

Akteure: Informatiker
Beschreibung: Als Teil der Planung und des Entwurfs der Maschine entwirft der Informatiker den informationstechnischen Aufbau der Software zur Erfüllung der softwaregesteuerten Funktionalität der Maschine.

UC4.2 – Abbildung der Softwaremodelle

Akteure: Informatiker
Beschreibung: Der Informatiker entwirft die Software in Form von UML-Modellen zur Struktur- und Verhaltensmodellierung der Maschine, zum Ablauf auf programmierbaren Maschinenkomponenten, typischerweise SPSen. Er stellt dabei den Zusammenhang zwischen Hardware

und Software her. Erweiternde Anwendungsfälle zur Stakeholder-spezifischen Umsetzung und Analyse, die nicht Teil des Frameworks sind, finden sich unter Anhang A.8.3.

UC4.3 – Sensor-/Aktor-/Schnittstellenanbindung

Akteure: Informatiker
Beschreibung: Der Informatiker entwickelt die Anbindung von Sensoren und Aktoren, erstellt Treiber und implementiert andere Schnittstellen, wie beispielsweise zur Anbindung von Datenbanken. Dies geschieht auf der Grundlage von SysML-Struktur- und Verhaltensdiagrammen zur Gestaltung von UML-Komponenten.
Vorbedingungen: Tatsächlich eingesetzte Komponenten müssen bekannt sein.

UC4.4 – Entwurf Softwarealgorithmen

Akteure: Informatiker
Beschreibung: Der Informatiker entwirft Softwarealgorithmen, basierend auf SysML-Verhaltensdiagrammen zur Gestaltung und Konkretisierung von UML-Komponenten.
Vorbedingungen: Tatsächlich eingesetzte Komponenten müssen bekannt sein.

UC4.5 – Entwurf Benutzeroberflächen

Akteure: Informatiker
Beschreibung: Der Informatiker plant das Frontend der Softwareanwendung der Maschine auf Basis von SysML-Strukturdiagrammen.

UC4.6 – Gestaltung der Sicherheitsfunktionen

Akteure: Maschinenbauer, Elektroingenieur, Informatiker
Beschreibung: Der Informatiker erweitert das Software-Design zur Erfüllung der vom Sicherheitsingenieur identifizierten Sicherheitsfunktionen. Dabei achtet er auf die Erfüllung der Ausfallsicherheit auf Basis von Maßnahmen gegen CCF und Diagnosemaßnahmen, um die identifizierte Sicherheitsfunktion umzusetzen.
Vorbedingungen: Identifizierte Sicherheitsfunktionen.

Sicherheitsingenieur

Der Sicherheitsingenieur erweitert die Sichten des Maschinenbauers, Elektroingenieurs und Informatikers für die Bewertung der funktionalen Sicherheit, indem er seine Kenntnisse und Fähigkeiten in der Risikoanalyse, -bewertung und -minderung einbringt. Hierbei gibt er mitunter Grundlage für die Modelle der anderen Stakeholder, um sicherzustellen, dass die Maschine den geltenden Sicherheitsstandards und -vorschriften entspricht. Dies macht es umso wichtiger, dass der Sicherheitsingenieur eng mit dem Maschinenbauer, Elektroingenieur und Informatiker zusammenarbeitet, um das Gesamtmodell sicher und zuverlässig modellieren und relevante Sicherheitsaspekte angemessen berücksichtigen zu können. Abbildung 3.5 definiert die Kontexte des Sicherheitsingenieurs in Bezug zur Gesamtmodellierung. Aufgrund der ausführlichen

Grundlagen aus Abschnitt 2.1.3 und dem Fokus dieser Arbeit auf die Aspekte der funktionalen Sicherheit, wurden die Anwendungsfälle feiner gegliedert.

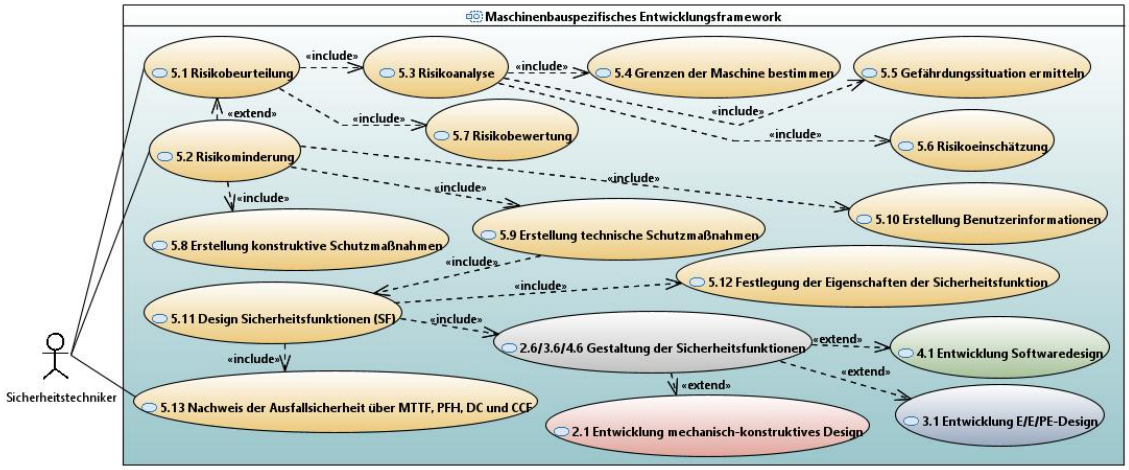


Abbildung 3.5: Anwendungskontext - Anwendungsfälle Sicherheitsingenieur

UC5.1 – Risikobeurteilung

Akteure: Sicherheitsingenieur
Beschreibung: Zur Überprüfung der Planung und des Entwurfs der Maschine, einschließlich mechanischer, elektrotechnischer und informationstechnischer Planung, führt der Sicherheitsingenieur eine Risikobeurteilung durch, in der er die Gefährdungen der Maschine aufschlüsselt und deren Auswirkungen für Mensch, Maschine und Umwelt bestimmt.
Vorbedingungen: Plan der Anforderungen und erster Entwurf der Subsysteme der Maschine.

UC5.2 – Risikominderung

Akteure: Sicherheitsingenieur
Beschreibung: Anhand der Risikobeurteilung der Maschine führt der Sicherheitsingenieur risikomindernde Maßnahmen unter Vorgabe der Normen ein.
Vorbedingungen: Risikoitem vorhanden.

UC5.3 – Risikoanalyse

Akteure: Sicherheitsingenieur
Beschreibung: Der Sicherheitsingenieur analysiert die Teile der Maschine, um mögliche Gefährdungen bis hin zu Folgen zu bestimmen.
Vorbedingungen: Plan der Anforderungen und erster Entwurf der Teile der Maschine.

UC5.4 – Grenzen der Maschine bestimmen

Akteure: Sicherheitsingenieur
Beschreibung: Der Sicherheitsingenieur nutzt die Anforderungen an die Maschine und die identifizierten Maschinenteile als Grundlage zur Bestimmung der Grenzen der Maschine und als Rahmen für die Risikoanalyse.

UC5.5 – Gefährdungssituation ermitteln

Akteure: Sicherheitsingenieur
Beschreibung: Anhand der identifizierten Gefährdungen und abhängigen Maschinenteile identifiziert der Sicherheitsingenieur Gefährdungssituationen, bestehend aus den ihr ausgesetzten Personen, dem Gefährdungsereignis, der Ursache und den Lebensphasen, in denen die Ursache auftritt.
Vorbedingungen: Anforderungen und Grenzen der Maschine vorhanden.

UC5.6 – Risikoeinschätzung

Akteure: Sicherheitsingenieur
Beschreibung: Anhand einer oder mehrerer gleichförmiger Gefährdungssituationen der Maschine leitet der Sicherheitsingenieur Effekte ab, die entstehen können. Die Einschätzung leitet er vom Schadensausmaß und der Eintrittswahrscheinlichkeit ab.
Vorbedingungen: Gefährdungssituation identifiziert.

UC5.7 – Risikobewertung

Akteure: Sicherheitsingenieur
Beschreibung: Der Sicherheitsingenieur berechnet aus dem Schadensausmaß, der Gefährdungsexposition, der Möglichkeit zur Vermeidung oder Begrenzung des Schadens und dem Eintritt eines Gefährdungsereignisses das resultierende Risiko, mittels Risikograph.
Vorbedingungen: Risikoitems vorhanden.

UC5.8 – Erstellung konstruktive Schutzmaßnahmen

Akteure: Sicherheitsingenieur
Beschreibung: Der Sicherheitsingenieur entwickelt risikomindernde Schutzmaßnahmen, durch konstruktive Vermeidung der Ursache, konstruktive Minderung des Gefährdungsereignisses oder Minderung der entstehenden Effekte.
Vorbedingungen: Risikoitem vorhanden.
Nachbedingungen: Konstruktive Anforderungen.

UC5.9 – Erstellung technische Schutzmaßnahmen

Akteure: Sicherheitsingenieur
Beschreibung: Insofern konstruktive Schutzmaßnahmen nicht ausreichen, entwickelt der Sicherheitsingenieur risikomindernde technische Schutzmaßnahmen, die die Auswirkungen des Gefährdungsereignisses und der Effekte mindern.
Vorbedingungen: Risikoitem vorhanden.
Nachbedingungen: Technische Anforderungen.

UC5.10 – Erstellung Benutzerinformationen

Akteure: Sicherheitsingenieur
Beschreibung: Insofern technische Maßnahmen nicht ausreichen, sind Benutzerinformationen zu entwickeln, die vom Bedienpersonal zu beachten sind, um das Restrisiko zu mindern.
Vorbedingungen: Risikoitem vorhanden.
Nachbedingungen: Benutzerinformationsanforderungen.

UC5.11 – Design Sicherheitsfunktionen

Akteure: Sicherheitsingenieur
Beschreibung: Der Sicherheitsingenieur entwirft die Art der Maßnahmen und Sicherheitsfunktionen.
Vorbedingungen: Technische Schutzmaßnahme bestimmt.

UC5.12 – Festlegung der Eigenschaften der Sicherheitsfunktionen

Akteure: Sicherheitsingenieur
Beschreibung: Der Sicherheitsingenieur entwirft das Verhalten und legt die notwendige Sicherheitsintegrität der Sicherheitsfunktionen fest.
Vorbedingungen: Technische Schutzmaßnahme bestimmt.
Nachbedingungen: Hardwareanforderungen und Softwareanforderungen (sofern softwaregesteuert).

UC5.13 – Nachweis der Ausfallsicherheit über MTTF, PFH, DC und CCF

Akteure: Sicherheitsingenieur
Beschreibung: Anhand der durch den Maschinenbauer, Elektroingenieur und Informatiker entwickelten Sicherheitsfunktionen und der identifizierten Ausfallparameter einzelner Komponenten und Subsysteme entwickelt der Sicherheitsingenieur den erreichten PL jeder einzelnen Sicherheitsfunktion.
Vorbedingungen: Design der Sicherheitsfunktion.

Diese Anwendungskontexte und -fälle tragen zur Forschungsaufgabe F1/TB bei und zeigen, wie ein maschinenbauspezifisches Entwicklungsframework aus der Perspektive der technischen Stakeholder aufgebaut und befüllt werden kann. Sie stellen die Grundlage für ein detailliertes konzeptionelles Modellieren dar, das auf einem gemeinsamen Rahmenwerk basiert. Dieses Rahmenwerk wird in den nächsten Abschnitten eingeführt und ab sofort als maschinenbauspezifische MBSE-Framework-Erweiterung (MMBSEFE) bezeichnet.

3.2.3 Gemeinsames Informationsmodell

Für die Konzeption eines gemeinsamen Informationsmodells für den Maschinenbau, im Speziellen für Prüfmaschinen, wird auf Abschnitt 2.1.4 und dem RAAML-Ansatz basierend MMBSEFE entwickelt. Es besteht aus einer aus MOF abgeleiteten Metamodellerweiterung, um die domänenspezifischen Begriffe und Spezialisierungen in der Maschinenindustrie zu unterstützen. Sein Hauptzweck besteht darin, eine gemeinsame Kommunikationsbasis und Grundlage zur Entwicklung von Prüfmaschinen aufzubauen. MMBSEFE kann als Zusammenschluss unterschiedlicher Pakete mit Profilen und Bibliotheken betrachtet werden, die die wesentlichen Sichtweisen für die Entwicklung sicherheitskritischer Anwendungen in der Maschinenindustrie

abbilden. Hierbei können in den Profilen Stereotypen Einsatz finden, die Stakeholder-spezifische Begriffe auf SysML-Elemente aufprägen, um diese später während der Systementwicklung als Knoten einsetzen zu können.

Um die Vorteile von SysML zu nutzen und von den vorhandenen Modellierungswerkzeugen zu profitieren, dürfen keine Widersprüche zwischen der Erweiterung, UML und SysML eingeführt werden. Dies wäre der Fall, wenn die Profile und Bibliotheken wie in der skizzieren Architektur a), links in Abbildung 3.6 dargestellt, entwickelt werden, da hier neue Konstrukte in MOF aufgenommen werden müssen. Außerdem lassen sich auf diese Weise keine Mechanismen oder Standardbibliotheken von UML und SysML nutzen, was sich negativ auf die Wiederverwendbarkeit auswirkt. Deshalb wird die skizziere Metaarchitektur b), rechts in Abbildung 3.6, gewählt, um eine Erweiterung zu erstellen.

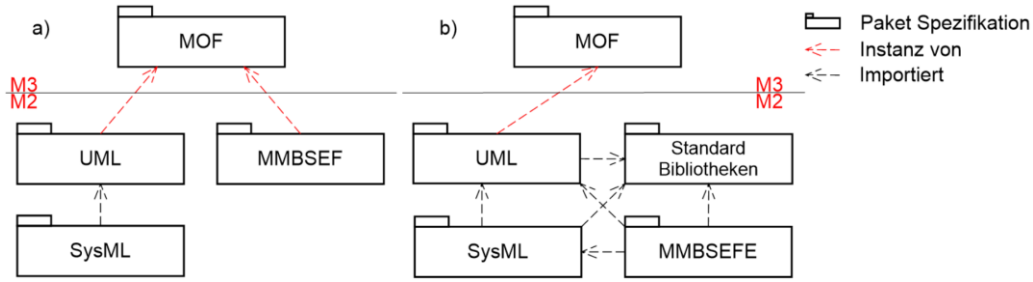


Abbildung 3.6: Mögliche Schichtenmodelle Metaarchitektur MMBSEFE a) Erweiterung MOF (b) Erweiterung UML & SysML

Unter Anwendung dieser Konfiguration entstehen zwei verschiedene Lösungsstrategien. Die erste Möglichkeit besteht im Hinzufügen abgeleiteter Klassen der grundlegenden Elemente direkt in UML und SysML, die SysML mit allen Mechanismen der notwendigen Paradigmen für MMBSEFE ausstattet. Die andere Möglichkeit besteht darin, soweit möglich vorhandene Kindklassen und Spezialisierungen aus UML und SysML wiederzuverwenden, um sie dem Framework zuzuordnen. Das UML-Metamodell ist in der Regel nicht für Modifikationen offen, wodurch die zweite Lösung das Architekturkonzept besser unterstützt.

Standardmodellbibliotheken, in Abbildung 3.6 Metaarchitektur b) dargestellt, können in MBSE-Frameworks integriert werden, um den Modellierungsprozess z.B. von Echtzeitsystemen und Datenqualitätsaspekten zu erleichtern und zu verbessern. Eine Möglichkeit besteht darin, eine Modellierungssprache für MMBSEFE zu definieren, die die Syntax und Semantik der Bibliotheken direkt einbindet. Diese Modellierungssprache kann dadurch eine bessere Basis für die Entwicklung der Stakeholder-spezifischen Systemmodelle liefern, indem die Anforderungen und Funktionalitäten basierend auf realen eingebetteten Eigenschaften des Systems und Datenqualitätsaspekten beschrieben werden. Es entsteht eine verbesserte Qualität durch präzisere und formalisierte Modelle, die zu einer höheren Effizienz beitragen, da direkt eine Basis

für die Stakeholder-spezifischen Modelle in den Entwicklungsumgebungen entsteht. Zusätzlich lassen sich die Standardbibliotheken wiederverwenden, um ähnliche Maschinen oder Maschinentypen umzusetzen, was zusätzlich Zeit und Kosten spart. Es entsteht eine gemeinsame Sprache, was die Interoperabilität nicht nur zwischen verschiedenen Systemen, sondern auch zwischen Organisationen erleichtern kann. Für MMBSEFE sollen daher die Standardbibliotheken zu ISO 80000 [26, 275] und Modeling and Analysis of Real-Time and Embedded Systems (MARTE) [276] verwendet werden.

Die Norm ISO 80000 definiert einen Leitfaden für eine einheitliche und konsistente Verwendung von physikalischen Größen und Einheiten, mathematische Zeichen und Symbole, SI-Einheiten und Einheiten außerhalb des SI-Systems, um die Kommunikation und das Verständnis zwischen verschiedenen Ländern, Organisationen und Branchen zu verbessern [275]. Der ISO 80000 Standard ist Teil der OMG SysML Spezifikation und kann als Erweiterung von SysML verstanden werden [26]. Diese Erweiterung definiert eine Modellbibliothek von SysML QuantityKind- und Unit-Definitionen für eine Teilmenge der Quantitäten und Einheiten. Sie bietet einen zusammenfassenden Überblick über Definitionen von Einheiten und Mengenarten, die nach ISO 80000-Teil 1 Präfixe, Teil 2 mathematische Zeichen und Symbole, Teil 3 Raum und Zeit, Teil 4 Mechanik, Teil 5 Thermodynamik, Teil 6 Elektromagnetismus, Teil 7 Licht, Teil 9 Physikalische Chemie und Molekularphysik, Teil 10 Atom- und Kernphysik und Teil 13 Informationswissenschaft und -technologie gruppiert und indiziert sind.

MARTE ist eine Erweiterung von UML, die speziell für die Modellierung und Analyse von Echtzeitsystemen und eingebetteten Systemen von der OMG publiziert wurde. Die MARTE-Erweiterung bietet in einer eigenen Spezifikation zusätzliche Konzepte und Modellierungselemente, die es ermöglichen, spezifische Aspekte von Echtzeitsystemen wie zeitliche Einschränkungen und Verteilungsfunktionen, Ressourcenverbrauch und andere Einschränkungen zu modellieren [276]. SysML wurde zwar speziell für die Modellierung von Systemen entwickelt, doch enthält es keine spezifischen Elemente für die Modellierung von Echtzeitsystemen. In SysML können MARTE-Elemente für zeitliche Ereignisse, Bedingungen und Limits, Ressourcenverbrauch, Verarbeitungsleistung und Speicherverbrauch verwendet werden, um Echtzeitaspekte von Systemen und somit Wahrscheinlichkeitsverteilungen zu modellieren und die Qualität entwickelter Maschinen zu verbessern.

Im nächsten Schritt ist es wichtig, Sichten auszuwählen, die leicht als Stakeholder-spezifische Modelle erkennbar gemacht werden können, um den Lernaufwand für die Ingenieure zu begrenzen. Andererseits ist es wesentlich, die Semantik der UML-Diagramme für jede Einschränkung des ursprünglichen Diagramms zu respektieren.

In SysML werden verschiedene Diagramme verwendet, um die Struktur von Systemen zu beschreiben, die in diesem Framework wiederverwendet werden sollen. Blockdefinitionsdiagramme (BDD) sollen genutzt werden, um die Struktur einer Maschine im Ganzen zu beschreiben, wohingegen Internal-Blockdiagramme (IBD) die Stakeholder-spezifischen Vernetzungen von Maschinenteilen, Komponenten und Subkomponenten beschreiben.

Für die Entwicklung von Maschinen wird zunächst eine grundlegende Struktur mithilfe von Paket-Diagrammen entworfen. Eine Zusammenfassung der Formalisierungen basierend auf grau hinterlegten Paketen aus RAAML findet sich in Abbildung 3.7. Jeder Maschinentyp wird als `Construction` betrachtet und muss unabhängig der Domäne oder den Stakeholdern anwendbar sein. Aus dieser Konstruktion (`ConstructionStructure`) kann ein Konzept (`MachineConstructionConcept`) entwickelt werden, das eine grobe Struktur zwischen den bekannten Stakeholder-Sichten und Maschinenteilen für Maschinentypen herstellt. Hierzu wird ein Paket erstellt, das den Aufbau der Maschine (`MachineAssemblyStructure`) in Maschinenteile unterteilt und den entsprechenden Sichten zuordnet. Die Unterteilung orientiert sich dabei an der funktionalen Sicherheit. Um die Anwendung des Frameworks für Sondermaschinen und insbesondere für Prüfmaschinen zu erleichtern, werden zusätzliche Spezialisierungen eingesetzt.

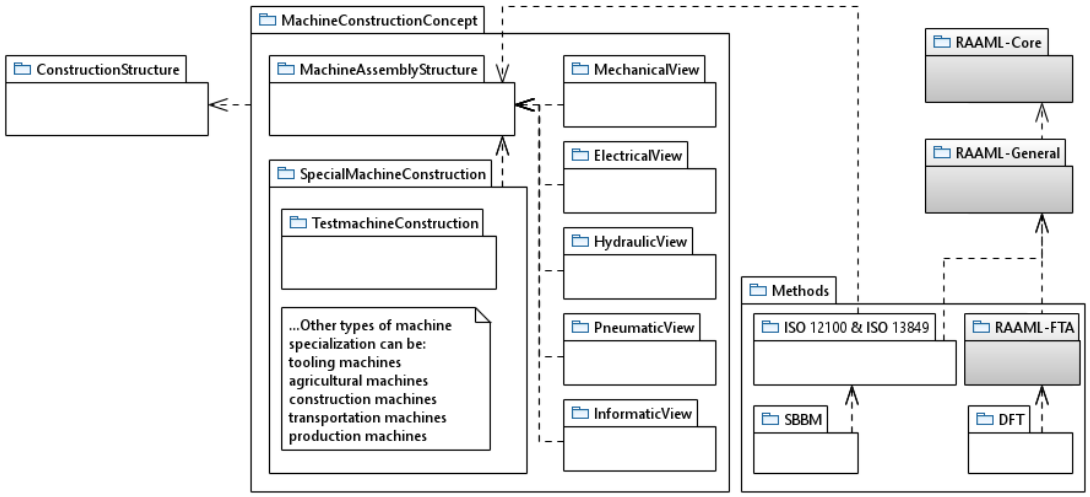


Abbildung 3.7: Grobe MMBSEFE-Struktur für die Anwendungsdomäne Links: Entwicklungsingenieure, Rechts: RAAML Erweiterung)

Wie aus Abschnitt 2.1.4 hervorgeht, sind Sondermaschinen (`SpecialMachineConstruction`) keinem speziellen Maschinentyp zugeordnet und folgen

daher dem allgemeinen Aufbau von Maschinen. Prüfmaschinen (`TestmachineConstruction`) hingegen spezialisieren das Konzept, um die Maschinenteile und deren Funktionen in Prüfmaschinen genauer abbilden zu können. Eine weitere Spezialisierung für Zugprüf-, Druckprüf- und Batterieprüfmaschinen wird vermieden, da es sich dabei nur um eine Teilmenge von Prüfmaschinen handelt, die ebenfalls im Anwendungsmodell der entsprechenden Maschine ausgewählt werden können.

Die folgenden Unterabschnitte beschreiben detailliert die Hierarchie der einzelnen Pakete und die Abhängigkeiten untereinander auf Basis von Abbildung 3.7 und abhängig der in Abschnitt 2.1.1 bis 2.1.4 bereitgestellten Grundlagen. Das Profil jedes Pakets definiert die Erweiterungsartefakte, verknüpft das Framework mit UML- und SysML-Artefakten und baut Abhängigkeiten des Profils mit anderen Profilen des Frameworks auf. Die Bibliothek hingegen bildet die Struktur des Pakets und Abhängigkeiten zu anderen Bibliotheken ab. Ein detaillierter Aufbau der `MachineConstructionConcept`-Struktur findet sich unter Anhang A.9.1. Die Beschreibung der sicherheitsbezogenen Anbindung an RAAML für die Anwendungsdomäne findet sich in Abschnitt 3.2.11.

3.2.4 Construction Structure

Auf Basis von [32] und anderen Ansätzen in Abschnitt 2.1.3 folgt, dass jede technische Konstruktion im Allgemeinen als Gesamtkonstruktion betrachtet werden kann, die aus einer Vielzahl von Teilen besteht. Generell können also nicht nur CAD-Zeichnungen, sondern auch Schaltpläne und Softwaremodelle in Komponenten und rekursive Subkomponenten unterteilt werden, wie bereits über die OMG UML Spezifikation [25] klar ersichtlich wird. Die Gesamtkonstruktion kann eine Maschine, aber auch ein Gebäude, eine Brücke, ein elektronisches Gerät oder eine Softwareanwendung darstellen. Diese Teile können sowohl physischer als auch virtueller Natur sein und umfassen im Bauwesen beispielsweise Fundamente, Stützstrukturen, Tragwerke und Verbindungsstücke, im Allgemeinen jedoch verschiedene konstruktive, mechanische, elektromechanische, elektrische, elektronische und programmierbare elektronische Aufbauten oder auch Software.

Dementsprechend muss es für jedes Teil einer Konstruktion möglich sein, rekursiv verschachtelte Teile aufzubauen, die bis zu einzelnen Grundelementen führen. Es ist jedoch wichtig zu beachten, dass die Zuordnung von Teilen zu einer bestimmten Ebene in der Konstruktion von der Konstruktion selbst und auch vom Modellierer abhängt. So kann ein Element, das als Grundelement in einer Konstruktion definiert ist, auch als Teilkonstruktion in einer anderen Konstruktion verwendet werden.

Dieser Formalismus soll als Grundlage für MMBSEFE dienen. Die Bibliothek und das Profil sind im Folgenden in Abbildung 3.8 und Abbildung 3.9 beschrieben. Erläuterungen zu den einzelnen Elementen finden sich für eine übersichtliche Darstellung unter Anhang A.9.2.

ConstructionStructure-Bibliothek

Abbildung 3.8 gibt das allgemeine Konzept einer Konstruktion wieder.

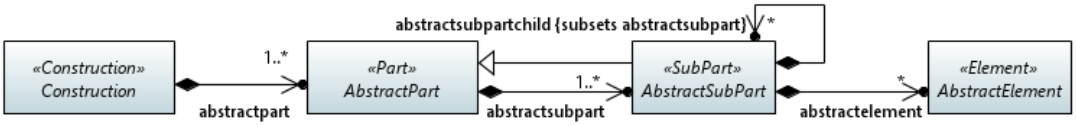


Abbildung 3.8: ConstructionStructure-Bibliothek

Jede `Construction` besteht aus mindestens einem `AbstractPart`. Jeder `AbstractPart` besteht aus mindestens einem `AbstractSubPart`, der eine Spezialisierung von `AbstractPart` darstellt. Jeder `AbstractSubPart` kann aus weiteren `AbstractSubPart` und `AbstractElement`-Elementen bestehen, wobei mindestens ein `AbstractSubPart` oder `AbstractElement` angegeben werden muss. `AbstractElement` stellt die unterste Ebene im Framework dar.

ConstructionStructure-Profil

Abbildung 3.9 gibt die Stereotypen des allgemeinen Konzepts wieder.

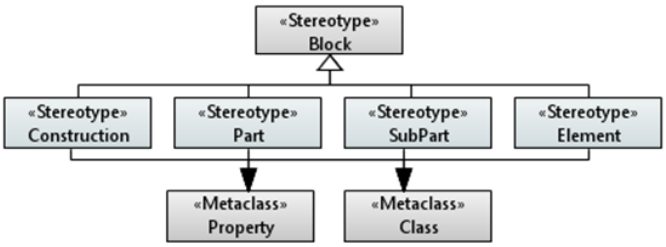


Abbildung 3.9: ConstructionStructure-Profil

Die Stereotypen `Construction`, `Part`, `SubPart` und `Element` repräsentieren Spezialisierungen von SysML Block sowie Erweiterungen von UML Property und UML Class. Sie nutzen die folgenden Funktionalitäten des Blockkonzepts: Generalisierung, Ports, Teile, Wert-Eigenschaften und Parametrik. Die Stereotypen werden benötigt, um Konstruktionen, Teile, untergeordnete Teile und Elemente von anderen Arten von Blöcken zu unterscheiden. Die Erweiterung von `Property` ermöglicht es, die Stereotypen ebenso in IBD zu nutzen, um Strukturen von Konstruktionen zu erstellen.

3.2.5 Machine Assembly Structure

Das Paket `MachineAssemblyStructure` implementiert das zuvor beschriebene `ConstructionStructure`-Konzept für Maschinen und fügt die Unterteilung in Stakeholder-

spezifische Sichten hinzu, um die in der Maschinenindustrie üblichen Maschinenteile konkret darzustellen. Zur Schaffung einer ausreichenden Grundlage für die Risikobeurteilung nach ISO 12100 werden im Zusammenhang stehende Maschinenteile in vier Kategorien unterteilt, die Einfluss auf die Risikobeurteilung haben können.

Die Umgebung einer Maschine beschreibt die räumlichen Grenzen, in denen die Maschine eingesetzt wird. Dies umfasst nach ISO 12100 den Aufstellort sowie Bewegungsbereiche, Platzbedarf von Personen, Arbeitsflächen und andere räumliche Bedingungen wie Staub- oder Feuchtigkeitsresistenz. Die Umgebung kann somit die Stabilität und Standfestigkeit der Maschine beeinflussen oder besondere Sicherheitsvorkehrungen erfordern, um Bediener und andere Personen vor Gefahren wie Lärm, Strahlung oder chemischen Stoffen zu schützen.

Bei der Betrachtung des Fertigungsprozesses sowie der Qualitätssicherung und -kontrolle einer Maschine ist es von entscheidender Bedeutung, die zu produzierenden oder zu testenden Teile angemessen zu berücksichtigen. Beispielsweise kann ein zu produzierendes oder zu testendes Teil eine besondere Handhabung erfordern, die zusätzliche Risiken mit sich bringt, oder es kann das Vorhandensein von gefährlichen Stoffen erfordern, die spezielle Schutzvorkehrungen notwendig machen.

Die Hauptbestandteile einer Maschine stellen die funktionalen Teile der Maschine dar, die sich teilweise an den Gefährdungsgruppen (vgl. Anhang A.5) orientieren, um eine eindeutige und lückenlose Zuordnung zur Risikobeurteilung zu garantieren. Die Unterteilung in funktionale Teile einer Maschine wurde gemeinsam mit einem erfahrenen Sicherheitsingenieur von ProNES diskutiert und identifiziert. Diese funktionalen Teile umfassen die mechanisch-konstruktive Struktur, bewegliche Teile, Messeinheiten, Befülleinheiten, geräuschproduzierende Teile, Zuführungs- und Entnahmeeinrichtungen, schneidende und scherende Teile, produktproduzierende Teile, Temperiereinheiten, schwingende Teile, Bestrahleinrichtungen, Eingabe-, Steuerungs- und Anzeigeeinheiten sowie Energiequellen. Die Zuordnung dieser Teile zu Gefährdungen ist jedoch nicht direkt, sondern sie dienen lediglich als Hinweis auf potenzielle Risiken. Zum Beispiel können bewegliche Teile nicht nur mechanische und elektrische Gefahren, sondern auch Vibrationen verursachen, ohne schwingende Teile zu sein. Ein weiteres Beispiel sind Lichtgitter, die zwar keine Bestrahleinrichtungen sind, aber Strahlungsgefahren verursachen können.

Als Letztes sind ebenso sicherheitsbezogene Teile der Maschine zu betrachten, die bereits als Ergebnis der Risikominderung identifiziert wurden. Diese müssen sorgfältig bewertet und in die Analyse einbezogen werden, um sicherzustellen, dass sie korrekt ausgelegt, installiert und

funktionsfähig sind und den relevanten Normen und Vorschriften entsprechen. Sicherheitsrelevante Teile werden in der Regel durch Sicherheitseinrichtungen und Sicherheitssteuerungen dargestellt, wobei Sicherheitssteuerungen Teil von Sicherheitseinrichtungen sein können.

Im Hinblick auf die spezifischen Anforderungen der Stakeholder muss jedem Teil einer Maschine ein mechanischer Aufbau zugeordnet werden. Dies bildet die Grundlage für das E/E/PE, hydraulische, pneumatische und Software-Design, aus denen ein Gesamtkonzept entsteht. Nicht alle Sichten sind jedoch für jeden Teil relevant. Beispielsweise spielt das elektronische Design bei der Modellierung eines Maschinenrahmens keine große Rolle, während das Software-Design bei rein elektrischen oder elektronischen Teilen nicht von Bedeutung ist. Es kann natürlich auch sein, dass sich Teile der Maschine Unterbaugruppen teilen, die dann nur im relevanten Teil der Maschine beschrieben werden. So kann beispielsweise eine Achsenansteuerung als Einzelteil des Maschinenrahmens modelliert, jedoch aus der elektrotechnischen Sicht nur als Teil eines bewegenden Teils ausgeführt werden.

Dieser Formalismus dient als Grundlage für den Aufbau von Maschinen in MMBSEFE.

Im Folgenden wird die Bibliothek beschrieben. Erläuterungen zu den einzelnen Elementen und die Darstellung des Profils finden sich für eine übersichtlichere Darstellung in Anhang A.9.3.

MachineAssemblyStructure-Bibliothek

Abbildung 3.10 gibt das allgemeine Konzept einer Maschine wieder.

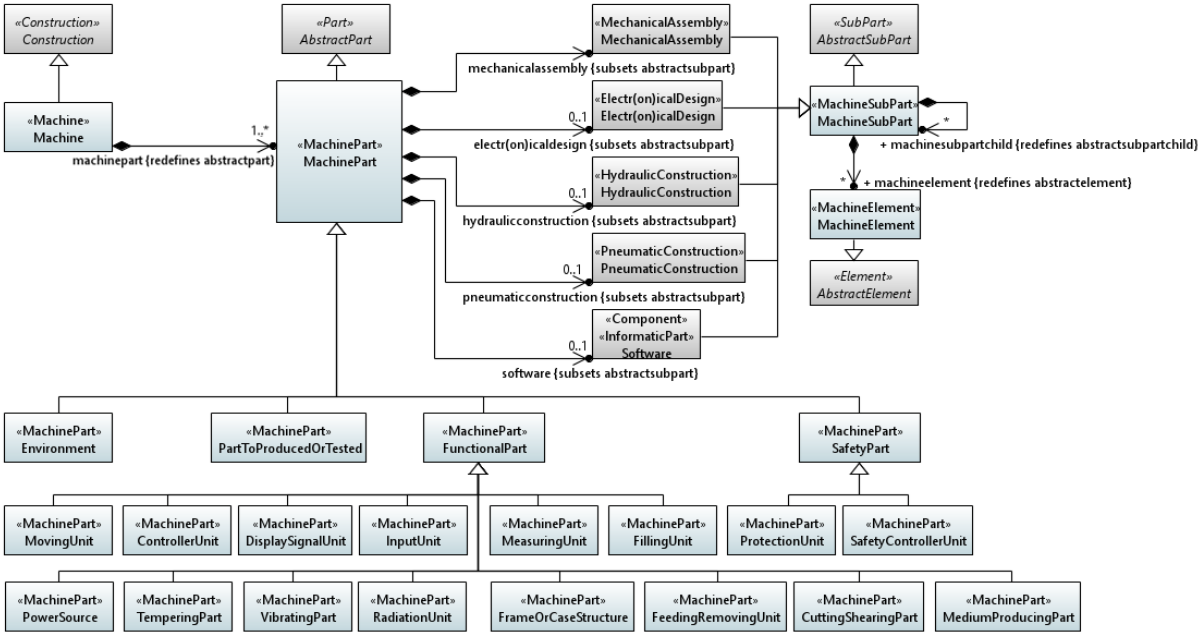


Abbildung 3.10: MachineAssemblyStructure-Bibliothek

`Machine` spezialisiert die Klasse `Construction`, `MachinePart` die Klasse `AbstractPart`, `MachineSubPart` die Klasse `AbstractSubPart` und `MachineElement`

die Klasse AbstractElement. Eine Machine besitzt mindestens einen MachinePart und definiert hierfür die Komposition von Construction zu AbstractPart neu. MachinePart definiert die Komposition von AbstractPart zu AbstractSubPart neu, um Subsets für MechanicalAssembly, Electr(on)icalDesign, HydraulicConstruction, PneumaticConstruction und Software zu schaffen, die jeweils Spezialisierungen von AbstractSubPart sind. Zusätzlich ist MachinePart eine Generalisierung der vier Kategorien, wobei FunctionalPart und SafetyPart Generalisierungen der allgemeinen funktionalen und sicherheitskritischen Maschinenteile darstellen. Zuletzt definiert MachineSubPart die Kompositionen zu untergeordneten Teilen und Elementen neu.

3.2.6 Testmachine Construction

Das Paket TestmachineConstruction wurde entwickelt, um das Konzept der MachineAssemblyStructure für den Einsatz bei der Gestaltung von Prüfmaschinen zu spezialisieren. Es wurde festgestellt, dass in der Entwicklung von Prüfmaschinen kein großer Unterschied zu anderen Maschinen besteht, da die gleichen Stakeholder involviert sind. Das Paket TestmachineConstruction konzentriert sich jedoch auf die spezifischen Bedürfnisse von Prüfmaschinen und bietet daher optimierte Lösungen für Prüfmaschinenteile.

Somit wird das Konzept der MachineAssemblyStructure für Prüfmaschinen wiederverwendet, um basierend auf der ISO 12100 eine effiziente und sichere Arbeitsweise zu gewährleisten und die Anforderungen an die Gestaltung von Maschinen einzuhalten. TestmachineConstruction konzentriert sich darauf, das Konzept der MachineAssemblyStructure für Prüfmaschinen anwendbar zu machen. Hierzu werden die zuvor im Einklang mit ISO 12100 definierten Maschinenteile konkretisiert, um eine einfachere Anwendung für Prüfmaschinen zu bewirken. Erläuterungen zu den einzelnen Elementen und dem Profil finden sich in Anhang A.9.4.

TestmachineConstruction-Bibliothek

Die aus den allgemeinen Maschinenteilen entwickelte Bibliothek findet sich in Abbildung 3.11. Für eine bessere Lesbarkeit wird diese im Querformat dargestellt.

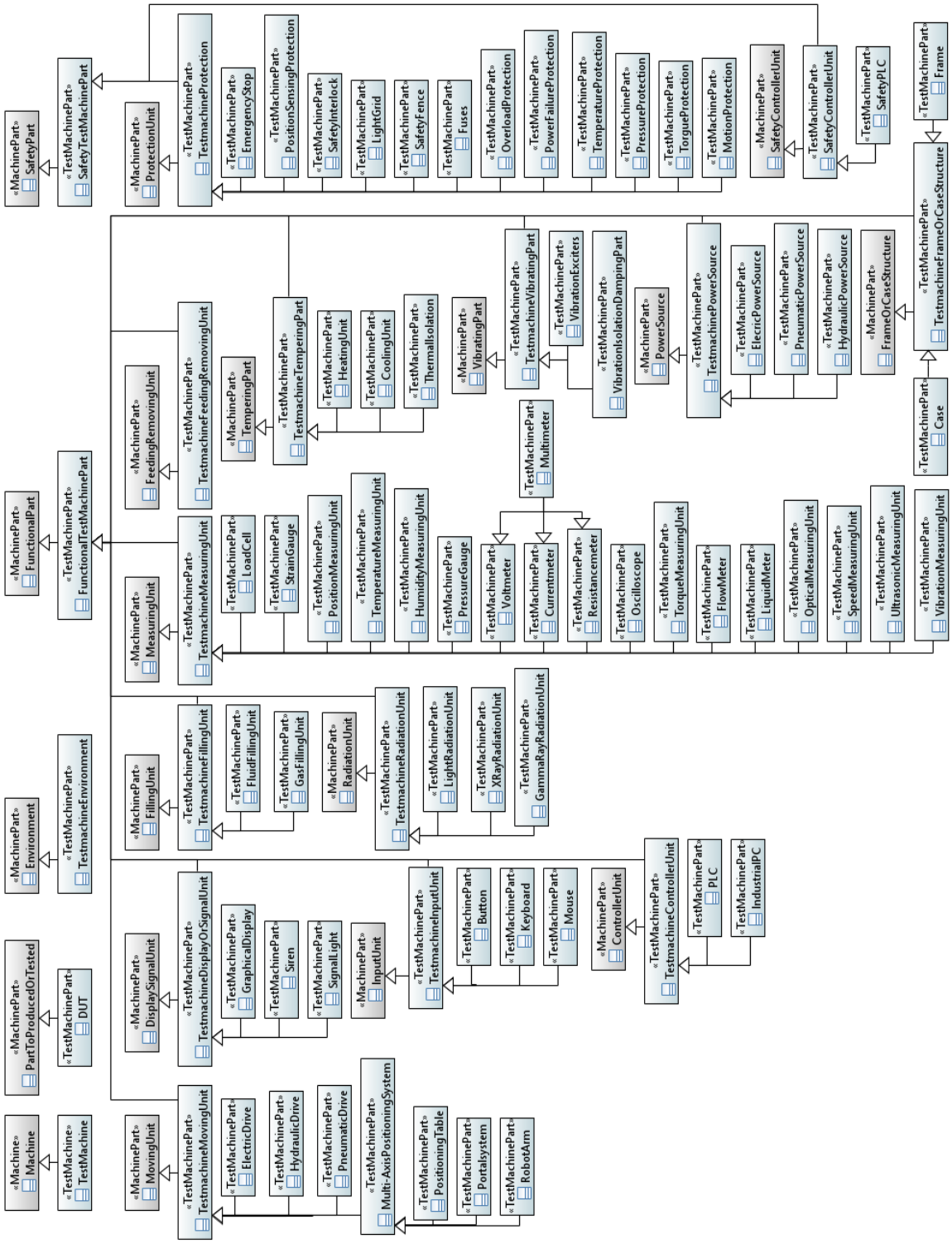


Abbildung 3.11: TestmachineConstruction-Bibliothek

Gemäß den Spezifikationen sollen Prüfmaschinen ausschließlich für das Testen von Prüflingen verwendet werden, nicht jedoch zur Produktion von Teilen. Daher wird eine Spezialisierung für `PartToProducedOrTested` für Prüflinge definiert. Die mechanisch-konstruktive Struktur der Prüfmaschinen, genannt `FrameOrCaseStructure`, ist identisch, jedoch wird eine Unterscheidung in Gehäuse und Rahmen zur einfachen Unterscheidung vorgenommen. Für die beweglichen Teile der Prüfmaschine, genannt `MovingUnit`, können unterschiedliche

Arten von Antriebstypen definiert werden, wie elektrische, pneumatische und hydraulische Antriebe sowie mehrachsige Antriebs- und Positioniersysteme wie Roboterarme, Portalsysteme oder Positioniertische. Für das Messen von Prüflingen, genannt `MeasuringUnit`, können Arten von physikalischen Messeinheiten eingesetzt werden, wie Kraft-, Dehnungs-, Temperatur-, Spannungs-, Widerstands- und Frequenzmessungen. Die Befülleinheiten, genannt `FillingUnit`, füllen die Prüfmaschine oder den Prüfling entweder mit Gas oder Flüssigkeit. Zuführungs- und Entnahmeeinrichtungen, genannt `FeedingRemovingUnit`, können nicht weiter konkretisiert werden. Prüfmaschinen haben keine Teile, die für schneidende oder scherende Bewegungen zuständig sind, da diese eher in der Produktion als beim Testen eingesetzt werden. Ebenso gibt es keine mediumproduzierende Teile, da diese meist nur bei der Produktion in chemischen Prozessen Einsatz finden. Unter der Kategorie der Temperiereinheiten, genannt `TemperingPart`, fallen Heiz- oder Kühlsysteme sowie Isolationen. Unter der Kategorie der schwingenden Teile, genannt `VibratingPart`, fallen Schwingungserreger sowie Schwingungsisolierungen oder -dämpfungen. Bestrahleinrichtungen, genannt `RadiationUnit`, finden bei Prüfmaschinen in Form von Lichtbestrahlung, Röntgenbestrahlung oder Gammabestrahlung Einsatz. Die Eingabeeinheiten, genannt `InputUnit`, umfassen Maus, Tastatur oder andere Schaltflächen. Als Steuerungseinheiten, genannt `ControllerUnit`, werden in der Regel Industrie-PCs und SPSen eingesetzt. Anzeige- und Signaleinheiten, genannt `DisplaySignalUnit`, umfassen Bildschirme, Sirenen oder Signallichter wie Ampeln. Die Energiequellen können elektrischer, pneumatischer oder hydraulischer Natur sein. Ähnlich wie bei normalen Steuereinheiten umfassen Sicherheitssteuerungen, genannt `SafetyControllerUnit`, in der Regel SPSen, jedoch keine Industrie-PCs, da diese überwiegend nicht ausreichend sicher sind. Zuletzt fallen unter Sicherheitseinrichtungen, genannt `ProtectionUnit`, Sicherheitsfunktionalitäten wie Nothalt, Schutz durch Positionserfassung, Lichtgitter und Sicherheitszäune sowie Temperaturüberwachungen.

Die entwickelten Teile sind für Prüfmaschinen entwickelt, aber nicht auf die Anwendung dieser beschränkt. Für den Fall, dass diese Unterteilung bei der Entwicklung einer Prüfmaschine nicht ausreichend ist, werden spezielle Klassen von Prüfmaschinenteilen aufgebaut, die von Entwicklern genutzt werden können, um Teile einer Prüfmaschine im gemeinsamen Framework zu identifizieren.

3.2.7 Mechanical View

Für den Maschinenbauer liegt der Fokus beim mechanisch-konstruktiven Aufbau auf der sorgfältigen Auswahl einzelner Bestandteile der Maschine und deren Verhalten, um eine hohe Funktionalität und eine lange Lebensdauer zu gewährleisten. Um diese Anforderungen zu erfüllen, wird die bisherige Struktur aus `MachineAssemblyStructure` weitergenutzt und verfeinert. Die Bibliothek wird im Folgenden durch Abbildung 3.12 beschrieben. Hierbei werden jedem mechanischen Element ein Gewicht, eine Position, ein Volumen, Dimensionen und eine Orientierung zugeordnet. Dies ist essenziell für die Modellierung im Zusammenhang zur funktionalen Sicherheit, dem späteren Aufbau der CAD-Modelle und der Fertigung der Maschine.

Um diese Eigenschaften umzusetzen, wird die ISO 80000 SysML-Bibliothek genutzt und erweitert. Die Bibliothek bietet bereits die Datentypen `mass`, `position vector`, `volume` und `plane angle`. Um auch dreidimensionale Winkel darstellen zu können, wird der Datentyp `plane angle` erweitert, indem er in einem Datentyp mit α, β und γ eingesetzt wird

Darüber hinaus ist es wichtig, dass jedes mechanische Element auch Eigenschaften besitzt, die die Beschaffenheit der eingesetzten Materialien, wie Festigkeit, Elastizität, Dichte und Härte, beschreiben. Diese Informationen sind nicht nur wichtig für die Modellierung, sondern auch für die Fertigung und Wartung der Maschine.

Um den Zusammenhang und das Verhalten zwischen mechanischen Elementen in IBD zu beschreiben, soll eine einfache SysML-Connector-Beziehung Anwendung finden. Abhängigkeiten zu Kontaktoberflächen und Verbindungstypen können dann im Nachhinein im CAD-Modell konkretisiert werden.

Dieser Formalismus dient als Grundlage für den Aufbau der mechanischen Sichtweise in MMBSEFE. Erläuterungen zu den einzelnen Elementen und die Darstellung des Profils finden sich für eine übersichtlichere Darstellung unter Anhang A.9.5.

Mechanical-Bibliothek

Abbildung 3.12 gibt das allgemeine Konzept des mechanisch-konstruktiven Aufbaus wieder.

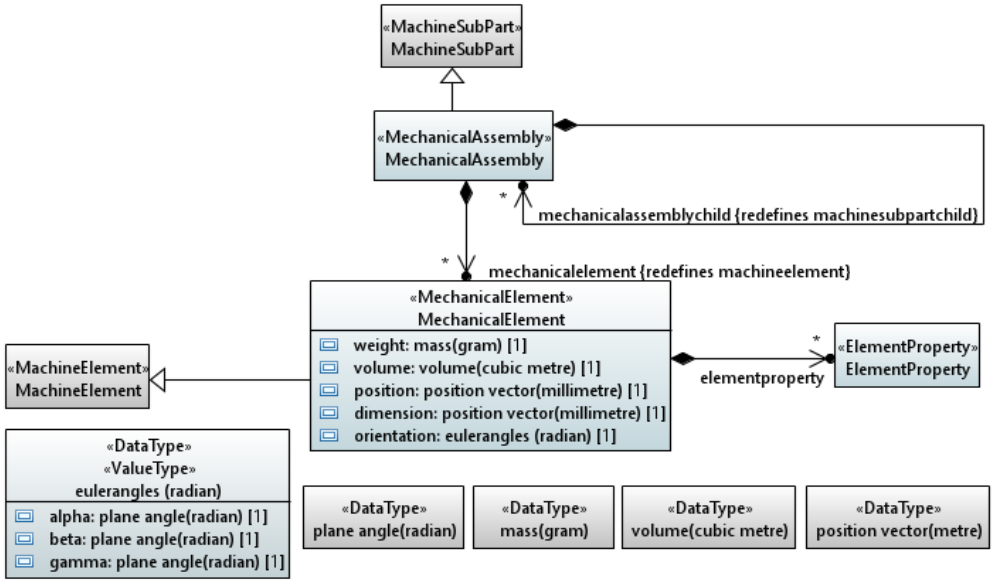


Abbildung 3.12: MechanicalView-Bibliothek

`MechanicalAssembly` spezialisiert die Klasse `MachineSubPart` und `MechanicalElement` die Klasse `MachineElement`. `MechanicalAssembly` definiert die Kompositionen für untergeordnete mechanische Konstruktionen und Elemente neu.

`MechanicalElement` definiert eine Komposition zu `ElementProperty`, um zusätzliche Materialeigenschaften festzulegen. Hierbei können bei der späteren konkreten Modellierung einer Maschine zusätzliche Datentypen aus ISO 80000 genutzt werden, die die Eigenschaften korrekt beschreiben.

3.2.8 Electrical View

Für den Elektroingenieur liegt der Fokus beim elektrotechnischen Design auf der Abbildung der Vernetzung zwischen elektrotechnischen Elementen. Um diese Anforderungen zu erfüllen, wird ebenso hier die bisherige Struktur aus `MachineAssemblyStructure` weitergenutzt und verfeinert.

Jedes elektrotechnische Element ist einem mechanischen Element zugeordnet. Elektrische Elemente sollen nicht weiterverfolgt werden, da diese zu allgemein und vielfältig sind, um einheitliche Regeln aufzuerlegen. So können relevante spezifische Elemente wie Widerstände, Transistoren oder Spulen, sofern für die Modellierung relevant, direkt aus elektrotechnischen Elementen erzeugt werden. Ähnliches gilt für die meisten elektronischen Elemente, wobei Sensoren und Aktoren eine Ausnahme bilden. Diese spielen eine spezielle Rolle für die funktionale Sicherheit, beispielsweise bei der Entwicklung von Sicherheitsfunktionen, weshalb sie in den Modellen als diese eindeutig gekennzeichnet werden sollen. Zusätzlich kann es interessant sein,

wo Sensoren angebracht werden. Programmierbare elektronische Elemente sollen durch einen Controller abgebildet werden können.

Um den Zusammenhang und das Verhalten zwischen elektrotechnischen Elementen beschreiben zu können, sollen über Item-Flow verbundene SysML-Ports Anwendung finden. Hierbei wird angenommen, dass jeder Sensor, jeder Controller und jeder Aktor eine Spannungsversorgung vom Typ `source voltage` aus ISO 80000 besitzt. Zusätzlich besitzen Sensoren mindestens einen Port, um gemessene Größen in Form von elektrischen Signalen für die elektrische Weiterverarbeitung bereitzustellen und Aktoren mindestens einen Port, um Signale zu empfangen. Hierbei können natürlich auch weitere Ports, beispielsweise zur Konfiguration, Einsatz finden, die sich aber nicht so einfach formalisieren lassen. Controller besitzen neben ihrer Spannungsversorgung Eingänge und Ausgänge. Welchen Datentyp die Ports besitzen, ist abhängig der übermittelten Signale, jedoch kann über Item-Flow im IBD die Kompatibilität der Ports sichergestellt werden.

Dieser Formalismus bildet die Grundlage für die elektrotechnische Sichtweise in MMBSEFE. Die Bibliothek wird in Abbildung 3.13 dargestellt. Weitere Informationen zu den einzelnen Elementen und dem Profil finden Sie für eine übersichtlichere Darstellung unter Anhang A.9.6.

Electrical-Bibliothek

Abbildung 3.13 gibt das allgemeine Konzept des elektrotechnischen Aufbaus wieder.

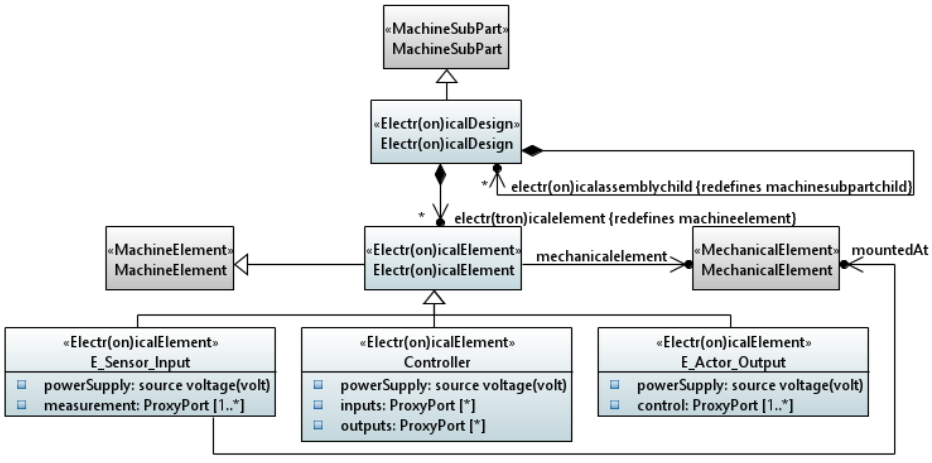


Abbildung 3.13: ElectricalView-Bibliothek

`Electr(on)icalDesign` spezialisiert die Klasse `MachineSubPart` und `Electr(on)icalElement` die Klasse `MachineElement`. `Electr(on)icalDesign` definiert die Kompositionen für untergeordnete elektrische und elektronische Konstruktionen und Elemente neu.

Jedes `Electr(on)icalElement` besitzt eine Assoziation zu einem `MechanicalElement`, um den Zusammenhang der Modelle zu beschreiben. Jeder Sensor besitzt zusätzlich eine Assoziation zu einem `MechanicalElement`, um festzustellen, wo genau dieser angebracht ist, da beispielsweise bei einem Digitalmultimeter die Messdatenerfassung im Gerät und die Anbringung der Messspitze nicht übereinstimmen müssen.

3.2.9 Hydraulic View und Pneumatic View

Bei Hydraulik und Pneumatik liegt der Fokus jeweils auf der Abbildbarkeit vernetzter hydraulischer und pneumatischer Elemente. Um diese Anforderungen zu erfüllen, wird die bisherige Struktur aus `MachineAssemblyStructure` weitergenutzt und verfeinert. Diese werden aufgrund einer identischen Modellierung im Weiteren zusammengefasst behandelt.

Jedes hydraulische und pneumatische Element ist einem mechanischen Element zugeordnet. Elemente wie einfache Ventile, Leitungen und Zylinder sollen nicht weiterverfolgt werden, da diese keine relevante Rolle für die gemeinsame Entwicklung bieten und relevante Eigenschaften bereits durch die Mechanik geklärt werden. Eine Sonderrolle spielen hydraulische und pneumatische Sensoren und Aktoren, die abhängig des elektrotechnischen Designs mechanischen Elementen zugeordnet werden können.

Dieser Formalismus bildet die Basis für die hydraulische und pneumatische Perspektive in MMBSEFE. Die Bibliotheken werden in Abbildung 3.14 und Abbildung 3.15 dargestellt. Weitere Informationen zu den einzelnen Elementen und die Darstellung der Profile finden sich für eine übersichtlichere Darstellung der Arbeit unter Anhang A.9.7.

Hydraulic-Bibliothek

Abbildung 3.14 gibt das allgemeine Konzept für Hydraulik wieder.

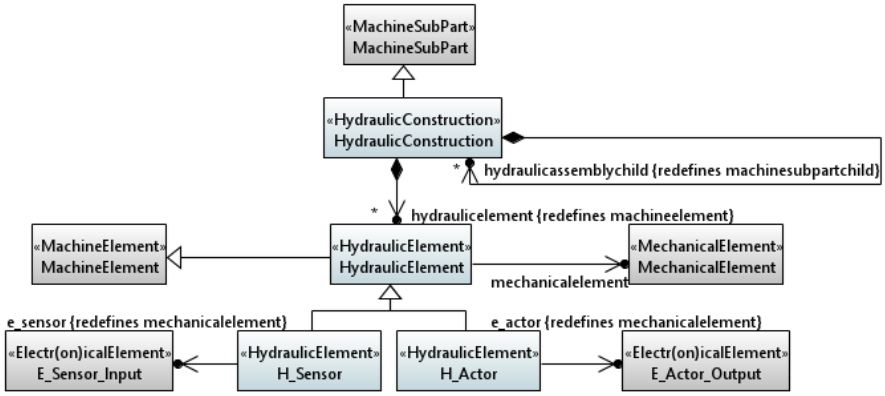


Abbildung 3.14: HydraulicView-Bibliothek

`HydraulicConstruction` spezialisiert die Klasse `MachineSubPart` und `HydraulicElement` die Klasse `MachineElement`. `HydraulicConstruction` definiert die Kompositionen für untergeordnete mechanische Konstruktionen und Elemente neu.

Jedes `HydraulicElement` besitzt eine Assoziation zu einem `MechanicalElement`, um den Zusammenhang der Modelle zu beschreiben, es sei denn, es handelt sich um einen hydraulischen Sensor oder Aktor. In diesem Fall wird die Assoziation neu definiert und einem `E_Sensor_Input` bzw. `E_Actor_Output` zugeordnet.

Pneumatic-Bibliothek

Abbildung 3.15 gibt, analog zur Hydraulik, das allgemeine Konzept für Pneumatik wieder.

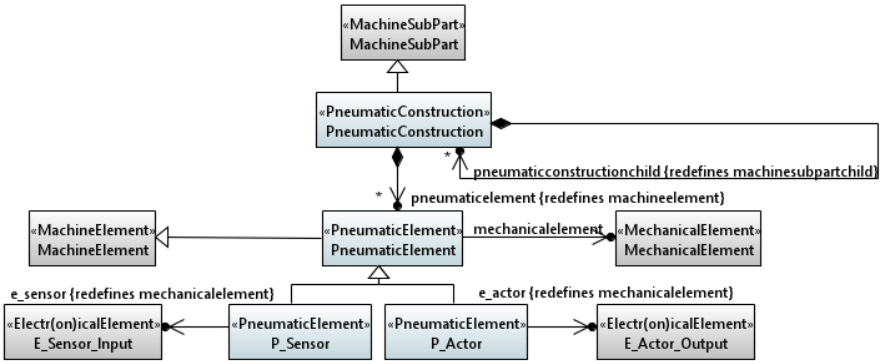


Abbildung 3.15: PneumaticView-Bibliothek

`PneumaticConstruction` spezialisiert die Klasse `MachineSubPart` und `PneumaticElement` die Klasse `MachineElement`. `PneumaticConstruction` definiert die Kompositionen für untergeordnete mechanische Konstruktionen und Elemente neu.

Jedes `PneumaticElement` besitzt, wie auch ein `HydraulicElement`, eine Assoziation zu einem `MechanicalElement`, um den Zusammenhang der Modelle zu beschreiben. Falls es sich um einen pneumatischen Sensor oder Aktor handelt, wird die Assoziation neu definiert und einem `E_Sensor_Input` bzw. `E_Actor_Output` zugeordnet.

3.2.10 Informatic View

Die informationstechnische Sichtweise für den Informatiker bezieht sich auf die Art und Weise, wie Software in einem System modelliert werden kann. UML ist eine Modellierungssprache, die speziell für Softwareanwendungen entwickelt wurde und ermöglicht somit bereits eine informationstechnische Sichtweise.

Eine Möglichkeit, Software eines Systems zu modellieren, ist die Verwendung von Komponenten. Diese können bereits rekursiv ineinander verschachtelt werden und können somit die

verschiedenen Schichten einer Softwarearchitektur repräsentieren, um die informationstechnische Sichtweise in die `MachineAssemblyStructure` zu integrieren.

Jedes Software-Element wird einem Controller zugeordnet, um sicherzustellen, dass jede Softwarekomponente innerhalb MMBSEFE einer elektrotechnischen Sichtweise zugeordnet ist und die Schnittstellen zwischen elektrotechnischen Design und Software-Design konform sind. Durch diese Zuordnung kann jedem Software-Element eine Hardware zugeordnet werden. Dies ist besonders wichtig bei der Softwareentwicklung, da es eine grundlegende Voraussetzung für das Testen, Debuggen und Verifizieren von Software ist.

Dieser Formalismus dient als Grundlage für die informationstechnische Sichtweise in MMBSEFE. Erläuterungen zu den einzelnen Elementen und die Darstellung des Profils finden sich auch hier im Anhang unter A.9.9.

Informatic-Bibliothek

Abbildung 3.16 gibt das allgemeine Konzept des informationstechnischen Aufbaus wieder.

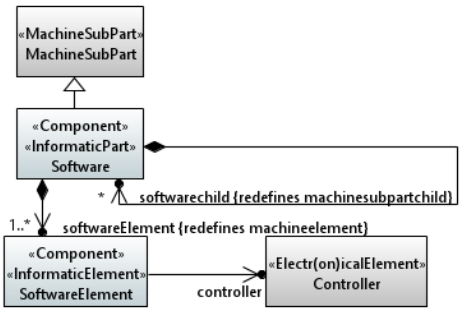


Abbildung 3.16: InformaticView-Bibliothek

`Software` spezialisiert die Klasse `MachineSubPart` und `SoftwareElement` die Klasse `MachineElement`. `Software` definiert die Kompositionen für untergeordnete Softwareteile und -elemente neu.

Jedes `SoftwareElement` besitzt eine Assoziation zu einem `Controller`, um den Zusammenhang innerhalb der Modelle zu beschreiben,

3.2.11 ISO 12100 & ISO 13849

Die sicherheitsorientierte Perspektive bezieht sich auf die Verbindung von Risikobeurteilung und Risikominderung auf Grundlage der Konzepte zur Sicherheit von Maschinen. Hierbei wird RAAML, wie in Abbildung 3.7 dargestellt, erweitert. Das ISO12100 & ISO 13849-Paket beinhaltet Elemente, die die sicherheitsbezogene Anforderungsspezifikation im Bereich der funktionalen Sicherheit von Maschinen unterstützen, wie sie in den Standards ISO 12100 und ISO

13849 definiert sind. Durch die bereits verwobene Anwendung der Standards, wie unter Abschnitt 2.1.3 in Abbildung 2.3 und Abbildung 2.4 gezeigt, bietet sich ein gemeinsames Metamodell an. Das Paket definiert die Konzepte aus RAAML Core- und General-Paket neu, um fehlende domänenspezifische Begriffe abzubilden und die Modellierung für den Sicherheitsingenieur in der Maschinenindustrie zu erleichtern.

Das ISO12100 & ISO 13849-Paket ermöglicht die Modellierung von Risikographen, die typischerweise zur Identifizierung und Abschätzung von Gefährdungen verwendet werden. Als Grundlage für eine mögliche Auslegung der Integration in die Maschinenbaudomäne dient das RAAML-Paket ISO 26262 für die Automobilindustrie.

Die Grenzen der Maschine legen die grundlegenden Anforderungen an eine Maschine fest und sind der Ausgangspunkt jeder Risikobeurteilung. Um die Risikobeurteilung darzustellen, muss das Konzept der `Situation` aus Core übernommen und für Gefährdungssituationen und Gefährdungsereignisse spezialisiert werden. Dadurch können identifizierte Gefährdungen direkt Gefährdungssituationen zugeordnet werden. Um eine direkte Abhängigkeit zwischen entwickelten Teilen einer Maschine und den davon ausgehenden Gefährdungen zu schaffen, werden `MachinePart`-Elemente direkt mit der Gefährdung assoziiert. Jede Gefährdungssituation besitzt dann ein auslösendes Gefährdungsereignis und assoziierte Aktoren, die der Situation ausgesetzt sind. Ein Gefährdungsereignis kann möglicherweise nur in bestimmten Betriebsmodi auftreten, was über eine Erweiterung von `UseCase` möglich wird. Als letztes Element in der Kette besitzt ein Gefährdungsereignis immer mindestens eine Ursache, die das Ereignis auslöst und in verschiedenen Lebensphasen auftreten kann. Abbildung 3.17 stellt diese Beziehungen vereinfacht über eine Elementstruktur dar.

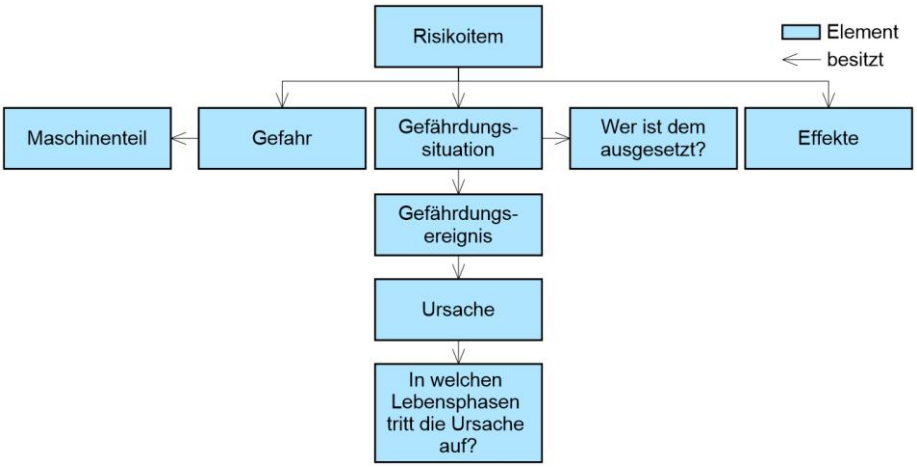


Abbildung 3.17: Bestimmung eines Risikoitems

Ein spezialisiertes Risikoitem aus einem Block wird verwendet, um die unter einer Gefährdung auftretenden Gefährdungssituationen und Effekte zusammenzufassen. Es bietet die

Möglichkeit, die Risikoprioritätszahl zu berechnen und risikomindernde Maßnahmen zu bestimmen. Hierfür müssen die Konzepte `Hazard`, `DysfunctionalEvent`, `Scenario`, `AbstractEvent`, `AbstractCause` und `AbstractEffect` aus der General Concepts-Bibliothek erweitert werden.

Aus risikomindernden Maßnahmen können verschiedene Anforderungen abgeleitet werden, darunter technische Anforderungen mit Hardware- und Softwareanforderungen, funktionale Sicherheitsanforderungen, die aus technischen Anforderungen bestehen, sowie Anforderungen für Benutzerinformationen. Hierfür wird das Konzept der Anforderungsdefinition über `AbstractRequirements` erweitert. Funktionale Sicherheitsanforderungen repräsentieren die Anforderungen an Sicherheitsfunktionen, die eine Zerlegung des PL_r erfordern. Zusätzlich sollen die Begriffe Fehlerausschluss und permanente Störung auf Basis von `Fault` und CCF, gefährliche Ausfälle und systematische Ausfälle auf Basis von `FailureMode` spezialisiert werden, um die Einschätzung und Kategorisierung von Ausfällen und die Bestimmung der notwendigen Risikominderung zu erleichtern.

Für die Risikoabschätzung, -bewertung und -minderung legt die ISO 13849 verschiedene Definitionen für Value-Types nahe, die für die beschriebenen Elemente und für die SBBM als Typdefinitionen dienen können. Diese Value-Types umfassen:

- Möglichkeit zur Vermeidung/Begrenzung P1 und P2 in der Gefährdungssituation,
- Häufigkeit/Dauer der Exposition F1 und F2 im Gefährdungsereignis,
- Eintrittswahrscheinlichkeit eines Ereignisses O1 und O2 in der Ursache,
- Schwere der Verletzung S1 und S2 im Effekt,
- Maßnahmen gegen CCF,
- PL a bis PL e,
- Kategorien B bis 4 und
- Lebensphasen Transport, Montage, Installation, in Betrieb nehmen, Betrieb, Einlernen, Instandhaltung, Fehlersuche und Beseitigung, Reinigung, Wartung, außer Betrieb nehmen, Demontage und Entsorgung in der Ursache,

die teilweise die allgemeinen Notationen aus RAAML neu definieren.

Dieser Formalismus bildet die Grundlage für die Sichtweise des Sicherheitsingenieurs in MMBSEFE. Im Folgenden wird die Bibliothek und das Profil durch Abbildung 3.18 und Abbildung 3.19 beschrieben. Detaillierte Erläuterungen zu den einzelnen Elementen und Stereotypen finden sich in Anhang A.9.10.

ISO 12100 & ISO 13849-Bibliothek

Abbildung 3.18 gibt das allgemeine Konzept der Risikobeurteilung im Maschinenbau wieder. Für eine bessere Lesbarkeit wird diese im Querformat dargestellt.

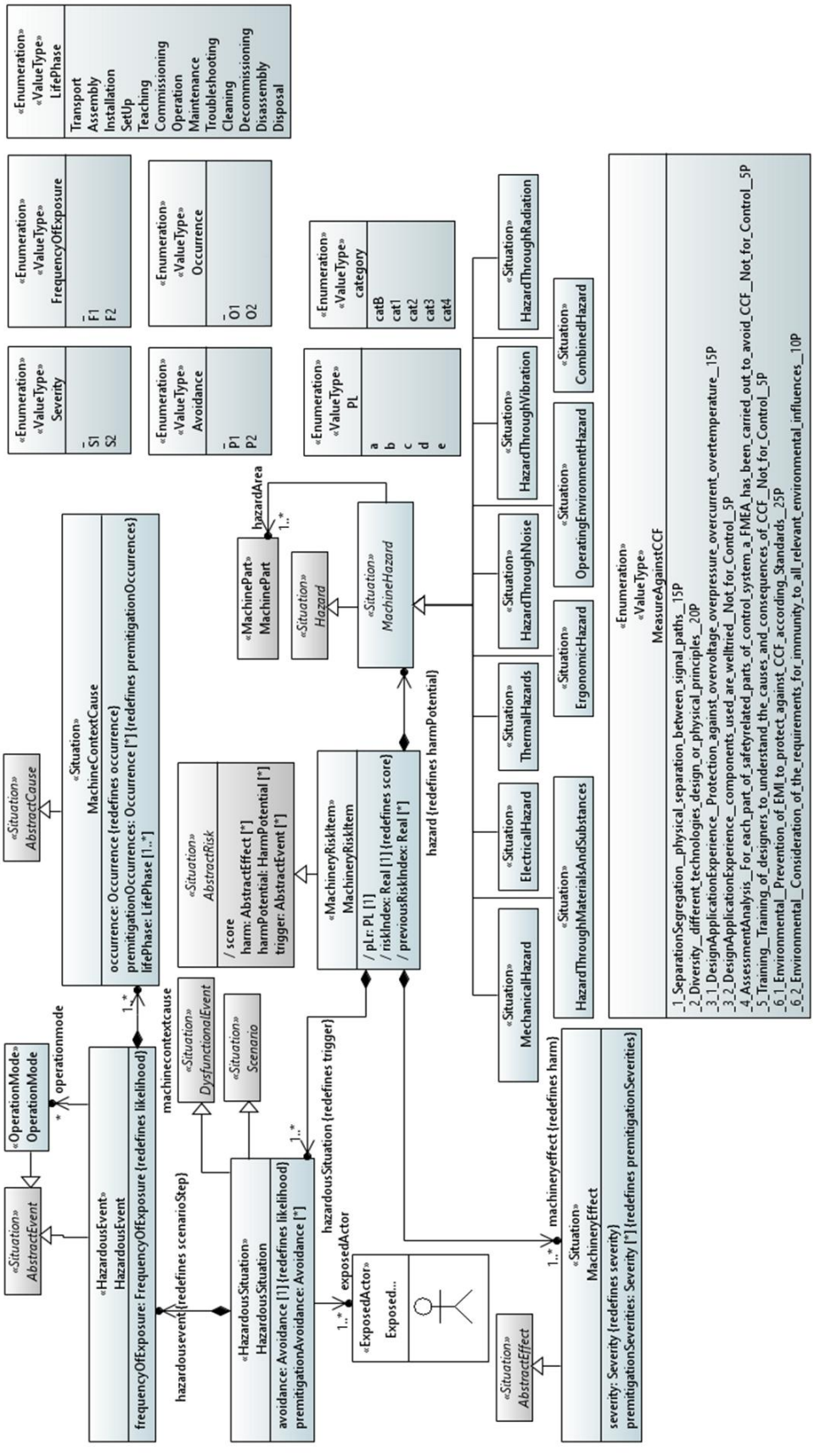


Abbildung 3.18: ISO12100&ISO13849-Bibliothek

Die Klasse MachineryRiskItem ist eine Spezialisierung der abstrakten Klasse AbstractRisk und enthält Kompositionen zu MachineHazard, die von Hazard abgeleitet ist, MachineryEffect, die von AbstractEffect abgeleitet ist, und HazardousSituation, die von DysfunctionalEvent und Scenario abgeleitet ist. MachineryRiskItem definiert die Attribute riskIndex für score, hazard für harmPotential, machineryEffect für harm und hazardousSituation für trigger neu. Zusätzlich werden die Attribute PLr und previousRiskIndex definiert. MachineHazard ist eine Generalisierung der verschiedenen Gefährdungsgruppen MechanicalHazard, ElectricalHazard, ThermalHazard, HazardThroughNoise, HazardThroughVibration, HazardThroughRadiation, HazardThroughMaterialAndSubstances, ErgonomicHazard, OperatingEnvironmentHazard und CombinedHazard. Es besitzt eine Assoziation zu MachinePart mit mindestens einem zugeordneten Maschinenteil. HazardousSituation besitzt eine Assoziation zu mindestens einem ExposedActor und eine Komposition zu einem HazardousEvent. Außerdem werden die Attribute avoidance aus likelihood und zusätzlich premitigationAvoidance mit dem ValueType Avoidance definiert. HazardousEvent besitzt eine Assoziation zu OperationMode, kann aber auch ohne existieren. Es enthält eine Komposition zu MachineContextCause mit mindestens einer Ursache. Zusätzlich enthält es das Attribut frequencyOfExposure, das auch likelihood neu definiert. MachineContextCause besitzt die Attribute occurrence und premitigationOccurrence, die mit dem ValueType Occurrence neu definiert sind. Es enthält auch das zusätzliche Attribut lifePhase mit den Lebensphasen, in denen die Ursache auftritt. Zuletzt besitzt MachineryEffect die Attribute severity und premitigationSeverity, die mit dem ValueType Severity neu definiert sind.

ISO 12100 & ISO 13849-Profil

Abbildung 3.19 gibt die Stereotypen der Risikobeurteilung im Maschinenbau wieder.

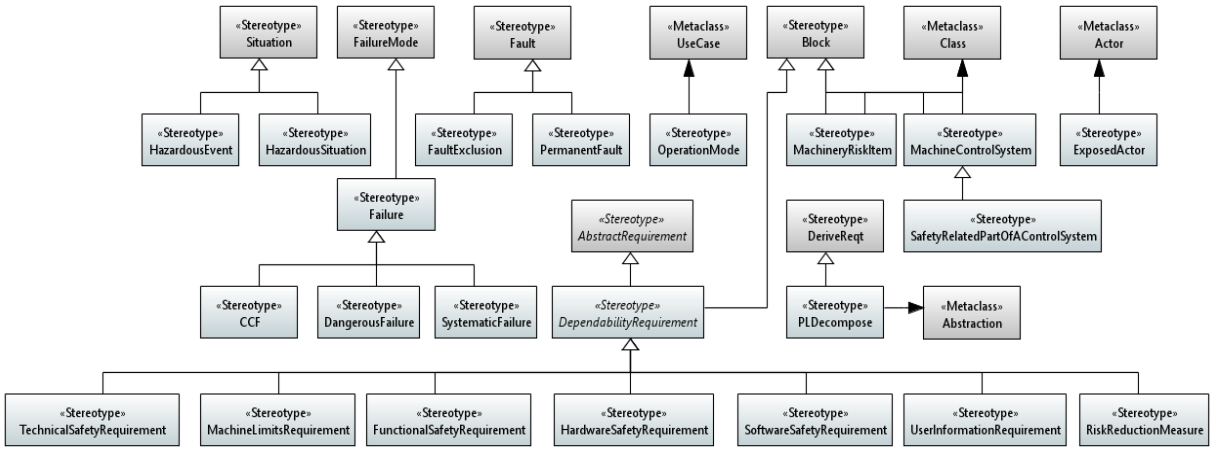


Abbildung 3.19: ISO12100&ISO13849-Profil

Der Stereotyp `DependabilityRequirement` ist eine Spezialisierung des abstrakten Stereotyps `AbstractRequirement` und generalisiert Anforderungen in MMBSEFE, einschließlich `MachineLimitRequirement`, `RiskReductionMeasure`, `FunctionalSafetyRequirement`, `TechnicalSafetyRequirement`, `HardwareRequirement`, `SoftwareRequirement` und `UserInformationRequirement`. `PLDecompose` wird aus `DeriveReqt` spezialisiert und `Abstraction` erweitert. Der Stereotyp `Failure` spezialisiert `FailureMode` und bildet eine Generalisierung für `CCF`, `DangerousFailure` und `SystematicFailure`. Diese Stereotypen können dazu beitragen, verschiedene Fehlertypen zu spezifizieren, müssen jedoch nicht vom Sicherheitsingenieur ausgeführt werden. Ähnliches gilt für `FaultExclusion` und `PermanentFault`, die aus `Fault` spezialisiert sind, sowie für `MachineControlSystem`, das aus `Block` spezialisiert ist und zu `SafetyRelatedPartOfAControlSystem` erweitert wird, da diese bereits aus elektro- und informationstechnischer Sicht hervorgehen können. Die Stereotypen `HazardousEvent` und `HazardousSituation` spezialisieren sich aus `Situation`. `MachineryRiskItem` spezialisiert sich aus `Block` und erweitert `Class`. `OperationMode` ist eine Erweiterung von `UseCase`, während `ExposedActor` eine Erweiterung von `Actor` ist.

3.2.12 Inhaltlicher Beitrag

In diesem Abschnitt wird der inhaltliche Beitrag der vorgestellten Modellierung und Gestaltung für die Forschungsaufgaben FA1-TB des Typs Theoriebildung im Bereich der maschinenbauspezifischen System-, Software- und sicherheitsbezogenen Integration zusammengefasst.

Die Anwendungsfälle, die aus Perspektive der verschiedenen technischen Stakeholder für eine typische Maschinenentwicklung erforderlich sind, wurden in Abschnitt 3.2.2 definiert und zueinander in Zusammenhang gebracht. Hierbei wurden 32 Anwendungsfälle definiert, um die

Anforderungen, die durch die Forschungsfragen vorgegeben wurden, zu erfüllen. Die in Abschnitt 3.2.1 definierte Struktur zur Modellierung einer gemeinsamen Sprache wird in Abschnitt 3.2.3 für das gemeinsame Informationsmodell und unter Bezug des State of the Art bekannter Ansätze aus Abschnitt 2.1.3 und 2.1.4 eingeführt und legt den OMG RAAML-Ansatz, das Vorgehen von Brahmi et al. [32] und die Erweiterung durch Standard-Bibliotheken als Grundlage für die Modellierung fest. Zusätzlich entsteht die konkrete Integrationsarchitektur der verschiedenen technischen Stakeholder, um diese im Kontext von MMBSEFE und der enthaltenen Stakeholder-spezifischen Konzepte in Profilen und Bibliotheken zu modellieren.

Aufbauend auf der grundlegenden Struktur einer Konstruktion in Abschnitt 3.2.4 bildet sich der Aufbau von Maschinen in Abschnitt 3.2.5 und spezialisiert daraus den Aufbau von Prüfmaschinen in Abschnitt 3.2.6. Die Grundlage für die abstrakte Abbildung der Modelle in Abhängigkeit zueinander, wie durch die Anwendungsfälle gefordert, findet sich für die Mechanik in Abschnitt 3.2.7, für die Elektronik in Abschnitt 3.2.8 und für die Hydraulik und Pneumatik in Abschnitt 3.2.9 sowie für die konkrete Abbildung der informationstechnischen Modellierung in Abschnitt 3.2.10. So entsteht ein standardisierter Zugang der Stakeholder-spezifischen Modelle für andere Stakeholder, was das Verständnis für das Gesamtsystem verbessert. Zusätzlich entstehen durch einen generalisierten Ansatz Erweiterungsmöglichkeiten von MMBSEFE, um die Integration zukünftiger Entwicklungen zu unterstützen. Beispielsweise können mechanische Modelle durch Kontaktflächen und Verbindungsarten oder elektrotechnische Modelle durch konkrete Bauteile und Leitungseigenschaften erweitert werden. Die sicherheitskritische Modellierung in Abschnitt 3.2.11 integriert und erweitert RAAML und MMBSEFE für den Sicherheitsingenieur, um die Risikobeurteilung darauf abzubilden und Vorgaben für die Risikominderung zu geben, die wiederum von den anderen Stakeholdern bearbeitet werden müssen. Hierbei liegt der Vorteil in der direkten Abhängigkeit der Gefährdungen in Risikoitems, mit den entsprechenden modellierten oder zu modellierenden Maschinenteilen. Die modellierten Teile geben dem Sicherheitsingenieur eine sehr genaue Grundlage zur Risikobeurteilung und die zu modellierenden Teile aus den Sicherheitsanforderungen zur Risikominderung den technischen Stakeholdern genaue Vorgaben zur Umsetzung und Kommunikation in den Modellen.

Um die Risikominderung nachzuweisen, müssen Sicherheitsanalysemethoden wie FTA, DFT oder SBBM aus dem Maschinenbau angewendet werden. In den folgenden Abschnitten werden Konzepte und Modelle vorgestellt, die eine Abschätzung des Nachweises auf der Grundlage von Wahrscheinlichkeiten und anderen Faktoren ermöglichen.

3.3 Integration der Analysemethoden

Dieser Abschnitt zeigt die Modellierung von Sicherheitsanalysemethoden des Maschinenbaus auf und behandelt damit FA2-TB. Hierfür werden die unter Abschnitt 2.2 beschriebenen Analysemethoden in Unterabschnitt 3.3.1, in Bezug zur Modellierung, genauer untersucht und verglichen, um grundlegende Präferenzen darzustellen und eine Schlussfolgerung zur Modellierung typischer maschinenbauspezifischer Analysemethoden zu ziehen. Im zweiten und dritten Unterabschnitt 3.3.2 und 3.3.3 werden jeweils die Grundlagen zur Modellierung der Analysen DFT und SBBM zunächst verbal beschrieben und folgend in Metamodellen spezifiziert. Unterabschnitt 3.3.4 fasst am Ende den inhaltlichen Beitrag zur Arbeit kurz zusammen.

3.3.1 Grundlagen zur Modellierung

Im vorliegenden Abschnitt werden die relevanten Methoden zur Sicherheitsanalyse für die Maschinenindustrie unter Bezugnahme auf Abschnitt 2.2.1 und 2.2.2 genauer untersucht und verglichen. Dadurch sollen Grundlagen für die Erweiterung von MMBSEFE zur Integration weiterer deduktiver Analysemethoden in RAAML geschaffen werden.

Tabelle 3.1 gibt einen Überblick über die in Abschnitt 2.1.3 beschriebenen verwandten Arbeiten und setzt diese in Bezug zu den in der Domäne Maschinenindustrie üblicherweise eingesetzten Analysenmethoden. Hierbei kann festgestellt werden, dass bisher keine Profile für DFT, Risikograph und SBBM existieren. Es soll sich auf die spezifischen Analysen Risikograph und SBBM konzentriert werden, wobei der Risikograph bereits durch Abschnitt 3.2.11 abgebildet wurde. DFTs können ebenso für die Beurteilung einer Risikominderung einen sinnvollen Beitrag leisten und dienen daher zusätzlich als Vergleich zur Bewertung mit SBBM.

Veröffentlichungen	FME (C/D) A	FTA	DFT	Risi-ko-graph	SBBM	Realisiert über	Grund-legende Analy-sen	Domänen-spezifi-sche Ana-lysen
Joshi et al. '07 [209]		X				AADL	X	
Xiang et al. '11 [210]		X	X			RCM	X	X
David et al. '10 [211]	X	X				Alta-Rica	X	X
Hecht et al. '15 [69]	X					Alta-Rica	X	
Yakymets et al. '18 [70]	X	X				Alta-Rica	X	X
Helle '12 [75]						SysML	X	
Thramboulidis et al. '10 [76]						Profile		
Mhenni et al. '14 [68]	X	X				Profile	X	
Biggs et al. '16 [77]	X	X				Profile	X	X
Baklouti et al. '19 [212]	X	X	X			Profile	X	X
Biggs et al. '19 [48]	X	X				Profile	X	
Berres et al. '21 [49]		X				Profile	X	X

Tabelle 3.1: Verwandte Arbeiten typischen Sicherheitsanalysemethode im Maschinenbau

Die Betrachtung von Tabelle 3.1 zeigt zusätzlich, dass im Vergleich zu den OMG-Veröffentlichungen von Biggs et al. [48] und Berres et al. [49], Baklouti et al. [212] bereits einen anfänglichen Beitrag zur Modellierung von DFTs geleistet hat. Dieser Ansatz ist allerdings relativ grob und basiert nicht auf Standardbegriffen der funktionalen Sicherheit, was die Vernetzung zu anderen Analysen verhindert. Für die Risikographmethode und die SBBM fehlt im generellen ein generalisierter Ansatz, der auf Standardbegriffen der funktionalen Sicherheit basiert, wie bereits in Abschnitt 3.2.2 beschrieben. RAAML bietet daher durch die bereits enthaltenen Analysen FTA, FMEA, HAZOP, STPA und GSN sowie den Core- und General-Paketen als Ausgangspunkt die beste Wahl.

3.3.2 Metamodellumsetzung für DFTs

Zur Entwicklung eines Architekturkonzeptes für DFTs, mit dem Ziel der einfachen Einbindung in verschiedene Domänen für Sicherheitsingenieure, wurde das RAAML-FTA-Paket mit seinem Profil und der Bibliothek erweitert. Die Erweiterungen führen die für DFTs typischen Abhängigkeitsmuster, wie z.B. Ersatzteilmanagement, Redundanzen und zeitliche Abhängigkeiten, ein, die für mehr Praktikabilität in Bereichen wie der Maschinenindustrie genutzt werden können.

Die bereits in Abbildung 3.7 beschriebene Einordnung in MMBSEFE zeigt die Erweiterungen in Bezug auf die bestehende OMG RAAML-Paketstruktur (vgl. Abbildung A.9 in Anhang A.7.2). Innerhalb des bestehenden Pakets `Methods` wird das Paket `FTA` für DFTs erweitert. Das DFT-Profil definiert die Stereotypen der dynamischen Gates, Events und Fehlerbäume, die sich auf das FTA-Profil beziehen. Die DFT-Bibliothek beschreibt das Verhalten und die Parameter als Erweiterung auf Basis der FTA-Bibliothek.

Das DFT-Paket soll ebenso separat, ohne Abhängigkeiten zu MMBSEFE, verwendet werden können, um den Ansatz für andere Domänen abstrahierbar zu halten und dynamische Gatter, Ereignisse und Fehlerbäume in anderen geeigneten domänenspezifischen Modellen verwenden zu können. Dies wird im Weiteren speziell gekennzeichnet und benannt.

Dementsprechend muss basierend auf Abschnitt 2.2.1 für DFTsBasic der Fehlerbaum erweitert werden, um neben der maximalen Ausfallwahrscheinlichkeit, bei der der Fehlerbaum fehlschlägt, die Dauer anzugeben, bis zu der der Fehlerbaum nicht fehlschlagen darf. Da neben der Ausfallwahrscheinlichkeit nun auch die Ausfallzeit eine Rolle spielt, kann im Fehlerbaum ebenso MTTF berechnet werden, was sich im dynamischen Top-Ereignis widerspiegeln muss. Da in der FTA-Bibliothek bereits AND, OR und SEQ Gates existieren, können diese als Grund-

lage für die Erweiterung in DFTs genutzt werden, wobei das durch die OMG bereits vorgesehene Attribut zur Priorität in den Events genutzt werden kann, um die Priorisierung der Eingänge durchzuführen. Für das FDEP-Gate existiert nur noch ein Eventeingang, der vom auslösenden Ereignis kommt. Abgängige Events müssen zusätzlich definiert werden und der TargetEvent-Ausgang dient nur noch als Dummy. Für das SPARE-Event teilen sich die Eventeingänge in einen Eingang für die primäre Komponente und mehreren Eingängen für die Ersatzkomponenten. Zusätzlich muss ein Attribut über eine ValueType-Definition den Typ regeln, inwieweit es sich um ein CSP, WSP oder HSP handelt. Als Letztes werden die Basisereignisse für dynamische Basisereignisse erweitert, die sich in nicht reparierbare, reparierbare und latente Ereignisse teilen. Hierbei muss jeweils ein Ruhefaktor und Abdeckungsfaktor enthalten sein, für den Fall, dass es sich um eine Ersatzkomponente eines SPARE-Gates handelt und die Wahrscheinlichkeit, dass der Ausfall der Komponente zum Ausfall des Systems führt, ungleich eins ist. Für jedes dynamische Basisereignis wird ein Maschinenelement aus MMBSEFE assoziiert. Sollte die Analyse ohne das Framework eingesetzt werden, kann dieses entweder außen vor gelassen werden oder durch ein Element eines anderen domänenspezifischen Frameworks ersetzt werden. Jedes reparierbare Basisereignis besitzt zusätzlich eine Reparaturrate und jedes latente Basisereignis ein Inspektionsintervall. Um die Verteilungsfunktionen spezifizieren zu können, wird das OMG MARTE-Paket angewendet, das über einen Verteilungsfunktionsblock die statische Ausfallwahrscheinlichkeit ersetzt. Zusätzlich werden Verteilungsfunktionen für die Exponentialfunktion, die LogNormal-Funktion und die Weibull-Funktion eingesetzt. Auf diese Art und Weise ist es möglich, weitere Verteilungsfunktionen in das Framework zu integrieren, wobei für die meisten Domänen diese drei Funktionen ausreichend erscheinen.

Dieser Formalismus bildet die Grundlage für die Metamodelle der DFTs in MMBSEFE. Im Folgenden wird die Bibliothek und das Profil durch Abbildung 3.20 und Abbildung 3.21 beschrieben. Detaillierte Erläuterungen zu den einzelnen Elementen und Stereotypen finden sich in Anhang A.9.11.

DFT-Bibliothek

Abbildung 3.20 enthält das Konzept der dynamischen Fehlerbaumanalyse.

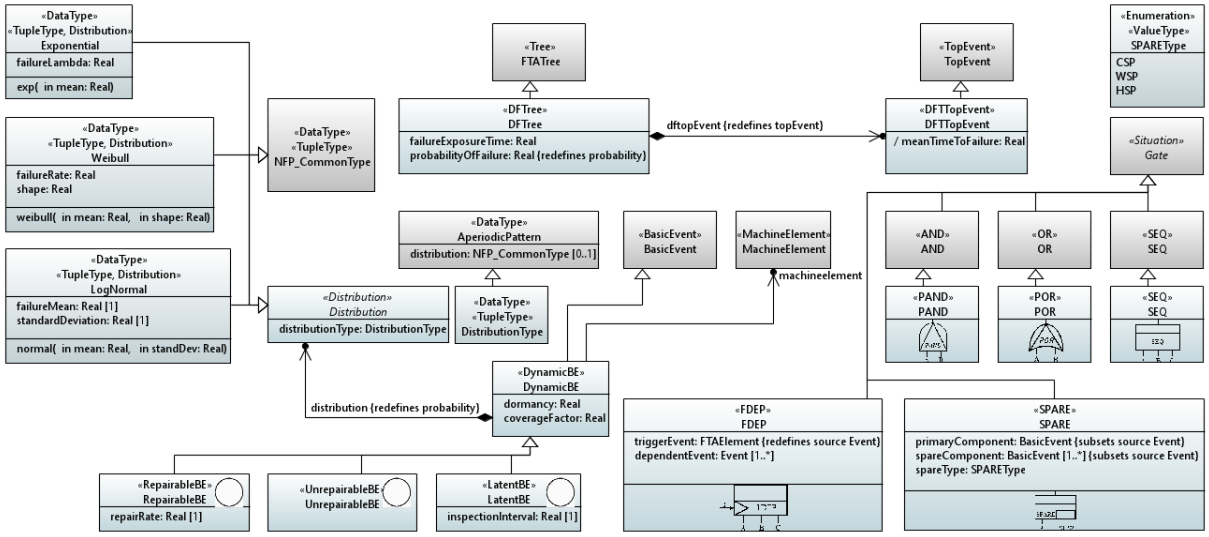


Abbildung 3.20: DFT-Bibliothek

Es gibt verschiedene Klassen von Objekten, die im Zusammenhang der Analyse verwendet werden. Eine Klasse ist der `DFTree`, der spezialisiert ist von `FTATree`. Der `DFTree` fügt das Attribut `failureExposureTime` hinzu und definiert die Bedeutung von `probability` für `probabilityOfFailure` und `topEvent` für `dftopEvent` neu. `DFTree` bildet hierfür eine Komposition zur Klasse `DFTTopEvent`, die spezialisiert ist auf den `TopEvent` und zusätzlich zum Attribut `property` das Attribut `meanTimeToFailure` definiert.

Die Gates `PAND`, `POR` und `SEQ` sind spezialisiert auf die Gates `AND`, `OR` und `SEQ`, während `FDEP` und `SPARE` direkt von der Klasse `Gate` spezialisiert sind. Das Gate `FDEP` definiert das Attribut `triggerEvent` neu und fügt das Attribut `dependentEvent` für ein oder mehrere abhängige Ereignisse hinzu. Das Gate `SPARE` unterteilt `sourceEvent` und definiert das Attribut `primaryComponent` und `spareComponent` neu, wobei mehrere `spareComponent` genutzt werden können. Zusätzlich besitzt `SPARE` einen `spareType` mit dem Value-Type `SPAREType`, um zwischen den verschiedenen Typen zu unterscheiden.

Eine weitere Klasse ist `DynamicBE`, die von `BasicEvent` spezialisiert und mit einem `MachineElement` assoziiert ist. `DynamicBE` definiert das Attribut `dormancy` für die prozentuale Ruhezeit und das Attribut `coverageFactor` für den Abdeckungsfaktor. Mit einer Komposition zu `Distribution` definiert `DynamicBE` das Attribut `probability` neu, um jeweils eine der verschiedenen Verteilungsformen `Exponential`, `Weibull` und `LogNormal` als Teil jedes dynamischen Basisereignisses zu betrachten. Die Klassen `RepairableBE`, `UnrepairableBE` und `LatentBE` spezialisieren den Block `DynamicBE`, wobei `RepairableBE` das Attribut `repairRate` und `LatentBE` das Attribut `inspectionInterval` definiert.

DFT-Profil

Abbildung 3.21 enthält die Stereotypen der dynamischen Fehlerbaumanalyse.

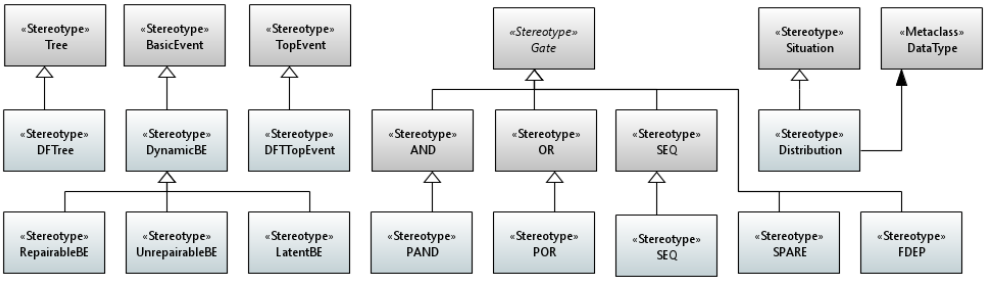


Abbildung 3.21: DFT-Profil

Das DFT-Profil besteht fast nur aus Spezialisierungen aus den Stereotypen vom FTA-Profil. `DFTree` ist eine Spezialisierung von `Tree`, `DFTTopEvent` ist eine Spezialisierung von `TopEvent`, `DynamicBE` ist eine Spezialisierung von `BasicEvent`, `PAND` ist eine Spezialisierung von `AND`, `POR` ist eine Spezialisierung von `OR`, `SEQ` ist eine Spezialisierung von `SEQ` und `SPARE` sowie `FDEP` sind Spezialisierungen von `Gate`. Zusätzlich bildet `DynamicBE` eine Generalisierung für die dynamischen Basisereignisse `RepairableBE`, `UnrepairableBE` und `LatentBE`. Als Letztes wird zusätzlich der Stereotyp `Distribution` definiert, der eine Spezialisierung von `Situation` darstellt und die Metaklasse `DataType` erweitert, um in der Bibliothek auf `NFP_CommonType` abgebildet werden zu können.

3.3.3 Metamodellumsetzung für SBBM

Zur Entwicklung eines Architekturkonzeptes für die SBBM, mit dem Ziel der einfachen Anwendbarkeit für Sicherheitsingenieure in der Maschinenindustrie und Interkonnektivität zu anderen Analysemethoden, wurden die grundlegenden RAAML-Core und RAAML-General-Pakete mit ihren Profilen und Bibliotheken erweitert. Die Erweiterungen führen die für SBBM typischen Abhängigkeitsmuster und Blockstrukturen ein, um die Ausfallwahrscheinlichkeit von Sicherheitsfunktionen, abhängig der Struktur, DC und CCF zu berechnen und eine praxisnahe Anwendung für die Maschinenindustrie zu garantieren. Aufgrund der Domänenausrichtung von ISO 13849 [34] ist das Metamodell lediglich für den Maschinenbau anwendbar.

Die bereits in Abbildung 3.7 dargestellte Einordnung in MMBSEFE zeigt die Erweiterungen in Bezug auf die bestehende OMG RAAML-Paketstruktur (vgl. Abbildung A.9 in Anhang A.7.2). Im bestehenden Paket `Methods` wird das domänenspezifische Paket `ISO 12100 & ISO 13849` für SBBM erweitert. Unter `Methods` schafft der Ansatz hierfür ein Paket `SBBM`. Das SBBM-Profil definiert die Stereotypen für zu bewertende Sicherheitsfunktionen, Teilfunktio-

nen, Blöcke, auslösende Ereignisse und Reaktionsereignisse, sowie entsprechenden Verbindungen zwischen den Elementen, die sich, ähnlich des FTA-Profils, auf das Core-Profil beziehen. Die SBBM-Bibliothek beschreibt das Verhalten und die Parameter als Erweiterung der General Concepts-Bibliothek.

Das SBBM-Paket soll wie DFTs separat, ohne Abhängigkeiten zu MMBSEFE, verwendet werden können, um den Ansatz für andere Framework-Erweiterungen unabhängig zu halten und das vereinfachte Konzept der RBD zur Zuverlässigkeitsbestimmung in anderen geeigneten domänenspezifischen Modellen verwenden zu können. Dies wird im Weiteren speziell gekennzeichnet und benannt.

Die Konzeption wurde nach Vorbild des FTA-Pakets vorgenommen. Eine Sicherheitsfunktion ist ein Szenario und dysfunktionales Ereignis, das aus einem auslösenden Ereignis, einem Reaktionsereignis und Teilfunktionen besteht, sowie einen PL_r zur Vorgabe eines PL und PFH_D zur Ergebnisberechnung besitzt. Jede Teilfunktion, auch Subsystem genannt, entspricht ebenso einem dysfunktionalen Ereignis und deckt die Kategorie und Maßnahmen gegen Common Cause Failure (CCF) ab, die ebenso wie PL bereits im ISO 12100 & ISO 13849-Paket spezifiziert wurden, um PFH_D aus Blöcken zu berechnen. Blöcke repräsentieren ebenso dysfunktionale Ereignisse und besitzen Attribute zur Angabe von $MTTF_D$, Diagnosedeckungsgrad (DC) und der Gebrauchsdauer. Jedes Subsystem ist neben einer Teilfunktion auch als Block zu betrachten, der Berechnungsfaktoren selbst beinhalten muss, besonders wenn es sich um ein gekapseltes Subsystem handelt. Andernfalls dienen die Attribute als Zwischenergebnisse. Jeder Kanal muss aus mindestens einem Block bestehen. Ein Block kann eine Eingangseinheit, eine Logik, eine Ausgabeeinheit, eine Testeinrichtung oder ein Ausgang einer Testeinrichtung sein. Logik bildet hierbei Zustandsdiagramme. Zusätzlich können nach den Angaben in SISTEMA folgende Verfeinerungen angewendet werden. Ein Block kann zur besseren Verschachtelung der Funktionalität zusätzlich aus Elementen bestehen. Elemente sind hierbei ebenso Blöcke, die Eingangseinheiten, Logik oder Ausgabeeinheiten darstellen können. Zur besseren Darstellung sollen Subsysteme ebenso in Eingangseinheiten, Logik und Ausgabeeinheiten konfiguriert werden. Zu jedem Block wird ein Maschinenelement aus MMBSEFE assoziiert, sofern es sich um keinen der Spezialformen handelt. Bei einer Eingangseinheit, Logik und Ausgabeeinheit wird stattdessen ein E/E/PE-Sensor-Element, Controller-Element oder E/E/PE-Aktor-Element assoziiert. Sollte die Analysemethode ohne MMBSEFE eingesetzt werden, kann dieses entweder außen vor gelassen oder durch ein Element eines anderen domänenspezifischen Frameworks ersetzt werden. Zur Umsetzung in Abhängigkeit zu RAAML müssen, ähnlich FTA, die Konzepte `Scenario` und

`AbstractEvent` sowie das darauf aufbauende `DysfunctionalEvent` aus der General Concepts-Bibliothek erweitert werden.

Dieser Formalismus bildet die Grundlage für die Metamodelle der SBBM in MMBSEFE. Im Folgenden wird die Bibliothek und das Profil durch Abbildung 3.22 und Abbildung 3.23 beschrieben. Detaillierte Erläuterungen zu den einzelnen Elementen und Stereotypen finden sich in Anhang A.9.12.

SBBM-Bibliothek

Abbildung 3.22 enthält das Konzept der sicherheitsbezogenen Blockdiagrammmethode.

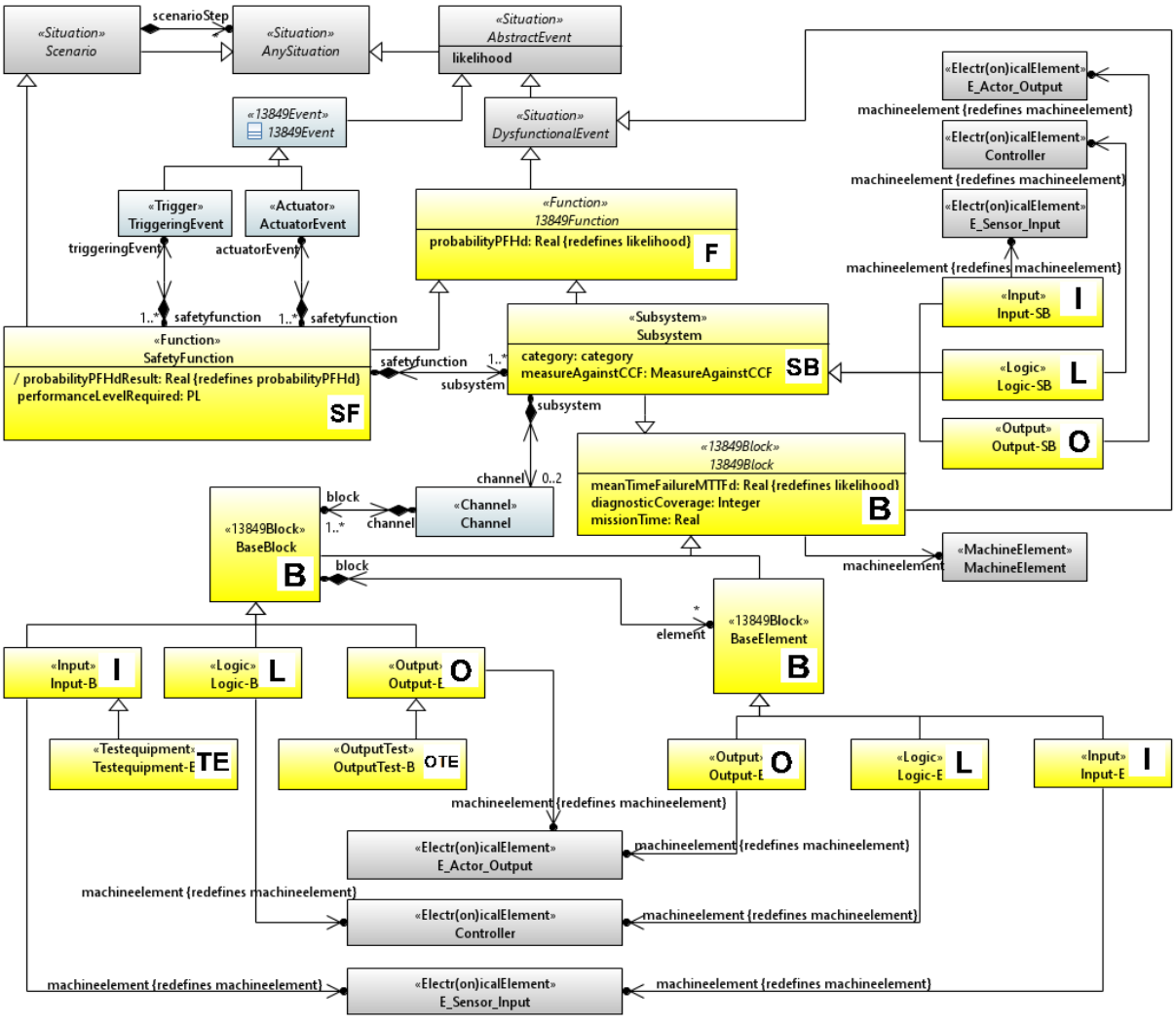


Abbildung 3.22: SBBM-Bibliothek

Die Klassen `13849Function` und `13849Block` leiten sich von der Klasse `DysfunctionalEvent` ab. `13849Function` überschreibt die Bedeutung von `likelihood` für `probabilityPFHd`, während `13849Block` `likelihood` für `meanTimeFailureMTTFd` neu definiert. Zusätzlich fügt `13849Block` die Attribute `diagnosticCoverage` und `missionTime` hinzu und assoziiert ein `MachineElement`.

Die Klassen SafetyFunction und Subsystem spezialisieren jeweils die Klasse 13849Function. SafetyFunction wird neben 13849Function für Scenario spezialisiert und definiert probabilityPFHd für probabilityPFHdResult neu. Sie definiert auch das Attribut performanceLevelRequired. SafetyFunction besitzt eine Komposition zu TriggeringEvent und ActuatorEvent, die aus 13849Event spezialisiert sind, welches wiederum aus AbstractEvent spezialisiert ist. Sie besitzt auch eine Komposition zu mindestens einem Subsystem.

Subsystem ist aus 13849Function und 13849Block spezialisiert und besitzt eine Komposition zu bis zu zwei Channel. Ein Channel hat eine Komposition mit mindestens einem BaseBlock, welches wiederum eine Komposition zu einem BaseElement besitzen kann. BaseBlock und BaseElement sind beide spezialisiert aus 13849Block.

Schließlich ist Subsystem eine Generalisierung von Input-SB, Logic-SB und Output-SB. BaseBlock ist eine Generalisierung von Input-B, Logic-B und Output-B, während BaseElement eine Generalisierung von Input-E, Logic-E und Output-E ist. Die Klassen Input-B, Input-SB und Input-E definieren machineelement für machineelement neu, wobei der Typ E_Sensor_Input verwendet wird. Logic-B, Logic-SB und Logic-E definieren machineelement für machineelement neu, wobei der Typ Controller verwendet wird. Schließlich definieren Output-B, Output-SB und Output-E machineelement für machineelement neu, wobei der Typ E_Actor_Output verwendet wird. Zusätzlich ist Testequipment-B eine Spezialisierung von Input-B, während OutputTest-B eine Spezialisierung von Output-B ist.

SBBM-Profil

Abbildung 3.23 enthält die Stereotypen der sicherheitsbezogenen Blockdiagrammmethode.

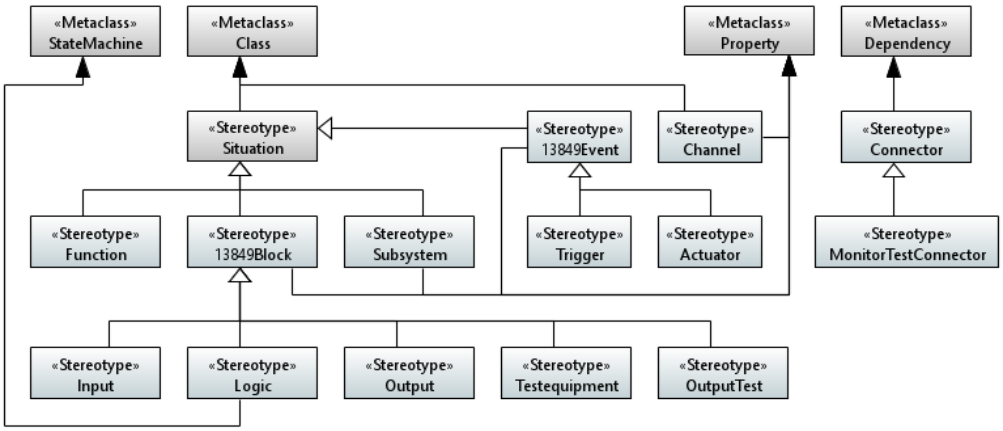


Abbildung 3.23: SBBM-Profil

Das SBBM-Profil besteht im Kern aus den vier Stereotypen `Function`, `Subsystem`, `13849Block` und `13849Event` spezialisiert aus dem Stereotyp `Situation`. `13849Block` ist eine Generalisierung der Stereotypen `Input`, `Logic`, `Output`, `Testequipment` und `OutputTest`. `Logic` stellt zusätzlich eine Erweiterung der Metaklasse `StateMachine` dar, um auf dem gleichen Element Zustandsdiagramme abbilden zu können. `13849Event` ist eine Generalisierung von `Trigger` und `Actuator`. Ein `Channel` hat keinen direkten Bezug zu RAAML, erweitert jedoch die Metaklasse `Class`, spezialisiert `Block` und erweitert, genauso wie `13849Block`, `Subsystem` und `13849Event` die Metaklasse `Property`. Um nicht nur die Struktur einer Sicherheitsfunktion über Property-Stereotypen in einem IBD darzustellen, sondern auch die Verbindungen zwischen ihnen, erweitert `Connector` die Metaklasse `Dependency` und generalisiert `MonitorTestConnector`.

3.3.4 Inhaltlicher Beitrag

In diesem Abschnitt wird der inhaltliche Beitrag der vorgestellten Modellierung und Gestaltung für die Forschungsaufgabe FA2-TB des Typs Theoriebildung im Bereich der Modellierung maschinenbauspezifischer Sicherheitsanalysemethoden zusammengefasst.

Für MMBSEFE wurden Metamodelle und Bibliotheken geschaffen, die die relevanten maschinenbauspezifischen Analysemethoden DFT und SBBM zum Nachweis der Risikominderung definieren. Abschnitt 3.3.1 beschreibt RAAML als das am besten geeignete Framework für die Modellierung. Mittels des zuvor definierten ISO12100&ISO13849-Pakets zur maschinenbauspezifischen Modellierung der funktionalen Sicherheit konnte bereits in Abschnitt 3.2.11 die Risikographmethode als wichtiges Glied der Risikobeurteilung in die Metamodelle eingebunden werden. Mittels weiterer Erweiterungen und Spezialisierungen konnten nun neben anderen üblichen maschinenbauspezifischen Analysemethoden, wie FMEA und FTA, die bereits Bestandteil von RAAML sind, DFTs und die SBBM in die gleiche Modellebene integriert werden. Hierdurch entsteht die Möglichkeit zum Informationsaustausch und der Anbindung verschiedener Analysemethoden, sodass einzelne situationsbasierte Elemente in verschiedenen Analysemethoden gleichzeitig eingesetzt werden können oder Grundlage für andere liefern können. Es entsteht eine weitgehend vollständige Abdeckung aller notwendigen Analysemethoden zur Risikobeurteilung und Risikominderung im Maschinenbau.

In Abschnitt 3.3.2 wurden das Profil und die Bibliothek von DFTs beschrieben. Durch die Verallgemeinerung von FTA konnten viele Fehlerbaumelemente in ihrer grundlegenden Anordnung und ihrem Einsatzzweck weiterverwendet werden. Die Erweiterung bietet damit die Möglichkeit, PF_D über eine bestimmte Zeit und MTTF-Werte über dynamische Gates zu berechnen.

Das Konzept der dynamischen Basisereignisse wurde so erstellt, dass neben den beschriebenen Verteilungsfunktionen weitere Verteilungsfunktionen angefügt werden können. Ähnliches gilt für die verschiedenen Typen von dynamischen Basiselementen. Beispielsweise könnten hierdurch zusätzlich Ereignisse berücksichtigt werden, die beispielsweise nach bestimmten Frequenzmustern auftreten, wodurch allerdings auch ein weiterer Verteilungstyp berücksichtigt werden müsste. Durch die Assoziation der Basisereignisse mit dem MachineAssemblyStructure-Paket können die modellierten Maschinenelemente direkt nachvollzogen werden, um beispielsweise bei einem Austausch der Komponente diese einfach anpassen zu können, ohne den Zusammenhang zur Analyse zu verlieren.

In Abschnitt 3.3.3 wurden das Profil und die Bibliothek der SBBM beschrieben. Durch die Spezialisierung der Elemente aus RAAML-Core und -General können die Elemente der SBBM nahtlos in die gemeinsamen Metamodelle integriert werden. Aufgrund des ähnlichen Aufbaus im Vergleich zu FTA und DFT können vergleichbare Analyseergebnisse erzielt werden, was einen Mehrwert durch die freie Wahl der Analysemethode bietet. Die Erweiterung bietet die Möglichkeit, PFH_D mit Annahme einer exponentiellen Verteilungsfunktion zu berechnen, indem $MTTF_D$, DC, CCF und Gebrauchsdauer für jedes Subsystem bzw. den darin enthaltenen SBBM-Blöcken bereitgestellt werden. Durch die Erweiterung der MachineAssemblyStructure-, Electrical- und ISO12100 & ISO 13849-Bibliothek zur Assoziation mit allgemeinen Maschinen- oder E/E/PE-Elementen (Sensoren, Controller oder Aktoren), lassen sich SBBM-Blöcke direkt mit Komponenten in Verbindung bringen. Dies fördert im Vergleich zu DFTs eine feinere Abhängigkeitsbeziehung mit Sensor-, Controller- und Aktor-Komponenten, was den elektrotechnischen und informationstechnischen Entwurf stärker koppelt. Zusätzlich bietet es sich an, die Modellelemente von Komponenten mit Subsystemen und SBBM-Blöcken auf die gleichen SysML-Blöcke aufzuprägen. Dies stärkt die Abhängigkeitsbeziehung und es lassen sich Werte, auf die während der Modellierung, Implementierung und dem Aufbau besonders zu achten ist, wie CCF und die Umsetzung von DC, ebenso in anderen Stakeholder-spezifischen Modellen betrachten.

Um die Leistung der Analysemethoden Risikograph, SBBM und DFT in Form der bereits theoretisch im Stand der Technik beschrieben Vorgaben nachzuweisen, sollen in der Implementierung die Analysen implementiert und in der Evaluierung dieser Arbeit weiter belegt werden.

3.4 MBSE-Prozesse für KMUs

In diesem Abschnitt werden kleine und mittelständische Unternehmen (KMUs) unter den nach Abschnitt 2.3.1 identifizierten Hindernissen und Bedürfnissen bei der Anwendung von MBSE

in der MMBSEFE-Modellierung besonders berücksichtigt und damit FA3-TB abgehandelt. Das bedeutet, dass die Verwendung von modellierten UML- und SysML-Elementen zur Darstellung und Nutzung technischer und sicherheitsbezogener Merkmale KMU-freundlich vereinfacht werden sollen. Um dieses Ziel zu erreichen, wird in Unterabschnitt 3.4.1 ein Ansatz zur einfachen Anwendung von MMBSEFE in KMUs basierend auf einem Anwendungsprofil vorgestellt. In Unterabschnitt 3.4.2 folgt das Konzept der losen Toolkopplung zum Einsatz etablierter Werkzeuge. Schließlich wird in Unterabschnitt 3.4.3 die Nutzung der in KMUs etablierten Entwicklungsumgebung LabVIEW mit Programmiersprache G zur Umsetzung der losen Toolkopplung erläutert. In diesen Abschnitten wird zusätzlich auf weitere Faktoren eingegangen, die die Einschränkung und einfache Anwendbarkeit der Framework-Erweiterung in KMUs wie ProNES betreffen.

3.4.1 Vorgehen zur vereinfachten Anwendbarkeit von MMBSEFE in KMUs

Während der Entwicklung eines flexiblen, nutzerzentrierten Ansatzes für Anwendungsfälle in Abschnitt 3.2.2 wurde indirekt die Modellierung eines Architekturkonzepts für KMUs in der Maschinenindustrie berücksichtigt. Dies geschah bereits, um im Sinne der Agilität und Zusammenarbeit Modellelemente und relevante Artefakte so abzubilden, dass sie von den Stakeholdern verstanden und in Verbindung zueinander gebracht werden können. Hierfür wurden SysML-Elemente in domänenspezifische Profile verkleidet, um einen Mehrwert für die Modellierung zu bieten und das Verständnis zu fördern. Der Fokus liegt hierbei auf dem einfachen Verständnis und der Verbindung der Modelle der technischen Stakeholder untereinander und zu den Analysemodellen.

Es stellt sich allerdings die Frage, wie der Lernaufwand bei der Einarbeitung in MMBSEFE begrenzt werden kann, um die finanziellen und zeitlichen Ressourcen für den Erwerb und die Einarbeitung zu schonen. Um die vollständige Abbildung der Artefakte zu gewährleisten, fehlerhafte Modellierung zu vermeiden, die Nachvollziehbarkeit zu verbessern und den Interpretationsspielraum zu verringern, bieten sich neben der Modellierung in UML- und SysML-Modellierungswerkzeugen, auch Anwendungsprofile, Object Constraint Language (OCL in Anhang A.3.5) und Cascading Style Sheets (CSS in Anhang A.3.6) an. Der Einsatz von OCL-Constraints und die Entwicklung eines Anwendungsprofils verringert die Fehleranfälligkeit bei der Anwendung der Framework-Erweiterung, da Entwickler automatisch angehalten sind, ihre Modelle gerecht dem Framework zu füllen und zu vernetzen. Zusätzlich kann durch CSS-Da-

teien das visuelle Erscheinungsbild von angewendeten Stereotypen, speziell für Analysemethoden, so angepasst werden, dass es den Normen entspricht und zu einer einfachen Umsetzung der Anwendungsmodelle beiträgt. Dabei besteht ebenso die Möglichkeit, über ein weiteres Attribut in Elementen, wie beispielsweise bei der SBBM die Sicherheitsfunktion, Subsysteme und Blöcke, die Ergebnisdarstellung zu verbessern. Mit der Definition eines ValueType `result`, der in den Stereotypen vergeben wird, lassen sich über einen Eintrag in einer CSS-Datei Requirement-, Klassen- und Property-Elemente farblich kennzeichnen.

Anwendungsprofile tragen dazu bei, die Modellierung von Anwendungsmodellen zur Unterstützung der Begriffswelt der Stakeholder zu vereinfachen, indem ein direkt anwendbares Profil die MMBSEFE-Bibliotheken aufprägt. Dies stärkt das Vertrauen in die gemeinsame Datenbasis, da Fehler bei der Anwendung reduziert werden, was die Innovationsbereitschaft erhöhen und den Mehrwert für die einzelnen Stakeholder steigern soll. Die Framework-Erweiterung bietet somit Grundlagen zur Modellierung der Leistung, Sicherheit und Widerstandsfähigkeit von Maschinen sowie eine Verbesserung der Wartbarkeit in Bezug zum erfolgreichen Einsatz von MBSE in KMUs aus Abschnitt 2.3.1, was sich positiv auf Qualität, Kosten, Dauerhaftigkeit und Effizienz auswirken soll.

Um ein Anwendungsprofil zur einfachen Anwendbarkeit von MMBSEFE umzusetzen, werden die Profile und Bibliotheken, wie beispielhaft in Abbildung 3.24 für die SBBM-Bibliothek dargestellt, in Stereotypen verpackt.

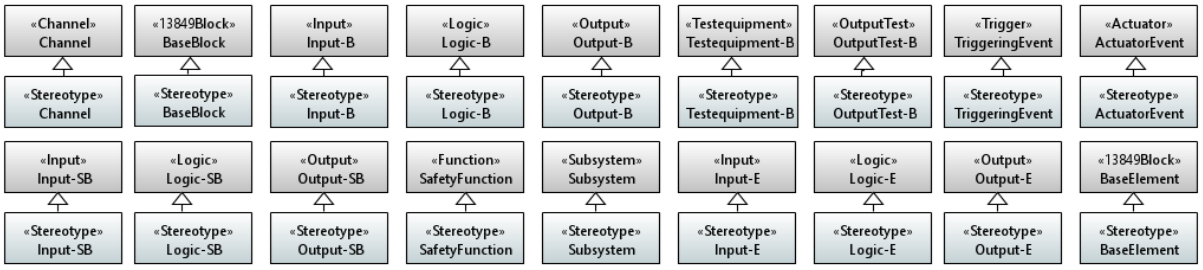


Abbildung 3.24: Anwendungsprofil SBBM

Durch Stereotypbildung können die zu modellierenden Elemente und Abhängigkeiten nicht nur über Nutzung und Generalisierung der Bibliotheksartefakte eingesetzt und in Beziehung gebracht werden, wie bereits durch die bisherige Modellierung möglich, sondern es können über eine direkte Profil-Anwendung der Elemente, wie in Abbildung 3.25 gezeigt, Attribute aus den Bibliotheken direkt angesprochen und zugewiesen werden. In vorkonfigurierten Feldern und durch Drop-Down-Auswahl stehen dabei lediglich die passenden Elemente zur Auswahl.

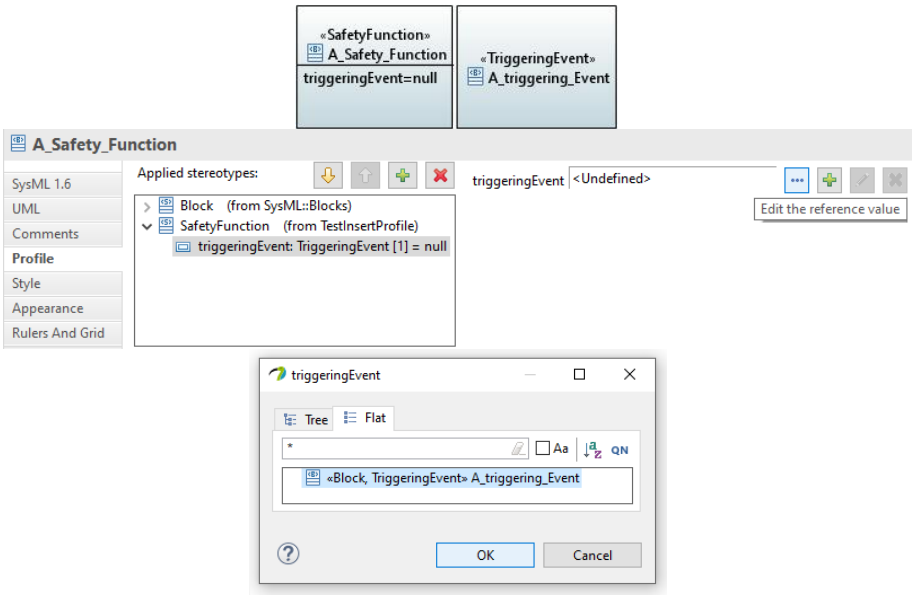


Abbildung 3.25: Anwendungsprofil einfaches Beispiel Stereotyp SafetyFunction
Von Oben nach Unten: Beispiel zu SafetyFunction und TriggeringEvent, Konfiguration zu angewendeten Stereotypen, Auswahlmenu für TriggeringEvent

Anwendungsprofile bieten zudem den Vorteil der einfachen Interpretierbarkeit, da beispielsweise der UML2-XMI-Standard von Eclipse Stereotypen und deren Attribute in einfachen XML-Elementen speichert. Diese lassen sich zum Export Stakeholder-spezifischer Informationen einfach auslesen und beim Import von Daten einfach abgleichen.

Ähnlich SBBM werden im Anwendungsprofil Stereotypen für Prüfmaschinen, für die verschiedenen technischen Sichtweisen und für DFTs modelliert. Das Paket der Risikobeurteilung durch Risikographen erhält kein Anwendungsprofil, da es sich aufgrund der klaren Abhängigkeiten und Berechnungsfunktionen im weiteren Verlauf der Arbeit zum Vergleich des Implementierungsaufwandes in Bezug zu den anderen Analysemethoden bzw. einer engen Toolkopplung anbietet.

3.4.2 Lose Toolkopplung

Nach einer umfangreichen Literaturrecherche in den Abschnitten 2.2.2 und 2.2.3 wurden die Begriffe der losen und engen Toolkopplung eingeführt, die sich auf die Verbindung von Sicherheitsanalysewerkzeugen beziehen und in Tabelle 3.2 verschiedenen relevanten Veröffentlichungen zugeordnet sind. Auf Grundlage der Herausforderungen, die in Abschnitt 2.3.1 beschrieben wurden, fiel die Entscheidung auf die Integration bestehender bekannter bewährter Werkzeuge durch lose Toolkopplung. Dadurch kann beispielsweise für SBBM ein in KMUs der Maschinenindustrie etabliertes und bewährtes Analysewerkzeug wie SISTEMA [222] weiterverwendet werden. Es entsteht aber auch die Möglichkeit, etablierte Entwicklungswerkzeuge, wie AutoCAD, EPLAN oder TwinSAFE lose zu koppeln. Dadurch

entsteht eine nahtlose Informationsverarbeitung, ein Austausch in jedem Prozessschritt der mechanischen, E/E/PE, softwaretechnischen und sicherheitstechnischen Entwicklung und die Nutzung bestehenden Knowhows auf einer gemeinsamen Informationsbasis.

Veröffentlichungen	Toolkopplung
Joshi et al. '07 [184]	Lose
Xiang et al. '11 [185]	Lose
David et al. '10 [186]	Lose
Hecht et al. '15 [54]	Lose
Yakymets et al. '18 [53]	Lose
Helle '12 [58]	Lose
Thramboulidis et al. '10 [59]	Lose
Mhenni et al. '14 [35]	Lose
Biggs et al. '16 [60]	Lose
Baklouti et al. '19 [188]	Lose
Biggs et al. '19 [37]	Eng
Berres et al. '21 [38]	Eng

Tabelle 3.2: Zusammenfassung der Veröffentlichungen zur Toolkopplung

Durch die lose Kopplung der Informationsbasis mit externen Werkzeugen können alle erforderlichen Informationen Stakeholder-spezifisch exportiert und in bestehenden Entwicklungs- und Analysewerkzeugen importiert und analysiert werden. Um einen generalisierbaren Ansatz zu schaffen, soll zunächst ein Konzept zur Toolkopplung über Modelltransformatoren auf der Grundlage von Metadaten entwickelt werden, welches den Ex- und Import von Artefakten erleichtert. Hierbei bieten sich aufgrund des einfachen Datenaustauschs XMI-unterstützende Modellierungswerkzeuge an.

Das in Abbildung 3.26 dargestellte Architekturkonzept wurde entwickelt, um eine einfache Integration und Erweiterbarkeit zu erreichen. Es besteht zunächst aus einer MBSE-Plattform, die bereits die OMG-Standards für UML und SysML unterstützt. Die von der OMG bereitgestellten RAAML-Pakete können somit leicht in die Modellierungsplattform integriert und mit MMBSEFE domänenspezifisch angereichert werden. Dadurch entsteht für KMUs eine einfache Wartbarkeit und Erweiterbarkeit für die Entwicklung und Erweiterung der Anwendungsmodelle. SysML-Modellierungsdokumente können mittels des XMI-Standards für den einfachen Export in externe Entwicklungs- und Analysewerkzeuge die Grundlage bieten. Besonders relevant für die funktionale Sicherheit, bei loser Kopplung können bereits etablierte validierte und zertifizierte Analysewerkzeuge genutzt werden. Dies unterstützt die Erweiterbarkeit mit geringen Ressourcen. Das entwickelte allgemeine Konzept gibt keine Vorgaben zur Art der Umsetzung von Transformern, erfordert jedoch ein Design in Kohärenz mit den Spezifikationen der RAAML- und MMBSEFE-Pakete dieser Arbeit.

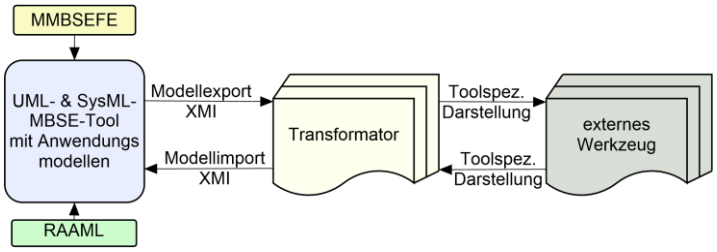


Abbildung 3.26: Vereinfachtes Architekturkonzept zur Toolkopplung

Abbildung 3.26 beinhaltet einen iterativen Prozess, um das Sicherheitsdesign und die Erfüllung der Anforderungen zu erreichen. In jedem Zyklus des Prozesses können die Erkenntnisse der vorherigen Iteration einbezogen werden und zur Verbesserung des Designs beitragen. Das Konzept kann als Unterzyklus in Software-Engineering-Prozessmodellen, wie dem V-Modell, verwendet werden. Hierbei wird zunächst die MBSE-Plattform durch RAAML und MMBSEFE erweitert, um Entwicklungs- und Sicherheitsmodelle der gemeinsamen Informationsbasis darzustellen. Anschließend können die Modelle mittels XMI exportiert und mittels eines sogenannten Transformators in eine werkzeugspezifische Darstellung transformiert werden. Im Kontext dieser Arbeit soll auf die sicherheitsbezogenen Analysen und die Bestimmung entsprechender quantitativer Maße in den Analysewerkzeugen eingegangen werden. Über einen XMI-Import können die geeignet rücktransformierten Analyseergebnisse in die Modellebene zurückintegriert werden. Hierbei können sie mit den gestellten Anforderungen verglichen und erfüllte und nicht erfüllte Anforderungen benutzerfreundlich durch unterschiedliche Farben markiert werden. Dies ermöglicht einen direkten Rückschluss auf die Güte der Entwicklung und bildet so die Grundlage für den nächsten Iterationszyklus. So können ggf. weitere Maßnahmen unter den Stakeholdern diskutiert werden, um entsprechende Anforderungen zu erfüllen. Analog zu den normativen Spezifikationen aus IEC 61508 können Stakeholder die Systemmodelle ebenso für Model-Based Testing verwenden, was den MBSE-Ansatz weiterführt. Analog hierzu, allerdings nicht Teil dieser Arbeit, ist vorstellbar, einen Export in CAD-Werkzeuge oder Werkzeuge zur Schaltplanerstellung umzusetzen. Der Export in Werkzeuge zur Codegenerierung von Softwarekomponenten nimmt jedoch eine Sonderstellung ein, da bereits etablierte Vorgehensweisen und Werkzeuge zur automatischen Codegenerierung von endlichen Automaten [277, 278] für softwarebasierte Sicherheitsfunktionen in SPS-Code [197, 279, 280] existieren. Aufgrund des fehlenden Forschungsbedarfs wird hierauf im Weiteren nur kurz eingegangen. Die tatsächliche Umsetzung der CAD-Modelle, der Schaltpläne und des Softwarecodes kann nachfolgend innerhalb der Entwicklerwerkzeuge geschehen.

Auf die gleiche Weise wie im OMG RAAML-Ansatz ergibt sich bei diesem Ansatz eine einfache Nachvollziehbarkeit zwischen dem Systemmodell und den Analysemodellen durch die

enge Modellkopplung, wobei die Modelle durch MMBSEFE sogar noch stärker gekoppelt sind, da Analyseartefakte direkt auf Systemartefakte zeigen. Dies begünstigt eine einfache Wartung der Analysemodelle bei Änderungen des Systemmodells und die Möglichkeit, Analysemethoden auf Systemebene einfacher zu erstellen. Im Gegensatz zum OMG-Ansatz führt die externe Durchführung über die lose Kopplung der Analysewerkzeuge jedoch zu einer lockereren Verknüpfung, was eine schlechtere Verfolgbarkeit von Entitäten, wie beispielsweise die Beziehungen zwischen und Auswirkungen auf Berechnungen, zur Folge hat und keine Informationen darüber gibt, wie genau die Analysemodelle tatsächlich umgesetzt und berechnet werden. Das bedeutet für diesen Ansatz, dass Referenzen zwischen Entitäten aus den Analysealgorithmen nicht berücksichtigt werden können, da die Analysemethode selbst nicht im Modell bekannt ist. So entsteht allerdings aus dem OMG-Ansatz ein Konzept, das für KMUs leichter anwendbar und erweiterbar ist, vor allem, wenn für bestimmte Analyseverfahren bereits etablierte Analysewerkzeuge existieren.

Um einen effizienten und ressourcenschonenden Prozess zu gewährleisten, empfiehlt es sich, eine bereits etablierte Entwicklungsumgebung für die Entwicklung von Modelltransformatoren auszuwählen, um die eigenständige Wartbarkeit und Erweiterbarkeit in KMUs zu erleichtern.

3.4.3 LabVIEW zur Entwicklung von Modelltransformatoren

Das Konzept ermöglicht es, Transformatoren in jeder Entwicklungsumgebung und Programmiersprache zu implementieren. Dies hat den Nebeneffekt, dass andere Interessierte das Konzept je nach verfügbarem Knowhow und Format an ihre Umgebungen übertragen, anpassen oder erweitern können. In Bezug zu Eclipse Papyrus ist das Eclipse Modeling Framework (EMF) die Standardlösung, das aufgrund des integrierten Editors und der engeren Kopplung meist als erste Wahl zur Umsetzung von Transformatoren dient [281]. Wie Cederbladh et al. [281] erläutern, ermöglicht EMF die Erstellung spezifischer Views, die eine bessere Visualisierung und Verwaltung komplexer Systeme erlauben. Dies unterstützt die Entwickler dabei, maßgeschneiderte Lösungen für die jeweiligen Anforderungen zu erstellen [281]. In KMUs des Maschinenbaus unterstützen allerdings einige anwendungsbezogene Vorteile die Entscheidung gegen EMF. EMF basiert auf und erweitert MOF und bietet daher auf der einen Seite einen sehr integrativen, flexiblen und erweiterbaren Ansatz zur Entwicklung von Transformationen, doch fehlen oft die finanziellen und zeitlichen Ressourcen in KMUs für eine spezifische Schulung der Entwickler im eher für Informatiker ausgelegten EMF. Auf der anderen Seite ist das Knowhow und die benutzerfreundliche graphische Programmieroberfläche um LabVIEW in

der Maschinenindustrie und den dort etablierten Testumgebungen und Prüfmaschinen weit verbreitet [81, 282, 283]. National Instruments ist nach eigenen Aussagen sogar der Marktführer beim Aufbau von Test- und Messlösungen [284]. Bei ProNES sind z.B. mehr als 50 % der angestellten Softwareentwickler reine LabVIEW-Entwickler. Ein immer wichtigerer Faktor in der Systementwicklung ist auch die notwendige Entwicklungszeit, aufgrund dessen LabVIEW häufig als Rapid-Prototyping-Tool [83] in der Maschinenindustrie eingesetzt wird. LabVIEW bietet zudem Bibliotheksfunktionen, die XML-Dateien und somit auch die UML2-XMI-Dateien von Eclipse Papyrus einfach lesen und interpretieren können [285]. Das LabVIEW-Knowhow kann daher auch genutzt werden, um Transformatoren für Entwicklerwerkzeuge einzuführen, doch konzentriert sich die Arbeit auf die Einbindung der Analysewerkzeuge.

3.4.4 Inhaltlicher Beitrag

In diesem Abschnitt wird der inhaltliche Beitrag der vorgestellten Modellierung und Gestaltung für die Forschungsaufgabe FA3-TB des Typs Theoriebildung im Bereich eines MBSE-Prozesses für KMUs zusammengefasst.

Im ersten Schritt wurde unter Abschnitt 3.4.1 ein Ansatz zur Vereinfachung der Anwendbarkeit von MMBSEFE durch ein Anwendungsprofil vorgestellt. Darüber hinaus wurde vorgeschlagen, OCL und CSS einzusetzen, um MMBSEFE-Einschränkungen einzuhalten und eine verbesserte Darstellung zu ermöglichen, was den Einarbeitungsaufwand zusätzlich reduziert. Im nächsten Abschnitt 3.4.2 wurde dann ein Architekturkonzept zur losen Kopplung von etablierten und bewährten Entwickler- und Analysewerkzeugen mit MMBSEFE präsentiert. Dies ermöglicht eine kostengünstige und ressourcenschonende Anbindung dieser Werkzeuge mit Ergebnisdatenrückkopplung. Durch den Einsatz einer dem Knowhow des KMUs angepassten Programmiersprache, wie in Abschnitt 3.4.3 für LabVIEW diskutiert, können Transformatoren zusätzlich schnell und effizient entwickelt und gewartet werden.

Basierend auf diesem Konzept zur Toolintegration können die nächsten Schritte zur Integration von Werkzeugen zur Berechnung von Risikographen, DFTs und SBBM abgeleitet werden.

3.5 Zusammenfassung

In diesem Kapitel wurde eine mögliche Modellierung für das Konzept und Design eines maschinenbauspezifischen MBSE-Frameworks für die Entwicklung sicherheitskritischer Anwendungen in KMUs gegeben. Die Modellierung ist in der Lage, die Anforderungen, die sich aus den Forschungsfragen ergeben, zu erfüllen. Die Modellierung zeigt, dass eine Interaktion der Stakeholder Maschinenbauingenieur, Elektroingenieur, Informatiker und Sicherheitsingenieur

in einer MBSE-Framework-Erweiterung entwickelt werden kann, die praktisch die Entwicklung von Prüfmaschinen verbessert. Es wurden aufeinander aufbauende Bibliotheken und Profile für jede Stakeholder- und analysespezifische Sichtweise vorgestellt und auf dieser Grundlage Konzepte zur KMU-freundlichen Anwendung (Anwendungsprofil, OCL, CSS, Lose Toolkopplung mit etablieren Werkzeugen) eingeführt, die eine einfache wiedererkennbare Modellierung der Begriffswelt der Stakeholder im Anwendungsfeld Maschinenbau ermöglichen. Neben den bereits integrierten Sichtweisen und Analysen wurde die Framework-Erweiterung so ausgelegt, dass sie einfach erweitert, abstrahiert und wiederverwendet werden kann.

Tabelle 3.3 zeigt den Prozess in der Erfüllung der Forschungsaufgaben (FA). Alle bereits behandelten Forschungsaufgaben sind mit grauem Hintergrund dargestellt. Die entsprechenden Modellierungsergebnisse werden in fett-kursiver Schrift unter jeder FA des Typs Theoriebildung beschrieben.

Forschungsaufgaben (FA)		
FA	**Beschreibung**	**Abschnitt**
Maschinenbauspezifische Modellierung		
FA1-B	Recherche einer gemeinsamen Modellebene und Architektur für MBSE im Maschinenbau, im Speziellen für Prüfmaschinen, und Grundlage zur Modularisierung	2.1
FA1-TB OH1.1 OH2.1 OH2.2	Design eines modularen Informationsmodells und einer Architektur für MBSE im Maschinenbau, im Speziellen für Prüfmaschinen (um einerseits Stakeholder-spezifische Baugruppen und Elemente modellieren und andererseits domänenspezifische Analysen, wie Risikographen durchführen zu können) ***Anwendungsfälle aller Stakeholder (vgl. Abbildung 3.1) und MMBSEFE (vgl. Abbildung 3.7)***	3.2
FA1-I	Umsetzung einer modularisierten gemeinsamen Modellebene und Architektur für MBSE im Maschinenbau, im Speziellen für Prüfmaschinen	
FA1-E	Evaluation einer modularisierten gemeinsamen Modellebene und Architektur für MBSE im Maschinenbau, im Speziellen für Prüfmaschinen	
Integration der Analysen des Maschinenbaus		
FA2-B	Identifikation maschinenbauspezifische Analysemethoden und passender Frameworks	2.2
FA2-TB OH2.3 OH2.4	Modellierung zur Integration der maschinenbauspezifische Analysemethoden dynamische Fehlerbaumanalyse (DFT) und sicherheitsbezogene Blockdiagrammmethode (SBBM) zum Nachweise der Risikominderung ***Einbindung von DFTs (vgl. Abbildung 3.20) und der SBBM (vgl. Abbildung 3.22) in MMBSEFE***	3.3
FA2-I	Entwurf und Implementierung zur Integration maschinenbauspezifische Analysemethoden	
FA2-E	Evaluation typischer Analysemethoden	

Forschungsaufgaben (FA)		
FA	**Beschreibung**	**Abschnitt**
MBSE-Prozesse für KMUs		
FA3-B	Identifikation der Hindernisse zur Anwendbarkeit von MBSE in KMUs	2.3
FA3-TB OH3.1	Vorgehen zur Anwendbarkeit von MBSE in KMUs unter Berücksichtigung der Aspekte Agilität, Vertrauen, Zusammenarbeit und Effizienz ***Architekturkonzept für KMUs (vgl. Abschnitt 3.4.1 bis 3.4.3)***	3.4
FA3-I	Implementierung eines Konzeptes für KMUs in der Maschinenindustrie	
FA3-E	Evaluation mit KMU	

Tabelle 3.3: Forschungsaufgaben nach Modellierung

Da die in diesem Kapitel vorgestellte Modellierung die verbleibenden Herausforderungen aus Kapitel 2 anspricht und somit die Forschungsaufgaben des Typs Theoriebildung vollständig beantwortet, sieht Nunamaker folgend eine detaillierte Implementierung und Evaluation vor, um zu bestätigen, dass die initialen Anforderungen, die sich aus den Forschungsfragen ergeben, gelöst werden können. In Kapitel 4 werden daher die verbleibenden Forschungsziele des Typs Implementierung aus Tabelle 3.3 angegangen.

Um diese verbleibenden offenen Herausforderungen anzugehen, wurde eine prototypische Umsetzung der präsentierten Bibliotheken und Profile entwickelt und evaluiert. Dies wird in den nächsten beiden Kapiteln detailliert dargestellt. Kapitel 4 liefert Informationen in die Umsetzungen der Pakete und der Transformatoren, einschließlich der Implementierungen in LabVIEW-G-Code zur Kompensation mangelnden Informatik-Knowhows im Maschinenbau mittelständischer Unternehmen. Kapitel 5 enthält eine Fallstudie zur Bewertung der entwickelten Konzepte und der Umsetzung, einschließlich Vergleich von Analyseergebnissen und Experteninterviews, und validiert damit schließlich, dass die in diesem Abschnitt vorgestellte Modellierung verwendet werden kann, um die Einsetzbarkeit von MBSE für KMUs in der Maschinenindustrie für die Entwicklung von Prüfmaschinen zu verbessern.

4 Implementierung

In den folgenden Abschnitten werden die Forschungsziele des Typs Implementierung gemäß dem Ansatz von Nunamaker (vgl. Abschnitt 1.4) beschrieben. Die Hauptziele dieser Implementierung sind der Nachweis der Machbarkeit der entwickelten Konzepte (Proof-Of-Concepts), die Vorbereitung einer detaillierten Evaluation und die Untersuchung weiterer Beobachtungen während der Implementierungsphase. Dieses Kapitel, wie bereits die Kapitel 2 und 3, ist auf die Struktur der Forschungsaufgaben ausgerichtet, wobei für eine zusammenhängende Implementierung MBSE-Prozesse für KMUs aus Abschnitt 3.4 in die maschinenbauspezifische Modellierung integriert wird. Abschnitt 4.1 zeigt hierfür den Aufbau von MMBSEFE in einer Modellierungsplattform und die Implementierung zur Schaffung eines anwendbaren Konzeptes für KMUs. In Abschnitt 4.2 wird zunächst das Vorgehen zum Aufbau der unter Abschnitt 3.2 und 3.3 vorgestellten Pakete zur domänenspezifischen Modellierung dargestellt und darauf aufbauend die Implementierung von Transformatoren zur losen Toolkopplung entsprechend Abschnitt 3.4.2 und 3.4.3 beschrieben. Schließlich wird in Abschnitt 4.3 das Kapitel zusammengefasst und es werden verbleibende Herausforderungen diskutiert.

4.1 Maschinenbauspezifische Modellierung

In diesem Abschnitt wird beschrieben, wie die maschinenbauspezifische Modellierung auf einer Modellierungsplattform implementiert wird. Dabei werden neben der Umsetzung der Profile und Bibliotheken Einschränkungen für eine einfache Anwendbarkeit in KMUs direkt berücksichtigt, um FA1-I und FA3-I abzudecken. Unterabschnitt 4.1.1 enthält die begründete Auswahl der Modellierungsplattform und die Integration von Basistechnologien für die Umsetzung. Unterabschnitt 4.1.2 zeigt die Integration von MMBSEFE, die Implementierung von OCL-Einschränkungen für MMBSEFE-Elemente und die veranschaulichte Darstellung von Elementen und Ergebnisdatenmarkierungen durch den Einsatz von CSS-Dateien.

4.1.1 Modellierungswerkzeug

Als Modellierungsplattform für die Proof-of-Concepts-Implementierung wird unter den in Anhang A.3.9 aufgezählten MBSE-Plattformen die bereits erwähnte Modellierungsplattform Eclipse Papyrus [93] ausgewählt. Es gibt eine Reihe von Gründen für diese Entscheidung, die im Folgenden genannt werden:

Model-Based Systems Engineering sicherheitskritischer Anwendungen im Maschinenbau mittelständischer Unternehmen © 2024 by Nick Berezowski is licensed under CC BY 4.0

- Papyrus ist eine Open-Source-Software, die kostenlos genutzt werden kann.
- Papyrus ist plattformübergreifend und kann auf Windows-, Mac- und Linux-Betriebssystemen ausgeführt werden.
- Durch die zugrundeliegenden Technologien ist Papyrus eine mächtige Modellierungsplattform, die die Möglichkeit bietet, UML- und SysML-Modelle mit anderen Modellierungserweiterungen und -sprachen, wie MARTE und Business Process Model and Notation (BPMN) zu nutzen und zu verbinden.
- Papyrus bietet eine große Auswahl an weiteren Werkzeugen, einschließlich einer Palette von Zeichenwerkzeugen und einer reichen Bibliothek von Modellelementen.
- Es bietet eine integrierte Entwicklungsumgebung (IDE) für die Modellierung, Implementierung und Überprüfung von Modellen.
- Papyrus verfügt über eine intuitiv zu bedienende Benutzeroberfläche, die eine schnelle Modellierung ermöglicht.
- Papyrus ist in der Lage, die Verwaltung großer Modelle zu unterstützen, die eine große Anzahl von Diagrammen, Elementen und Verbindungen umfassen können.
- Papyrus ist vollständig in Eclipse integriert und kann daher problemlos mit anderen Eclipse-Plug-Ins und -Werkzeugen verwendet werden.
- Papyrus unterstützt die Integration von Code-Generierungswerkzeugen, Code-Vervollständigung und Debugging-Werkzeugen, wodurch die Implementierung von Modellen, beispielsweise in Java oder Funktionsblockdiagramme erleichtert wird.
- Papyrus bietet eine umfangreiche Unterstützung für die Zusammenarbeit, einschließlich Versionskontrolle, Modellvergleich und Modellzusammenführung.
- Papyrus verfügt über eine starke Community, die eine breite Palette von Ressourcen, Dokumentationen und Support bietet.

Papyrus als Modellierungsplattform liefert damit eine etablierte, lauffähige und passende Lösung als Grundlage der Implementierung, insbesondere für Windows-basierte Entwickler-PCs, die von vielen Stakeholdern im Maschinenbau genutzt werden.

Im ersten Schritt wird RAAML in Papyrus integriert. Da es sich bei der Spezifikation lediglich um einen Vorschlag handelt, existiert bisher keine Integration für Papyrus. Die aktuelle Spezifikation auf der OMG-Seite [286] stammt von einem MagicDraw-UML-Exporter, weshalb die Bibliotheken und Profile in einer MagicDraw-Umgebung reimportiert und erneut für den UML2-XMI-Standard exportiert wurden, um sie für den Papyrus-Import bereitzustellen. Fehlende oder verloren gegangene Teile sowie die Aktualisierung der Spezifikation konnten durch

die ausführliche RAAML-Spezifikation [236] rekonstruiert werden, was eine solide Basis für die weitere Entwicklung von MMBSEFE bietet.

Papyrus unterstützt zudem OCL mittels eines integrierten OCL-Editors basierend auf EMF [287], der es ermöglicht, OCL-Ausdrücke direkt in den Modellen zu schreiben, zu validieren und auszuführen.

ISO 80000 kann als vorhandenes Standardpaket direkt über den Import von Bibliotheken integriert werden, um physikalische Größen auf der Metamodellebene und Anwendermodellebene zu modellieren. Leider gibt es kein Plug-In, das ISO 80000 in Papyrus abbilden kann. Daher wurde zunächst die Spezifikation von der OMG-Seite [288] als Grundlage angenommen, was sich jedoch als problematisch erwies, da der XMI-Standard nicht mit dem UML2-XMI-Standard von Papyrus kompatibel ist und keine Importfunktionen existieren. Durch eine Forenrecherche wurde jedoch eine UML2-XMI-Version der ISO 80000-Bibliothek [289] in SysML 1.4 gefunden, die für den vorgesehenen Einsatz in SysML 1.6 integriert werden konnte.

MARTE konnte hingegen direkt durch die Installation eines MARTE-Plug-Ins [290] importiert werden, was die Nutzung von Aspekten für Echtzeitsysteme und die Erweiterung im Rahmen der Modellierung für Wahrscheinlichkeitsverteilungen, wie in Abschnitt 3.2.3 beschrieben, ermöglicht.

Durch diese Integrationen in Papyrus ist es möglich, Modelle zu erstellen, die vollständiger und konsistenter mit den Anforderungen der Stakeholder umgehen, sowohl bei der Metamodellumsetzung, als auch bei der Anwendungsmodellierung.

4.1.2 Maschinenbauspez. Bibliotheken und Profile (MMBSEFE)

Ein Profil in Papyrus stellt ein EMF-Modell dar, das dem UML Ecore Metamodell entspricht. Für die Erstellung eines neuen Profils müssen die Stereotypen, Eigenschaften, Assoziationen und andere Elemente aus MMBSEFE so umgesetzt werden, dass die Elemente ihren domänen- und Stakeholder-spezifischen Modellierungen sowie deren Eigenschaften und Beziehungen zwischen den Elementen entsprechen. Papyrus ermöglicht die Erstellung von UML-Profilen mithilfe des Profildiagrammeditors, der eine Palette mit allen erforderlichen Elementen bereitstellt. Die Eigenschaften der Elemente können dann über die Eigenschaftsansicht festgelegt werden. Jeder Stereotyp in einem Profil erweitert ein UML-Konzept, wie bereits als Metaklasse bezeichnet, welches durch den Import der Metaklasse und das Hinzufügen geeigneter Erweiterungslinks zwischen Stereotypen und Metaklassen-Elementen erreicht wird. Nach der Erstellung wird das Profil auf ein UML-Modell angewendet, das der Modellierung des Verhaltens

der Bibliothek dient und als Grundlage für die UML-Modelle der Anwendungsinstanzen verwendet werden kann, auf die die definierten Stereotypen angewendet werden können. Nur wenn beispielsweise ein Stereotyp das UML::Class-Element erweitert, können Stakeholder ihn auf ausgewählte Instanzen von UML::Class in ihren Modellen anwenden.

Um die Erklärbarkeit der in Abschnitt 3.4.1 angesprochenen Konzepte zur vereinfachten Anwendbarkeit für die Stakeholder in KMUs des Maschinenbaus zu verbessern, sollen diese im Folgenden anhand eines Beispiels veranschaulicht werden. Hier wird das BDD in Abbildung 4.1 und das IBD in Abbildung 4.2 für eine Sicherheitsfunktion zur Beschreibung des sicherheitsbezogenen Blockdiagramms angenommen.

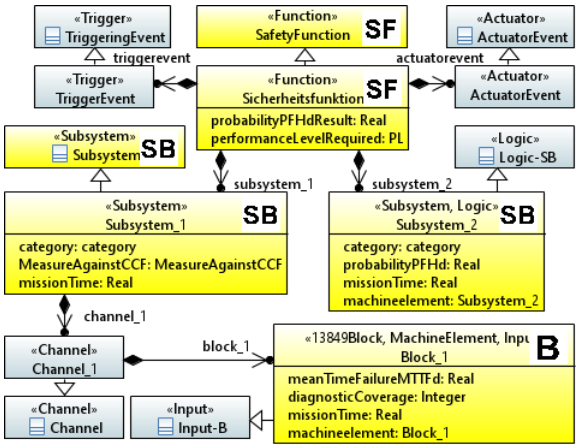


Abbildung 4.1: Beispiel Sicherheitsfunktion BDD

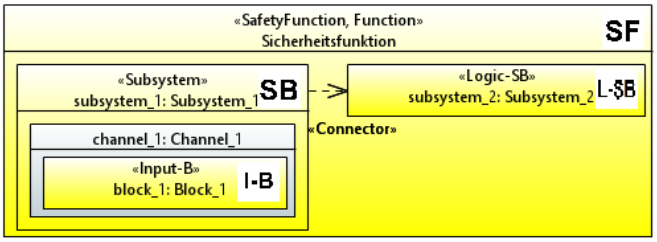


Abbildung 4.2: Beispiel Sicherheitsfunktion IBD

Die aus `SafetyFunction` spezialisierte `Sicherheitsfunktion` besteht aus zwei Subsystemen `Subsystem_1` und `Subsystem_2`, wobei `Subsystem_1` den `Channel_1` mit dem Eingangsblock `Block_1` besitzt. Für `Sicherheitsfunktion` wurden die Attributen `probabilityPFHdResult` für den PFH_D-Ergebniswert und `performanceLevelRequired` für PL_r vorgesehen. `Subsystem_1` verfügt über die Attribute zur Darstellung der Kategorie, der Maßnahmen gegen CCF und der Betriebszeit sowie `Block_1` über die Attribute für $MTTF_D$, DC, Betriebszeit und einem auf sich selbst zugeordneten Maschinenelement. Bei `Subsystem_2` soll es sich um ein gekapseltes Subsystem handeln, dass daher eine Kategorie, PFH_D, eine Betriebszeit und ein auf sich selbst zugeordnetes Maschinenelement besitzt.

Einschränkung durch OCL-Constraints

Eine der Einschränkungen der UML-Profile in Papyrus ist, dass Verbindungen zwischen Stereotypen nur dann als Kanten in einem Diagramm instanziiert werden können, wenn sie ein Konnektor-Meta-Element (z.B. UML::Association) erweitern, so wie auch in der Modellierung, beispielsweise in Abschnitt 3.3.3, beschrieben. Die in Abbildung 4.1 dargestellten Kompositionen oder der `Connector`-Stereotyp in Abbildung 4.2 enthalten keine Informationen über die

Stereotypen, die sie verbinden können, weshalb solche Einschränkungen durch manuelles Schreiben von OCL-Constraints festgelegt werden können. OCL-Constraints sind dabei ein Beispiel für die Flexibilität von OCL, um Zusammenhänge von Klassenattributen zu definieren und zu implementieren, in denen aber auch Berechnungen und Bedingungen von Klassenattributen in UML-Modellen beschrieben werden können, was zur Verbesserung der Modellierung und Implementierung von Prüfmaschinen beiträgt. So kann beispielsweise definiert werden, ob die Quell- und Zielknoten vom richtigen Typ sind, der Konnektor in die richtige Richtung zeigt oder alle notwendigen Attribute gesetzt wurden. Zusätzlich wird so die Usability der domänenspezifischen Profile verbessert, die Einarbeitungszeit durch Feedback zur korrekten Modellierung verkürzt und die Fehleranfälligkeit bei der Nutzung von UML und SysML für MMBSEFE verringert. Um dem entgegenzuwirken, werden die unter Abschnitt 3.2 und 3.3 beschriebenen Elemente durch OCL-Constraints auf ihre Konformität überprüft.

Für die beispielhafte Sicherheitsfunktion aus Abbildung 4.1 und Abbildung 4.2 bedeutet dies, dass die in Listing 4.1 bis Listing 4.8 beschriebenen OCL-Constraints für die Profilelemente und die in Listing 4.9 bis Listing 4.13 beschriebenen OCL-Constraints für die Bibliothekselemente vergeben werden. Optionale Bestandteile, welche die Bibliotheken, wie in Abschnitt 3.3.2 und 3.3.3 angesprochen, auch im Kontext von RAAML ohne MMBSEFE nutzbar machen, sind grau hinterlegt.

```
1  --Function stereotype can only be applied on any class specialized from SafetyFunction or
   13849Function from SBBM-Library
2  self.base_Class->asSet()->closure(general).name->includes('SafetyFunction') or
   self.base_Class->asSet()->closure
```

Listing 4.1: OCL-Constraint Function von SBBM-Profil

```
1  if not self.base_Class->isEmpty() then
2      --Trigger stereotype can only be applied on any class specialized from TriggerEvent
       from SBBM-Library
3      self.base_Class->asSet()->closure(general).name->includes('TriggerEvent')
4  else
5      --Trigger stereotype can only be applied on any property whose type is specialized from
       TriggerEvent from SBBM-Library
6      self.base_Property.type->asSet()->closure(general).name->includes('TriggerEvent')
7  endif
```

Listing 4.2: OCL-Constraint Trigger von SBBM-Profil

```
1  if not self.base_Class->isEmpty() then
2      --Actuator stereotype can only be applied on any class special-ized from ActuatorEvent
       from SBBM-Library
3      self.base_Class->asSet()->closure(general).name->includes('ActuatorEvent')
4  else
5      --Actuator stereotype can only be applied on any property whose type is specialized
       from ActuatorEvent from SBBM-Library
6      self.base_Property.type->asSet()->closure(general).name->includes('ActuatorEvent')
7  endif
```

Listing 4.3: OCL-Constraint Actuator von SBBM-Profil

```
if not self.base_Class->isEmpty() then
    --Subsystem stereotype can only be applied on any class specialized from Subsystem from
    SBBM-Library
    self.base_Class->asSet()->closure(general).name->includes('Subsystem')
else
    --Subsystem stereotype can only be applied on any property whose type is specialized
    from Subsystem from SBBM-Library
    self.base_Property.type->asSet()->closure(general).name->includes('Subsystem')
endif
```

Listing 4.4: OCL-Constraint Subsystem von SBBM-Profil

```
if not self.base_Class->isEmpty() then
    --Input stereotype can only be applied on any class specialized from Input-SB, Input-B
    or Input-E from SBBM-Library
    self.base_Class->asSet()->closure(general).name->includes('Input-SB') or
    self.base_Class->asSet()->closure(general).name->includes('Input-B') or
    self.base_Class->asSet()->closure(general).name->includes('Input-E')
else
    --Input stereotype can only be applied on any property whose type is specialized from
    Input-SB, Input-B or Input-E from SBBM-Library
    self.base_Property.type->asSet()->closure(general).name->includes('Input-SB') or
    self.base_Property.type->asSet()->closure(general).name->includes('Input-B') or
    self.base_Property.type->asSet()->closure(general).name->includes('Input-E')
endif
```

Listing 4.5: OCL-Constraint Input von SBBM-Profil

```
if not self.base_Class->isEmpty() then
    --Logic stereotype can only be applied on any class specialized from Logic-SB, Logic-B
    or Logic-E from SBBM-Library
    self.base_Class->asSet()->closure(general).name->includes('Logic-SB') or
    self.base_Class->asSet()->closure(general).name->includes('Logic-B') or
    self.base_Class->asSet()->closure(general).name->includes('Logic-E')
else
    --Logic stereotype can only be applied on any property whose type is specialized from
    Logic-SB, Logic-B or Logic-E from SBBM-Library
    self.base_Property.type->asSet()->closure(general).name->includes('Logic-SB') or
    self.base_Property.type->asSet()->closure(general).name->includes('Logic-B') or
    self.base_Property.type->asSet()->closure(general).name->includes('Logic-E')
endif
```

Listing 4.6: OCL-Constraint Logic von SBBM-Profil

```
if not self.base_Class->isEmpty() then
    --Channel stereotype can only be applied on any class specialized from Channel-B from
    SBBM-Library
    self.base_Class->asSet()->closure(general).name->includes('Channel')
else
    --Channel stereotype can only be applied on any property whose type is specialized from
    Channel-B from SBBM-Library
    self.base_Property.type->asSet()->closure(general).name->includes('Channel')
endif
```

Listing 4.7: OCL-Constraint Channel von SBBM-Profil

```
--supplier and client of Connector must be a 13849Block
13849Block.allInstances().base_Class->includesAll(self.base_Dependency.client) and
13849Block.allInstances().base_Class->includesAll(self.base_Dependency.supplier)
```

Listing 4.8: OCL-Constraint Connector von SBBM-Profil

```
--SafetyFunction should have a performanceLevelRequired, a
triggeringEvent, an actuatorEvent and a subsystem
self.performanceLevelRequired<>null and
self.actuatorEvent<>null and
self.triggeringEvent<>null and self.attributes->select(a | a.name.startsWith('subsystem'))-
>notEmpty())
```

Listing 4.9: OCL-Constraint SafetyFunction von SBBM-Bibliothek

```
--Subsystem should have a category
self.category<>null
```

Listing 4.10: OCL-Constraint Subsystem von SBBM-Bibliothek

```
1  --Logic-SB should have a machineelement
2  self.machineelement<>null
```

Listing 4.11: OCL-Constraint Logic-SB von SBBM-Bibliothek

```
1  --Channel should have a block
2  self.attributes->select(a | a.name.startsWith('block'))->notEmpty()
```

Listing 4.12: OCL-Constraint Channel von SBBM-Bibliothek

```
1  --Input-B should have a machineelement
2  self.machineelement<>null
```

Listing 4.13: OCL-Constraint Input-B von SBBM-Bibliothek

Wie ersichtlich, ähneln oder wiederholen sich einige Constraints inhaltlich, wie beim Vergleich von Listing 4.2, Listing 4.3, Listing 4.4 und Listing 4.7, Listing 4.5 und Listing 4.6 oder Listing 4.11 und Listing 4.13 zu sehen. Eine vollständige Auflistung aller tatsächlich eingesetzten OCL-Constraints findet sich in den ausführlichen Funktionsbeschreibungen der Profil- und Bibliothekselemente in Anhang A.9.

Benutzerdefinierte Palette

Durch den vorangegangenen Abschnitt können Stakeholder die Bibliotheken und Profile in ihren Entwicklungsphasen auf SysML-Diagramme relativ einfach anwenden. Ein Stereotyp kann dabei bereits nur auf Instanzen der Meta-Elemente angewendet werden, die er erweitert. Trotzdem bleibt es für Stakeholder wichtig, das Konzept zu verstehen, um sich Meta-Elemente und Basisstereotypen zu merken, die vom Profil erweitert werden. Papyrus bietet drei Möglichkeiten zur Erstellung einer benutzerdefinierten Palette, um dieses Problem zu umgehen und UML-Elemente in einem Schritt zu erstellen und Stereotypen auf sie anzuwenden. Die erste Option ist die Nutzung der Anpassungsfunktion, um benutzerdefinierte Paletten zu erstellen. Allerdings muss dies jedes Mal manuell durchgeführt werden und kann nicht gemeinsam genutzt werden. Die zweite Option ist die manuelle Definition einer XML-basierten Palettenkonfigurationsdatei, die jedoch von Eclipse nicht empfohlen wird, da sie nicht die volle Nutzung der Papyrus-Funktionalität ermöglicht und auf einem veralteten Framework basiert. Die dritte und angewendete Option beinhaltet die Erstellung eines Papyrus-Editors, der mit dem UML-Profil assoziiert ist, was die manuelle Erstellung einer Reihe von miteinander verbundenen Modellen und Artefakten beinhaltet, einschließlich eines Paletten-Konfigurationsmodells und eines ElementTypesConfiguration-Modells, das verwendet wird, um Stereotypen im UML-Profil mit seinem UML-Meta-Element und der konkreten Syntax des Meta-Elements zu assoziieren.

Veranschaulichte Darstellung durch CSS-Konfiguration

Die Definition von benutzerdefinierten Shapes für die instanziierten Stereotypen ist eine weitere Möglichkeit, um die Usability zu verbessern. Hierfür können Scalable Vector Graphics (SVG)-Shapes während der Profilerstellung und –anwendung an Stereotypen gebunden

werden. Um sicherzustellen, dass die Darstellung der Elemente eindeutig ist, bietet es sich hierfür an, die Elemente so zu gestalten, wie es in den Spezifikationen der Methoden vorgesehen ist. In einigen Modellierungswerkzeugen, wie MagicDraw, ist es möglich, Stereotypen direkt Icons zuzuweisen. In Papyrus bietet sich zusätzlich die Möglichkeit, basierend auf dem UML2-XMI-Standard von Eclipse, die Darstellung von Stereotypen über CSS-Dateien zu steuern. Um SVG Shapes sichtbar zu machen, müssen benutzerdefinierte Stilregeln in den CSS-Dateien beschrieben werden, da sich die an einem Stereotyp gebundene SVG-Form standardmäßig mit der Standardform des Basis-Meta-Elements überlappt. Eine CSS-Datei wird dabei nur einmal geschrieben, ist einfach zu editieren und kann jedes Mal bei jedem neuen Diagramm manuell geladen werden. So entsteht die Möglichkeit einer einfacheren Wartung der Darstellung und somit eine gute und anpassbare Integration in das Gesamtsystem.

Um die Struktur einer CSS-Datei zu veranschaulichen, dient der Stereotyp `Logic-SB` aus Abbildung 4.2 der Beispielsicherheitsfunktion. Dieser entsteht im Anwendungsprofil aus dem Bibliothekselement, wie in Abschnitt 3.4.1 beschrieben. Listing 4.14 gibt hierbei die zur Darstellung verwendeten Teile aus der CSS-Datei wieder. Die ersten beiden Regeln deaktivieren Compartments (Zeile 1 bis 4) sowie die Anzeige der Form und der Titel (Zeile 5 bis 8) für alle Logic-Elemente, während die dritte Regel (Zeile 9 bis 20) verschiedene Stileigenschaften für Logic-SB enthält. Das Element soll keine UML-Symbolik enthalten (Zeile 10), eine gelbe Hintergrundfarbe (Zeile 11) und einen sichtbaren Rand aufweisen (Zeile 14). Das neue Symbol des Elements soll aus der SVG-Datei „Logic-SB.svg“ stammen (Zeile 15 bis 17), die in den Ressourcen des Profils hinterlegt ist, und seine Richtung wird auf 2 (oben rechts) festgelegt (Zeile 18). Zudem wird ein zu langer Name des Elements automatisch umgebrochen (Zeile 19).

```
1  [appliedStereotypes~="Logic"] > Compartment{
2       visible:false; /* Set the visibility of the compartment to false */
3       textAlignment:center; /* Center align the text within the compartment */
4  }
5  [appliedStereotypes~="Logic"] > Compartment[type="compartment_shape_display"]{
6       visible:false; /* Set the visibility of the shape from compartments to false */
7       showTitle:false; /* Do not display the title of the compartment */
8  }
9  [appliedStereotypes~="Logic-SB"]{
10      elementIcon:false; /* Do not show element icon */
11      fillColor:yellow; /* Set background color of element to yellow */
12      displayIcon:false; /* Do not display icon */
13      displayHeader:false; /* Do not display header */
14      displayBorder: true; /* Show border around element */
15      svgFile:"plattform:/resource/com.nb.raaml.13849.uml.profiles/shapes/Logic-SB.svg";
        /* SVG file for the element */
16      followSVGSymbol:true; /* show SVG symbol for shape */
17      shapeVisibility:true; /* set shape to visible */
18      shapeDirection:2; /* show shape on top right */
19      isNameWrap:true /* Enable text wrap for name */
20 }
```

Listing 4.14: CSS-Eigenschaften für UnrepairableBE

CSS-Definitionen für weitere Stereotypen und Elemente sind nach dem gleichen Schema aufgebaut, wobei sich kleine Unterschiede ergeben, so ist beispielsweise für Gates zum Aufbau von DFTs `displayBorder` gleich `false`.

Ein weiterer Vorteil beim Einsatz von CSS-Dateien entsteht für die Ergebnisdarstellung. Werden Ergebnisse aus den Analysewerkzeugen zurücktransformiert, wird das Nichterfüllen einer Anforderung durch eine Analyse visuell unterstützt. Über den in den Stereotypen zusätzlich vergebenen ValueType `result` in Abbildung 4.3 können fehlgeschlagene und bestandene Elemente farblich hinterlegt werden. Hierfür wird über die CSS-Erscheinungsbild-Eigenschaften in Listing 4.15 das Aussehen verändert.

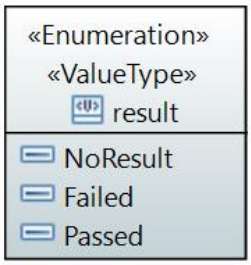


Abbildung 4.3 Attribut zur farblichen Kennzeichnung der Rückkopplungsergebnisse

```
1  Class[result=Passed] {
2       fillColor:#A6C198; /* Sets every class
        with result=Passed to green */
3  }
4  Property[result=Passed] {
5       fillColor:#A6C198; /* Sets every
        property with result=Passed to green */
6  }
7  Class[result=Failed] {
8       fillColor:#E3A49C /* Sets every class
        with result=Passed to red */
9  }
10 Property[result=Failed] {
11      fillColor:#E3A49C /* Sets every property
        with result=Passed to red */
12 }
```

Listing 4.15: CSS -Eigenschaften zur farbliche Kennzeichnung der Ergebnisse

Der CSS-Code in Listing 4.15 definiert Stilregeln für bestimmte Klassen und Eigenschaften für die Werte `Passed` und `Failed`. Für den Wert `Passed` wird die Hintergrundfarbe einer UML-Klasse im BDD oder eines UML-Property im IBD mittels `fillColor` auf den hexadezimalen Wert `#A6C198` (Hellgrün) und für den Wert `Failed` auf den Wert `#E3A49C` (Hellrot) gesetzt. Die Attribute wurden jeweils in `FunctionalSafetyRequirement`, `DFTree`, `DFTTopEvent`, `SafetyFunction`, `Subsystem`, `Block` und `Element` vergeben.

4.2 Risikobeurteilung

In diesem Abschnitt wird das Vorgehen zur Umsetzung der Stakeholder-spezifischen Modelle in Abhängigkeit zur Risikobeurteilung und die Implementierung der Analysemethoden beschrieben. Der Abschnitt deckt somit die übrigen Teilaspekte zu FA1-I in Form eines Ablaufes und alle Aspekte zu FA2-I ab. Die Zusammenführung der Aspekte begründet sich in der Zusammengehörigkeit, um eine kohärente Risikobeurteilung umzusetzen. Daher wird zunächst in Unterabschnitt 4.2.1 das Vorgehen zur Umsetzung der Modelle beschrieben. Unterabschnitt 4.2.2 gibt folgend einen tiefen Einblick in die Implementierung der losen Kopplung für DFTs,

Unterabschnitt 4.2.3 zeigt die Implementierung des Transformators der SBBM und Unterabschnitt 4.2.4 die Implementierung zur Berechnung von Risikographen.

4.2.1 Vorgehen

Um den unter Abschnitt 2.1.3 in Abbildung 2.3 und Abbildung 2.4 illustrierten Ablauf einer Risikobeurteilung abzubilden, wurde unter Abschnitt 3.2.11 das ISO12100&ISO13849-Paket modelliert. Um die Basis für eine Evaluation zu schaffen, fehlt jedoch ein Konzept, das MMBSEFE mit dem Prozess in Verbindung bringt und die Modellierung aus Anwendersicht darstellt. Abbildung 4.4 und Abbildung 4.5 stellen diese Verbindung her.

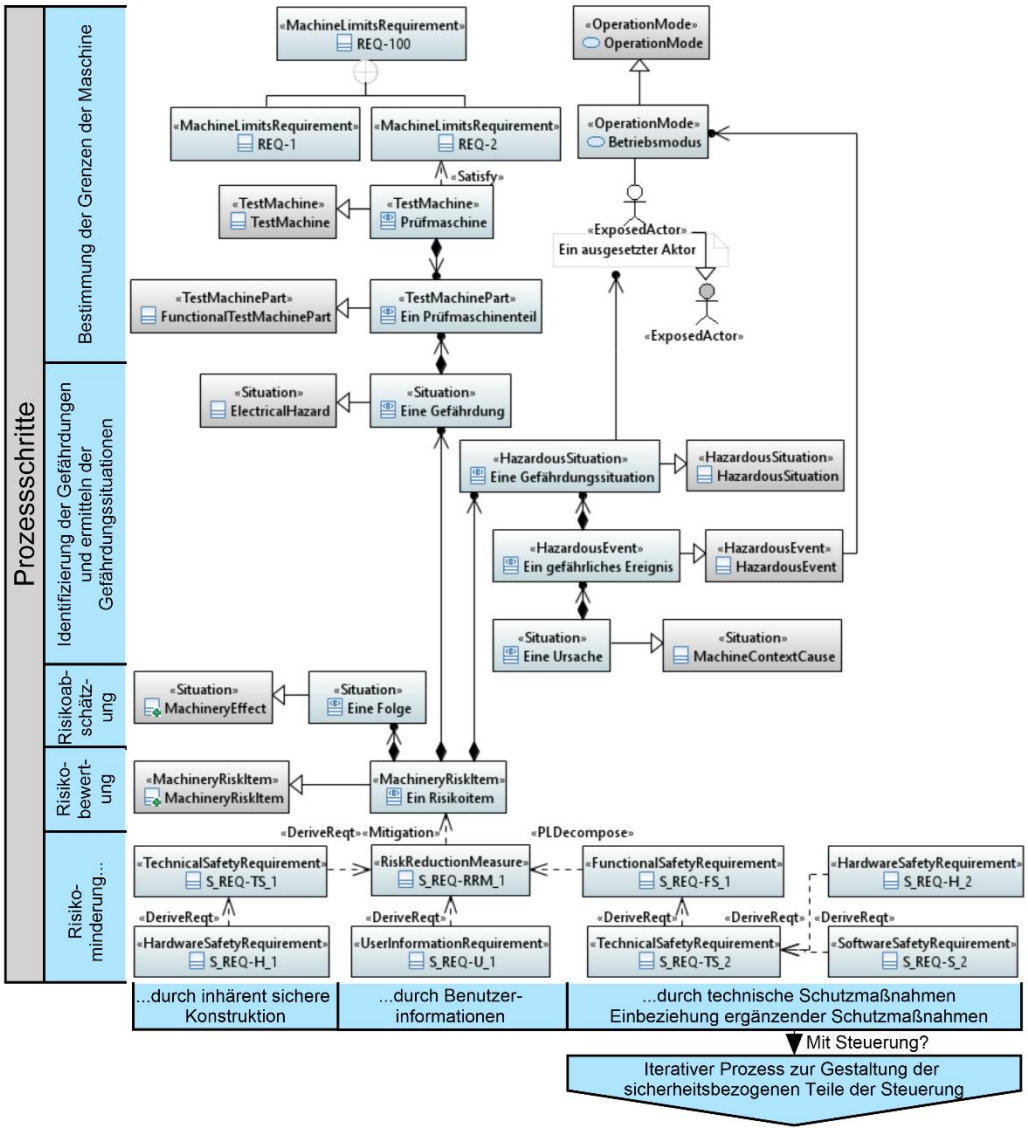


Abbildung 4.4: MMBSEFE-Ablauf zur Risikobeurteilung

Im ersten Prozessschritt in Abbildung 4.4 findet sich die Bestimmung der Grenzen der Maschine, welcher die Anforderungen an die Prüfmaschine festlegt, die Prüfmaschine selbst mit Prüfmaschinenteilen verbindet und Anwendungsfälle für Aktoren definiert, um alle

notwendigen Informationen zur Identifizierung von Gefährdungen und Situationen zu liefern. Der Sicherheitsingenieur nutzt diese Informationen, um basierend auf den identifizierten Gefahren im zweiten Schritt die entsprechenden Situationen zu bestimmen, jedem Fall ein auslösendes Ereignis zuzuordnen und die Ursachen für jedes Ereignis zu bestimmen. Während der Risikoabschätzung im dritten Schritt werden die Folgen und Faktoren zur Bestimmung der Risikoprioritätszahl ermittelt. Zusammengefasst im Risikoitem findet die Risikobewertung statt, die dann verschiedene risikomindernde Maßnahmen mit sich führen kann. Die Implementierung zur externen Risikoberechnung findet sich unter Abschnitt 4.2.4.

Schutzmaßnahmen zur inhärent sicheren Konstruktion und technische Schutzmaßnahmen ohne Software bestehen lediglich aus abgeleiteten Hardwareanforderungen, während Benutzerinformationsanforderungen lediglich der Dokumentation und Beschilderung der Anlage dienen. Falls es sich um technische Schutzmaßnahmen mit einer Steuerung handelt, ist der Aufbau in Abbildung 4.5 zu berücksichtigen.

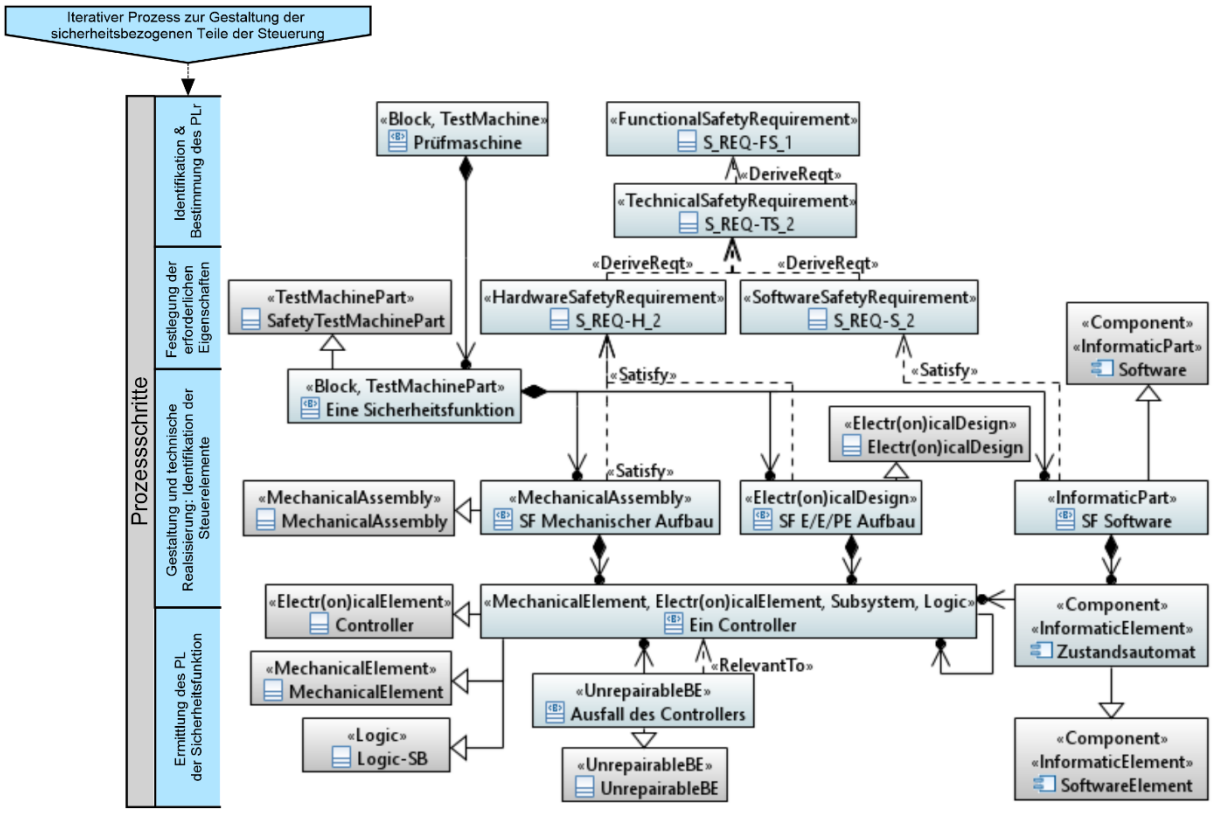


Abbildung 4.5: MMBSEFE-Ablauf Bestimmung softwaregesteuerte Risikominderung

Die Bestimmung der Risikominderung bei softwaregesteuerten Sicherheitsfunktionen beginnt mit den spezifischen Anforderungen aus der Risikobeurteilung. Diese beschreiben die Identifikation, Festlegung der erforderlichen Eigenschaften und Bestimmung des PL_r durch die Anforderungen. Basierend auf diesen Anforderungen wird das Modell der Prüfmaschine durch sicherheitsbezogene Teile, also den Sicherheitsfunktionen, und enthaltene mechanische, E/E/PE

und Softwareelemente erweitert, um die Gestaltung und technische Realisierung umzusetzen. In Abbildung 4.5 überlappt sich dies mit der Ermittlung des PL einer Sicherheitsfunktion, da für jedes relevante Element, wie von Abschnitt 2.2.1 gefordert, gleichzeitig $MTTF_D$, DC, CCF usw. in Bezug zu SBBM und λ_D in Bezug zu DFTs bestimmt werden. SBBM verwendet für Blöcke und Subsysteme das gleiche Element und bildet somit nach Abschnitt 3.3.3 eine Assoziation auf sich selbst. DFTs bilden stattdessen für dynamische Basiselemente nach Abschnitt 3.3.2 eine Assoziation auf das unabhängig entwickelte Maschinenelement. Alternativ wird in Abbildung 4.5 für dynamische Basiselemente ebenso die originale Modellierung über `RelevantTo` aus RAAML gezeigt, insofern die Analysemethoden ohne die Systemmodelle von MMBSEFE, wie unter Abschnitt 3.3.2 und 3.3.3 angesprochen, genutzt werden. Die externe Verifikation eines sicherheitsbezogenen Blockdiagramms und eines dynamischen Fehlerbaums finden sich in Abschnitt 4.2.2 und 4.2.3.

Nach der Validierung und dem Nachweis der ausreichenden Risikominderung sind die Modelle in den Entwicklerwerkzeugen umzusetzen. Die Umsetzung der Sicherheitsfunktionen in Form von CAD-Zeichnungen und Schaltplänen fällt jedoch, wie bereits beschrieben, nicht in den Umfang dieser Arbeit. Für den SPS-Code handelt es sich durch den Aufbau des Entwurfs in Zustandsautomaten bei entsprechenden Transformatoren um bereits vorhandene Standardlösungen oder um reine Transformatorentwicklungen. Dass die allgemeine Codegenerierung aus UML-Zustandsdiagrammen etabliert ist, zeigt dabei Sunitha et al. [277], die einen Überblick zum Aufbau von Codegenerierungswerkzeugen zur automatischen Codegenerierung aus UML-Zustandsdiagrammen geben. Zusätzlich können bereits verschiedene praktische Lösungen zur Codegenerierung aus Zustandsautomaten in [278] gefunden werden. Zur Generierung von SPS-Code aus UML-Zustandsdiagrammen existieren ebenso bereits Veröffentlichungen, wie Witsch et al. [197], Fantuzzi et al. [279] oder Vogel-Heuser et al. [280], die auf die Transformationen eingehen und Lösungen für verschiedene SPS-Programmiersprachen, wie Ablaufsprache oder Kontaktplan darstellen. Für die Entwicklung von sicherheitsbezogener Anwendersoftware (SRASW) bei Beckhoff-Komponenten auf Basis von UML-Zustandsdiagrammen innerhalb Papyrus bietet sich beispielsweise die Transformation in die TwinCAT Safety PLC Umgebung an, die den Code zur Anwendungsentwicklung in der C++ basierten Hochsprache Safety C [161] realisiert. Über ein C++-Code-Generierungs-Plug-In lässt sich C++-Code aus Papyrus-Modellen erzeugen, der dann als Grundlage für Safety C dient. Aspekte der Codegenerierung werden daher aus wissenschaftlicher Sicht als weitgehend gelöst angesehen und in dieser Arbeit nicht weiter betrachtet.

4.2.2 DFTs

Über das in Abschnitt 3.3.2 gezeigte Paket wurden die Bibliothek und das Profil für DFTs in Papyrus entwickelt. Zur Umsetzung der losen Toolkopplung wurden unter Nutzung der Stereotypen und Attribute, deren Verwendung in der DFT-Bibliothek bereitgestellt wird, das in Abschnitt 3.4.1 vorgestellte Konzept zur vereinfachten Anwendbarkeit und das in Abschnitt 3.4.2 entwickelte Architekturkonzept implementiert. Hierbei werden die Modelle über XMI-Transformatoren in ein externes Analysewerkzeug exportiert, dort die Analyse durchführt und die Analyseergebnisse wieder in die Modellierungsumgebung importiert. Zunächst wurde ein Werkzeug für die Analyse der DFTs ausgewählt, nachfolgend das Architekturkonzept angewendet und dann die Module implementiert.

Werkzeugauswahl

Als Erstes wurden der Fault Tree Analyzer in RAM Commander [241] und das Onlinewerkzeug Fault Tree Analyzer [242] von ALD untersucht, um das Konzept der losen Kopplung zu überprüfen. Für die qualitative Analyse sind die Werkzeuge in der Lage, die Zuverlässigkeit und die erwartete Anzahl von Ausfällen bis zu einem festgelegten Zeitpunkt zu berechnen. RAM Commander kann auch die Verfügbarkeit zu einem bestimmten Zeitpunkt sowie die langfristige durchschnittliche Verfügbarkeit ermitteln. Jedoch ergab sich nach der Entwicklung eines ersten XMI-Transformators für das Onlinewerkzeug und weiteren Recherchen, dass die Verwendung der minimalen Cutsets-Berechnung der Werkzeuge einen negativen Einfluss auf die Genauigkeit der Berechnungen hat. Zudem mangelt es bei RAM Commander an Import- und Exportformaten, die die dynamischen Gate-Typen unterstützen. Die Werkzeuge DFTCalc [243] und Stormchecker [244] arbeiten hingegen mit dem einfach zu interpretierenden Galileo-Inputformat, das unter anderem verschiedene Verteilungsfunktionen unterstützt und einen Wiederherstellungsfaktor enthält, der die Reparierbarkeit darstellt [219]. Eine vollständige Beschreibung des Formats findet sich in Dugan et al. [219]. Die in beiden Werkzeugen eingesetzten Continuous-Time-Markov-Chains (CTMC) können eine detaillierte und möglichst präzise Berechnungsmöglichkeit für DFTs bieten. Im Gegensatz hierzu scheitert die traditionelle Methode zur DFT-Berechnung oft an der kombinatorischen Explosion der Zustandsanzahl. Detaillierte Erklärungen zu angemessenen Anwendungen von CTMC finden sich in [208]. Stormchecker bietet neben der Berechnung der Wahrscheinlichkeit eines Ausfalls über die Zeit für verschiedene Verteilungsfunktionen auch andere Modelle wie Markov-Automaten oder GSPN und die Berechnung der MTTF [244]. Im Gegensatz dazu handelt es sich bei DFTCalc leider um ein in den letzten Jahren wenig aktualisiertes Werkzeug, weshalb letztlich Stormchecker als finales Analysewerkzeug ausgewählt wurde. Eine Anleitung zur Anwendung findet sich in [291].

Implementierung

Wie in Abbildung 4.6 dargestellt, teilt sich die Implementierung, ausgehend vom Architekturkonzept in Abschnitt 3.4.2 Abbildung 3.26, in die fünf Module XMI-Export und -Transformation in LabVIEW, Transformation in Analysewerkzeug-Inputformat für Stormchecker, Analysewerkzeug-Ausführung in Stormchecker, Analyseergebnisverarbeitung für Stormchecker und XMI-Transformation zu Systemmodell.

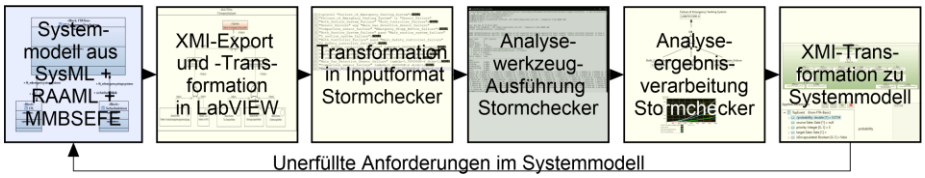


Abbildung 4.6: Implementierungskonzept zur Toolkopplung von DFTs

Im Folgenden finden sich die Struktur und der Code für die verschiedenen Module.

Analysemodellexport und Transformation

Zur Umsetzung des XMI-Exports von System- und Analysemodellen ins Galileo-Inputformat von Stormchecker wurde der Einsatz der standardmäßigen, aber umständlich zu benutzenden, LabVIEW-Bibliotheken zum Lesen und Schreiben von XML-Schemata und die Möglichkeit, XML-Parser aufzubauen, verworfen. Stattdessen wurde das Virtual Instrument (VI)-Paket „EasyXML Toolkit for LabVIEW“ [292] von JKI eingesetzt. Der Hauptvorteil besteht darin, dass EasyXML nicht das von NI definierte LabVIEW-Datenschema verwendet, sondern, wie in Abbildung 4.7 für die Stereotypen im XMI-Format in Listing 4.16 veranschaulicht, die Datenbezeichnungen als XML-Entitätsnamen nutzt. Dies ermöglicht es, beliebige XML-Daten mit XML-Entitätsattributen schnell und einfach über zusammengesetzte Datenstrukturen (LabVIEW-Cluster), vergleichbar mit struct in C, zu schreiben und zu lesen.

```
420 …
421 <DFT:PAND xmi:id="_T75IoLzREeyXq7bdefCi2A"
    sourceEvent="_M4CfgL4-Eey_DbqpSfFQcQ …"
    targetEvent="_M3BhuG5_Aye_BuhnFuOqfZ"
    base_Property="_2JXJUKheEeyfZMbPrmskNA"/>
422 <DFT:UnrepairableBE
    xmi:id="_pVWX0MoqEeyuVo6QDHG6hw"
    base_Property="__fL14KbBEey6nrYj_IEODA"
    sourceGate="_T75IoLzREeyXq7bdefCi2A"
    distribution="_Hjkl86GRghrRu7FuCkxGh8"… />
423 <DFT:Exponential
    xmi:id="_Hjkl86GRghrRu7FuCkxGh8"
    base_Property="_RWA_8FVKEe2w9MOYmNroAA"
    failureLambda="1.0E-8"… />
424 …
```

Listing 4.16: XMI-Format Stereotypes in Papyrus

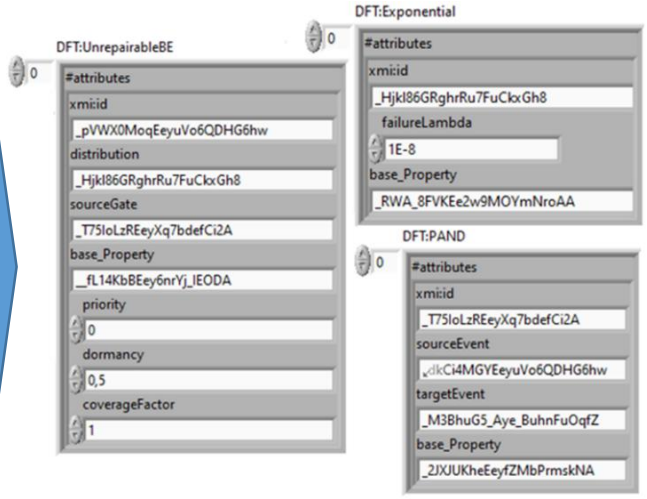


Abbildung 4.7: XMI-Cluster in LabVIEW

Es wurden sechs Hauptfunktionen für den Export und die Transformation entworfen und in LabVIEW G-Code implementiert. Im Folgenden wird die Implementierung ausführlich beschrieben, um die Zusammenhänge für mit G-Code weniger vertrauten Informatikern nachvollziehbar zu gestalten. Grundsätzlich handelt es sich um eine datenflussgesteuerte Sprache, die in gewohnter Leserichtung aufgebaut wird.

Die erste Funktion in Abbildung 4.8 beinhaltet den XMI-Export (1), wie durch Abbildung 4.7 veranschaulicht, und einen Import als zeilenbasierte Textdatei (2) in LabVIEW, um im Folgeschritt die Klassifikatoren-Namen zu bestimmen.

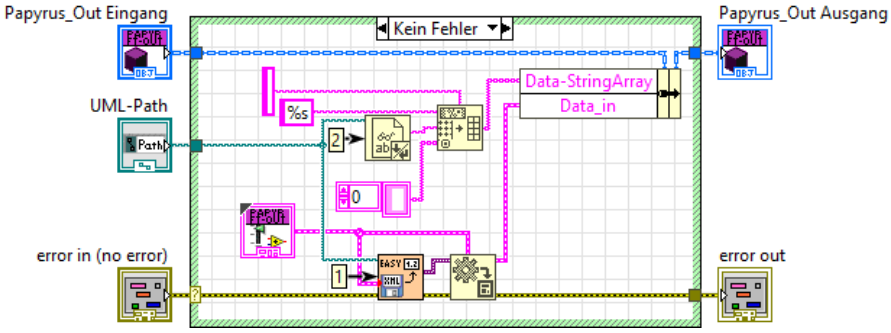


Abbildung 4.8: XMI-Export in LabVIEW - G-Code

In der zweiten Funktion, abgebildet in Abbildung 4.9, werden die Namen aller Klassifikatoren und `Violate`-Beziehungen zu Anforderungen eingelesen (1), der zu analysierende Fehlerbaum vom Benutzer ausgewählt (2), sofern mehrere Fehlerbäume enthalten, da Stormchecker nur einen Fehlerbaum gleichzeitig berechnen kann, und die Attribut-Werte des dynamischen Fehlerbaums ausgelesen (3). Das Einlesen der Namen der Klassifikatoren und `Violate`-Beziehungen findet sich für einen besseren Überblick unter Anhang A.10.1 in Abbildung A.24.

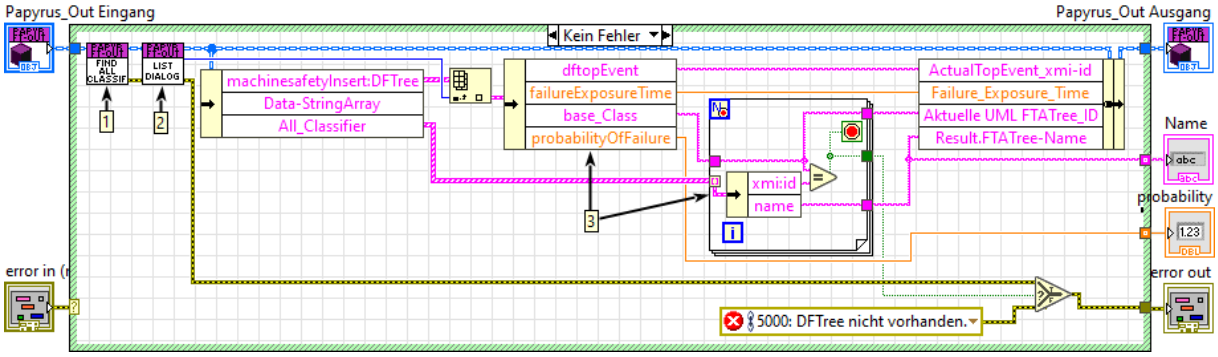


Abbildung 4.9: Fehlerbaum bestimmen und auslesen - G-Code

In einer ersten Version wurden die Klassifikatoren und `Violate`-Beziehungen über das XMI-Schema-Cluster eingelesen, was sich aufgrund der möglichen Verschachtelungstiefe als relativ umständlich erwies. In der aktuellen Version werden die Klassifikatoren-Namen und XMI-IDs stattdessen aus dem eingelesenen zeilenbasierten String-Array geparst.

Die dritte Funktion liest das dynamische Top-Event aus, bestimmt anhand des Attributs `targetGate` das verbundene Gate und anhand des Attributs `sourceGate` der Intermediate-Events und Basic-Events deren Eingangsereignisse, ähnlich der Vorgehensweise in Abbildung 4.9. Genauso bildet sich die vierte Funktion, die die gleiche Aufgabe für die Intermediate-Events übernimmt. Die letzte Funktion dient der Korrelation der Basic-Events mit den entsprechenden Verteilungsfunktionen und speichert die relevanten Werte im Zwischenformat.

Analyseinputformat

Stormchecker bietet das Galileo-Format, das, wie im Folgenden in Listing 4.17 anhand des Beispiels in Abbildung 4.10 gezeigt, eine einfache zeilenbasierte Aufstellung des Fehlerbaums ermöglicht.

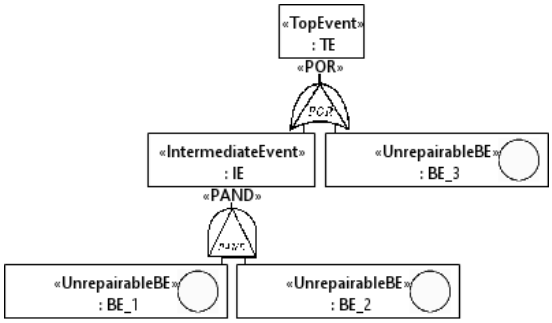


Abbildung 4.10: Beispiel Fehlerbaum

```
1 toplevel "TE";
2 "TE" por "IE" "BE_3";
3 "IE" pand "BE_1" "BE_2";
4 "BE_1" lambda=0.5 cov=0.0 res=0.0 dorm=0.5;
5 "BE_2" lambda=0.5 cov=0.0 res=0.0 dorm=0.5;
6 "BE_3" lambda=1.0 cov=0.0 res=0.0 dorm=0.5;
```

Listing 4.17: Fehlerbaum in Galileoformat

Aufgrund dieses einfach zu interpretierenden Formats besteht das Modul lediglich aus drei Funktionen, wie Abbildung 4.11 verdeutlicht.

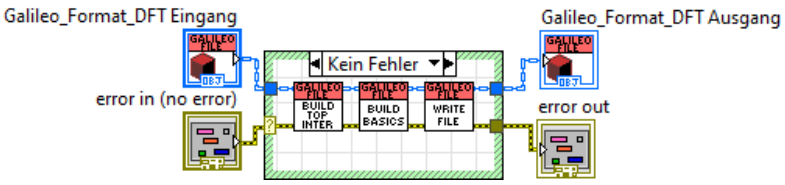


Abbildung 4.11: Fehlerbaumtransformation in Galileo-Format - G-Code

Die erste Funktion erstellt die Zeilen für Top- und Intermediate-Events, die zweite Funktion erstellt die Basic-Events und die dritte Funktion erstellt eine Galileo *.dft-Datei für die Ausführung. Abbildung 4.12 zeigt die Teilfunktion zur Erstellung eines dynamischen Basic-Events mit dem Fall der exponentiellen Verteilung aus dem Zwischenformat.

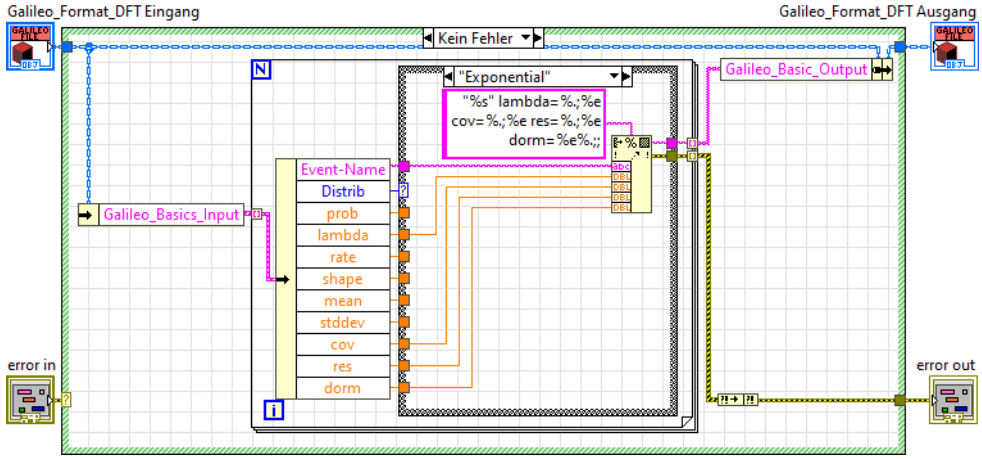


Abbildung 4.12: Basic-Events in Galileo-Format - G-Code

Durch die vorgegebene und leicht zu interpretierende Grammatik aus [219] für das Galileo-Format entsteht so neben dem Format zur Analyse auch ein leicht von Menschen lesbares und interpretierbares Importformat mit geringer Fehleranfälligkeit.

Analyseausführung

Stormchecker beinhaltet verschiedene Ausführungsmöglichkeiten für das Werkzeug. Neben der Einbindung in eine virtuelle Maschine, was für diese Arbeit einen unnötigen Overhead verursachen würde, und der Nutzung des Source-Codes, was einen erhöhten Integrationsaufwand bedeutet, findet sich eine Umsetzung in Docker Containern [291]. Docker Container ähneln virtuellen Maschinen, jedoch mit weniger Overhead. Da diese Einbindung zusätzlich aktuellere Versionen von Storm liefert, wurde diese verwendet. Somit lässt sich, wie in Abbildung 4.13 beschrieben, Stormchecker über einen Konsolenaufruf von LabVIEW aus starten (1), über eine virtuelle Eingabe des entsprechenden Fehlerbaums und von Randbedingungen übergeben (2) und das Ergebnis auslesen (3).

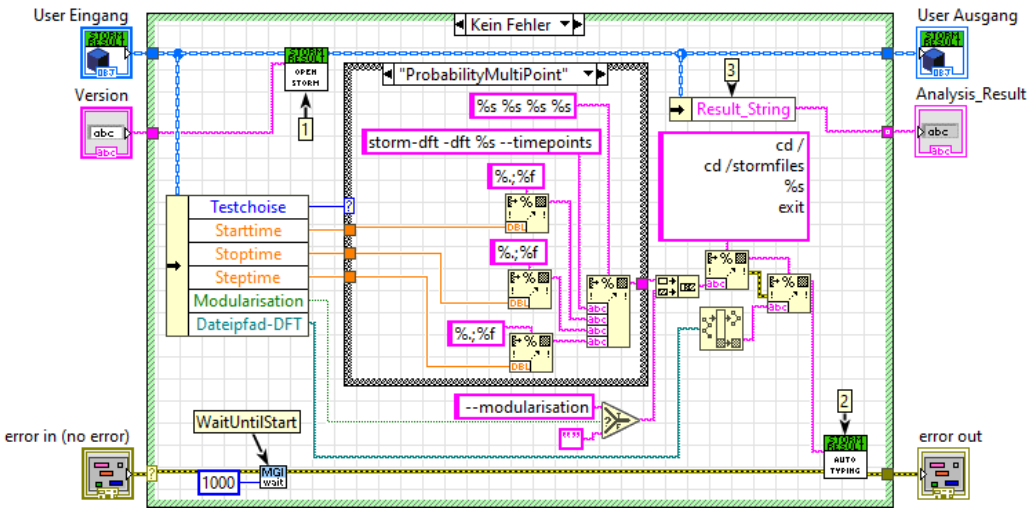


Abbildung 4.13: Ausführen eines Fehlerbaums Storm Docker Container - G-Code

Eine Beispiel-Ergebnisantwort von Stormchecker findet sich in Abbildung 4.14.

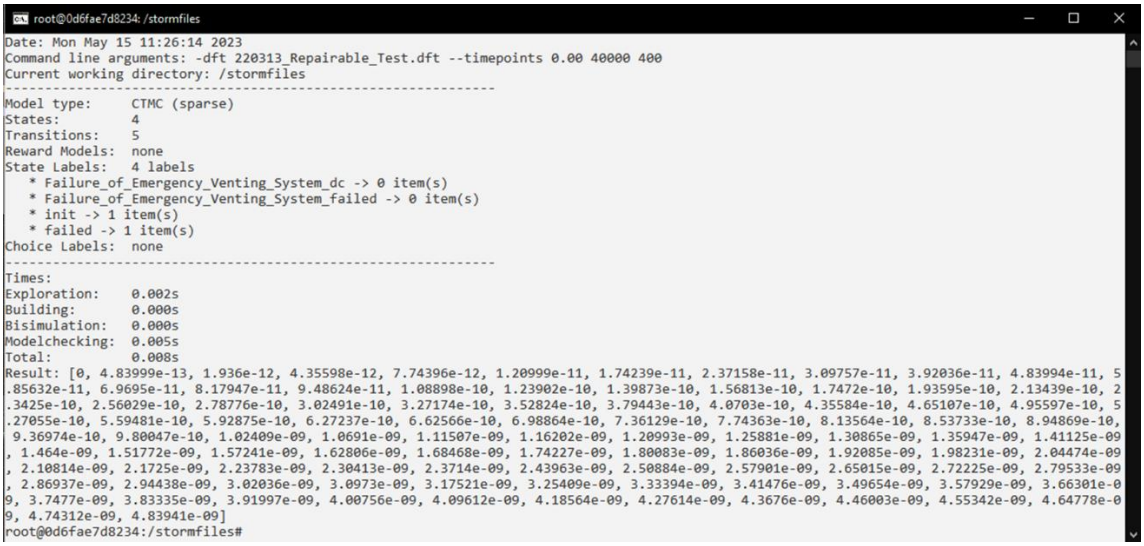


Abbildung 4.14: Beispiel-Ergebnisdaten Stormchecker

Da jeweils nur PF oder MTTF für das Top-Event eines Fehlerbaums in Stormchecker valide und vollständig berechnet werden können, wird dieses Vorgehen für die Berechnung von MTTF des Top-Events wiederholt.

Analyseergebnisverarbeitung

Aus der Antwort des Kommandozeilenaufrufs in Abbildung 4.13 (3) lassen sich die Ergebnisdaten parsen. Im unteren Teil von Abbildung 4.15 wird das Ergebnis für das Top-Event ausgelesen (1) und im oberen Teil, falls vorhanden, die modularisierten Ergebnisse von Intermediate- und Basic-Events geparst (2).

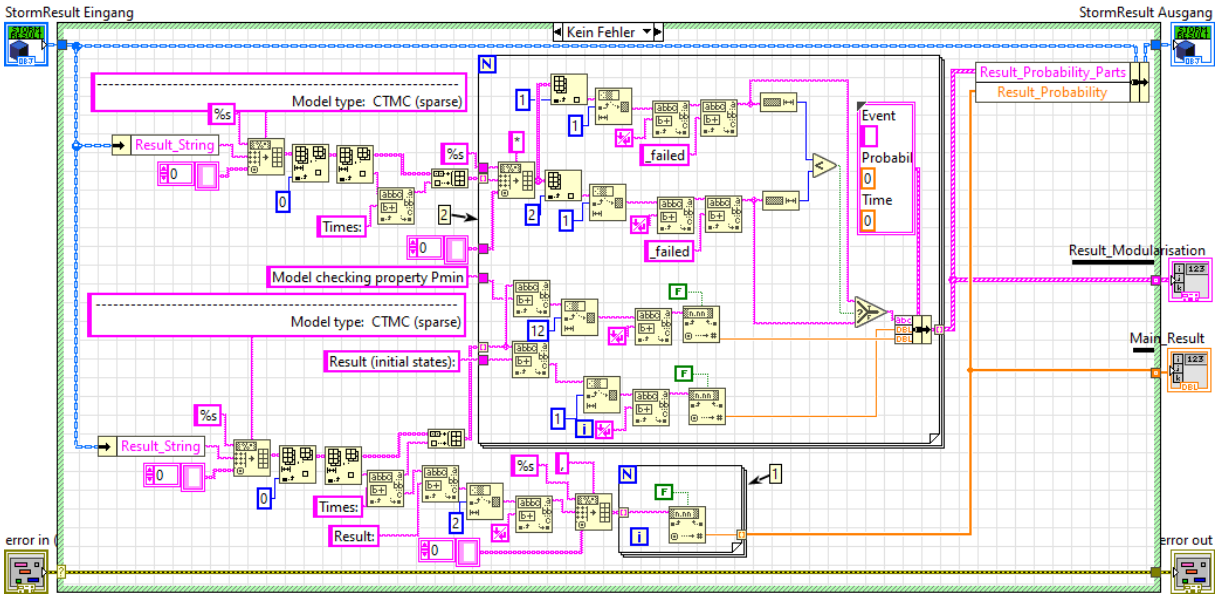


Abbildung 4.15: Ergebnisdaten-Parser Stormchecker - G-Code

Stormchecker kann nur die erste Unterebene der Ergebnisdaten bereitstellen, weshalb die Ergebnisdaten nicht modularisiert werden, sondern das Vorgehen zur Berechnung aller relevanten IntermediateEvents einzeln wiederholt wird, so wie bereits für MTTF beschrieben.

Dadurch, dass die Ergebnisdaten für das Top-Event und alle relevanten Intermediate-Events mit einer konfigurierbaren Abtastrate berechnet werden, ist es möglich, die Ergebnisverläufe in LabVIEW zu visualisieren. Dies schafft eine zusätzliche Möglichkeit der Analyse, unabhängig des Imports in die Systemmodelle, um entscheidungsbildende Prozesse unter den Stakeholdern anzustoßen.

Analyseergebnisreimport

Zur Aktualisierung und Ergebnisdatenrückkopplung in die Modellebene innerhalb Papyrus dient das letzte Modul, das aus fünf, in Abbildung 4.16 dargestellten Schritten besteht.

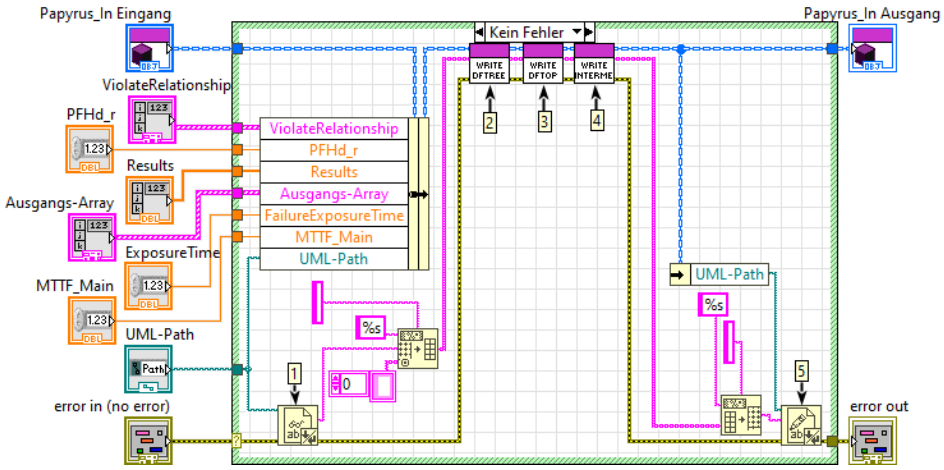


Abbildung 4.16: Ergebnisdaten-Papyrus-Import DFTs - G-Code

Im ersten Schritt wird die aktuelle *.uml-Datei zeilenweise eingelesen (1), um die abzuändernden Zeilen der Stereotypen zu suchen und mit den Ergebniswerten ersetzen zu können. Im zweiten, dritten und vierten Schritt (2, 3, 4) werden jeweils die Daten für `DFTree`, `DFTTopEvent` und `IntermediateEvent`-Elemente des entsprechenden Fehlerbaums ersetzt. Abbildung 4.17 zeigt dies demonstrativ für dynamische Fehlerbäume. Das VI sucht nach passenden Elementen über die XMI-ID und den Stereotypennamen (1). Für passende Elemente wird die Zeile neu geschrieben (2) und anhand des Ergebnisses des Top-Events des Fehlerbaums das Attribut `result` des Anwendungsprofils bestimmt (3).

Das VI für die dynamischen Top-Events beinhaltet zusätzlich eine ähnliche Funktion, abgebildet in Abbildung 4.18. Diese bewertet anhand der `Violate`-Beziehung von einem `DFTTopEvent` zu einem `FunctionalSafetyRequirement` die entsprechende Anforderung und lässt so eine farbliche Markierung in Verbindung stehender verletzter oder nicht verletzter Anforderungen zu.

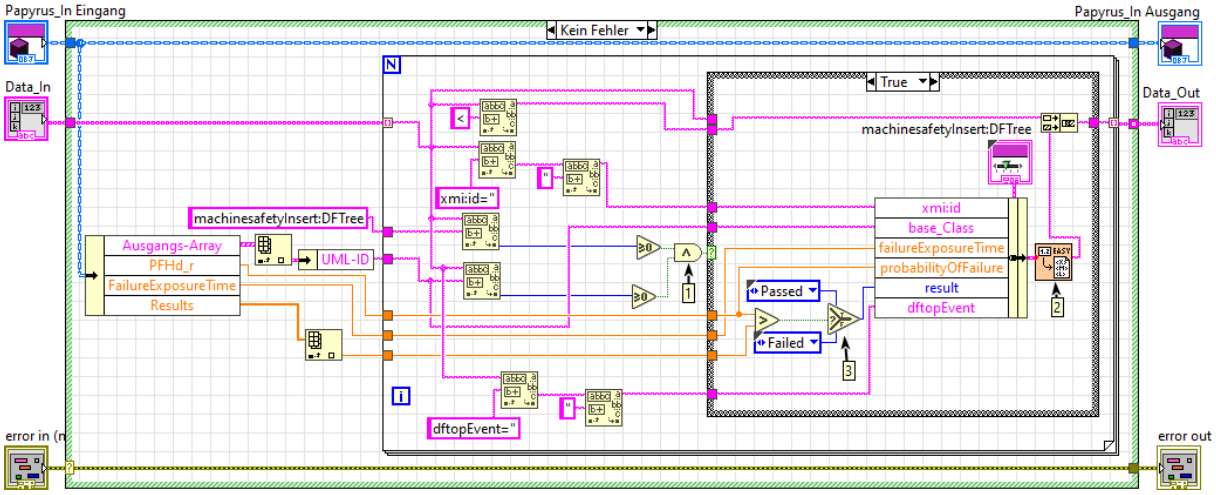


Abbildung 4.17: Ergebnisdaten-Papyrus-DFTree-Parser - G-Code

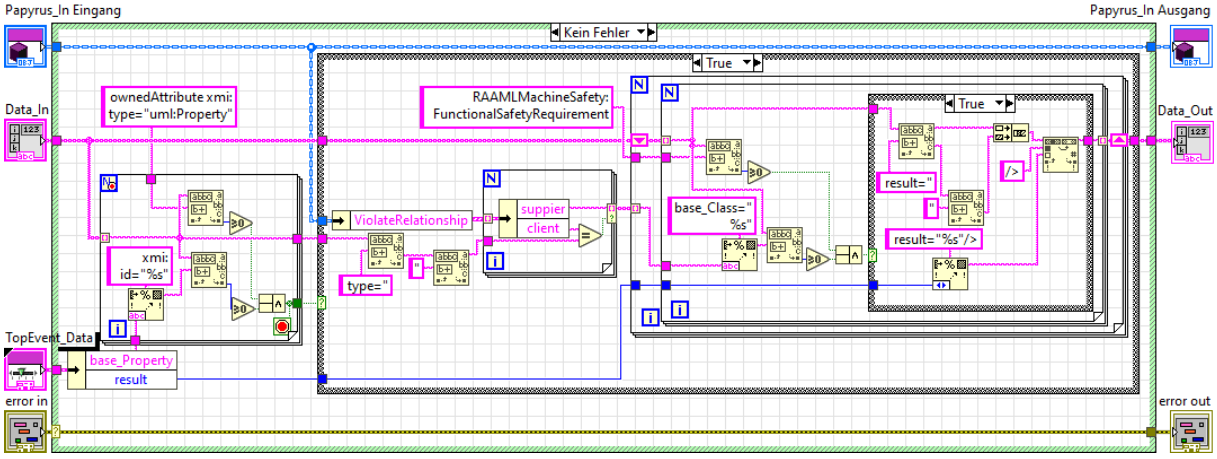


Abbildung 4.18: Ergebnisdaten-Papyrus-Requirement-Parser - G-Code

Im letzten Schritt in Abbildung 4.16 (5) wird die aktuelle *.uml-Datei mit den aktualisierten Daten überschrieben und geschlossen.

Der Vorteil eines zeilenweisen Abgleichs des gemeinsamen Modells im Vergleich zum Einlesen der gesamten Datei über EasyXML liegt klar in der Datenkonsistenz. Es hat sich herausgestellt, je mehr Daten eingelesen werden, je höher steigt auch die Wahrscheinlichkeit von Auslesefehlern aufgrund einer lückenhaften Abbildung des XML-Schemas. Indem nur die XML-Elemente verändert werden, die mit der Analyse primär in Verbindung stehen und nicht relevante Datensätze unberührt bleiben, kann die *.uml-Datei von Papyrus problemlos mit geringer Fehleranfälligkeit überschrieben werden. Nach Abschluss des Analyseergebnisimports muss das Projekt lediglich in Papyrus neu geladen werden, um die Aktualisierungen zu übernehmen.

4.2.3 SBBM

Ähnlich DFTs wurden über das in Abschnitt 3.3.3 vorgestellte Paket die Bibliothek und das Profil für SBBM in Papyrus entwickelt. Auch hier wurden zur Umsetzung der losen Kopplung die Modellierungsvorgaben aus Abschnitt 3.4 genutzt, durch die die Modelle über LabVIEW-

basierte XMI-Transformatoren in ein externes Analysewerkzeug exportiert, dort die Analysen durchgeführt und die Ergebnisse wieder in die Modellierungsumgebung importiert werden. Hierzu wird auch hier zunächst ein Werkzeug ausgewählt, das für die Analyse der SBBM geeignet ist, und nachfolgend das Architekturkonzept angewendet und Module implementiert.

Werkzeugauswahl

Wie aus der Recherche unter Abschnitt 2.2.3 hervorgeht, wurden lediglich zwei Werkzeuge identifiziert. Allein durch den Umstand, dass es sich bei PAScal Safety Calculator [247] von PILZ um ein kostenpflichtiges Werkzeug handelt, scheidet dieses Werkzeug für diese Arbeit aus. Im Gegensatz hierzu ist SISTEMA kostenfrei und arbeitet mit einem Open-Source XML-Format, in dem die Projekte abgespeichert werden. Dies bietet die Möglichkeit, von außen SISTEMA-Projekte zu erstellen und Ergebnisse auszutauschen. In der Praxis steht SISTEMA heute meist isoliert und findet keine Einbettung in MBSE-Frameworks.

Implementierung

Wie in Abbildung 4.19 dargestellt, teilt sich die Implementierung, ausgehend vom Architekturkonzept in Abschnitt 3.4.2 Abbildung 3.26, genauso wie die Toolkopplung von DFTs in Abbildung 4.6 in die fünf Module XMI-Export und -Transformation in LabVIEW, Transformation in Analysewerkzeug-Inputformat, Analysewerkzeug-Ausführung, Analyseergebnisverarbeitung und XMI-Transformation zu Systemmodell. Aufgrund der gleichen Vorgehensweise wie in Abschnitt 4.2.2 werden lediglich die Unterschiede aufgezeigt. Die komplette Modulbeschreibung findet sich im Anhang A.10.1.

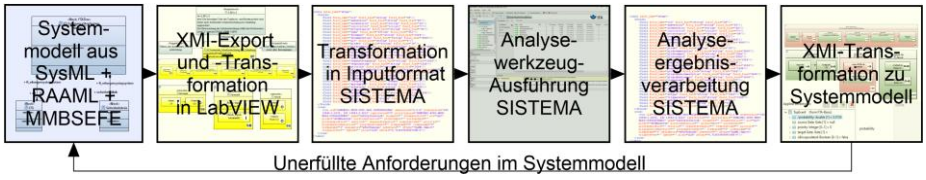


Abbildung 4.19: Implementierung SBBM

Zur Umsetzung des XMI-Exports von System- und Analysemodellen im *.uml-Format aus Papyrus wurde das gleiche Vorgehen, wie bereits für DFTs in Abbildung 4.7 veranschaulicht, angewendet und ein Zwischenformat zur Weiterverarbeitung aus allen relevanten Elementen erzeugt (vgl. Anhang A.10.1 Abschnitt Analysemodellexport und Transformation).

Auf Basis des Zwischenformats wurde das SISTEMA-Werkzeug angebunden. Das XML-Format der Projekt-Dateien besteht aus Tabellen für die verschiedenen Projektbestandteile. Hierzu gehört das Projekt selbst, die Sicherheitsfunktionen, Subsysteme, Channel, Blöcke und Elemente. SISTEMA nutzt dabei ein unter dem Programmverzeichnis angegebenes XML-Schema für die Elemente zur Überprüfung des XML-Formats [293].

Das Hauptelement des Dokuments ist `<xmldocdata>`. Innerhalb dieses Elements gibt es zu jedem Projektbestandteil, wie im Folgenden für Sicherheitsfunktionen in Listing 4.18 gezeigt, ein `<table>`-Element. Innerhalb des `<table>`-Elements findet sich das `<fields>`-Element mit mehreren `<field>`-Elementen, die die einzelnen Felder eines Projektbestandteils repräsentieren. Jedes `<field>`-Element hat verschiedene Attribute, wie `field_name`, `field_kind` und `field_Size`. Abhängig dieser Attribute findet sich das `<rows>`-Element mit mehreren `<row>`-Elementen, die die einzelnen Datensätze repräsentieren.

```
<table table_name="sfops">
   <fields>
      <field field_name="oid" field_kind="string" field_Size="38"/>
      <field field_name="ssmversion" field_kind="string" field_Size="40"/>
      <field field_name="normversion" field_kind="string" field_Size="40"/>
      <field field_name="isprotected" field_kind="integer" field_Size="0"/>
      <field field_name="projectopoid" field_kind="string" field_Size="38"/>
      <field field_name="name" field_kind="string" field_Size="512"/>
      <field field_name="document" field_kind="string" field_Size="1024"/>
      <field field_name="documentation" field_kind="string" field_Size="4000"/>
      <field field_name="plr" field_kind="string" field_Size="4"/>
      <field field_name="plrdet" field_kind="string" field_Size="18"/>
      <field field_name="plrdocumentation" field_kind="string" field_Size="2000"/>
      <field field_name="plrdocument" field_kind="string" field_Size="1024"/>
      <field field_name="plrstandard" field_kind="string" field_Size="2000"/>
      <field field_name="plrstandardfile" field_kind="string" field_Size="1024"/>
      <field field_name="reaction" field_kind="string" field_Size="1024"/>
      <field field_name="requestfrequency" field_kind="string" field_Size="1024"/>
      <field field_name="responsetime" field_kind="string" field_Size="1024"/>
      <field field_name="priority" field_kind="string" field_Size="1024"/>
      <field field_name="plrgraphdocumentation" field_kind="string" field_Size="2000"/>
      <field field_name="plrgraphdocument" field_kind="string" field_Size="1024"/>
      <field field_name="riskparamf" field_kind="integer" field_Size="0"/>
      <field field_name="riskparamp" field_kind="integer" field_Size="0"/>
      <field field_name="riskparams" field_kind="integer" field_Size="0"/>
      <field field_name="wparameter" field_kind="string" field_Size="11"/>
      <field field_name="safestate" field_kind="string" field_Size="1024"/>
      <field field_name="sftype" field_kind="string" field_Size="1024"/>
      <field field_name="triggerevent" field_kind="string" field_Size="1024"/>
      <field field_name="opmode" field_kind="string" field_Size="50"/>
      <field field_name="plcal" field_kind="string" field_Size="4"/>
      <field field_name="pfhcal" field_kind="string" field_Size="32"/>
   </fields>
   <rows>
      <row oid="06E4E841-5E7B-565C-9401-81DFA2B21BBA" ssmversion="2.0.8" normversion="ISO
      13849-1:2015, ISO 13849-2:2012" isprotected="0" projectopoid="39BECB8C-4098-4BDF-
      B0A9-CC89A993DE00" name="SafetyInterlock" document="" documentation="" plr="plC"
      plrdet="detMeasures" plrdocumentation="" plrdocument="" plrstandard=""
      plrstandardfile="" reaction="" requestfrequency="" responsetime="" priority=""
      plrgraphdocumentation="" plrgraphdocument="" riskparamf="0" riskparamp="0"
      riskparams="1" wparameter="wParUnknown" safestate="" sftype="SysML-Import"
      triggerevent="" opmode="" plcal="plB" pfhcal="4,71205350664729E-6"/>
   </rows>
</table>
```

Listing 4.18: XML-Format SISTEMA für eine Sicherheitsfunktion

Aufgrund dieses Formats musste in LabVIEW lediglich das XML-Schema für die Felder-Definitionen erstellt werden, um automatisch Cluster für die Übergabe der Datensätze zu generieren, was die Fehleranfälligkeit reduzierte.

Das Modul zur Transformation ins Analyseinputformat besteht daher aus zwei Teilen. Der erste Teil bildet sich aus sieben, den Projektbestandteilen zugeordneten, VIs, abgebildet in Abbildung 4.20, die im zweiten Teil der Erstellung der *.ssm-Datei für SISTEMA dienen.

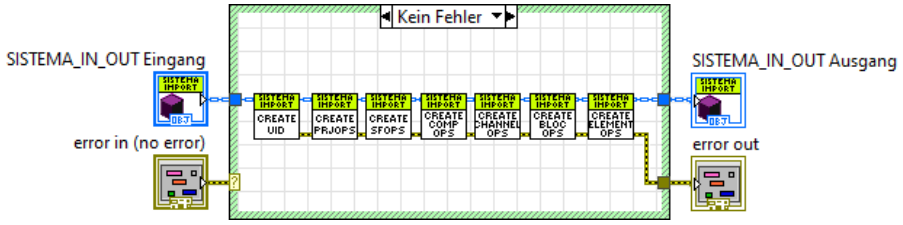


Abbildung 4.20: SISTEMA-Input-Transformation - G-Code

In Abbildung 4.20 werden zuerst die IDs erstellt, die SISTEMA benötigt, um Abhängigkeiten zwischen den Bestandteilen festzulegen. Das zweite VI legt dann das Projekt innerhalb des Formats an. Auf dieser Basis können in VI drei bis sieben Sicherheitsfunktionen, Subsysteme, Channel, Blöcke und Elemente nacheinander in das SISTEMA-Format abgebildet werden.

Es wurde keine offizielle Dokumentation des Formats im von SISTEMA bereitgestellten XML-Schema und weiteren Quellen gefunden. Daher musste die genaue Bedeutung jedes Attributs interpretiert und getestet werden. Eine vollständige Auflistung der Attribute für Sicherheitsfunktionen, wie in Listing 4.18 gefordert, findet sich im Code in Abbildung 4.21.

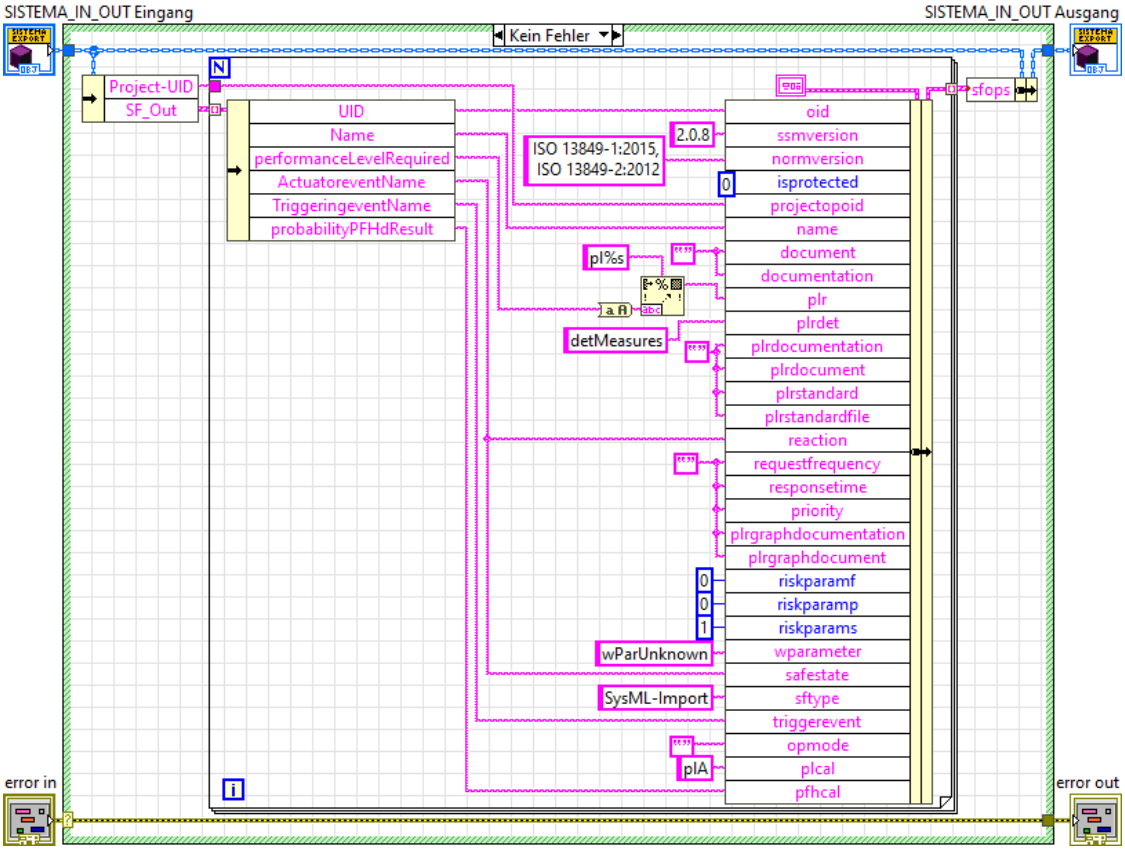


Abbildung 4.21: SISTEMA-Input-Transformation Subsystem - G-Code

Abhängig von verschiedenen Bedingungen, wie bei Subsystemen, Channel, Blöcken und Elementen notwendig, werden die Attribute gesetzt. Einige Attribute sind zwar nicht primär für die Analyse erforderlich oder dienen als mögliche zukünftige Erweiterungen, helfen aber bei

der vollständigen Abbildung des Formats und erleichtern die zukünftige Erweiterbarkeit des Transformators.

Durch das XML-Format von SISTEMA entsteht so neben dem Format zur Analyse auch ein leicht erweiterbares und interpretierbares Importformat mit geringer Fehleranfälligkeit und einfacher Wartbarkeit.

Nachdem die Teildatensätze im SISTEMA-Input-Format erstellt wurden, werden diese zusammengesetzt und die *.ssm-Datei für SISTEMA erzeugt (vgl. Anhang A.10.1 Abschnitt Analyseinputformat).

Die Analyseausführung kann ähnlich der Ausführung von DFTs in Stormchecker verstanden werden, wobei SISTEMA direkt durch einen Konsolenaufruf mittels des Systembefehlausführen-VI gestartet und über virtuelle Tastenkombinationen gesteuert wird (vgl. Anhang A.10.1 Abschnitt Analyseausführung).

Nach der Ausführung folgt die Ergebnisverarbeitung, die umgekehrt der Analyseinputformat-Transformation das SISTEMA-XML-Format nutzt, um die Daten ins Zwischenformat zu transformieren und basierend darauf verschiedene Überprüfungen anzustellen (vgl. Anhang A.10.1 Abschnitt Analyseergebnisverarbeitung).

Das Modul zur Aktualisierung und Ergebnisdatenrückkopplung der SBBM in Papyrus gestaltet sich strukturell wieder sehr ähnlich dem Ansatz im DFT-Transformator (vgl. Anhang A.10.1 Abschnitt Analyseergebnisreimport).

4.2.4 Risikograph

Wie bereits für DFTs und SBBM wurde auch die Implementierung zur Berechnung von Risikographen basierend auf der losen Kopplung, wie in Abschnitt 3.4 beschrieben, umgesetzt. Hierfür wurde ebenso das in Abschnitt 3.2.11 vorgestellte Paket mit Bibliothek und Profil in Papyrus entwickelt. Im Gegensatz zu DFTs und SBBM erfolgt die Analyse durch einfache Berechnungen mittels Addition und Multiplikation der Risikoitems direkt in LabVIEW. Dadurch entfällt die Auswahl eines Werkzeugs, wodurch das Architekturkonzept eine vereinfachte Anwendung findet. Ein weiterer Unterschied besteht darin, dass kein Anwendungsprofil verwendet wird, wodurch die LabVIEW-basierte XMI-Transformation anders ausfällt.

Wie in Abbildung 4.22 dargestellt, teilt sich die Implementierung, ausgehend vom Architekturkonzept in Abschnitt 3.4.2 Abbildung 3.26, diesmal nur in die drei Module XMI-Export und -Transformation in LabVIEW, Analyse-Ausführung in LabVIEW und XMI-Transformation zu Systemmodell. Ähnlich wie für SBBM in Abschnitt 4.2.3 werden lediglich die Unterschiede

aufgezeigt. Die kompletten Modulbeschreibungen für den XMI-Export und Transformation in LabVIEW sowie den XMI-Reimport zum Systemmodell finden sich im Anhang A.10.2.

Abbildung 4.22: Implementierung Risikograph

Für den XMI-Export von System- und Analysemodellen im *.uml-Format aus Papyrus wurde das gleiche Vorgehen wie für DFTs in Abbildung 4.7 angewendet. Da kein Anwendungsprofil verwendet wird, enthalten die exportierten Stereotypen lediglich die XMI-ID des UML-Elements in Papyrus, welches die entsprechenden Attribute der Risikograph-Elemente enthält. Dies hat den Vorteil, dass die Attributwerte nur an einer Stelle verfügbar sind und die Vollständigkeit der Attributzuordnungen dort direkt durch OCL-Constraints überprüft wird. Im Vergleich zur Umsetzung für DFTs und SBBM mit einem Anwendungsprofil können dort Fehler auftreten, wenn die Attributwerte in den Stereotypen des Anwendungsprofils nicht mit denen in den UML-Elementen in Papyrus synchron sind. Der Transformator zur Umsetzung der Risikobeurteilung durch Risikographen repräsentiert daher ein robusteres Vorgehen. Allerdings ist die Verarbeitung der Daten aus Papyrus für den Transformator komplexer, da zunächst die XMI-IDs den UML-Elementen über den Umweg eines zusätzlichen Zwischenformats zugeordnet werden müssen, bevor die Risikoitems zusammengesetzt werden können (vgl. Anhang A.10.2 Abschnitt Analysemodellexport und Transformation).

Die Analyseausführung besteht lediglich aus folgender in Abbildung 4.23 dargestellten Hauptfunktion mit dem SubVI in Abbildung 4.24 und dessen SubVI in Abbildung 4.25. Hierbei werden in einer Schleife alle zuvor identifizierten Risikoitems durchlaufen und die Risikoprioritätszahl (RPZ) sowie das PL_r berechnet.

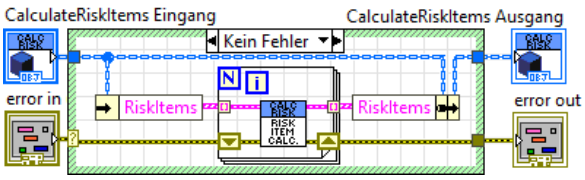


Abbildung 4.23: Analyseausführung aller Risikoitems - G-Code

Das VI in Abbildung 4.24 bestimmt das PL_r für das ungeminderte Risiko, welches die Rahmenbedingungen für den Nachweis der Risikominderung innerhalb von DFTs und SBBM liefert, die RPZ der aktuellen Risikominderung und die RPZ vorheriger Risikominderungen.

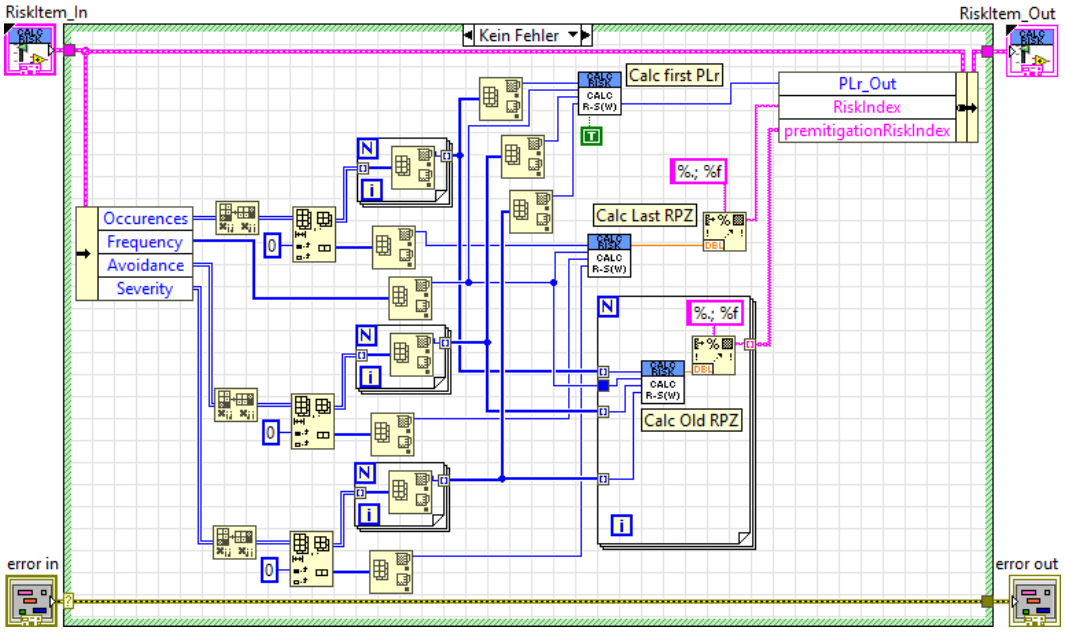


Abbildung 4.24: Analyseausführung eines Risikoitems für PL_r und RPZ - G-Code

Aus den Berechnungsfaktoren Schwere der Verletzung (S), Häufigkeit und Dauer (F), Möglichkeit zur Vermeidung (P) und Wahrscheinlichkeit (P) aus Abschnitt 2.2.1 Abbildung 2.10 ergibt sich die in Tabelle 4.1 dargestellte Wertetabelle. Zur Demonstration werden die Berechnungen einfach gehalten, weshalb die gleiche Wertetabelle für PL und RPZ gelten soll. Nach den Vorgaben aus ISO 12100 in Verbindung mit den Vorgaben aus ISO 13849 gilt lediglich eine Funktion aller Parameter. Zur Veranschaulichung wird daher die Funktion

$$RPZ = S * (F + P + O) \text{ gewählt.}$$

Formel 4.1: Berechnung der Risikoprioritätszahl

Zur Schaffung einer einfachen Berechnungsgrundlage wird der RPZ daher an PL angepasst. Hierfür werden die folgenden Werte für die Parameter aus Abbildung 2.10 festgelegt:

$$S1 = 1; S2 = 1{,}5; F1 = 1;\ F2 = 2; P1 = 1;\ P2 = 2; O1 = 1\ O2 = 2$$

Formel 4.2: Parameterwerte zur Berechnung der Risikoprioritätszahl

Somit ergeben sich die in Tabelle 4.1 dargestellten RPZ und Einschätzungen des Risikos.

S	1	1	1	1	1	1	1	1	1,5	1,5	1,5	1,5	1,5	1,5	1,5	1,5
F	1	1	1	1	2	2	2	2	1	1	1	1	2	2	2	2
P	1	1	2	2	1	1	2	2	1	1	2	2	1	1	2	2
O	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2
PL	a	a	a	b	a	b	b	c	b	c	c	d	c	d	d	e
RPZ	3	4	4	5	4	5	5	6	4,5	6	6	7,5	6	7,5	7,5	9

Tabelle 4.1: Wertetabelle PL und Risikoprioritätszahl

Daraus ergibt sich die Abschätzung der RPZ:

- RPZ ≤ 4 ein vernachlässigbares Risiko;
- 4 < RPZ ≤ 5 ein geringes Risiko, das toleriert werden kann, sollten keine adäquaten risikomindernden Maßnahmen gefunden werden;
- 5 < RPZ ≤ 6 ein zu hohes Risiko, das nicht toleriert werden darf;
- 6 < RPZ ≤ 7,5 ein viel zu hohes Risiko, das gemindert werden muss und;
- 7,5 < RPZ ≤ 9 ein maximales Risiko, das gemindert werden muss.

In Abbildung 4.25 findet sich die Implementierung von Formel 4.1 und Formel 4.2, sowie die Korrelation zum PL_r. Hierbei wird für ein besseres Verständnis ein Formelknoten (1) und Mathscript-Knoten (2) eingesetzt. Somit lässt sich die Berechnung für alle RPZ und dem PL_r einfach und vergleichbar mit anderen Sprachen darstellen.

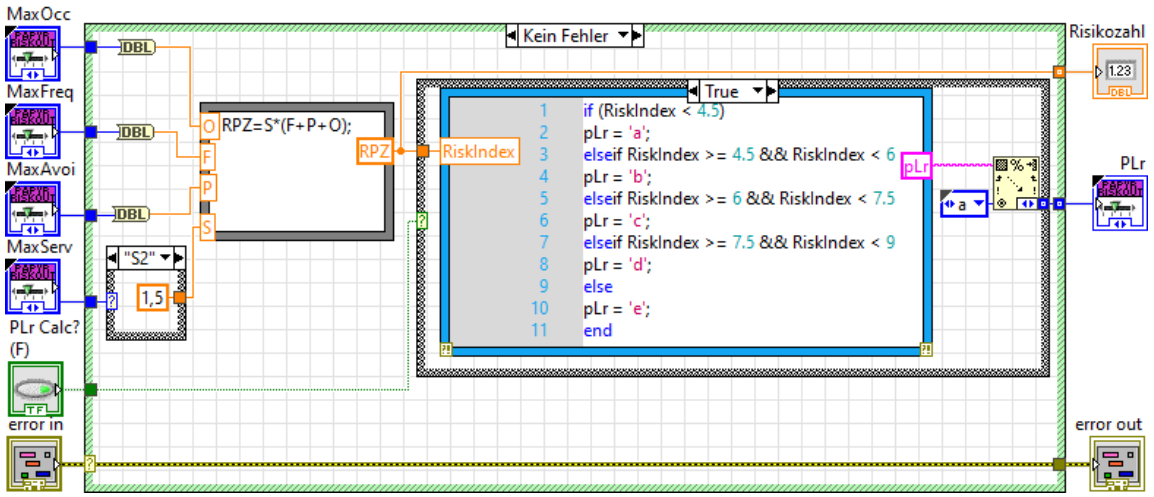


Abbildung 4.25: Analyseausführung Risikoberechnung - G-Code

Beim Reimport der Ergebnisse wird ein ähnlicher Prozess wie bei DFTs und SBBM angewendet, indem die aktuelle *.uml-Datei gelesen und die entsprechenden Stellen ausgetauscht werden. Aufgrund der Implementierung ohne Anwendungsprofil muss jedoch das zusätzliche Zwischenformat verwendet werden, um zunächst die Werte der Risikoitem-Attribute dort zu ersetzen und dann basierend hinterlegten Positionsdaten die Risikoitems in der *.uml-Datei zu finden und auszutauschen.

4.3 Zusammenfassung

In diesem Kapitel wurde die prototypische Implementierung der Konzepte aus Kapitel 3 beschrieben, um zu veranschaulichen, wie die Modellierung auf einer Modellierungsplattform umgesetzt werden kann. Zusätzlich zur Hauptimplementierung der MMBSEFE-Metamodelle in Papyrus wurde die Konformität über OCL überprüft, die Darstellbarkeit über CSS verbessert

und Transformatoren in LabVIEW implementiert, um die Integrationsfähigkeit der erstellten Lösung in eine Standard-MBSE-Plattform zu zeigen.

In Abschnitt 4.1 wurde die Umsetzung von MMBSEFE beschrieben. Auf Basis der Modellierungsplattform Papyrus wurden die unter Abschnitt 3.2 und 3.3 entwickelten Profile und Bibliotheken unter dem in Abschnitt 3.4 entwickelten Architekturkonzept für KMUs weiter ausgearbeitet. Tabelle 4.2 gibt eine kurze Zusammenfassung der insgesamt 265 umgesetzten Elemente, zugeordnet zu den Profilen und Bibliotheken. Die Einarbeitung in die MMBSEFE-Struktur wurde hierbei über OCL und CSS so vereinfacht, dass die aufzubringenden Ressourcen von KMUs auf ein Minimum beschränkt und Fehler bei der Anwendung verringert werden können. Hierbei wurden insgesamt 87 OCL-Constraints, 71 Selektoren für 27 verschiedene Elemente mit 329 Erscheinungsbild-Eigenschaften für FTA und DFTs sowie 36 Selektoren für 18 verschiedene Elemente mit 206 Erscheinungsbild-Eigenschaften für SBBM eingesetzt. Dies bringt den zusätzlichen Vorteil eines nachvollziehbaren Dateninputs für die Analysen, in denen speziell für die explizite Anbindung der Transformatoren eine Einhaltung und Verständlichkeit der Modellierungsvorgaben dringend notwendig ist.

Pakete	**Anzahl Elemente in**	
	Profilen	**Bibliotheken**
Construction	4	4
MachineAssembly	4	29
TestmachineConstruction	2	80
MechanicalView	3	4
ElectricalView	2	5
HydraulicView	2	4
PneumaticView	2	4
InformaticView	2	2
ISO 12100 & ISO 13849	22	26
DFT	12	17
SBBM	14	21
Gesamt	69	196

Tabelle 4.2: Implementierte Profil- und Bibliothekselemente

Basierend auf den in Abschnitt 4.1 vorgestellten Umsetzungen wurde in Abschnitt 4.2 das Vorgehen und die prototypische Anbindung der Werkzeuge für DFTs, SBBM und Risikographen beschrieben. Hierbei kann festgestellt werden, dass im Gegensatz zur engen Toolkopplung die lose Toolkopplung für DFTs, SBBM und Risikographmethode einen nicht nur für KMUs attraktiven Ansatz bietet. Indem nur ein Transformator, in diesem Falle in LabVIEW, zu bestehenden validierten und betriebsbewährten Werkzeugen aufzubauen ist, im Gegensatz zur Ab-

bildung eines kompletten Analysewerkzeugs in der Modellierungsumgebung, steigt das Vertrauen in den Analyseprozess. Um die Fehlerwahrscheinlichkeit bei der Transformation der Modelle zu minimieren, wurden Eingabe- und Ergebnisdaten innerhalb von Plausibilitätsprüfungen verglichen sowie die Ergebnisdaten in die gemeinsamen Modelle übertragen und für DFTs und SBBM visualisiert. Die korrekte Transformation aller Transformatoren wurde anhand von Beispielen überprüft. Um die vollständige Korrektheit der Transformatoren nachzuweisen und diese kommerziell zu nutzen, werden zusätzliche Methoden, beispielsweise Code-Qualitätsprüfung durch Strukturprüfungen oder Black-Box-Tests [11], notwendig, die eine Validierung und Zertifizierung in Verbindung der Vorgaben aus der Grundnorm ISO 61508 ermöglichen. Für die Implementierung von Transformatoren in dieser Arbeit wurde die prototypische Umsetzung als ausreichend angesehen.

Um die Transformatoren zu den verschiedenen Werkzeugen zu vergleichen, wurde der Codeumfang und der Zeitaufwand für die verschiedenen Module dokumentiert und analysiert. Der Codeumfang von LabVIEW-Programmen lässt sich nicht direkt in Source Lines of Code (SLOC) beschreiben. Stattdessen wird eine LabVIEW-Metrik zur Analyse der Codekomplexität über die Knotenanzahl verwendet, der nach [294] ein Eins-zu-eins-Verhältnis zwischen LabVIEW-Knoten und SLOC zugeordnet werden kann. Die in Tabelle 4.3 dargestellte Statistik zur Codekomplexität fasst die Ergebnisse zusammen.

Modul\Werkzeug	Stormchecker	SISTEMA	Risikograph
Modellexport	1371	1199	1210
Inputformat	141	716	-
Ausführung	566	98	75
Ergebnisverarbeitung	160	471	-
Ergebnisimport	2581	2641	2.867
Gesamt	4819	5125	4152

Tabelle 4.3: Codeumfang in LabVIEW-Knoten

Durch das minimale Eingabe- und effektive Ausgabeformat von Stormchecker konnte der erste Prototyp des Stormchecker-Transformators mit nur etwa 4819 Codezeilen in weniger als drei Wochen implementiert werden. Die Hauptkomplexität entsteht hierbei allerdings durch das eingesetzte EasyXML-Toolkit, das dennoch sehr performant arbeitet. Durch das mehrfache Ausführen und Verarbeiten des Stormchecker-Aufrufs ergibt sich allerdings auch hier eine höhere Codekomplexität. Der SISTEMA-Transformator für die SBBM wurde als zweites implementiert, was sich auch in der Entwicklungszeit des ersten Prototypens von zwei Wochen niederschlägt. Durch das XML-Importformat von SISTEMA gestaltete sich der exakte Aufbau des Schemas wieder etwas komplexer, was sich allerdings durch die gemachten Erfahrungen und

wiederverwendbaren Code zeitlich kompensieren ließ. Zudem ist die Komplexität der Ausführung mit 98 Knoten sehr performant. Durch das nicht genutzte Anwendungsprofil für Risikographen zeigt sich, dass der Modellexport und Ergebnisimport für Risikographen, trotz der wenigen notwendigen Elemente komplexer ausfällt. Der erste Prototyp des Risikograph-Transformators zur Ausführung in LabVIEW wurde dennoch innerhalb von wenigen Tagen entwickelt. Die kurzen Entwicklungszeiten demonstrieren damit die Rapid-Prototyping-Möglichkeiten mit LabVIEW und eine einfache und schnelle Anwendbarkeit des Konzeptes für LabVIEW-versierte KMUs in der Maschinenindustrie.

Diese zeigt auf, dass es möglich ist, die Modellierung von MMBSEFE entsprechend bestehender sicherheitskritischer Standards und Technologien zu erstellen und Analysewerkzeuge anzubinden. Tabelle 4.4 fasst den Beitrag dieser Implementierungen zu den Forschungsfragen zusammengefasst. Während die prototypische Proof-Of-Concepts-Implementierung gemäß der logischen Reihenfolge der Bausteine strukturiert wurde, werden in dieser Zusammenfassung diese Bausteine mit den ursprünglichen Problemfeldern in der letzten Spalte der Tabelle 4.4 abgeglichen. Die prototypische Umsetzung von MMBSEFE und der entsprechenden Transformatoren zeigt dabei, dass die konzeptuelle Modellierung aus Kapitel 3 prototypisch implementiert und die allgemeinen Anforderungen erfüllt werden konnten. Die verbleibenden Forschungsziele des Typs Experiment sind in Tabelle 4.4 aufgeführt. Bereits gelöste Forschungsziele sind grau hinterlegt.

Forschungsaufgaben (FA)		
FA	**Beschreibung**	**Abschnitt**
Maschinenbauspezifische Modellierung		
FA1-B	Recherche einer gemeinsamen Modellebene und Architektur für MBSE im Maschinenbau, im Speziellen für Prüfmaschinen, und Grundlage zur Modularisierung	2.1
FA1-TB	Modellierung der Anwendungsfälle und Metamodelle für alle Stakeholder in einer maschinenbauspezifischen Model-Based Systems Engineering Framework-Erweiterung	3.2
FA1-I	Umsetzung einer modularisierten gemeinsamen Modellebene und Architektur für MBSE im Maschinenbau, im Speziellen für Prüfmaschinen	4.1.1 4.2.1
FA1-E	Evaluation einer modularisierten gemeinsamen Modellebene und Architektur für MBSE im Maschinenbau, im Speziellen für Prüfmaschinen	
Integration der Analysen des Maschinenbaus		
FA2-B	Identifikation maschinenbauspezifische Analysemethoden und passender Frameworks	2.2
FA2-TB	Modellierung der Metamodelle für die Analysemethoden dynamische Fehlerbaumanalyse (DFT) und sicherheitsbezogene Blockdiagrammmethode (SBBM) zum Nachweise der Risikominderung	3.3
FA2-I	Entwurf und Implementierung zur Integration maschinenbauspezifische Analysemethoden	4.2.2– 4.2.4
FA2-E	Evaluation typischer Analysemethoden	

Forschungsaufgaben (FA)		
FA	**Beschreibung**	**Abschnitt**
MBSE-Prozesse für KMUs		
FA3-B	Identifikation der Hindernisse zur Anwendbarkeit von MBSE in KMUs	2.3
FA3-TB	Modellierung eines Architekturkonzeptes zur Anwendbarkeit von MBSE in KMUs unter Berücksichtigung der Aspekte Agilität, Vertrauen, Zusammenarbeit und Effizienz	3.4
FA3-I	Implementierung eines Konzeptes für KMUs in der Maschinenindustrie	4.1.2
FA3-E	Evaluation mit KMU	

Tabelle 4.4: Forschungsaufgaben nach Implementierung

Die prototypische Umsetzung von MMBSEFE in Papyrus und der in diesem Abschnitt beschriebenen Transformatoren findet sich auf Github [295] und ist zum Download unter [296] verfügbar.

Zuletzt muss die Methodik und Implementierung gemäß den ursprünglichen Forschungsfragen bewertet werden. Diese Aufgabe wird im folgenden Evaluationskapitel behandelt. Da alle bisher angesprochenen Bereiche Teile eines gemeinsamen Entwicklungsprozesses darstellen, soll zur Beantwortung der Forschungsaufgaben des Typs Evaluation ein komplettes Projekt für eine Prüfmaschine umgesetzt werden.

5 Evaluation

In diesem Kapitel werden die Forschungsziele aus Kapitel 2 (Beobachtung), Kapitel 3 (Theoriebildung) und Kapitel 4 (Implementierung) gemäß dem Ansatz von Nunamaker (vgl. Abschnitt 1.4) anhand des ursprünglichen Satzes von Forschungsfragen (vgl. Abschnitt 1.2.6) im Rahmen der in Abschnitt 2.3.2 eingeführten empirischen Evaluationsmethoden bewertet. Hierfür wurde eine experimentelle Evaluation durchgeführt, bei der im Rahmen der Case Study Research Methodik eine beispielhafte Fallstudie zum Aufbau einer Prüfmaschine mit dem Schwerpunkt auf Sicherheitsbetrachtungen erstellt wurde. Um die empirische Bewertung direkt auf die Forschungsfragen auszurichten, ist dieses Kapitel ebenso in drei Teile strukturiert, von denen jeder einen der beschriebenen Forschungsbereiche behandelt.

In Abschnitt 5.1 wird der Aufbau der Prüfmaschine zunächst für UC1.1 nach Projektierungsaspekten verbal beschrieben und für UC1.2, spezifischer UC5.4, in SysML-Requirements als Startpunkt der Evaluation des modellbasierten Ansatzes umgesetzt. Dies legt die Grundlage für UC1.3 zur Umsetzung der Stakeholder-spezifischen Modelle, um die Grundlage der Risikobeurteilung nach der Methodik aus ISO 12100 für UC5.1 zu legen. Abschnitt 5.2 bietet den Aufbau und die Analyse der aus der Risikobeurteilung und Risikominderung nach ISO 13849 hervorgehenden Sicherheitsfunktionen durch DFTs und die SBBM zur Erfüllung von UC5.2 sowie eine Berechnung mit Vergleich der Ergebnisse ausgewählter Sicherheitsfunktionen. Zusätzlich zu dieser quantitativen Evaluation soll unter Abschnitt 5.3 schließlich die Forschungsfrage zur Anwendbarkeit des MBSE-Prozesses für KMUs am Beispiel einer Fallstudie im KMU ProNES geklärt werden. Zur qualitativen Evaluation werden hier anhand der Evaluationsmethode und des Prozesses aus Abschnitt 2.3.2 Interviews durchgeführt und ausgewertet, die die Eignung des modellbasierten Ansatzes zur Integration des Stakeholders Sicherheitsingenieur in KMUs darstellen soll.

Am Anfang jedes Unterabschnitts innerhalb der Abschnitte 5.1 und 5.2 werden die zu behandelnden Teile anhand des Ablaufs der Risikobeurteilung nach ISO 12100 aus Abschnitt 2.1.3 und des darauf aufbauenden Vorgehens zur Risikobeurteilung und -minderung aus Abschnitt 4.2.1 auf Basis von MMBSEFE definiert. Dies dient dazu, Aspekte der vorgeschlagenen Lösung im Hinblick auf die entsprechende Forschungsfrage auf Basis von Fallunterscheidungen zu belegen oder zu widerlegen. In jedem dieser Unterabschnitte werden die Durchführung und

Model-Based Systems Engineering sicherheitskritischer Anwendungen im Maschinenbau mittelständischer Unternehmen © 2024 by Nick Berezowski is licensed under CC BY 4.0

die Ergebnisse anhand des zum Zweck der Evaluation modellierten Projektes beschrieben, beobachtete positive oder negative Effekte der Modellierung aufgeführt und offene Punkte diskutiert. Das modellierte Projekt, als Gegenstand der Evaluation, legt hierbei das gemeinsame Modell aller Stakeholder dar. Zur qualitativen Evaluation über Interviews in Abschnitt 5.3 werden in dessen Unterabschnitten die Prozessschritte aus Abschnitt 2.3.2 schrittweise durchgeführt. Dies beinhaltet alle Prozessschritte, wobei die ausführlichen Antworten der Stakeholder im Anhang A.12 den einzelnen Stakeholdern und entsprechenden Fragestellungen zugeordnet sind. Abschließend bietet jeder Abschnitt eine Diskussion und Zusammenfassung der Ergebnisse. Mit diesem Ansatz werden alle in Abschnitt 1.2 definierten Problemfelder behandelt und die bisher beschriebene Lösung gegen die Forschungsfragen bewertet.

5.1 Maschinenbauspezifisches Modellierungsbeispiel

In diesem Abschnitt wird eine detaillierte Bewertung der Modellierungslösung in Bezug auf die maschinenbauspezifische Integration mit Fokus auf die Risikobeurteilung durchgeführt und somit FA1-E dargestellt. Auf Basis des in Abbildung 4.4 definierten Ablaufs und der umfangreichen Modellierung teilt sich der Abschnitt in vier Unterabschnitte, die die Maschine modellieren und die verschiedenen Teile der Risikobeurteilung behandeln. Im Unterabschnitt 5.1.1 wird die Modellierung zur Bestimmung der Grenzen der Maschine beschrieben, indem Anforderungen an die Prüfmaschine und Prüfmaschinenteile modelliert, der Anwendungskontext bestimmt und Anwendungsfälle erstellt werden. Hierdurch soll festgestellt werden, ob die in Anhang A.4 beschriebenen Aspekte adäquat abgebildet werden können. Der zweite Teil in Abschnitt 5.1.3 befasst sich mit der Identifikation von Gefährdungen und ermittelt Gefährdungssituationen, die die Grundlage zur in Abschnitt 5.1.2 beschriebenen Risikoabschätzung liefern sollen. Schließlich wird in Abschnitt 5.1.4 überprüft, inwieweit die gesammelten Informationen eine ausreichende Grundlage für die Risikobewertung und Identifikation von Sicherheitsfunktionen liefern. Hierbei soll beurteilt werden, ob die definierten Anwendungsfälle aus Abschnitt 3.2.2 zur Risikobeurteilung durch den Sicherheitsingenieur verfügbar sind, um die geplanten Ergebnisse zu erzielen. Abschnitt 5.1.5 fasst die Eignung des MMBSEFE-Ansatzes in Hinsicht auf Modellierbarkeit kurz zusammen.

Für die Fallstudie wurde gemeinsam mit ProNES ein Evaluationsprojekt anhand der in Abbildung 5.1 dargestellten Batteriezellenprüfmaschine definiert. Hierbei lag der eigene Hauptaufwand in der eigenständigen Entwicklung, wobei der physische Aufbau der funktionalen Teile der Maschine bereits vorhanden war. Im Hinblick auf die fachlichen Expertisen der Stakeholder wurde Beratung von ProNES-Mitarbeitern eingeholt. Dadurch können die Modellierung und

die Informatik-Aussagen in Eigenverantwortung vertreten werden, während sich die Aussagen zum Maschinenbau und der Elektrotechnik hauptsächlich auf der Kommunikation mit den Stakeholdern von ProNES begründen.

Das Konzept der Prüfmaschine soll auf eine spezialisierte Anwendung für die Überprüfung von Batteriezellen ausgerichtet werden. Es handelt sich hierbei um eine moderne Vorrichtung zur präzisen Prüfung und Analyse von Batteriezellen in Batteriemodulen.

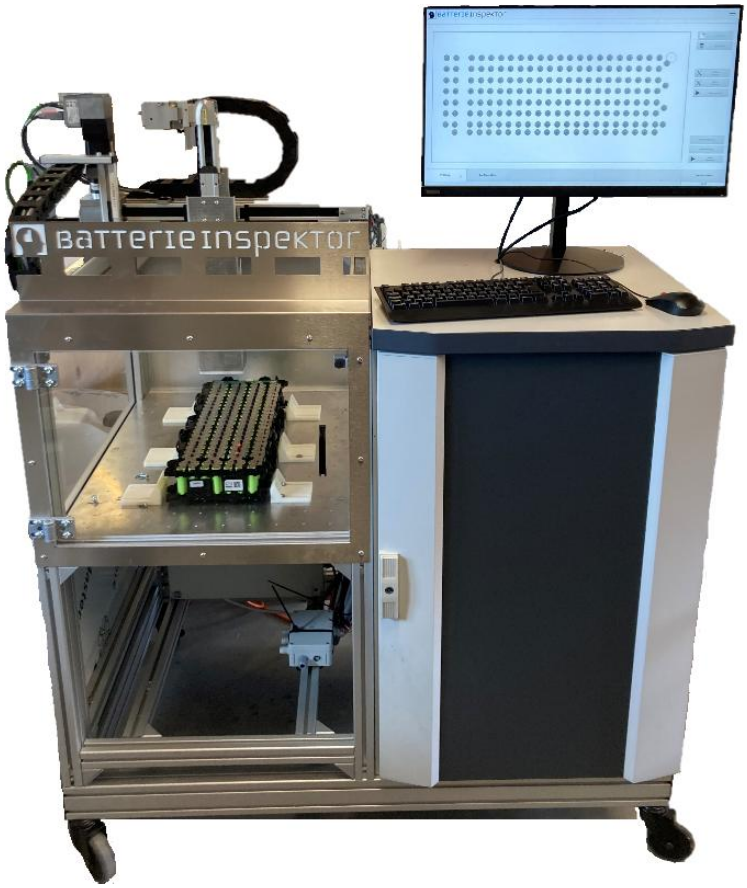


Abbildung 5.1: Batteriezellenprüfmaschine

Die Prüfmaschine besteht aus einem stabilen Tisch, der als Grundlage dient und auf dem der Batteriezellen-Prüfling sicher in einer speziellen Halterung eingelegt wird. Das Herzstück der Batteriezellenprüfmaschine ist ein Portalsystem, das über drei Linearachsen verfügt. Dieses System ermöglicht es dem Prüfkopf, präzise Bewegungen in den drei Dimensionen auszuführen, um eine genaue Positionierung über dem Prüfling zu gewährleisten. Die X- und Y-Achsen bieten eine horizontale Bewegungsfreiheit, während die Z-Achse für die vertikale Ausrichtung zuständig ist. Der Prüfkopf ist auf der Z-Achse des Portalsystems montiert und mit einem Digitalmultimeter ausgestattet, das in der Lage ist, Spannungsmessungen mit hoher Genauigkeit durchzuführen. Das Digitalmultimeter bietet hierfür verschiedene Messmodi, über die die gemessenen Werte mit einer Messunsicherheit von 0,01 % innerhalb 500 ms angezeigt und mit

Sollwerten verglichen werden können. Das System ist in der Lage, eine Vielzahl von Batteriemodulen mit unterschiedlichen Batteriezellen zu testen und die Ergebnisse zuverlässig zu erfassen. Für Evaluationszwecke dient das Batteriemodul AKASystem 15 OEM 50 PRC der Firma BorgWarner Akasol AG.

Auf Basis der Spannungsmessungen soll das Batteriemodul als in Ordnung (IO) oder nicht in Ordnung (NIO) eingestuft werden. Die Anlage soll für das Bedienpersonal so einfach wie möglich bedienbar sein. Es soll einen Startbutton geben, der eine Prüfung startet, sowie Maus und Tastatur, um vergangene Testergebnisse aus einer Datenbank abzurufen. Der Bediener erhält hierbei bei jeder Messung eine graphische Anzeige der gemessenen Werte und eine farbliche Kennung, ob alle Zellen eines Moduls korrekte Spannungswerte (min. 1,5V von 2,3V) besitzen. Insofern alle Zellen IO sind, bekommt der Bediener die Freigabe, den Prüfling als IO einzustufen. Für Inbetriebnahme, Wartung und Instandhaltung soll es einen Service-Modus geben, in dem die Anlage von geschultem Personal inspiziert und gewartet werden kann. Währenddessen muss ein voller Zugriff auf die interne Arbeitsweise der Anlage garantiert werden.

5.1.1 Bestimmung der Grenzen der Maschine

Als Erstes müssen die Grenzen der Maschine bestimmt werden, um UC5.4 zu evaluieren. Dafür werden die zuvor verbal beschriebenen Systemvorgaben aus dem Lastenheft in der gemeinsamen Modellebene mithilfe der modellierten `MachineLimitsRequirements` aus Abschnitt 3.2.11 umgesetzt. Dies ermöglicht die Definition von Verwendungsgrenzen, räumlichen Grenzen, zeitlichen Grenzen und weiteren Grenzen zur Umgebung gemäß den Anforderungen der ISO 12100. Auf dieser Grundlage kann das Systemdesign entwickelt werden. Abbildung 5.2 veranschaulicht diesen Prozess der Anforderungserstellung.

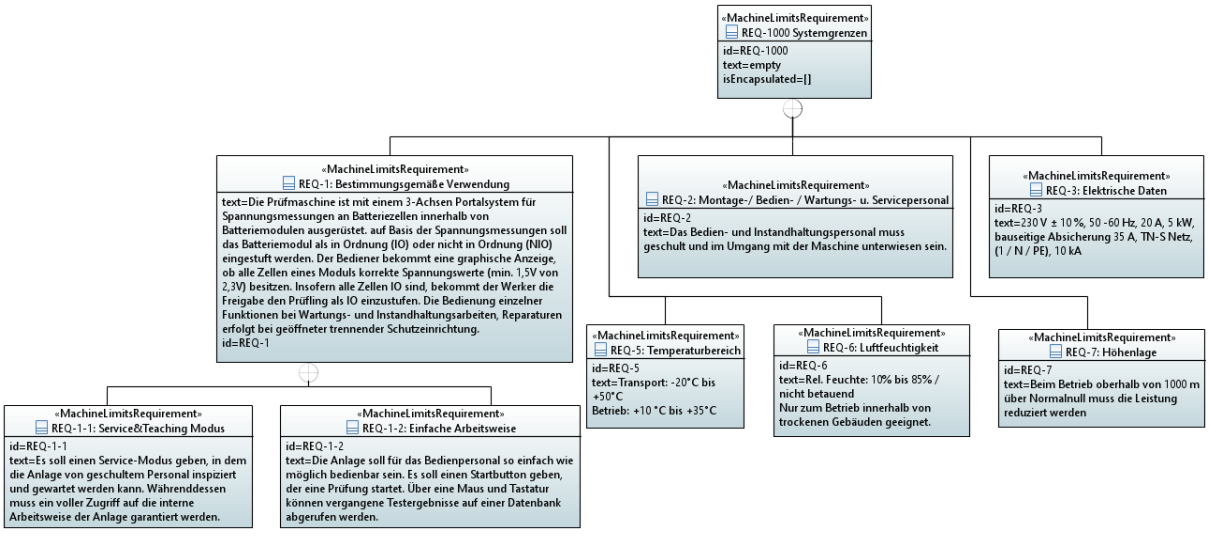


Abbildung 5.2: Grenzen der Maschine - Anforderungen

Um darauf aufbauend eine Grundlage zur Modellierung der Prüfmaschinenteile zu schaffen, werden von der Framework-Erweiterung die Metamodelle aus Abschnitt 3.2.6 verwendet, um Betriebsarten zu bestimmen und Schnittstellen zu definieren. Hierfür ist es zunächst erforderlich, den Anwendungskontext zu definieren, wie in Abbildung 5.3 dargestellt. Aus diesem Kontext geht hervor, dass die Prüfmaschine für den Einsatz in einer Industriehalle vorgesehen ist. Neben dem Bedienungs-, Service- und Instandhaltungspersonal können auch Dritte wie Reinigungspersonal oder vorbeilaufende Personen sowie andere Maschinen oder herumliegende Teile der Umgebung der Maschine ausgesetzt sein. Zur Modellierung wird bereits der Stereotyp `ExposedActor` aus Abschnitt 3.2.11 verwendet.

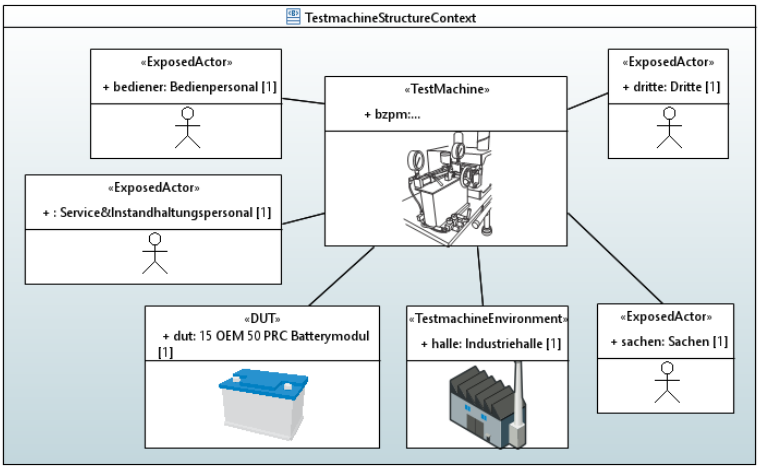


Abbildung 5.3: Grenzen der Maschine - Anwendungskontext

Basierend auf den grundlegenden verbalen Beschreibungen in Zusammenarbeit mit allen Stakeholdern können die relevanten Prüfmaschinenteile bestimmt werden. Wie in Abbildung 5.4 dargestellt, bestehen diese Teile neben dem 3-Achsen-Portalsystem, dem konstruktiven Aufbau und dem Voltmeter logischerweise aus einer Spannungsversorgung, einer Beckhoff-SPS, einer Bedieneinheit und einem Monitor. Zusätzlich gehören ein Rollwagen zur Zuführung und Entnahme des Prüflings sowie eine Signalsäule zur Anzeige des Betriebszustands der Prüfmaschine zu den Prüfmaschinenteilen. Die Unterteilung der Prüfmaschinenteile dient lediglich der Darstellung des funktionalen Verhaltens und einer übersichtlichen Risikobeurteilung. Je nach Prüfmaschine kann die Unterteilung feiner oder grober ausfallen.

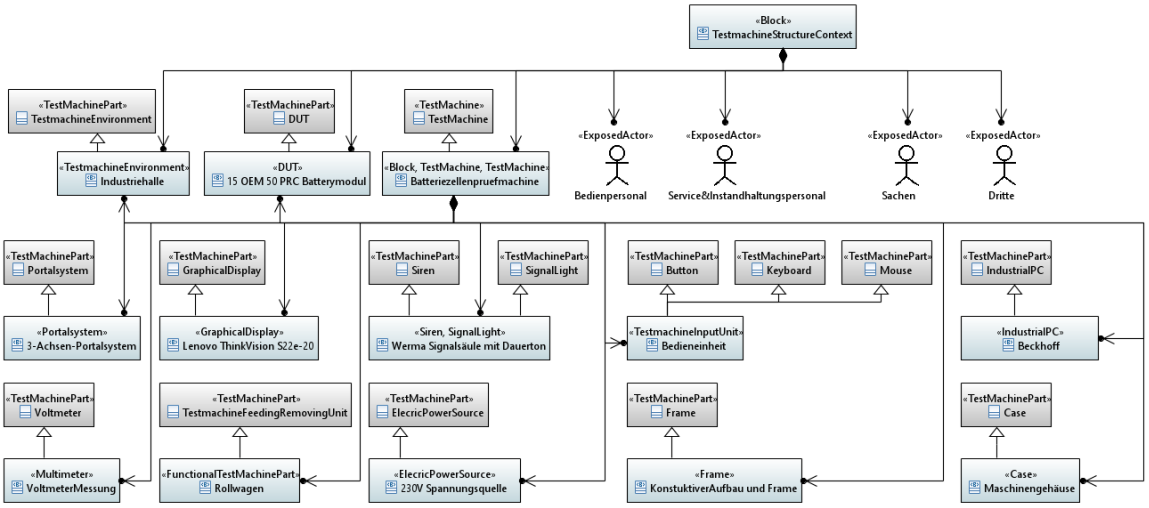


Abbildung 5.4: Grenzen der Maschine - Prüfmaschinenteile

Nachdem der strukturelle Aufbau der Prüfmaschine festgelegt ist, werden die verschiedenen Anwendungsfälle den identifizierten Akteuren zugeordnet, um verschiedene Betriebsarten umzusetzen. Abbildung 5.5 dient einerseits der Bestimmung der Grenzen der Maschine, bildet aber auch über die enthaltenen Modi einen Teil des Verhaltensmodells des Informatikers ab und zeigt dabei UC4.1 mit den Grundlagen für UC4.2.

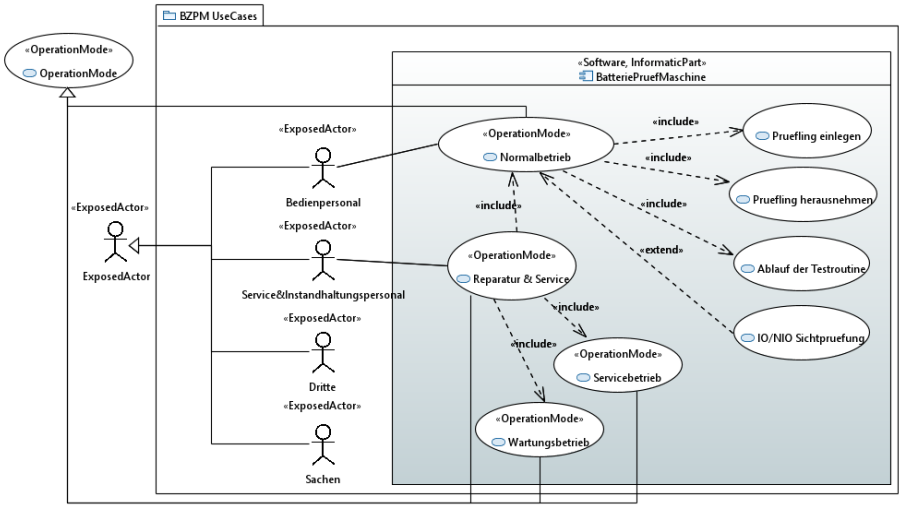


Abbildung 5.5: Grenzen der Maschine – Anwendungsfälle und Betriebsarten

Unter Verwendung des Stereotyps `OperationMode` aus Abschnitt 3.2.11 werden die in Abbildung 5.5 dargestellten Anwendungsfälle später auch den entsprechenden Gefährdungsereignissen zugeordnet, um Betriebsarten zuzuweisen. Dadurch entsteht eine strukturelle und verhaltensbezogene Darstellung aller erforderlichen Grenzen der Maschine für die Identifizierung von Gefährdungen und als Grundlage zur Modellierung von Gefährdungssituationen.

Um von der Gestaltung für die Stakeholder Maschinenbauer und Elektroingenieur ein anschauliches Bild zu vermitteln und UC2.1 bis UC2.5 sowie UC3.1 bis UC3.5 zu erfüllen,

wird anschließend anhand des Portalsystems das mechanische und elektrotechnische Design erläutert. Hierbei bietet die Gestaltung einzelner Maschinenteile keinen direkten Mehrwert für die Risikobeurteilung, da diese vor dem konkreten Design der Maschine stehen muss, die Konkretisierung kann aber dennoch einen indirekten Mehrwert fürs Verständnis und die Arbeitsweise einzelner Teile der Prüfmaschine bieten, was dem Sicherheitsingenieur die Arbeit erleichtert. Im Extremfall bedeutet dies aber auch eine mögliche Neumodellierung, insofern das Design am Ende der Risikobeurteilung als unzureichend eingestuft wird.

Da es sich bei SysML bereits um eine für Softwareentwicklungen und im Speziellen für Schnittstellenbeschreibungen und Algorithmen geeignete Modellierungssprache handelt, soll im Weiteren kein Fokus auf der Evaluation von UC4.3 bis UC4.5 gelegt werden. Eine im Verlauf des Evaluationsprojektes für die sicherheitsbezogene Entwicklung entwickelte Schnittstellenbeschreibung zur Evaluation von UC4.3 findet sich unter Anhang A.11.2 in Abbildung A.46. Zustandsdiagramme können die Softwarealgorithmen für UC4.4 beschreiben und finden sich unter Abschnitt 5.2 in Abbildung 5.31, sowie unter Anhang A.11.5 in Abbildung A.53, Abbildung A.65, Abbildung A.66 und Abbildung A.71 zur sicherheitsbezogenen Entwicklung. Hiermit können beide Use-Cases erfüllt werden. Für UC4.5 können lediglich Struktur und Verhalten modelliert werden, aber keine graphische Darstellung eines Mensch-Maschine-Interface (HMI). Ein entsprechendes HMI muss in der programmatischen Umsetzung innerhalb einer Entwicklungsumgebung wie TwinCAT stattfinden.

Der erste Schritt zur Evaluation von UC2.1 und UC3.1 stellt die Erstellung der Stakeholder-Aufteilung dar. Durch die Spezialisierung aus den unter Abschnitt 3.2.7 und 3.2.8 definierten Elementen `MechanicalAssembly` und `Electr(on)icalDesign` sowie Anwendung der Stereotypen lassen sich die Blöcke in Abbildung 5.6 dem zuvor modellierten 3-Achsen-Portalsystem zuweisen.

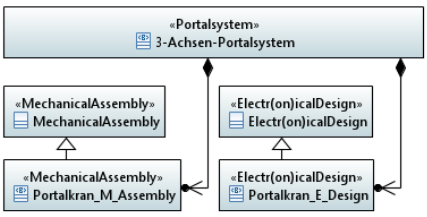


Abbildung 5.6: Portalsystem – BDD zur Stakeholder-Aufteilung

Anhand dieser Zuweisung entsteht die Grundlage für die Stakeholder-spezifischen BDD zur Evaluation von UC2.2 und UC3.2. Als Grundlage für den mechanisch konstruktiven Aufbau wird, wie in Abbildung 5.7 dargestellt, angenommen, dass jede Achse aus einer Befestigung,

einer Führungsschiene und dem Motor mit Ansteuerung besteht. Somit existiert für das Maschinendesign keine programmierbare Komponente und damit kein Software-Design. Zur Spezifikation von Materialien und Komponenteneigenschaften und damit der Evaluation von UC2.3 bis UC2.5 wurden neben den bereits durch die Metamodelle gegebenen Werten zu Gewicht, Maßen, Volumen, Position, Dimension und Orientierung Elementeigenschaften angelegt, die die minimale Zugfestigkeit von 245 N/mm^2 und Materialauswahl für eine leichte Bauweise durch Aluminiumprofile spezifiziert. Für die Z-Achse kann zusätzlich ein Haltemoment von 1,0607 Nm angegeben werden. Die Werte stammen aus Herstellerangaben und aus Rücksprachen mit dem Maschinenbauer von ProNES.

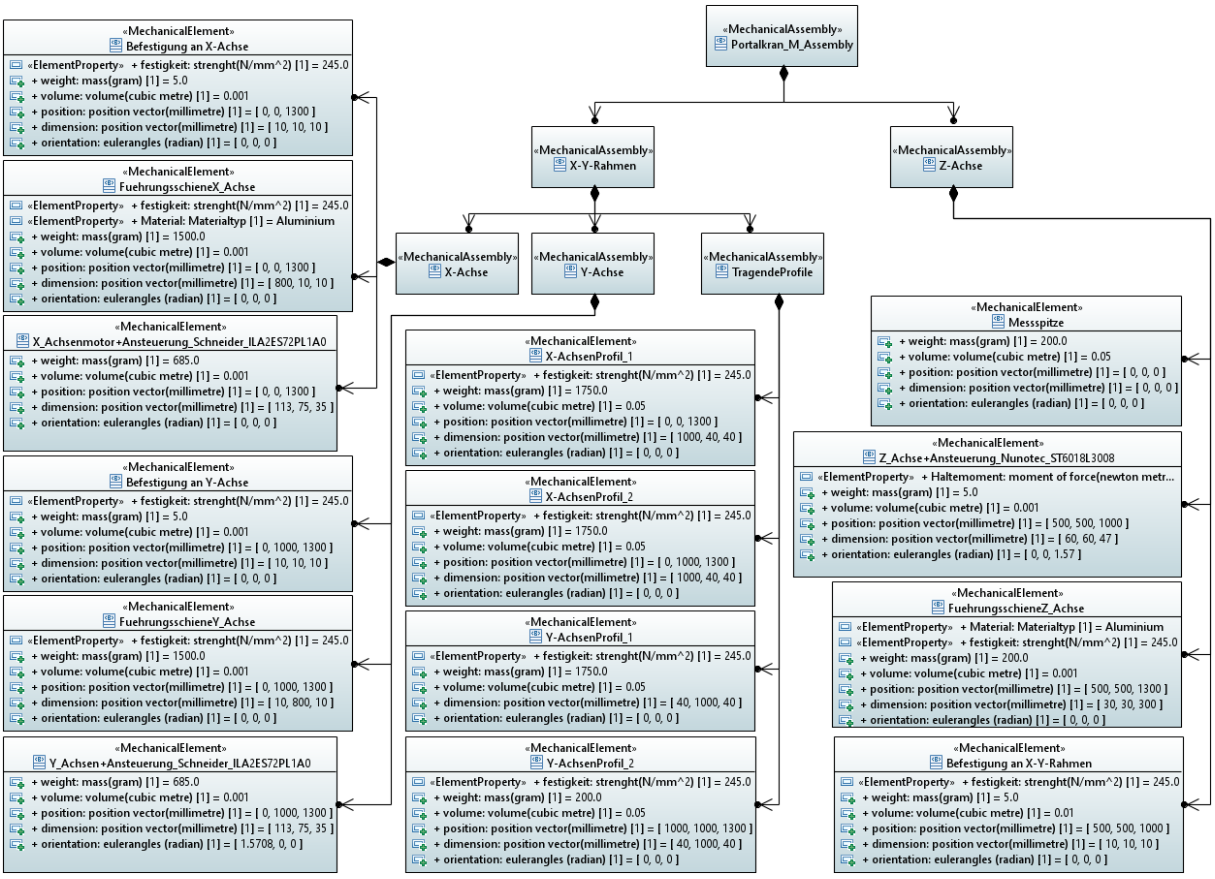


Abbildung 5.7: Portalsystem – BDD des Maschinenbauers

Um die Struktur und Vernetzung der einzelnen mechanischen Komponenten abzubilden, wird das in Abbildung 5.8 dargestellte IBD entworfen. Hieraus geht hervor, welche Baugruppen und Komponenten miteinander in Verbindung stehen. Beispielsweise geht hervor, dass die Messspitze an der Spitze der Z-Achse angebracht werden muss. Diese Darstellung kann von dem Maschinenbauer neben den Eigenschaften aus Abbildung 5.7 als Grundlage genutzt werden, um das maschinenbauspezifische Modell in AutoCAD umzusetzen.

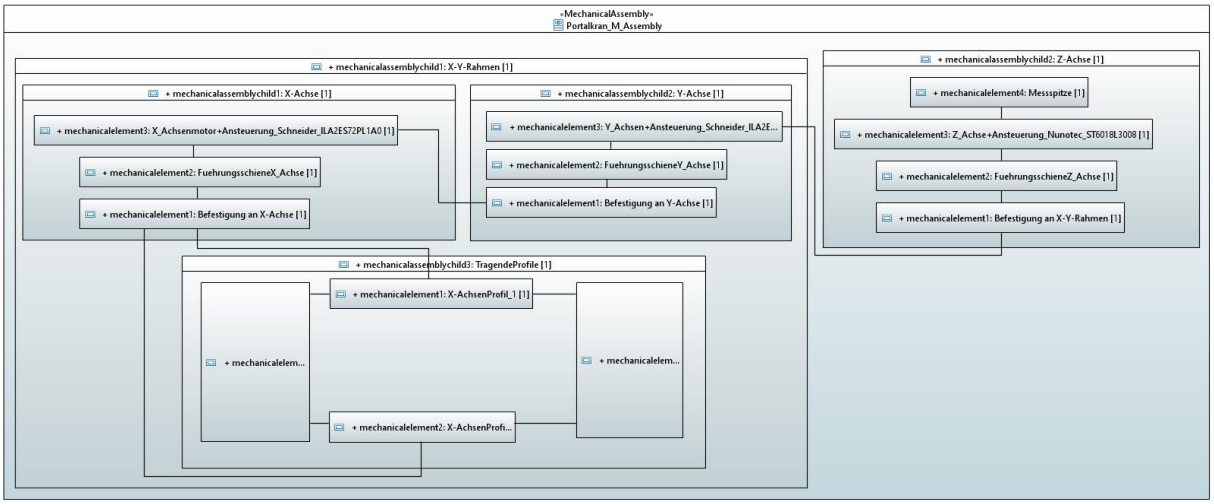


Abbildung 5.8: Portalsystem – IBD des Maschinenbauers

Anhand dieser Darstellung und der bereits im mechanischen Design entwickelten Komponenten wird das elektrotechnische Design entwickelt, um UC3.3 und UC3.4 zu evaluieren. Eine weiterführende Evaluation der UC3.3 bis UC3.6 findet sich unter Abschnitt 5.2.1 zur Gestaltung der Sicherheitsfunktionen. Bei der Durchführung der Fallstudie hat sich allerdings gezeigt, dass die Darstellbarkeit des mechanischen Modells noch verbessert werden kann. So bieten sich die zusätzliche Modellierung von Kontaktflächen, Art der Verbindung und Justierung, wie bereits durch [32] beschrieben, und einzelne Kategorien für Bauteilarten an.

Bei der elektrotechnischen Gestaltung können, so wie in Abbildung 5.9 umgesetzt, dieselben SysML-Elemente erweitert werden, was zum einen die Nachvollziehbarkeit stärkt und zum anderen eine eindeutigere Informationssammlung generiert. Eine andere Möglichkeit wäre, verschiedene SysML-Blöcke zu nutzen, wobei das elektrotechnische Element durch die Anwendung der Framework-Erweiterung dennoch auf das mechanische Element verweist.

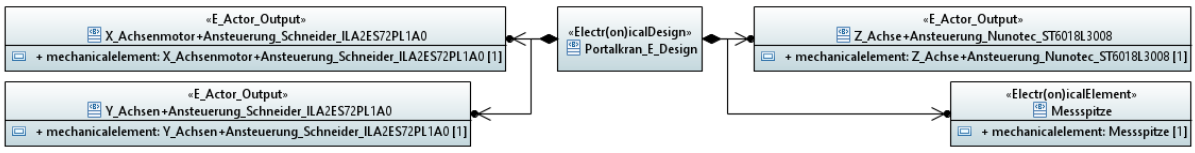


Abbildung 5.9: Portalsystem – BDD des Elektroingenieurs

Ein wesentlicher Teil des elektrotechnischen Designs ist die elektrische Verbindung zwischen den Komponenten, die sich auf Basis der durch die Framework-Erweiterung vergebenen Proxy Ports aller elektrotechnischen Elemente modellieren lässt. In einem IBD, wie dem in Abbildung 5.10, lässt sich ein entsprechender Item-Flow aufbauen. Neben dem aus dem Framework bekannten Ports sind hierfür einem elektrotechnischen Design eines Prüfmaschinenteils ebenso entsprechende Proxy Ports zuzuweisen, die die Signale nach außen führen und die Möglichkeit geben, das Prüfmaschinenteil mit anderen Teilen der Prüfmaschine zu verbinden, woraus das

elektrotechnische Gesamtdesign der Prüfmaschine hervorgeht. Für das Portalsystem kommt hierbei die Spannungsversorgung von der Spannungsversorgung der Batteriezellenprüfmaschine, die 24 V mit 10 A bereitstellt, und das Ethernet-Interface sowie A1, A2, B1 und B2 von der Beckhoff-SPS, die die Steuersignale für X-, Y- und Z-Achse bereitstellen. Die zweiadrige Messleitung führt hingegen zur Beckhoff-SPS und dem dort enthaltenen Digitalmultimeter. Hierbei kann sowohl im mechanischen als auch im elektrotechnischen Design eine Baugruppe, Unterbaugruppe oder eine Komponente in verschiedenen Maschinenteilen Einsatz finden, um das Gesamtdesign zu beschreiben. Beispielsweise bildet das mechanische Design des Portalsystems eine Baugruppe des konstruktiven Aufbaus und die Messspitze im Portalsystem eine Komponente des Voltmeters.

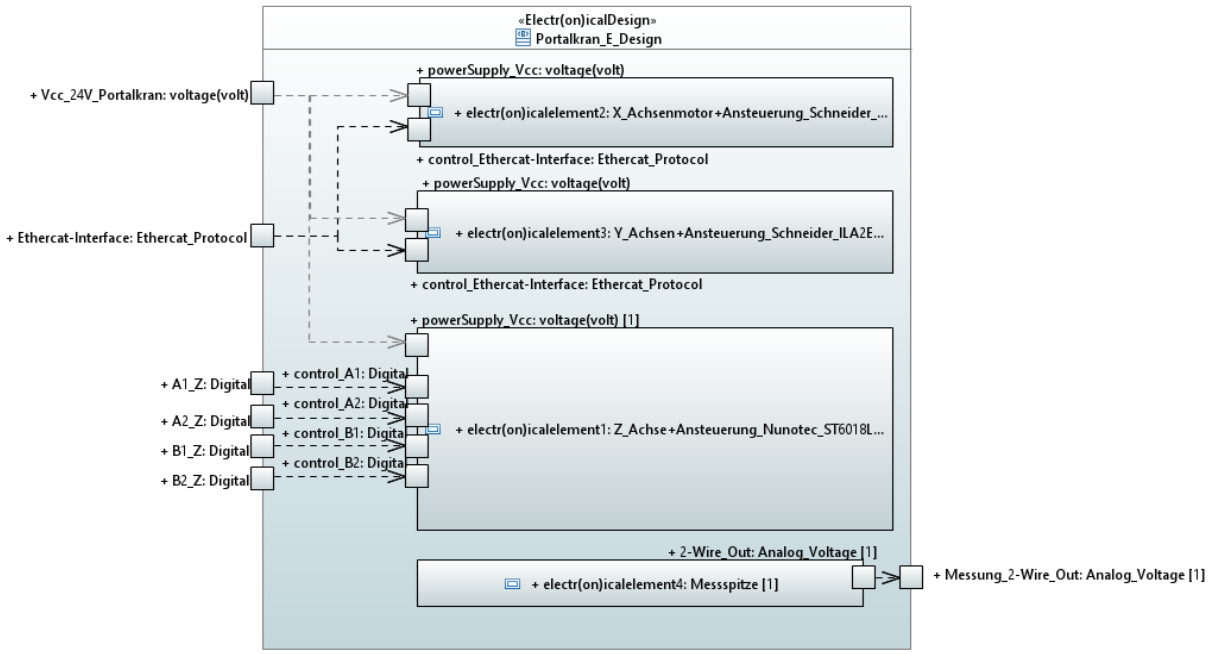


Abbildung 5.10: Portalsystem – IBD des Elektroingenieurs

Das elektrotechnische Design kann ebenso für ein gemeinsames Verständnis und die Weiternutzung in anderen Stakeholder-spezifischen Modellen, wie dem informationstechnischen Design, abgebildet und erweitert werden. Dies schafft einen Mehrwert in der Vernetzung durch voneinander abhängigen SysML-Elementen der gleichen Baugruppen, Unterbaugruppen und Komponenten. Ähnlich des mechanischen Designs bildet das elektrotechnische Design einen ersten Einblick in die Vernetzung der Komponenten und Grundlage für die Schaltplanerstellung in EPLAN, es kann sich allerdings auch hier anbieten, für eine einfachere Anwendbarkeit Kategorien für E/E/PE-Bauteilelemente zu schaffen.

5.1.2 Risikoeinschätzung

Im Allgemeinen kann der UC5.6 durch die Einschätzung von Gefährdungssituationen erfüllt werden. Hierbei werden die möglichen Folgen bestimmt, die das ungeminderte Ausmaß der Gefährdung beschreiben. Dies ermöglicht eine umfassende Risikoeinschätzung bei der Risikobeurteilung der Batteriezellenprüfmaschine. Durch eine sorgfältige Analyse der Gefährdungssituationen und der möglichen Folgen können Maßnahmen ergriffen werden, um das Risiko zu minimieren und die Sicherheit zu erhöhen. Insgesamt wurden die in Abbildung 5.11 dargestellten 24 Folgen identifiziert, von denen jede einzelne aus `MachineryEffect` aus Abschnitt 3.2.11 spezialisiert wird.

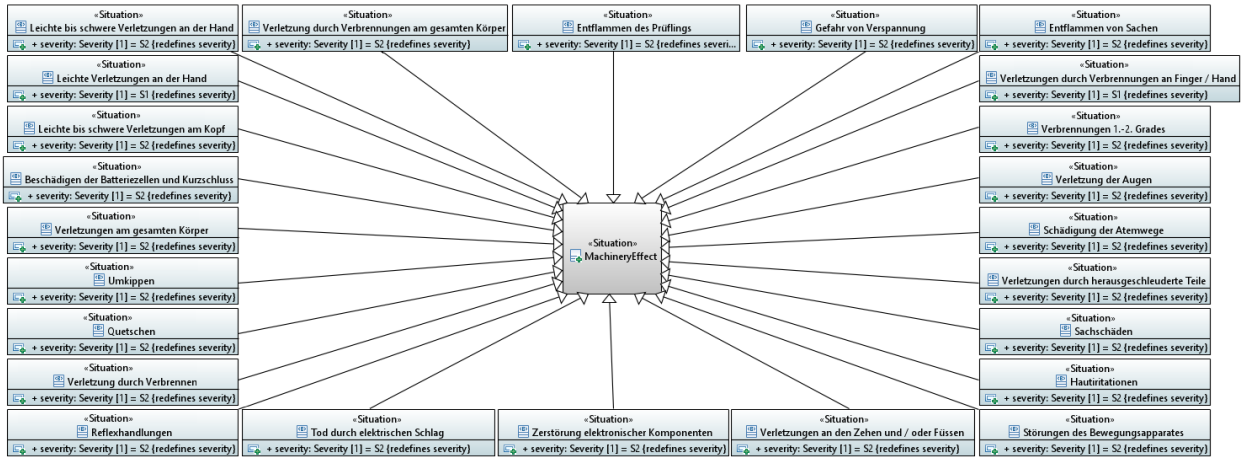


Abbildung 5.11: Bestimmung der Folgen von Gefährdungssituationen

Zusätzlich kann jeder Folge eine `Severity` zugeordnet werden, die die Schwere beschreibt. So haben beispielsweise Verbrennungen an Finger und Hand nur leichte reversible Verletzungen mit Stufe S1 zur Folge, wohingegen Verletzungen der Augen irreversible Verletzungen und somit S2 darstellen können. Zusätzlich können ebenso Folgen wie die Zerstörung von elektrischen Komponenten modelliert werden, die sich zwar nicht direkt auf Verletzungen von Menschen auswirken, doch diese sekundär durch unvorhersehbares Verhalten bei Fehlfunktion zur Folge haben können. Hierbei fällt direkt die Korrelation von Ursache und Folge ins Auge, die durch den Risikographen und die modellierten Artefakte zwar nicht direkt verfolgt wird, aber in Relation gesetzt werden kann. Beispielsweise verdeutlichen die zuvor unter Ursachen benannten Reflexhandlungen, dass die Ursachen eines Ereignisses die Folge eines anderen darstellen kann, was unabhängig der aufgezeigten Evaluation die Grundlage für Ursache-Wirkungs-Diagramme (UWD) bietet. Somit kann UC5.3 komplett abgebildet werden.

5.1.3 Identifizierung der Gefährdungen

Nachdem die Maschinengrenzen festgelegt wurden, beginnt der Prozess zur Risikobeurteilung (vgl. Abbildung 4.4 aus Abschnitt 4.2.1). Dabei werden die modellierten Prüfmaschinenteile betrachtet, ihr Verwendungszweck untersucht und potenzielle Gefährdungen erfasst. Es ist wichtig zu betonen, dass die Identifizierung von Gefährdungen, die Beschreibung von Gefährdungssituationen und die Identifizierung von Auswirkungen eine subjektive Einschätzung darstellen, die jeder Sicherheitsingenieur unter Anwendung der Grundlagen zur Entwicklung sicherheitskritischer Systeme und dem Stand der Technik individuell auf Basis seiner Erfahrungen vornimmt. Es besteht die Möglichkeit, dass dieselbe Maschine von verschiedenen Sicherheitsingenieuren unterschiedlich eingeschätzt wird, was zu unterschiedlichen Maßnahmen und Sicherheitsfunktionen führen kann, die allerdings den gleichen Berechnungsgrundlagen unterliegen. Die Einschätzungen innerhalb dieser Fallstudie stützen sich auf bereitgestellte Risikobeurteilungen ähnlicher Prüfmaschinen und Diskussionen mit dem Sicherheitsingenieur von ProNES. Auf Basis dessen wurden die Systemmodelle der Prüfmaschine selbst mit den Modellen der `ISO12100&ISO13849`-Bibliothek (Abbildung 3.18 in Abschnitt 3.2.11) erweitert.

Zur Evaluation von UC5.5 wurden zunächst die in Abbildung 5.12 abgebildeten Gefahren nach Arten von Gefährdungen für die Batteriezellenprüfmaschine identifiziert. Dabei bilden die einzelnen Gefahren Spezialisierungen der unter Abschnitt 3.2.11 in Abbildung 3.18 entwickelten Gruppen. Neben den mechanischen Gefährdungen durch Schneiden, Stoßen oder Quetschen an sich bewegenden und nicht bewegenden Teilen, die relativ offensichtlich erscheinen, werden zusätzlich die Standfestigkeit, mechanische Festigkeit oder Stechbewegungen betrachtet. Dabei werden risikomindernde Eigenschaften von Komponenten bei der ersten Risikoanalyse zunächst nicht betrachtet, um das komplette Spektrum der Gefährdungen zu erfassen. Im Verlauf entsteht dann die Möglichkeit, diese Sammlung entsprechend zu erweitern. Da es sich um einen iterativen Prozess handelt, fließen dabei spätestens in der zweiten Iteration Gefährdungen mit ein, die durch Sicherheitsmaßnahmen auftreten. Hierbei bietet sich das Beispiel der optischen Strahlung durch ein für den Wartungs- und Servicebetrieb zu gestaltendes Lichtgitter an, das als Sicherheitsfunktion während der ersten Iteration bei der Risikominderung eingeführt wurde.

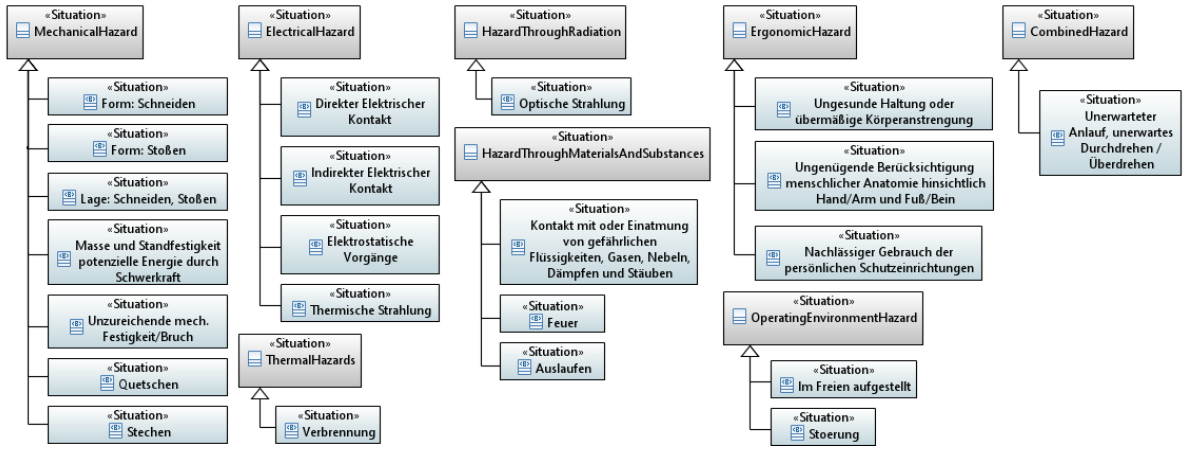


Abbildung 5.12: Bestimmung der Gefährdungen BDD

Abhängig der zu entwickelnden Maschine sind nicht alle Gefährdungsgruppen zu betrachten. Bei der Batteriezellenprüfmaschine treten keine Gefährdungen durch Lärm oder Vibration auf, da es in den zuvor entwickelten Prüfmaschinenteilen keine Teile wie beispielsweise große Wechselstrommotoren gibt, die sekundär Lärm oder Vibrationen erzeugen.

Da es zu umfangreich und teilweise für die Informatik nicht relevant ist, im Verlauf dieses Kapitels alle Gefährdungen zu beschreiben, sollen im weiteren Verlauf der Evaluation vornehmlich der direkte elektrische Kontakt, thermische Strahlung und Verbrennung betrachtet werden. Der unerwartete Anlauf der Maschine dient zusätzlich beim Vergleich identifizierter Risikoitems und risikomindernder Maßnahmen und Nachweis der vollständigen Darstellbarkeit der Modelle innerhalb des Evaluationsprojektes.

Jeder Gefährdung können wie vorgesehen eine oder mehrere Maschinenteile über Assoziationen zugeordnet werden, wie durch Abbildung 5.13 für den direkten elektrischen Kontakt, die thermische Strahlung und die Verbrennung veranschaulicht. Für den direkten elektrischen Kontakt handelt es sich um alle elektrischen Teile der Batteriezellenprüfmaschine, die an einer Spannungsversorgung angeschlossen sind. Die IBD dienen später der Veranschaulichung und Assoziation innerhalb eines Risikoitems, genauer beschrieben unter Abschnitt 5.1.4.

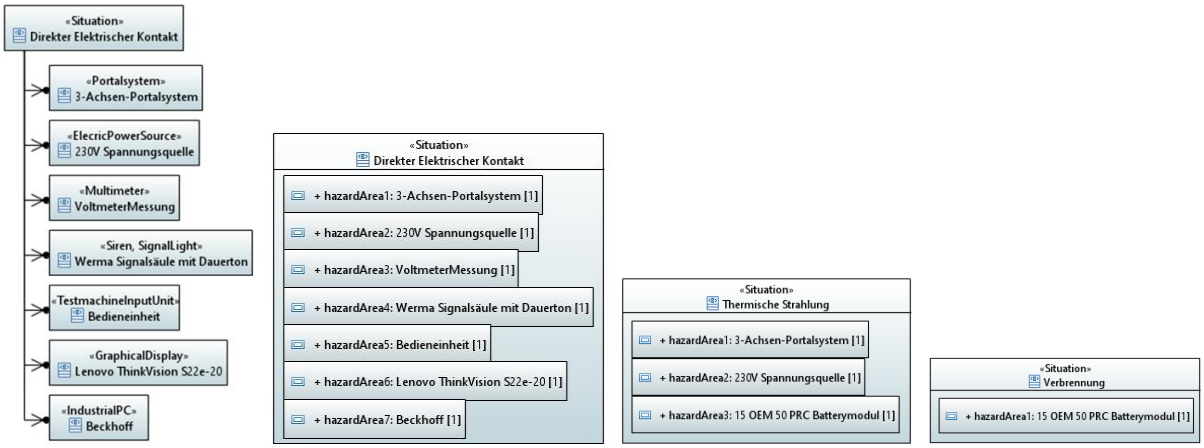


Abbildung 5.13: Bestimmung der relevanten Maschinenteile von Gefährdungen (Von links nach rechts: direkter elektrischer Kontakt BDD, direkter elektrischer Kontakt IBD, thermische Strahlung IBD, Verbrennung IBD)

Nachdem die Maschinenteile zugeordnet sind, können ineinander verschachtelt die Gefährdungssituationen, Gefährdungsereignisse und Ursachen ermittelt werden. Insgesamt wurden für die Batteriezellenprüfmaschine 37 Gefährdungssituationen identifiziert, die 34 Gefährdungsereignisse besitzen, die wiederum 25 Ursachen besitzen. Somit entsteht die Möglichkeit, mithilfe der Framework-Erweiterung, die zu entwickelnden Elemente durch mehrfache Verwendung zu minimieren, was zu einer übersichtlicheren und verstärkt vernetzten Risikobeurteilung beiträgt. Beispielsweise wird beim mechanischen Design zwischen innerer und äußerer Konstruktion unterschieden, was in den Gefährdungssituationen modelliert wird, sich aber auf dieselben Gefährdungsereignisse auswirkt. Ein Beispiel findet sich unter Anhang A.11.1 in Abbildung A.41.

Abbildung 5.14 zeigt die Gefährdungssituationen der elektrischen Gefährdungen, spezialisiert aus `HazardousSituation` aus Abschnitt 3.2.11. Dabei wirken sich direkte und indirekte Stromschläge lediglich auf Bedienpersonal, Service- & Instandhaltungspersonal und Dritte aus, wohingegen elektrostatische Aufladungen und chemische Vorgänge ebenso Sachen schädigen können.

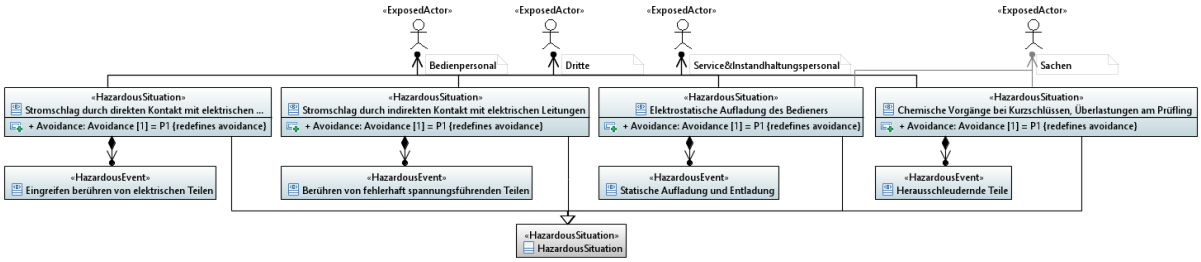


Abbildung 5.14: Bestimmung der elektrischen Gefährdungssituationen

Mit der Zuordnung jeweils eines Gefährdungsereignisses in Abbildung 5.14 können diesen, wie in Abbildung 5.15 aufgezeigt, Betriebsarten zugewiesen werden. Jedes Ereignis ist zunächst

aus `HazardousEvent` aus Abschnitt 3.2.11 zu spezialisieren. Für die Batteriezellenprüfmaschine existiert hier allerdings kein direkter Unterschied, da elektrische Gefährdungsereignisse unabhängig von den Betriebsarten sind. Das Ereignis `statische Aufladung und Entladung` modelliert hierbei das Auslösen eines Ereignisses durch mehrere in Zusammenhang stehende Ursachen. Zunächst kann sich durch unachtsame Handhabung und der Nichtbeachtung von Sicherheitsvorschriften, wie das Tragen von EDV-Kleidung, eine Person statisch aufladen, um sich dann mit oder ohne Reflexhandlungen an elektrischen Teilen zu entladen, was im Verlauf der Risikobeurteilung zu Effekten wie leichte Verletzungen an der Hand oder die Zerstörung von elektronischen Komponenten führen kann. Das Risikoitem findet sich zum Vergleich unter Anhang A.11.1 in Abbildung A.42.

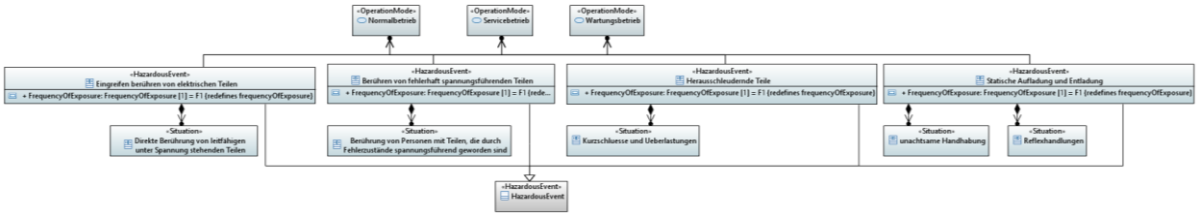


Abbildung 5.15: Bestimmung der elektrischen Gefährdungsereignisse

Zur konkreten Evaluation der Ursachen dient Abbildung 5.16, in der die Ursachen von `MachineContextCause` spezialisiert werden. Damit eine Ursache relevant ist, sollte diese mindestens in einer Lebensphase auftreten. In diesem Fall können alle Ursachen den Lebensphasen Installation, Einrichten, Teaching, in Betrieb nehmen, Betrieb, Instandhaltung, Reinigung, Troubleshooting, Wartung, außer Betrieb nehmen, Demontage und Entsorgung zugeordnet werden.

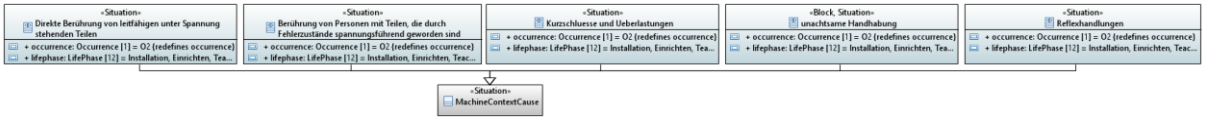


Abbildung 5.16: Bestimmung der elektrischen Ursachen für Gefährdungsereignisse

Die zwölf Lebensphasen können allerdings lediglich einem rein informativen Hintergrund und der Dokumentation dienen, anders als die in Abbildung 5.14, Abbildung 5.15 und Abbildung 5.16 beschriebenen Attribute `Avoidance`, `FrequencyOfExposure` und `Occurence`, die spezifisch die Berechnungsgrundlage des Risikos bilden. Damit lässt sich ableiten, dass Gefährdungssituationen hinreichend umgesetzt werden können und UC5.3 so weit erfüllt werden kann, da alle relevanten Merkmale und Assoziationen zwischen den Elementen modelliert werden können. Es fehlt allerdings noch eine Risikoeinschätzung, um die Grundlage für die Risikobewertung und der damit verbundenen Risikoitems in Abschnitt 5.1.4 zu schaffen. Die Risikoeinschätzung soll hierfür im folgenden Abschnitt diskutiert werden.

5.1.4 Risikobewertung und Identifikation der risikomindernden Maßnahmen

Anhand dieser Gefährdungen und Gefährdungssituationen aus Abschnitt 5.1.3 und der Folgen aus Abschnitt 5.1.2 kann im Folgenden die Risikobewertung zur Evaluation von UC5.7 durchgeführt werden, die die Grundlage für UC5.2 liefert. Hierzu werden, wie in Abbildung 5.17 für den direkten elektrischen Kontakt demonstriert, die einzelnen Elemente einem Risikoitem zugeordnet. Über den `IDCarrier` aus RAAML Core lässt sich eine menschenlesbare ID zuweisen, die eine spätere Zuordnung und beispielsweise in tabellarischer Form dargestellte Auflistung vereinfacht. Das Beispiel in Abbildung 5.17 gibt an, dass die Gefährdung `Direkter Elektrischer Kontakt` durch die Gefährdungssituation `Stromschlag durch direkten Kontakt mit elektrischen Leitungen` hervorgerufen werden kann und zu `Verletzung durch Verbrennen`, `Reflexhandlungen` und ebenso `Tod durch elektrischen Schlag` führen kann.

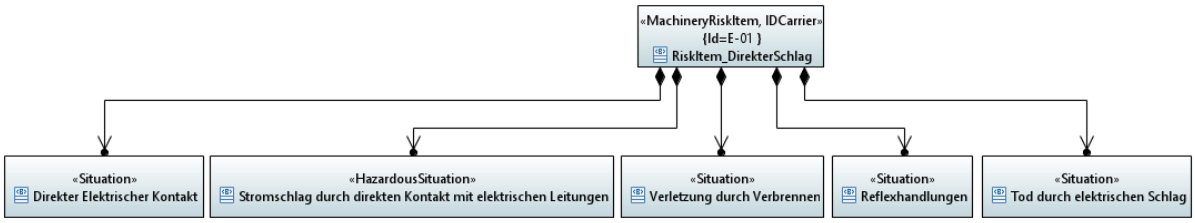


Abbildung 5.17: Zusammenhang zwischen Gefährdung, Gefährdungssituation und Folgen anhand direkten elektrischen Kontakt

Zur korrekten Evaluation und Umsetzung der Risikobewertung werden entsprechende IBD der Risikoitems erstellt. Dies bietet die Möglichkeit, auf einem Blick alle Attribute zu betrachten und die Einflussfaktoren zur Berechnung des PL_r sowie des Risikoindexes, falls noch nicht vom Sicherheitsingenieur komplettiert, auszufüllen. Abbildung 5.18 illustriert dies für den direkten elektrischen Schlag und bereits durchgeführter Berechnung von PL_r und Risikoindex.

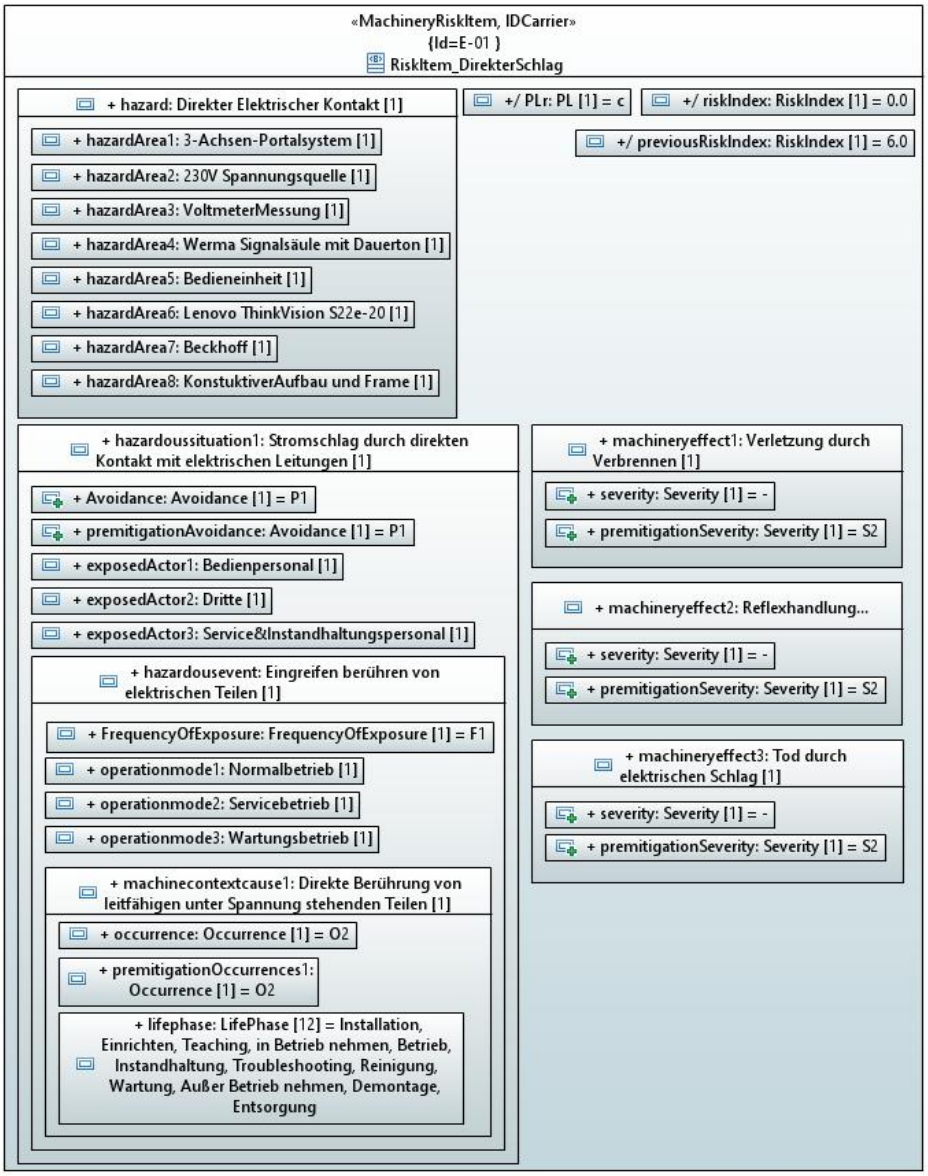


Abbildung 5.18: Bewertung des Risikos eines direkten elektrischen Schlags

Nach der ersten Berechnung des Risikoindexes (vgl. Abbildung 5.18 Attribut `previousRiskIndex`), wie in Abschnitt 4.2.4 beschrieben, hat sich ein zu hohes Risiko herausgestellt, das nicht toleriert werden darf. Zur Risikominderung wurden dann die risikomindernden Maßnahmen in Abbildung 5.19 entwickelt. Ein direkter elektrischer Schlag kann hier durch eine sichere elektrische Konstruktion verhindert werden, die sich im Einsatz bewährter Bauteile und der normativen Vorgaben gemäß EN 60204 zur Prüfung der Bauteile und Gesamtkonstruktion äußert. Weil ein gewisses Restrisiko nicht auszuschließen ist, soll in der Betriebsanleitung ein Warnhinweis enthalten sein, der den fachgemäßen Umgang mit der Batteriezellenprüfmaschine durch Fachpersonal vorschreibt. Dadurch kann das Ausmaß eines elektrischen Schlags auf ein vernachlässigbares Maß reduziert und der Risikoindex beim zweiten Durchlauf der Risikobewertung auf null reduziert werden. Die Berechnung des PL_r wurde zwar durchgeführt, findet

aber an dieser Stelle keinen weiteren Nutzen. Zu dessen Evaluation dienen die im Weiteren beschriebenen Beispiele aus der Risikobeurteilung.

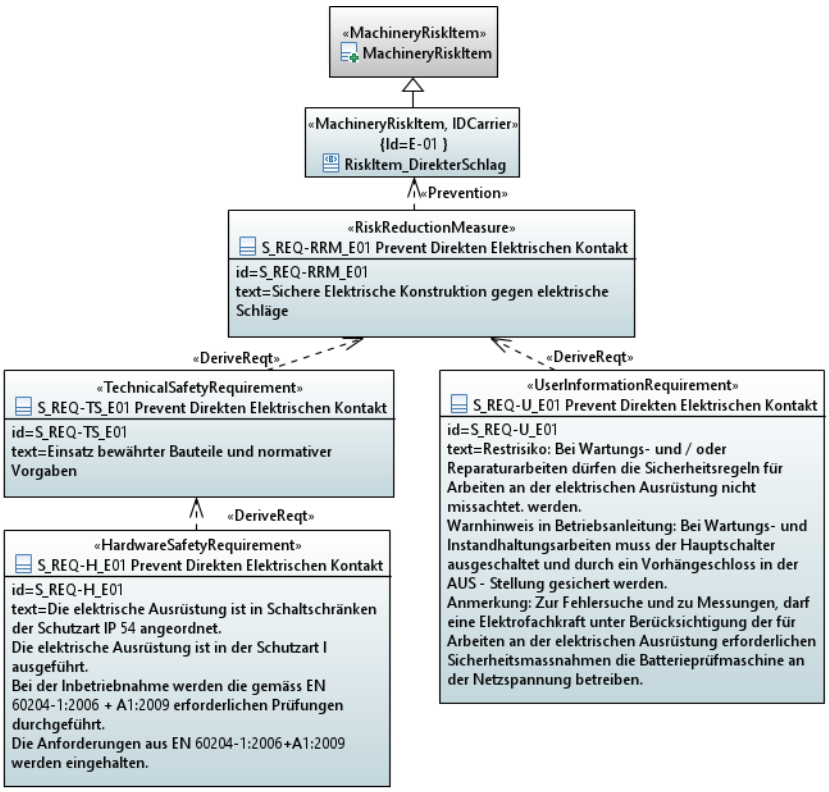


Abbildung 5.19: Identifikation der Maßnahmen gegen direkten elektrischen Schlag

Da das Risiko nicht durch technische Schutzmaßnahmen reduziert werden kann, besteht die risikomindernde Maßnahme lediglich in einer inhärent sicheren Konstruktion zur Evaluation von UC5.8, die sich in einer Hardwareanforderung widerspiegelt, und in einer Benutzerinformation zur Evaluation von UC5.10.

Für die Gefährdung der thermischen Strahlung sieht dies etwas anders aus. Aus der Gefährdungssituation von chemischen Vorgängen bei Kurzschlüssen kann die Spannungsversorgung überlastet werden, was zum Entflammen des Prüflings, der Anlage oder anderer in der Umgebung stehenden Objekten und Verbrennungen am gesamten Körper aller in der Nähe befindlichen Personen führen kann. Es entsteht ein ähnlicher Risikoindex, wie zuvor beim direkten elektrischen Schlag. Eine ausführliche Darstellung des Risikoitems findet sich zum Vergleich unter Anhang A.11.1 in Abbildung A.66. Zur Minderung des Risikos soll, wie in Abbildung 5.20 dargestellt, ein sicheres Design nach Schutzart IP54 als konstruktive Maßnahme zugrunde gelegt werden. Da dies nicht ausreichend ist, wird eine technische Schutzmaßnahme in Form eines Überspannungsschutzes installiert, die zur Evaluation von UC5.9 dienen soll. Sie besteht aus Sicherungen, Motorschutzschaltern in den Motoren, Leitungsschutzschaltern in Leistungs-

komponenten, Strombegrenzung in den Servoumrichtern, Strombegrenzung in den Stromversorgungsgeräten sowie eine ausreichende Dimensionierung von Kabeln und Leitungen gemäß EN 60204.

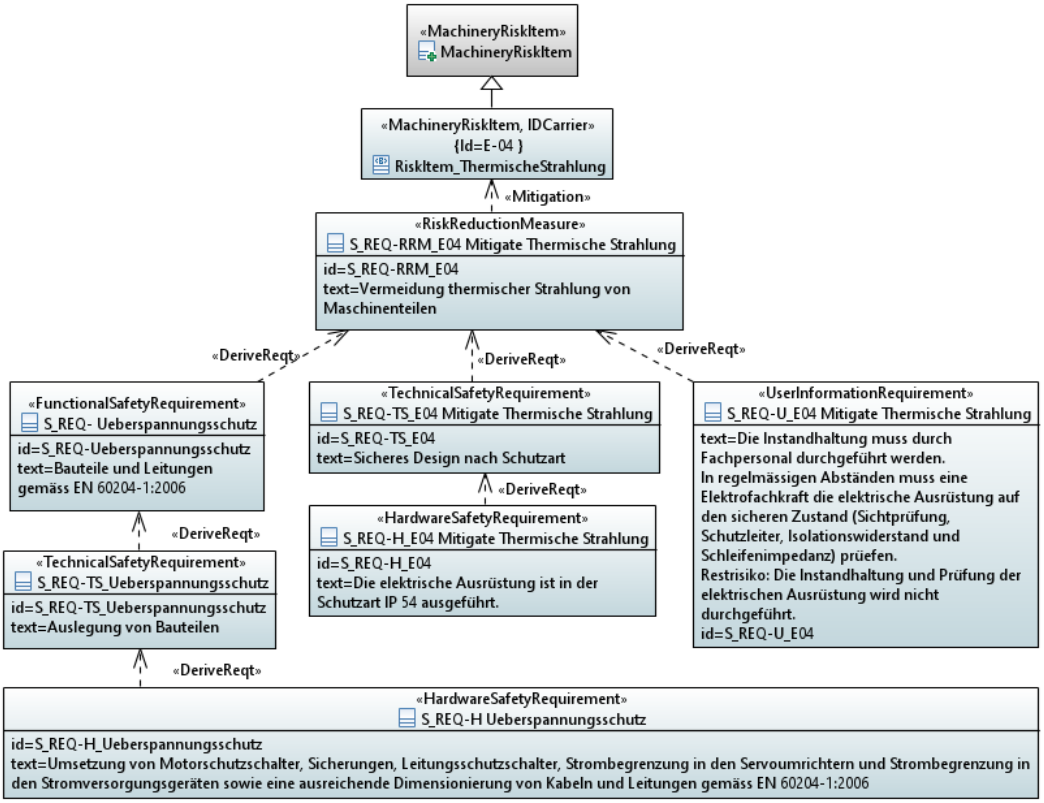


Abbildung 5.20: Identifikation der Maßnahmen gegen thermische Strahlung

Dabei ermöglicht die Modellierung die Nutzung derselben Sicherheitsanforderungen für unterschiedliche Risikoitems. Dies kann beim Vergleich von Abbildung 5.21 und Abbildung 5.22 beobachtet werden, in denen softwaregesteuerte Sicherheitsfunktionen zur Evaluation von UC5.11 und UC5.12 die Risikominderung abbilden.

Für die Verbrennung findet sich das bewertete Risikoitem unter Anhang A.11.1 in Abbildung A.44 und für den unerwarteten Anlauf in Abbildung A.45. Zur Minderung von Verbrennungen entsteht ein PL_r c, welches für eine trennende Schutzeinrichtung und Temperaturerfassung umgesetzt werden muss. Beim weiteren Verlauf der Risikobeurteilung kommt für das Risikoitem zur Bewertung eines unerwarteten Anlaufens, durch die Vielzahl an möglichen Gefährdungssituationen und die hohen Auswirkungen, ein PL_r d zum Tragen. Dies bedeutet somit, dass die in beiden Fällen benötigte trennende Schutzeinrichtung nach dem höheren PL zu modellieren und umzusetzen ist und dass PL_r c überschrieben wird.

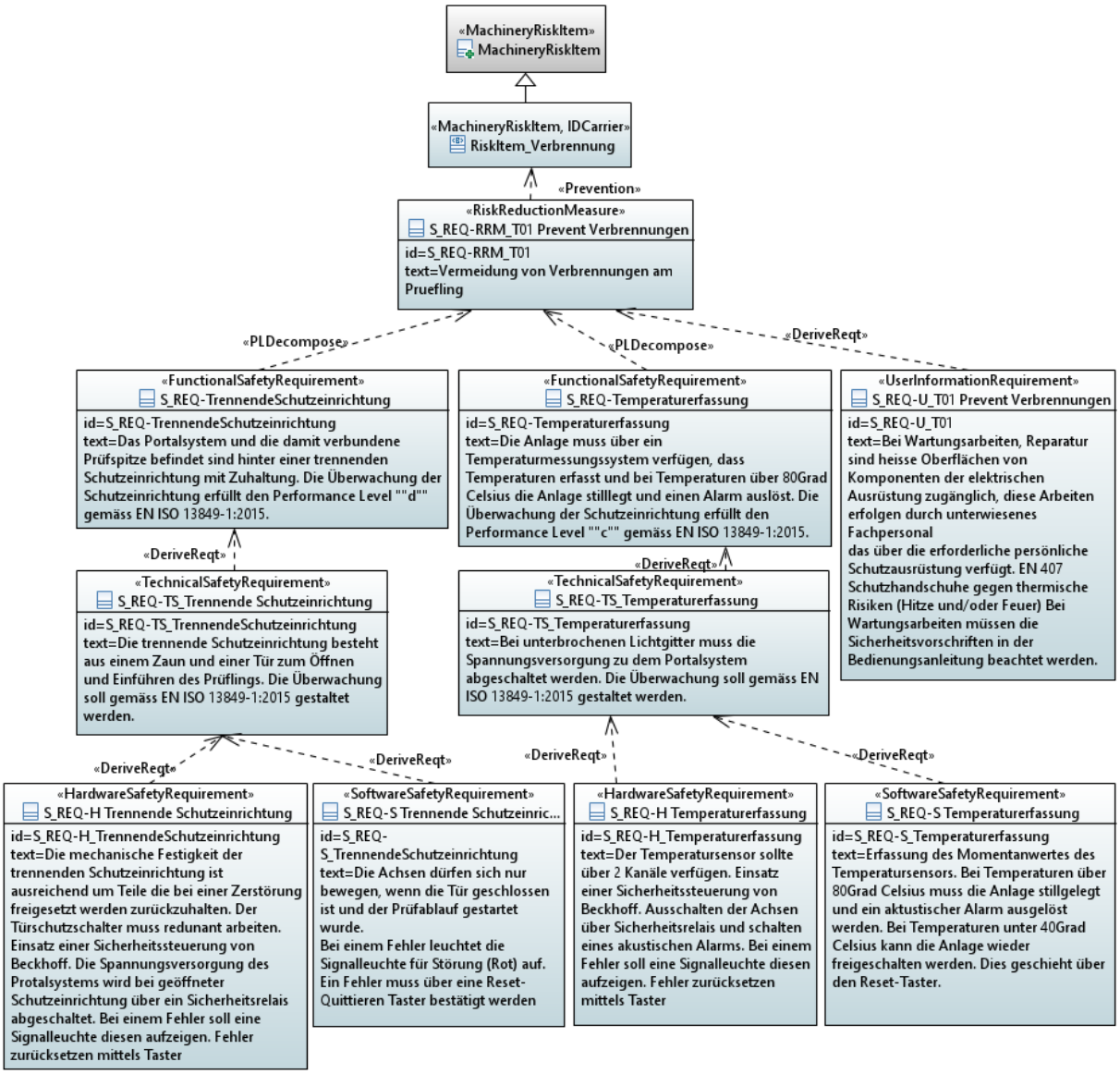


Abbildung 5.21: Identifikation der Maßnahmen gegen Verbrennungen

Abbildung 5.21 zeigt die Aufteilung von Hardware- und Softwaresicherheitsanforderungen, zugeordnet zu technischen Sicherheitsanforderungen, die wiederum Teil von funktionalen Sicherheitsanforderungen sind, die für die Sicherheitsfunktionen den jeweilige PL_r spezifizieren. Die Hardware- und Softwareanforderungen wurden hierbei gemeinsam mit den technischen Stakeholdern Maschinenbauer, Elektroingenieur und Informatiker entwickelt, um sicherzustellen, dass diese voraussichtlich den vorgegebenen Sicherheitsanforderungen entsprechen. Dies bestätigt die Modellierbarkeit von UC5.11 und UC5.12 über Anforderungsbeschreibungen.

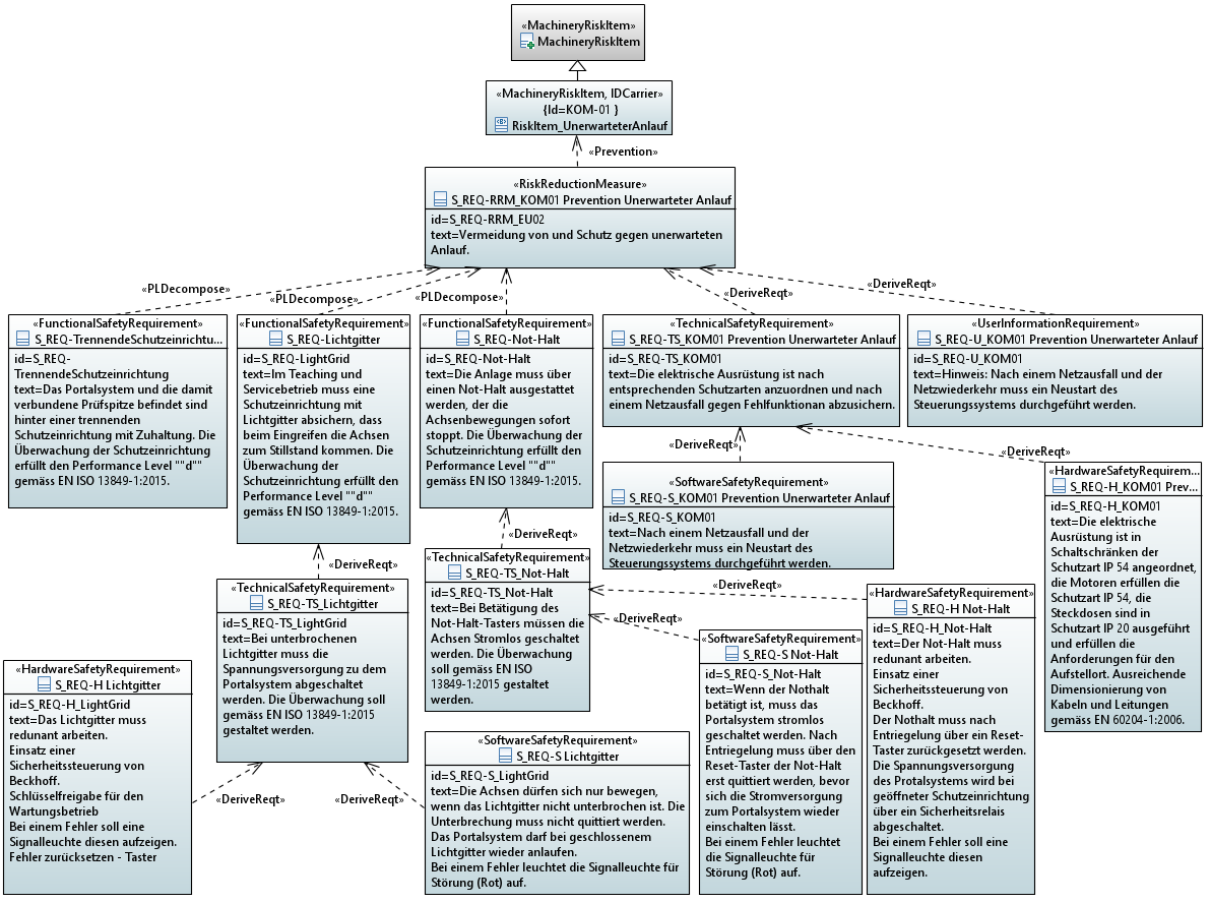


Abbildung 5.22: Identifikation der Maßnahmen gegen einen unerwarteten Anlauf

Somit kann abschließend zur Evaluation der Risikobeurteilung festgestellt werden, dass durch die bereitgestellte Form der Modellierung eine übersichtliche Risikobeurteilung entsteht. Doch kann die Zuweisung von Gefährdungssituationen und Folgen innerhalb eines Risikoitems darunter leiden, da sie untereinander nicht zugeordnete Elemente darstellen, ohne direkten Bezug zueinander (vgl. Abbildung A.44). Die Berechnung des Maximalwerts für `Avoidance`, `Severity`, `FrquencyOfExposure` und `Occurence` führt dadurch zu höheren modellierten Risiken und damit zur Notwendigkeit höherwertiger Sicherheitsfunktionen. Eine Möglichkeit, diese Zuweisungsprobleme bei der Modellierung zu vermeiden und Risikoindex sowie PL_r zu begrenzen, besteht darin, in jedem Risikoitem lediglich einer Situation eine Folge zuzuweisen, was aber eine höhere Anzahl von Elementen begründet.

Damit kann für UC5.8 bis UC5.10 eine hinreichende Modellierbarkeit nachgewiesen werden.

5.1.5 Diskussion

Die Gesamtbewertung der Modellierung zur Risikobeurteilung in der Maschinenindustrie und zur Abbildung von Prüfmaschinen ist Teil jedes Unterabschnitts in Abschnitt 5.1. Da die Risikobeurteilung und die Abbildung von Prüfmaschinen die Gesamtlösung darstellen, trägt jeder

bewertete Aspekt direkt zu dieser Forschungsaufgabe bei. Unter Abschnitt 5.1.1 bis 5.1.4 wurden die Anwendungsfälle des in der Modellierung vorgestellten Anwendungskontextes für Maschinenbauer, Elektroingenieur, Informatiker und Sicherheitsingenieur verwendet, um zu bestätigen, dass die Modellierung von Prüfmaschinen und der Risikobeurteilung in der Lage sind, die Anforderungen zu erfüllen. In allen Fällen wurde das Ergebnis als positiv bewertet und Kritikpunkte (vgl. S. 171, S. 172 und S. 183) beschrieben.

Es muss jedoch zusätzlich festgestellt werden, dass die Abbildung von Prüfmaschinen und die Risikobeurteilung derzeit nicht wie UML und SysML vollständig in die Modellierungsumgebung integriert sind. So besitzen die MMBSEFE-Elemente in Papyrus keinen eigenen Reiter zur Konfiguration der Einstellungen, sondern sind über den Profilanwendungsreiter anzuwenden. Um eine weitere Verbesserung von MMBSEFE für weniger erfahrene Entwickler zu erreichen, müssen ggf. weitere Anwendungsfälle berücksichtigt oder Änderungen an den vorhandenen Anwendungsfällen vorgenommen werden. Wenn die Framework-Erweiterung zudem in anderen Modellierungsumgebungen als Papyrus angewendet werden soll, müssen möglicherweise andere Technologien und angepasste Darstellungen gewählt werden. So entspricht der Einsatz von CSS in MagicDraw nicht den Möglichkeiten in Papyrus. Es kann zudem davon ausgegangen werden, dass andere Anwendungen von Prüfmaschinen einen anderen Ansatz haben und die Metamodelle der modellierbaren Prüfmaschinenteile möglicherweise erweitert oder teilweise geändert werden müssen, um solchen Anforderungen gerecht zu werden. Positiv stellt sich allerdings dar, dass Prüfmaschinenteile ebenso für andere Maschinentypen als Vorlage dienen können. Hierbei sind die modellierten Prüfmaschinenteile abhängig der notwendigen Funktionalitäten, die durchaus große Ähnlichkeiten zu Funktionalitäten anderer Maschinentypen haben können. Die notwendigen Funktionalitäten der Prüfmaschinenteile konnten im Rahmen der Arbeit zwar breitflächig, doch nicht vollständig betrachtet werden, da hierfür weitere Fallstudien notwendig werden. Dies stellt ein typisches Problem der Modellierung dar. Diese und weitere Anpassungen können bei Bedarf Teil zukünftiger Arbeiten sein.

Die Bewertung der Forschungsaufgabe zeigt damit, dass die Integration der Modellierungslösung für die ausgewählte Implementierung umsetzbar ist. Wenn MMBSEFE eine weitflächige Anwendung finden soll, sind jedoch noch weitere Verbesserungen notwendig.

5.2 Integration der Analysemethoden des Maschinenbaus

In diesem Abschnitt wird eine detaillierte Bewertung der Integration verschiedener Sicherheitsanalysemethoden der Maschinenindustrie vorgestellt und somit FA2-E umgesetzt. Ziel dieser Aktivität ist es, zu bewerten, ob die Modellierung und Implementierung der Risikobeurteilung

durch Risikographen und des Nachweises zur Risikominderung durch DFTs und SBBM eine anwendbare Lösung bieten kann. Um dies zu erreichen, wurden in der Fallstudie gemäß der Forschungsaufgabe des Typs Experiment im Bereich Integration der Analysen des Maschinenbaus die Sicherheitsanalysemethoden DFT und SBBM eingesetzt und verglichen. Unterabschnitt 5.2.1 zeigt die Gestaltung einer Sicherheitsfunktion auf und führt hierbei die aus Abschnitt 5.1 bekannte Prüfmaschinenmodellierung fort, um die Modellierung von Sicherheitsfunktionen zu bestätigen. Unterabschnitt 5.2.2 bewertet den Einsatz von DFTs und Unterabschnitt 5.2.3 den von SBBM anhand einer Sicherheitsfunktion. Innerhalb von Unterabschnitt 5.2.4 wird ein Vergleich der Ergebnisse der zwei Analysemethoden anhand aller identifizierten softwaregesteuerten Sicherheitsfunktionen dargestellt und in Unterabschnitt 5.2.5 wird das angewandte Architekturkonzept zur losen Toolkopplung der Risikographmethode mit einer engen Toolkopplung verglichen und bewertet. Schließlich gibt Unterabschnitt 5.2.6 eine Zusammenfassung des Abschnitts 5.2.

5.2.1 Gestaltung der Sicherheitsfunktionen

Zur Weiterführung der in Abschnitt 5.1 eingeführten Fallstudie wurden die Sicherheitsfunktionen für die Batteriezellenprüfmaschine entwickelt, um zu analysieren, inwieweit MMBSEFE in der Lage ist, UC2.6, UC3.6 und UC4.6 zu erfüllen. Hierfür werden weitere und detailliertere Stakeholder-spezifische Modelle entwickelt und somit die Basis für die Bestimmung der Risikominderung durch DFTs und SBBM gelegt. Beispiele für Stakeholder-spezifische Modelle wurden bereits in Abbildung 5.5 bis Abbildung 5.10 gegeben. In diesem Abschnitt wird nun eine detailliertere Modellierung mit Schwerpunkt auf die Nutzbarkeit der Metamodelle der Sicherheitsanalysen gegeben.

Im ersten Schritt werden die zuvor in Abbildung 5.4 entwickelten Prüfmaschinenteile der Batteriezellenprüfmaschine in Abbildung 5.23 für die Sicherheitsfunktionen erweitert. Neue Maschinenteile sind hellgelb eingefärbt. Die Grundlage legt eine Beckhoff Sicherheits-SPS Lösung `TwinSafe System` zur Umsetzung der SRASW für alle softwaregesteuerten Sicherheitsfunktionen. Diese kann als Erweiterung der zuvor definierten Beckhoff-SPS `Beckhoff` verstanden werden. Zusätzlich werden für jede identifizierte Sicherheitsfunktion entsprechende Maschinenteile aus den in Abschnitt 3.2.6 bereitgestellten Stereotypen abgeleitet.

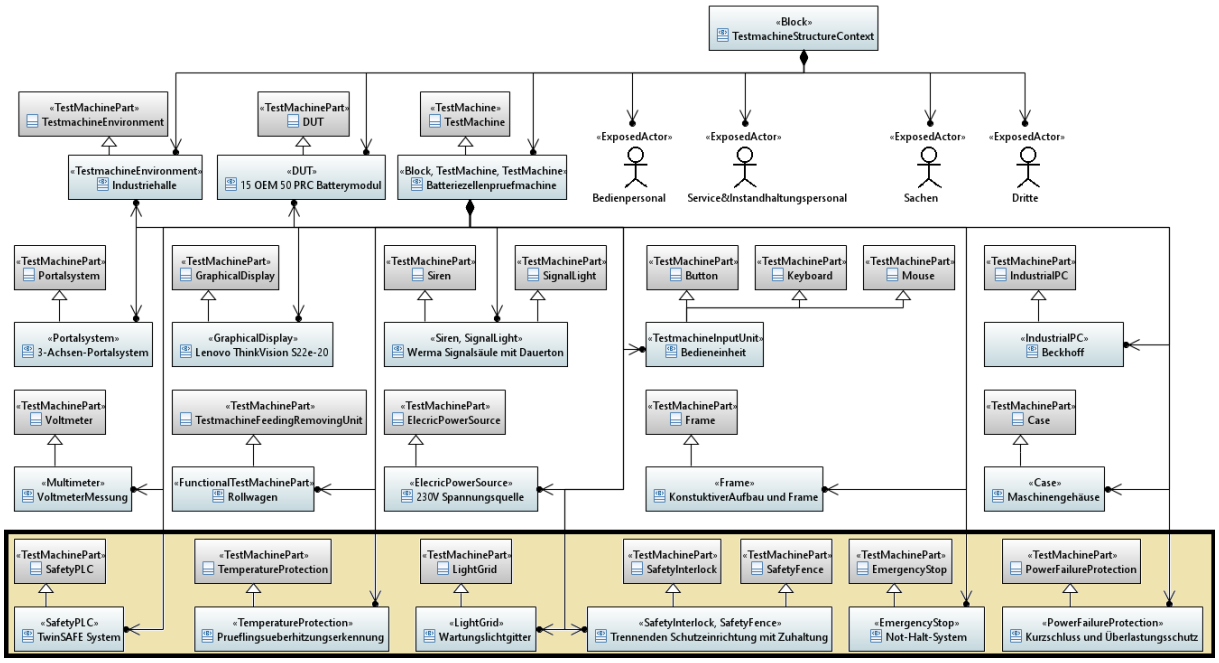


Abbildung 5.23: Grenzen der Maschine – Prüfmaschinenteile nach Risikobeurteilung

Im nächsten Schritt werden die zuvor in Abbildung 5.5 entwickelten Anwendungsfälle in Abbildung 5.24 für die Sicherheitsfunktionen erweitert, was die Grundlage für UC4.6 des Informatikers bildet. Neue Anwendungsfälle sind hellgelb eingefärbt. Hierbei werden ausschließlich die für die Arbeit interessanten softwaregesteuerten Maßnahmen ausgewählt und betrachtet. Beim Einlegen oder Herausholen des Prüflings muss im Normalbetrieb die Schutztür geöffnet und geschlossen werden, was den Anwendungsfall `TurOeffnen` beeinflusst, unter dem die trennende Schutzeinrichtung umgesetzt wird. Hierbei tritt zusätzlich der Anwendungsfall `Schutz vor Eingreifen in die Anlage` mit der Sicherheitsfunktion des Lichtgitters in Kraft, wenn die Anlage im Service- oder Wartungsbetrieb ausgeführt wird. Zur Abdeckung der Sicherheitsmaßnahmen laufen beide Sicherheitsfunktionen parallel, wobei bei einer Schlüsselfreigabe für Service oder Wartungsarbeiten die Schutztür der trennenden Schutzeinrichtung deaktiviert werden kann. Die Sicherheitsfunktion zur Umsetzung des Nothalts befindet sich im Anwendungsfall `Not-Halt betätigen`. Zuletzt findet sich hinter dem Anwendungsfall `Temperaturüberwachung` die Prüflingsüberhitzungserkennung, ausgerichtet auf den Hauptgefährdungsbereich des Prüflings.

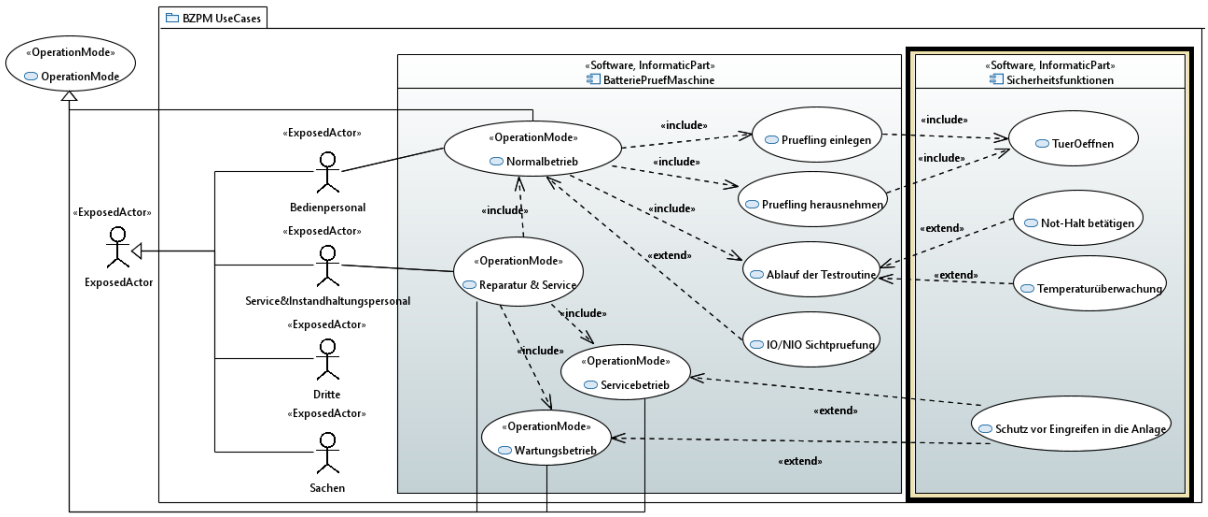


Abbildung 5.24: Grenzen der Maschine – Anwendungsfälle und Betriebsarten nach Risikobeurteilung

Als Umsetzung einer Sicherheitsfunktion wird im Folgenden exemplarisch die Prüflingsüberhitzungserkennung ausgewählt. Analog hierzu findet sich unter Anhang A.11.3 in Abbildung A.47 bis Abbildung A.53 die Entwicklung der Sichten der technischen Stakeholder zum Nothalt-System.

Ähnlich Abbildung 5.6 repräsentiert Abbildung 5.25 die Einteilung des Maschinenteils in die Sichtweisen der technischen Stakeholder. Hierbei findet sich nun zusätzlich die Entwicklung von Software durch den Informatiker.

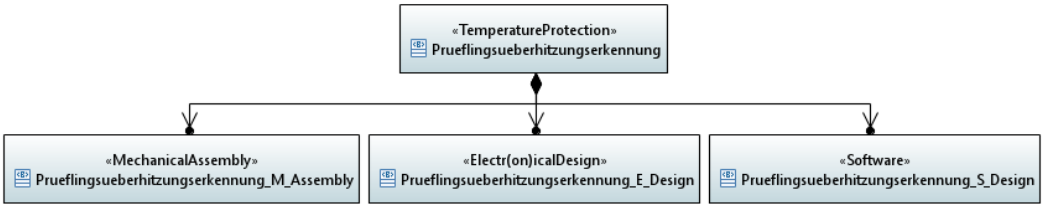


Abbildung 5.25: Prüflingsüberhitzungserkennung – Stakeholder-Aufteilung

Für den Maschinenbauer liegt der Fokus auf den zusätzlichen Elementen, die für die mechanisch-konstruktive Entwicklung der Prüflingsüberhitzungserkennung notwendig werden. In Abbildung 5.26 sollen daher die Eingangsmodule, das `TwinSafe System` selbst und die Ausgangsmodule außen vor gelassen werden, was sich in einem übersichtlicheren Teildesign im Vergleich zu Abbildung 5.7 äußert. Elemente ohne direkten Einfluss für den Maschinenbauer, können in anderen Maschinenteilen zusammengefasst werden, wie in diesem Fall `Beckhoff` bzw. `TwinSAFE System`.

Im IBD in Abbildung 5.27 kann der Maschinenbauer die Beziehungen zwischen den einzelnen Komponenten aufbauen. Da für die Sicherheitsfunktion lediglich der Alarm aus der Signalsäule

und der Reset-Taster aus der Bedieneinheit relevant sind, können auch hier weitere Maschinenelemente ausgeblendet werden. Im Fall der Signalsäule und der Bedieneinheit handelt es sich um Maschinenteile, die bereits zuvor vollständig definiert wurden. So entsteht zusätzlich die Möglichkeit, auf die anderen IBD der Maschinenteile zu navigieren, um einen ausreichenden Kontext zu liefern. Abbildung 5.27 zeigt zusätzlich, dass der Jumo Temperatursensor PT100 am Y-Achsen-Profil angebracht werden muss, um die Temperaturerkennung direkt über dem Prüfling zu erreichen.

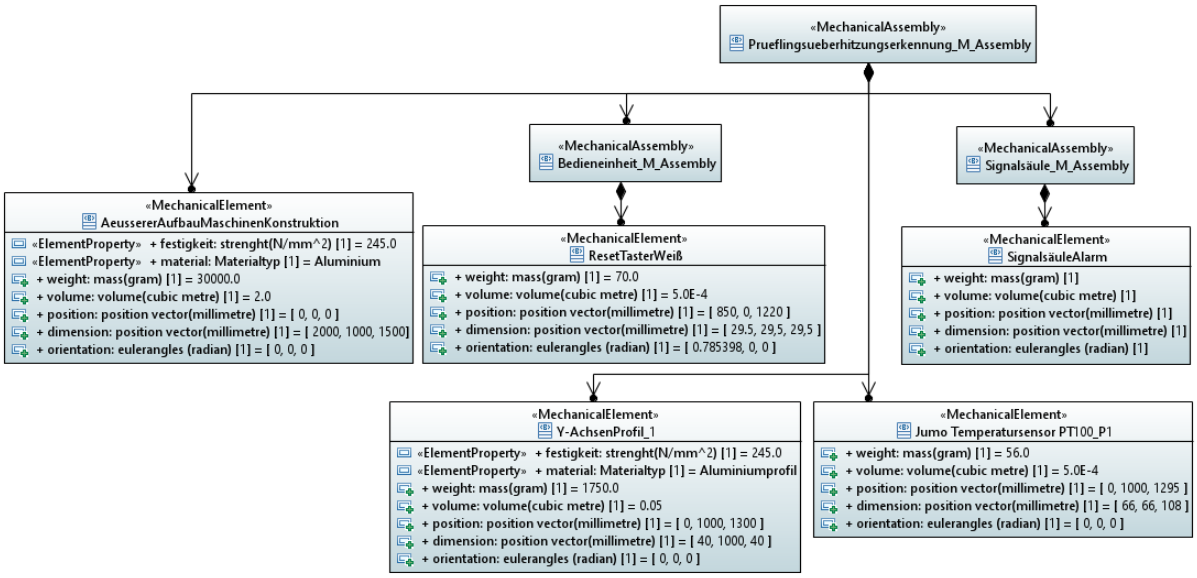


Abbildung 5.26: Prüflingsüberhitzungserkennung – BDD des Maschinenbauers

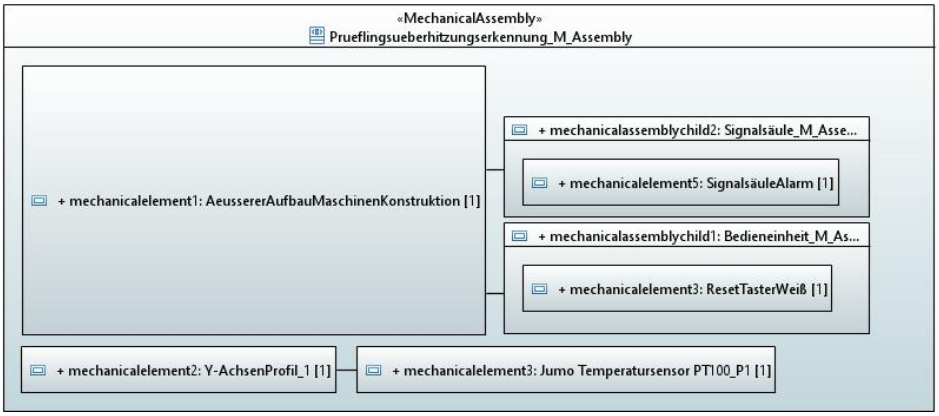


Abbildung 5.27: Prüflingsüberhitzungserkennung – IBD des Maschinenbauers

Somit wird für UC2.6 erweiternd zu UC2.1 eine ausreichende Modellierbarkeit erreicht.

Für den Elektroingenieur liegt der Fokus auf der vollständigen Abbildung aller elektrischen und elektronischen Elemente der Sicherheitsfunktion. Dies ist besonders wichtig, da jedes einzelne Element zur Verschaltung beiträgt und als Teil des Nachweises der Risikominderung in Betracht kommen kann. Abbildung 5.28 legt daher alle für die Prüflingsüberhitzungserkennung

eingesetzten Elemente fest. Zur Umsetzung der Sicherheitsfunktion werden laut des entsprechenden `HardwareSafetyRequirement` in Abbildung 5.21 ein Reset-Taster, ein Temperaturfühler PT100, eine SPS, ein Sicherheitsrelais und ein Alarmsignal benötigt. Da Temperaturfühler vereinfacht dargestellt nur temperaturabhängige Widerstände darstellen, wird ebenso ein Temperaturregler JUMO SafetyM vorgesehen, der ein analoges Eingangssignal für die Verarbeitung in der SPS bereitstellt. Die Beckhoff-SPS selbst besteht aus einem digitalen Eingangsmodul EL1904, einem analogen Eingangsmodul EL3124, der Verarbeitungseinheit EL6900 und einer Ausgangseinheit EL2904.

Über die in Abbildung 5.28 dem `Prueflingsueberhitzungserkennung_E_Design` zugewiesenen Elemente lässt sich im IBD in Abbildung 5.29 eine schaltplanähnliche Darstellung umsetzen. Anhand der bereits in der Bibliothek aus Abschnitt 3.2.8 Abbildung 3.13 verankerten Ports und zusätzlichen modellierten Ports werden die einzelnen Elemente miteinander verbunden und so ein Informationsfluss definiert. Abbildung 5.29 soll lediglich als Beispiel dienen und kann für eine detailliertere Verschaltungsdarstellung detaillierter ausgeführt werden.

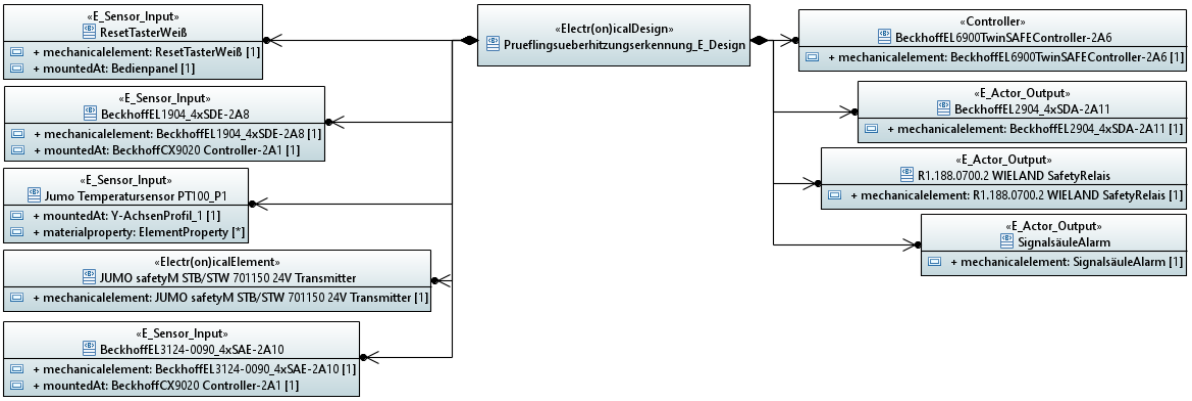


Abbildung 5.28: Prüflingsüberhitzungserkennung – BDD des Elektroingenieurs

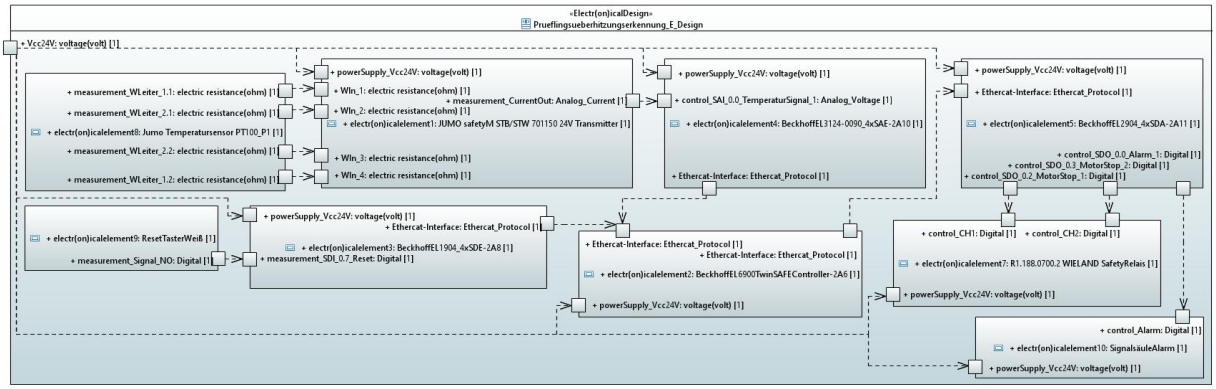


Abbildung 5.29: Prüflingsüberhitzungserkennung – IBD des Elektroingenieurs

Somit wird für UC3.6 erweiternd zu UC3.1 eine ausreichende Modellierbarkeit erreicht.

Den letzten Schritt zum Aufbau der technischen Modelle betrifft den Informatiker. Laut Abbildung 5.30 findet hierbei lediglich eine Komponente Einsatz, die auf der Beckhoff-SPS läuft und das Zustandsdiagramm zur Verhaltensbeschreibung der Software enthält.

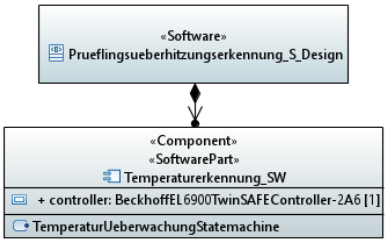


Abbildung 5.30: Prüflingsüberhitzungserkennung – BDD des Informatikers

Das in Abbildung 5.31 dargestellte Zustandsdiagramm wurde anhand des `SoftwareSafety-Requirement` in Abbildung 5.21 entwickelt. Es gibt an, dass bei einer gemessenen Temperatur von größer 80 °C der Alarm eingeschaltet und die Motoren der Achsen abgeschaltet werden. Bei einer Anfangsbedingung unter -20 °C wird die Sicherheitsfunktion nicht aktiviert. Ist die Sicherheitsfunktion ausgelöst, lässt sich diese über den Reset-Taster nur zurücksetzen, sofern die Temperatur unter 40 °C gesunken ist und somit wieder eine sichere Arbeit an der Prüfmaschine möglich ist. Das Zustandsdiagramm spezifiziert das Verhalten des Algorithmus zur Prüflingsüberhitzungserkennung vollständig.

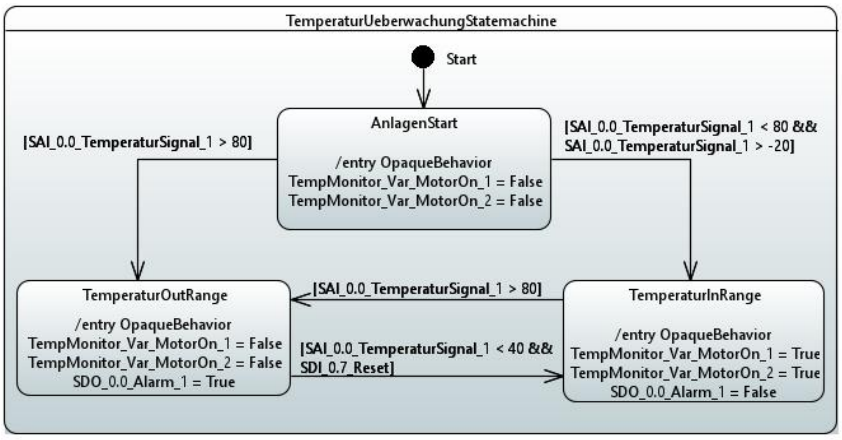


Abbildung 5.31: Prüflingsüberhitzungserkennung – Zustandsdiagramm

Die Zustandsdiagramme der weiteren softwaregesteuerten Sicherheitsfunktionen befinden sich in Anhang A.11.5. Dabei können mehrere Sicherheitsfunktionen die Abschaltung der Achsenmotoren einleiten. Daher werden statt der tatsächlichen Ausgänge Variablen gesetzt, die im übergeordneten Diagramm (Abbildung A.65) mit den Ergebnissen der anderen Funktionen in Abbildung A.53, Abbildung A.66 und Abbildung A.71 weiterverarbeitet werden. Zusätzlich wird ein Eingangs- und Ausgangsinterface, dargestellt in Abbildung A.46, vorgesehen, um die

als digitale und analoge Spannungswerte konfigurierten Signale aus Abbildung 5.29 in die Software zu übertragen. Ein solches Interface trägt später bei der Entwicklung der Software zur klareren Datentypdefinition, Abbildung von invertierten Signalen, durch Öffner- oder Schließerkontakte und analogen Signalskalierung bei.

Somit wird für UC4.6 erweiternd zu UC4.1 eine ausreichende Modellierbarkeit erreicht.

5.2.2 Evaluation des Einsatzes von DFTs

Zur Bewertung der Modellierung von DFTs wird im Folgenden der dynamische Fehlerbaum für die Prüflingsüberhitzungserkennung entwickelt, berechnet und evaluiert. Hierfür sind zunächst die Ausfallraten der einzelnen Komponenten aus Abbildung 5.28 und Abbildung 5.29 vom Sicherheitsingenieur gemeinsam mit dem Elektroingenieur zu bestimmen, um die Grundlagen für die Bestimmung der Risikominderung zu schaffen. Diese lassen sich aus Datenblättern der Hersteller ablesen. Zusätzlich dienen die bereits in Abschnitt 2.2.1 in Formel 2.2 eingeführten und in Formel 5.1 auf λ_D umgestellten Formeln als Berechnungsgrundlage. Dabei wird MTTR als nahe null angesehen, wodurch MTTF gleich MTBF entspricht.

$$\lambda_D = \frac{0{,}1 * n_{op}}{B_{10D} * 8760h} = \frac{RDF}{MTTF * 8760h} = \frac{RDF}{(MTBF - MTTR) * 8760h}$$

Formel 5.1: Berechnung λ_D eines Elements [34]

Aus Formel 5.1 ergeben sich folgende in Tabelle 5.1 beschriebenen Ausfallraten (λ_D). Für den Reset-Taster wird vom Hersteller ein B_{10D}-Wert von 3.000.000 Zyklen bereitgestellt. Über die Annahme einer mittleren Betriebszeit von 365 Tagen im Jahr und 12 Stunden am Tag sowie einer mittleren Zeit zwischen zwei aufeinanderfolgenden Zyklen von maximal 15 min ergibt sich eine mittlere Zahl von 17520 Schaltspielen pro Jahr und somit ein λ_D von $6{,}667*10^{-8}$. Für alle restlichen Komponenten werden MTTF, MTBF oder direkt $MTTF_D$ bzw. PFH_D angegeben. Die Einführungen der Kenngrößen sind in Abschnitt 2.1.3 auf S. 32–34 und Abschnitt 2.2.1 auf S. 53–54 detailliert beschrieben.

Komponente	Herstellerangabe	Ausfallrate
Reset-Taster	$B_{10D} = 3.000.000$	$n_{op} = 17520$ $\lambda_D = 6{,}667 * 10^{-8}$
Beckhoff EL1904_4xSDE	$PFH_D = 1{,}1 * 10^{-9}$	$\lambda_D = 1{,}1 * 10^{-9}$
JUMO PT100	$MTTF_D = 171$	$\lambda_D = 6{,}676 * 10^{-7}$
JUMO safetyM STB/STW 701150	$MTTF_D = 336$	$\lambda_D = 3{,}397 * 10^{-7}$
Beckhoff EL3124-0090_4xSAE	$MTBF_D = 950000(\text{h})$ $RDF = 50\ \%$ (k. A.)	$\lambda_D = 5{,}285 * 10^{-7}$
BeckhoffEL6900TwinSAFE	$PFH_D = 1 * 10^{-9}$	$\lambda_D = 1 * 10^{-9}$
BeckhoffEL2904_4xSDA	$PFH_D = 1{,}3 * 10^{-9}$	$\lambda_D = 1{,}3 * 10^{-9}$
Wieland R1.188.0700.2 S-Relais	$PFH_D = 3 * 10^{-8}$	$\lambda_D = 3 * 10^{-8}$
Signalsäule Werma Alarm	$MTTF = 5$ $RDF = 50\ \%$ (k. A.)	$\lambda_D = 1{,}142 * 10^{-5}$

Tabelle 5.1: Berechnung der Ausfallkriterien der Prüflingsüberhitzungserkennung

Auf dieser Grundlage lassen sich, wie in Abbildung 5.32 gezeigt, die Bestandteile des DFT auf Basis der Metamodelle aus Abschnitt 3.3.2 und der elektrotechnischen Sicht aus Abbildung 5.29 erstellen.

Jede Komponente wird als nicht reparierbar mit einer negativ exponentiellen Verteilungsfunktion angenommen und den entsprechenden Maschinenelementen zugeordnet. Da der Reset-Taster und das damit verbundene Beckhoff-Eingangsmodul EL1994 keinen Einfluss auf die Sicherheitsfunktion besitzen, werden diese Komponenten in der folgenden Analyse außen vor gelassen. Für den dynamischen Fehlerbaum wird eine Betriebszeit der Anlage von 20 Jahren, also 175200 Stunden, festgelegt, nach der PF_D für PL_r c einen Wert von 0,5256 nicht überschreiten darf. Für die Verbindung und Anzeige des Fehlerbaums wird ein BDD und IBD erstellt, die in Abbildung 5.33 und Abbildung 5.34 dargestellt sind.

Abbildung 5.32 und Abbildung 5.33 zeigen bereits die Ergebnisse des Fehlerbaums nach der Durchführung der Analyse durch Stormchecker unter Nutzung des in Abschnitt 4.2.2 beschriebenen Transformators zum Nachweis der Risikominderung. Hierbei sind die Ergebnisse negativ ausgefallen, da PF_D bei ~0,8973 und somit über dem geforderten Limit von 0,5256 liegt. Zusätzlich liegt der berechnete $MTTF_D$-Wert unter 20 Jahren. In der Visualisierung wurden deshalb die verletzte Anforderung, der dynamische Fehlerbaum und das Top-Event durch den Transformator und die CSS-Datei rot eingefärbt. Dies ermöglicht es, die nicht erfüllten Anforderungen schnell zu erkennen und entsprechende Änderungen in den Modellen zu planen.

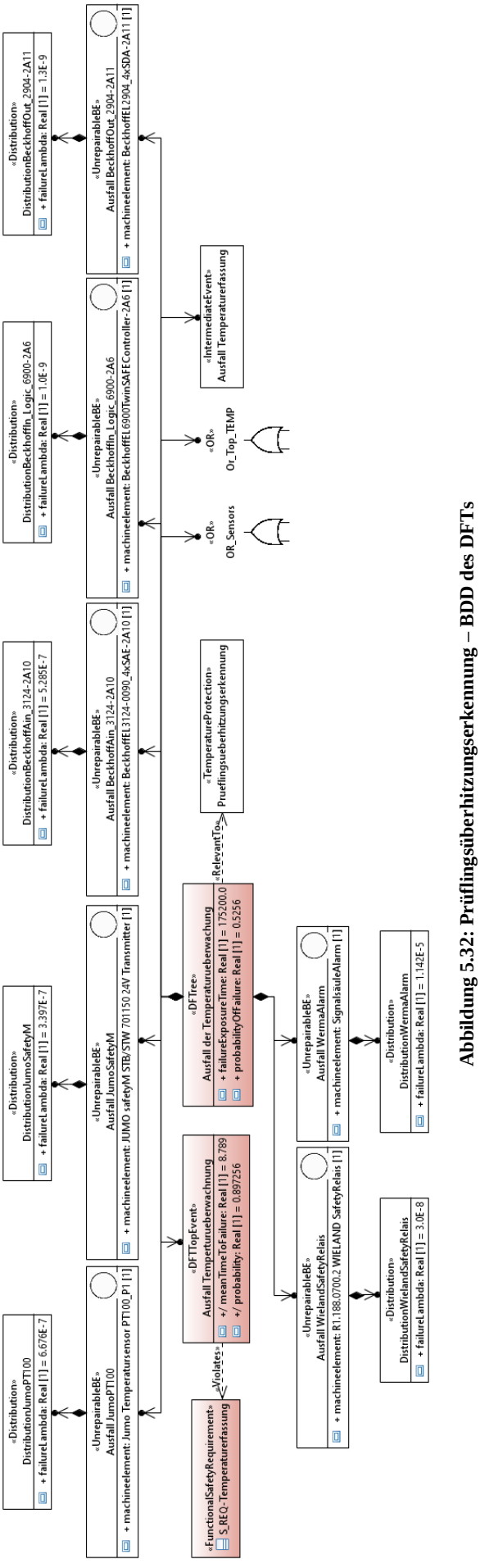


Abbildung 5.32: Prüflingsüberhitzungserkennung – BDD des DFTs

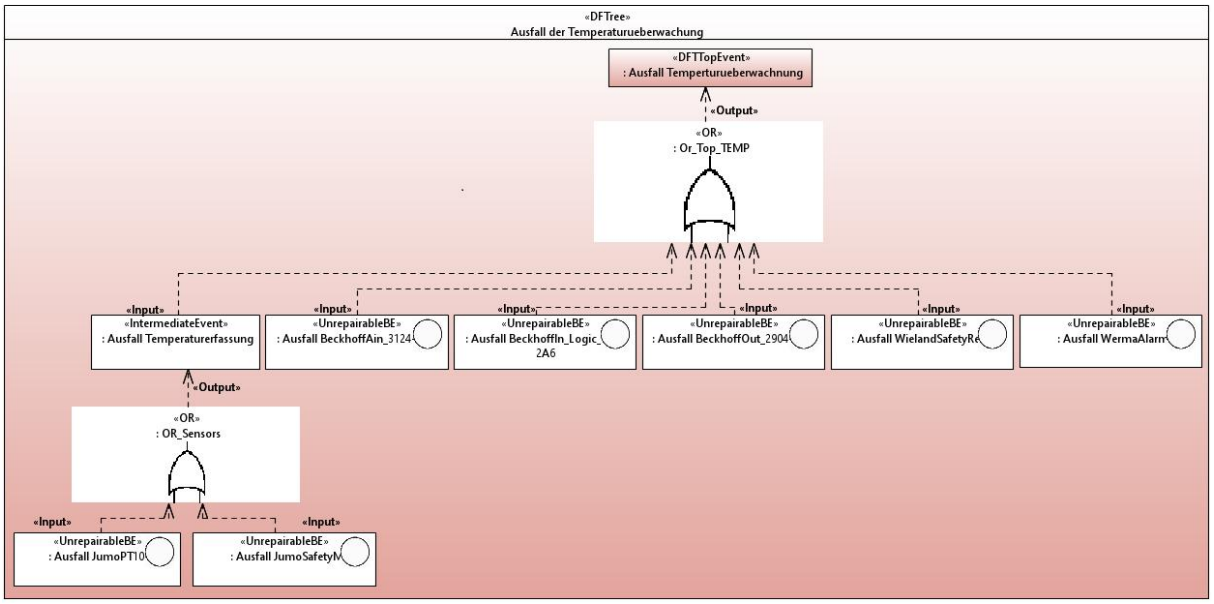


Abbildung 5.33: Prüflingsüberhitzungserkennung – IBD des nicht erfüllten DFTs

Es wird nach Diskussion unter den Stakeholdern angenommen, dass eine Nachbesserung der Sicherheitsfunktion durch einen verbesserten akustischen Alarm veranlasst wird, der als neue Hauptkomponente für den Alarm dienen soll, um auf Basis der zuvor höchsten Ausfallrate die Gesamtausfallsicherheit zu erhöhen. Die modifizierten Modelle im Vergleich zu Abbildung 5.26 bis Abbildung 5.32 finden sich in Abbildung A.58 bis Abbildung A.62. Über ein zusätzliches PAND mit dem existierenden Alarm entsteht ein neues `IntermediateEvent` für den Fehlerbaum in Abbildung A.63. Über die Herstellerangabe für MTTF von 576 Jahre und einem nicht angegebenen RDF, also 50 %, berechnet sich λ_D 9,909*10^{-8}. Somit ergibt sich für den neuen Fehlerbaum in Abbildung 5.34 nach 20 Jahren eine Ausfallwahrscheinlichkeit von ~0,2441. Der $MTTF_D$-Wert für den Fehlerbaum der Sicherheitsfunktion beträgt nun 72,3 Jahre.

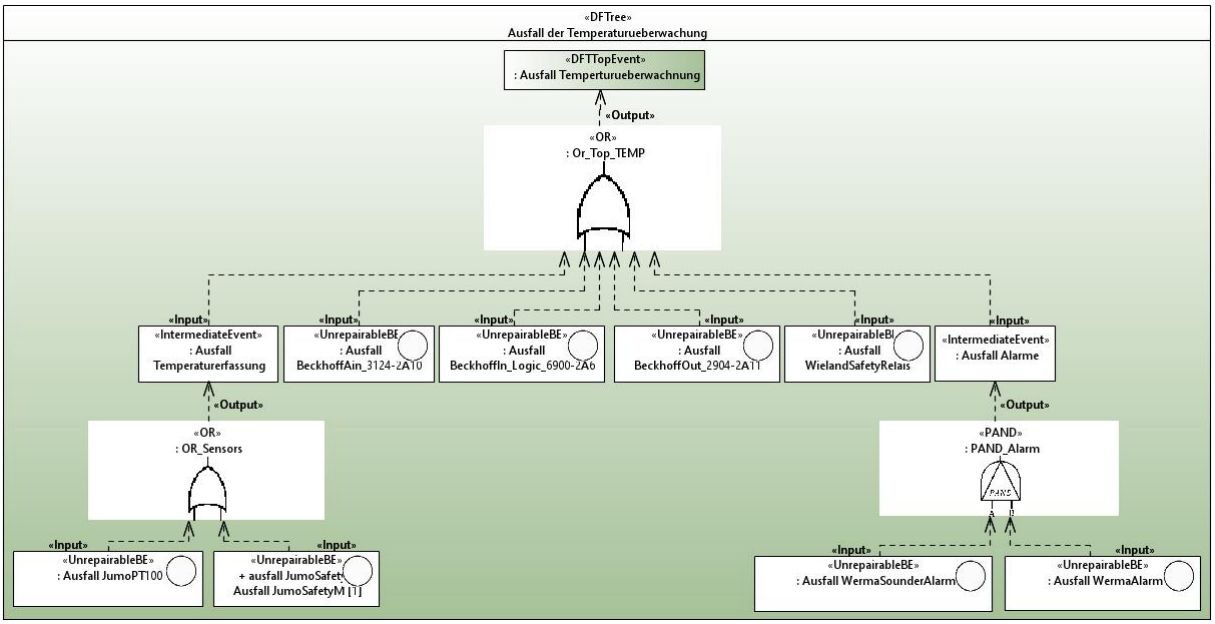


Abbildung 5.34: Prüflingsüberhitzungserkennung – IBD des erfüllten DFTs

Somit kann UC5.13 zum Nachweis der geforderten Ausfallsicherheit über die Modellierung von DFTs als erfüllt angesehen werden, wobei DFTs die Kategorie, DC und CCF nicht betrachten. Diese werden von der Analyse nicht benutzt, da es sich um andere Berechnungsmodelle handelt. Hierfür behandeln DFTs komplexere Gate-Typen, die eine strukturell genauere Analyse ermöglichen. Verschiedene Abhängigkeiten könnten zwar teilweise aufwendig im Fehlerbaum berücksichtigt werden, doch schadet dies vermeintlich der Übersichtlichkeit der Analyse und erhöht Fehlerpotentiale bei der Abbildung. Eine Bewertung hierzu kann Teil weiterer Arbeiten sein. Eine strukturell ähnliche Aufteilung der Komponenten soll im Folgenden dazu dienen, einen besseren Vergleich zu SBBM unter ähnlichen Grundannahmen zu schaffen.

5.2.3 Evaluation des Einsatzes von SBBM

Zur Bewertung der Modellierung von SBBM wird im Folgenden das sicherheitsbezogene Blockdiagramm für die Prüflingsüberhitzungserkennung entwickelt, berechnet und evaluiert. Hierfür ist wie in Abschnitt 5.2.2 die Ausfallwahrscheinlichkeit einzelner Komponenten zu bestimmen. Zusätzlich sind für alle Komponenten, deren Herstellerangaben kein gekapseltes Subsystem mit Angabe von PFH_D und Kategorie vorgeben, Subsysteme, DC und Maßnahmen gegen CCF, im Folgenden MgCCF genannt, zu bestimmen. Tabelle 5.2 erweitert hierbei die Angaben aus Tabelle 5.1. Die JUMO-Komponenten bilden ein einkanaliges Subsystem aus bewährten Bauteilen und somit Kategorie 1. Da in Kategorie 1 DC und MgCCF keine Relevanz haben, kann der vom Hersteller angegebene DC-Wert von 90 % und MgCCF-Wert von 80 Punkten bei Verwendung beider Komponenten in einer Zweileiterschaltung und 1oo2-Architektur außen vor gelassen werden. Über das analoge Eingangsmodul EL3124-0090 und der Logik EL6900 kann eine Plausibilitätsprüfung durchgeführt werden, die nach Herstellerangabe in einem DC von 90 % resultiert, doch ebenso bei Einsatz in Kategorie 1 nicht relevant ist. Bei den Beckhoff-Komponenten EL6900 und EL2904 sowie beim Wieland Safety-Relais handelt es sich nach Herstellerangaben um gekapselte Subsysteme, die neben den PFH_D-Werten aus Tabelle 5.1 jeweils Kategorie 4 besitzen. Für MgCCF in Beckhoff EL3124-0090 und den Werma-Alarm sind nach Tabelle A.5 die Maßnahmen 3.1, 3.2, 4, 5, 6.1 und 6.2 erfüllt und somit entsteht über die Punktevergabe jeweils ein MgCCF-Wert von 65 Punkten. Für den Werma-Alarm lässt sich zur Abschätzung von DC eine Fehlererkennung durch den Prozess wählen. Dabei gilt nach Tabelle A.4 die Prozessdiagnoserate (Testrate) (r_t) geteilt durch die Anforderungsrate der Sicherheitsfunktion (r_d). Das Ausgangsmodul wird 1250-mal pro Sekunde getestet und mit 100 Zyklen pro Sekunde angefordert, was einem Wert von 12,5 entspricht. Nach Tabelle A.4 ergibt dies ein DC-Wert von 90 %. Da hier ebenso Kategorie 1 umgesetzt wird, spielen diese Werte allerdings zunächst keine Rolle für die Ausführung.

Komponente	Weitere Herstellerangaben	Weitere Ausfallkriterien
JUMO PT100 JUMO safetyM STB/STW 701150	DC = 90 % MgCCF = 80 Punkte	Kategorie 1
BeckhoffEL3124-0090_4xSAE	-	Kategorie 1 DC = 90 % MgCCF = 65 Punkte
BeckhoffEL6900TwinSAFE	Kategorie 4	-
BeckhoffEL2904_4xSDA	Kategorie 4	-
Wieland R1.188.0700.2 S-Relais	Kategorie 4	-
Signalsäule Werma Alarm	-	Kategorie 1 DC = 90 % MgCCF = 65 Punkte

Tabelle 5.2: Erweiterung der Ausfallkriterien der Prüflingsüberhitzungserkennung

Auf dieser Grundlage lassen sich, wie in Abbildung 5.35 dargestellt, die Bestandteile des sicherheitsbezogenen Blockdiagramms auf Basis der Metamodelle aus Abschnitt 3.3.3 und der elektrotechnischen Sicht aus Abbildung 5.29, erstellen. Jede Komponente wird durch die Stereotypen der SBBM erweitert, um die entsprechenden Werte zuzuordnen. Dadurch können die entsprechenden Kenngrößen ebenso in den Modellen der anderen Stakeholder referenziert werden, was die gemeinsame Entwicklung stärkt. Für jedes Element des Blockdiagramms wird eine Betriebszeit von 20 Jahren angenommen, bei der PFH_D für PL_r c den Wert $3*10^{-6}$ nicht überschreiten darf. Für die Verbindung und Anzeige des Blockdiagramms wird ein IBD erstellt, das in Abbildung 5.36 oben dargestellt ist.

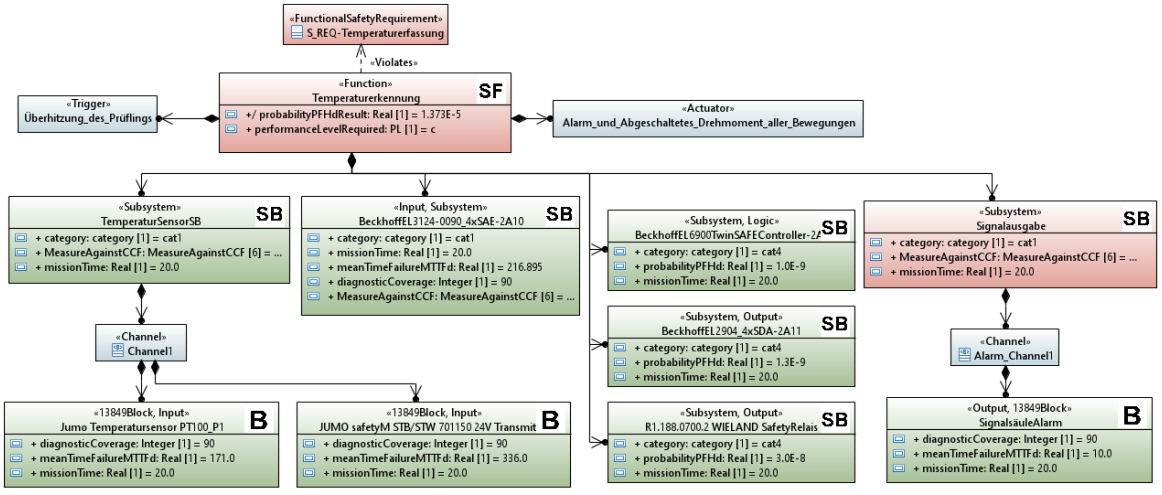


Abbildung 5.35: Prüflingsüberhitzungserkennung – BDD der SBBM

Abbildung 5.35 und Abbildung 5.36 oben zeigen bereits die Ergebnisse des Blockdiagramms nach der Durchführung der Analyse durch SISTEMA unter Nutzung des in Abschnitt 4.2.3 entwickelten Transformators zum Nachweis der Risikominderung nach ISO13849. Hierbei sind die Ergebnisse zunächst ebenfalls negativ ausgefallen, da der PFH_D bei $1{,}373*10^{-5}$ und somit über dem Limit für PL c liegt. Dies entspricht einem $MTTF_D$-Wert von nur 9 statt 20 Jahren. In

der Visualisierung wurden die verletzte Anforderung, die Sicherheitsfunktion, die Subsysteme und die Blöcke durch den Transformator und die CSS-Datei rot eingefärbt. Zur verbesserten Darstellung wurde die Einfärbung der Subsysteme und Blöcke im BDD deaktiviert. Somit ist es auch hier möglich, nicht erfüllte Anforderungen in anderen Diagrammen, wie Abbildung 5.21 nachzuvollziehen und entsprechende Änderungen in den Modellen zu planen.

Wie bereits dargelegt, wurde unter den Stakeholdern eine Nachbesserung der Sicherheitsfunktion durch einen verbesserten akustischen Alarm vereinbart. Zusätzlich erfüllt das modellierte Subsystem `Signalausgabe` nicht die in Tabelle A.6 aufgeführten Voraussetzungen für Kategorie 1. Für die modifizierten Modelle in Abbildung A.58 bis Abbildung A.62 mit dem zweiten akustischen Alarm Werma-Midi-Sounder wurde das Subsystem `Signalausgabe` in Abbildung A.64 modifiziert. `Signalausgabe` beinhaltet nun einen zweiten Kanal, mit dem das Subsystem der Struktur für Kategorie 3 entspricht. Mit dem aus Tabelle 5.1 bekannten $MTTF_D$ von 1152 Jahren und den bereits für Signalsäule Werma-Alarm aus Tabelle 5.2 bestimmten DC- und MgCCF-Werten lässt sich das neue sicherheitsbezogene Blockdiagramm berechnen. Es ergibt sich für die modifizierte Sicherheitsfunktion (Abbildung 5.36 unten) ein PFH_D von $2{,}389*10^{-6}$ und somit ein $MTTF_D$-Wert von 47,784 Jahren.

Somit kann nachgewiesen werden, dass UC5.13 über die Modellierung von SBBM erfüllt werden kann. Für SBBM lassen sich allerdings auch Erweiterungen diskutieren. So unterstützt SISTEMA Bibliotheken, in denen Hersteller ihre Produkte abbilden können, um diese dann für die SBBM direkt einzufügen. Hierbei ist es möglich, dies ebenso in SysML umzusetzen und so die Anwendbarkeit der Modelle weiter zu vereinfachen. Um die Evaluation abzuschließen, findet sich ein Vergleich von DFT und SBBM im folgenden Abschnitt.

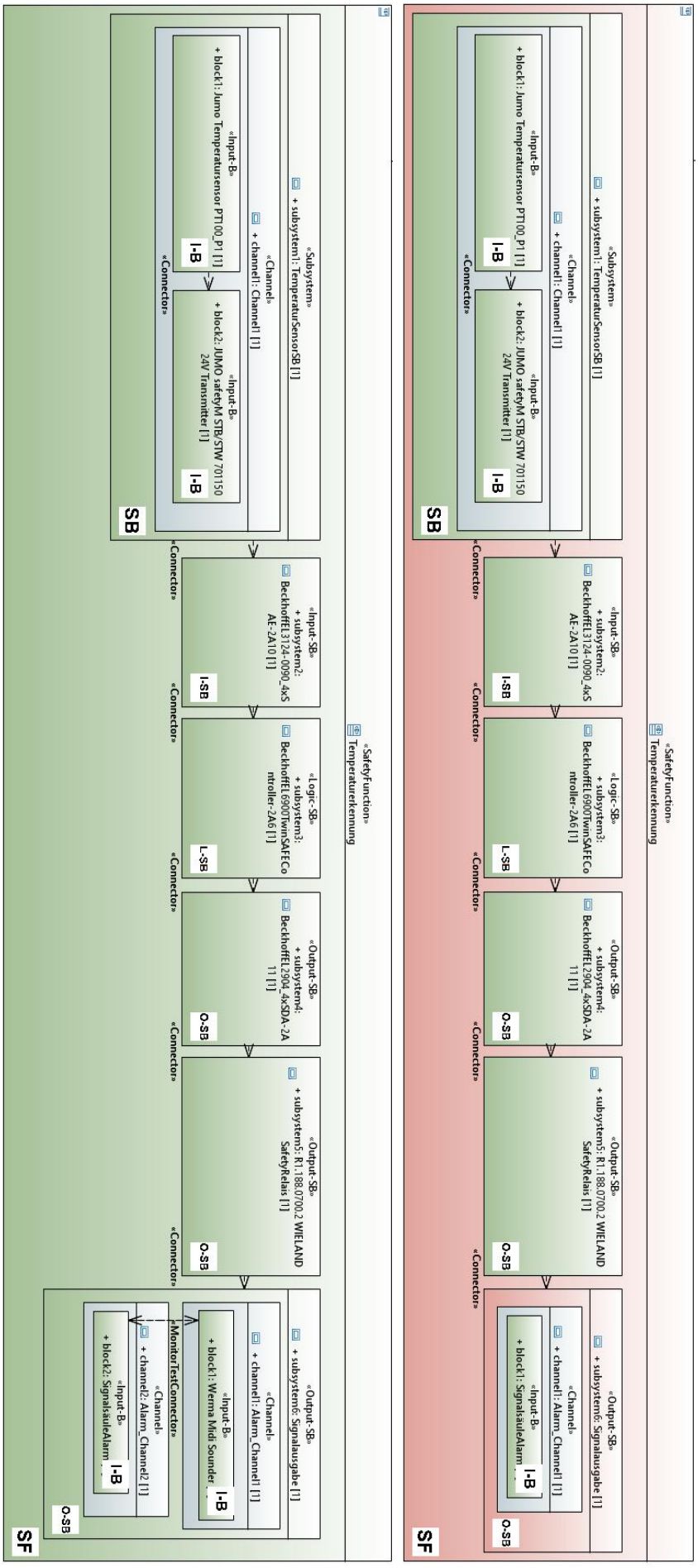


Abbildung 5.36: Prüflingsüberhitzungserkennung – IBD des Blockdiagramms
Oben: nicht erfüllte Anforderungen, Unten: erfüllte Anforderungen

5.2.4 Vergleich DFTs und SBBM

Für UC5.13 können die Ergebnisse der beiden Analysemethoden DFT und SBBM aus Abschnitt 5.2.2 und 5.2.3 miteinander verglichen werden. Hierbei fließen in Tabelle 5.3 ebenso die Ergebnisse aus den anderen Sicherheitsfunktionen mit ein, die im Anhang A.11.3, A.11.4 und A.11.5 näher beschrieben sind. Die Ergebnisse sind bei erster Betrachtung relativ ähnlich, aufgrund der Untersuchung der gleichen Funktionen mit unterschiedlichen Methoden. Der kleine Unterschied zwischen den PFH_D-Werten bei der Prüflingsüberhitzungserkennung im unerfüllten Fall entsteht hierbei beispielsweise durch die Begrenzung des PFH_D des Subsystems für Beckhoff EL3124-0090 auf $1{,}142*10^{-6}$ aufgrund des Einsatzes in Kategorie 1 der SBBM. Hierbei kann im Generellen festgestellt werden, dass bei einfachen Strukturen die verschiedenen Analysen zu ähnlichen Einschätzungen der Ausfallwahrscheinlichkeit kommen und die unterschiedlichen Berechnungsmethoden geringen Einfluss auf die Größendimensionen von PFH_D haben. Bei erfüllten Anforderungen im Fall aller vier Sicherheitsfunktionen hingegen sind die Ergebnisse etwas unterschiedlicher, da SBBM die Berechnungen nach Formel A.1 und Vorgaben aus Tabelle A.6 mit einbezieht sowie DFTs komplexere Verknüpfungsstrukturen besitzen. Bei allen Sicherheitsfunktionen entsteht dabei der Effekt, dass durch die vereinfachte Berücksichtigung der Kategorie, des DC-Wertes und des CCF-Wertes die PFH_D-Werte einzelner Subsysteme verringert oder begrenzt werden. Es handelt sich somit um ein einfach anwendbares, aber zur Auslegung von Sicherheitsfunktionen ungenaueres Verfahren. Zusätzlich beziehen die DFTs der erfüllten Prüflingsüberhitzungserkennung und aller anderer Sicherheitsfunktionen PANDs zur Priorisierung von Sensoren und Aktoren mit ein, was die PFH_D-Werte verringert. Je nachdem, wie groß der Einfluss einzelner Subsysteme ist, desto geringer fallen die positiven und negativen Abweichungen zwischen DFTs und SBBM aus. Besonders lassen sich diese Auswirkungen bei den Ergebnissen aus dem DFT des Nothalts feststellen, da hier sogar ein PAND über alle drei Kontakte des Nothalt-Tasters modelliert wird, im Gegensatz zu SBBM, wo nur eine einfache Redundanz modelliert werden kann. Dadurch entsteht ein deutlich geringerer PFH_D-Werte im Fehlerbaum.

Somit handelt es sich bei SBBM um einen einfacher anwendbaren, aber ungenaueren, Ansatz zum Nachweis der Risikominderung in der Maschinenindustrie für Sicherheitsfunktionen, da über Kategorie, DC und CCF zusätzliche Kriterien mit einfließen, wohingegen DFTs auch für andere Zwecke der Risikobeurteilung Einsatz finden können, wie die Bewertung des gesamten Designs zur Berechnung des PF_D-Wertes. Zusätzlich sind über DFTs drei und mehrkanalige Redundanzen möglich, wie die Nothalt-Funktion unter Anhang A.11.3 demonstriert, wohingegen SBBM ohne weiteres Zutun auf zwei Kanäle pro Subsystem begrenzt ist.

Sicherheitsfunktion	DFT Ergebnis	SBBM Ergebnis
Prüflingsüberhitzungserkennung (oben unerfüllt, unten erfüllt)	PF_D=0,897; $MTTF_D$=8,789 ($\underline{PFH_D=1,299*10^{-5}}$)	$\underline{PFH_D=1,373*10^{-5}}$
	PF_D=0,244; $MTTF_D$=72,313 ($\underline{PFH_D=1,579*10^{-6}}$)	$\underline{PFH_D=2,389*10^{-6}}$
Trennende Schutzeinrichtung	PF_D=0,129; $MTTF_D$=132,22 ($\underline{PFH_D=8,634*10^{-7}}$)	$\underline{PFH_D=7,675*10^{-7}}$
Nothalt	$PF_D=6,041*10^{-3}$; $MTTF_D$=2871,375 ($\underline{PFH_D=3,976*10^{-8}}$)	$\underline{PFH_D=7,736*10^{-8}}$
Wartungslichtgitter	$PF_D=6,82*10^{-3}$; $MTTF_D$=1994,848 ($\underline{PFH_D=5,723*10^{-8}}$)	$\underline{PFH_D=3,868*10^{-8}}$

Tabelle 5.3: Ergebnisse DFT und SBBM

Im konkreten Fallbeispiel hat sich gezeigt, dass im Gegensatz zum RAAML-Ansatz durch die lose Kopplung der Analysewerkzeuge Beziehungen zu Parametern und die Art der Berechnungen nicht vollständig nachverfolgt werden können, da Berechnungen innerhalb der Analyse stattfinden. Die Implementierung der Transformatoren kann dabei potenziell Fehler einführen, die RAAML vermeidet. Durch den Einsatz etablierter Werkzeuge entsteht der Vorteil, dass keine Validierung der Werkzeuge notwendig wird, allerdings kann es notwendig werden, die Transformatoren zu validieren. In der Fallstudie hat sich jedoch gezeigt, dass durch die Rückkopplung der Ergebnisse die Möglichkeit besteht, die Transformatoren über einen Vergleich der Datensätze in LabVIEW auf die korrekte Datenübertragung zu überprüfen.

5.2.5 Evaluation des Einsatzes von Risikographen

Zur Bewertung der Analysewerkzeugumsetzung für Risikographen und des Architekturkonzepts zur losen Toolkopplung soll im Folgenden die lose und enge Toolkopplung anhand des Risikographen verglichen werden. Hierfür wurde das Konzept der engen Toolkopplung für Risikographen mittels eines Constraint implementiert. Zur Berechnung der Risikoprioritätszahl (RPZ) und des PL_r dient das OCL-Constraint in Listing 5.1 für das Bibliothekselement des Risikoitems in Bezug zur Modellierung aus Abbildung 3.18 und analog zur Berechnung über die lose Toolkopplung in LabVIEW aus Abbildung 4.25. Als Referenz für ein solches Vorgehen wurde die OCL-Standard-Spezifikation [297] verwendet, welche die verschiedenen Arten zur Nutzung von OCL-Ausdrücken in UML-Modellen beschreibt und auch bereits als Referenzbeispiel für die Einschränkungen von MMBSEFE diente. Aus den bereits in Abschnitt 4.2.4 in Formel 4.1, Formel 4.2 und Tabelle 4.1 beschriebenen Berechnungsgrundlagen wird das Attribut `riskIndex` der Klasse `MachineryRiskItem` definiert und berechnet, indem die höchsten Werte für die Risikodimensionen nach Formel 4.2 bestimmt und für die Risikoprioritätszahl nach Formel 4.1

berechnet werden. Die resultierende RPZ wird dann, so wie bereits in Abbildung 4.25, verwendet, um den Wert des Attributs `pLr` bei der ersten Ausführung des Constraint zu setzen.

```
1  context MachineryRiskItem
2  inv RiskIndexCalculation:
3  --Bestimme die höchsten Werte für die verschiedenen Risikodimensionen
4  let avoidanceValues : OrderedSet(String) = self.hazardousSituation-
   >collect(avoidance)->asOrderedSet()
5  let maxAvoidance : Real = if avoidanceValues->indexOf(avoidanceValues-
   >last()) = 'P1' then 1.0 else 2.0 endif
6  let frequencyValues : OrderedSet(String) = self.hazardousSituation-
   >closure(hs | hs.hazardousEvent)->collect(frequencyOfExposure)-
   >asOrderedSet()
7  let maxFrequency : Real = if frequencyValues->indexOf(frequencyValues-
   >last()) = 'F1' then 1.0 else 2.0 endif
8  let occurenceValues : OrderedSet(String) = self.hazardousSituation-
   >closure(hs | hs.hazardousEvent)->closure(mc | mc. machineContextCause)-
   >collect(occurence)->asOrderedSet()
9  let maxOccurence : Real = if occurenceValues->indexOf(occurenceValues-
   >last()) = 'O1' then 1.0 else 2.0 endif
10 let severityValues : OrderedSet(String) = self.machineryEffect-
   >collect(severity)->asOrderedSet()
11 let maxSeverity : Real = if severityValues->indexOf(severityValues-
   >last()) = 'S1' then 1.0 else if severityValues->indexOf(severityValues-
   >last()) = 'S2' then 1.5 else 0.0 endif
12
13 -- Berechne den neuen Risikoindex (RPZ)
14 let newRiskIndex : Real = maxSeverity * ( maxFrequency + maxOccurence +
   maxAvoidance )
15
16 in
17   --Wenn der Risikoindex noch nicht gesetzt wurde, setze ihn und
     berechne pLr
18   if self.riskIndex = null then
19     self.riskIndex = newRiskIndex
20     if newRiskIndex < 4.5 then
21       self.pLr = 'a'
22     else if newRiskIndex >= 4.5 and newRiskIndex < 6 then
23       self.pLr = 'b'
24     else if newRiskIndex >= 6 and newRiskIndex < 7.5 then
25       self.pLr = 'c'
26     else if newRiskIndex >= 7.5 and newRiskIndex < 9 then
27       self.pLr = 'd'
28     else
29       self.pLr = 'e'
30     endif
31   --Wenn der Risikoindex bereits gesetzt wurde, setze den vorherigen
     Risikoindex und den neuen Risikoindex und füge den vorherigen
     Risikoindex zur Liste der vorherigen Risikoindizes hinzu
32   else
33     let lastRiskIndex : Real = self.riskIndex
34     in
35       self.riskIndex = newRiskIndex
36       self.previousRiskIndex->append(lastRiskIndex)
37   endif
```

Listing 5.1: OCL-Constraint für Risikoitem

Die Implementierung dieses Constraints besteht aus einer Reihe von OCL-Ausdrücken, die zusammenarbeiten, um die Berechnung der RPZ durchzuführen und den Wert des Attributs `pLr` zu setzen. Zunächst werden die höchsten Werte für die verschiedenen Risikodimensionen mithilfe von `let`-Ausdrücken und `OrderedSet()`-Funktionen berechnet (Zeile 1 bis 11). Anschließend wird die neue RPZ als Summe und Produkt der höchsten Werte entsprechend Formel 4.1 berechnet (Zeile 14). Wenn der Wert des Attributs `riskIndex`, der die RPZ repräsentiert, noch nicht gesetzt wurde, wird er gesetzt und der Wert des Attributs `pLr` basierend auf dem Risikoindex ausgegeben (Zeile 18 bis 30). Wenn `riskIndex` bereits gesetzt

wurde, wird die vorherige RPZ in `previousRiskIndex` gespeichert und die neue RPZ berechnet (Zeile 33 bis 36).

Das in Listing 5.1 dargestellte OCL-Constraint `RiskIndexCalculation` ist ein Beispiel dafür, wie OCL verwendet werden kann, um die komplexe Semantik von Klassenattributen in UML-Modellen auszulesen und Vorschriften zu definieren und zu implementieren. Es demonstriert dabei die enge Toolkopplung, ähnlich der Vorgehensweise von RAAML.

Beim Vergleich der losen und engen Toolkopplung stellt sich heraus, dass durch die einfachen Berechnungsgrundlagen der Analysemethode in der losen Kopplung ein vermeidbarer erhöhter Mehraufwand generiert wird. Hierbei besteht das OCL-Constraint lediglich aus 37 Zeilen, wobei die Umsetzung in LabVIEW über 4000 Codezeilen mit einer Analyseausführung von 75 Zeilen besitzt. Einer der Hauptaufwände entsteht beim Export und erneuten Import der Risikoitems, der sich für die einfache Funktion zur Berechnung vermeintlich kaum lohnt.

Die Risikographmethode stellt somit durch das bereits modellierte Architekturkonzept in den Profilen und Bibliotheken und der einfachen Ergebnisberechnungen ein gutes Beispiel für eine einfache enge Toolkopplung, ähnlich des RAAML-Ansatzes für FMEA oder FTA, dar. Das entwickelte OCL-Constraint ist einfach zu warten und umgeht mögliche Fehler bei der Transformation im Vergleich zur losen Kopplung. So stehen zwar alle Informationen in den UML-Modellen bereit, doch verlangen die Sicherheitsnormen nach validierten und betriebsbewährten Werkzeugen zur Analyse. Dies ist bei so einfachen Berechnungsgrundlagen relativ einfach zu validieren und der Grund für den RAAML-Ansatz, doch mit steigender Komplexität, wie bei DFT oder SBBM, steigt auch die Fehleranfälligkeit und somit der Test- und Validierungsaufwand des Analysewerkzeugs. Gegen die Implementierung zur Analyseausführung von Risikographen in LabVIEW spricht auch, dass es sich um kein validiertes betriebsbewährtes Werkzeug handelt, was im Vergleich zur engen Kopplung einen höherer Test- und Validierungsaufwand bedeutet. Allerdings lassen sich hier schneller zusätzliche Funktionen implementieren. In Abschnitt 4.2.4 konnten beispielsweise alte RPZ einfacher berechnet werden, es ermöglicht aber auch die einfache Anbindung eines Werkzeugs zur Dokumentation. Ein weiterer Vorteil der engen Kopplung ergibt sich durch die innerhalb der Modelle nachvollziehbaren Berechnungen und der Verfolgbarkeit der Algorithmen.

Ähnlich des Ansatzes der OMG bietet sich daher eine enge Toolkopplung für Analysemethoden mit einfachen Berechnungsfunktionen eher an, wobei es eine Rolle spielt, ob die Analysedaten nach der Analyse ebenso extern weiterverwendet oder zusätzliche Daten generiert werden sollen. Schlussfolgernd bietet sich das Architekturkonzept dieser Arbeit zur losen Kopplung eher

für komplexere Berechnungsfunktionen und die Anbindung externer etablierter Werkzeuge an. Ein weiterer zu betrachtender Aspekt ist das vorhandene Knowhow. So kann die lose Kopplung für Risikographen über ein LabVIEW-basierten Transformator in LabVIEW das LabVIEW-Knowhow eines Prüfmaschinenherstellers besser nutzen und erhöht so die Eigenständigkeit zur Anwendbarkeit.

5.2.6 Diskussion

Die Bewertung der Modellierungslösung für die Analysemethoden zur Risikobeurteilung und beinhaltenden Bewertung der Risikominderung in der Maschinenindustrie ist Teil jedes Unterabschnitts in Abschnitt 5.2. Abschnitt 5.2.1 erweitert die Evaluation der Sichtweisen der Stakeholder Maschinenbauer, Elektroingenieur und Informatiker für die Entwicklung von softwarebasierten Sicherheitsfunktionen und zeigt, dass eine Umsetzung in MMBSEFE als Grundlage für die Sicherheitsbetrachtungen und die Weiterentwicklung in Stakeholder-spezifischen Modellen möglich ist. In Abschnitt 5.2.2 und 5.2.3 wurden die Analysemethoden DFT und SBBM anhand einer beispielhaften softwarebasierten Sicherheitsfunktion durchgeführt. Hiermit zeigt sich, dass die Analysen in einen MBSE-Prozess eingebunden werden können, was die verbleibenden UC5.11 bis UC5.13 des Sicherheitsingenieurs einschließt und die Gesamtmodellierung für die Batterieprüfmaschine verbessert. Je nachdem, wie die Analysen angewendet werden, kann die Integration jedoch in weiteren Schritten verbessert werden (vgl. S. 195 und S. 197). Dies kann Teil zukünftiger Arbeiten sein. Die Ergebnisse der Analysen wurden in Abschnitt 5.2.4 verglichen und zeigen SBBM als die zu präferierende Analysemethode für den Nachweis der Risikominderung in der Maschinenindustrie. Aufgrund einer leicht einzusetzenden kritischeren Analyse, durch die die verschiedenen Berechnungsgrundlagen vereinfacht einbezogen werden können, entsteht eine überschätzte Ausfallwahrscheinlichkeit und damit eine höhere Sicherheit in der Abschätzung. Die Evaluation zur Risikographmethode wurde teilweise bereits in Abschnitt 5.1.4 dargestellt. Zusätzlich wurde im Rahmen von Abschnitt 5.2.5 für die Risikographmethode ein Vergleich zwischen loser und enger Toolkopplung angestellt, der darstellt, dass sich für Analysemethoden mit einfachen Berechnungsfunktionen eine enge Toolkopplung eignet, sofern diese durch das Knowhow der Stakeholder umsetzbar ist.

Die Bewertung der Forschungsaufgabe zeigt damit, dass die Integration und Anwendung von maschinenbauspezifischen Analysemethoden für die ausgewählte Implementierung umsetzbar ist, jedoch ebenso, dass noch Erweiterungsmöglichkeiten bestehen.

5.3 MBSE-Prozesse für KMUs

Für die Evaluation der entwickelten Konzepte und zum Nachweis der Anwendbarkeit in KMUs wird in diesem Abschnitt der Prozess zur qualitativen Evaluation aus Abschnitt 2.3.2 angewendet. Um FA3-E zu beantworten, werden hierfür Interviews zur Eignung von MMBSEFE zusammen mit einem in der Domäne der Maschinenindustrie etablierten KMU wie ProNES durchgeführt. Im Speziellen sind unter Vorgabe des in Abschnitt 2.3.2 identifizierten Prozesses einer qualitativen Evaluation durch Interviews Fragenkataloge zur Erwartungshaltung vor und den Grad der Zufriedenheit mit den Modellen der Fallstudie nach der Implementierung zu entwickeln, die Teilnehmer auszuwählen und die Interviews durchzuführen. Auf Basis von Transkriptionen sind die Antworten zu analysen und zu interpretieren. Für eine übersichtliche Nachvollziehbarkeit der Befolgung dieser Prozessschritte besteht die Gliederung der folgenden Unterabschnitte aus diesen Schritten, wobei die Schritte 2, 4 und 5 doppelt zu durchlaufen sind.

5.3.1 Schritt 1 – Evaluationsgegenstand und –ziele

Ziel ist es, herauszufinden, ob die Umsetzung von MMBSEFE für KMUs der Maschinenindustrie einen praktikablen Prototyp bietet, der als Ausgangspunkt für eine Lösung dienen oder weiterentwickelt werden kann. Entsprechend dem Evaluationsprozess werden zunächst der Evaluationsgegenstand und das Evaluationsziel präzisiert.

Das Evaluationsziel besteht darin, qualitative Daten zu den Erfahrungen und Meinungen der Stakeholder vor der Entwicklung der Fallstudie zu erfassen und diese mit Aspekten wie allgemeiner Zufriedenheit, wahrgenommener Effektivität und Effizienz sowie spezifischen Vor- und Nachteilen des MBSE-Frameworks nach deren Entwicklung zu vergleichen. Zusätzlich können Metriken wie der Implementierungsaufwand von Transformatoren und die Genauigkeit von Analyseergebnissen quantitativ erfasst werden.

5.3.2 Schritt 2a – Fragenkatalog zur Erwartungshaltung

Um den aktuellen Stand in der Entwicklung innerhalb von ProNES festzustellen, wurden für die Interviews zur Erwartungshaltung folgende Fragestellungen entwickelt:

1. Können Sie mir beschreiben, wie Sie normalerweise Ihre konstruktiven Modelle, Schaltpläne, Software und Risikobeurteilung erstellen/entwickeln?
2. Wie sieht Ihre Zusammenarbeit mit anderen Teammitgliedern aus?
3. Wie kommunizieren Sie während des Entwicklungsprozesses mit anderen Teammitgliedern? Wie sieht die Motivation aus, sich in ein Framework/Modell einzuarbeiten?
4. Haben Sie Erfahrungen mit MBSE-Frameworks?

5. Was würden Sie sich für ein solches Framework wünschen/ Was wäre ihrer Meinung nach notwendig, um ein solches Framework erfolgreich einzuführen?
6. Sehen Sie spezielle Vorteile oder Nachteile für die Einführung einer solchen Innovation in der Firma? Wie würde es Ihnen bei der Entwicklung helfen? Gibt es evtl. spezielle Informationen, die Sie von anderen Stakeholdern erhalten möchten, um Ihre Arbeit zu verbessern?

Zur Validierung des Fragenkatalogs wurde dieser der Firmenleitung vorgelegt und gemeinsam besprochen, um dessen Eignung und Korrelation mit Firmenleitlinien festzustellen. Es traten keine Bedenken auf.

Zur Vollständigkeit findet sich der ausführliche Fragenkatalog zur Erwartungshaltung der Stakeholder im Anhang A.12.1.

5.3.3 Schritt 3 – Teilnehmer

Der Fragenkatalog soll jeweils einem Stakeholder der verschiedenen Sichtweisen präsentiert und mit ihm diskutiert werden. Unter den Interviewten befanden sich ein erfahrener Maschinenbauer, der ebenso die anfängliche Rolle des Sicherheitsingenieurs vertreten hat, ein Elektroingenieur mit langjähriger Erfahrung in der Schaltplanentwicklung, ein Informatiker, der hauptverantwortlich für die SPS-Softwareentwicklung bei ProNES ist, und ein erfahrener Sicherheitsingenieur, der im Laufe des Projektes eingestellt wurde, um komplexe Sicherheitsbetrachtungen bei ProNES intern umfassender durchführen zu können.

5.3.4 Schritt 4a – Interviews zur Erwartungshaltung

Die Interviews zur Erwartungshaltung wurden in freien Gesprächen geführt, während dessen die vordefinierten Fragenkataloge (vgl. Anhang A.12.1) durchgearbeitet wurden. Die Interviews zu Beginn des Projektes mit dem Maschinenbauer, dem Informatiker und dem damaligen Sicherheitsingenieur wurden zur besseren Auswertung per Audiorekorder aufgezeichnet. Aufgrund von zeitlichen Ressourcenproblemen des Elektroingenieurs konnte das Interview zur Erwartungshaltung lediglich schriftlich durchgeführt werden. Aufgrund des Wechsels des anfänglichen Sicherheitsingenieurs durch einen erfahreneren Sicherheitsingenieur während der Projektlaufzeit wurden die anfänglichen Fragestellungen mit dem neuen Sicherheitsingenieur erneut diskutiert, schriftlich dokumentiert und in den Antworten des ursprünglichen Sicherheitsingenieurs ergänzt. Ausufernde Antworten, die nicht direkt mit dem Projekt in Verbindung standen, sowie firmeninterne vertrauliche Informationen, Informationen aus vergangenen Pro-

jekten oder persönliche Daten wurden aus der Dokumentation entfernt (Anonymisierung qualitativer Daten nach Meyermann et al. [298]). Alle relevanten und projektbezogenen Aussagen wurden hingegen wortwörtlich dokumentiert. Entsprechende zentrale Aussagen werden im Folgenden aus den Protokollen zitiert oder zusammengefasst.

Maschinenbauer

Der Maschinenbauer, der hauptsächlich mit AutoCAD arbeitet, hat eine klare Meinung zu einem möglichen MBSE-Framework und dessen Integration in seine Arbeitsprozesse. Er entwickelt eigenständig Konstruktionsmodelle und sieht die Bedeutung direkter Kommunikation mit Teamkollegen wie Elektroingenieuren, Softwareentwicklern und Sicherheitsingenieuren als essenziell an. Diese Kommunikation erfolgt effizient durch mündlichen Austausch, Teams-Chat oder E-Mail. Ein MBSE-Framework empfindet er als potenziell zeitaufwendig und mit wenig Mehrwert verbunden, da es Mehrarbeit bedeuten könnte. Trotzdem erkennt er den Vorteil einer gemeinsamen Kommunikationsbasis, die auch für externe und nicht-technische Stakeholder zugänglich ist. Diese könnte die häufigen Missverständnisse mit Kunden und die Probleme bei der Beschaffung von Komponenten verringern, die aktuell durch die dokumentenzentrierte Vorgehensweise entstehen. Der Maschinenbauer steht also einer Integration eines solchen Frameworks skeptisch gegenüber, sieht aber den potenziellen Nutzen in verbesserten Kommunikationswegen.

Elektroingenieur

Der Elektroingenieur, der hauptsächlich EPLAN zur Entwicklung seiner Schaltpläne verwendet, sieht die Einführung eines MBSE-Frameworks als überwiegend positiv an, trotz der Herausforderungen durch den zeitlichen Aufwand. Er betont, dass er gleichzeitig mehrere Projekte betreut und selten Zeit für Weiterbildungen hat. Für eine erfolgreiche Implementierung eines solchen Frameworks fordert er die Organisation von Workshops und ausreichende Einarbeitungszeit. Ohne diese Maßnahmen befürchtet er, dass das Framework nicht effektiv genutzt wird, was die Investition finanziell unrentabel machen könnte. Dennoch erkennt er die Vorteile eines MBSE-Frameworks an, vor allem in Bezug auf die Nachvollziehbarkeit von Änderungen in Schaltplänen, da durch die gemeinsamen Modelle eine verbesserte Übersichtlichkeit entstehen würde.

Informatiker

Der Informatiker betont die Vorteile der Nutzung eines gemeinsamen MBSE-Frameworks und der Modellierung mit SysML für die Programmierung von SPSen auf der TwinSAFE Plattform

von Beckhoff. Er sieht einen erheblichen Nutzen darin, wenn seine Software-Modelle unmittelbar bezüglich ihrer Übereinstimmung mit Systemanforderungen, Sicherheitsvorgaben sowie Risikobewertung und -minderung gekoppelt sind. Eine zentrale Datenbank oder ein Repository für Sicherheitsfunktionen könnte die Wiederverwendung von Funktionen über verschiedene Maschinen und Hersteller hinweg erleichtern, wodurch eine einheitliche Funktionalität gewährleistet wird. Zusätzlich wünscht er sich detailliertere Informationen von anderen Stakeholdern über spezifische Eingangs- und Ausgangsbelegungen der Hardware, um das Verhalten der Software entsprechend anpassen zu können. Der Informatiker betont, dass eine solche Plattform die Entwicklungseffizienz erhöhen und die Kommunikation zwischen den beteiligten Parteien verbessern würde.

Sicherheitsingenieur

Die Sicherheitsingenieure des Unternehmens stellen die Herausforderungen bei der Bewertung der funktionalen Sicherheit, insbesondere von mechanischen und elektrischen Ausrüstungen, in den Fokus, da diese bisher meist eher getrennt betrachtet werden. Sie betonen ihre Verantwortung für sowohl direkte als auch indirekte Gefahren, die von den Geräten ausgehen, was eine genaue Kenntnis der Maschinenelemente und Normen voraussetzt. Diese detaillierte Einsicht in die Entwicklung wird von ihnen als essentiell angesehen, da das Unternehmen ständig neue Maschinen für unterschiedliche Anwendungen entwickelt, was die Risikoanalyse erheblich verkompliziert. Die Sicherheitsingenieure nutzen derzeit die WEKA-Software [246] zur Risikobeurteilung und SISTEMA zur Risikominderung und sehen die Potenziale, die bisherigen dokumentenbasierten Prozesse zu verbessern. Sie gehen allerdings davon aus, dass sich die Risikobeurteilung aufgrund komplexer Normensammlungen und Bauteillisten so umfangreich darstellt, dass sie unmöglich in einem gemeinsamen Framework mit den relevanten Systemmodellen abgebildet werden kann. Eine weitere Herausforderung sehen sie in der Vielfältigkeit des Unternehmens. Neben Batteriezellenprüfmaschinen stellt das Unternehmen andere Prüfmaschinen und Sondermaschinen her, was die Risikoanalyse verkompliziert, da immer aufs Neue Situationen und Auswirkungen abgeschätzt werden müssen.

5.3.5 Schritt 5a – Transkription, Analyse und Interpretation zur Erwartungshaltung

Die projektbezogenen Antworten aller interviewten Stakeholder finden sich vollständig transkribiert im Anhang A.12.2 bis A.12.5. Hierbei variiert die Erwartungshaltung der Stakeholder hinsichtlich der Einführung eines MBSE-Frameworks in die Modellentwicklungen stark, je nach Rolle und Perspektive der Beteiligten. Der Maschinenbauer sieht es skeptisch und

betont die Relevanz direkter Kommunikation, erkennt jedoch den Vorteil einer gemeinsamen Kommunikationsbasis. Der Elektroingenieur sieht überwiegend positiv auf das Framework, fordert jedoch Workshops und Einarbeitungszeit für eine effektive Nutzung. Der Informatiker hebt die Vorteile der Modellierung mit SysML hervor und sieht erheblichen Nutzen in der Kopplung von Software-Modellen mit Systemanforderungen. Beide Sicherheitsingenieure betonen die Herausforderungen bei der Bewertung der funktionalen Sicherheit und kritisieren an der Umsetzbarkeit aufgrund umfangreicher Normen und Bauteillisten.

Die Antworten zur Erwartungshaltung wurden als grundlegendes Ziel für die Modellentwicklungen in Abschnitt 3.2 und 3.3 genutzt, sowie daraus die Vorgehensweise zur vereinfachten Anwendbarkeit von MMBSEFE in KMUs in Abschnitts 3.4 entwickelt. Hierbei wurde besondere Rücksicht auf eine enge Modellkopplung, eine möglichst einfache Einarbeitung, effektive Nutzung und einfache Erweiterbarkeit gelegt, um den Aufwand der Anwendbarkeit zu verringern, Konflikte zu reduzieren und Konzepte so komplex wie nötig, aber so schlank wie möglich zu konzipieren.

5.3.6 Schritt 2b - Fragenkatalog zur Zufriedenheit

Zur Vorbereitung der Interviews zur Zufriedenheit der Stakeholder nach Umsetzung der Fallstudie wurden folgende Fragestellungen vorbereitet, wobei Fragestellung 4 und 5 auf den Antworten zur Erwartungshaltung aufbauen:

1. Ich habe Ihnen mein Projekt gezeigt. Wie denken Sie darüber? Hat es Ihre Erwartungen getroffen?
2. Welche Vorteile sehen Sie nun in dieser engen Verbindung zwischen den Modellen? Wie würde sich das Framework auf Ihre Zusammenarbeit im Team ausgewirken?
3. Welche Verbesserungsmöglichkeiten oder Herausforderungen sehen Sie bei der Anwendung des Frameworks?
4. Es wurde damals angegeben, dass es wichtig ist, dass das Framework ebenfalls bei der Kommunikation mit Kunden helfen soll. Trifft dies Ihrer Meinung nach zu?
5. Es wurde damals angegeben, dass die Ressourcen zur Einführung des Frameworks zu knapp sind. Sehen Sie das immer noch so, sollte das Framework außerhalb dieses Forschungsprojektes breitflächig in der Firma eingesetzt werden?
6. Könnte das Framework ebenso bei der Beschaffung von Materialien, also dem Einkauf von Maschinenteilen helfen?
7. Haben Sie noch weitere Anmerkungen oder Nachteile, die sie zu meinem Projekt beitragen möchten? Gibt es noch andere Bereiche/Teile, die verbessert werden könnten?

8. Sehen Sie das Framework als vollständigen Ersatz für die bisherige mündliche Kommunikation?

Zur Validierung des Fragenkatalogs wurde dieser auch der Firmenleitung vorgelegt und gemeinsam besprochen, um dessen Eignung und Übereinstimmung mit den Firmenleitlinien sicherzustellen. Es traten dabei keine Bedenken auf.

Zur Vollständigkeit findet sich der ausführliche Fragenkatalog zur Zufriedenheit der Stakeholder im Anhang A.12.6.

5.3.7 Schritt 4b – Interviews zur Zufriedenheit

Nach Implementierung des Projekts wurden dieselben Stakeholder wie zur Erwartungshaltung aus Abschnitt 5.3.4 nun zur Zufriedenheit befragt. Hierbei wurde ein Fallbeispiel präsentiert, positive und negative Eindrücken aufgenommen sowie die Zufriedenheit anhand des Fragenkatalogs (vgl. Anhang A.12.6) erfasst und gemeinsam reflektiert. Die in Abschnitt 5.1 und 5.2 innerhalb von drei Monaten (~600 Personenstunden) und unter Rücksprache mit den Stakeholdern entwickele Fallstudie diente hierfür als Diskussionsgrundlage und Anwendungsbeispiel für MMBSEFE. Die Interviews zum Abschluss des Projektes zur Zufriedenheit der Stakeholder wurden mit dem Maschinenbauer, Informatiker und Sicherheitsingenieur zur besseren Auswertung ebenso aufgezeichnet. Aufgrund wiederholt zeitlich knapper Ressourcen auf Seiten des Stakeholders Elektroingenieur und seines Ausscheidens aus dem Unternehmen konnte seine Befragung lediglich schriftlich gestellt und dokumentiert werden. Die ausführlichen Antworten der Stakeholder sind im Anhang A.12 den Stakeholdern und Fragestellungen zugeordnet. Ausufernden Antworten, firmeninterne vertrauliche Informationen, Informationen über vergangene Projekte und persönliche Daten wurden ähnlich Abschnitt 5.3.4 nach Meyermann et al. [298] aus den Protokollen entfernt. Alle relevanten und projektbezogenen Aussagen wurden hingegen wiederum wortwörtlich dokumentiert und werden im Folgenden in den zentralen Aussagen zitiert oder zusammengefasst.

Maschinenbauer

Der Maschinenbauer zeigt gemischte Gefühle bezüglich des vorgestellten Projekts. Er anerkennt das Potenzial des Frameworks für verbesserte Kommunikation und Zusammenarbeit, zeigt sich jedoch skeptisch über den zusätzlichen Aufwand und die praktische Anwendbarkeit. Seine Hauptwerkzeuge, wie AutoCAD, scheinen bereits all seine Bedürfnisse zu erfüllen, und er sieht das neue Framework eher als eine Komplikation an, die möglicherweise das Rad neu erfindet, anstatt eine echte Innovation zu bieten.

Die Hauptvorteile des Frameworks sieht er in der verbesserten Teamkommunikation und der potenziellen Klärung von Verantwortlichkeiten. Er sieht auch einen möglichen Nutzen in der Kundenkommunikation, wo das Framework helfen könnte, Kundenanforderungen genauer zu erfassen und Lösungen effektiver zu vermitteln. Trotzdem bleibt er bei seiner bevorzugten Nutzung von AutoCAD für die meisten Aufgaben und zeigt sich zurückhaltend gegenüber der Idee, das Framework als vollständigen Ersatz für bestehende Systeme zu sehen.

Herausforderungen sieht der Maschinenbauer insbesondere in der Überladung mit Informationen und der Möglichkeit, unwichtige Daten auszublenden, um den Arbeitsprozess nicht unnötig zu komplizieren. Er äußert Bedenken hinsichtlich der Ressourcen, die für eine Einarbeitung in das neue System erforderlich wären, und zweifelt daran, ob die Vorteile den Aufwand rechtfertigen würden. Abschließend merkt er an, dass Anpassbarkeit und Fehleranfälligkeit kritische Aspekte sind, die verbessert werden müssten, um das Framework attraktiver für den praktischen Einsatz zu machen.

Elektroingenieur

Der Elektroingenieur äußert sich insgesamt zufrieden mit der engen Verbindung seiner Modelle zum mechanischen Design, was ihm eine präzisere Entwicklung und eine verbesserte Nachvollziehbarkeit von Änderungen ermöglicht. Diese Integration führt zu effizienteren Arbeitsprozessen und verbessert die Qualitätssicherung, indem elektrotechnische Komponenten direkt mit mechanischen Entwicklungen interagieren. Dies erleichtert die Teamarbeit und ermöglicht eine schnellere Identifikation von potenziellen Fehlerquellen.

Trotz der Vorteile gibt es auch Herausforderungen und Verbesserungspotenziale. Der Elektroingenieur empfindet die doppelte Entwicklung in Papyrus und EPLAN als mühsam und wünscht sich eine direktere Anbindung an EPLAN, um Arbeitsprozesse zu vereinfachen. Ebenso sieht er das Potenzial für externe Überprüfungen der Modelle, beispielsweise durch LabVIEW, was die Transparenz und Zugänglichkeit erhöhen könnte. Die Eindeutigkeit der Modelle muss ebenfalls verbessert werden, um die Qualität weiter zu steigern.

Ein wesentlicher Nachteil ist die Einarbeitungszeit in das Framework, die aufgrund knapper Ressourcen als Herausforderung gesehen wird. Der Elektroingenieur ist bereit, sich weiter einzuarbeiten, meint jedoch, dass die Anwendung des Frameworks außerhalb eines Forschungsprojekts bei ProNES schwierig ist. Dies liegt hauptsächlich an den begrenzten Ressourcen und der fehlenden Zeit für eine umfassende Einarbeitung, die von der Unternehmensleitung unterstützt werden müsste.

Informatiker

Der Informatiker ist mit dem MMBSEFE-Modellierungstool sehr zufrieden und betrachtet es als eine passende Lösung für seine Arbeit. Dieses Framework bietet eine klare und einfache Struktur, die ihm erlaubt, seine gewohnte Arbeitsweise weitgehend beizubehalten, wobei nur minimale Anpassungen nötig sind. Es verbessert die Effizienz, indem es die direkte Integration von E/E/PE-Design ermöglicht, was potenzielle Fehlerquellen frühzeitig erkennbar macht. Der Informatiker profitiert besonders davon, dass er nun die Schnittstellen zum Schaltplan direkt im Modell sehen und bearbeiten kann. Diese Transparenz fördert eine effektivere Verifizierung und Kommunikation der Schnittstellen, was zu einer verbesserten Modellierungsebene führt.

Die Einbindung des Frameworks in seine tägliche Arbeit ermöglicht es dem Informatiker, Modelle selbst zu gestalten und anzupassen, was ihm eine bessere Kontrolle und Übersicht über die Systemarchitektur gibt. Dadurch können Entwurfsfehler minimiert und die Entwicklungsdynamik erhöht werden. Er schätzt auch die verbesserte Kommunikation mit anderen Teammitgliedern, die durch das Framework erleichtert wird.

Der Informatiker sieht kaum Nachteile oder wesentliche Verbesserungspotenziale. Einzige Ausnahme ist die Implementierung eines Transformators oder Codegenerators zu TwinSAFE, welche als geringfügig angesehen wurde (vgl. S. 42, S. 125 und S. 141) und keinen Forschungsmehrwert bildet.

Sicherheitsingenieur

Für den Sicherheitsingenieur besteht ein großer Vorteil von MMBSEFE darin, dass bereits entwickelte Gefährdungssituationen, Folgen und Items für neue Maschinen genutzt werden können. Dadurch entstehen im Laufe der Zeit ein Katalog oder Bibliotheken, auf die das Team zurückgreifen kann. Diese Wiederverwendung ermöglicht eine effiziente und standardisierte Vorgehensweise und einfachere Kombination und Koordination bei der Identifizierung und Bewertung von Gefährdungen, im Gegensatz zu einfachen Listen. Er sieht einen weiteren Vorteil darin, dass seine Analysewerkzeuge direkt mit dem Framework gekoppelt werden können. Dadurch wird die manuelle Übertragung vermieden und die Effizienz gesteigert, da er sich kaum in neue Umgebungen einarbeiten muss. Die enge Modellkopplung, direkte Anwendbarkeit über Anwendungsprofile, OCL-Constraints und verbesserte Darstellbarkeit über CSS erleichtert den Analyseprozess und sorgt für eine nahtlose Integration der Ergebnisse. Zusätzlich hat der Sicherheitsingenieur die Möglichkeit, seine Analysen gemeinsam mit dem Informatiker besser in MMBSEFE einzubinden oder sogar mit anderen Werkzeugen wie der WEKA-Software über LabVIEW-Transformatoren zu koppeln, sofern eine passende Schnittstelle gefunden

wird. Dadurch eröffnen sich weitere Möglichkeiten zur Optimierung der Analyseprozesse und zur Integration verschiedener Analysewerkzeuge, was zu einer umfassenderen und genaueren Bewertung der Sicherheitsaspekte führt.

Der Sicherheitsingenieur betont, dass er vom gemeinsamen Framework überzeugt ist, jedoch anmerkt, dass eine bessere Einbindung von Normen wünschenswert wäre.

Aktuell sind nur die Normen ISO 12100 und ISO 13849 im Framework berücksichtigt. Er würde sich ebenso ein Anwendungsprofil für Risikographen wünschen, da er bei dem Interview zur Modellierung von DFTs und SBBM den Mehrwert von Drop-Down-Menus als sehr hilfreich empfand. Der Sicherheitsingenieur erkennt den Wert einer umfangreicheren Einbindung von Normen, um den geltenden Sicherheitsstandards und Vorschriften gerecht zu werden. Eine Erweiterung des Frameworks um eine größere Bandbreite von Normen würde seiner Meinung nach zu einer verbesserten Unterstützung bei der Konformität mit den erforderlichen Sicherheitsrichtlinien führen.

5.3.8 Schritt 5b – Transkription, Analyse und Interpretation zur Zufriedenheit

Die ausführlichen Antworten aller interviewten Stakeholder finden sich vollständig transkribiert im Anhang A.12.7 bis A.12.10. Die wesentlichen Erkenntnisse aus den Interviews werden im Folgenden zusammengefasst, um im Kontext des festgelegten Evaluationsziels Aussagen über die erreichte Effektivität, Effizienz und Benutzerfreundlichkeit zu geben. Zuletzt finden sich Empfehlungen basierend auf den gewonnenen Erkenntnissen.

Zusammenfassung der Aussagen

Zusammenfassend zeigt Tabelle 5.4 den Zufriedenheitsgrad der Stakeholder. Hierbei wird in verschiedenen Kategorien unterschieden: ob MMBSEFE im Allgemeinen den Erwartungen entspricht, ob MMBSEFE die Zusammenarbeit und Kommunikation innerhalb des technischen Teams sowie mit dem Einkauf und den Kunden verbessert, ob MMBSEFE die mündliche Kommunikation ersetzen kann und welche weiteren Verbesserungsmöglichkeiten und Herausforderungen identifiziert wurden.

	Maschinen-bauer	Elektro -ingenieur	Informatiker	Sicherheits-ingenieur
Erwartungen getroffen	Nein, bestenfalls teilweise	Ja, positiv überrascht	Ja, überzeugt	Ja, positiv beeindruckt
Verbesserung der Zusammenarbeit und Kommunikation	Ja, wenn verfeinert	Ja, Einfache Nachvollziehbarkeit von Änderungen	Ja, Verknüpfungen und Schnittstelleninfos, aber initialer Aufwand hoch	Positiver, besserer Einfluss in Teamarbeit
Angesprochene Verbesserungsmöglichkeiten & Herausforderungen	Zu viele Informationen, Überkompliziertheit, Anpassbarkeit	Einarbeitungszeit, Schulungsbedarf, Workshops, bessere Einarbeitung	Integration von Codegeneratoren & konstruktiven Werkzeugen, Benutzeroberfläche	Integration weiterer Normen, verbesserte Nutzung wie WEKA
Verbesserung der Kommunikation zum Kunden	Ja, aber besser mit CAD möglich	Ja, verbessert die Kundensicht	Nicht wirklich, da eher auf Entwickler konzentriert	Ja, Verbesserung Risikokommunikation
Einsetzbarkeit	Nein, Ressourcen zu knapp	Evtl. langfristig lohnend	Ja, langfristige Vorteile überwiegen	Möglich
Nutzbarkeit für den Einkauf	Ja, aber AutoCAD bevorzugt	Unsicher, EPLAN bevorzugt	Potenzial vorhanden, jedoch ungewiss	Nicht beantwortet
Ersatz mündlicher Kommunikation	Nein	Nein	Ergänzend, nicht ersetzend	Ergänzend, nicht ersetzend

Tabelle 5.4: Zusammenfassung zur Zufriedenheit der Stakeholder

Auf Basis der gegebenen Zusammenfassungen können die Aussagen der Stakeholder auf die Kriterien Effektivität, Effizienz und Benutzerfreundlichkeit abgebildet werden.

Effektivität

Die Effektivität des MBSE-Ansatzes in der Fallstudie wird durch die positiven Rückmeldungen hinsichtlich der verbesserten Kommunikation und Zusammenarbeit bestätigt (vgl. S. 407, S. 408, S. 410 und S. 412). Die Anwendung von UML- und SysML-Modellen ermöglicht eine konsistente und klare Modellierung, was insbesondere für den Elektroingenieur wichtig ist (vgl. S. 406, S. 408 und S. 412). Diese Modelle unterstützen die Vollständigkeit der technischen Dokumentation und gewährleisten die Korrektheit der Lösungen, indem sie eine gemeinsame Sprache schaffen und so das Verständnis verbessern (vgl. S. 403 und S. 413).

Effizienz

Die Effizienz des Einsatzes von MBSE wird jedoch durch die Anforderungen an die Einarbeitung und die Integration in bestehende Prozesse relativiert (vgl. S. 407, S. 409 und S. 410). Während die Stakeholder die verbesserte Nachvollziehbarkeit und Fehlerreduktion durch MBSE schätzen, zeigt sich, dass der initiale Zeitaufwand für Schulungen und die Anpassung der Arbeitsweisen einen kritischen Punkt darstellt (vgl. S. 409 und S. 410). Der Elektroingenieur und der Informatiker weisen darauf hin, dass trotz der Effektivität des Systems der Zeitaufwand für die Einarbeitung die kurzfristige Effizienz beeinträchtigt (vgl. S. 409 und S. 410). Langfristig könnte jedoch eine gesteigerte Effizienz erzielt werden, wenn das System vollständig integriert und von allen Stakeholdern genutzt wird (vgl. S. 409 und S. 411).

Benutzerzufriedenheit

Die Benutzerzufriedenheit spiegelt sich in den differenzierten Meinungen und dem Bedürfnis nach weiterer Anpassung wieder (vgl. S. 406, S. 408, S. 410 und S. 412). Während der Informatiker und der Elektroingenieur eine hohe Zufriedenheit mit der Flexibilität und der Integrationsfähigkeit des Systems ausdrücken (vgl. S. 408 und S. 410), äußern die Sicherheitsingenieure Bedenken hinsichtlich der fehlenden Normen und Standards (vgl. S. 413). Um die Benutzerzufriedenheit weiter zu erhöhen, ist es entscheidend, dass das MBSE-Framework an die spezifischen Anforderungen der unterschiedlichen technischen Disziplinen angepasst wird und dass eine kontinuierliche Weiterentwicklung und Support gewährleistet sind (vgl. S. 408, S. 410, S. 411 und S. 412).

Empfehlung

Die Interviews zeigen somit, dass, reflektiert an der Fallstudie, MMBSEFE in Teilen eine adäquate Lösung für die Modelle der Stakeholder darstellen kann, die die Effektivität und teilweise die Effizienz der Entwicklungsprozesse verbessert sowie die Benutzerzufriedenheit steigert. Es bleibt jedoch wichtig, dass das System kontinuierlich evaluiert und entsprechend den Rückmeldungen der Stakeholder verbessert und erweitert wird, um Erweiterungspotentiale wie die Anbindung von Werkzeugen der technischen Stakeholder, das Anlegen von Bibliotheken für Prüfmaschinenteile und Risikoitems sowie die Integration weiterer Standards umzusetzen und somit eine optimale Balance zwischen Effektivität, Effizienz und Benutzerzufriedenheit zu erreichen.

5.4 Zusammenfassung

In diesem Kapitel wurde eine Fallstudie detailliert beschrieben, die die verschiedenen Aspekte von MMBSEFE (maschinenbauspezifische Model-Based Systems Engineering Framework-Erweiterung) sowie seine entsprechenden Konzepte und Implementierungen nutzt, um die Lösung zu bewerten. Basierend auf dieser konkreten Fallstudie wurden die Problemfelder dieser Arbeit zur Umsetzung einer maschinenbauspezifischen MBSE-Framework-Erweiterung, zur Integration von Sicherheitsanalysen des Maschinenbaus und zur Anwendbarkeit in KMUs angesprochen, um die Forschungsaufgaben des Typs Experiment umzusetzen. Hierfür wurden Modelle entworfen und analysiert sowie die Ergebnisse interpretiert und diskutiert.

In Abschnitt 5.1 wurde zunächst die konkrete Prüfmaschine modelliert und anhand dessen die Abbildbarkeit der Risikobeurteilung, aufbauend auf der Bestimmung der Grenzen der Maschine, über die Risikoeinschätzung bis hin zur Risikobewertung über die Einflussfaktoren des Risikographen aus ISO13849 behandelt und dargestellt. Die Bewertung zeigt, dass die Modellierungslösung und Implementierung die Stakeholder hinreichend abbildet und in einen gemeinsamen MBSE-Prozess einbindet, das die Anwendungsfälle erfüllt und einen anwendbaren Ansatz mit Möglichkeit zur Anwendung aller Prozesse zur Risikobeurteilung (vgl. Abschnitt 2.1.3) vorgibt. Abschnitt 5.2 hat die Stakeholder-spezifischen Modelle für Sicherheitsfunktionen aufgebaut und dargestellt, dass die Modelle und Transformatoren für den Nachweis der Risikominderung die Analysemethoden hinreichend darstellen und Werkzeuge hinreichend anbinden. Dabei liegt die Präferenz aufgrund der einfacheren Anwendbarkeit für KMUs auf SBBM. Für Risikographen hat sich ergeben, dass sich eine enge Toolkopplung für Analysen mit einfachen Berechnungsfunktionen durchaus anbieten kann. Schließlich haben in Abschnitt 5.3 Stakeholder der verschiedenen Fachbereiche das Fallbeispiel in Einzelinterviews bewertet und die Verbesserung der Effektivität, Effizienz und Benutzerfreundlichkeit durch Einführung von MMBSEFE bestätigt sowie ihre Akzeptanz zum Einsatz geäußert. Zusammenfassend wurden alle vorab definierten Problemfelder durch das Fallbeispiel dieses Kapitels behandelt, wobei für alle in dieser Arbeit beschriebenen offenen Herausforderungen Lösungen gefunden wurden. Tabelle 5.5 zeigt dies und bietet zusätzlich eine stichwortartige Zusammenfassung der Forschungsaufgaben und Verweise in die entsprechenden Abschnitte der Arbeit.

Forschungsaufgaben (FA)		
FA	**Beschreibung**	**Abschnitt**
Maschinenbauspezifische Modellierung		
FA1-B	Recherche einer gemeinsamen Modellebene und Architektur für MBSE im Maschinenbau, im Speziellen für Prüfmaschinen, und Grundlage zur Modularisierung	2.1
FA1-TB	Modellierung der Anwendungsfälle und Metamodelle für alle Stakeholder in einer maschinenbauspezifischen Model-Based Systems Engineering Framework-Erweiterung	3.2
FA1-I	Umsetzung einer modularisierten gemeinsamen Modellebene und Architektur für MBSE im Maschinenbau, im Speziellen für Prüfmaschinen	4.1.1 4.2.1
FA1-E	Evaluation einer modularisierten gemeinsamen Modellebene und Architektur für MBSE im Maschinenbau, im Speziellen für Prüfmaschinen	5.1
Integration der Analysen des Maschinenbaus		
FA2-B	Identifikation maschinenbauspezifische Analysemethoden und passender Frameworks	2.2
FA2-TB	Modellierung der Metamodelle für die Analysemethoden dynamische Fehlerbaumanalyse (DFT) und sicherheitsbezogene Blockdiagrammmethode (SBBM) zum Nachweise der Risikominderung	3.3
FA2-I	Entwurf und Implementierung zur Integration maschinenbauspezifische Analysemethoden	4.2.2– 4.2.4
FA2-E	Evaluation typischer Analysemethoden	5.2
MBSE-Prozesse für KMUs		
FA3-B	Identifikation der Hindernisse zur Anwendbarkeit von MBSE in KMUs	2.3
FA3-TB	Modellierung eines Architekturkonzeptes zur Anwendbarkeit von MBSE in KMUs unter Berücksichtigung der Aspekte Agilität, Vertrauen, Zusammenarbeit und Effizienz	3.4
FA3-I	Implementierung eines Konzeptes für KMUs in der Maschinenindustrie	4.1.2
FA3-E	Evaluation mit KMU	5.3

Tabelle 5.5: Forschungsaufgaben nach Evaluation

Diese Evaluation zeigt somit, dass die Modellierung und Implementierung der in dieser Arbeit beschriebenen Konzepte nicht nur zur Weiterentwicklung, sondern auch zur Verbesserung von modellbasierter Entwicklung in kleinen und mittelständischen Unternehmen beitragen kann.

Während der Evaluation haben sich allerdings Möglichkeiten zur Erweiterbarkeit der Modelle gezeigt, die im Rahmen zukünftiger Arbeiten Ansätze liefern können. Hierunter zählen die Erweiterbarkeit der Modelle des mechanischen (S. 171) und des elektrotechnischen Designs (S. 172) sowie die Implementierung von Transformatoren zu Entwicklerwerkzeugen oder zur externen Validierung der Modelle (S. 211), um die Bedürfnisse von Maschinenbauer und Elektroingenieur detaillierter abzubilden und deren Werkzeuge anzubinden. Ebenso wurde identifiziert, dass eine Erweiterung zur Zuweisung von Gefährdungssituationen und Effekten zur besseren Nachvollziehbarkeit und genaueren Berechenbarkeit einzelner Risikoitems beitragen kann (S. 183). Beim Einsatz in weiteren Evaluationsprojekten anderer Maschinen kann es notwendig werden, Anpassungen oder Erweiterungen von Prüfmaschinenteilen umzusetzen, um

entsprechende Funktionen abbilden zu können (S. 184). Sollte das Konzept in anderen Modellierungsplattformen, wie MagicDraw Einsatz finden, kann dies zur Anpassung der Konzepte bei der Darstellung von Elementen über CSS führen (S. 184). Ebenso lässt sich MMBSEFE noch tiefer in Papyrus integrieren, was die Anwendbarkeit weiter vereinfacht (S. 184). Für die Sicherheitsbetrachtungen wurde erwähnt, dass DFTs ebenso andere Verteilungsfunktionen besitzen können und die Möglichkeit besteht, DC oder CCF einzubeziehen (S. 195) sowie, dass Herstellerbibliotheken zur automatischen Befüllung von SBBM-Elementen in SysML eingebunden werden könnten (S. 197). Zuletzt könnte es sich nach Feedback des Sicherheitsingenieurs anbieten, weitere domänenspezifische und -unabhängige Standards in RAAML einzubinden (S. 212).

6 Zusammenfassung der Arbeit und Ausblick

Das letzte Kapitel dieser Arbeit gliedert sich in die Zusammenfassung der Arbeit in Abschnitt 6.1 und einen Ausblick in Abschnitt 6.2 über mögliche zukünftige Arbeiten zur Fortführung der entwickelten Ansätze.

6.1 Zusammenfassung

Die vorliegende Arbeit entstand vor dem Hintergrund der zunehmenden Komplexität von Systemen in allen Industriebereichen, insbesondere in Bezug auf die wachsende Verflechtung von Software und Hardware. Die steigende Komplexität führt dabei zu komplexeren Sicherheitsanforderungen und einem verstärkten Einsatz von Software. Trotz dieser Entwicklungen ist der sich als verbesserungsfähige Engineering-Praxis herausgestellte dokumentenzentrierte Ansatz aufgrund häufig mangelnder Innovationsfähigkeiten in vielen kleinen und mittleren Unternehmen (KMUs) noch weit verbreitet. Ein Problem entsteht dabei durch die lose Verbindung von Modellen verschiedener Stakeholder. Ohne hinreichende Integration verschiedener Modelle auf Basis einer gemeinsamen Datenbasis können leicht Inkonsistenzen entstehen, die erst spät im Entwicklungsprozess erkannt werden. Dies erhöht den Testaufwand und macht notwendige Änderungen schnell kostspieliger. Um diesen Herausforderungen zu begegnen, ist es notwendig, Model-Based Systems Engineering (MBSE) einzuführen, was die Grundlage für eine gemeinsame Datenbasis legt, in der sich die verschiedenen Stakeholder abbilden können und die als Kommunikationsbasis dient. Die Systems Modeling Language (SysML), die bereits eine Erweiterung der Unified Modeling Language (UML) darstellt, kann hierbei als weitverbreiteter Standard für MBSE angesehen werden. Die Arbeit konzentriert sich auf die Domäne des Maschinenbaus, insbesondere auf die Anwendung in Prüfmaschinen, die historisch gesehen oft vernachlässigt wurde. Wenige MBSE-Ansätze wurden bisher für den Maschinenbau und die entsprechenden Sicherheitsstandards (ISO 12100, ISO 13849 und IEC 62061) entwickelt, obwohl die Relevanz durch die zunehmende Komplexität von Systemen und Softwareanteilen steigt. Die Regeln für Unternehmen, die Systeme auf Basis entsprechender Sicherheitsstandards mit entsprechenden Eigenschaften entwickeln wollen, sind zunächst unabhängig der Unternehmensgröße, allerdings wurde in dieser Arbeit Rücksicht auf die Einschränkungen von KMUs

Model-Based Systems Engineering sicherheitskritischer Anwendungen im Maschinenbau mittelständischer Unternehmen © 2024 by Nick Berezowski is licensed under CC BY 4.0

genommen. KMUs verfügen häufig über begrenzte Ressourcen, setzen dabei aber oft teure vorzertifizierte Lösungen ein. Der Mangel an Knowhow in KMUs ist in diesem Zusammenhang ein weiteres Hindernis. Finanziell stehen oft nicht genug Mittel für kostenintensive Weiterbildungen zur Verfügung, zeitlich steht oft nicht genug Zeit für Einarbeitungen in neue Modellierungswerkzeuge zur Verfügung und personell stehen KMUs oft unter starkem Projektdruck, der ebenso die Ressourcen zur Weiterbildung und Einarbeitung verringert. Durch Erfahrungsberichte vom KMU ProNES Automation GmbH (ProNES) hat sich zudem ein geringer Marktzugang zur eigenständigen Entwicklung von Lösungen im Bereich der Entwicklung von sicherheitskritischen Maschinen herausgestellt.

Aus dieser Motivation entstand das Hauptziel dieser Arbeit. Es soll die Qualität und Sicherheit von Maschinen, im speziellen Prüfmaschinen, erhöht werden, um bessere Entwicklungsergebnisse zu erzielen, wobei gleichzeitig die Anforderungen und Ressourcen von KMUs berücksichtigt werden sollen, um einen besseren Marktzugang zu schaffen. Aus diesem Hauptziel wurden drei wichtige Forschungsziele (FZ) zur Anwendung von MBSE bei der Entwicklung von sicherheitskritischen Systemen in KMUs des Maschinenbaus identifiziert und Ansätze entwickelt, implementiert und evaluiert.

FZ1: Maschinenbauspezifische Integration von System-, Software- und sicherheitsbezogenen Modellen

Im Rahmen von FZ1 lag der Fokus darauf, die Integration verschiedener domänenspezifischer Begriffe in einem gemeinsamen Framework auf Basis von SysML zu ermöglichen. Durch diese Integration sollten Gemeinsamkeiten und eine bessere Kommunikation zwischen den Stakeholdern innerhalb der Domäne der Maschinenindustrie hergestellt werden. Die Notwendigkeit dieser Integration ergab sich aus der Tatsache, dass die Komplexität von Maschinensystemen eine umfassende Zusammenarbeit verschiedener Bereiche (Mechanik, Elektronik, Hydraulik, Pneumatik und Informatik) erfordert. Durch die Verwendung eines einheitlichen Frameworks können die Modelle und Informationen aus den verschiedenen Gebieten effizient zusammengeführt und gemeinsam genutzt werden. Dadurch wird eine konsistente und kohärente Darstellung des Gesamtsystems ermöglicht, was die Grundlage für eine verbesserte Entwicklung, Analyse und Sicherheitsbewertung bildet.

FZ2: Integration der Analysemethoden aus dem Maschinenbau

FZ2 bestand darin, domänenspezifische Analysemethoden aus dem Maschinenbau in das unter FZ1 geschaffene MBSE-Framework zu integrieren, um ein gemeinsames Modell zu schaffen, in welchen Funktionen und Eigenschaften der Analysemethoden dargestellt werden können. Es

wurde festgestellt, dass es bisher keine allgemeingültige wiederverwendbare MBSE-Framework-Erweiterung für die Integration von maschinenbauspezifischen Analysemethoden, wie Risikographen, sicherheitsbezogene Blockdiagramme (SBBM) und dynamische Fehlerbäume (DFT), gab. Bestehende Ansätze wie die Risk Analysis and Assessment Modeling Language (RAAML) der Object Management Group (OMG), die auf erweiterbaren grundlegenden Begriffen der funktionalen Sicherheit beruht, waren zum Zeitpunkt der Arbeit nicht für den Maschinenbau ausgelegt. Daher wurden die grundlegenden Begriffe in RAAML für Risikographen und SBBM erweitert, um die maschinenbauspezifischen Analysemethoden angemessen modellieren zu können, sowie DFTs auf Basis der bereits enthaltenen Fehlerbaumanalyse erweitert.

FZ3: MBSE-Prozesse für KMUs

FZ3 konzentrierte sich auf die Entwicklung von MBSE-Prozessen und Konzepten, die speziell auf die Bedürfnisse von KMUs zugeschnitten sind und die Anwendung der unter FZ1 und FZ2 entwickelten MBSE-Framework-Erweiterung erleichtern. Es wurde festgestellt, dass es bisher kein einfach anwendbares Konzept für Sicherheitsbewertungsprozesse und bereits betriebsbewährte Werkzeuge gab, die speziell auf die Bedürfnisse von KMUs zugeschnitten sind. FZ3 bestand also darin, anwendbare Konzepte zu entwickeln, die den KMUs ermöglichen, MBSE effektiv in ihre Prozesse zu integrieren und die Sicherheitsbewertung ihrer Maschinen zu erleichtern.

Methodische Ergebnisse

Um die spezifischen Aspekte der technischen Stakeholder im Maschinenbau, insbesondere bei der Entwicklung von Prüfmaschinen, angemessen abzubilden, wurde SysML/UML erweitert (FZ1). Dies geschah durch die Entwicklung spezifischer Profile und Bibliotheken, die die notwendigen domänenspezifischen Begriffe und Modelle enthalten. Dadurch konnten die technischen Anforderungen und Eigenschaften der Maschinen in der Modellierungsumgebung präzise dargestellt werden, was eine verbesserte Kommunikation und Zusammenarbeit zwischen den technischen Stakeholdern ermöglicht, was wiederum zu einer höheren Qualität der Entwicklungsprozesse führt (FZ1). Um die spezifischen Aspekte von Sicherheitsingenieuren im Maschinenbau auf der gemeinsamen Modellebene umsetzen zu können, wurden Profile und Bibliotheken als Erweiterung zu RAAML entwickelt, die die Begrifflichkeiten aus ISO 12100 und ISO 13849 nutzen, um eine umfassende Darstellung der Sicherheitsaspekte in den Modellen zu ermöglichen (FZ1). Dadurch konnten Sicherheitsanforderungen und -funktionen systematisch und einheitlich modelliert werden, was zu einer verbesserten Sicherheitsbewertung und Risikobeurteilung beiträgt (FZ1).

Zur Anwendung verschiedener domänenspezifischer Analysemethoden zur Sicherheitsanalyse wurde RAAML für Risikographen, DFTs und SBBM erweitert (FZ2). Dies ermöglicht eine quantitative Bewertung durch die Anreicherung der gemeinsamen Modelle mit den Grundlagen zur Modellierung und Berechnung der Sicherheitsanalysen. Um den Sicherheitsingenieur während des Entwicklungsprozesses zu führen und die systematische Identifizierung und Bewertung von Sicherheitsrisiken zu unterstützen, wurden Prozesse zur Risikobeurteilung und Risikominderung modelliert (FZ2). Die Modellierung dieser Prozesse erleichtert die Kommunikation und Koordination und trägt dazu bei, die Erfassung und Bewertung von potenziellen Risiken zu unterstützen und den Nachweis erforderlicher Risikominderungen von softwaregesteuerten Sicherheitsfunktionen in Maschinen zu erbringen (FZ2).

Für KMUs wurde ein Architekturkonzept geschaffen, das durch eine lose Kopplung von betriebsbewährten Werkzeugen für die Sicherheitsanalyse eine Anbindung an die Modellierungsplattform und die angereicherten Modelle schafft (FZ3). Dieses Konzept umfasste den Export von Artefakten aus der Modellierungsumgebung, die anschließende Analyse und Auswertung der Sicherheitsaspekte mit den externen Werkzeugen sowie den Reimport der Ergebnisdaten. Durch diese losere Kopplung, im Gegensatz zur engen Kopplung des OMG-Ansatzes über die Implementierung der Werkzeuge in der Modellebene, können KMUs ihre bereits etablierten Analysewerkzeuge weiterverwenden und erhalten so eine höhere Flexibilität über das Architekturkonzept (FZ3). Um das Architekturkonzept anwendbar für Prüfmaschinen-entwickelnde KMUs im Maschinenbau zu machen, wurde NI LabVIEW zur Anbindung der Analysewerkzeuge anstelle des Eclipse Modeling Frameworks eingesetzt (FZ3). Dadurch können die KMUs ihre bestehenden Programmiererfahrungen zur Anbindung von Analysewerkzeugen effizient nutzen. Dies erleichtert die Umsetzung und ermöglicht den KMUs eine schnellere und reibungslosere Integration weiterer Sicherheitsanalysewerkzeuge (FZ3).

Insgesamt führten diese methodischen Ergebnisse zu einem umfassenden Ansatz, der die spezifischen Anforderungen des Maschinenbaus, die Integration von Sicherheitsaspekten und -analysen sowie die Bedürfnisse von KMUs berücksichtigt.

Implementierungsergebnisse

Die Implementierung dieser Arbeit bildete den praktischen Nachweis der Machbarkeit der entwickelten Konzepte und diente als Grundlage für die Evaluation der Gesamtlösung. Für die Implementierung der Metamodelle wurde die Open Source SysML/UML Modellierungsplattform Papyrus genutzt. Diese Plattform wurde ausgewählt, weil sie die Möglichkeit bietet, die

entwickelten Konzepte und Erweiterungen in einem gängigen Modellierungsumfeld zu integrieren. Dabei steht es offen, auch andere UML-Plattformen zu verwenden und die Lösung auf diese Arbeitsumgebungen anzuwenden. Die entwickelten Modelle und Konzepte wurden in Papyrus vollständig umgesetzt. Dies beinhaltete die Erstellung der spezifischen Profile und Bibliotheken für die Stakeholder Maschinenbauer, Elektroingenieur, Informatiker und Sicherheitsingenieur und für die domänenspezifischen Analysemethoden Risikographen, DFT und SBBM. Für die Anbindung externer Werkzeuge wurden Transformatoren in LabVIEW implementiert, um die Anforderungen von KMUs hinsichtlich der Nutzung ihrer Programmiererfahrung zu erfüllen. Hierbei wurden das Werkzeug Stormchecker für DFT und SISTEMA für SBBM an die Modellierungsumgebung angebunden. Zusätzlich wurde ein Analysewerkzeug für Risikographen in LabVIEW und in OCL (Object Constraint Language) entwickelt, um einen Vergleich zwischen loser und enger Kopplung beispielhaft durchführen zu können. Der Implementierungsaufwand war nicht unerheblich.

Insgesamt bilden die Implementierungsergebnisse einen praktischen Beweis für die Umsetzbarkeit der Metamodelle und des Architekturkonzeptes sowie die Grundlage für die folgende Evaluation.

Evaluationsergebnisse

Die Evaluation der Arbeit basiert auf einer im Rahmen der Arbeit durchgeführten Fallstudie mit dem KMU ProNES, in der eine Batteriezellenprüfmaschine entwickelt wurde. Dabei konnten die verschiedenen Aspekte der Maschine mithilfe der entwickelten Plattform-Erweiterung modelliert, implementiert und quantitativ bewertet werden. Die Fallstudie zeigt, dass der Ansatz die Anforderungen und Bedürfnisse der Stakeholder in der Praxis erfolgreich erfüllen konnte, indem alle definierten Anwendungsfälle der Stakeholder hinreichend abgedeckt und eine Modellierung und Analyse der Batteriezellenprüfmaschine ermöglicht wurde. Der Ansatz erwies sich somit als anwendbar für alle im Prozess der Risikobeurteilung identifizierten Anwendungsfälle und für die modellierten Artefakte. Die entwickelten Erweiterungen von SysML/UML und RAAML ermöglichen eine konsistente Darstellung der Maschinenfunktionen, Eigenschaften und Sicherheitsaspekte. Ein zusätzlicher Aspekt der Evaluation war der mögliche Vergleich zwischen DFTs und SBBM für dieselben Sicherheitsfunktionen. Dabei wurde festgestellt, dass SBBM mathematisch ungenauer sein kann, aber aufgrund seiner einfacheren Anwendbarkeit besonders für Stakeholder mit geringerer Expertise geeignet ist. DFTs hingegen bieten eine präzisere Analyse, erfordern jedoch eine gewisse Erfahrung und erweitertes Knowhow. Der Vergleich zwischen loser und enger Toolkopplung für den Risikographen ergab, dass die Ausführung im Modell auf Basis von OCL weniger aufwendig ist, jedoch der

Ansatz von RAAML mehr Knowhow im Unternehmen erfordert. Die lose Kopplung hingegen ermöglicht eine flexible Integration externer Werkzeuge und berücksichtigt die individuellen Präferenzen und Ressourcen der KMUs. Die Bewertung der Anwendbarkeit der Modelle und Konzepte erfolgte durch Interviews von einzelnen ProNES-Mitarbeitern ohne eine statistische Absicherung. Die Ergebnisse zeigten im Wesentlichen eine positive Resonanz und Akzeptanz der entwickelten Lösung. Die KMU-Mitarbeiter haben verschiedene Vorteile in der Nutzung der Modelle erkannt, bei gleichzeitiger Nennung von Verbesserungsmöglichkeiten. Sie empfanden den Ansatz als nützlich und praktikabel, insbesondere aufgrund der Möglichkeit, ihr bestehendes Knowhow weiter zu nutzen und ressourcenschonend zu erweitern. Abschließend wurden wünschenswerte Erweiterungen und mögliche zukünftige Arbeiten identifiziert.

Die Evaluationsergebnisse liefern wertvolle Erkenntnisse und Feedback, die dazu beitragen können, den Ansatz und die Lösung weiter zu optimieren und den Bedürfnissen der KMUs weitergehend gerecht zu werden. Die erfolgreiche Fallstudie bestätigt die Wirksamkeit der entwickelten Modelle und Konzepte bei ProNES für die Batteriezellenprüfmaschine. Dies unterstreicht insgesamt, aufgrund der Bedürfnisse von KMUs, die Relevanz zur Steigerung von Qualität und Sicherheit bei der Entwicklung von Prüfmaschinen und somit die Hoffnung, dass diese Arbeit einen Beitrag leistet, die Situation für KMUs zu verbessern.

6.2 Ausblick

In der Evaluierung wurde deutlich, dass einige der vorgestellten Lösungen weiterentwickelt werden können. Für zukünftige Arbeiten ergeben sich daher mehrere mögliche Erweiterungen, auf die im Folgenden kurz eingegangen werden soll, ohne Anspruch auf Vollständigkeit.

Einsatz in weiteren Anwendungsfällen und Verbesserung der Usability

Um die Wirksamkeit und Anwendbarkeit der entwickelten Modelle und Konzepte in unterschiedlichen Kontexten zu überprüfen, bietet sich der Einsatz in weiteren Anwendungsfällen und Fallstudien an. Dadurch können die Ergebnisse validiert und mögliche notwendige Anpassungen oder Erweiterungen ermittelt werden. Im Rahmen dessen kann die Usability der Modellierungsumgebung weiter verbessert werden, um die Anwendung der Modelle und Konzepte noch benutzerfreundlicher zu gestalten. Dies beinhaltet mögliche Optimierungen der Benutzeroberfläche, der Interaktionsmöglichkeiten und der Unterstützung bei der Modellierung.

Erweiterung der Sicherheitsanalysemodelle

Für sicherheitsbezogene Blockdiagramme existieren verschiedene Bibliotheken von Herstellern, die bereits feste Ausfallwerte für ihre Bauteile oder Teilsysteme bereitstellen. Diese sind

in standardisierten Formaten erhältlich und beispielsweise direkt in SISTEMA integrierbar. Es handelt sich entweder um direkte SISTEMA-Bibliotheken oder um XML-Bibliotheken vom Verband Deutscher Maschinen- und Anlagenbauer (VDMA). Durch die Integration dieser Bibliotheken für standardisierte Sicherheitsfunktionen und Komponenten lässt sich zusätzlich die Anwendbarkeit komfortabler gestalten.

Ebenso gibt es für dynamische Fehlerbäume Optionen zur verbesserten Anwendung. So ist es nicht unüblich, dass Basisereignisse aufgrund mehrfacher Ausführung von Sensoren oder Aktoren mehrfach ausgeführt werden. Ein Replikationsfaktor kann hierbei die Darstellung des Fehlerbaums vereinfachen, indem beispielsweise bei einem UND-Gate vier identische Ereignisse durch eines mit einem Replikationsfaktor von vier ersetzt wird.

Implementierung & Validierung von Transformatoren sowie Generierung von Software
Die vorliegende Arbeit enthält eine prototypische Implementierung von Transformatoren, daher sollte beachtet werden, dass keine Ansprüche an kommerzielle und validierbare Qualität gestellt werden. Wenn eine solche Qualität angestrebt wird, müssen entsprechende Arbeiten geleistet werden. Außerdem, sollte in der Zukunft die Generierung von Software für den kommerziellen Einsatz angestrebt werden, wird neben entsprechenden Transformatoren aus UML-Zustandsdiagrammen eine sorgfältige Prüfung der Generierung notwendig, um sicherzustellen, dass der Code korrekt und sicher funktioniert. Hierbei kann unter anderem eine Testsuite oder andere Verfahren Einsatz finden, um Compiler, Bibliotheken, Linker oder Optimizer zu überprüfen. Ein Beispiel wären Prüfungen auf Plausibilität während der Laufzeit, indem auf Unter- und Obergrenzen, andere Vor- und Nachbedingungen oder ähnliches geprüft wird, um entsprechende Korrekturmaßnahmen zu treffen.

Weiterentwicklung der Stakeholder-spezifischen Modellierung
Die Modelle für die mechanische und elektrotechnische Modellierung können durch genauere Element-Bibliotheken verbessert werden und Transformatoren zur Werkzeuganbindung von CAD und Schaltplan mit Testsuiten entwickelt werden. Um die Anwendung bei der Entwicklung sicherheitskritischer Maschinen weiter zu erleichtern, können ebenso weitere Erweiterungen umgesetzt werden, die nicht Teil der Arbeit waren. So hat sich über das Feedback des Sicherheitsingenieurs herausgestellt, dass die Abbildung weiterer Normen, wie beispielsweise die ISO 13850 zur Entwicklung von Nothalt-Funktionen, in den Metamodellen für ihn die Anwendbarkeit erleichtern. So könnte beispielsweise die Entwicklung von Anforderungen an das System und die Abbildung auf Maschinenteile anhand eines Anforderungskatalogs weiter verbessert werden. Dies ist zwar generell nicht notwendig, um eine erfolgreiche Risikobeurteilung

durchzuführen, doch bietet es eine Hilfestellung für Sicherheitsingenieure in der Framework-Erweiterung, ähnlich etablierter Risikobeurteilungssoftware-Lösungen wie WEKA-Software.

Anwendung für andere Maschinen und für andere Domänen

Die entwickelten Modelle können auch auf andere Maschinen angewendet werden, um die Vollständigkeit der Maschinenteile und Prüfmaschinenteile zu prüfen. Dies ermöglicht es, die Modelle gegebenenfalls zu erweitern, zu ändern oder zu verbessern, um eine breitere Palette von Maschinentypen abzudecken.

Die Erweiterbarkeit der Modelle für andere Domänen ist ein weiterer wichtiger Aspekt. Die Flexibilität des Ansatzes ermöglicht es, die technischen und sicherheitsbezogenen Metamodelle an die spezifischen Anforderungen und Bedürfnisse verschiedener Domänen anzupassen oder zu ergänzen. Dies eröffnet Möglichkeiten für die Anwendung in der Automobilindustrie, Medizintechnik und vielen anderen Domänen, sofern die spezifischen Anforderungen wie Risikomodelle und Sicherheitsanalysen entsprechend an die Domäne angepasst werden.

Anpassung von Normen und Sicherheitsaspekten

Mit der am 29.06.2023 offiziell veröffentlichten Maschinenverordnung EU 2023/1230 wird die Maschinenrichtlinie 2006/42/EG mit einer Übergangsfrist von 42 Monaten ersetzt [299]. Durch die Aktualisierung und Neufassung als Maschinenverordnung ist Security nun als fester Bestandteil integriert. Dies wird sich voraussichtlich auf die Anpassungen von Normen durch die entsprechenden Gremien auswirken. Dies erfordert eine mögliche Anpassung und Erweiterung der Modelle, um entsprechende neue sicherheitskritische Safety- und Security-Aspekte angemessen zu berücksichtigen.

Literatur

[1] Grimsehl, A., "Die Anfänge der Normen zur funktionalen Sicherheit," *CITplus*, Jg. 25, Nr. 10, S. 64–65, 2022, doi: 10.1002/citp.202201034.

[2] Sambeth, J., "Der Seveso-Unfall," *Nachrichten aus Chemie, Technik und Laboratorium*, Jg. 30, Nr. 5, S. 367–371, 1982, doi: 10.1002/nadc.19820300504.

[3] Weber, H., *Die Software-Krise und Ihre Macher* (Springer Compass Ser). Berlin, Heidelberg: Springer Berlin / Heidelberg, 1992.

[4] Hamilton, P., *Wege aus der Softwarekrise: Verbesserungen bei der Softwareentwicklung*. Berlin, Heidelberg: Springer, 2008.

[5] *DIN EN 61508-1 (VDE 0803-1):2011-02, Funktionale Sicherheit sicherheitsbezogener elektrischer/elektronischer/programmierbarer elektronischer Systeme – Teil 1: Allgemeine Anforderungen*, 61508-1, Verein Deutscher Ingenieure und Deutsches Institut für Normung, Feb. 2011.

[6] *DIN EN 61508-2 (VDE 0803-2):2011-02, Funktionale Sicherheit sicherheitsbezogener elektrischer/elektronischer/programmierbarer elektronischer Systeme – Teil 2: Anforderungen an sicherheitsbezogene elektrische/elektronische/programmierbare elektronische Systeme*, 61508-2, Verein Deutscher Ingenieure und Deutsches Institut für Normung, Feb. 2011.

[7] *DIN EN 61508-3 (VDE 0803-3):2011-02, Funktionale Sicherheit sicherheitsbezogener elektrischer/elektronischer/programmierbarer elektronischer Systeme – Teil 3: Anforderungen an Software*, 61508-3, Verein Deutscher Ingenieure und Deutsches Institut für Normung, Feb. 2011.

[8] *DIN EN 61508-4 (VDE 0803-4):2011-02, Funktionale Sicherheit sicherheitsbezogener elektrischer/elektronischer/programmierbarer elektronischer Systeme – Teil 4: Begriffe und Abkürzungen*, 61508-4, Verein Deutscher Ingenieure und Deutsches Institut für Normung, Feb. 2011.

[9] *DIN EN 61508-5 (VDE 0803-5):2011-02, Funktionale Sicherheit sicherheitsbezogener elektrischer/elektronischer/programmierbarer elektronischer Systeme – Teil 5: Beispiele zur Ermittlung der Stufe der Sicherheitsintegrität*, 61508-5, Verein Deutscher Ingenieure und Deutsches Institut für Normung, Feb. 2011.

Model-Based Systems Engineering sicherheitskritischer Anwendungen im Maschinenbau mittelständischer Unternehmen © 2024 by Nick Berezowski is licensed under CC BY 4.0

[10] *DIN EN 61508-6 (VDE 0803-6):2011-02, Funktionale Sicherheit sicherheitsbezogener elektrischer/elektronischer/programmierbarer elektronischer Systeme – Teil 6: Anwendungsrichtlinie für IEC 61508-2 und IEC 61508-3*, 61508-6, Verein Deutscher Ingenieure und Deutsches Institut für Normung, Feb. 2011.

[11] *DIN EN 61508-7 (VDE 0803-7):2011-02, Funktionale Sicherheit sicherheitsbezogener elektrischer/elektronischer/programmierbarer elektronischer Systeme – Teil 7: Überblick über Verfahren und Maßnahmen*, 61508-7, Verein Deutscher Ingenieure und Deutsches Institut für Normung, Feb. 2011.

[12] Leveson, N. G. und Thomas, J. P., "Certification of Safety-Critical Systems," *Commun. ACM*, Jg. 66, Nr. 10, S. 22–26, 2023. doi: 10.1145/3615860. [Online]. Verfügbar unter: https://doi.org/10.1145/3615860

[13] Rierson, L., *Developing Safety-Critical Software: A Practical Guide for Aviation Software and DO-178C Compliance*. Boca Raton: CRC Press, 2017. Zugriff am: 16. September 2019. [Online]. Verfügbar unter: https://www.taylorfrancis.com/books/9781315218168

[14] *ISO 26262-2:2018-12 Straßenfahrzeuge - Funktionale Sicherheit - Teil 2: Management der Funktionalen Sicherheit*, 26262-2, Deutsches Institut für Normung, Dez. 2018. [Online]. Verfügbar unter: https://www.beuth.de/de/norm/iso-26262-2/300423923

[15] *DIN EN 61511-1:2019-02, Funktionale Sicherheit - PLT-Sicherheitseinrichtungen für die Prozessindustrie - Teil 1: Allgemeines, Begriffe, Anforderungen an Systeme, Hardware und Anwendungsprogrammierung*, 61511-1, Deutsches Institut für Normung, Berlin, Feb. 2019. [Online]. Verfügbar unter: https://www.beuth.de/de/norm/din-en-61511-1/296008238

[16] *DIN EN 60601-1 VDE 0750-1:2013-12 Medizinische elektrische Geräte - Teil 1: Allgemeine Festlegungen für die Sicherheit einschließlich der wesentlichen Leistungsmerkmale (IEC 60601-1:2005 + Cor. :2006 + Cor. :2007 + A1:2012): Deutsche Fassung EN 60601-1:2006 + Cor. :2010 + A1:2013*, 60601-1, Verein Deutscher Ingenieure und Deutsches Institut für Normung, Dez. 2013. [Online]. Verfügbar unter: https://www.beuth.de/de/norm/din-en-60601-1/193923032

[17] *DIN EN ISO 13849-2:2013-02, Sicherheit von Maschinen_- Sicherheitsbezogene Teile von Steuerungen_- Teil_2: Validierung (ISO_13849-2:2012); Deutsche Fassung EN_ISO_13849-2:2012*, 13849-2, Deutsches Institut für Normung, Berlin.

[18] Hoffmann, A., Hoffmann, H. und Leimeister, J. M., "Anforderungen an Software Requirement Pattern in der Entwicklung sozio-technischer Systeme," *1617-5468*, 2012. [Online]. Verfügbar unter: https://dspace.gi.de/handle/20.500.12116/17843

[19] Born, M., Holz, E. und Kath, O., *Softwareentwicklung mit UML 2: Die "neuen" Entwurfstechniken UML 2, MOF 2 und MDA*, 2004. Aufl. (Programmer's choice). München: Addison-Wesley, 2005.

[20] Poole, J. D., "Model-driven architecture: Vision, standards and emerging technologies," in *Workshop on Metamodeling and Adaptive Object Models, ECOOP*, Bd. 50, 2001. [Online]. Verfügbar unter: https://masters.donntu.ru/2010/fknt/frunt/library/elarticle5.pdf

[21] Estefan, J. A., "Survey of model-based systems engineering (MBSE) methodologies," *Incose MBSE Focus Group*, Jg. 25, Nr. 8, S. 1–12, 2007. [Online]. Verfügbar unter: https://edisciplinas.usp.br/enrol/index.php?id=72954

[22] Object Management Group, "OMG Systems Modeling Language (OMG SysML™), V1.0: OMG Available Specification," Sep. 2007. [Online]. Verfügbar unter: https://www.omg.org/spec/SysML/1.0/PDF

[23] Feiler, P., Gluch, David und Hudak, J., "The Architecture Analysis & Design Language (AADL): An Introduction," Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA, Rep. CMU/SEI-2006-TN-011, 2006. [Online]. Verfügbar unter: https://22132398.fs1.hubspotusercontent-na1.net/hubfs/22132398/LML%20specification%201.3%20(1).pdf

[24] Dam, S., "Lifecycle Modeling Language (LML) SPECIFICATION," [Online]. Verfügbar unter: http://www.lifecyclemodeling.org/spec/1.3

[25] Object Management Group, *Unified Modeling Language®(OMG UML®) Version 2.5.1*, 2017. Zugriff am: 10. Februar 2023. [Online]. Verfügbar unter: https://www.omg.org/spec/UML/2.5.1/About-UML/#:~:text=A%20specification%20defining%20a%20graphical%20language%20for%20visualizing%2C,and%20documenting%20the%20artifacts%20of%20distributed%20object%20systems.

[26] Object Management Group, "OMG Systems Modeling Language (OMG SysML™): Version 1.6," 2019. [Online]. Verfügbar unter: https://www.omg.org/spec/SysML/1.6/

[27] Friedenthal, S., Dori, D. und Mordecai, Y. "Modeling Standards - SEBoK." Zugriff am: 23. Dezember 2020. [Online.] Verfügbar: https://www.sebokwiki.org/wiki/Modeling_Standards

[28] D'Ambrosio, J. und Soremekun, G., "Systems engineering challenges and MBSE opportunities for automotive system design," in *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Banff, AB, Okt. 2017 - Okt. 2017, S. 2075–2080, doi: 10.1109/SMC.2017.8122925.

[29] Cimatti, A. *et al.*, "NuSMV 2: An OpenSource Tool for Symbolic Model Checking," in *Computer Aided Verification* (Lecture Notes in Computer Science 2404), G. Goos, J. Hartmanis, J. van Leeuwen, E. Brinksma und K. G. Larsen, Hg., Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, S. 359–364. [Online]. Verfügbar unter: http://e-prints.biblio.unitn.it/85/1/16.pdf

[30] Batteux, M., Prosvirnova, T. und Rauzy, A. B., "AltaRica 3.0 in ten modelling patterns," *IJCCBS*, Jg. 9, 1/2, 2019, Art. Nr. 98809, doi: 10.1504/IJCCBS.2019.098809.

[31] Papadopoulos, Y. "HiP-HOPS." Zugriff am: 17. Mai 2024.488Z. [Online.] Verfügbar: https://hip-hops.co.uk/

[32] Brahmi, R., Hammadi, M., Choley, J.-Y., Trigui, M. und Aifaoui, N., "A SysML profile for mechanical assembly," in *2020 IEEE International Systems Conference (SysCon)*, 2020, doi: 10.1109/syscon47679.2020.9275865.

[33] Drave, I. *et al.*, "Modeling mechanical functional architectures in SysML," in *Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems*, New York, NY, USA, 2020, doi: 10.1145/3365438.3410938.

[34] *DIN EN ISO 13849-1:2016-06, Sicherheit von Maschinen_- Sicherheitsbezogene Teile von Steuerungen_- Teil_1: Allgemeine Gestaltungsleitsätze (ISO_13849-1:2015); Deutsche Fassung EN_ISO_13849-1:2015*, 13849-1, Deutsches Institut für Normung, Berlin.

[35] *DIN EN 61131-3:2014-06, Speicherprogrammierbare Steuerungen_- Teil_3: Programmiersprachen (IEC_61131-3:2013); Deutsche Fassung EN_61131-3:2013*, 61131-3, Deutsches Institut für Normung, Berlin.

[36] Vogel-Heuser, B., "Usability Experiments to Evaluate UML/SysML-Based Model Driven Software Engineering Notations for Logic Control in Manufacturing Automation," *Journal of Software Engineering and Applications*, Jg. 07, 2014. doi: 10.4236/jsea.2014.711084. [Online]. Verfügbar unter: https://www.scirp.org/html/5-9301970_51054.htm

[37] Tapp, C., "An Introduction to MISRA C++," *SAE Int. J. Passeng. Cars – Electron. Electr. Syst.*, Jg. 1, Nr. 1, S. 265–268, 2008. doi: 10.4271/2008-01-0664. [Online]. Verfügbar unter: http://dx.doi.org/10.4271/2008-01-0664

[38] Huelke, M., Becker, N. und Eggeling, M., *Sicherheitsbezogene Anwendungssoftware von Maschinen – Die Matrixmethode des IFA*. DGUV/IFA, 2016.

[39] Bennett, M. A., McDermott, R. und Beauregard, M., *The Basics of FMEA*. Productivity Press, 2017.

[40] Mhenni, F., Choley, J.-Y., Riviere, A., Nguyen, N. und Kadima, H., "SysML and safety analysis for mechatronic systems," in *2012 9th France-Japan & 7th Europe-Asia Congress on Mechatronics (MECATRONICS) / 13th Int'l Workshop on Research and Education in Mechatronics (REM)*, 2012, doi: 10.1109/mecatronics.2012.6451042.

[41] Cancila, D. *et al.*, "Sophia: a modeling language for model-based safety engineering," in *Proc. 2nd Int. Workshop on Model Based Architecting and Construction of Embedded Systems*, 2009, S. 11–26. [Online]. Verfügbar unter: https://ceur-ws.org/Vol-507/ACES-MB2009_Proceedings.pdf#page=11

[42] Jiang, Y., Bai, N., Yang, H., Zhang, H., Wang, Z. und Liu, X., "MBSE-based functional hazard assessment of civil aircraft braking system," in *2020 5th International Conference on Mechanical, Control and Computer Engineering (ICMCCE)*, 2020, S. 460–464, doi: 10.1109/ICMCCE51767.2020.00107.

[43] Mordecai, Y., Orhof, O. und Dori, D., "Model-based interoperability engineering in systems-of-systems and civil aviation," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, Jg. 48, Nr. 4, S. 637–648, 2016. [Online]. Verfügbar unter: https://www.researchgate.net/profile/Yaniv-Mordecai/publication/307974034_Model-Based_Interoperability_Engineering_in_Systems-of-Systems_and_Civil_Aviation/links/5a0026170f7e9b9968c784ac/Model-Based-Interoperability-Engineering-in-Systems-of-Systems-and-Civil-Aviation.pdf

[44] Clegg, K., Li, M., Stamp, D., Grigg, A. und McDermid, J., "Integrating existing safety analyses into SysML," in *Model-Based Safety and Assessment: 6th International Symposium, IMBSA 2019, Thessaloniki, Greece, October 16-18, 2019, Proceedings 6*, 2019, S. 63–77, doi: 10.1007/978-3-030-32872-6_5.

[45] Clegg, K., Li, M., Stamp, D., Grigg, A. und McDermid, J., "A SysML Profile for Fault Trees—Linking Safety Models to System Design," in *Computer Safety, Reliability, and Security: 38th International Conference, SAFECOMP 2019, Turku, Finland, September 11-13, 2019, Proceedings 38*, Bd. 11698, 2019, S. 85–93, doi: 10.1007/978-3-030-26601-1_6.

[46] Clegg, K., Stamp, D. und McDermid, J., "Binding Fault Logic to System Design: a SysML approach," in *Proceedings of the 31st European Safety and Reliability Conference, Research Publishing (S) Pte Ltd*, 2021, S. 880–887, doi: 10.3850/981-973-0000-00-0.

[47] Liu, H., Liu, J., Yin, W., Sun, H. und Yang, C., "Safety SysML: An Executable Safety-Critical Avionics Requirement Modeling Language," in *2022 IEEE 22nd International Conference on Software Quality, Reliability and Security (QRS)*, 2022, S. 388–399, doi: 10.1109/QRS57517.2022.00047.

[48] Biggs, G., Post, K., Armonas, A., Yakymets, N., Juknevicius, T. und Berres, A., "OMG standard for integrating safety and reliability analysis into MBSE: Concepts and applications," in *INCOSE International Symposium*, Bd. 29, 2019, S. 159–173.

[49] Berres, A., Post, K., Armonas, A., Hecht, M., Juknevičius, T. und Banham, D., "OMG RAAML standard for model-based Fault Tree Analysis," *INCOSE International Symposium*, Jg. 31, Nr. 1, S. 1349–1362, 2021, doi: 10.1002/j.2334-5837.2021.00905.x.

[50] Diatte, K., O'Halloran, B. und van Bossuyt, D. L., "The Integration of Reliability, Availability, and Maintainability into Model-Based Systems Engineering," *Systems*, Jg. 10, Nr. 4, S. 101, 2022. doi: 10.3390/systems10040101. [Online]. Verfügbar unter: https://www.mdpi.com/1727110

[51] Ahlbrecht, A. und Durak, U., "Integrating Safety into MBSE Processes with Formal Methods," in *2021 IEEE/AIAA 40th Digital Avionics Systems Conference (DASC)*, San Antonio, TX, USA, Okt. 2021 - Okt. 2021, S. 1–9, doi: 10.1109/DASC52595.2021.9594315.

[52] Dandashi, F., "Modeling Security Views with Unified Architecture Framework, Risk Assessment and Analysis Modeling Language, and Systems Modeling Language," MITRE Technical Report MTR220019, 2022. [Online]. Verfügbar unter: https://apps.dtic.mil/sti/pdfs/ad1173412.pdf

[53] Yakymets, N., Ionescu, M. A. und Atienza Alonso, D., "Metamodel for Safety Risk Management of Medical Devices Based on ISO 14971," in *The ACM/IEEE 26th International Conference on Model-Driven Engineering Languages and Systems*, 2023. [Online]. Verfügbar unter: https://infoscience.epfl.ch/server/api/core/bitstreams/3cdb9eef-2fd0-4a5d-bd66-de0a55b9aad9/content

[54] Rauzy, A. B. und Haskins, C., "Foundations for model-based systems engineering and model-based safety assessment," *Systems Engineering*, Jg. 22, Nr. 2, S. 146–155, 2019. doi: 10.1002/sys.21469. [Online]. Verfügbar unter: https://doi.org/10.1002/sys.21469

[55] Woschke, T., Haase, H. und Kratzer, J., "Resource scarcity in SMEs: effects on incremental and radical innovations," *Management Research Review*, Jg. 40, Nr. 2, S. 195–217. doi: 10.1108/MRR-10-2015-0239. [Online]. Verfügbar unter: https://doi.org/10.1108/MRR-10-2015-0239

[56] Muller, P., Gagliardi, D., Caliandro, C., Bohn, N. U. und Klitou, D., "Annual Report on European SMEs 2013/2014--A Partial and Fragile Recovery Final Report-July 2014 SME Performance Review 2013/2014," *Annu. Rep. Eur. SMEs SME Perform. Rev.*, 2014. [Online]. Verfügbar unter: https://www.researchgate.net/publication/266384367_Annual_Report_on_European_SMEs_20132014_-_A_Partial_and_Fragile_Recovery_Final_Report_-July_2014_SME_Performance_Review_20132014_for_the_European_Commission

[57] Lindner, D., Ott, M. und Leyh, C., "Der digitale Arbeitsplatz – KMU zwischen Tradition und Wandel," *HMD*, Jg. 54, Nr. 6, S. 900–916, 2017. doi: 10.1365/s40702-017-0370-x. [Online]. Verfügbar unter: https://link.springer.com/article/10.1365/s40702-017-0370-x

[58] Chapurlat, V. und Nastov, B., "Deploying MBSE in SME context: revisiting and equipping Digital Mock-Up," *2020 IEEE International Symposium on Systems Engineering (ISSE)*, S. 1–8, 2020, doi: 10.1109/ISSE49799.2020.9272230.

[59] Kamaruddin, N., Safiyah, R. D. und Wahab, A., "Small and medium enterprise business solutions using data visualization," *Bulletin of Electrical Engineering and Informatics*, Jg. 9, Nr. 6, S. 2562–2568, 2020. doi: 10.11591/eei.v9i6.2463. [Online]. Verfügbar unter: https://doi.org/10.11591/eei.v9i6.2463

[60] Hilderman, V. und Baghi, T., *Avionics certification: a complete guide to DO-178 (software), DO-254 (hardware)*. Leesburg, VA: Avionics Communications, 2007.

[61] Kääriäinen, J., Saari, L., Tihinen, M. und Perätalo, S., "Supporting the digital transformation of SMEs—trained digital evangelists facilitating the positioning phase," *International journal of information systems and project management*, Jg. 11, Nr. 1, S. 5–27, 2023. [Online]. Verfügbar unter: https://aisel.aisnet.org/ijispm/vol11/iss1/2

[62] Parente, R., Feola, R. und Celenta, R., "Leveraging collaboration between academic research and SMEs to support digital transformation in the agri-food Italian industry: the case of Santomiele," in *Cases on Digital Entrepreneurship*, Edward Elgar Publishing, 2023, S. 58–74.

[63] ProNES Automation GmbH. "Digitalisierung nach Maß | ProNES." Zugriff am: 29. März 2023.438Z. [Online.] Verfügbar: https://www.prones.de/

[64] ProNES Automation GmbH. "ProNES | batterieinspektor." Zugriff am: 29. März 2023.438Z. [Online.] Verfügbar: https://www.batterieinspektor.de/

[65] Weyrich, M. und Ebert, C., "Reference Architectures for the Internet of Things," *IEEE Softw.*, Jg. 33, Nr. 1, S. 112–116, 2015. doi: 10.1109/MS.2016.20. [Online]. Verfügbar unter: https://www.researchgate.net/publication/288855901_Reference_Architectures_for_the_Internet_of_Things

[66] Kagermann, H., Lukas, W.-D. und Wahlster, W., "Industrie 4.0: Mit dem Internet der Dinge auf dem Weg zur 4. industriellen Revolution," *VDI nachrichten*, Jg. 13, Nr. 1, S. 2–3, 2011. [Online]. Verfügbar unter: https://www.dfki.de/fileadmin/user_upload/DFKI/Medien/News_Media/Presse/Presse-Highlights/vdinach2011a13-ind4.0-Internet-Dinge.pdf

[67] Bursać, N., "Model Based Systems Engineering zur Unterstützung der Baukastenentwicklung im Kontext der Frühen Phase der Produktgenerationsentwicklung," IPEK, Institut für Produktentwicklung, 2016. [Online]. Verfügbar unter: https://www.researchgate.net/profile/nikola-bursac/publication/303824514_model_based_systems_engineering_zur_unterstutzung_der_baukastenentwicklung_im_kontext_der_fruhen_phase_der_produktgenerationsentwicklung/links/57568fcc08ae0405a57592a3/model-based-systems-engineering-zur-unterstuetzung-der-baukastenentwicklung-im-kontext-der-fruehen-phase-der-produktgenerationsentwicklung.pdf

[68] Mhenni, F., Nguyen, N. und Choley, J.-Y., "Automatic fault tree generation from SysML system models," in *2014 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, Besacon, Jul. 2014 - Jul. 2014, S. 715–720, doi: 10.1109/AIM.2014.6878163.

[69] Hecht, M., Dimpfl, E. und Pinchak, J., "Using SysML to Automatically Generate of Failure Modes and Effects Analyses," *INCOSE International Symposium*, Jg. 25, Nr. 1, S. 1357–1372, 2015, doi: 10.1002/j.2334-5837.2015.00135.x.

[70] Yakymets, N., Sango, M., Dhouib, S. und Gelin, R., "Model-Based Engineering, Safety Analysis and Risk Assessment for Personal Care Robots," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Madrid, Okt. 2018 - Okt. 2018, S. 6136–6141, doi: 10.1109/IROS.2018.8594115.

[71] Baklouti, A., Nguyen, N., Choley, J.-Y., Mhenni, F. und Mlika, A., "Free and open source fault tree analysis tools survey," in *2017 Annual IEEE International Systems Conference (SysCon)*, 2017, doi: 10.1109/syscon.2017.7934794.

[72] Lisagor, O., Kelly, T. und Niu, R., "Model-based safety assessment: Review of the discipline and its challenges," in *The Proceedings of 2011 9th International Conference on Reliability, Maintainability and Safety*, 2011, doi: 10.1109/icrms.2011.5979344.

[73] Favaro, J., Mazzini, S., Popov, P. T. und Strigini, L., "AQUAS: A project to bridge the gaps between safety and security processes," *Ada User Journal*, Jg. 39, Nr. 4, S. 261–263, 2018. [Online]. Verfügbar unter: https://openaccess.city.ac.uk/id/eprint/22211/

[74] Nguyen, N., Mhenni, F. und Choley, J.-Y., "Redundancy handling with model-based systems engineering," in *26th European Safety and Reliability Conference (ESREL)*, 2016. [Online]. Verfügbar unter: https://www.researchgate.net/profile/nga-nguyen/publication/305412045_redundancy_handling_with_model-based_systems_engineering/links/581a070308ae3c82664c11e3/redundancy-handling-with-model-based-systems-engineering.pdf

[75] Helle, P., "Automatic SysML-based safety analysis," in *Proceedings of the 5th International Workshop on Model Based Architecting and Construction of Embedded Systems - ACES-MB '12*, Innsbruck, Austria, I. Ober, Hg., 2012, S. 19–24, doi: 10.1145/2432631.2432635.

[76] Thramboulidis, K. und Scholz, S., "Integrating the 3+1 SysML view model with safety engineering," in *2010 IEEE 15th Conference on Emerging Technologies & Factory Automation (ETFA 2010)*, Bilbao, Sep. 2010 - Sep. 2010, S. 1–8, doi: 10.1109/ETFA.2010.5641353.

[77] Biggs, G., Sakamoto, T. und Kotoku, T., "A profile and tool for modelling safety information with design information in SysML," *Software & Systems Modeling*, Jg. 15, Nr. 1, S. 147–178, 2016. doi: 10.1007/s10270-014-0400-x. [Online]. Verfügbar unter: https://doi.org/10.1007/s10270-014-0400-x

[78] Schnellbach, A., *Fail-operational automotive systems: Doctoral Thesis*. Südwestdeutscher Verlag für Hochschulschriften, 2018.

[79] Menager, N., "Effizienzsteigerung bei Auslegung und Inbetriebnahme Mechatronischer Systeme Durch Verwendung Modellbasierter Entwicklungsmethoden auf Basis Offener Standards: Dissertation," Dissertation, Duisburg, Essen, Universität Duisburg-Essen, 2017, 2017. [Online]. Verfügbar unter: https://eur-lex.europa.eu/legal-content/DE/TXT/PDF/?uri=CELEX:32023R1230

[80] Choueiri, E. Y., "A Critical History of Electric Propulsion: The First 50 Years (1906-1956)," *Journal of Propulsion and Power*, Jg. 20, Nr. 2, S. 193–203, 2004, doi: 10.2514/1.9245.

[81] Hu, Y., Wang, T., Chen, T., Song, N., Yao, K. und Luo, Y., "Design and implementation of testing system of LED driver power based on LabVIEW," *Optik*, Jg. 200, S. 163411, 2020, doi: 10.1016/j.ijleo.2019.163411.

[82] Xie, Y. B., You, D. H. und Huang, S. Y., "A LabVIEW-based testing system for the rogowski optical current transformer," *Automation of Electric Power Systems*, Nr. 28, 2004, Art. Nr. 22. [Online]. Verfügbar unter: https://en.cnki.com.cn/article_en/cjfdtotal-dlxt200422022.htm

[83] Parish, D. J. und Reeves-Hardcastle, P., "Rapid prototyping using the LabVIEW environment," in *Conference Record. AUTOTESTCON'96*, 1996, S. 235–238, doi: 10.1109/autest.1996.547703.

[84] Reeves-Hardcastle, P. und Parish, D. J., "Using LabVIEW to provide rapid prototyping environment for the teaching of avionics systems," *IEEE Aerosp. Electron. Syst. Mag.*, Jg. 10, Nr. 12, S. 9, 1995. doi: 10.1109/62.480824. [Online]. Verfügbar unter: http://dx.doi.org/10.1109/62.480824

[85] Haquin, C. *et al.*, "Development of a Safety Classified System with LabVIEW and EPICS," in *Proc. 16th Int. Conf. on Accelerator and Large Experimental Control Systems (ICALEPCS'17)*, 2018, S. 1221–1223, doi: 10.18429/JACoW-ICALEPCS2017-THCPA04.

[86] *ISO 13849-1:2015 - Safety of machinery — Safety-related parts of control systems — Part 1: General principles for design*, 13849-1, ISO, 2015.

[87] Obermeier, M., Braun, S. und Vogel-Heuser, B., "A Model-Driven Approach on Object-Oriented PLC Programming for Manufacturing Systems with Regard to Usability," *IEEE Trans. Ind. Inf.*, Jg. 11, Nr. 3, S. 790–800, 2015. doi: 10.1109/tii.2014.2346133. [Online]. Verfügbar unter: http://dx.doi.org/10.1109/tii.2014.2346133

[88] Cederbladh, J., Cicchetti, A. und Suryadevara, J., "Early validation and verification of system behaviour in model-based systems engineering: a systematic literature review," *ACM Transactions on Software Engineering and Methodology*, Jg. 33, Nr. 3, S. 1–67, 2024. doi: 10.1145/3631976. [Online]. Verfügbar unter: https://doi.org/10.1145/3631976

[89] Mhenni, F., Nguyen, N. und Choley, J.-Y., "SafeSysE: A safety analysis integration in systems engineering approach," *IEEE Systems Journal*, Jg. 12, Nr. 1, S. 161–172, 2016. doi: 10.1109/JSYST.2016.2547460. [Online]. Verfügbar unter: https://ieeexplore.ieee.org/document/7458116

[90] Sträter, O., *Cognition and Safety : An Integrated Approach to Systems Design and Assessment*. Routledge, 2016. [Online]. Verfügbar unter: https://www.taylorfrancis.com/books/mono/10.4324/9781315260006/cognition-safety-oliver-str%c3%a4ter

[91] No Magic. "Magic Draw." Zugriff am: 29. März 2023.623Z. [Online.] Verfügbar: https://docs.nomagic.com/display/MD190/19.0+LTR+Version+News

[92] Modeliosoft. "Modelio SysML Architect." Zugriff am: 29. März 2023.324Z. [Online.] Verfügbar: https://www.modeliosoft.com/en/modules/sysml-architect.html

[93] Eclipse. "Eclipse Papyrus: Modeling environment." Zugriff am: 13. Februar 2023.401Z. [Online.] Verfügbar: https://www.eclipse.org/papyrus/

[94] Batteux, M., Prosvirnova, T., Rauzy, A. und Kloul, L., "The AltaRica 3.0 project for model-based safety assessment," in *2013 11th IEEE International Conference on Industrial Informatics (INDIN)*, 2013, doi: 10.1109/indin.2013.6622976.

[95] Li, Y., Gong, Q. und Su, D., "Model-based System Safety Assessment of Aircraft Power Plant," *Procedia Engineering*, Jg. 80, S. 85–92, 2014. doi: 10.1016/j.proeng.2014.09.063. [Online]. Verfügbar unter: https://www.sciencedirect.com/science/article/pii/s1877705814011564

[96] Krishnan, R. und Bhada, S. V., "An Integrated System Design and Safety Framework for Model-Based Safety Analysis," *IEEE Access*, Jg. 8, S. 146483–146497, 2020, doi: 10.1109/ACCESS.2020.3015151.

[97] Biggs, G., Juknevicius, T., Armonas, A. und Post, K., "Integrating Safety and Reliability Analysis into MBSE: overview of the new proposed OMG standard," *INCOSE International Symposium*, Jg. 28, Nr. 1, S. 1322–1336, 2018, doi: 10.1002/j.2334-5837.2018.00551.x.

[98] Boehm, B., Valerdi, Ricardo und Honour, E., "The ROI of systems engineering: Some quantitative results for software-intensive systems," *Systems Engineering*, Jg. 11, Nr. 3, S. 221–234, 2008, doi: 10.1002/sys.20096.

[99] D.K. Hitchins, "Systems engineering: in search of the elusive optimum," *Engineering Management Journal*, Jg. 8, Nr. 4, S. 195–207, 1998. doi: 10.1049/em:19980412. [Online]. Verfügbar unter: https://digital-library.theiet.org/content/journals/10.1049/em_19980412

[100] Colette Darcy, Jimmy Hill, TJ McCabe und Philip McGovern, "A consideration of organisational sustainability in the SME context: A resource-based view and composite model," *European Journal of Training and Development*, Jg. 38, Nr. 5, S. 398–414. doi:

10.1108/EJTD-10-2013-0108. [Online]. Verfügbar unter: https://www.emerald.com/insight/content/doi/10.1108/ejtd-10-2013-0108/full/pdf

[101] Laforet, S. und Tann, J., "Innovative characteristics of small manufacturing firms," *Journal of Small Business and Enterprise Development*, Jg. 13, Nr. 3, S. 363–380. doi: 10.1108/14626000610680253. [Online]. Verfügbar unter: https://www.dguv.de/medien/ifa/de/pra/softwa/sistema/kochbuch/sistema_kochbuch1_de.pdf

[102] Hausman, A., "Innovativeness among small businesses: Theory and propositions for future research," *Industrial Marketing Management*, Jg. 34, Nr. 8, S. 773–782, 2005. doi: 10.1016/j.indmarman.2004.12.009. [Online]. Verfügbar unter: https://www.sciencedirect.com/science/article/pii/s0019850105000039

[103] Stade, M. J. C., Reckin, R., Brandenburg, S. und Thüring, M., "Usability in KMU etablieren: Von schneller Problemlösung zu ressourcenorientiertem Usability Engineering," in *Mensch & Computer 2013 - Workshopband*, OLDENBOURG WISSENSCHAFTSVERLAG, 2013, S. 19–28. [Online]. Verfügbar unter: http://dx.doi.org/10.1524/9783486781236.19

[104] Garengo, P., Biazzo, S. und Bititci, U. S., "Performance measurement systems in SMEs: A review for a research agenda," *International Journal of Management Reviews*, Jg. 7, Nr. 1, S. 25–47, 2005, doi: 10.1111/j.1468-2370.2005.00105.x.

[105] Gal, A. R., Rat, C. L. und Toadere, C. I., "Barriers to the Implementation of the Quality Management System in Small and Medium-Sized Enterprises," *Ovidius University Annals, Economic Sciences Series, Ovidius University of Constantza, Faculty of Economic Sciences XX (1), August*, S. 618–625, 2020. [Online]. Verfügbar unter: https://stec.univ-ovidius.ro/html/anale/RO/2020/Section%204/10.pdf

[106] Keizer, J. A., Dijkstra, L. und Halman, J. I. M., "Explaining innovative efforts of SMEs.: An exploratory survey among SMEs in the mechanical and electrical engineering sector in The Netherlands," *Technovation*, Jg. 22, Nr. 1, S. 1–13, 2002. doi: 10.1016/S0166-4972(00)00091-2. [Online]. Verfügbar unter: https://www.sciencedirect.com/science/article/pii/s0166497200000912

[107] Pitsch, S. und Sommerhoff, B., "Erstellung einer unternehmensindividuellen Roadmap für die Industrie 4.0 in KMU," *Heutige und zukünftig Herausforderungen an die Qualitätswissenschaften in Forschung und Praxis, Bericht zur GQW-Jahrestagung*, S. 209–234, 2017. [Online]. Verfügbar unter: https://opus4.kobv.de/opus4-fau/files/8328/gqw+tagungsband_2017_opus.pdf#page=219

[108] Nouar, D. und Popovics, P., "Use of methods and tools for an effective small and medium-sized enterprise in Szabolcs-Szatmár-Bereg county in Hungary," *Applied Studies in Agribusiness and Commerce*, Jg. 16, Nr. 2, 2022. [Online]. Verfügbar unter: https://doi.org/10.19041/APSTRACT/2022/2/1

[109] Parrott, E. L., "A PPE Use Case on Configuration Management Approach for MBSE," in *2023 IEEE Aerospace Conference*, 2023, S. 1–8, doi: 10.1109/AERO55745.2023.10116017.

[110] Kießling, S., Möser, S., Eisenträger, M. und Skrytutskyi, S., "FLUCH UND SEGEN VON INDUSTRIE 4.0–EIN EINDRUCK AUS DEM SONDERMASCHINENBAU," *9. FACHTAGUNG ANLAGENBAU DER ZUKUNFT*, S. 47. [Online]. Verfügbar unter: https://www.iff.fraunhofer.de/content/dam/iff/de/dokumente/publikationen/anlagenbau-der-zukunft-2016-tagungsband-fraunhofer-iff.pdf#page=49

[111] Berezowski, N. und Haid, M., "MBSE for SMEs with Domain-Specific Safety Analyses and Loose Tool Coupling," in *Real-Time Meeting of the Gesellschaft für Informatik*, 2022, S. 72–81.

[112] Nunamaker, J. F., Chen, M., JR. und Purdin, T. D. M., "Systems Development in Information Systems Research," *Journal of Management Information Systems*, doi: 10.1080/07421222.1990.11517898.

[113] Armoush, A., "Towards the integration of security and safety patterns in the design of safety-critical embedded systems," in *2022 4th International Conference on Applied Automation and Industrial Diagnostics (ICAAID)*, Bd. 1, 2022, S. 1–6, doi: 10.1109/ICAAID51067.2022.9799513.

[114] Salehi Fathabadi, A., Snook, C., Dghaym, D., Hoang, T. S., Alotaibi, F. und Butler, M., "Designing Critical Systems Using Hierarchical STPA and Event-B," in *International Conference on Rigorous State-Based Methods*, 2023, S. 220–237, doi: 10.1007/978-3-031-33163-3_17.

[115] Agbo, C. und Mehrpouyan, H., "Conflict analysis and resolution of safety and security boundary conditions for industrial control systems," in *2022 6th International Conference on System Reliability and Safety (ICSRS)*, 2022, S. 145–156, doi: 10.1109/ICSRS56243.2022.10067393.

[116] Jung, R., "Ohne Security keine Safety," *ATZextra*, Jg. 24, Nr. 1, S. 32–35, 2019. doi: 10.1007/s35778-019-0006-6. [Online]. Verfügbar unter: https://doi.org/10.1007/s35778-019-0006-6

[117] Norman, D. A. und Draper, S. W., *User centered system design: New perspectives on human-computer interaction*, 1986. [Online]. Verfügbar unter: https://dl.acm.org/doi/10.1145/3686169.3686184

[118] Object Management Group, *OMG Unified Modeling Language (OMG UML), Superstructure, v2. 1.2*, 2007. [Online]. Verfügbar unter: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=eb2f430d1fd600cb5adeaf1a64f95adf8a8e7198

[119] Colomb, R. *et al.*, "The Object Management Group Ontology Definition Metamodel," in *Ontologies for software engineering and software technology*, C. Calero, F. Ruiz und M. Piattini, Hg., Berlin: Springer, 2006, S. 217–247. [Online]. Verfügbar unter: http://dx.doi.org/10.1007/3-540-34518-3_8

[120] Wymore, A. W., *Model-based systems engineering: An introduction to the mathematical theory of discrete systems and to the tricotyledon theory of system design* (Systems engineering series). Boca Raton, Fla.: CRC Press, 1993. Zugriff am: 10. Januar 2023. [Online]. Verfügbar unter: https://www.tandfonline.com/doi/abs/10.1080/03081079608945147?journalCode=ggen20

[121] Dickerson, C. E. und Mavris, D., "A Brief History of Models and Model Based Systems Engineering and the Case for Relational Orientation," *IEEE Systems Journal*, Jg. 7, Nr. 4, S. 581–592, 2013. doi: 10.1109/jsyst.2013.2253034. [Online]. Verfügbar unter: http://dx.doi.org/10.1109/jsyst.2013.2253034

[122] Object Management Group, "OMG Meta Object Facility (MOF) Core Specification: Version 2.5.1," 2019. [Online]. Verfügbar unter: https://www.omg.org/spec/MOF/2.5.1

[123] Cabot, J. und Gogolla, M., "Object Constraint Language (OCL): A Definitive Guide," in *Formal Methods for Model-Driven Engineering* (Lecture Notes in Computer Science), M. Bernardo, V. Cortellessa und A. Pierantonio, Hg., Berlin, Heidelberg: Springer Nature, 2012, S. 58–90. [Online]. Verfügbar unter: https://doi.org/10.1007/978-3-642-30982-3_3

[124] "MDT/Papyrus/UserGuide/CSS - Eclipsepedia." Zugriff am: 8. Mai 2023.593Z. [Online.] Verfügbar: https://wiki.eclipse.org/MDT/Papyrus/UserGuide/CSS

[125] Object Management Group, *XML Metadata interchange (XMI). Version 2.5. 1*, 2015. Zugriff am: 10. Februar 2023. [Online]. Verfügbar unter: https://www.omg.org/spec/XMI/2.5.1

[126] Stahl, T., Völter, M. und Czarnecki, K., *Model-driven software development: technology, engineering, management*. John Wiley & Sons, Inc., 2006.

[127] Beckmann, K. und Thoss, M., "A model-driven software development approach using OMG DDS for wireless sensor networks," in *Software Technologies for Embedded and Ubiquitous Systems: 8th IFIP WG 10.2 International Workshop, SEUS 2010, Waidhofen/Ybbs, Austria, October 13-15, 2010. Proceedings 8*, 2010, S. 95–106, doi: 10.1007/978-3-642-16256-5_11.

[128] Friedenthal, S., Moore, A. und Steiner, R., *A Practical Guide to SysML: The Systems Modeling Language*. Morgan Kaufmann, 2014.

[129] Kaslow, D., Ayres, B., Cahill, P. T., Hart, L. und Yntema, R., "A Model-Based Systems Engineering (MBSE) approach for defining the behaviors of CubeSats," in *2017 IEEE Aerospace Conference*, 2017, doi: 10.1109/aero.2017.7943865.

[130] McDermott, T. A., Hutchison, N., Clifford, M., van Aken, E., Salado, A. und Henderson, K., "Benchmarking the benefits and current maturity of model-based systems engineering across the enterprise," *Systems Engineering Research Center (SERC)*, 2020. [Online]. Verfügbar unter: https://sercuarc.org/wp-content/uploads/2020/03/SERC-SR-2020-001-Benchmarking-the-Benefits-and-Current-Maturity-of-MBSE-3-2020.pdf

[131] Darke, P. und Shanks, G., "Stakeholder viewpoints in requirements definition: A framework for understanding viewpoint development approaches," *Requirements Eng*, Jg. 1, Nr. 2, S. 88–105, 1996, doi: 10.1007/BF01235904.

[132] Ryschkewitsch, M., Schaible, D. und Larson, W., "The art and science of systems engineering," in *Systems Research Forum*, Bd. 3, 2009, S. 81–100, doi: 10.1142/S1793966609000080.

[133] Cloutier, R., "Introduction to this Special Edition on Model-based Systems Engineering," *INSIGHT*, Jg. 12, Nr. 4, S. 7–8, 2009, doi: 10.1002/inst.20091247.

[134] Holowenko, O., Oehm, L., Majschak, J.-P. und Ihlenfeldt, S., *Informationsaustausch im interdisziplinären Entwicklungsprozess*, 2018. [Online]. Verfügbar unter: http://www.maschinenrichtlinie.de/fileadmin/dokumente/2006-42-EG_maschinenrichtlinie_de.pdf

[135] Grote, K.-H. und Hefazi, H., *Springer Handbook of Mechanical Engineering*. Springer Nature, 2021.

[136] Ogli, I. M. R. und Ogli, T. A. I., "A Role of Mechanical Engineering in Mechatronics," *JournalNX*, S. 824–828, 2020. [Online]. Verfügbar unter: https://repo.journalnx.com/index.php/nx/article/view/1690

[137] Zalilov, R. *et al.*, "CAD systems and computer graphics in the training of specialists in the field of mechanical engineering," *Journal of Critical Reviews*, Jg. 7, Nr. 2, S. 162, 2020.

[138] Engelke, H.-J., *SOLIDWORKS 2021: 3D-Druck*. Norderstedt: Books on Demand GmbH, 2020. [Online]. Verfügbar unter: https://api.pageplace.de/preview/dt0400.9783753485287_a41151581/preview-9783753485287_a41151581.pdf

[139] Ridder, D., *AutoCAD 2009*. MITP-Verlags GmbH & Co. KG, 2008.

[140] T. Lerchbaumer, *Entwicklung einer Open-Source Programmierhilfe für Roboterzellen.* [Online]. Verfügbar unter: https://opus.fhv.at/files/4197/lerchbaumer-thomas_thesis.pdf

[141] Groover, M. und Zimmers, E., *CAD/CAM: Computer-Aided Design and Manufacturing*. Pearson Education, 1983.

[142] Ji, Q. und Marefat, M. M., "Machine interpretation of CAD data for manufacturing applications," *ACM Computing Surveys (CSUR)*, Jg. 29, Nr. 3, S. 264–311, 1997, doi: 10.1145/262009.262012.

[143] Tomiyama, T. und Yoshikawa, H., "Knowledge engineering and CAD," *Future Generation Computer Systems*, Jg. 1, Nr. 4, S. 237–243, 1985. doi: 10.1016/0167-739X(85)90012-3. [Online]. Verfügbar unter: https://www.sciencedirect.com/science/article/pii/0167739x85900123

[144] Henderson, K. und Salado, A., "Is CAD A Good Paradigm for MBSE?," *INCOSE International Symposium*, Jg. 31, Nr. 1, S. 144–157, 2021, doi: 10.1002/j.2334-5837.2021.00830.x.

[145] Haberhauer, H. und Bodenstein, F., *Maschinenelemente: Gestaltung, Berechnung, Anwendung*. Springer-Verlag, 2013.

[146] Bajaj, M., Cole, B. und Zwemer, D., "Architecture to geometry-integrating system models with mechanical design," in *AIAA SPACE 2016*, 2016, S. 5470. [Online]. Verfügbar unter: https://doi.org/10.2514/6.2016-5470

[147] Brown, W. J., Ruh, W. A. und Maginnis, F. X., *Enterprise Application Integration: A Wiley Tech Brief*. John Wiley & Sons, 2002.

[148] IBM. "IBM Engineering Systems Design Rhapsody." Zugriff am: 29. März 2023.571Z. [Online.] Verfügbar: https://www.ibm.com/de-de/products/systems-design-rhapsody

[149] Office of Occupational Statistics and Employment Projections. "Electrical and Electronics Engineers : Occupational Outlook Handbook." Zugriff am: 12. Februar

2023.369Z. [Online.] Verfügbar: https://www.bls.gov/ooh/architecture-and-engineering/electrical-and-electronics-engineers.htm#tab-2

[150] Matić, D., Lukač, D., Bugarski, V., Kulić, F. und Nikolić, P., "Computer aided design with Eplan electric P8 educational," *Journal on Processing and Energy in Agriculture*, Jg. 20, Nr. 2, S. 102–105, 2016. [Online]. Verfügbar unter: https://scindeks.ceon.rs/article.aspx?artid=1821-44871602102M

[151] Melkebeek, J. A., *Electrical Machines and Drives: Fundamentals and Advanced Modelling* (Power Systems Ser). Cham: Springer International Publishing AG, 2018. [Online]. Verfügbar unter: https://ebookcentral.proquest.com/lib/kxp/detail.action?docID=5231559

[152] Vukosavić, S. N., *Electrical machines*. New York: Springer, 2014.

[153] Gischel, B., *Handbuch EPLAN Electric P8*. Carl Hanser Verlag GmbH Co KG, 2020.

[154] Karlo Delmis, "Laboratory Scale Drilling Rig," Montanuniversität. [Online]. Verfügbar unter: https://pure.unileoben.ac.at/portal/en/publications/laboratory-scale-drilling-rig(963ce9b2-381b-4fac-b146-6b3f8ccc6009).html

[155] Wikipedia. "Electronic component." Zugriff am: 12. Februar 2023. [Online.] Verfügbar: https://en.wikipedia.org/w/index.php?title=Electronic_component&oldid=1138384967

[156] Sarma, M. S., *Introduction to electrical engineering*. Oxford University Press New York, 2001.

[157] Coy, W., Nake, F., Pflüger, J.-M., Rolf, A., Seetzen, J. und Siefkes, D., *Sichtweisen der Informatik*. Springer-Verlag, 2013.

[158] Office of Occupational Statistics and Employment Projections. "Computer and Information Research Scientists : Occupational Outlook Handbook." Zugriff am: 12. Februar 2023.316Z. [Online.] Verfügbar: https://www.bls.gov/ooh/Computer-and-Information-Technology/Computer-and-information-research-scientists.htm#tab-2

[159] Peng, S.-L., Jain, L. C., Chang, R.-S., Lin, C.-C., Yang, C.-N. und Pan, J.-S., *Advances in intelligent systems and applications: Proceedings of the International Computer Symposium ICS 2012, held at Hualien, Taiwan, December 12-14, 2012* (Smart innovations, systems and technologies 20-21). Heidelberg: Springer, 2013.

[160] Werner, M. C. und Schneider, K., "From IEC 61131-3 Function Block Diagrams to Sequentially Constructive Statecharts," in *2022 Forum on Specification & Design Languages (FDL)*, 2022, doi: 10.1109/fdl56239.2022.9925656.

[161] Beckhoff Automation GmbH & Co. KG. "TwinCAT 3 Safety PLC: Dokumentation PC-basierte Sicherheitssteuerung." Zugriff am: 10. Mai 2023.880Z. [Online.] Verfügbar: https://download.beckhoff.com/download/document/automation/twinsafe/tcsafetyplc_de.pdf

[162] REFA.de. "Sicherheitsingenieur." Zugriff am: 13. Februar 2023.172Z. [Online.] Verfügbar: https://refa.de/berufe/sicherheitsingenieur

[163] *DIN EN ISO 12100:2011-03, Sicherheit von Maschinen_- Allgemeine Gestaltungsleitsätze_- Risikobeurteilung und Risikominderung (ISO_12100:2010); Deutsche Fassung EN_ISO_12100:2010*, 12100, Deutsches Institut für Normung, Berlin.

[164] BfGA Beratungsgesellschaft für Arbeits- und Gesundheitsschutz mbH. "Sicherheitsingenieur - Definition." Zugriff am: 13. Februar 2023.116Z. [Online.] Verfügbar: https://www.bfga.de/arbeitsschutz-lexikon-von-a-bis-z/fachbegriffe-s-u/sicherheitsingenieur/

[165] Gehlen, P., *Funktionale Sicherheit von Maschinen und Anlagen: Umsetzung der Europäischen Maschinenrichtlinie in der Praxis*. John Wiley & Sons, 2010.

[166] Löw, P., Pabst, R. und Petry, E., *Funktionale Sicherheit in der Praxis: Anwendung von DIN EN 61508 und ISO/DIS 26262 bei der Entwicklung von Serienprodukten*. dpunkt.verlag, 2011.

[167] Börcsök, J., *Funktionale Sicherheit: Grundzüge sicherheitstechnischer Systeme*, 5. Aufl. Berlin: VDE Verlag, 2021. [Online]. Verfügbar unter: http://www.content-select.com/index.php?id=bib_view&ean=9783800753581

[168] Schweiz, Europarat, European Communities und Europäische Union, *Richtlinie 2006/42/EG des Europäischen Parlaments und des Rates vom 17. Mai 2006 über Maschinen und zur Änderung der Richtlinie 95/16/EG (Neufassung): Ablage L157/24 : (Berichtigung des Art. 25 Abs. 1 der Richtlinie 2006/42/EG - Ablage L76/35 - am Schluss beigefügt) ; [(Text von Bedeutung für den EWR)]*. Bern: Staatssekretariat für Wirtschaft SECO Arbeitsbedingungen; Vertrieb BBL Bundesamt für Bauten und Logistik, 2009.

[169] *DIN EN IEC 62061 (VDE 0113-50) Funktionale Sicherheit sicherheitsbezogener elektrischer, elektronischer und programmierbarer elektronischer Steuerungssysteme: (IEC 62061:2005 + A1:2012 + A2:2015); Deutsche Fassung EN 62061:2005 + Cor.:2010 + A1:2013 + A2:2015*, 62061, Deutsche Kommission Elektrotechnik, Elektronik, Informationstechnik, Berlin.

[170] Hauke, M. *et al.*, *Funktionale Sicherheit von Maschinensteuerungen: Anwendung der DIN EN ISO 13849*. Berlin: DGUV/IFA, 2017.

[171] *DIN EN ISO 13849-1, Sicherheit von Maschinen - sicherheitsbezogene Teile von Steuerungen. Teil 1, Allgemeine Gestaltungsleitsätze (ISO/DIS 13849-1.2:2021): = Safety of machinery - safety-related parts of control systems. Part 1, General principles for design (ISO/DIS 13849-1.2:2021)*, 13849-1, Deutsches Institut für Normung, Berlin.

[172] Scholl, F., "Volkswirtschaft und Statistik: Maschinenbau in Zahl und Bild 2022," 2022. [Online]. Verfügbar unter: https://www.vdma.org/documents/34570/6128644/Maschinenbau%20in%20Zahl%20und%20Bild%202022.pdf/43a31467-dc91-1bd9-41ee-97413c4e769d

[173] Kagermann, H., Wahlster, W. und Helbig, J., "Umsetzungsempfehlungen für das Zukunftsprojekt Industrie 4.0: Abschlussbericht des Arbeitskreises Industrie 4.0," *Frankfurt am Main*, 2013.

[174] Statista. "Ausgaben für Innovationen im deutschen Maschinenbau bis 2023 | Statista." Zugriff am: 20. März 2023. [Online.] Verfügbar: https://de.statista.com/statistik/daten/studie/157563/umfrage/ausgaben-fuer-innovationen-im-maschinen-und-anlagenbau-in-deutschland/

[175] "Maschine," *Duden.de*, 27. April 2018. Zugriff am: 20. März 2023.898Z. [Online.] Verfügbar: https://www.duden.de/node/94207/revision/1227146

[176] Weck, M., *Maschinenarten, Bauformen und Anwendungsbereiche*. VDI-Verlag, 1991.

[177] Brecher, C. und Weck, M., *Werkzeugmaschinen 1-Maschinenarten und Anwendungsbereiche (VDI-Buch)(German Edition)*. Springer, 2006.

[178] Pöschl, S., Helbig, T., Jacobi, H. F. und Bauernhansl, T., "Aktuelle Forschungsansätze für den Sondermaschinenbau," *Definition und Forschungsergebnisse. wt Werkstatttechnik online (11/12)*, 2016. [Online]. Verfügbar unter: https://www.vdma.org/documents/34570/6128644/Maschinenbau%20in%20Zahl%20und%20Bild%202022.pdf/43a31467-dc91-1bd9-41ee-97413c4e769d

[179] Duden, "Sondermaschine," *Duden.de*, 18. Mai 2018. Zugriff am: 20. März 2023.060Z. [Online.] Verfügbar: https://www.duden.de/node/168298/revision/1237464

[180] Verein Deutscher Ingenieure, *VDI 2221 Blatt 1, Entwicklung technischer Produkte und Systeme - Modell der Produktentwicklung: = Design of technical products and systems - model of product design* (VDI-Richtlinien). Berlin: Beuth Verlag GmbH, 2019.

[181] Verein Deutscher Ingenieure, *VDI 2221 Blatt 2, Entwicklung technischer Produkte und Systeme - Gestaltung individueller Produktentwicklungsprozesse: = Design of technical products and systems - configuration of individual product design processes* (VDI-Richtlinien). Berlin: Beuth Verlag GmbH, 2019.

[182] Uhlig, R., *SPS - modellbasierter Steuerungsentwurf für die Praxis: Modellierungsmethoden aus der Informatik in der Automatisierungstechnik*. Oldenbourg Industrieverlag, 2006.

[183] Bassi, L., Secchi, C., Bonfe, M. und Fantuzzi, C., "A SysML-Based Methodology for Manufacturing Machinery Modeling and Design," *IEEE/ASME Trans. Mechatron.*, Jg. 16, Nr. 6, S. 1049–1062, 2011. doi: 10.1109/tmech.2010.2073480. [Online]. Verfügbar unter: http://dx.doi.org/10.1109/tmech.2010.2073480

[184] Guizani, A., Hammadi, M., Choley, J.-Y., Soriano, T., Abbes, M. S. und Haddar, M., "Multidisciplinary approach for optimizing mechatronic systems: Application to the optimal design of an electric vehicle," in *2014 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, 2014, doi: 10.1109/aim.2014.6878046.

[185] Hammadi, M., "Contribution à l'intégration de la modélisation et la simulation multiphysique pour conception des systèmes mécatroniques," Ecole Centrale Paris École nationale d'ingénieurs de Sfax (Tunisie), 2012. [Online]. Verfügbar unter: https://theses.hal.science/tel-00711469/

[186] Mhenni, F., Choley, J.-Y., Penas, O., Plateaux, R. und Hammadi, M., "A SysML-based methodology for mechatronic systems architectural design," *Advanced Engineering Informatics*, Jg. 28, Nr. 3, S. 218–231, 2014. doi: 10.1016/j.aei.2014.03.006. [Online]. Verfügbar unter: https://www.sciencedirect.com/science/article/pii/s1474034614000342

[187] Barbedienne, R., Penas, O., Choley, J.-Y. und Gasser, L., "TheReSE: SysML extension for thermal modeling," in *2015 Annual IEEE Systems Conference (SysCon) Proceedings*, 2015, doi: 10.1109/syscon.2015.7116768.

[188] Ben Hadj, R., Trigui, M. und Aifaoui, N., "Toward an integrated CAD assembly sequence planning solution," *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, Jg. 229, Nr. 16, S. 2987–3001, 2014. doi: 10.1177/0954406214564412. [Online]. Verfügbar unter: http://dx.doi.org/10.1177/0954406214564412

[189] Koller, R. und Kastrup, N., *Prinziplösungen zur Konstruktion technischer Produkte*. Springer-Verlag, 2013.

[190] Pahl, G. und Beitz, W., *Engineering design: A systematic approach*. Springer Science & Business Media, 2013.

[191] Mhenni, F., Choley, J.-Y. und Nguyen, N., "SysML extensions for safety-critical mechatronic systems design," in *2015 IEEE International Symposium on Systems Engineering (ISSE)*, 2015, S. 242–247, doi: 10.1109/SysEng.2015.7302764.

[192] Barber, A., *Pneumatic Handbook*. Elsevier, 1997.

[193] Roberson, J. A., Cassidy, J. J. und Chaudhry, M. H., *Hydraulic Engineering*. John Wiley & Sons, 1998.

[194] *DIN ISO 1219-2, Fluidtechnik - graphische Symbole und Schaltpläne. Teil 2, Schaltpläne (ISO 1219-2:2012): = Fluid power systems and components - graphical symbols and circuit diagrams. Part 2, Circuit diagrams (ISO 1219-2:2012) = Transmissions hydrauliques et pneumatiques - symboles graphiques et schémas de circuit. Partie 2, Schémas de circuit (ISO 1219-2:2012)*, 1219-2, Deutsches Institut für Normung, Berlin.

[195] Katzke, U., Vogel-Heuser, B. und Member, I., "Combining UML with IEC 61131-3 languages to preserve the usability of graphical notations in the software development of complex automation systems," *IFAC Proceedings Volumes*, Jg. 40, Nr. 16, S. 90–94, 2007. doi: 10.3182/20070904-3-KR-2922.00016. [Online]. Verfügbar unter: https://doi.org/10.3182/20070904-3-KR-2922.00016

[196] Ramos-Hernandez, D. N., Fleming, P. J. und Bass, J. M., "A novel object-oriented environment for distributed process control systems," *Control Engineering Practice*, Jg. 13, Nr. 2, S. 213–230, 2005. doi: 10.1016/j.conengprac.2004.03.007. [Online]. Verfügbar unter: https://doi.org/10.1016/j.conengprac.2004.03.007

[197] Witsch, D. und Vogel-Heuser, B., "PLC-statecharts: An approach to integrate UML-statecharts in open-loop control engineering-aspects on behavioral semantics and model-checking," *IFAC Proceedings Volumes*, Jg. 44, Nr. 1, S. 7866–7872, 2011. doi: 10.3182/20110828-6-IT-1002.02207. [Online]. Verfügbar unter: https://doi.org/10.3182/20110828-6-IT-1002.02207

[198] Witsch, D., *Modellgetriebene Entwicklung von Steuerungssoftware auf Basis der UML unter Berücksichtigung der domänenspezifischen Anforderungen des Maschinen-und Anlagenbaus* (2). Sierke Verlag, 2013.

[199] *IEC 61508:2010, Functional safety of electrical/electronic/programmable electronic safety-related systems*, 61508-1, International Electrotechnical Commission, 2010.

[200] Bertsche, B. und Dazer, M., "Fehlerbaumanalyse (Fault Tree Analysis, FTA)," in *Zuverlässigkeit im Fahrzeug-und Maschinenbau: Ermittlung von Bauteil-und System-Zuverlässigkeiten*, Springer, 2023, S. 149–183. [Online]. Verfügbar unter: https://doi.org/10.1007/978-3-662-65024-0_4

[201] *IEC 62671:2013 - Nuclear power plants - Instrumentation and control important to safety - Selection and use of industrial digital devices of limited functionality*, 62671, Institute of Electrical and Electronics Engineers.

[202] *Analysetechniken für die Funktionsfähigkeit von Systemen - Verfahren für die Fehlzustandsart und -auswirkungsanalyse (FMEA): SN EN 60812*IEC 60812:2006 = Techniques d'analyse de la fiabilité du système - Procédure d'analyse des modes de défaillance et de leurs effets (AMDE) = Analysis techniques for system reliability - Procedure for failure mode and effects analysis (FMEA)*, 62502, Electrosuisse und Schweizerische Normen-Vereinigung. [Online]. Verfügbar unter: http://shop.snv.ch/

[203] *IEC 2006 International Electrotechnical Commission IEC 61025:2006(E) Fault Tree analysis (FTA); Geneva (Switzerland).*

[204] *Verfahren zur Analyse der Zuverlässigkeit - Ereignisbaumanalyse (ETA): SN EN 62502*IEC 62502:2010 = Techniques d'analyse de la sûreté de fonctionnement - Analyse par arbre d'événement (AAE) = Analysis techniques for dependability - Event Tree Analysis (ETA)*, 62502, Electrosuisse und Schweizerische Normen-Vereinigung. [Online]. Verfügbar unter: http://shop.snv.ch/

[205] *Techniken für die Analyse der Zuverlässigkeit - Verfahren mit dem Zuverlässigkeitsblockdiagramm und Boole'sche Verfahren: SN EN 61078*IEC 61078:2006 = Techniques d'analyse pour la sûreté de fonctionnement - Bloc-diagramme de fiabilité et méthodes booléennes = Analysis techniques for dependability - Reliability block diagram and Boolean methods*, 61078, Electrosuisse und Schweizerische Normen-Vereinigung. [Online]. Verfügbar unter: http://shop.snv.ch/

[206] Leveson, N. G. und Thomas, J. P., *System Theoretic Process and Analysis Handbook,*. Eigenverlag, 2018. Zugriff am: 22. März 2023. [Online]. Verfügbar unter: https://psas.scripts.mit.edu/home/get_file.php?name=STPA_handbook.pdf

[207] GSN. "GSN community standard version 1." Zugriff am: 22. März 2023.248Z. [Online.] Verfügbar: https://scsc.uk/gsn

[208] Junges, S., Guck, D., Katoen, J.-P. und Stoelinga, M., "Uncovering Dynamic Fault Trees," in *2016 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, Toulouse, France, Jun. 2016 - Jul. 2016, S. 299–310, doi: 10.1109/DSN.2016.35.

[209] Joshi, A., Vestal, S. und Binns, P., "Automatic Generation of Static Fault Trees from AADL Models," 2007. [Online]. Verfügbar unter: https://hdl.handle.net/11299/217313

[210] Xiang, J., Yanoo, K., Maeno, Y. und Tadano, K., "Automatic Synthesis of Static Fault Trees from System Models," in *2011 Fifth International Conference on Secure Software Integration and Reliability Improvement*, Jeju Island, Korea (South), Jun. 2011 - Jun. 2011, S. 127–136, doi: 10.1109/SSIRI.2011.32.

[211] David, P., Idasiak, V. und Kratz, F., "Reliability study of complex physical systems using SysML," *Reliability Engineering & System Safety*, Jg. 95, Nr. 4, S. 431–450, 2010, doi: 10.1016/j.ress.2009.11.015.

[212] Baklouti, A., Nguyen, N., Mhenni, F., Choley, J.-Y. und Mlika, A., "Improved Safety Analysis Integration in a Systems Engineering Approach," *Applied Sciences*, Jg. 9, Nr. 6, S. 1246, 2019. doi: 10.3390/app9061246. [Online]. Verfügbar unter: https://www.mdpi.com/2076-3417/9/6/1246

[213] Berres, A. und Schumann, H., "Closing the safety process gap: Early integration of safety assessment methods into systems engineering," in *Tag des Systems Engineering*, M. Maurer und S.-O. Schulze, Hg., München: Carl Hanser Verlag GmbH & Co. KG, 2014, S. 143–152. [Online]. Verfügbar unter: https://doi.org/10.3139/9783446443761.015

[214] Volk, M., Irshad, M. I., Katoen, J.-P., Sher, F., Stoelinga, M. I. A. und Zafar, A., "SAFEST: the static and dynamic fault tree analysis tool," in *33rd European Safety and Reliability Conference, ESREL 2023*, 2023, S. 193–200, doi: 10.3850/978-981-18-8071-1_P407-cd.

[215] Dugan, J. B., Bavuso, S. J. und Boyd, M. A., "Dynamic fault-tree models for fault-tolerant computer systems," *IEEE Trans. Rel.*, Jg. 41, Nr. 3, S. 363–377, 1992, doi: 10.1109/24.159800.

[216] Ruijters, E. und Stoelinga, M., "Fault tree analysis: A survey of the state-of-the-art in modeling, analysis and tools," *Computer Science Review*, Jg. 15-16, S. 29–62, 2015. doi: 10.1016/j.cosrev.2015.03.001. [Online]. Verfügbar unter: https://www.sciencedirect.com/science/article/pii/S1574013715000027

[217] Aslansefat, K., Kabir, S., Gheraibia, Y. und Papadopoulos, Y., "Dynamic Fault Tree Analysis: State-of-the-Art in Modeling, Analysis, and Tools," in *Reliability management and engineering: Challenges and future trends* (Advanced research in reliability and system assurance engineering), H. Garg und M. Ram, Hg. Boca Raton, FL: CRC Press, 2020, S. 73–112.

[218] Dugan, J. B. und Trivedi, K. S., "Coverage modeling for dependability analysis of fault-tolerant systems," *IEEE Transactions on Computers*, Jg. 38, Nr. 6, S. 775–787, 1989, doi: 10.1109/12.24286.

[219] Sullivan, K. und Dugan, J. B. "Galileo User's Manual & Design Overview: Version 2.11-Alpha." Zugriff am: 29. März 2023.189Z. [Online.] Verfügbar: https://www.cse.msu.edu/~cse870/Materials/FaultTolerant/manual-galileo.htm#Editing%20in%20the%20Textual%20View

[220] Basavalingappa, A., Passage, J. M., Shen, M. Y. und Lloyd, JR, "Electromigration: Lognormal versus weibull distribution," in *2017 IEEE International Integrated Reliability Workshop (IIRW)*, 2017, S. 1–4, doi: 10.1109/IIRW.2017.8361224.

[221] Baklouti, A., Nguyen, N., Mhenni, F., Choley, J.-Y. und Mlika, A., "Dynamic fault tree generation for safety-critical systems within a systems engineering approach," *IEEE Systems Journal*, Jg. 14, Nr. 1, S. 1512–1522, 2019, doi: 10.1109/JSYST.2019.2930184.

[222] Institut für Arbeitsschutz, "Das SISTEMA-Kochbuch: Vom Schaltbild zum Performance Level - Quantifizierung von Sicherheitsfunktionen mit SISTEMA," [Online]. Verfügbar unter: https://www.dguv.de/medien/ifa/de/pra/softwa/sistema/kochbuch/sistema_kochbuch1_de.pdf

[223] Pétin, J.-F., Evrot, D., Morel, G. und Lamy, P., "Combining SysML and formal methods for safety requirements verification," in *22nd International Conference on Software \& Systems Engineering and their Applications*, CDROM. Zugriff am: 5. November 2010. [Online]. Verfügbar unter: https://hal.archives-ouvertes.fr/hal-00533311

[224] Dhouib, S., Kchir, S., Stinckwich, S., Ziadi, T. und Ziane, M., "RobotML, a Domain-Specific Language to Design, Simulate and Deploy Robotic Applications," in *Simulation, Modeling, and Programming for Autonomous Robots* (Lecture Notes in Computer Science), D. Hutchison et al., Hg., Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, S. 149–160.

[225] Thramboulidis, K., Soliman, D. und Frey, G., "Towards an automated verification process for industrial safety applications," in *IEEE International Conference on Automation Science and Engineering (CASE), 2011: Trieste, Italy, 24-27 August 2011*, Trieste, Italy, M. Y. Wang, Hg., 2011, S. 482–487, doi: 10.1109/CASE.2011.6042451.

[226] Müller, M., Roth, M. und Lindemann, U., "The hazard analysis profile: Linking safety analysis and SysML," in *2016 Annual IEEE Systems Conference (SysCon)*, 2016, S. 1–7, doi: 10.1109/SYSCON.2016.7490532.

[227] Mhenni, F., Choley, J.-Y. und Nguyen, N., "An integrated design methodology for safety critical systems," in *2016 Annual IEEE Systems Conference (SysCon)*, 2016, S. 1–6, doi: 10.1109/SYSCON.2016.7490580.

[228] Choley, J.-Y., Mhenni, F., Nguyen, N. und Baklouti, A., "Topology-based safety analysis for safety critical CPS," *Procedia computer science*, Jg. 95, S. 32–39, 2016, doi: 10.1016/j.procs.2016.09.290.

[229] Käßmeyer, M., Moncada, D. S. V. und Schurius, M., "Evaluation of a systematic approach in variant management for safety-critical systems development," in *2015 IEEE 13th International Conference on Embedded and Ubiquitous Computing*, 2015, S. 35–43, doi: 10.1109/EUC.2015.12.

[230] Antonino, P. O., Moncada, D. V., Kuhn, T., Schneider, D. und Trapp, M., "Integrated Model-based Safety Engineering with I-SafE," in *Embedded Software Engineering Kongress 2015 (ESE 2015)*, 2015. [Online]. Verfügbar unter: https://www.researchgate.net/publication/299492731_Integrated_Model-based_Safety_Engineering_with_I-SafE

[231] Velasco Moncada, D. S., "Hazard-driven realization views for Component Fault Trees," *Software and Systems Modeling*, Jg. 19, Nr. 6, S. 1465–1481, 2020, doi: 10.1007/s10270-020-00792-8.

[232] Huang, Z., Swalgen, S., Davidz, H. und Murray, J., "MBSE-assisted FMEA approach—Challenges and opportunities," in *2017 Annual Reliability and Maintainability Symposium (RAMS)*, 2017, S. 1–8, doi: 10.1109/RAM.2017.7889722.

[233] Huang, Z., Hansen, R. und Huang, Z., "Toward FMEA and MBSE Integration," in *2018 Annual Reliability and Maintainability Symposium (RAMS)*, 2018, S. 1–7, doi: 10.1109/RAM.2018.8463084.

[234] Dumont, J. "Eclipse Safety Framework." Zugriff am: 6. Juli 2024.182Z. [Online.] Verfügbar: https://eclipse.dev/esf/index.php

[235] Berres, A. und Schumann, H., "Automatic generation of fault trees: A survey on methods and approaches," *ESREL 2016*, 2016, doi: 10.1201/9781315374987-377.

[236] Object Management Group. "Risk Analysis and Assessment Modeling Language Specification Version 1.0 beta 2." Zugriff am: 22. März 2023.236Z. [Online.] Verfügbar: https://www.omg.org/spec/RAAML/1.0/Beta2/PDF

[237] Nordmann, A. und Munk, P., "Lessons learned from model-based safety assessment with SysML and component fault trees," in *Proceedings of the 21th ACM/IEEE International Conference on Model Driven Engineering Languages and Systems*, 2018, S. 134–143, doi: 10.1145/3239372.3239373.

[238] Munk, P. und Nordmann, A., "Model-based safety assessment with SysML and component fault trees: application and lessons learned," *Software and Systems Modeling*, Jg. 19, Nr. 4, S. 889–910, 2020. doi: 10.1007/s10270-020-00782-w. [Online]. Verfügbar unter: https://doi.org/10.1007/s10270-020-00782-w

[239] Electric Power Research Institute. "Computer Aided Fault Tree Analysis System (CAFTA), Version 6.0b." Zugriff am: 27. März 2023.787Z. [Online.] Verfügbar: https://www.epri.com/research/products/3002004316

[240] Delange, J. "EMFTA: an Open Source Tool for Fault Tree Analysis." Zugriff am: 27. März 2023.044Z. [Online.] Verfügbar: https://insights.sei.cmu.edu/blog/emfta-an-open-source-tool-for-fault-tree-analysis/

[241] ALD Ltd. "RAMS Reliability, Availability, Maintainability and Safety Software." Zugriff am: 27. März 2023.909Z. [Online.] Verfügbar: https://aldservice.com/RAMS-Reliability-Availability-Maintainability-and-Safety-Software.html

[242] ALD Ltd. "ALD Fault Tree Analysis (FTA)." Zugriff am: 27. März 2023.898Z. [Online.] Verfügbar: https://aldservice.com/Reliability/fault-tree-analysis.html

[243] Arnold, F., Belinfante, A., van der Berg, F., Guck, D. und Stoelinga, M., "DFTC alc: a tool for efficient fault tree analysis," in *International Conference on Computer Safety, Reliability, and Security*, 2013, S. 293–301, doi: 10.1007/978-3-642-40793-2_27.

[244] Hensel, C., Junges, S., Katoen, J.-P., Quatmann, T. und Volk, M., "The probabilistic model checker Storm," *Int J Softw Tools Technol Transfer*, 2021, doi: 10.1007/s10009-021-00633-z.

[245] Durga Rao, K., Gopika, V., Sanyasi Rao, V., Kushwaha, H. S., Verma, A. K. und Srividya, A., "Dynamic fault tree analysis using Monte Carlo simulation in probabilistic safety assessment," *Reliability Engineering & System Safety*, Jg. 94, Nr. 4, S. 872–883, 2009, doi: 10.1016/j.ress.2008.09.007.

[246] Ertelt, J. "Softwaregestützte Risikobeurteilung | WEKA." Zugriff am: 26. März 2023.525Z. [Online.] Verfügbar: https://www.weka.de/produktsicherheit/softwaregestuetzte-risikobeurteilung/

[247] Pilz GmbH. "PAScal Safety Calculator: Performance level in accordance with EN ISO 13849-1 - Pilz INT." Zugriff am: 16. Mai 2023.997Z. [Online.] Verfügbar: https://www.pilz.com/en-INT/products/software/services-software/pascal-safety-calculator

[248] Bartels, J. und Vogel, B., "Systementwicklung für die Automatisierung im Anlagenbau (System Engineering Approach for Plant Automation)," 2001. doi:

10.1524/auto.2001.49.5.214. [Online]. Verfügbar unter: https://doi.org/10.1524/auto.2001.49.5.214

[249] Henderson, K. und Salado, A., "The effects of organizational structure on MBSE adoption in industry: Insights from practitioners," *Engineering Management Journal*, Jg. 36, Nr. 1, S. 117–143, 2024. doi: 10.1080/10429247.2023.2210494. [Online]. Verfügbar unter: https://www.researchgate.net/publication/370988730_The_Effects_of_Organizational_Structure_on_MBSE_Adoption_in_Industry_Insights_from_Practitioners

[250] Schweizerische Normen-Vereinigung, *Ergonomische Anforderungen für Bürotätigkeiten mit Bildschirmgeräten - Teil 11: Anforderungen an die Gebrauchstauglichkeit; Leitsätze (ISO 9241-11:1998): SN EN ISO 9241-11 = Exigences ergonomiques pour travail de bureau avec terminaux à écrans de visualisation (TEV) - Partie 11: Lignes directes concernant l'utilisabilité (ISO 9241-11:1998) = Ergonomic requirements for office work with visual display terminals (VDTs) - Part 11: Guidance on usability (ISO 9241-11:1998)*. Winterthur: Schweizerische Normen-Vereinigung (SNV), 1999.

[251] Sarodnick, F. und Brau, H., *Methoden der usability evaluation*. Hogrefe, vorm. Verlag Hans Huber, 2011.

[252] Bella, E. E., Creff, S., Gervais, M.-P. und Bendraou, R., "ATLaS: A framework for traceability links recovery combining information retrieval and semi-supervised techniques," in *2019 IEEE 23rd International Enterprise Distributed Object Computing Conference (EDOC)*, 2019, S. 161–170, doi: 10.1109/EDOC.2019.00028.

[253] Burton-Jones, A., Wand, Y. und Weber, R., "Guidelines for empirical evaluations of conceptual modeling grammars," *Journal of the Association for Information Systems*, Jg. 10, Nr. 6, S. 1, 2009. doi: 10.17705/1jais.00201. [Online]. Verfügbar unter: https://www.researchgate.net/publication/220580371_Guidelines_for_Empirical_Evaluations_of_Conceptual_Modeling_Grammars

[254] Backhaus, K., Erichson, B., Gensler, S., Weiber, R. und Weiber, T., "Einführung in die empirische Datenanalyse," in *Multivariate Analysemethoden: Eine anwendungsorientierte Einführung* (Springer eBook Collection), K. Backhaus, B. Erichson, S. Gensler, R. Weiber und T. Weiber, Hg., 16. Aufl. Wiesbaden: Springer Gabler, 2021, S. 1–60. [Online]. Verfügbar unter: https://link.springer.com/chapter/10.1007/978-3-658-32425-4_1

[255] Köstner, H., "Grundlagen der Datenanalyse," *Empirische Forschung in den Wirtschafts- und Sozialwissenschaften klipp & klar*, S. 111–146, 2022. doi: 10.1007/978-3-658-38599-6_6. [Online]. Verfügbar unter: https://link.springer.com/chapter/10.1007/978-3-658-38599-6_6

[256] Böhmert, C. und Abacioglu, F., "Grundlagenbeitrag: Quantitative Befragungen," in *Evaluationsmethoden der Wissenschaftskommunikation*, P. Niemann, Hg., 1. Aufl. Wiesbaden: Springer Fachmedien Wiesbaden GmbH, 2023, S. 69–83. [Online]. Verfügbar unter: http://dx.doi.org/10.1007/978-3-658-39582-7_5

[257] Peck, L. R., "Leveraging experimental evaluations for understanding causal mechanisms," *New Directions for Evaluation*, Jg. 2020, Nr. 167, S. 145–160, 2020. doi: 10.1002/ev.20422. [Online]. Verfügbar unter: https://www.researchgate.net/publication/346111065_Leveraging_Experimental_Evaluations_for_Understanding_Causal_Mechanisms

[258] Kardorff, E. von und Schönberger, C., "Qualitative Evaluationsforschung," in *Handbuch Qualitative Forschung in der Psychologie* (Springer Reference), G. Mey und K. Mruck, Hg., 2. Aufl. Wiesbaden: Springer, 2020, S. 135–157. [Online]. Verfügbar unter: https://doi.org/10.1007/978-3-658-26887-9_26

[259] "Warum eine qualitative Evaluation?," in *Qualitative Evaluation: Der Einstieg in die Praxis* (Springer eBook Collection Humanities, Social Science), U. Kuckartz, T. Dresing, S. Rädiker und C. Stefer, Hg., 2. Aufl. Wiesbaden: VS Verlag für Sozialwissenschaften, 2008, S. 11–14. [Online]. Verfügbar unter: https://link.springer.com/chapter/10.1007/978-3-531-91083-3_1

[260] Cousin, G., "Case Study Research," *Journal of geography in higher education*, Jg. 29, Nr. 3, S. 421–427, 2005, doi: 10.1080/03098260500290967.

[261] Siau, K. und Rossi, M., "Evaluation techniques for systems analysis and design modelling methods – a review and comparative analysis," *Information Systems Journal*, Jg. 21, Nr. 3, S. 249–268, 2011, doi: 10.1111/j.1365-2575.2007.00255.x.

[262] Katzke, U., Vogel-Heuser, B. und Fischer, K., "Analysis and state of the art of modules in industrial automation," *Automatisierungstechnische Praxis (atp)*, Jg. 46, Nr. 1, S. 23–31, 2004. [Online]. Verfügbar unter: https://api.semanticscholar.org/CorpusID:116122315

[263] Beywl, W., "Evaluationsmodelle und qualitative Methoden," *Flick, Uwe (Hg.): Qualitative Evaluationsforschung. Konzepte, Methoden, Umsetzung, Reinbek bei Hamburg*, S. 92–116, 2006, doi: 10.13140/RG.2.2.34480.07686.

[264] Sarbu, C. und Jantschi, L., "Statistic validation and evaluation of the analytic methods by comparative studies. I. Analytical methods validation by regression analysis," *Revista de Chimie*, Jg. 49, Nr. 1, S. 19–24, 1998. [Online]. Verfügbar unter: https://www.rese-

archgate.net/publication/299016053_Statistic_validation_and_evaluation_of_the_analytic_methods_by_comparative_studies_I_Analytical_methods_validation_by_regression_analysis

[265] Nielsen, J., *Usability Engineering*. Morgan Kaufmann, 1994.

[266] Obermeier, M., Schutz, D. und Vogel-Heuser, B., "Evaluation of a newly developed model-driven PLC programming approach for machine and plant automation," in *2012 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2012, doi: 10.1109/icsmc.2012.6377957.

[267] Stromman, M., Sierla, S. und Koskinen, K., "Control Software Reuse Strategies with IEC 61499," in *2005 IEEE Conference on Emerging Technologies and Factory Automation*, null, doi: 10.1109/etfa.2005.1612749.

[268] Sierla, S., Christensen, J., Koskinen, K. und Peltola, J., "Educational approaches for the industrial acceptance of IEC 61499," in *2007 IEEE Conference on Emerging Technologies & Factory Automation (EFTA 2007)*, 2007, doi: 10.1109/efta.2007.4416807.

[269] Volk, S. C., "Evaluation der Wissenschaftskommunikation: Modelle, Stufen, Methoden," in *Evaluationsmethoden der Wissenschaftskommunikation*, Springer Fachmedien Wiesbaden Wiesbaden, 2023, S. 33–49. [Online]. Verfügbar unter: https://library.oapen.org/bitstream/handle/20.500.12657/61908/978-3-658-39582-7.pdf?sequence=1#page=42

[270] Blöbaum, B., Nölleke, D. und Scheu, A. M., "Das Experteninterview in der Kommunikationswissenschaft," *Handbuch nicht standardisierte Methoden in der Kommunikationswissenschaft*, S. 175–190, 2016. doi: 10.1007/978-3-658-01656-2_11. [Online]. Verfügbar unter: https://link.springer.com/chapter/10.1007/978-3-658-01656-2_11#citeas

[271] Abdinnour, S. und Saeed, K., "User perceptions towards an ERP system: Comparing the post-implementation phase to the pre-implementation phase," *Journal of Enterprise Information Management*, Jg. 28, Nr. 2, S. 243–259, 2015. doi: 10.1108/JEIM-10-2013-0075. [Online]. Verfügbar unter: https://doi.org/10.1108/JEIM-10-2013-0075

[272] Savelberg, W., Moser, A., Smidt, M., Boersma, L., Haekens, C. und van der Weijden, T., "Protocol for a pre-implementation and post-implementation study on shared decision-making in the surgical treatment of women with early-stage breast cancer," *BMJ open*, Jg. 5, Nr. 3, e007698, 2015. doi: 10.1136/bmjopen-2015-007698. [Online]. Verfügbar unter: https://doi.org/10.1136/bmjopen-2015-007698

[273] Bernstein, D. und Booch, G., "The UML and the Rational Unified Process," *IEEE Softw.*, Jg. 37, Nr. 6, S. 12, 2020. doi: 10.1109/MS.2020.3019539. [Online]. Verfügbar unter: https://www.computer.org/csdl/magazine/so/2020/06/09238473/1oa116k3WF2

[274] Hathaway, T. und Hathaway, A., *Data Flow Diagrams: Simply Put!: Process Modeling Techniques for Requirements Elicitation and Workflow Analysis*. BA-EXPERTS, 2016.

[275] *DIN EN ISO 80000-1, Größen und Einheiten. Teil 1, Allgemeines (ISO/DIS 80000-1:2022): = Quantities and units. Part 1, General (ISO/DIS 80000-1:2022)*, 80000-1, Deutsches Institut für Normung, Berlin.

[276] Object Management Group, *UML Profile for MARTE: Modeling and Analysis of Real-Time Embedded Systems: Version 1.2.* Zugriff am: 23. April 2021. [Online]. Verfügbar unter: https://www.omg.org/spec/MARTE/1.2/PDF

[277] E. V., S. und Samuel, P., "Automatic Code Generation From UML State Chart Diagrams," *IEEE Access*, Jg. 7, S. 8591–8608, 2019. doi: 10.1109/ACCESS.2018.2890791. [Online]. Verfügbar unter: https://ieeexplore.ieee.org/document/8600324

[278] GitHub. "Code Generation from UML Statechart: Praktische Lösungen." Zugriff am: 26. Juli 2023.816Z. [Online.] Verfügbar: https://github.com/topics/uml-state-machine

[279] Fantuzzi, C., Secchi, C. und Bonfè, M., "A Design Pattern for translating UML software models into IEC 61131-3 Programming Languages," *IFAC Proceedings Volumes*, Jg. 44, Nr. 1, S. 9158–9163, 2011. doi: 10.3182/20110828-6-IT-1002.02893. [Online]. Verfügbar unter: https://www.sciencedirect.com/science/article/pii/S1474667016450822

[280] Vogel-Heuser, B., Witsch, D. und Katzke, U., "Automatic code generation from a UML model to IEC 61131-3 and system configuration tools," in *2005 International Conference on Control and Automation*, Bd. 2, 2005, 1034-1039 Vol. 2, doi: 10.1109/ICCA.2005.1528274.

[281] Cederbladh, J. *et al.*, "Towards Automating Model-Based Systems Engineering in Industry-An Experience Report," in *The 18th Annual IEEE International Systems Conference (SYSCON 2024)*, 2024, doi: 10.1109/SysCon61195.2024.10553610.

[282] Cheng, C.-L., Yeh, J. C., Chern, S. C. und Lan, Y.-H., "Automatic testing system based on LabVIEW for dc motor of portable washing machine," in *3rd IEEE Conference on Industrial Electronics and Applications, 2008: ICIEA 2008 ; 3 - 5 June 2008, Singapore*, Singapore, 2008, S. 489–493, doi: 10.1109/ICIEA.2008.4582563.

[283] Wang, G., Jiao, S. und Song, H., "Mine Pump Comprehensive Performance Testing System Based on Labview," in *2009 International Conference on Measuring Technology and Mechatronics Automation*, 2009, doi: 10.1109/icmtma.2009.179.

[284] National Instruments. "Weltweit führend beim Testen." Zugriff am: 11. Juni 2023.270Z. [Online.] Verfügbar: https://www.ni.com/de-de.html?cid=PSEA-7013q000001rC7dAAE-Consideration-GOGSE_146824689006&utm_keyword=ni&gad=1&gclid=CjwKCAjw4ZWkBhA4EiwAVJXwqYjfxGN1I8UHIEXppiv-fHvCGqvTiTD2CSzBS_VmY_W7GPi2ypLFLZxoC7GoQAvD_BwE

[285] Auer, M. E., Zutin, D. G. und Rajyaguru, C., "A LabVIEW toolkit for the development of iLab batched lab servers," in *2011 IEEE Global Engineering Education Conference (EDUCON)*, 2011, doi: 10.1109/educon.2011.5773107.

[286] Object Management Group. "About the Risk Analysis and Assessment Modeling Language Specification Version 1.0." Zugriff am: 11. Mai 2023.676Z. [Online.] Verfügbar: https://www.omg.org/spec/RAAML/

[287] Eclipse. "Eclipse OCL™ (Object Constraint Language)." Zugriff am: 11. Mai 2023.643Z. [Online.] Verfügbar: https://projects.eclipse.org/projects/modeling.mdt.ocl

[288] Object Management Group. "About the OMG Systems Modeling Language Specification Version 1.7 beta." Zugriff am: 11. Mai 2023.857Z. [Online.] Verfügbar: https://www.omg.org/spec/OCL/2.4/PDF

[289] Maggi, B. "478131 – [SysML 1.4] Add a template for creating model with ISO80000 library." Zugriff am: 11. Mai 2023EDT. [Online.] Verfügbar: https://bugs.eclipse.org/bugs/show_bug.cgi?id=478131

[290] Guindon, C. "Eclipse software repository | MARTE." Zugriff am: 11. Mai 2023.767Z. [Online.] Verfügbar: https://download.eclipse.org/modeling/mdt/papyrus/components/marte/

[291] Hensel, C., Junges, S., Katoen, J.-P., Quatmann, T. und Volk, M. "Storm -- A Modern Probabilistic Model Checker -- Running Storm on DFTs." Zugriff am: 29. März 2023.087Z. [Online.] Verfügbar: https://www.stormchecker.org/documentation/usage/running-storm-on-dfts.html

[292] GitHub. "GitHub - JKISoftware/JKI-EasyXML: JKI EasyXML Toolkit for LabVIEW." Zugriff am: 15. Mai 2023.406Z. [Online.] Verfügbar: https://github.com/JKISoftware/JKI-EasyXML

[293] Institut für Arbeitsschutz. "Das SISTEMA-Kochbuch 5: SISTEMA-Bibliotheken." Zugriff am: 10. Juli 2023. [Online.] Verfügbar: https://www.dguv.de/medien/ifa/de/pra/softwa/sistema/kochbuch/sistema_kochbuch5_de.pdf

[294] National Instruments. "Einschätzung der Codekomplexität in LabVIEW." Zugriff am: 16. Mai 2023. [Online.] Verfügbar: https://www.ni.com/de-de/shop/labview/estimating-code-complexity-in-labview.html

[295] Berezowski, N. "Github repository of MMBSEFE and Transformators." Zugriff am: 24. Mai 2023. [Online.] Verfügbar: https://github.com/Toxilano?tab=repositories

[296] Berezowski, N. "MMBSEFE and Transformators Documentation: PW: D15ert4t10n_B3r320Wsk1." Zugriff am: 24. Mai 2023.

[297] Object Management Group, *Object Constraint Language Specification, Version 2.4. Official Specification*, 2014. Zugriff am: 10. Februar 2023. [Online]. Verfügbar unter: https://www.omg.org/spec/OCL/2.4

[298] Meyermann, A. und Porzelt, M., *Hinweise zur Anonymisierung qualitativer Daten* (in de). DIPF | Leibniz-Institut für Bildungsforschung und Bildungsinformation : Frankfurt am Main, doi: 10.25656/01:21968.

[299] *Verordnung (EU) 2023/ des Europäischen Parlaments und des Rates vom 14. Juni 2023 über Maschinen und zur Aufhebung der Richtlinie 2006/42/EG des Europäischen Parlaments und des Rates und der Richtlinie 73/361/EWG des Rates*, EU 2023/1230. [Online]. Verfügbar unter: https://eur-lex.europa.eu/legal-content/DE/TXT/?uri=CELEX:32023R1230

[300] Berezowski, N. und Haid, M., "P6.1 - High-Assurance System Development with LabVIEW," *Tagungsband*, S. 669–675, 2016. doi: 10.5162/sensoren2016/P6.1. [Online]. Verfügbar unter: https://www.ama-science.org/proceedings/details/2426

[301] Berezowski, N., Haid, M. und Hartmann, M.-E. "Empfehlungen zur Anwendung von graphischen Programmiersprachen in sicherheitsbezogenen Systementwicklungen: Conference: AALE 2019 - 16. Fachkonferenz "Autonome und intelligente Systeme in der Automatisierungstechnik"" Proceedings: AALE 2019. Zugriff am: 4. Oktober 2019. [Online.] Verfügbar: https://www.vde-verlag.de/proceedings-de/564860017.html

[302] Berezowski, N., Haid, M. und Hartmann, M.-E., "Recommendations for Developing Safety-Related Systems with Graphical Languages," in *CERC*, 2019, S. 89–102. [Online]. Verfügbar unter: https://ceur-ws.org/Vol-2348/paper07.pdf

[303] Berezowski, N., Haid, M., Biswas, J. und Boyaci, I., "P3. 3 Safety Sensor Applications with Graphical Programming," *SMSI 2020-Measurement Science*, S. 321–322, 2020. doi:

10.5162/SMSI2020/P3.3. [Online]. Verfügbar unter: https://www.ama-science.org/proceedings/details/3780

[304] Berezowski, N., Haid, M., Biswas, J. und Boyaci, I., "Graphical Programming Languages for Functional Safety," in *2020 IEEE Symposium on Product Compliance Engineering - (SPCE Portland)*, Portland, OR, Nov. 2020 - Nov. 2020, S. 1–10, doi: 10.1109/SPCE50045.2020.9296160.

[305] Berezowski, N. und Haid, M., "How to Use a Graphical Programming Language in Functional Safety, using the Example of Lab VIEW," in *2020 IEEE International RF and Microwave Conference (RFM)*, 2020, S. 1–4, doi: 10.1109/RFM50841.2020.9344729.

[306] Berezowski, N. und Haid, M., "MBSE for SMEs with Domain-Specific Safety Analyses and Loose Tool Coupling," in *2020 IEEE International Conference on Sustainable Engineering and Creative Computing (ICSECC)*, Bd. 674, 2020, S. 72–81, doi: 10.1007/978-3-031-32700-1_8.

[307] Berezowski, N. und Haid, M., "A Brief Subset for Functional Safety in LabVIEW," in *2020 IEEE International Conference on Advent Trends in Multidisciplinary Research and Innovation (ICATMRI)*, 2020, S. 1–8, doi: 10.1109/ICATMRI51801.2020.9398473.

[308] Holt, J., *UML for Systems Engineering: Watching the Wheels*. IET, 2004.

[309] Object Management Group, "Information technology - Object Management Group Systems Modeling Language (OMG SysML)," 2017. [Online]. Verfügbar unter: https://www.omg.org/spec/SysML/ISO/19514/PDF

[310] Holt, J. und Perry, S., "SysML for Systems Engineering, 2nd Edition: A Model-Based Approach," IET, 2013, doi: 10.1049/PBPC010E. [Online]. Verfügbar unter: https://digital-library.theiet.org/doi/book/10.1049/pbpc010e

[311] Graves, H. und Bijan, Y., "Using formal methods with SysML in aerospace design and engineering," *Ann Math Artif Intell*, Jg. 63, Nr. 1, S. 53–102, 2011, doi: 10.1007/s10472-011-9267-5.

[312] Object Management Group, *Business Process Model and Notation (BPMN): Version 2.0.2*, 2013. Zugriff am: 8. April 2023. [Online]. Verfügbar unter: https://www.omg.org/spec/BPMN/2.0.2/PDF

[313] Object Management Group, *Software & Systems Process Engineering Meta-Model Specification: Version 2.0*, 2008. [Online]. Verfügbar unter: https://www.omg.org/spec/SPEM/2.0/PDF

[314] Object Management Group, *UML Infrastructure 2.4.1*, 2011. [Online]. Verfügbar unter: http://www. omg. org/spec/UML/2.4. 1

[315] Wikipedia. "CSS." Zugriff am: 8. Mai 2023. [Online.] Verfügbar: https://en.wikipedia.org/w/index.php?title=CSS&oldid=1153654931

[316] Steinberg, D., *EMF : Eclipse Modeling Framework: Eclipse Modeling Framework*, 2. Aufl. (The eclipse series EMF). Place of publication not identified: Addison Wesley, 2009.

[317] Budinsky, F., Ellersick, R., Steinberg, D., Grose, T. J. und Merks, E., *Eclipse modeling framework: A developer's guide*, 6. Aufl. (The eclipse series). Boston, Mass., Munich: Addison-Wesley, 2007.

[318] Sysmlforum. "Commercial, Free & Open Source SysML Tools for MBSE." Zugriff am: 13. Februar 2023.838Z. [Online.] Verfügbar: https://sysmlforum.com/sysml-tools/

[319] Nyamsi, E. A., "Eclipse-Papyrus-Framework," in *IT-Lösungen auf Basis von SysML und UML*, Springer Vieweg, Wiesbaden, 2020, S. 193–296. [Online]. Verfügbar unter: https://link.springer.com/chapter/10.1007/978-3-658-29057-3_3

[320] Sparx Systems. "SPARXSYSTEMS: UML, SysML, BPMN, Togaf, Updm vereint in Enterprise Architect." Zugriff am: 29. März 2023.194Z. [Online.] Verfügbar: https://www.sparxsystems.de/

[321] PTC. "Windchill Modeler (formerly Integrity Modeler) Software To Simplify Your Design Process." Zugriff am: 29. März 2023.673Z. [Online.] Verfügbar: https://www.ptc.com/en/products/windchill/modeler

[322] Hecht, M., Chuidian, A., Tanaka, T. und Raymond, R., "Automated Generation of FMEAs using SysML for Reliability, Safety, and Cybersecurity," in *2020 Annual Reliability and Maintainability Symposium (RAMS)*, 2020, S. 1–7, doi: 10.1109/RAMS48030.2020.9153708.

[323] Molhanec, M., Zhuravskaya, O., Povolotskaya, E. und Tarba, L., "The ontology based FMEA of lead free soldering process," in *Proceedings of the 2011 34th International Spring Seminar on Electronics Technology (ISSE)*, Tratanska Lomnica, Slovakia, Mai. 2011 - Mai. 2011, S. 267–273, doi: 10.1109/ISSE.2011.6053871.

[324] Brusa, E., "Digital Twin: Toward the Integration Between System Design and RAMS Assessment Through the Model-Based Systems Engineering," *IEEE Systems Journal*, Jg. 15, Nr. 3, S. 3549–3560, 2021. doi: 10.1109/JSYST.2020.3010379. [Online]. Verfügbar unter: https://www.researchgate.net/publication/343407633_Digital_Twin_Toward_the_Integration_Between_System_Design_and_RAMS_Assessment_Through_the_Model-Based_Systems_Engineering

[325] Clemens, P. L., Simmons, R. J. und Cincinnati, O., "System safety and risk management: A guide for engineering educators," *Lesson II Risk assessment matrix. NIOSH Instructional Module. Cincinnati, OH, USA: US Department of Health and Human Services*, 1998.

[326] Wang, J. X. und Roush, M. L., *What Every Engineer Should Know About Risk Engineering and Management*. CRC Press, 2000.

[327] Friedberg, I., McLaughlin, K., Smith, P., Laverty, D. und Sezer, S., "STPA-SafeSec: Safety and security analysis for cyber-physical systems," *Journal of information security and applications*, Jg. 34, S. 183–196, 2017, doi: 10.1016/j.jisa.2016.05.008.

[328] Nohl, J. und Thiemecke, H., *Systematik zur Durchführung von Gefährdungsanalysen* (Schriftenreihe der Bundesanstalt für Arbeitsschutz Forschung FB 536). Bremerhaven: Wirtschaftsverl. NW, 1988.

[329] Agerer, M. S. "Risikomatrix nach Nohl." [Online.] Verfügbar: http://www.maschinensicherheit.net/07-seiten/0590-risikomatrix-nohl.php

[330] *DIN ISO/TR 14121-2:2013-02: Sicherheit von Maschinen - Risikobeurteilung - Teil 2: Praktischer Leitfaden und Verfahrensbeispiele*, 14121-2, Deutsches Institut für Normung.

A Anhang

A.1 Publikationen & Vorträge

- Als Nacharbeit zur Masterthesis wurde für die Fachausstellung Sensor & Test 2016 eine Veröffentlichung mit dem Titel „High-Assurance System Development with LabVIEW“ geschrieben, welche die Grundlagen der sicherheitsbezogenen graphischen Programmierung und den notwenigen Schritten in LabVIEW in den Anfängen beschrieb [300].
- Im Zusammenhang des gleichen Themengebietes wurde für die Fachausstellung Sensor & Test 2016 eine Veröffentlichung mit dem Titel „High-Assurance System Development with LabVIEW“ geschrieben, welche die Grundlagen der sicherheitsbezogenen graphischen Programmierung und den notwenigen Schritten in LabVIEW in den Anfängen beschrieb [300].
- Auf der AALE Conference 2019 wurde eine Publikation und kurzer Vortrag zum Thema „Empfehlungen zur Anwendung von graphischen Programmiersprachen in sicherheitsbezogenen Systementwicklungen“ gestaltet. Hierbei sollten erste weitere Schritte, die sich im Rahmen der aktuellen Promotion entwickeln sollten, beschrieben und ein erster Ausblick gegeben werden [301].
- Während des zweiten Doktorantenkolloquium des Promotionszentrums für angewandte Informatiker der hessischen Hochschulen PZAI fand die Parallelveranstaltung CERC Conference 2019 statt, auf der ein Vortrag und wissenschaftliche Veröffentlichung zum Thema „Recommendations for Developing Safety-Related Systems with Graphical Languages“ veröffentlicht wurde [302].
- Zur SMSI 2020 wurde eine Veröffentlichung mit dem Titel „Safety Sensor Applications with Graphical Programming“ eingereicht, das für eine Posterpräsentation angenommen wurde. Leider fiel die Konferenz aufgrund der COVID-19-Pandemie aus. Das Short Paper ist dennoch online verfügbar [303].
- Im Mai 2020 sollte das IEEE Symposium ISPCE 2020 in Chicago (USA) stattfinden. Es wurde ein Vortrag und eine IEEE Veröffentlichung für dieses Event erstellt. Bedauerlicherweise fiel die Veranstaltung aufgrund der COVID-19-Pandemie aus. Als Ersatz wurden die Unterlagen der wissenschaftlichen Veröffentlichung an IEEE XPLORE und

Model-Based Systems Engineering sicherheitskritischer Anwendungen im Maschinenbau mittelständischer Unternehmen © 2024 by Nick Berezowski is licensed under CC BY 4.0

das IEEE Symposium SPCE 2020 in Portland (USA) weitergeleitet. Die Veröffentlichung mit dem Titel „Graphical Programming Languages for Functional Safety“ wurde für einen Vortrag an der SPCE 2020 angenommen. Dieser fand am 16. November 2020 online statt [304].

- Die Veröffentlichung „How to Use a Graphical Programming Language in Functional Safety, Using the Example of LabVIEW“ wurde auf der IEEE International RF and Microwave Conference (RFM) 2020 veröffentlicht, wozu am 16. Dezember 2020 ein Online-Vortrag auf der Konferenz gehalten wurde [305].
- Die Veröffentlichung „Graphical Programming Languages for Functional Safety using the example of LabVIEW“ wurde auf der IEEE ICSECC 2020 veröffentlicht. Zusätzlich wurde der Best Paper Award verliehen. Der Online-Vortrag fand am 16. Dezember 2020 mit großem Anklang statt [306].
- Die Veröffentlichung „A Brief Subset for Functional Safety in LabVIEW“ wurde auf der IEEE ICATMRI 2020 veröffentlicht und es wurde am 30. Dezember 2020 ein Online-Vortrag gehalten [307].
- Zuletzt wurde für die Echtzeit Konferenz der Gesellschaft für Informatik 2022 eine Veröffentlichung mit dem Namen „MBSE for SMEs with Domain-Specific Safety Analyses and Loose Tool Coupling“ geschrieben und in Springer veröffentlicht. Hierzu wurde am 11. November 2022 ein Vortrag gehalten [111].

A.2 Statement zum Themen und Betreuerwechsel

Im Jahr 2020, während des zweiten Jahres der Promotion, wurde ein Wechsel des Themas und des Betreuers veranlasst. Das ursprüngliche Thema, „Empfehlungen zur Anwendung von graphischen Programmiersprachen mit uneingeschränktem Sprachumfang in sicherheitsbezogenen Systementwicklungen“, wurde nach internen Besprechungen und unabhängigen Gutachten als nicht erfolgsversprechend eingestuft. Nach verschiedenen Rücksprachen mit der neuen Betreuung durch Prof. Dr. Reinhold Kröger wurde das Thema dieser Arbeit gewählt.

A.3 „OMG-Umfeld“

A.3.1 UML

Die Unified Modeling Language (UML) ist der de facto Standard für die Entwicklung von Software-Systemen und –Prozessen [121]. UML ist eine graphische Sprache, die Semantik und Notation für objektorientierte Problemlösung bereitstellt [25]. Die Verwendung von Modellen

ist für die Softwareentwicklung sowohl aus ingenieurtechnischer als auch aus kommunikativer Sicht wichtig, ähnlich wie bei Plänen, die von Architekten gezeichnet werden, bei der Konstruktion von Gebäuden [121]. Je komplexer das Gebäude, desto kritischer ist die Kommunikation zwischen Architekt und Bauherr sowie zwischen Architekt und Kunde.

Laut Holt [308] können verschiedene Typen von UML-Diagrammen für die Systementwicklung eingesetzt werden, um verschiedene Aspekte eines Systems oder Prozesses zu modellieren:

- Use-Case Diagramm: Ein Use-Case Diagramm dient zur Darstellung von Funktionen oder Funktionalitäten, die ein System bereitstellt, und den Akteuren, die diese Funktionalitäten nutzen.
- Klassendiagramm: Ein Klassendiagramm zeigt die Beziehungen zwischen Klassen und Objekten eines Systems. Es kann auch die Attribute und Methoden jeder Klasse darstellen.
- Package Diagramm: Ein Package Diagramm veranschaulicht die Organisation von Klassen und anderen Elementen in einem System in gruppierten „Packages“.
- Sequenzdiagramm: Ein Sequenzdiagramm zeigt die Interaktionen zwischen Objekten in einem System in der Zeit. Es veranschaulicht, wie Nachrichten zwischen Objekten übertragen werden.
- Zustandsdiagramm: Ein Zustandsdiagramm zeigt die möglichen Zustände und Übergänge eines Objektes im Laufe der Zeit.
- Aktivitätsdiagramm: Ein Aktivitätsdiagramm zeigt die Abläufe und Abhängigkeiten von Aktionen in einem System oder Prozess.
- Kompositionsstrukturdiagramm: Ein Kompositionsstrukturdiagramm zeigt die Beziehungen zwischen Teilen und Ganzen in einem System, wie z.B. die Beziehungen zwischen Komponenten einer Anwendung.

UML bietet somit bereits eine Vielzahl von Diagrammtypen für die Modellierung von Software-Systemen an, jedoch fehlen wichtige Funktionalitäten wie Anforderungsanalyse und -management, höhere Abstraktionsebene für Systemkomponentenbeziehungen und die Möglichkeit zur quantitativen Beschreibung von Systemkomponenteneigenschaften und -verhaltensweisen durch Parametrisierung, die für die Beschreibung von technischen Systemen notwendig werden.

A.3.2SysML

Die Systems Modeling Language (SysML) bietet Semantik und Notationen, um die Spezifikation, Analyse, Design, Verifikation und Validierung von technische Systementwicklungen zu unterstützen, die Hardware, Software, Daten, Parametrik, Personal, Prozesse und Einrichtungen einschließen [121]. Es ist ein auf Meta-Modell-Ebene definiertes Profil, das UML für technische Systementwicklungen und MBSE spezialisiert [309], wie in Abbildung A.1 beziehungstechnisch illustriert. SysML nutzt die gleichen primitiven Datentypen wie UML und bietet die Möglichkeit zur Erweiterung durch Standardprofile. Die SysML-Profile definieren Blocks, Aktivitäten, Anforderungen, Allokationen und Wechselwirkungen detaillierter. Außerdem besitzen die SysML-Bibliotheken zusätzliche primitive Value Types, Control Values, Unit Kinds, und Quantity Kinds [309]. SysML soll dadurch eine verbesserte Unterstützung für Spezifikation, Design, Analyse, Verifikation und Validierung bieten [309]. SysML unterstützt ebenso den Austausch von Modellen und Daten über XMI und AP233 [128], wodurch es sich besonders für MBSE eignet.

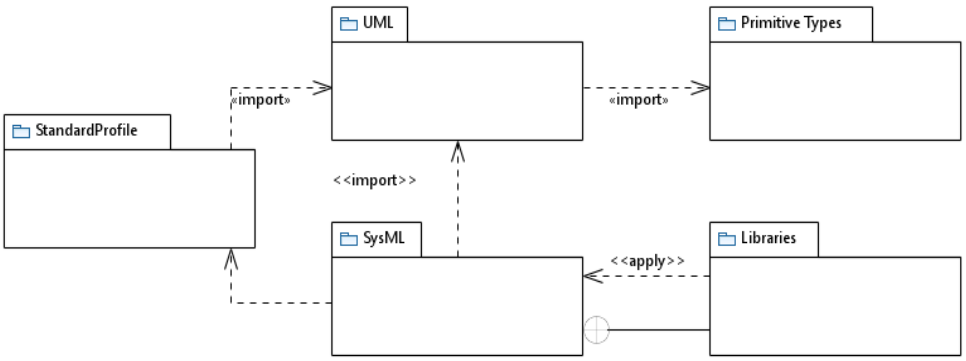


Abbildung A.1: SysML Erweiterung von UML [26]

Abbildung A.2 zeigt die Arten von Diagrammen, die von SysML verwendet werden. Es existieren drei primäre Typen von Diagrammen auf der höchsten Hierarchieebene: Verhalten, Struktur und Requirements [26].

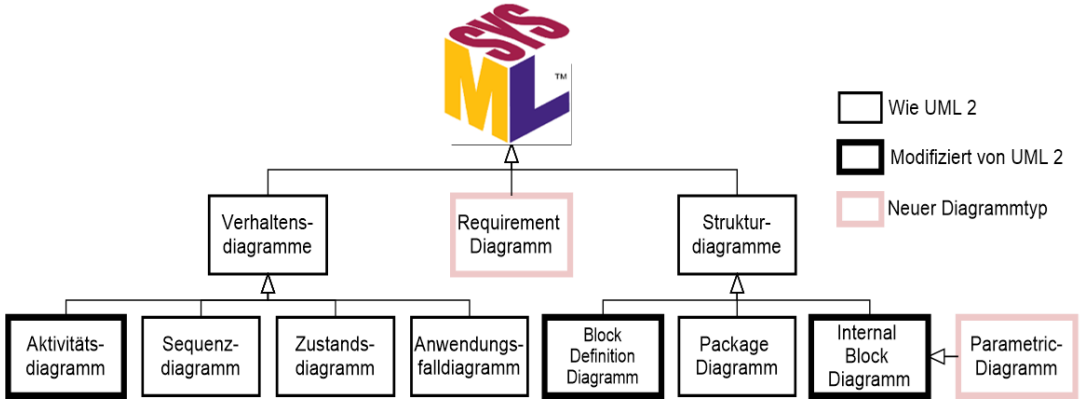


Abbildung A.2: SysML Diagrammtypen [22]

Der neue Typ des Requirement Diagrams definiert Systemanforderungen direkt auf Modell-Ebene und verknüpft sie mit Verhaltens- und Strukturdiagrammen [309]. Auf der nächsten Ebene macht SysML spezialisierte Modifikationen an UML-Aktivitätsdiagrammen und hat signifikante Modifikationen an UML-Klassendiagrammen vorgenommen, die durch UML-Kompositionsstrukturen erweitert wurden, um Blockdiagramme erstellen zu können. Das Parametric Diagramm ermöglicht die Definition mathematischer Beziehungen zwischen den Teilen des zu entwerfenden Systems und kombiniert insbesondere mathematische Formeln für die Analyse kritischer Systemparameter. Falls korrekt verwendet, simulieren Parameterdiagramme den Inhalt von Blockdefinitionsdiagrammen (BDD) und Internal-Blockdiagrammen (IBD) [309]. Während das BDD eine Erweiterung des Klassendiagramms ist, stellt das IBD ein Blockstrukturdiagramm innerhalb eines BDD dar [309].

Das Blockkonzept bietet die Möglichkeit, komplexe Systeme durch die Definition von Blöcken und deren Beziehungen zu modellieren. Ein Block repräsentiert dabei einen Teil des Systems, der bestimmte Eigenschaften und Verhaltensweisen besitzt. Funktionalitäten des Blockkonzepts nach [26] sind:

- Eigenschaften: Blöcke können verschiedene Eigenschaften haben, die deren Verhalten oder Interaktionen mit anderen Blöcken beeinflussen.
- Ports: Blöcke können Ein- und Ausgangs-Ports besitzen, um mit anderen Blöcken zu kommunizieren.
- Verbindungen: Verbindungen können zwischen den Ports von Blöcken hergestellt werden, um die Kommunikation und Interaktion zwischen ihnen zu ermöglichen.
- Generalisierung: Blöcke können von anderen Blöcken abgeleitet werden, um ähnliche Eigenschaften oder Verhaltensweisen zu erben.
- Teile: Ein Block kann andere Blöcke als Teile beinhalten, um komplexe Systeme aufzubauen.
- Wert-Eigenschaften: Blöcke können auch Eigenschaften haben, die einen bestimmten Wert besitzen.
- Parametrik: Parameter können verwendet werden, um die Eigenschaften oder das Verhalten von Blöcken zu spezifizieren.

SysML wurde schnell zu einem Standard für die Modellierung und Analyse von Systemen und hat sich zu einem wichtigen Instrument für Ingenieure und Organisationen entwickelt, die komplexe Systeme entwickeln und analysieren müssen [310]. Dies ermöglicht es Ingenieuren über

die MBSE-Strukturen, komplexe Systeme visuell darzustellen, zu analysieren und zu simulieren, bevor sie tatsächlich gebaut werden, was Zeit und Kosten spart und zu besseren Ergebnissen führt [128].

Inzwischen wird SysML weit verbreitet in verschiedenen Domänen und Anwendungsbereichen eingesetzt, darunter Luft- und Raumfahrt oder Automobilbau [128, 311]. Es hat sich als ein leistungsfähiger Standard für die Modellierung und Analyse von Systemen etabliert und wird weiterhin von der OMG und anderen Organisationen weiterentwickelt und erweitert, um den sich ändernden Bedürfnissen von Ingenieuren und Organisationen gerecht zu werden [21].

A.3.3MOF

Der Standard der OMG für das Verwalten von Metadaten ist die Meta Object Facility (MOF) [122]. Viele von der OMG standardisierte Technologien, wie UML [118], Business Process Model and Notation (BPMN) [312], Software Process Engineering Metamodel (SPEM) [313] und verschiedene UML-Profile, nutzen MOF oder von MOF abgeleitete Technologien, insbesondere XMI [125], für den Austausch und die Verarbeitung von Modellinformationen, die sich auf Metadaten konzentrieren.

Die 4-Schichten-Architektur kann wie folgt beschrieben werden:

- Die Information-Schicht (M0), die die zu beschreibenden Daten aus der realen Welt enthält.
- Die Model-Schicht (M1), die die Informationen aus der realen Welt beschreibt und klassifiziert, indem sie Metadaten generiert.
- Die Metamodel-Schicht (M2), die die Struktur und Semantik von Modellen beschreibt und klassifiziert, indem sie Meta-Metadaten generiert.
- Die Meta-Metamodel-Schicht (M3), die die Struktur und Semantik von Meta-Metadaten beschreibt und klassifiziert.

Wir können die 4-Schicht-Architektur auch als umgekehrte Hierarchie betrachten, wie in Abbildung A.3 veranschaulicht. Hierbei können Daten aus einer Schicht als Instanz einer Informationseinheit aus der übergeordneten Schicht verstanden werden. Die Beziehung zwischen den Schichten wird oft mithilfe des Schlüsselbegriffs „instanceOf“ definiert.

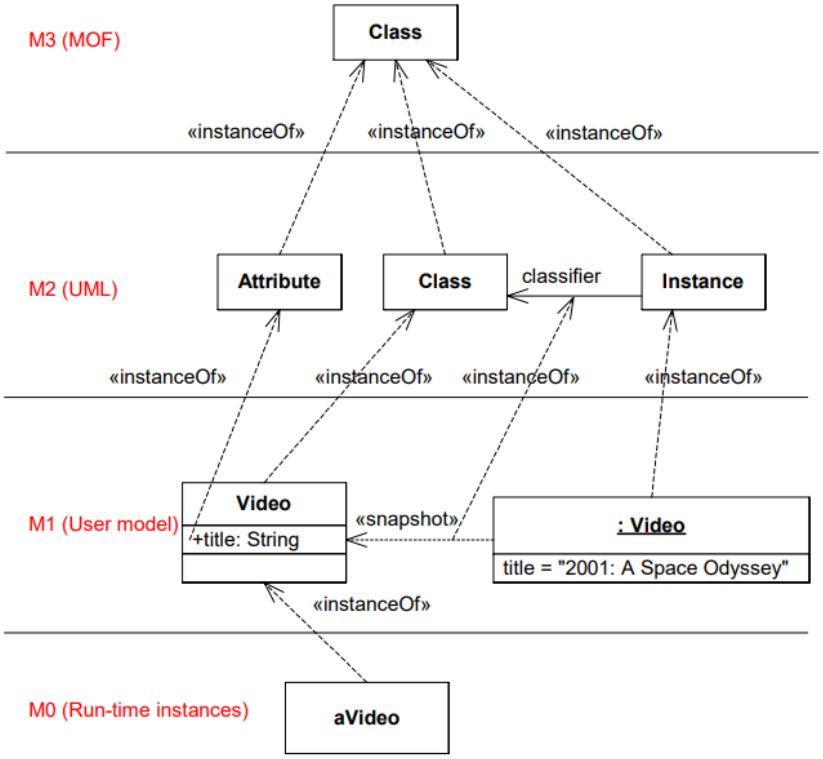


Abbildung A.3: 4-Schichten-Metadaten-Architektur abgeleitet von [314]

Die OMG verwendet MOF als standardisierte Metasprache, um andere Modellierungssprachen wie UML zu spezifizieren. Dies macht MOF zum Standard für die M3-Schicht der 4-Schichten-Architektur für Metadaten und legt die Beziehung zum UML-Standard.

A.3.4Profile von Profilen

Es ist möglich, ein UML-Profil oder ein SysML-Profil auf ein anderes Profil anzuwenden [26]. Indem man ein Profil auf ein anderes anwendet, kann man neue Modellierungskonstrukte definieren, die auf den Konstrukten des Basis-Profils aufbauen. Diese Technik wird verwendet, um eigene Anforderungen und Konventionen zu definieren, die über die Standard-Konstrukte von UML oder SysML hinausgehen.

Wenn man ein Profil auf ein anderes anwendet, kann man es erweitern oder ändern, um es an die spezifischen Bedürfnisse eines Projekts oder einer Domäne anzupassen. Ein Beispiel könnte sein, dass man ein UML-Profil für die Modellierung von Geschäftsprozessen erstellt und dieses Profil dann auf ein anderes UML-Profil für die Modellierung von Softwareanwendungen anwendet. Auf diese Weise kann man die Geschäftsprozesse in einem Software-Kontext modellieren.

Ein weiteres Beispiel könnte sein, dass man ein SysML-Profil für die Modellierung von elektrischen Systemen erstellt und dieses Profil dann auf ein UML-Profil für die Modellierung von

sicherheitskritischen Systemen anwendet. Auf diese Weise kann man die spezifischen Anforderungen an die Sicherheit von elektrischen Systemen modellieren.

Es ist wichtig zu beachten, dass bei der Anwendung von Profilen aufeinander die Vereinbarkeit der Konstrukte sichergestellt sein muss, um sinnvolle Ergebnisse zu erzielen. Außerdem kann es notwendig sein, die Regeln für die Verwendung der Konstrukte, beispielsweise durch Object Constraint Language (OCL) innerhalb des erweiterten Profils zu definieren.

A.3.5OCL

Die Object Constraint Language wurde ursprünglich als Ergänzung zur UML-Notation entwickelt, um die Einschränkungen von UML und allgemein jeder graphischen Notation bei der präzisen Spezifikation detaillierter Aspekte eines Systems zu überwinden [123]. Die von der OMG definierte Sprache zur Beschreibung von Constraints oder Einschränkungen in der Modellierung kann als Teil einer UML-Modellierung oder als eigenständige Sprache verwendet werden [297]. Sie wird eingesetzt, um die Integrität, die Kohärenz und die Konsistenz von Modellen zu gewährleisten, wodurch sie ein wesentlicher Bestandteil jeder modellgetriebenen Entwicklung darstellt und als Standard-Sprache für die Ausdrucksform aller Arten von (Meta-)Modell-Abfrage, Manipulation und Spezifikationsanforderungen verwendet wird [123]. Einsatz findet sie oft, um Modellumwandlungen auszudrücken, Regeln für die Kohärenz aufzustellen oder Code-Generierungsvorlagen zu generieren [123]. Zum Beispiel kann man mit OCL angeben, dass eine bestimmte Eigenschaft eines Objektes immer positiv sein muss oder dass zwei bestimmte Objekte eine eindeutige Beziehung haben müssen. Die Verwendung von OCL ermöglicht es, die Modellierung von Systemen zu automatisieren und zu überprüfen, um sicherzustellen, dass die erforderlichen Anforderungen erfüllt werden.

A.3.6CSS

Cascading Style Sheets (CSS) ist eine Sprache, die zur Gestaltung verwendet wird. Mit CSS kann man das Aussehen von Elementen wie Text, Bildern, Tabellen und Formularen gestalten und formatieren [315]. Durch die Trennung von Inhalt und Darstellung kann man das Design komplett unabhängig vom Inhalt erstellen und so die Wartbarkeit und Flexibilität verbessern. CSS ermöglicht die Verwendung von verschiedenen Selektoren, Eigenschaften und Werten, um Elemente zu identifizieren und ihnen verschiedene Stile zuzuweisen. Mit CSS kann man unter anderem Schriftarten, Farben, Hintergrundbilder, Abstände und Positionen von Elementen definieren [315].

In der visuellen Darstellung von Modellen und Diagrammen in Modellierungswerkzeugen für UML und SysML ermöglicht CSS die Formatierung, Gestaltung und Anpassung des visuellen Erscheinungsbilds der Diagramme [124]. In diesem Kontext wird CSS Syntax verwendet, um das Aussehen und das Layout von Elementen in UML- und SysML-Diagrammen anzupassen [124]. Die Integration ist jedoch von Werkzeug zu Werkzeug unterschiedlich und hängt von den Funktionen und Möglichkeiten des jeweiligen Werkzeugs ab.

A.3.7XMI

XML Metadata Interchange ist ein Standard, der es ermöglicht, mit Metadaten, die in der Regel in Modellierungswerkzeugen erstellt werden, zu interagieren und diese auszutauschen [125]. XMI basiert auf der eXtensible Markup Language (XML), die es ermöglicht, die Daten in einer lesbaren und verständlichen Form über XML-Elemente und Attribute zu formatieren und zu speichern [125]. Der Standard der OMG beinhaltet Standardmechanismen zur Verknüpfung von Objekten innerhalb der gleichen Datei oder zwischen verschiedenen Dateien und die Validierung von XMI-Dokumenten mithilfe von XML-Schemata. Die entstehende Objektidentität ermöglicht es, Objekte anhand von Identifier (IDs) und Universally Unique Identifier (UUIDs) von anderen Objekten zu referenzieren.

XMI wird in der Softwareentwicklung und Modellierung häufig verwendet, um Modelle zwischen verschiedenen Werkzeugen und Systemen zu übertragen und zu synchronisieren, ohne dass dabei Informationen verloren gehen. XMI ermöglicht es auch, Modellierungsinhalte, wie Klassendiagramme, Aktivitätsdiagramme, Zustandsdiagramme und andere Artefakte, in einer einheitlichen Form zu speichern und zu teilen.

A.3.8EMF

Das von Eclipse entwickelte Eclipse Modeling Framework-Projekt beschreibt ein Modellierungs-Framework und Codegenerierungswerkzeug für die Entwicklung von Anwendungen auf Basis eines strukturierten Datenmodells [316]. Die Modellspezifikation wird in XMI beschrieben und EMF bietet Werkzeuge und Laufzeitunterstützung, um eine Reihe von Java-Klassen und Adapter-Klassen zu generieren [317]. Diese Klassen ermöglichen eine Anzeige und eine Befehlsbasierte Bearbeitung des Modells, sowie einen grundlegenden Editor [317]. EMF ist ein weit verbreiteter Standard für Datenmodelle und wird von vielen Technologien und Frameworks genutzt, einschließlich Server-Lösungen, Persistenz-Frameworks, UI-Frameworks und Transformationen [316].

A.3.9MBSE-Plattformen

Eine MBSE-Plattform ist eine integrierte Softwareumgebung, die speziell für die Modellierung von Systemen und Prozessen konzipiert wurde. Sie bietet Werkzeuge und Funktionen für die Modellierung, Simulation, Analyse und Überprüfung von Systemen, einschließlich Hardware, Software, Daten, Prozeduren und Personen. Diese Plattformen unterstützen die MBSE-Methodik, indem sie die Zusammenarbeit von Stakeholdern erleichtern und die Übertragbarkeit von Modellen und Daten fördern. Einige der bekanntesten Plattformen für MBSE, mit Unterstützung für UML und SysML sind auf dem offiziellen SysML-Forum [318] zu finden und können folgendermaßen zusammengefasst werden:

- Papyrus [93, 318, 319]:
    - Entwickelt von Eclipse
    - Leicht zu zeichnende SysML-Diagrammnotation
    - Einfache XMI-Interoperabilität
    - Kostenlos und OpenSource
- SysML Architect [92, 318]:
    - Entwickelt von Modelio
    - Leicht zu zeichnende SysML-Diagrammnotation
    - automatische Erstellung von Dokumentationen
    - Kostenlos und OpenSource
- Rational Rhapsody [148, 318]:
    - Entwickelt von IBM
    - eines der am weitesten verbreiteten Werkzeuge für MBSE
    - starke Integration mit anderen IBM-Werkzeugen und –Produkten (DOORS)
    - Kostenpflichtig
- MagicDraw [91, 318]:
    - Entwickelt von NoMagic
    - Breite Palette an Modellierungsmethoden
    - größere Anzahl von integrierten Werkzeugen und Funktionen
    - Kostenpflichtig
- Enterprise Architect [318, 320]:
    - Entwickelt von Sparx Systems
    - Breite Palette an Modellierungsmethoden
    - Benutzerfreundlich
    - Kostenpflichtig, doch günstig

- Windchill Modeler [318, 321]:
    - Entwickelt von PTC
    - Integriertes Requirement-Management
    - Integrierte Entwicklungsumgebung und Berichtsfähigkeit
    - Kostenpflichtig

A.4 Bestimmung der Grenzen der Maschine

Zur Bestimmung der Grenzen einer Maschine können laut ISO 12100 [163] die folgenden unter A.4.1 bis A.4.5 beschriebenen Eigenschaften einer Maschine in die Betrachtungen eingebunden werden.

A.4.1 Verwendungsgrenzen

- Bestimmungsgemäße Verwendung
- Vorhersehbare Fehlanwendung
- Verschiedene Betriebsarten
- Unterschiedliche Eingriffsmöglichkeiten
- Einsatzbereich der Maschine in Bezug auf die Eigenschaften des Anwenders
- Niveau der Benutzer in Hinblick auf Ausbildung, Erfahrung, Fähigkeiten
- Weitere Personen in Gefahrenbereich

A.4.2 Räumliche Grenzen

- Bewegungs-/Verfahrbereiche inkl. Sicherheitsabstände
- Platzbedarf von Personen, Arbeitsplätze/-flächen
- Materialbereitstellung/-abfuhr
- Schnittstellen

A.4.3 Zeitliche Grenzen

- Grenzen der Lebensdauer der Maschine oder von Bauteilen
- Empfohlene Prüffristen, Wartungs-, Instandsetzungsintervalle

A.4.4 Weitere Grenzen

- Ausgangsstoffe, Hilfs-, Betriebsstoffe, Abprodukte
- Umgebungsbezogen (Temperatur, Sauberhaltung, Witterung usw.)

A.4.5 Information

- Schnittstellen
- Ein-/Ausgaben
- Übergeordnete Steuerkreise

A.5 Gefährdungsgruppen

Die in dieser Arbeit beschriebenen Gefährdungsgruppen und unter A.5.1 bis A.5.10 zusammengefassten möglichen Gefahren in den Gefährdungsgruppen, basierend auf ISO 12100 [163] können genutzt werden, um die Gefährdungen einer Maschine zu beschreiben und die Grundlage für die Risikobeurteilung in Form von Gefährdungssituationen der Gefährdungen und zugeordneter Effekte zu schaffen. Unter A.5.11 finden sich zusätzlich Beispiele mit zugeordneten Effekten, um das Verständnis der Zusammenhänge zu steigern.

A.5.1 Mechanische Gefährdungen

Mechanische Gefährdungen sind Gefahren, die von Maschinen, Geräten oder Anlagen ausgehen, die durch ihre Konstruktion, ihren Betrieb oder ihre Umgebung Verletzungen oder Gesundheitsschäden verursachen können.

A.5.2 Elektrische Gefährdungen

Elektrische Gefährdungen sind potenzielle Risiken, die von elektrischen Anlagen, Geräten und Maschinen ausgehen können.

A.5.3 Thermische Gefährdungen

Thermische Gefährdungen sind potenzielle Risiken für die Gesundheit und Sicherheit von Personen, die mit thermischen Prozessen oder Materialien arbeiten. Thermische Gefährdungen können durch eine Vielzahl von Quellen verursacht werden, wie z.B. heißes Metall, Dampf, Flüssigkeiten, Gase oder Flammen.

Zu den möglichen Auswirkungen von thermischen Gefährdungen gehören Verbrennungen, Verbrühungen, Hitzeschlag, Dehydration und Hitzekrämpfe. Die Schwere der thermischen Gefährdung hängt von Faktoren wie der Temperatur, der Dauer der Exposition, dem Körperbereich, der betroffen ist, und dem individuellen Gesundheitszustand ab.

Zur Minimierung von thermischen Gefährdungen sollten geeignete Schutzausrüstungen wie hitzebeständige Handschuhe, Schutzbrillen oder Schutzkleidung getragen werden. Auch eine korrekte Schulung der Mitarbeiter im Umgang mit den thermischen Gefahrenquellen sowie eine

regelmäßige Wartung und Überprüfung von Maschinen und Anlagen können dazu beitragen, das Risiko zu minimieren.

A.5.4 Gefährdungen durch Lärm

Gefährdungen durch Lärm können sich auf die Gesundheit und Sicherheit von Personen auswirken. Lärm kann Schallpegel erzeugen, die das Gehör schädigen, wie zum Beispiel bei langfristiger Exposition gegenüber Lärm in Arbeitsumgebungen wie Baustellen oder Fabriken. Dies kann zu Gehörschäden führen, die sowohl vorübergehend als auch dauerhaft sein können. Lärm kann auch stressig sein und zu Müdigkeit, Konzentrationsproblemen und Beeinträchtigung der Kommunikation führen, was wiederum das Risiko von Arbeitsunfällen erhöht. Es gibt auch gesetzliche Vorschriften für den maximal zulässigen Schallpegel in Arbeitsumgebungen, um das Risiko von Gehörschäden und anderen gesundheitlichen Auswirkungen zu minimieren.

A.5.5 Gefährdungen durch Vibration

Gefährdungen durch Vibration treten auf, wenn ein Körper in Schwingungen versetzt wird und dadurch mechanische Erschütterungen entstehen. Diese Erschütterungen können zu körperlichen Beschwerden führen und langfristig Schäden verursachen. Häufige Ursachen für Vibrationen sind beispielsweise Maschinen, Fahrzeuge, Werkzeuge oder auch der Gebrauch von Handys oder Tablets. Zu den möglichen gesundheitlichen Auswirkungen von Vibrationen zählen beispielsweise Schmerzen in Gelenken und Muskeln, Durchblutungsstörungen, Verletzungen an Sehnen und Nerven sowie Hörverlust. Auch psychische Belastungen wie Stress und Erschöpfung können durch Vibrationen verursacht werden.

A.5.6 Gefährdungen durch Strahlungen

Gefährdungen durch Strahlung können in verschiedenen Arbeitsbereichen und Situationen auftreten, z.B. bei der Arbeit mit ionisierender Strahlung in der Materialprüfung. Aber auch bei der Arbeit mit nicht-ionisierender Strahlung, wie z.B. elektromagnetischer Strahlung oder optischer Strahlung, können Gefährdungen auftreten. Gefährdungen durch Material und Substanzen

A.5.7 Gefährdungen durch Materialien und Substanzen

Gefährdungen durch Materialien und Substanzen bei Maschinen beziehen sich auf potenzielle Risiken, die von Stoffen ausgehen können, die bei der Bedienung, Wartung oder Reparatur von

Maschinen verwendet werden. Dazu gehören beispielsweise chemische Stoffe wie Lösungsmittel, Schmierstoffe, Reinigungsmittel und Farben, aber auch Materialien wie Metallspäne, Sägemehl und andere Abfälle, die während des Betriebs der Maschine entstehen können.

A.5.8Ergonomische Gefährdungen

Ergonomische Gefährdungen beziehen sich auf Belastungen des menschlichen Körpers, die durch eine ungünstige Gestaltung von Arbeitsplätzen, Arbeitsabläufen und Arbeitsmitteln entstehen können. Dazu gehören zum Beispiel eine schlechte Körperhaltung, eine falsche Sitzposition, wiederholte Bewegungen und ungünstige Lichtverhältnisse. Langfristig können solche Belastungen zu Gesundheitsschäden führen, wie zum Beispiel Rückenbeschwerden, Sehstörungen, Durchblutungsstörungen oder psychischen Belastungen. Um ergonomischen Gefährdungen vorzubeugen, müssen Arbeitsplätze und Arbeitsmittel ergonomisch gestaltet und die Arbeitsabläufe so organisiert werden, dass sie den Bedürfnissen und Fähigkeiten der Mitarbeiter entsprechen.

A.5.9Gefährdungen im Zusammenhand mit der Einsatzumgebung

Gefährdungen im Zusammenhang mit der Einsatzumgebung beziehen sich auf potenzielle Risiken und Schäden, die durch die Umgebung, in der eine Maschine oder Anlage betrieben wird, verursacht werden können. Hier sind einige Beispiele für solche Gefährdungen:

- Witterungsbedingungen: Eine Maschine oder Anlage kann durch extreme Witterungsbedingungen wie Hitze, Kälte, Feuchtigkeit oder Wind beschädigt werden. Die Arbeitnehmer, die mit der Maschine arbeiten, können auch durch diese Bedingungen gefährdet werden.
- Chemikalien: Der Einsatz von Maschinen und Anlagen in Bereichen, in denen Chemikalien oder andere gefährliche Substanzen verwendet werden, kann zu Gefährdungen führen. Die Arbeitnehmer können diesen Stoffen ausgesetzt sein und Verletzungen oder Gesundheitsschäden erleiden.
- Staub und Schmutz: In einigen Einsatzumgebungen kann eine Maschine oder Anlage durch Staub oder Schmutz verstopft oder beschädigt werden. Die Arbeitnehmer können durch die Exposition gegenüber diesen Materialien auch gesundheitlichen Risiken ausgesetzt sein.
- Elektromagnetische Felder: Elektrische oder elektronische Geräte können elektromagnetische Felder erzeugen, die sowohl für die Maschine als auch für die Arbeitnehmer in der Umgebung eine Gefahr darstellen können.

- Schwingungen und Erschütterungen: Eine Maschine oder Anlage, die in der Nähe von Schwingungsquellen wie Verkehr oder schweren Maschinen betrieben wird, kann durch Vibrationen und Erschütterungen beeinträchtigt werden. Die Arbeitnehmer, die in der Nähe dieser Maschinen arbeiten, können auch durch die Vibrationen gefährdet werden.
- Platzmangel: Eine Maschine oder Anlage, die in einem engen oder begrenzten Raum betrieben wird, kann zu Gefährdungen führen. Die Arbeitnehmer können Schwierigkeiten haben, sich sicher um die Maschine herumzubewegen, und es kann schwierig sein, im Notfall schnell zu reagieren.
- Licht: In einigen Einsatzumgebungen kann schlechtes Licht die Sichtbarkeit beeinträchtigen und zu Unfällen führen. Umgekehrt kann auch zu helles Licht zu Augenschäden führen.

A.5.10 Kombination von Gefährdungen

Eine Kombination von Gefährdungen tritt auf, wenn mehrere Gefährdungsarten gleichzeitig auftreten oder sich gegenseitig verstärken. Zum Beispiel kann eine Arbeitssituation, die Lärm und Vibrationen beinhaltet, zu einer höheren Belastung für den Arbeitnehmer führen als eine Situation, die nur Lärm oder nur Vibrationen beinhaltet. Eine Kombination von Gefährdungen kann auch auftreten, wenn eine Arbeitsumgebung thermische, mechanische und ergonomische Risiken birgt. In solchen Fällen müssen Gefährdungen berücksichtigt werden, um ein vollständiges Verständnis der Risiken zu erhalten und geeignete Maßnahmen zur Risikominderung zu ergreifen.

A.5.11 Gefährdungsgruppen mit Beispielen

Gruppe	Beispiele für Gefährdungen
Mechanische Gefährdungen	• Beschleunigung/Abbremsung • spitze Teile • Annäherung eines sich bewegenden Teils an ein feststehendes Teil • schneidende Teile • elastische Elemente • herabfallende Gegenstände • Schwerkraft • Höhe gegenüber dem Boden • Hochdruck • fehlende Standfestigkeit/-sicherheit • kinetische Energie • Beweglichkeit der Maschine • sich bewegende Teile • rotierende Teile • raue, rutschige Oberfläche • scharfe Kanten • Vakuum

Gruppe	Beispiele für Gefährdungen
Elektrische Gefährdungen	• Lichtbogen • elektromagnetische Vorgänge • elektrostatische Vorgänge • spannungsführende Teile • unzureichender Abstand zu unter Hochspannung stehenden Teilen • Überlast • Teile, die im Fehlerzustand spannungsführend geworden sind • Kurzschluss • Wärmestrahlung
Thermische Gefährdungen	• Explosion • Flamme • Objekte oder Materialien hoher oder niedriger Temperatur • Strahlung von Wärmequellen
Gefährdungen durch Lärm	• Kavitationsvorgänge • Abluftsystem • mit hoher Geschwindigkeit austretendes Gas • Herstellungsprozess (Stanzen, Schneiden usw.) • bewegliche Teile • reibende Flächen • mit Unwucht rotierende Teile • pfeifende Pneumatik-Einrichtungen; • verschlissene Teile
Gefährdungen durch Vibration	• Kavitationsvorgänge • Fehlausrichtung sich bewegender Teile • bewegliche Ausrüstung • reibende Flächen • mit Unwucht rotierende Teile • schwingende Ausrüstung • verschlissene Teile
Gefährdungen durch Strahlung	• ionisierende Strahlungsquelle • niederfrequente elektromagnetische Strahlung • optische Strahlung (infrarot, sichtbar und ultraviolett), einschließlich Laserstrahlen • hochfrequente elektromagnetische Strahlung
Gefährdungen durch Materialien und Substanzen	• Aerosol • biologische und mikrobiologische (virale oder bakterielle) Substanz • Brennstoff • Staub • Explosivstoff • Fasern • feuergefährliches Material • Flüssigkeit • Dämpfe • Gas • Oxidationsmittel
Ergonomische Gefährdungen	• Zugang • Gestaltung oder Anordnung von Anzeigen und optischen Displays • Gestaltung, Anordnung oder Erkennung von Steuerungseinrichtungen • Flackern, Blenden, Schattenbildung und stroboskopische Effekte • örtliche Beleuchtung • Anstrengung, psychische Überbelastung/Unterforderung • Körperhaltung • sich wiederholende Tätigkeiten • Sichtbarkeit
Gefährdungen im Zusammenhang mit der Einsatzumgebung	• Staub und Nebel • elektromagnetische Störungen • Blitzschlag

Gruppe	Beispiele für Gefährdungen
	• Feuchtigkeit • Verunreinigungen • Schnee • Temperatur • Wasser • Wind • Sauerstoffmangel
Kombination von Gefährdungen	• sich wiederholende Tätigkeit + Anstrengung + hohe Umgebungstemperatur • Fehlverhalten bei Ansteuerung der Anlage

Tabelle A.1: Beispiele für Gefährdungen [163]

A.6 Analysemethoden

A.6.1 FMEA

Die Ausfallarten- und Auswirkungsanalyse, englisch Failure Mode and Effect Analysis (FMEA), ist eine Analyse, die verwendet wird, um systematisch die Konsequenzen einzelner Ausfälle verschiedener Typen zu bestimmen und zu bewerten [11]. Sie kann als relativ fundamentale Analyse betrachtet werden, was sich auch im Aufbau innerhalb von RAAML äußert, in der sie hauptsächlich aus den bereits im General Concepts definierten Elementen besteht, ohne diese weitgehend zu spezifizieren. Als fundamentale Analyse kann sie nahezu in allen Domänen eingesetzt werden. Bei FMEA handelt es sich um ein kombiniertes Konzept, welches analytische Verfahren zur Gefährdungsermittlung einsetzt und eine Risikoabschätzung mittels qualitativer Beschreibungen vornimmt. Einerseits bietet sie dabei eine Risikoanalyse zur Optimierung durch Fehlerprävention und andererseits ermöglicht sie auch den Aufbau eines Wissensrepositoriums für Fehlerarten und entsprechende Schutzmaßnahmen [39]. Die Durchführung von FMEAs ist ein entscheidender Schritt in der Entwicklung sicherheitskritischer Systeme. Hecht et al. [322] zeigen, wie die Automatisierung von FMEAs mithilfe von SysML die Effizienz und Genauigkeit solcher Analysen erheblich verbessern kann.

Nach der FMEA muss jede Komponente eines Systems der Reihe nach analysiert werden [202]. Zunächst werden die Fehlermodi, ihre Ursachen und Auswirkungen bestimmt, anschließend werden die Verfahren zur Erkennung ermittelt und schließlich können die Stakeholder Empfehlungen aussprechen [202]. Die Reaktionen auf die Empfehlungen werden als Maßnahmen zur Behebung der Ursache betrachtet [202]. Stakeholder können im Nachhinein den DC und den SFF eines Elements, wie in [6] beschrieben, berechnen. Diese Werte werden bei der Komponentenabschätzung und bei der Überprüfung des Sicherheitsdesigns durch DFTs oder SBBM verwendet. Die Norm IEC 60812 [202] gibt einen tieferen Einblick in die Erstellung von FMEA

und die artverwanden Methoden FMECA zur Kritikalitätsanalyse und FMEDA zur Diagnoseanalyse. FMECA und FMEDA werden hauptsächlich in der Automobil- und Luft- und Raumfahrtindustrie eingesetzt, wobei sie hierauf nicht beschränkt sind.

FMEA kann in enger Verbindung mit anderen Analysen, wie einer FTA oder ETA stehen [202]. Wenn eine FMEA durchgeführt wird, ist deshalb eine Abstimmung zwischen Analysen dringend erforderlich, um unnötigen Mehraufwand zu vermeiden. Die Arbeit von Hecht et al. [322] illustriert die Entwicklung und Anwendung eines SysML-Plugins zur engen Modellkopplung von FMEAs. Diese Automatisierung reduziert den manuellen Aufwand und ermöglicht eine kontinuierliche und effiziente Überprüfung des Systemdesigns auf potenzielle Fehler und Schwachstellen. Der RAM Commander [241] von ALD stellt hingegen ein externes Analysewerkzeug zur Berechnung von FMEA dar [323]. Er unterstützt neben FMEA unter anderem FTA, ETA und RBD, wodurch zwischen den lose gekoppelten Analysen Anhängigkeiten festgelegt werden können. Aber auch die von RAAML spezifizierten Profile und Bibliotheken mit der Berechnung über Constraint-Blöcke stellen ein Werkzeug mit Kopplung der Analysewerkzeuge dar, wohingegen hier eine enge Kopplung der Modelle und eine enge Kopplung der Analysewerkzeuge umgesetzt wurde [49].

Die Methode FMEA wurde im Rahmen dieser Arbeit nicht tiefergehend behandelt, da sie bereits hinreichend bekannte Modellierungstechniken besitzt und bereits Bestandteil des RAAML-Frameworks ist. Ähnliches gilt für FMECA. Zusätzlich besitzt FMECA eine geringe Relevanz für die Maschinenindustrie und führt zu weit für den Rahmen dieser Arbeit. Nichts desto trotz existieren verschiedene Arbeiten, wie [232, 233, 324], die die Integration von FMEA und FMECA in MBSE-Prozesse bearbeiten. [233] bietet einen innovativen Ansatz zur Verbesserung der Analyse innerhalb des Systems Engineering, indem traditionelle FMEA-Aktivitäten in eine umfassendere, modellbasierte Entwicklungsstrategie integriert werden können, was die Effizienz und Effektivität der Analyseprozesse steigert und gleichzeitig die Sicherheit und Zuverlässigkeit des Systemdesigns unterstützt.

A.6.2ETA & UWD

Die Ereignisbaumanalyse (ETA) stellt ähnlich eines Ursache-Wirkungsdiagramms (UWD) oder einer FTA die Abfolge von Ereignissen in einem Modell dar [11]. Während ein UWD die Abfolge von Ereignissen als die Wirkung der Kombination von verschiedenen Basisereignissen aufzeigt, analysiert eine ETA die Reaktion und Abfolge von Ereignissen, basierend auf einem

Startereignis [325]. Das Ziel einer ETA stellt somit die Auswirkungsanalyse eines bereits ausgefallenen oder noch funktionsfähigen Systems, auf Basis eines speziellen Ereignisses dar [326]. Dabei kann ein UWD allerdings als Grundlage dienen [11].

Zu Beginn der ETA stehen die Ablaufbedingungen vom Startereignis zu den möglichen Folgeereignissen [11]. Eine Linie verbindet das Startereignis mit der ersten Bedingung des Ablaufs, an der sich das Diagramm in einen binären Zweig teilt, an dessen Ende wieder Ereignisse stehen. Neben der Wahrscheinlichkeit eines Aufrufs spielt hier oft auch die Anzahl von Aufrufen eine Rolle, was die Abhängigkeiten verschiedener Bedingungen und deren Berechnungen zusätzlich verkomplizieren kann [11].

Die Methode ETA und UWD besitzen im Rahmen dieser Arbeit eine geringe Relevanz und wurden daher nicht weiter tiefergehend verfolgt.

A.6.3GSN

Die Goal Structuring Notation (GSN) ist eine formale Methode zur Strukturierung von Argumenten, die verwendet wird, um sicherheitskritische Systeme zu analysieren und zu bewerten [207]. Sie wurde ursprünglich von John Rushby entwickelt und seit 2011 als internationaler Standard IEC 62531 anerkannt.

GSN ermöglicht es, die Beziehungen zwischen den verschiedenen Zielen und Argumenten, die in einem Argumentationsprozess verwendet werden, klar und präzise darzustellen. Es besteht aus einer Reihe von Diagrammen und Notationen, die es ermöglichen, den Pfad von den obersten Zielen hinunter zu den Unterzielen und schließlich zu den Nachweisen oder Beweisen zu verfolgen, die das System sicher machen [207].

Ein typisches GSN-Diagramm besteht aus einem zentralen Ziel, das durch ein Oval dargestellt wird, das von mehreren Unterzielen umgeben ist, die durch Rechtecke dargestellt werden. Diese Unterziele können weitere Unterziele enthalten, die zu bestimmten Nachweisen führen, die das Systemdesign unterstützen [207].

A.6.4STPA

System Theoretic Process and Analysis (STPA) ist eine Methode zur systematischen Analyse von Sicherheitsproblemen in komplexen Systemen. Die Methode wurde von Nancy Leveson am Massachusetts Institute of Technology (MIT) entwickelt und basiert auf der Systemtheorie, die das Verständnis von Systemen als Ganzes und ihrer Wechselwirkungen betont [327].

STPA geht über die traditionelle Fehlerbaumanalyse hinaus und betrachtet nicht nur die unmittelbaren Auswirkungen von Fehlern, sondern auch die zugrundeliegenden Ursachen und Bedingungen, die zu einem Fehler führen können [327]. Es konzentriert sich auf die systemischen Aspekte der Sicherheit und geht davon aus, dass Sicherheitsprobleme das Ergebnis von Wechselwirkungen zwischen verschiedenen Systemkomponenten sind.

STPA besteht aus einer Reihe von Schritten, die zur Identifizierung von Gefahren und zur Entwicklung von Sicherheitsmaßnahmen durchgeführt werden. Im ersten Schritt wird das System modelliert und die Funktionsweise der einzelnen Komponenten untersucht und im zweiten Schritt werden mögliche Fehler, Bedingungen und Ereignisse identifiziert, die zu Sicherheitsproblemen führen können [327]. Der nächste Schritt ist die Durchführung von STPA-Analysen, bei denen untersucht wird, wie verschiedene Systemkomponenten miteinander interagieren und wie dies zu Sicherheitsproblemen führen kann. Hierbei werden die Wechselwirkungen zwischen den verschiedenen Komponenten des Systems berücksichtigt und es werden sowohl technische als auch menschliche Aspekte betrachtet [327]. Auf der Grundlage der STPA-Analyse werden dann Sicherheitsmaßnahmen entwickelt, die dazu beitragen, die identifizierten Sicherheitsprobleme zu lösen oder zu verhindern. Hierbei werden sowohl technische als auch organisatorische Maßnahmen berücksichtigt, um sicherzustellen, dass das System insgesamt sicherer wird.

A.6.5Risikomatrix und Vergleich zum Risikographen

Die Risikomatrixmethode ist ein qualitatives Verfahren zur Risikoeinschätzung, um die Auswirkungen von potenziellen Gefahren zu quantifizieren. Sie benutzt dabei die bekannten Größen der Eintrittswahrscheinlichkeit und des Schadensausmaßes, um die Beurteilung sicherheitskritischer Maschinen durchzuführen [169].

Es gibt keine einheitliche Darstellungsform für die Risikomatrix. Die Grundnorm definiert lediglich eine allgemeine Risikoklassifizierung. Durch die Wahrscheinlichkeit und dessen Folgen erkannter Risiken lassen sich einzelne Komponenten oder das System im Gesamten in Risikoklassen einteilen [9]. Die Einteilung geschieht in vier Auswirkungstypen. Katastrophal bedeutet der Verlust mehrerer Menschenleben, kritisch der Verlust eines einzigen Lebens, gering lediglich schwere Verletzungen einer oder mehrerer Personen und unbedeutend mit maximal leichten Verletzungen [9]. In Abhängigkeit mit der Wahrscheinlichkeit bilden sich die Klassen I, inakzeptabel unter allen Umständen bis IV akzeptabel, wobei dennoch eine Überwachung notwendig bleiben kann [9]. Den Zusammenhang bildet Tabelle A.2.

Folgen: Wahrscheinlichkeit:	Katastrophal	Kritisch	Gering	Unbedeutend
Häufig	I	I	I	II
Wahrscheinlich	I	I	II	III
Gelegentlich	I	II	III	III
Zukünftig	II	III	III	IV
Unwahrscheinlich	III	III	IV	IV
Unglaublich	IV	IV	IV	IV

Tabelle A.2: Risikoklassen-Matrix zur Klassifizierung von Unfällen [9]

Risikomatrix und Risikograph werden jeweils verwendet, um die Gefährdungen anhand der Schwere der Verletzungen (S), der Häufigkeit der Exposition (F) und der Möglichkeit zur Vermeidung der Gefährdung (P) einzuschätzen. Eine Gegenüberstellung findet sich in Tabelle A.3.

	ISO 14849	**EN 62061**
S	S1 leichte, meist reversible Verletzungen S2 ernste, üblicherweise irreversible Verletzungen	Irreversible Verletzung bis Tod (4 Punkte) Irreversible Verletzung bis Verlust von Körperteilen (3 Punkte) Reversible Verletzungen bis medizinische Versorgung (2 Punkte) Reversible Verletzungen (1 Punkt)
F	F1 selten bis wenig häufig und/oder kurz F2 häufig bis dauernd und/oder lang	< 1 h (5Punkte) > 1 h bis < 1 Tag (5 Punkte*) > 1 Tag bis < 2 Wochen (4 Punkte*) > 2 Wochen bis < 1 Jahr (3 Punkte*) > 1 Jahr (2 Punkte*) * minus 1 bei Dauer < 10Min
P	P1 möglich unter bestimmten Bedingungen P2 kaum möglich	Unmöglich (5 Punkte) Selten (3 Punkte) Wahrscheinlich (1 Punkt)

Tabelle A.3: Gegenüberstellung Risikoeinschätzung ISO 13849 [34] und EN 62061 [169]

Die errechnete Punktzahl der Parameter innerhalb der Risikomatrix nach EN 62061 in Abbildung A.4 wirkt sich direkt auf das SIL aus.

Schwere (S)	Klasse (K)				
	4	5 bis 7	8 bis 10	11 bis 13	14 bis 15
4	SIL 2	SIL 2	SIL 2	SIL 3	SIL 3
3		(AM)	SIL 1	SIL 2	SIL 3
2			(AM)	SIL 1	SIL 2
1				(AM)	SIL 1

Abbildung A.4: Risikomatrix nach EN 62061 [169]

Zusätzlich hierzu existiert die Risikomatrix nach Nohl et al. [328] in Abbildung A.5, die einen Risikoindex von eins bis sieben angibt, wobei lediglich bei einem Index kleiner drei keine weitere Risikominderung notwendig wird.

Risikomatrix (nach Nohl)		Mögliche Schadensschwere S			
		Leichte Verletzungen oder Erkrankungen	Mittelschwere Verletzungen oder Erkrankungen	Schwere Verletzungen oder Erkrankungen	Möglicher Tod, Katastrophe
Wahrscheinlichkeit W	sehr gering	1	2	3	4
	gering	2	3	4	5
	mittel	3	4	5	6
	hoch	4	5	6	7

Abbildung A.5:Risikomatrix nach Nohl et al. [328, 329]

Ein weiteres Verfahren zur Bestimmung eines Risikoindexes findet sich in ISO 14121-2 [330], die einen praktischen Leitfaden und Verfahrensbeispiele für die Sicherheit von Maschinen angibt [330]. Hierbei findet erneut eine Unterteilung in drei Wahrscheinlichkeitsgrade, ähnlich des Risikographs der Grundnorm, aber eine Unterteilung in zwei Schweregrade, ähnlich des Risikographs aus ISO 13849 statt. Das Ergebnis bildet einen Risikoindex zwischen eins und sechs, wobei ähnlich Nohl et al. lediglich Risiken mit einem Risikoindex von eins und zwei vernachlässigbar sind. Der Guide setzt, wie in Abbildung A.6 dargestellt, den Risikographen und die Risikomatrix gleich, sodass auf Anwendungsebene kein realer Unterschied mehr besteht.

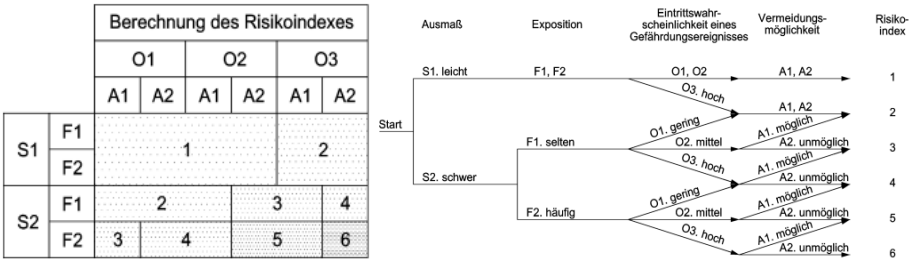


Abbildung A.6: Risikomatrix (links) und Risikograph (rechts) nach ISO 14121-2 [330]

Die Methode Risikomatrix wurde im Rahmen dieser Arbeit nicht tiefergehend behandelt, da sie ähnlich wie RBD nicht Teil der spezifischen Analysemethoden aus ISO 13849 ist.

A.6.6SBBM

Strukturanalyse

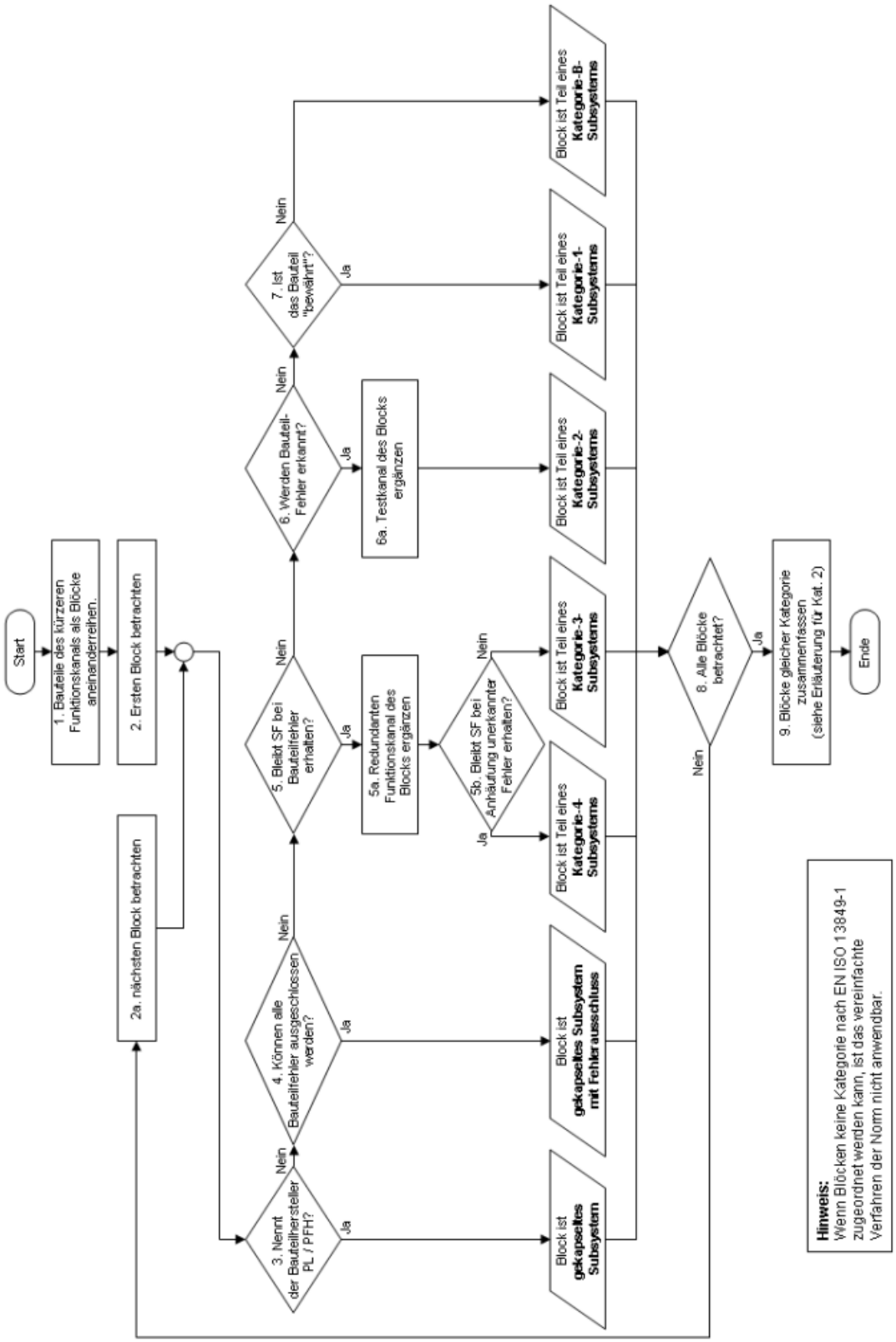


Abbildung A.7: Ablaufdiagramm einer Strukturanalyse [223]

Abschätzung des DC

Maßnahme	DC
Eingabeeinheit	
Zyklischer Testimpuls durch dynamischen Wechsel der Eingangssignale	90 %
Plausibilitätsprüfung, z.B. Verwendung der Schließer- und Öffnerkontakte von zwangsgeführten Relais	99 %
Kreuzvergleich von Eingangssignalen ohne dynamischen Test	0 % bis 99 %, abhängig davon, wie oft ein Signalwechsel durch die Anwendung erfolgt (siehe ***)
Kreuzvergleich von Eingangssignalen mit dynamischem Test, wenn Kurzschlüsse nicht bemerkt werden können (bei Mehrfach-Ein-/Ausgängen)	90 %
Kreuzvergleich von Eingangssignalen mit unmittelbarem und Zwischenergebnissen in der Logik (L) und zeitlich und logische Programmlaufüberwachung und Erkennung statischer Ausfälle und Kurzschlüsse (bei Mehrfach-Ein-/Ausgängen)	99 %
Indirekte Überwachung (z.B. Überwachung durch Druckschalter, elektrische Positionsüberwachung von Betätigungselementen)	90 % bis 99 %, abhängig von der Anwendung (siehe *)
Direkte Überwachung (z.B. elektrische Stellungsüberwachung der Steuerungsventile, Überwachung elektromechanischer Einheiten durch Zwangsführung)	99 %
Fehlererkennung durch den Prozess	0 % bis 99 %, abhängig von der Anwendung; diese Maßnahme ist allein nicht ausreichend für den PL_r e! (siehe * und **)
Überwachung einiger Merkmale des Sensors (Ansprechzeit, der Bereich analoger Signale, z.B. elektrischer Widerstand, Kapazität)	60 %
Logik	
Indirekte Überwachung (z.B. Überwachung durch Druckschalter, elektrische Positionsüberwachung von Betätigungselementen)	90 % bis 99 %, abhängig von der Anwendung (siehe *)
Direkte Überwachung (z. B. elektrische Überwachung der Steuerungsventile, Überwachung elektromechanischer Einheiten durch Zwangsführung)	99 %
Einfache zeitliche Programmlaufüberwachung (z.B. Zeitglied als Watchdog, mit Triggersignalen im Programm der Logik)	60 %
Zeitliche und logische Programmlaufüberwachung durch den Watchdog, wobei die Testeinrichtung Plausibilitätstests des Verhaltens der Logik durchführt	90 %
Selbsttest bei Anlauf, um verborgene Fehler in Teilen der Logik zu finden (z.B. Programm- und Datenspeicher, Eingangs-/Ausgangsanschlüsse, Schnittstellen)	90 %, (abhängig von der Testausführung)
Testung der Reaktionsmöglichkeit der Überwachungseinrichtung (z.B. Watchdog) durch den Hauptkanal nach Anlauf, oder wann immer die Sicherheitsfunktion angefordert wird, oder wann immer ein externes Signal dies durch eine Eingangseinrichtung anfordert	90 %
Dynamische Prinzipien (alle Bauteile der Logik erfordern eine Zustandsänderung EIN-AUS-EIN, wenn die Sicherheitsfunktion angefordert wird), z.B. Verriegelungsschaltungen in Relaistechnik	99 %
Invarianter Speicher: Signatur einfacher Wortbreite (8Bit)	90 %
Invarianter Speicher: Signatur doppelter Wortbreite (16Bit)	99 %

Maßnahme	DC
Varianter Speicher: RAM-Test durch Verwendung redundanter Daten, z. B. Flags, Merker, Konstanten, Timer und Kreuzvergleich dieser Daten	60 %
Varianter Speicher: Test der Lesbarkeit und der Beschreibbarkeit der verwendeten Speicherzellen	60 %
Varianter Speicher: RAM-Überwachung mit modifiziertem Hammingcode oder RAM-Selbsttest (z.B. „Galpat“ oder „Abraham“)	99 %
Verarbeitungseinheit: Selbsttest durch Software	60 % bis 90 %
Verarbeitungseinheit: Kodierte Verarbeitung	90 % bis 99 % (siehe *)
Fehlererkennung durch den Prozess	0 % bis 99 %, abhängig von der Anwendung; diese Maßnahme ist allein nicht ausreichend für den PL_r e! (siehe * und **)
Ausgabeeinheit	
Überwachung der Ausgänge durch einen Kanal ohne dynamischen Test	0 % bis 99 %, abhängig davon, wie oft ein Signalwechsel durch die Anwendung erfolgt (siehe ***)
Kreuzvergleich von Ausgangssignalen ohne dynamischen Test	0 % bis 99 %, abhängig davon, wie oft ein Signalwechsel durch die Anwendung erfolgt (siehe ***)
Kreuzvergleich von Ausgangssignalen mit dynamischem Test, ohne Erkennung von Kurzschlüssen (bei Mehrfach-Ein-/Ausgängen)	90 %
Kreuzvergleich von Ausgangssignalen mit unmittelbarem Ergebnis in der Logik (L) und zeitlich und logischer Softwareüberwachung des Programmablaufs und Erkennen statischer Ausfälle und Kurzschlüsse (bei Mehrfach-Ein-/Ausgängen)	99 %
Redundanter Abschaltpfad mit Überwachung der Betätigungselemente durch die Logik und Testeinrichtung	99 %
Indirekte Überwachung (z.B. Überwachung durch Druckschalter, elektrische Positionsüberwachung von Betätigungselementen)	90 % bis 99 %, abhängig von der Anwendung (siehe *)
Fehlererkennung durch den Prozess	0 % bis 99 %, abhängig von der Anwendung; diese Maßnahme ist allein nicht ausreichend für den PL_r e! (siehe * und **)
Direkte Überwachung (z. B. elektrische Überwachung der Steuerungsventile, Überwachung elektromechanischer Einheiten durch Zwangsführung)	99 %

Tabelle A.4: Abschätzung des DC nach ISO 13849 [34]

*) Für die DC-Maßnahme „Fehlererkennung durch den Prozess“ könnten die Anforderungsrate der Sicherheitsfunktion (r_d) und die Prozessdiagnoserate (Testrate) (r_t) zusammen mit einer Begrenzung des effektiven DC der geprüften Komponente berücksichtigt werden: $r_t/r_d = 1$ → DC ist auf 60 % begrenzt; $r_t/r_d = 10$ → DC ist auf 90 % begrenzt; $r_t/r_d = 100$ → DC ist auf 99 % begrenzt [171].

**) Der Effekt der Testrate (wie oft eine Signaländerung von der Anwendung durchgeführt wird) kann unter Verwendung der folgenden Einschränkungen für den effektiven DC der getesteten Komponente berücksichtigt werden:

Für Kategorie 3 und Kategorie 4:

r_t < 1/Jahr → DC beträgt 0 %;

$r_t \geq 1$/Jahr → DC ist auf 90 % begrenzt;

$r_t \geq 1$/Monat → DC ist auf 99 % begrenzt [171].

***) Abhängig des Signalwechsels ergibt sich ein prozentualer Anteil, der in Abhängigkeit von der jeweiligen Anwendung festzulegen ist [171].

Weitere Abschätzungen des DC finden sich in IEC 61508-2 [6] Tabelle A.2 bis A.15.

Verfahren zur Punktevergabe & Quantifizierung für Maßnahmen gegen CCF

Nr.	Maßnahme gegen CCF	Punktzahl
1	**Trennung/Abtrennung** Physikalische Trennung zwischen den Signalpfaden, z.B.: • Trennung der Verdrahtung/Verrohrung; • Erkennen von Kurzschlüssen und Unterbrechungen in Kabeln durch dynamische Prüfung; • getrennte Abschirmung des Signalpfads jedes Kanals; • ausreichende Luft- und Kriechstrecken auf gedruckten Schaltungen.	15
2	**Diversität** Unterschiedliche Technologien/Gestaltung oder physikalische Prinzipien werden verwendet, z. B.: • der erste Kanal elektronisch oder programmierbar elektronisch und der zweite Kanal elektromechanisch fest verdrahtet, • unterschiedliche Initiierung der Sicherheitsfunktion für jeden Kanal (z. B. Position, Druck, Temperatur), und/oder digitale und analoge Messung von Variablen (z.B. Abstand, Druck oder Temperatur) und/oder Bauteile von unterschiedlichen Herstellern	20
3	**Gestaltung/Anwendung/Erfahrung**	
3.1	Schutz gegen Überspannung, -druck, -strom, -temperatur usw.	15
3.2	Verwendung bewährter Bauteile	5
4	**Beurteilung/Analyse** Für jedes Teil von sicherheitsbezogenen Teilen eines Steuerungssystems wurde eine FMEA durchgeführt und deren Ergebnisse berücksichtigt, um Ausfälle infolge gemeinsamer Ursache bei der Gestaltung zu vermeiden.	5
5	**Kompetenz/Ausbildung** Ausbildung der Konstrukteure, um die Gründe und Auswirkungen von Ausfällen infolge gemeinsamer Ursache zu verstehen.	5
6	**Umgebung**	
6.1	Für elektrische/elektronische Systeme, Verhindern von Verunreinigungen und elektromagnetischen Störungen (EMV) zum Schutz vor Ausfällen infolge gemeinsamer Ursache entsprechend den einschlägigen Normen (z.B. IEC 61326-3-1) Fluidische Systeme: Filtrierung des Druckmediums, Verhinderung von Schmutzeintrag, Entwässerung von Druckluft, z.B. in Übereinstimmung mit den Anforderungen des Herstellers für die Reinheit des Druckmediums. Bei kombinierten Systemen sind beide Aspekte zu berücksichtigen.	25
6.2	Überwachung einiger Merkmale des Sensors (Ansprechzeit, der Bereich analoger Signale, z.B. elektrischer Widerstand, Kapazität)	10
	Gesamt Gesamtpunktzahl ≥ 65 → Anforderungen erreicht Gesamtpunktzahl < 65 → zusätzliche Maßnahmen auswählen	**[max. 100 erreichbar]**

Tabelle A.5: Maßnahmen gegen CCF nach ISO 13849 [34]

Berechnungsgrundlage für SBBM

Der $MTTF_D$ eines Kanals errechnet sich als der Kehrwert der Summe der Kehrwerte der $MTTF_D$-Werte aller Elemente, so wie in Formel A.1 angegeben. Eine zweikanalige Struktur

errechnet den Gesamt-$MTTF_D$-Wert einer Subfunktion durch die Formel in der zweiten Zeile von Formel A.1. Und der DC_{avg} über die gesamte Subfunktion findet sich in Zeile drei.

$$\text{Ein Kanal: } \frac{1}{MTTF_D} = \sum_{i=1}^{N} \frac{1}{MTTF_{Di}}$$

$$\text{Zwei Kanäle: } MTTF_D = \frac{2}{3}\left[MTTF_{DKanal1} + MTTF_{DKanal2} - \frac{1}{MTTF_{DKanal1}+MTTF_{DKanal2}}\right]$$

$$DC_{avg} = \frac{\frac{DC_1}{MTTF_{D1}} + \frac{DC_2}{MTTF_{D2}} + \cdots + \frac{DC_N}{MTTF_{DN}}}{\frac{1}{MTTF_{D1}} + \frac{1}{MTTF_{D2}} + \cdots + \frac{1}{MTTF_{DN}}}$$

Formel A.1: Berechnung $MTTF_D$ Kanäle [34]

Als generelle Anforderungen an die verschiedenen Kategorien werden zusätzliche Grenzwerte für $MTTF_D$, DC_{avg} und CCF angegeben, die die Anwendung weiter vereinfachen sollen. $MTTF_D$ und DC_{avg} haben eine Einteilung und niedrig, mittel und hoch, wohingegen CCF, soweit relevant, immer einen Wert von 65 Punkten oder mehr benötigt. Die weiteren Grenzwerte finden sich in Tabelle A.6.

Kat.	$MTTF_D$ jedes Kanals	DC_{avg}	CCF
B	niedrig bis mittel (3 Jahre ≤ $MTTF_D$ < 30 Jahre)	Kein (DC_{avg}<60 %)	Nicht relevant
1	Hoch (30 Jahre ≤ $MTTF_D$ < 100 Jahre)	Kein (DC_{avg}<60 %)	Nicht relevant
2	niedrig bis hoch (3 Jahre ≤ $MTTF_D$ < 100 Jahre)	mindestens niedrig (DC_{avg}≥60 %)	Maßnahmen im vorherigen Abschnitt
3	niedrig bis hoch (3 Jahre ≤ $MTTF_D$ < 100 Jahre)	mindestens niedrig (DC_{avg}≥60 %)	Maßnahmen im vorherigen Abschnitt
4	Hoch (30 Jahre ≤ $MTTF_D$ < 100 Jahre)	Hoch einschließlich Fehlerhäufung (DC_{avg}=99 %)	Maßnahmen im vorherigen Abschnitt

Tabelle A.6: Werte der Kategorien [34]

Schlussendlich lässt sich der PFH_D-Wert für die Sicherheitsfunktion berechnen. Ähnlich der $MTTF_D$-Berechnung für einen Kanal werden die Kehrwerte der einzelnen $MTTF_D$-Werte addiert.

A.7 OMG RAAML

A.7.1 Domainmodel

Das Kernkonzept Domainmodell besteht nach [48] aus folgenden in Abbildung A.8 illustrierten Bestandteilen. Bei der Betrachtung des Kernkonzepts fällt sofort die Grundstruktur auf. Das Grundelement stellt eine Situation dar, die mit anderen Situationen korreliert [48]. So kann die Abhängigkeit zwischen einzelnen Situationen modelliert werden, inwieweit sie relevant sind, sich gegen-

seitig beeinflussen oder kausale Beziehungen bestehen [48]. Hierbei werden Abhängigkeiten, Generalisierungen und Kompositionen geschaffen, um die Basis für die Sicherheitstechnik zu liefern [49].

Eine Situation beschreibt eine Menge von Situationsereignissen eines bestimmten Typs, wobei Parameter wie System, Ort, Zeit und Zustand durch Klassifikatoren und nicht durch individuelle Beschreibungen beschrieben werden [236]. Ein Situationsereignis ist ein System, das sich an einem bestimmten Ort, zu einer bestimmten Zeit und in einem bestimmten Zustand befindet [236]. Es ist beispielsweise möglich, dass eine Situation durch mehrere andere Situationen verursacht wird und umgekehrt kann eine Situation auch mehrere andere Situationen verursachen.

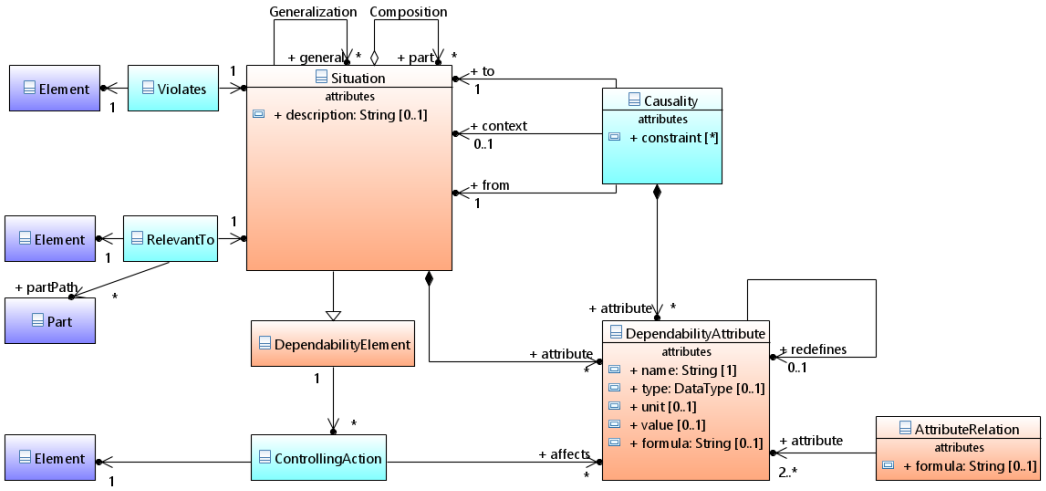


Abbildung A.8: Kernkonzept Domain Model

- Eine auftretende Situation ist definiert als ein System, das sich an einem bestimmten Ort, zu einer bestimmten Zeit und in einem bestimmten Zustand befindet.
- Eine elementare Situation kann als Klassifikator verstanden werden, der eine Menge von Situationsereignissen eines bestimmten Typs beschreibt.
- Parameter wie System, Ort, Zeit und Zustand werden durch Klassifikatoren und nicht durch Einzelbeschreibungen umgesetzt.
- Verschiedene Situationen können Generalisierungs- und Spezialisierungsbeziehungen zwischen ihnen haben.
- Situationen können quantitative Attribute haben, wie z. B. die Wahrscheinlichkeit des Auftretens, die über die Klasse `DependabilityAttribute` definiert werden können.
- Quantitative Attribute können untereinander und mit Attributen des Systems durch Formeln mithilfe der Klasse `AttributeRelation` in Beziehung gesetzt werden.

- Formeln und Berechnungen können in jeder Sprache ausgedrückt werden, die vom eingesetzten Modellierungswerkzeug verstanden werden, einschließlich OCL und anderer ausführbarer Sprachen.
- Verschiedene Situationen können mithilfe der `Causality`-Klasse miteinander verbunden werden, um semantische Beziehungen zwischen Situationen wie einfache Kausalität, bedingte Kausalität und probabilistische Verbindungen auszudrücken. Diese Beziehungen können auch quantitative Attribute haben, wie die Wahrscheinlichkeit des Auftretens der to-Situation, wenn die from-Situation eintritt.
- Eine nicht-elementare Situation (die Composition-Beziehung) ist ein Konzept, das mehrere elementare Situationen umfasst: ein einzelnes System oder eine Kombination mehrerer Systeme in einer veränderlichen Anordnung, die im Laufe der Zeit eine Folge von Zuständen durchläuft.
- Die `RelevantTo`-Beziehung wird verwendet, um Situationen mit Systemmodellelementen zu verknüpfen, um Kontext und Relevanz für die Situation zu schaffen.
- Situationen können über die `ControllingAction` entschärft, erkannt und verhindert werden. Die Verwendung dieser Beziehung führt zu neuen Sicherheitsanforderungen [49].

A.7.2Paketstruktur

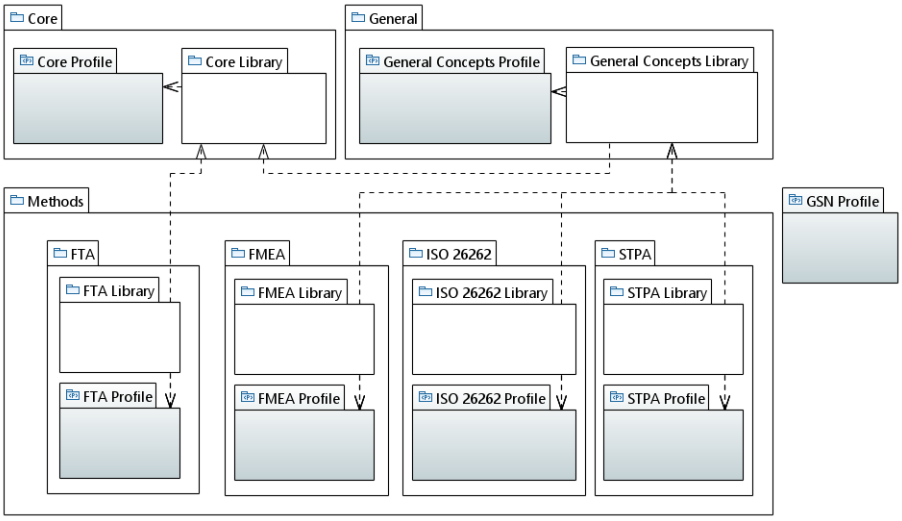


Abbildung A.9: RAAML Packet Struktur [16]

A.7.3FMEA

In der FMEA-Bibliothek findet sich die erste von RAAML beschriebene Anwendung zur Risikobeurteilung, dargestellt in Abbildung A.10. Hierbei wird im Speziellen die Berechnung des

Risikos, das bereits über General Concepts beschrieben wurde, für ein FMEA-Item unter Einbeziehung der Ausfallart, der Ursache und der Auswirkungen berechnet und in einer Risikoprioritätszahl Risk Priority Number (RPN) dargestellt. Die Berechnung findet in einem Constraint-Block statt, der die Einflussfaktoren, wie in IEC 60812 [202] vorgegeben, multipliziert.

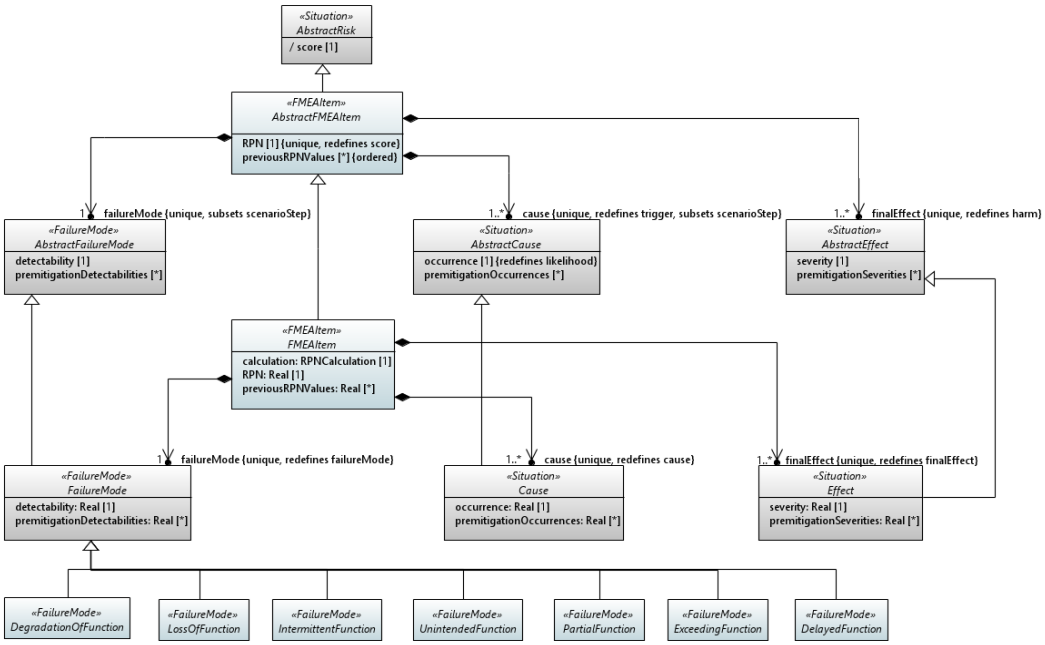


Abbildung A.10: RAAML FMEA Bibliothek [177]

A.7.4FTA

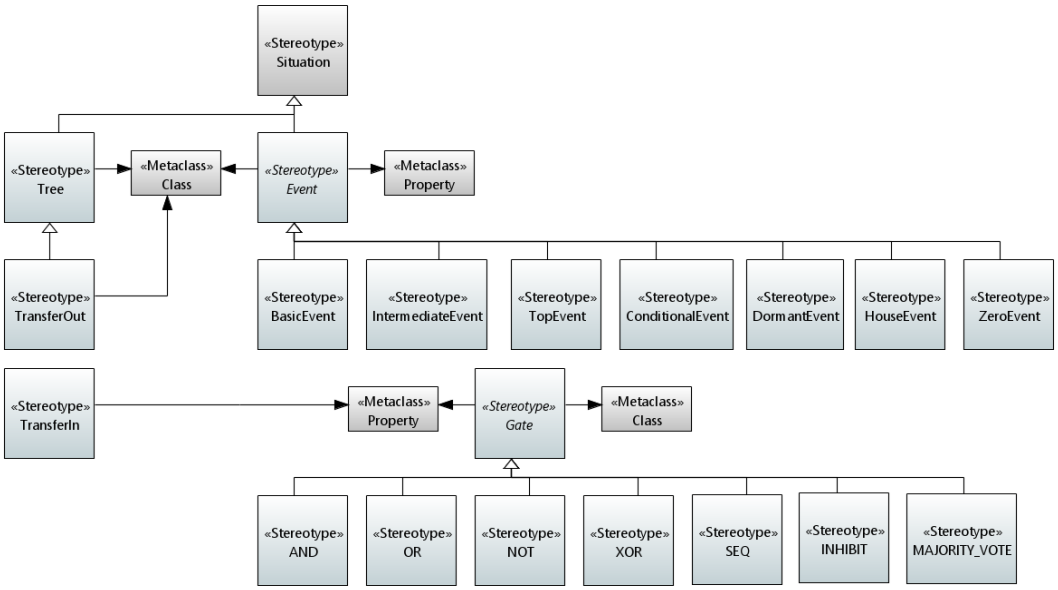


Abbildung A.11: RAAML FTA Profil [177]

TopEvent

Unerwünschtes Ereignis - Fehler oder Auswirkung - an der Spitze des Fehlerbaums.

IntermediateEvent

Ein Fehler, der aufgrund eines oder mehrerer vorangehender Ereignisse auftritt, die durch logische Gatter wirken.

BasicEvent

Ein grundlegender Anfangsfehler, der keine weitere Entwicklung erfordert.

ConditionalEvent

Spezifische Bedingungen oder Einschränkungen, die für ein beliebiges Logikgatter gelten.

DormantEvent

Ähnelt dem BasicEvent, zeigt aber den latenten Fehler an, der durch regelmäßige Tests entdeckt wird.

UndevelopedEvent

Ein Ereignis, das nicht weiter ausgearbeitet wird, weil es entweder von unzureichender Bedeutung ist oder weil keine Informationen verfügbar sind.

HouseEvent

Ein Ereignis, das so eingestellt werden kann, dass es eintritt oder nicht eintritt.

ZeroEvent

Ein Ereignis, das eine Bedingung oder ein Ereignis darstellt, das niemals eintreten wird.

AND-Gate

Das Ausgabeereignis tritt nur ein, wenn Eingabeereignisse eintreten.

Or-Gate

Das Ausgangsereignis tritt ein, wenn mindestens eines der Eingangsereignisse eintritt.

Not-Gate

Das Ausgabeereignis tritt ein, wenn das Eingabeereignis nicht eintritt.

XOR-Gate

Das Ausgabeereignis tritt ein, wenn genau eines der Eingabeereignisse eintritt.

SEQ-Gate

Das Ausgabeereignis tritt ein, wenn Eingabeereignisse in einer bestimmten Reihenfolge eintreten.

INHIBIT-Gate

Das Ausgangsereignis tritt ein, wenn das Eingangsereignis bei Vorliegen einer Freigabebedingung eintritt.

MAJORITY_VOTE-Gate

Das Ausgangsereignis tritt ein, wenn die Mehrheit der Eingangsereignisse eintritt. Es hat einen Schwellenwertparameter m.

A.8 Erweiterte externe Use-Cases der Stakeholder

A.8.1 Maschinenbauingenieur

Abbildung A.12 definiert die externen Kontexte des Maschinenbauers in Bezug zur Gesamtmodellierung.

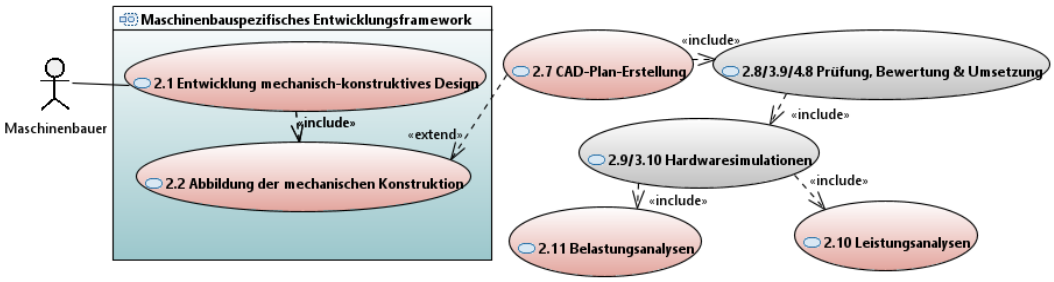


Abbildung A.12: Erweiterter Anwendungskontext - Anwendungsfälle Maschinenbauer

UC2.7 – CAD-Plan-Erstellung

Akteure: Maschinenbauer
Beschreibung: Der Maschinenbauer entwickelt die 3D-Modelle der Konstruktion auf Grundlage der SysML-Modelle
Vorbedingungen: Export der angereicherten Modelle ins CAD-Programm
Nachbedingungen: keine
Zusätzliche Bemerkung: Nicht Teil des Modell-Frameworks

UC2.8 – Prüfung, Bewertung & Umsetzung

Akteure: Maschinenbauer, Elektroingenieur, Informatiker
Beschreibung: Der Maschinenbauer überprüft und bewertet die CAD-Modelle, fertigt und bestellt die Komponenten und baut die Konstruktion auf.
Vorbedingungen: CAD-Plan vorhanden
Zusätzliche Bemerkung: Nicht Teil des Modell-Frameworks

UC2.9 – Hardwaresimulationen

Akteure: Maschinenbauer, Elektroingenieur
Beschreibung: Der Maschinenbauer simuliert die mechanische Funktionsweise.
Vorbedingungen: CAD-Plan vorhanden
Zusätzliche Bemerkung: Nicht Teil des Modell-Frameworks

UC2.10 – Leistungsanalysen

Akteure: Maschinenbauer
Beschreibung: Der Maschinenbauer führt Leistungsanalysen durch, um sicherzustellen, dass die Konstruktion den Leistungskriterien standhält.
Vorbedingungen: CAD-Plan vorhanden
Zusätzliche Bemerkung: Nicht Teil des Modell-Frameworks

UC2.11 – Belastungsanalysen

Akteure: Maschinenbauer
Beschreibung: Der Maschinenbauer führt Belastungsanalysen durch, um sicherzustellen, dass die Konstruktion den mechanischen Belastungen standhält.
Vorbedingungen: CAD-Plan vorhanden
Zusätzliche Bemerkung: Nicht Teil des Modell-Frameworks

A.8.2Elektroingenieur

Abbildung A.13 definiert die externen Kontexte des Maschinenbauers in Bezug zur Gesamtmodellierung.

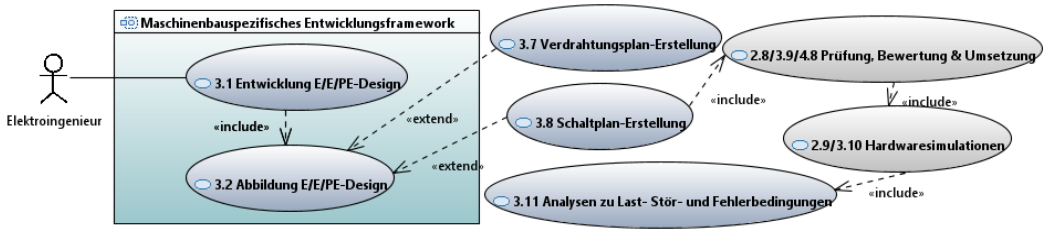


Abbildung A.13: Erweiterter Anwendungskontext - Anwendungsfälle Elektroingenieur

UC3.7 – Schaltplan-Erstellung

Akteure: Elektroingenieur
Beschreibung: Der Elektroingenieur entwickelt den Schaltplan der Maschine auf Grundlage der SysML-Modelle.
Vorbedingungen: Export der angereicherten Modelle ins computergestützte Schaltplanerstellungswerkzeug
Zusätzliche Bemerkung: Nicht Teil des Modell-Frameworks

UC3.8 – Verdrahtungsplan-Erstellung

Akteure: Elektroingenieur
Beschreibung: Der Elektroingenieur entwickelt den Verdrahtungsplan der Maschine auf Grundlage der SysML-Modelle.
Vorbedingungen: Export der angereicherten Modelle ins computergestützte Schaltplanerstellungswerkzeug
Zusätzliche Bemerkung: Nicht Teil des Modell-Frameworks

UC3.9 – Prüfung, Bewertung & Umsetzung

Akteure: Maschinenbauer, Elektroingenieur, Informatiker
Beschreibung: Der Elektroingenieur überprüft und bewertet die Schaltpläne, bestellt die Komponenten und baut die elektrotechnischen Bestandteile der Maschine auf.
Vorbedingungen: Schaltplan und Verdrahtungsplan vorhanden.
Zusätzliche Bemerkung: Nicht Teil des Modell-Frameworks

UC3.10 – Hardwaresimulationen

Akteure: Maschinenbauer, Elektroingenieur
Beschreibung: Der Elektroingenieur simuliert die elektrotechnische Funktionsweise.

Vorbedingungen: Schaltplan und Verdrahtungsplan vorhanden.
Zusätzliche Bemerkung: Nicht Teil des Modell-Frameworks

UC3.11 – Analysen zu Last-, Stör- und Fehlerbedingungen

Akteure: Elektroingenieur
Beschreibung: Der Elektroingenieur führt Analysen zu Last-, Stör- und Fehlerbedingungen durch, um sicherzustellen, dass das E/E/PE-Design den Leistungskriterien der Maschine standhält.
Vorbedingungen: Schaltplan und Verdrahtungsplan vorhanden.
Zusätzliche Bemerkung: Nicht Teil des Modell-Frameworks

A.8.3Informatiker

Abbildung A.14 definiert die externen Kontexte des Maschinenbauers in Bezug zur Gesamtmodellierung.

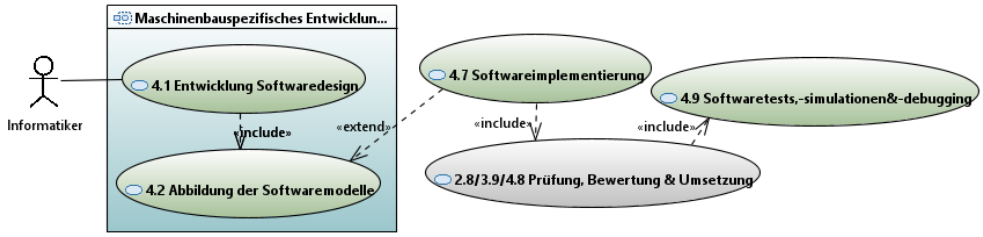


Abbildung A.14: Erweiterter Anwendungskontext - Anwendungsfälle Informatiker

UC4.7 – Softwareimplementierung

Akteure: Informatiker
Beschreibung: Der Informatiker entwickelt die Software der Maschine über die automatische Umwandlung der SysML-Modelle und Anreicherung mit fehlenden Informationen.
Vorbedingungen: Export der angereicherten Modelle ins Softwareentwicklungswerkzeug.
Zusätzliche Bemerkung: Nicht Teil des Modell-Frameworks

UC4.8 – Prüfung, Bewertung & Umsetzung

Akteure: Maschinenbauer, Elektroingenieur, Informatiker
Beschreibung: Der Informatiker überprüft und bewertet die Software, koordiniert sich mit dem Maschinenbauer und Elektroingenieur und kompiliert den Code auf die Steuerung, um die Funktionalität der Maschine zu testen.
Vorbedingungen: Hardwareaufbau und -fehlerüberprüfung durchgeführt
Zusätzliche Bemerkung: Nicht Teil des Modell-Frameworks

UC4.9 – Softwaretests, -simulationen und -debugging

Akteure: Informatiker
Beschreibung: Der Informatiker sucht und behebt zufällige Fehler in der Software und führt Simulationen und Tests durch, um Auswirkungen systematischer Fehler auf die mechanische Konstruktion und das elektrotechnische Design zu vermeiden.
Vorbedingungen: Software-Code vorhanden.

A.9 MMBSEFE-Funktionsbeschreibungen

Im Folgenden werden für jedes modellierte UML-Element und jeden Stereotyp prägnante Funktionsbeschreibungen angeführt. Dabei wird angegeben, ob es sich um ein abstraktes Element handelt, eventuelle Generalisierungen und Erweiterungen beschrieben und gegebenenfalls OCL-Constraints formuliert.

A.9.1 MachineConstructionConcept

Um die Abhängigkeiten der einzelnen Stakeholder im `MachineConstructionConcept` genauer zu beschreiben, wurden die in Abbildung A.15 dargestellten Profile und Bibliotheken zur Konkretisierung des Konzepts hinzugefügt. Hieraus ergeben sich zusätzliche oder veränderte Abhängigkeiten wie folgt:

- Blau markierte Abhängigkeiten: Profile hängen vom MachineAssemblyStructure-Profil ab, da dieses die Stereotypen für die Grundelemente definiert, die für die Entwicklung von Maschinen benötigt werden.
- Grüne markierte Abhängigkeiten: Bibliotheken hängen von der MachineAssemblyStructure-Bibliothek ab, da diese die Grundstruktur für die Entwicklung von Maschinen definiert.
- Gelb markierte Abhängigkeiten: Die MachineAssemblyStructure-Bibliothek hängt von allen Stakeholder-spezifischen Sichten ab, da diese Bibliothek die verschiedenen Sichten zusammenführt.
- Rot markierte Abhängigkeiten: Die Bibliotheken der elektrotechnischen, hydraulischen und pneumatischen Sicht hängen von der mechanisch-konstruktiven Sicht ab, da Elemente bei der Entwicklung von Maschinen auf physischen Bauteilen beruhen.
- Orange markierte Abhängigkeiten: Die Bibliotheken der hydraulischen, pneumatischen und informationstechnischen Sicht hängen von der elektrotechnischen Sicht ab, da hydraulischen und pneumatischen Elemente elektronisch gesteuert oder ausgelesen werden können und Software immer Teil einer Steuerungselektronik sein muss.

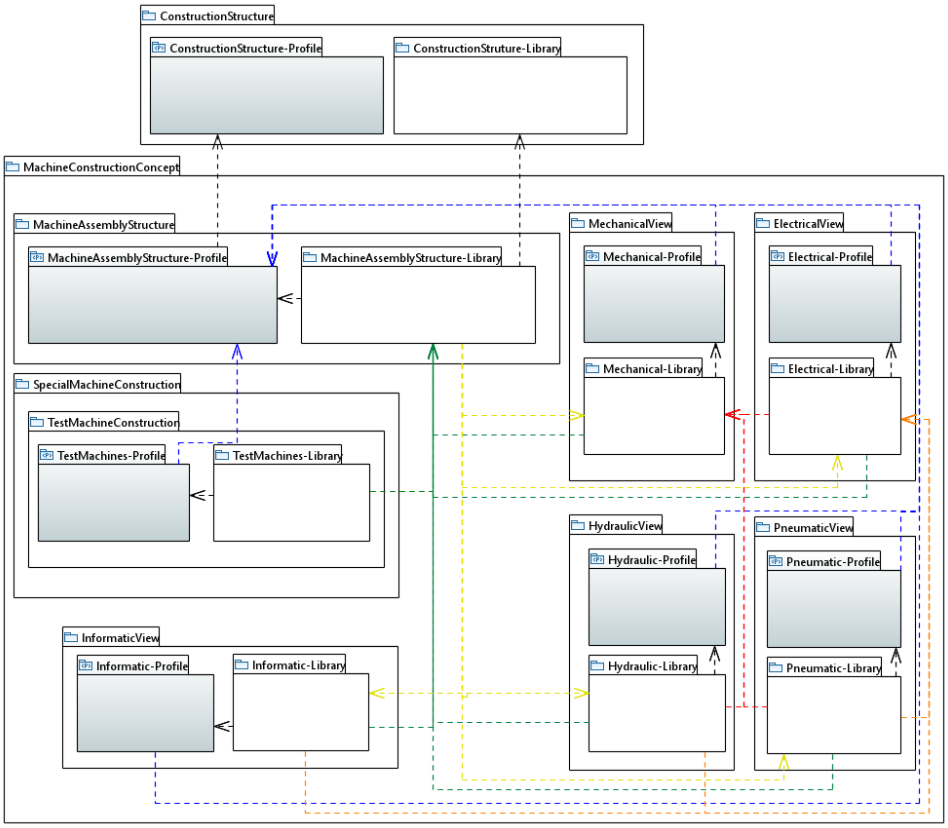


Abbildung A.15: Detaillierte MMBSEFE-Struktur für Entwicklungsingenieure

Wie sich diese Abhängigkeiten auswirken, wird im Folgenden für Elemente beschrieben.

A.9.2ConstructionStructure

ConstructionStructure-Bibliothek

Construction

Beschreibung: Eine `Construction` repräsentiert das gesamte System und dient als Basisklasse für Konstruktionen wie Maschinen, Gebäude oder Geräte. Jede Konstruktion besteht aus mindestens einem `AbstractPart`, der direkt durch eine Kompositionsbeziehung mit der `Construction` verbunden ist. Dies bedeutet, dass jede Konstruktion immer ein direktes Kind der `Construction` sein muss. Dabei muss jede Konstruktion aus mindestens einem `AbstractPart` bestehen.
Abstrakt: Ja
Applied Stereotype: Construction

AbstractPart

Beschreibung: Ein `AbstractPart` stellt ein Teilsystem einer Konstruktion dar. Teilsysteme einer Konstruktion erben von `AbstractPart`. Jeder untergeordnete Teil einer Konstruktion ist direkt mit dem Teil über eine Komposition verbunden. Dabei muss jeder Teil aus mindestens einem `AbstractSubPart` bestehen. Dies dient der späteren Stakeholder-spezifischen Sichten.
Abstrakt: Ja
Applied Stereotype: Part

AbstractSubPart

Beschreibung: Ein `AbstractSubPart` repräsentiert einen Teil eines Teilsystems. Jeder `AbstractSubPart` erbt von `AbstractPart`. Weitere untergeordnete Teile und Elemente sind direkt über eine Komposition mit dem Teil verbunden. Es ist nicht zwingend erforderlich, dass ein Teil eines Teils aus weiteren Teilen besteht. Allerdings muss mindestens ein untergeordnetes Teil oder Element definiert werden.
Abstrakt: Ja
Generalisierung: AbstractPart
Applied Stereotype: SubPart

AbstractElement

Beschreibung: Ein `AbstractElement` ist ein Bestandteil eines `AbstractSubPart` und repräsentiert die unterste Ebene einer Konstruktion. Elemente einer Konstruktion erben von `AbstractElement`. Eigenschaften, Parameter und Ports werden durch Klassifikatoren beschrieben, anstatt durch individuelle Beschreibungen festgelegt zu werden.
Abstrakt: Ja
Applied Stereotype: Element

ConstructionStructure-Profil

Construction

Beschreibung: Eine `Construction` ist ein SysML v1.6 Block. Sie nutzt die folgende Funktionalität des Blockkonzepts: Generalisierung, Teile, Wert-Eigenschaften und Parametrik. Der `Construction`-Stereotyp wird benötigt, um Konstruktionen von anderen Arten von Blöcken zu unterscheiden. Die Erweiterung von `Property` ermöglicht es, den Stereotyp ebenso in IBD zu nutzen.
Abstrakt: Nein
Generalisierung: Block
Erweiterung: Class, Property
Constraints: [1]

```
if not self.base_Class->isEmpty() then
      --Construction stereotype can only be applied
      on any class specialized from AbstractConstruc-
      tion from ConstructionStructure-Library
      self.base_Class->asSet()->closure(gen-
eral).name-
      >includes('AbstractConstruction')
else
      --Construction stereotype can only be applied
      on any property whose type is specialized from
      AbstractConstruction from ConstructionStruc-
      ture-Library
      self.base_Property.type->asSet()->closure(gen-
eral).name-
      >includes('AbstractConstruction')
endif
```

Part

Beschreibung: Ein `Part` ist ein SysML v1.6 Block. Er nutzt die folgende Funktionalität des Blockkonzepts: Generalisierung, Teile, Wert-Eigenschaften und Parametrik. Der `Part`-Stereotyp wird benötigt, um Konstruktionsteile von anderen Arten von Blöcken zu unterscheiden. Die Erweiterung von `Property` ermöglicht es, den Stereotyp ebenso in IBD zu nutzen.
Abstrakt: Nein
Generalisierung: Block
Erweiterung: Class, Property
Constraints: [1]

```
if not self.base_Class->isEmpty() then
      --Part stereotype can only be applied on any
      class specialized from AbstractPart from Con-
      structionStructure-Library
      self.base_Class->asSet()-
      >closure(general).name->includes('Abstract-
Part')
else
      --Part stereotype can only be applied on any
      property whose type is specialized from Ab-
      stractPart from ConstructionStructure-Library
      self.base_Property.type->asSet()-
      >closure(general).name->includes('Abstract-
Part')
endif
```

SubPart

Beschreibung: Ein Teil eines Teils `SubPart` ist ein SysML v1.6 Block. Er nutzt die folgende Funktionalität des Blockkonzepts: Generalisierung, Ports, Teile, Wert-Eigenschaften und Parametrik. Der `SubPart`-Stereotyp wird benötigt, um rekursive Teilkonstruktionen und später Stakeholder-spezifische Sichten von anderen Arten von Blöcken zu unterscheiden. Die Erweiterung von `Property` ermöglicht es, den Stereotyp ebenso in IBD zu nutzen.
Abstrakt: Nein
Generalisierung: Block
Erweiterung: Class, Property
Erweiterung: Class, Property
Constraints: [1]

```
if not self.base_Class->isEmpty() then
      --SubPart stereotype can only be applied on any
      class specialized from AbstractSubPart from
      ConstructionStructure-Library
      self.base_Class->asSet()-
      >closure(general).name-
      >includes('AbstractSubPart')
else
      --SubPart stereotype can only be applied on any
      property whose type is specialized from Ab-
      stractSubPart from Construction-Structure-Li-
      brary
      self.base_Property.type->asSet()-
      >closure(general).name-
      >includes('AbstractSubPart')
endif
```

Element

Beschreibung: Ein `Element` ist ein SysML v1.6 Block. Es nutzt die folgende Funktionalität des Blockkonzepts wieder: Generalisierung, Ports, Teile, Wert-Eigenschaften und Parametrik. Der Construction-Stereotyp wird benötigt, um Konstruktionen von anderen Arten von Blöcken zu unterscheiden. Die Erweiterung von `Property` ermöglicht es, den Stereotyp ebenso in IBD zu nutzen.
Abstrakt: Nein
Generalisierung: Block
Erweiterung: Class, Property
Constraints: [1]

```
if not self.base_Class->isEmpty() then
      --Element stereotype can only be applied on any
      class specialized from AbstractElement from
      ConstructionStructure-Library
      self.base_Class->asSet()-
      >closure(general).name-
      >includes('AbstractElement')
else
      --Element stereotype can only be applied on any
      property whose type is specialized from Ab-
      stractElement from Construction-Structure-Li-
      brary
      self.base_Property.type->asSet()-
      >closure(general).name-
      >includes('AbstractElement')
endif
```

A.9.3MachineAssemblyStructure

MachineAssemblyStructure-Bibliothek

Machine

Beschreibung: Eine `Machine` repräsentiert das gesamte System und dient als Erweiterungsklasse für Maschinen. Jede Maschine besteht aus mindestens einem `MachinePart`, der direkt durch eine Kompositionsbeziehung mit der Maschine verbunden ist. Dies bedeutet, dass jede Maschine immer ein direktes Kind der `Construction` darstellt. Dabei muss jede Maschine aus mindestens einem `MachinePart` bestehen.
Abstrakt: Nein
Generalisierung: Construction
Applied Stereotype: Machine

MachinePart

Beschreibung: Ein `MachinePart` stellt einen zu betrachtenden Teil einer Maschine dar und spezialisiert dabei `AbstractPart`. Maschinenteile einer Maschine erben von `MachinePart`. Jedes Maschinenteil besteht aus einem `MechanicalAssembly` und kann zusätzlich ein `Electr(on)icalDesign`, eine `HydraulicConstruction`, eine `PneumaticConstruction` und eine `Software` beinhalten. Die entsprechenden Kompositionen der Sichtweisen in `MachinePart` bilden jeweils einen Subset des

`abstractsubpart`-Attributs. Um die Konformität mit MMBSEF zu gewährleisten, wird ein Constraint vergeben.
Abstrakt: Nein
Generalisierung: AbstractPart
Applied Stereotype: MachinePart
Constraints: [1]

```
--Every MachinePart should have a mechanicalassembly
self.attributes-
>select(a | a.name.
startsWith('mechanicalassembly'))->notEmpty()
```

MachineSubPart

Beschreibung: Ein `MachineSubPart` spezialisiert `AbstractSubPart` und repräsentiert die übergeordnete Klasse der Stakeholder-spezifischen Sichten. Es legt somit die Grundlage für `MechanicalAssembly`, `Electr(on)icalDesign`, `HydraulicConstruction`, `PneumaticConstruction` und `Software`. Weitere untergeordnete Maschinenteile und Maschinenelemente sind direkt über eine Komposition verbunden und definieren die Attribute `abstractsubpartchild` und `abstractelement` neu. Es ist nicht zwingend erforderlich, dass ein Maschinenteil aus weiteren Teilsystemen besteht. Allerdings muss mindestens ein untergeordnetes Teil oder Element zugeordnet werden.
Abstrakt: Nein
Generalisierung: AbstractSubPart
Applied Stereotype: MachineSubPart

MachineElement

Beschreibung: Ein `MachineElement` ist ein Bestandteil eines `MachineSubPart` und repräsentiert die unterste Ebene einer Maschine. Elemente einer Maschine erben von `MachineElement`, wohingegen `MachineElement` selbst ein Kind von `AbstractElement` darstellt. Es dient als Grundelement aller in den Stakeholder-spezifischen Sichten definierten Elementen und bietet die Grundlage für spezifische Eigenschaften, Parameter und Ports.
Abstrakt: Nein
Generalisierung: AbstractElement
Applied Stereotype: MachineElement

MechanicalAssembly

Beschreibung: Ein `MechanicalAssembly` spezialisiert `MachineSubPart` und repräsentiert die Stakeholder-spezifische Sicht des Maschinenbauers für den mechanisch-konstruktiven Aufbau. `MechanicalAssembly` ist Teil der Mechanical-Bibliothek und wird in dieser näher beschriebt.
Abstrakt: Nein
Generalisierung: MachineSubPart
Applied Stereotype: MechanicalAssembly

Electr(on)icalDesign

Beschreibung: Ein `Electr(on)icalDesign` spezialisiert `MachineSubPart` und repräsentiert die Stakeholder-spezifische Sicht des Elektroingenieurs für den Aufbau des E/E/PE-Designs. `Electr(on)icalDesign` ist Teil der Electrical-Bibliothek und wird in dieser näher beschriebt.
Abstrakt: Nein
Generalisierung: MachineSubPart
Applied Stereotype: Electr(on)icalDesign

HydraulicConstruction

Beschreibung: Eine `HydraulicConstruction` spezialisiert `MachineSubPart` und repräsentiert die Stakeholder-spezifische Sicht des Maschinenbauers für den Aufbau hydraulischer Auslegungen und Schaltungen. `HydraulicConstruction` ist Teil der Hydraulic-Bibliothek und wird in dieser näher beschriebt.
Abstrakt: Nein
Generalisierung: MachineSubPart
Applied Stereotype: HydraulicConstruction

PneumaticConstruction

Beschreibung: Eine `PneumaticConstruction` spezialisiert `MachineSubPart` und repräsentiert die Stakeholder-spezifische Sicht des Maschinenbauers für den Aufbau pneumatischer Auslegungen und Schaltungen. `PneumaticConstruction` ist Teil der Pneumatic-Bibliothek und wird in dieser näher beschriebt.
Abstrakt: Nein
Generalisierung: MachineSubPart
Applied Stereotype: PneumaticConstruction

Software

Beschreibung: Eine `Software` spezialisiert `MachineSubPart` und repräsentiert die Stakeholder-spezifische Sicht des Informatikers für den Aufbau pneumatischer Auslegungen und Schaltungen. `Software` ist Teil der Informatic-Bibliothek und wird in dieser näher beschriebt.
Abstrakt: Nein
Generalisierung: MachineSubPart
Applied Stereotype: InformaticPart, Component

Environment

Beschreibung: Unter `Environment` fallen Teile, die zur Abbildung der Umgebung und zur Bestimmung der räumlichen Grenzen einer Maschine beitragen.
Abstrakt: Nein
Generalisierung: MachinePart
Applied Stereotype: MachinePart

PartToProducedOrTested

Beschreibung: `PartToProducedOrTested` bezieht sich auf Teile einer Maschine, die zur Herstellung oder zum Testen eines Produkts der Maschine zugeführt und wieder ausgegeben werden.
Abstrakt: Nein
Generalisierung: MachinePart
Applied Stereotype: MachinePart

FunctionalPart

Beschreibung: `FunctionalPart` bezieht sich auf Teile einer Maschine, die zur Funktion der Maschine beitragen.
Abstrakt: Nein
Generalisierung: MachinePart
Applied Stereotype: MachinePart

MovingUnit

Beschreibung: `MovingUnit` bezieht sich auf Teile einer Maschine, die Bewegungen ausführen. Hierunter fallen Achsen, Walzen oder Räder.
Abstrakt: Nein
Generalisierung: FunctionalPart
Applied Stereotype: MachinePart

ControllerUnit

Beschreibung: `ControllerUnit` bezieht sich auf Verarbeitungseinheiten einer Maschine.
Abstrakt: Nein
Generalisierung: FunctionalPart
Applied Stereotype: MachinePart

DisplaySignalUnit

Beschreibung: `DisplaySignalUnit` umfasst Teile einer Maschine, die dazu dienen, Informationen anzuzeigen oder Signale auszugeben und somit eine Interaktion zwischen Benutzer und Maschine zu ermöglichen.
Abstrakt: Nein
Generalisierung: FunctionalPart
Applied Stereotype: MachinePart

InputUnit

Beschreibung: `InputUnit` bezeichnet Teile einer Maschine, die für die Eingabe von Informationen oder Signalen zur Interaktion zwischen Benutzer und Maschine dienen.
Abstrakt: Nein
Generalisierung: FunctionalPart
Applied Stereotype: MachinePart

MeasuringUnit

Beschreibung: `MeasuringUnit` bezieht sich auf Teile einer Maschine, die physikalische Größen wie beispielsweise Länge, Zeit, Masse, Temperatur oder Spannung über Messinstrumente messen.
Abstrakt: Nein
Generalisierung: FunctionalPart
Applied Stereotype: MachinePart

FillingUnit

Beschreibung: `FillingUnit` bezieht sich auf Teile einer Maschine, die die Maschine mit einem gasförmigen oder flüssigen Medium füllen. Hierunter fallen beispielsweise pneumatische Systeme zur Herstellung von Unter- oder Überdruck.
Abstrakt: Nein
Generalisierung: FunctionalPart
Applied Stereotype: MachinePart

FeedingRemovingUnit

Beschreibung: `FeedingRemovingUnit` bezeichnet Teile einer Maschine, die dafür zuständig sind, produzierte oder zu testende Teile in die Maschine hineinzuführen oder aus ihr herauszuführen.
Abstrakt: Nein
Generalisierung: FunctionalPart
Applied Stereotype: MachinePart

CuttingShearingPart

Beschreibung: `CuttingShearingPart` bezieht sich auf schneidende oder scherende Teile einer Maschine, wie Messer, Scheren, Sägen oder Schneidräder.
Abstrakt: Nein
Generalisierung: FunctionalPart
Applied Stereotype: MachinePart

MediumProducingPart

Beschreibung: `MediumProducingPart` bezieht sich auf produktproduzierenden Teile einer Maschine, wie beispielsweise Mischbehälter in chemischen Prozessen.
Abstrakt: Nein
Generalisierung: FunctionalPart
Applied Stereotype: MachinePart

PowerSource

Beschreibung: `PowerSource` bezieht sich auf Energiequellen einer Maschine, unabhängig der Energieform. Hierunter fallen also beispielsweise elektrische, pneumatische und hydraulische, aber auch thermische Quellen.
Abstrakt: Nein
Generalisierung: FunctionalPart
Applied Stereotype: MachinePart

TemperingPart

Beschreibung: `TemperingPart` bezeichnet Teile einer Maschine, die zur Temperierung eingesetzt werden. Hierunter fallen beispielsweise hydraulische Kühlsysteme.
Abstrakt: Nein
Generalisierung: FunctionalPart
Applied Stereotype: MachinePart

VibratingPart

Beschreibung: `VibratingPart` bezeichnet vibrierenden Teile einer Maschine. Hierunter fallen beispielsweise Schüttelgeräte.
Abstrakt: Nein
Generalisierung: FunctionalPart
Applied Stereotype: MachinePart

RadiationUnit

Beschreibung: `RadiationUnit` bezeichnet strahlenden Teile einer Maschine, unabhängig der Strahlungsart. Es kann sich beispielsweise um thermische Strahlung, aber auch um optische oder ionisierende Strahlung, handeln.
Abstrakt: Nein
Generalisierung: FunctionalPart
Applied Stereotype: MachinePart

FrameOrCaseStructure

Beschreibung: `FrameOrCaseStructure` bezeichnet mit dem konstruktiven Aufbau einer Maschine zusammenhängenden Teile.
Abstrakt: Nein
Generalisierung: FunctionalPart
Applied Stereotype: MachinePart

SafetyPart

Beschreibung: `SafetyPart` bezieht sich auf Teile einer Maschine, die zur Sicherheit der Maschine nach Vorgaben der Maschinensicherheit beitragen.
Abstrakt: Nein
Generalisierung: MachinePart
Applied Stereotype: MachinePart

ProtectionUnit

Beschreibung: `ProtectionUnit` bezeichnet Teile einer Maschine, die sich auf Schutzeinrichtungen und Sicherheitsfunktionen beziehen.
Abstrakt: Nein
Generalisierung: SafetyPart
Applied Stereotype: MachinePart

SafetyControllerUnit

Beschreibung: `SafetyControllerUnit` bezeichnet sicherheitsrelevanten Verarbeitungseinheiten einer Maschine, wie PLCs oder andere Controller.
Abstrakt: Nein
Generalisierung: SafetyPart
Applied Stereotype: MachinePart

MachineAssemblyStructure-Profil

Gibt die Stereotypen des allgemeinen Konzepts eines konstruktiven Aufbaus in Abbildung A.16 wieder.

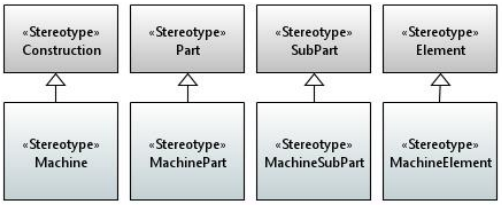


Abbildung A.16: MachineAssemblyStructure-Profil

Machine

Beschreibung: Eine `Machine` ist eine Spezialisierung des Stereotyps `Construction`. Der `Machine`-Stereotyp wird benötigt, um Maschinen von anderen Arten von Konstruktionen zu unterscheiden.
Abstrakt: Nein
Generalisierung: Construction
Erweiterung: Class, Property
Constraints: [1]

```
if not self.base_Class->isEmpty() then
      --Machine stereotype can only be applied on any
      class specialized from Machine from Machine-
      AssemblyStructure-Library
      self.base_Class->asSet()-
      >closure(general).name->includes('Machine')
else
      --Machine stereotype can only be applied on any
      property whose type is specialized from Machine
      from MachineAssemblyStructure-Library
      self.base_Property.type->asSet()-
      >closure(general).name->includes('Machine')
endif
```

MachinePart

Beschreibung: Ein `MachinePart` ist eine Spezialisierung des Stereotyps `Part`. Der `MachinePart`-Stereotyp wird benötigt, um Maschinenteile von anderen Arten von Teilen in Konstruktionen zu unterscheiden.
Abstrakt: Nein
Generalisierung: Part
Erweiterung: Class, Property
Constraints: [1]

```
if not self.base_Class->isEmpty() then
```

```
        --MachinePart stereotype can only be applied on
        any class specialized from MachinePart from Ma-
        chineAssemblyStructure-Library
        self.base_Class->asSet()-
        >closure(general).name->includes('MachinePart')
    else
        --MachinePart stereotype can only be applied on
        any property whose type is specialized from Ma-
        chinePart from Machine-AssemblyStructure-Li-
        brary
        self.base_Property.type->asSet()-
        >closure(general).name->includes('MachinePart')
    endif
```

MachineSubPart

Beschreibung: Ein `MachineSubPart` ist eine Spezialisierung des Stereotyps `SubPart`. Der `MachineSubPart`-Stereotyp wird benötigt, um rekursive Teilkonstruktionen und Stakeholder-spezifische Sichten voneinander zu unterscheiden.
Abstrakt: Nein
Generalisierung: SubPart
Erweiterung: Class, Property
Constraints: [1]

```
if not self.base_Class->isEmpty() then
        --MachineSubPart stereotype can only be applied
        on any class specialized from MachineSubPart
        from MachineAssemblyStructure-Library
        self.base_Class->asSet()-
        >closure(general).name-
        >includes('MachineSubPart')
    else
        --MachineSubPart stereotype can only be applied
        on any property whose type is specialized from
        MachineSubPart from MachineAssemblyStructure-
        Library
        self.base_Property.type->asSet()-
        >closure(general).name-
        >includes('MachineSubPart')
    endif
```

MachineElement

Beschreibung: Ein `MachineElement` ist eine Spezialisierung des Stereotyps `Element`. Der `MachineElement`-Stereotyp wird benötigt, um Maschinenelemente von anderen Arten von Elementen in Konstruktionen zu unterscheiden.
Abstrakt: Nein
Generalisierung: Element
Erweiterung: Class, Property
Constraints: [1]

```
if not self.base_Class->isEmpty() then
        --MachineElement stereotype can only be applied
        on any class specialized from MachineElement
        from MachineAssemblyStructure-Library
        self.base_Class->asSet()-
        >closure(general).name-
        >includes('MachineElement')
    else
```

```
                --MachineElement stereotype can only be applied
                on any property whose type is specialized from
                MachineElement from MachineAssemblyStructure-
                Library
                self.base_Property.type->asSet()-
                >closure(general).name-
                >includes('MachineElement')
        endif
```

A.9.4TestmachineConstruction

Testmachines-Bibliothek

TestMachine

Beschreibung: Eine `TestMachine` repräsentiert das gesamte System und dient als Erweiterungsklasse für Prüfmaschinen.
Abstrakt: Nein
Generalisierung: Machine
Applied Stereotype: TestMachine

TestmachineEnvironment

Beschreibung: Unter `TestmachineEnvironment` fallen Teile, die zur Abbildung der Umgebung und zur Bestimmung der räumlichen Grenzen einer Prüfmaschine beitragen.
Abstrakt: Nein
Generalisierung: Environment
Applied Stereotype: TestMachinePart

DUT

Beschreibung: `DUT` bezieht sich auf den Prüfling, englisch Device unter Test, der von der Prüfmaschine geprüft werden soll.
Abstrakt: Nein
Generalisierung: PartToProducedOrTested
Applied Stereotype: TestMachinePart

FunctionalTestMachinePart

Beschreibung: `FunctionalTestMachinePart` bezieht sich auf Teile einer Prüfmaschine, die zur Funktion der Prüfmaschine beitragen.
Abstrakt: Nein
Generalisierung: FunctionalPart
Applied Stereotype: TestMachinePart

TestmachineMovingUnit

Beschreibung: `TestmachineMovingUnit` bezieht sich auf Teile einer Prüfmaschine, die Bewegungen ausführen.
Abstrakt: Nein
Generalisierung: , MovingUnit
Applied Stereotype: TestMachinePart

ElectricDrive

Beschreibung: `ElectricDrive` bezieht sich auf durch elektrische Energie bewegte Teile einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachineMovingUnit
Applied Stereotype: TestMachinePart

HydraulicDrive

Beschreibung: `HydraulicDrive` bezieht sich auf durch hydraulische Energie bewegte Teile einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachineMovingUnit
Applied Stereotype: TestMachinePart

PneumaticDrive

Beschreibung: `PneumaticDrive` bezieht sich auf durch pneumatische Energie bewegte Teile einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachineMovingUnit
Applied Stereotype: TestMachinePart

Multi-AxisPositioningSystem

Beschreibung: `Multi-AxisPositioningSystem` bezieht sich auf mehrachsigen bewegten Teile einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachineMovingUnit
Applied Stereotype: TestMachinePart

PositioningTable

Beschreibung: `PositioningTable` bezieht sich auf mehrachsigen bewegten Teile einer Prüfmaschine in Form von Positionierungstischen.
Abstrakt: Nein
Generalisierung: Multi-AxisPositioningSystem
Applied Stereotype: TestMachinePart

Portalsystem

Beschreibung: `Portalsystem` bezieht sich auf mehrachsigen bewegten Teile einer Prüfmaschine in Form von Portalsystemen.
Abstrakt: Nein
Generalisierung: Multi-AxisPositioningSystem
Applied Stereotype: TestMachinePart

RobotArm

Beschreibung: `Portalsystem` bezieht sich auf mehrachsigen bewegten Teile einer Prüfmaschine in Form von Roboterarmen.

Abstrakt: Nein
Generalisierung: Multi-AxisPositioningSystem
Applied Stereotype: TestMachinePart

TestmachineControllerUnit

Beschreibung: `TestmachineControllerUnit` bezieht sich auf Verarbeitungseinheiten einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: FunctionalTestMachinePart, ControllerUnit
Applied Stereotype: TestMachinePart

PLC

Beschreibung: `PLC` bezieht sich auf Verarbeitungseinheiten einer Prüfmaschine in Form von SPSen.
Abstrakt: Nein
Generalisierung: TestmachineControllerUnit
Applied Stereotype: TestMachinePart

IndustrialPC

Beschreibung: `IndustricalPC`, kurz IPC, bezieht sich auf Verarbeitungseinheiten einer Prüfmaschine in Form von Industrie-PCs.
Abstrakt: Nein
Generalisierung: TestmachineControllerUnit
Applied Stereotype: TestMachinePart

TestmachineDisplaySignalUnit

Beschreibung: `TestmachineDisplaySignalUnit` umfasst Teile einer Prüfmaschine, die dazu dienen, Informationen anzuzeigen oder Signale auszugeben und somit eine Interaktion zwischen Benutzer und Prüfmaschine zu ermöglichen.
Abstrakt: Nein
Generalisierung: FunctionalTestMachinePart, DisplaySignalUnit
Applied Stereotype: TestMachinePart

GraphicalDisplay

Beschreibung: `GraphicalDisplay` umfasst graphisch anzeigenden Teile einer Prüfmaschine, wie Monitore o.ä.
Abstrakt: Nein
Generalisierung: TestmachineDisplaySignalUnit
Applied Stereotype: TestMachinePart

Sirenen

Beschreibung: `Sirenen` umfasst autistisch signalisierenden Teile einer Prüfmaschine, wie Sirenen o.ä.
Abstrakt: Nein
Generalisierung: TestmachineDisplaySignalUnit
Applied Stereotype: TestMachinePart

SignalLight

Beschreibung: `SignalLight` umfasst visuell signalisierenden Teile einer Prüfmaschine, wie Signalampeln o.ä.
Abstrakt: Nein
Generalisierung: TestmachineDisplaySignalUnit
Applied Stereotype: TestMachinePart

TestMachineInputUnit

Beschreibung: `TestMachineInputUnit` bezeichnet Teile einer Prüfmaschine, die für die Eingabe von Informationen oder Signalen zur Interaktion zwischen Benutzer und Prüfmaschine dienen.
Abstrakt: Nein
Generalisierung: FunctionalTestMachinePart, InputUnit
Applied Stereotype: TestMachinePart

Button

Beschreibung: `Button` umfasst Drucktaster oder Schalter einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestMachineInputUnit
Applied Stereotype: TestMachinePart

Keyboard

Beschreibung: `Keyboard` umfasst Tastaturen einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestMachineInputUnit
Applied Stereotype: TestMachinePart

Mouse

Beschreibung: `Mouse` umfasst Computermäuse einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestMachineInputUnit
Applied Stereotype: TestMachinePart

TestmachineMeasuringUnit

Beschreibung: `TestmachineMeasuringUnit` bezieht sich auf Teile einer Prüfmaschine, die physikalische Größen messen und in elektrische Signale umwandeln.
Abstrakt: Nein
Generalisierung: FunctionalTestMachinePart, MeasuringUnit
Applied Stereotype: TestMachinePart

LoadCell

Beschreibung: `LoadCell` bezieht sich auf Kraftmesszellen einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachineMeasuringUnit
Applied Stereotype: TestMachinePart

StrainGauge

Beschreibung: `StrainGauge` bezieht sich auf Dehnungsmessstreifen einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachineMeasuringUnit
Applied Stereotype: TestMachinePart

PositionMeasuringUnit

Beschreibung: `PositionMeasuringUnit` bezieht sich auf Positionsmesseinheiten einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachineMeasuringUnit
Applied Stereotype: TestMachinePart

TemperatureMeasuringUnit

Beschreibung: `TemperatureMeasuringUnit` bezieht sich auf Temperaturmesseinheiten einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachineMeasuringUnit
Applied Stereotype: TestMachinePart

HumidityMeasuringUnit

Beschreibung: `HumidityMeasuringUnit` bezieht sich auf Feuchtigkeitsmesseinheiten einer Prüfmaschine, unabhängig der Messart und des Mediums.
Abstrakt: Nein
Generalisierung: TestmachineMeasuringUnit
Applied Stereotype: TestMachinePart

PressureGauge

Beschreibung: `PressureGauge` bezieht sich auf Druckmesseinheiten einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachineMeasuringUnit
Applied Stereotype: TestMachinePart

Voltmeter

Beschreibung: `Voltmeter` bezieht sich auf Spannungsmesseinheiten einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachineMeasuringUnit
Applied Stereotype: TestMachinePart

Currentmeter

Beschreibung: `Currentmeter` bezieht sich auf Strommesseinheiten einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachineMeasuringUnit
Applied Stereotype: TestMachinePart

Resistancemeter

Beschreibung: `Resistancemeter` bezieht sich auf Widerstandsmesseinheiten einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachineMeasuringUnit
Applied Stereotype: TestMachinePart

Multimeter

Beschreibung: `Multimeter` bezieht sich auf Messeinheiten einer Prüfmaschine, die Spannungs-, Strom- und Widerstandsmessungen kombinieren.
Abstrakt: Nein
Generalisierung: Voltmeter, Currentmeter, Resistancemeter
Applied Stereotype: TestMachinePart

Oscilloscope

Beschreibung: `Oscilloscope` bezieht sich auf Oszilloskope einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachineMeasuringUnit
Applied Stereotype: TestMachinePart

TorqueMeasuringUnit

Beschreibung: `TorqueMesuringUnit` bezieht sich auf Drehmomentmessgeräte einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachineMeasuringUnit
Applied Stereotype: TestMachinePart

FlowMeter

Beschreibung: `FlowMeter` bezieht sich auf Durchflussmessgeräte einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachineMeasuringUnit
Applied Stereotype: TestMachinePart

LiquidMeter

Beschreibung: `LiquidMeter` bezieht sich auf Flüssigkeitsmessgeräte einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachineMeasuringUnit
Applied Stereotype: TestMachinePart

OpticalMeasuringUnit

Beschreibung: `OpticalMeasuringUnit` bezieht sich auf optische Messsensoren und -einheiten einer Prüfmaschine, wie Laserdioden mit Verarbeitungselektronik.
Abstrakt: Nein
Generalisierung: TestmachineMeasuringUnit
Applied Stereotype: TestMachinePart

SpeedMeasuringUnit

Beschreibung: `SpeedMeasuringUnit` bezieht sich auf Geschwindigkeitsmessgeräte einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachineMeasuringUnit
Applied Stereotype: TestMachinePart

UltrasonicMeasuringUnit

Beschreibung: `UltrasonicMeasuringUnit` bezieht sich auf Ultraschallmessgeräte einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachineMeasuringUnit
Applied Stereotype: TestMachinePart

VibrationMeasuringUnit

Beschreibung: `VibrationMeasuringUnit` bezieht sich auf Schwingungsmessgeräte einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachineMeasuringUnit
Applied Stereotype: TestMachinePart

TestmachineFillingUnit

Beschreibung: `TestmachineFillingUnit` bezieht sich auf Teile einer Prüfmaschine, die die Maschine mit einem Medium füllen.
Abstrakt: Nein
Generalisierung: FunctionalTestMachinePart, FillingUnit
Applied Stereotype: TestMachinePart

FluidFillingUnit

Beschreibung: `FluidFillingUnit` bezieht sich auf Teile einer Prüfmaschine, die die Maschine mit einem flüssigen Medium füllen.
Abstrakt: Nein
Generalisierung: TestmachineFillingUnit
Applied Stereotype: TestMachinePart

GasFillingUnit

Beschreibung: `GasFillingUnit` bezieht sich auf Teile einer Prüfmaschine, die die Maschine mit einem gasförmigen Medium füllen.
Abstrakt: Nein
Generalisierung: TestmachineFillingUnit
Applied Stereotype: TestMachinePart

TestmachineFeedingRemovingUnit

Beschreibung: `TestmachineFeedingRemovingUnit` bezeichnet Teile einer Prüfmaschine, die dafür zuständig sind, produzierte oder zu testende Teile in die Prüfmaschine hineinzuführen oder aus ihr herauszuführen.
Abstrakt: Nein
Generalisierung: FunctionalTestMachinePart, FeedingRemovingUnit
Applied Stereotype: TestMachinePart

TestmachinePowerSource

Beschreibung: `TestmachinePowerSource` bezieht sich auf Energiequellen einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: FunctionalTestMachinePart, PowerSource
Applied Stereotype: TestMachinePart

ElectricPowerSource

Beschreibung: `ElectricPowerSource` bezieht sich auf elektrische Energiequellen einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachinePowerSource
Applied Stereotype: TestMachinePart

PneumaticPowerSource

Beschreibung: `PneumaticPowerSource` bezieht sich auf pneumatische Energiequellen einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachinePowerSource
Applied Stereotype: TestMachinePart

HydraulicPowerSource

Beschreibung: `PneumaticPowerSource` bezieht sich auf hydraulische Energiequellen einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachinePowerSource
Applied Stereotype: TestMachinePart

TestmachineTemperingPart

Beschreibung: `TestmachineTemperingPart` bezeichnet Teile einer Prüfmaschine, die zur Temperierung eingesetzt werden.
Abstrakt: Nein
Generalisierung: FunctionalTestMachinePart, TemperingPart
Applied Stereotype: TestMachinePart

HeatingUnit

Beschreibung: `HeatingUnit` bezeichnet Heizeinheiten einer Prüfmaschine.

Abstrakt: Nein
Generalisierung: TestmachineTemperingPart
Applied Stereotype: TestMachinePart

CoolingUnit

Beschreibung: `CoolingUnit` bezeichnet Kühleinheiten einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachineTemperingPart
Applied Stereotype: TestMachinePart

ThermalIsolation

Beschreibung: `ThermalIsolation` bezeichnet thermisch isolierende Teile einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachineTemperingPart
Applied Stereotype: TestMachinePart

TestmachineVibratingPart

Beschreibung: `TestmachineVibratingPart` bezeichnet vibrierende Teile einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: FunctionalTestMachinePart, VibratingPart
Applied Stereotype: TestMachinePart

VibrationExciter

Beschreibung: `VibrationExciter` bezeichnet Vibrationserreger einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachineVibratingPart
Applied Stereotype: TestMachinePart

VibrationIsolationDampingPart

Beschreibung: `VibrationIsolationDampingPart` bezeichnet Vibrationsdämpfer einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachineVibratingPart
Applied Stereotype: TestMachinePart

TestmachineRadiationUnit

Beschreibung: `TestmachineRadiationUnit` bezeichnet strahlende Teile einer Prüfmaschine, unabhängig der Strahlungsart.
Abstrakt: Nein
Generalisierung: FunctionalTestMachinePart, RadiationUnit
Applied Stereotype: TestMachinePart

LightRadiationUnit

Beschreibung: `LightRadiationUnit` bezeichnet lichtausstrahlende Teile einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachineRadiationUnit
Applied Stereotype: TestMachinePart

XRayRadiationUnit

Beschreibung: `XRayRadiationUnit` bezeichnet röntgenstrahlende Teile einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachineRadiationUnit
Applied Stereotype: TestMachinePart

GammaRadiationUnit

Beschreibung: `GammaRadiationUnit` bezeichnet gammastrahlende Teile einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachineRadiationUnit
Applied Stereotype: TestMachinePart

TestmachineFrameOrCaseStructure

Beschreibung: `TestmachineFrameOrCaseStructure` bezeichnet mit dem konstruktiven Aufbau einer Prüfmaschine zusammenhängende Teile.
Abstrakt: Nein
Generalisierung: FunctionalTestMachinePart, FrameOrCaseStructure
Applied Stereotype: TestMachinePart

Case

Beschreibung: `Case` bezeichnet Hüllen oder Gehäuse einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachineFrameOrCaseStructure
Applied Stereotype: TestMachinePart

Frame

Beschreibung: `Frame` bezeichnet mechanischen Konstruktionen zum Aufbau der Struktur einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachineFrameOrCaseStructure
Applied Stereotype: TestMachinePart

SafetyTestMachinePart

Beschreibung: `SafetyTestMachinePart` bezieht sich auf Teile einer Prüfmaschine, die zur Sicherheit der Prüfmaschine nach Vorgaben der Maschinensicherheit beitragen.
Abstrakt: Nein

Generalisierung: SafetyPart
Applied Stereotype: TestMachinePart

TestmachineProtectionUnit

Beschreibung: `TestmachineProtectionUnit` bezeichnet Teile einer Prüfmaschine, die sich auf Schutzeinrichtungen und Sicherheitsfunktionen beziehen.
Abstrakt: Nein
Generalisierung: SafetyTestMachinePart, ProtectionUnit
Applied Stereotype: TestMachinePart

EmergencyStop

Beschreibung: `EmergencyStop` bezeichnet Nothalt- oder Not-Stoppeinrichtungen einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachineProtectionUnit
Applied Stereotype: TestMachinePart

PositionSensingProtection

Beschreibung: `PositionSensingProtection` bezeichnet Positionserkennungsschutzeinrichtungen einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachineProtectionUnit
Applied Stereotype: TestMachinePart

SafetyInterlock

Beschreibung: `SafetyInterlock` bezeichnet Sicherheitsverriegelungen einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachineProtectionUnit
Applied Stereotype: TestMachinePart

LightGrid

Beschreibung: `LightGrid` bezeichnet Lichtgitter einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachineProtectionUnit
Applied Stereotype: TestMachinePart

SafetyFence

Beschreibung: `SafetyFence` bezeichnet Sicherheitszäune einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachineProtectionUnit
Applied Stereotype: TestMachinePart

Fuses

Beschreibung: `Fuses` bezeichnet elektrische Sicherungen einer Prüfmaschine, wie Leitungsschutzschalter.

Abstrakt: Nein
Generalisierung: TestmachineProtectionUnit
Applied Stereotype: TestMachinePart

OverloadProtection

Beschreibung: `Fuses` bezeichnet elektrische Überlastungsschutzschalter einer Prüfmaschine, wie Überspannungsableiter.
Abstrakt: Nein
Generalisierung: TestmachineProtectionUnit
Applied Stereotype: TestMachinePart

PowerFailureProtection

Beschreibung: `PowerFailureProtection` bezeichnet Schutzeinrichtungen zum Schutz vor Stromausfällen einer Prüfmaschine, wie Backup-Batterien zum sicheren Abschalten.
Abstrakt: Nein
Generalisierung: TestmachineProtectionUnit
Applied Stereotype: TestMachinePart

TemperatureProtection

Beschreibung: `TemperatureProtection` bezeichnet Schutzeinrichtungen zum Überhitzungs- oder Unterkühlungsschutz einer Prüfmaschine oder eines Prüflings.
Abstrakt: Nein
Generalisierung: TestmachineProtectionUnit
Applied Stereotype: TestMachinePart

PressureProtection

Beschreibung: `PressureProtection` bezeichnet Schutzeinrichtungen zum Schutz vor zu hohem oder niedrigen Druck innerhalb oder außerhalb einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: TestmachineProtectionUnit
Applied Stereotype: TestMachinePart

TorqueProtection

Beschreibung: `TorqueProtection` bezeichnet Schutzeinrichtungen zum Schutz vor zu hohem Verdrehungen oder Verwindungen einer Prüfmaschine oder eines Prüflings.
Abstrakt: Nein
Generalisierung: TestmachineProtectionUnit
Applied Stereotype: TestMachinePart

MotionProtection

Beschreibung: `MotionProtection` bezeichnet Schutzeinrichtungen zur Erkennung und zum Schutz vor zu hohen Bewegungen einer Prüfmaschine, beispielsweise zum Erdbebenschutz oder auch rein mechanische Sicherheitsfunktionen zum Aufstellen der Anlage.
Abstrakt: Nein
Generalisierung: TestmachineProtectionUnit
Applied Stereotype: TestMachinePart

TestmachineSafetyControllerUnit

Beschreibung: `TestmachineSafetyControllerUnit` bezeichnet sicherheitsrelevante Verarbeitungseinheiten einer Prüfmaschine.
Abstrakt: Nein
Generalisierung: SafetyTestMachinePart, ProtectionUnit

SafetyPLC

Beschreibung: `SafetyPLC` bezieht sich auf sicherheitsrelevante Verarbeitungseinheiten einer Prüfmaschine in Form von Sicherheits-SPSen.
Abstrakt: Nein
Generalisierung: TestmachineControllerUnit
Applied Stereotype: TestMachinePart

TestMachine-Profil

Gibt die Stereotypen des Konzepts einer Prüfmaschine in Abbildung A.17 wieder.

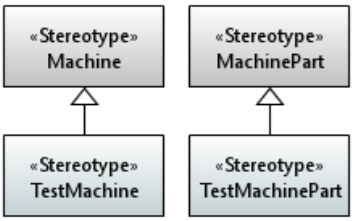


Abbildung A.17: TestmachineConstruction-Profil

TestMachine

Beschreibung: Eine `TestMachine` ist eine Spezialisierung des Stereotyps `Machine`. Der `TestMachine`-Stereotyp wird benötigt, um Prüfmaschinen von anderen Arten von Maschinen zu unterscheiden.
Abstrakt: Nein
Generalisierung: Machine
Erweiterung: Class, Property
Constraints: [1]

```
if not self.base_Class->isEmpty() then
      --TestMachine stereotype can only be applied on
      any class specialized from TestMachine from
      TestmachineConstruction-Library
      self.base_Class->asSet()-
      >closure(general).name->includes('TestMachine')
else
      --TestMachine stereotype can only be applied on
      any property whose type is specialized from
      TestMachine from Testmachine-Construction-Li-
      brary
      self.base_Property.type->asSet()-
      >closure(general).name->includes('TestMachine')
endif
```

TestMachinePart

Beschreibung: Ein `TestMachinePart` ist eine Spezialisierung des Stereotyps `MachinePart`. Der `TestMachinePart`-Stereotyp wird benötigt, um Maschinenteile für Prüfmaschinenteile im Prüfmaschinendesign zu unterscheiden.
Abstrakt: Nein
Generalisierung: MachinePart
Erweiterung: Class, Property
Constraints: [1]

```
if not self.base_Class->isEmpty() then
     --TestMachinePart stereotype can only be ap-
     plied on any class specialized from Test-
     MachinePart from TestmachineConstruction-Li-
     brary
     self.base_Class->asSet()-
     >closure(general).name-
     >includes('TestMachinePart')
else
     --TestMachinePart stereotype can only be ap-
     plied on any property whose type is specialized
     from TestMachinePart from Test-machineConstruc-
     tion-Library
     self.base_Property.type->asSet()-
     >closure(general).name-
     >includes('TestMachinePart')
endif
```

A.9.5MechanicalView

Mechanical-Bibliothek

MechanicalAssembly

Beschreibung: Ein `MechanicalAssembly` spezialisiert `SubPart` und repräsentiert die Sichtweise des Maschinenbauingenieurs für ein entsprechendes Maschinenteil. Untergeordnete mechanische Maschinenteile und Maschinenelemente sind direkt über eine Komposition verbunden und definieren die Attribute `machinesubpartchild` und `machineelement` neu. Es ist nicht zwingend erforderlich, dass ein Maschinenteil aus weiteren Teilsystemen besteht. Allerdings muss mindestens ein untergeordnetes Teil oder Element zugeordnet werden. Durch Definition der Attribute muss ein neues Constraint vergeben werden.
Abstrakt: Nein
Generalisierung: SubPart
Applied Stereotype: MechanicalAssembly
Constraints: [1]

```
--MechanicalAssembly should have a mechanicalassem-
blychild or a mechanicalelement
self.attributes-
>select(a | a.name.
startsWith('mechanicalassemblychild'))->notEmpty()
or
self.attributes-
>select(b | b.name.startsWith('mechanicalelement'))-
>notEmpty()
```

MechanicalElement

Beschreibung: Ein `MechanicalElement` ist ein Bestandteil eines `MechanicalAssembly` und repräsentiert die unterste Ebene im mechanischen Aufbau. Mechanische Elemente einer Maschine erben von `MachineElement`. Ein mechanisches Element besitzt die Attribute `weight`, `volume`, `position`, `dimension` und `orientation` aus dem ISO80000-Paket, wobei `eulerangles(radian)` ein zusammengesetzter Datentyp aus `plane angle(radian)` darstellt. Zusätzlich besitzt `MechanicalElement` eine Komposition zu `ElementProperty`, indem dem Element zusätzliche Eigenschaften zugewiesen werden können. Durch Definition der Attribute muss ein neues Constraint vergeben werden.
Abstrakt: Nein
Generalisierung: MachineElement
Applied Stereotype: MechanicalElement
Constraints: [1]

```
--MechanicalElement should have a value for weight,
volume, position, dimension and orientation
self.weight<>null and self.volume<>null and
self.position<>null and self.dimension<>null and
self.orientation<>null
```

ElementProperty

Beschreibung: Ein `ElementProperty` repräsentiert eine Eigenschaft eines mechanischen Elements.
Abstrakt: Nein
Applied Stereotype: ElementProperty

Eulerangles (radian)

Beschreibung: `Eulerangles(radian)` ist ein erweiterter Datentyp zu `planeangle(radian)`. Der Datentyp definiert die Attribute `alpha`, `beta` und `gamma` mit dem Datentyp `planeangle(radian)`.
Applied Stereotype: Datatype, ValueType

Mechanical-Profil

Gibt die Stereotypen des Konzepts für die mechanische Konstruktion in Abbildung A.18 wieder.

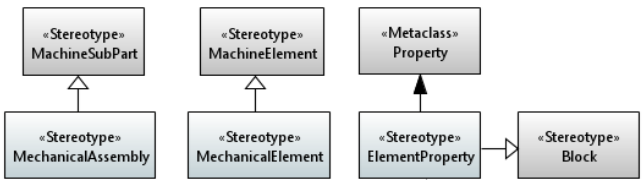


Abbildung A.18: MechanicalView-Profil

MechanicalAssembly

Beschreibung: Ein `MechanicalAssembly` ist eine Spezialisierung des Stereotyps `MachineSubPart`. Der `MechanicalAssembly`-Stereotyp wird benötigt, um rekursive Teilkonstruktionen des Maschinenbauers zu erstellen.
Abstrakt: Nein

Generalisierung: MachineSubPart
Erweiterung: Class, Property
Constraints: [1]

```
if not self.base_Class->isEmpty() then
      --MechanicalAssembly stereotype can only be ap-
      plied on any class specialized from Mechani-
      calAssembly from Mechanical-Library
      self.base_Class->asSet()-
      >closure(general).name-
      >includes('MechanicalAssembly')
else
      --MechanicalAssembly stereotype can only be ap-
      plied on any property whose type is specialized
      from MechanicalAssembly from Mechanical-Library
      self.base_Property.type->asSet()-
      >closure(general).name-
      >includes('MechanicalAssembly')
endif
```

MechanicalElement

Beschreibung: Ein `MechanicalElement` ist eine Spezialisierung des Stereotyps `MachineElement`. Der `MechanicalElement`-Stereotyp wird benötigt, um mechanische Elemente von anderen Arten von Elementen in Maschinenkonstruktionen zu unterscheiden.
Abstrakt: Nein
Generalisierung: MachineElement
Erweiterung: Class, Property
Constraints: [1]

```
if not self.base_Class->isEmpty() then
      --MechanicalElement stereotype can only be ap-
      plied on any class specialized from Mechani-
      calElement from Mechanical-Library
      self.base_Class->asSet()-
      >closure(general).name-
      >includes('MechanicalElement')
else
      --MechanicalElement stereotype can only be ap-
      plied on any property whose type is specialized
      from MechanicalElement from Mechanical-Library
      self.base_Property.type->asSet()-
      >closure(general).name-
      >includes('MechanicalElement')
endif
```

ElementProperty

Beschreibung: Ein `ElementProperty` ist ein SysML v1.6 ValueType. Der `ElementProperty`-Stereotyp wird benötigt, um physikalische Eigenschaften von mechanischen Elementen von anderen Attributen und Eigenschaften zu unterscheiden. Es entsteht die Möglichkeit ValueTypes aus ISO 80000 direkt zu erweitern und diese als Eigenschaft in den Attributen eines Elements direkt festzulegen.
Abstrakt: Nein
Generalisierung: ValueType
Erweiterung: Property

Constraints: [1]
```
--ElementProperty stereotype can only be applied on
any property whose type is specialized from Ele-
mentProperty from Mechanical-Library
self.base_Property.type->asSet()-
>closure(general).name-
>includes('ElementProperty')
```

A.9.6ElectricalView

Electrical-Bibliothek

Electr(on)icalDesign

Beschreibung: Ein `Electr(on)icalDesign` spezialisiert `SubPart` und repräsentiert die Sichtweise des Elektroingenieurs für ein entsprechendes Maschinenteil. Untergeordnete E/E/PE-Maschinenteile und Maschinenelemente sind direkt über eine Komposition verbunden und definieren die Attribute `machinesubpartchild` und `machineelement` neu. Es ist nicht zwingend erforderlich, dass ein Maschinenteil aus weiteren Teilsystemen besteht. Allerdings muss mindestens ein untergeordnetes Teil oder Element zugeordnet werden. Durch Definition der Attribute muss ein neues Constraint vergeben werden.
Abstrakt: Nein
Generalisierung: SubPart
Applied Stereotype: Electr(on)icalDesign
Constraints: [1]
```
--Electr(on)icalDesign should have an electr(on)ica-
lassemblychild or an electr(on)icalelement
self.attributes-
>select(a | a.name.
startsWith('electr(on)icalassemblychild'))-
>notEmpty() or
self.attributes-
>select(b | b.name.
startsWith('electr(on)icalelement'))->notEmpty()
```

Electr(on)icalElement

Beschreibung: Ein `Electr(on)icalElement` ist ein Bestandteil eines `Electr(on)ical-Design` und repräsentiert die unterste Ebene im E/E/PE-Aufbau. E/E/PE-Elemente einer Maschine erben von `MachineElement`. Jedes `Electr(on)icalElement` besitzt eine Assoziation zu einem `MechanicalElement`, um den Zusammenhang der Modelle zu beschreiben. Durch Definition des Attributs muss ein neues Constraint vergeben werden.
Abstrakt: Nein
Generalisierung: MachineElement
Applied Stereotype: Electr(on)icalElement
Constraints: [1]
```
--Electr(on)icalElement should have a mechanicalele-
ment associated
self.mechanicalelement<>null
```

E_Sensor_Input

Beschreibung: Ein `E_Sensor_Input` ist ein spezielles `Electr(on)icalElement` und dient der Modellierung von Sensoren oder Sensoreingängen. Er spezialisiert hierfür `Electr(on)icalElement` und definieren die Ports `powerSupply` zur Modellierung des

Spannungsflusses und `measurement` zur Modellierung von Signalen von Sensoren. Zusätzlich definiert die Assoziation mit einem `Machineelement` die korrekte Lage eines Sensors oder Eingangs, da Unterschiede zwischen Lage der Sensoreinheit und der Sensorspitze auftreten können. Durch Definition der Attribute muss ein neues Constraint vergeben werden.
Abstrakt: Nein
Generalisierung: Electr(on)icalElement
Applied Stereotype: Electr(on)icalElement
Constraints: [1]

```
--E_Sensor_Input should have a measurement and a mountedAt
self.attributes-
>select(a | a.name.startsWith('measurement'))-
>notEmpty() and
self.mountedAt<>null
```

Controller

Beschreibung: Ein `Controller` ist ein spezielles `Electr(on)icalElement` und dient der Modellierung von meist programmierbaren Verarbeitungseinheiten. Er spezialisiert hierfür `Electr(on)icalElement` und definieren die Ports `powerSupply` zur Modellierung des Spannungsflusses, `input` zur Modellierung von Eingangssignalen und `output` zur Modellierung Ausgangssignalen.
Abstrakt: Nein
Generalisierung: Electr(on)icalElement
Applied Stereotype: Electr(on)icalElement

E_Actor_Output

Beschreibung: Ein `E_Actor_Output` ist ein spezielles `Electr(on)icalElement` und dient der Modellierung von Aktoren oder Aktor-Eingängen. Er spezialisiert hierfür `Electr(on)icalElement` und definieren die Ports `powerSupply` zur Modellierung des Spannungsflusses und `control` zur Modellierung von Signalen zu Aktoren. Durch Definition der Attribute muss ein neues Constraint vergeben werden.
Abstrakt: Nein
Generalisierung: Electr(on)icalElement
Applied Stereotype: Electr(on)icalElement
Constraints: [1]

```
--E_Actor_Output should have a control
self.attributes-
>select(a | a.name.startsWith('control'))-
>notEmpty()
```

Electrical-Profil

Gibt die Stereotypen des Konzepts für das elektrotechnische Design in Abbildung A.19 wieder.

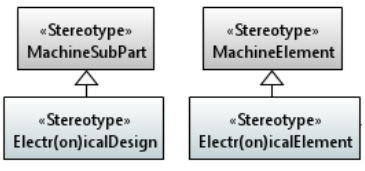


Abbildung A.19: ElectricalView-Profil

Electr(on)icalDesign

Beschreibung: Ein `Electr(on)icalDesign` ist eine Spezialisierung des Stereotyps `MachineSubPart`. Der `Electr(on)icalDesign`-Stereotyp wird benötigt, um rekursive Teilkonstruktionen des Elektroingenieurs zu erstellen.
Abstrakt: Nein
Generalisierung: MachineSubPart
Erweiterung: Class, Property
Constraints: [1]

```
if not self.base_Class->isEmpty() then
      --Electr(on)icalDesign stereotype can only be
      applied on any class specialized from
      Electr(on)icalDesign from Electrical-Library
      self.base_Class->asSet()-
      >closure(general).name-
      >includes('Electr(on)icalDesign')
else
      --Electr(on)icalDesign stereotype can only be
      applied on any property whose type is special-
      ized from Electr(on)icalDesign from Electrical-
      Library
      self.base_Property.type->asSet()-
      >closure(general).name-
      >includes('Electr(on)icalDesign')
endif
```

Electr(on)icalElement

Beschreibung: Ein `Electr(on)icalElement` ist eine Spezialisierung des Stereotyps `MachineElement'`. Der `'Electr(on)icalElement`-Stereotyp wird benötigt, um E/E/PE-Elemente von anderen Arten von Elementen in Maschinenkonstruktionen zu unterscheiden.
Abstrakt: Nein
Generalisierung: MachineElement
Erweiterung: Class, Property
Constraints: [1]

```
if not self.base_Class->isEmpty() then
      --Electr(on)icalElement stereotype can only be
      applied on any class specialized from
      Electr(on)icalElement from Electrical-Library
      self.base_Class->asSet()-
      >closure(general).name-
      >includes('Electr(on)icalDesign')
else
      --Electr(on)icalElement stereotype can only be
      applied on any property whose type is special-
      ized from Electr(on)icalElement from Electri-
      cal-Library
      self.base_Property.type->asSet()-
      >closure(general).name-
      >includes('Electr(on)icalElement')
endif
```

A.9.7HydraulicView

Hydraulic-Bibliothek

HydraulicConstruction

Beschreibung: Ein `HydraulicConstruction` spezialisiert `SubPart` und repräsentiert die hydraulische Sichtweise für ein entsprechendes Maschinenteil. Untergeordnete E/E/PE-Maschinenteile und Maschinenelemente sind direkt über eine Komposition verbunden und definieren die Attribute `machinesubpartchild` und `machineelement` neu. Es ist nicht zwingend erforderlich, dass ein Maschinenteil aus weiteren Teilsystemen besteht. Allerdings muss mindestens ein untergeordnetes Teil oder Element zugeordnet werden. Durch Definition der Attribute muss ein neues Constraint vergeben werden.
Abstrakt: Nein
Generalisierung: SubPart
Applied Stereotype: HydraulicConstruction
Constraints: [1]

```
--HydraulicConstruction should have an hydraulicas-
semblychild or an hydraulicelement
self.attributes-
>select(a | a.name.
startsWith('hydraulicassemblychild'))->notEmpty() or
self.attributes-
>select(b | b.name.startsWith('hydraulicelement'))-
>notEmpty()
```

HydraulicElement

Beschreibung: Ein `HydraulicElement` ist ein Bestandteil eines `HydraulicDesign` und repräsentiert die unterste Ebene im hydraulischen Aufbau. Hydraulische Elemente einer Maschine erben von `MachineElement`. Jedes `Electr(on)icalElement` besitzt eine Assoziation zu einem `MechanicalElement` und in den Spezialfällen `H_Sensor` und `H_Actor` zu `E_Sensor_Input` und `E_Actor_Output`, um die E/E/PE-Sicht in Beziehung zu setzen. Durch Definition des Attributs muss ein neues Constraint vergeben werden.
Abstrakt: Nein
Generalisierung: MachineElement
Applied Stereotype: HydraulicElement
Constraints: [1]

```
--HydraulicElement should have a mechanicalelement,
a e_sensor or a e_actor associated
self.mechanicalelement<>null or self.e_sensor<>null
or self.e_actor<>null
```

H_Sensor

Beschreibung: Ein `H_Sensor` ist ein spezielles `HydraulicElement` und dient der Modellierung von hydraulischen Sensoren. Er spezialisiert hierfür `HydraulicElement`. Er definiert die Assoziation mit einem `Machineelement` aus `HydraulicElement` für E_Sensor_Input neu, um die E/E/PE-Sicht in Beziehung zu setzen. Das notwendige Constraint ist bereits in `HydraulicElement` beschrieben.
Abstrakt: Nein
Generalisierung: HydraulicElement
Applied Stereotype: HydraulicElement

H_Actor

Beschreibung: Ein `H_Actor` ist ein spezielles `HydraulicElement` und dient der Modellierung von hydraulischen Aktoren. Er spezialisiert hierfür `HydraulicElement`. Er definiert die Assoziation mit einem `Machineelement` aus `HydraulicElement` für `E_Actor_Output` neu, um die E/E/PE-Sicht in Beziehung zu setzen. Das notwendige Constraint ist bereits in `HydraulicElement` beschrieben.
Abstrakt: Nein
Generalisierung: HydraulicElement
Applied Stereotype: HydraulicElement

Hydraulic-Profil

Gibt die Stereotypen des Konzepts für die hydraulische Konstruktion in Abbildung A.20 wieder.

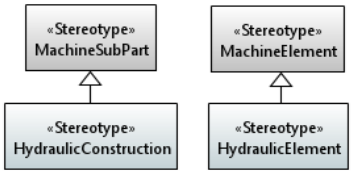


Abbildung A.20: HydraulicView-Profil

HydraulicConstruction

Beschreibung: Eine `HydraulicConstruction` ist eine Spezialisierung des Stereotyps `MachineSubPart`. Der `HydraulicConstruction`-Stereotyp wird benötigt, um rekursive Teilkonstruktionen der hydraulischen Konstruktion zu erstellen.
Abstrakt: Nein
Generalisierung: MachineSubPart
Erweiterung: Class, Property
Constraints: [1]

```
if not self.base_Class->isEmpty() then
      --HydraulicConstruction stereotype can only be
      applied on any class specialized from Hydrau-
      licConstruction from Hydraulic-Library
      self.base_Class->asSet()-
      >closure(general).name-
      >includes('HydraulicConstruction')
else
      -- HydraulicConstruction stereotype can only be
      applied on any property whose type is special-
      ized from HydraulicConstruction from Hydraulic-
      Library
      self.base_Property.type->asSet()-
      >closure(general).name-
      >includes('HydraulicConstruction')
endif
```

HydraulicElement

Beschreibung: Ein `HydraulicElement` ist eine Spezialisierung des Stereotyps `MachineElement`. Der `HydraulicElement`-Stereotyp wird benötigt, um hydraulische Elemente von anderen Arten von Elementen in Maschinenkonstruktionen zu unterscheiden.
Abstrakt: Nein
Generalisierung: MachineElement
Erweiterung: Class, Property
Constraints: [1]

```
if not self.base_Class->isEmpty() then
     --HydraulicElement stereotype can only be ap-
     plied on any class specialized from Hydrau-
     licElement from Hydraulic-Library
     self.base_Class->asSet()-
     >closure(general).name-
     >includes('HydraulicElement')
else
     -- HydraulicElement stereotype can only be ap-
     plied on any property whose type is specialized
     from HydraulicElement from Hydraulic-Library
     self.base_Property.type->asSet()-
     >closure(general).name-
     >includes('HydraulicElement')
endif
```

A.9.8PneumaticView

Pneumatic-Bibliothek

PneumaticConstruction

Beschreibung: Ein `PneumaticConstruction` spezialisiert `SubPart` und repräsentiert die hydraulische Sichtweise für ein entsprechendes Maschinenteil. Untergeordnete E/E/PE-Maschinenteile und Maschinenelemente sind direkt über eine Komposition verbunden und definieren die Attribute `machinesubpartchild` und `machineelement` neu. Es ist nicht zwingend erforderlich, dass ein Maschinenteil aus weiteren Teilsystemen besteht. Allerdings muss mindestens ein untergeordnetes Teil oder Element zugeordnet werden. Durch Definition der Attribute muss ein neues Constraint vergeben werden.
Abstrakt: Nein
Generalisierung: SubPart
Applied Stereotype: PneumaticConstruction
Constraints: [1]

```
--PneumaticConstruction should have an pneumaticas-
semblychild or an pneumaticelement
self.attributes-
>select(a | a.name.
startsWith('pneumaticassemblychild'))->notEmpty() or
self.attributes-
>select(b | b.name.startsWith('pneumaticelement'))-
>notEmpty()
```

PneumaticElement

Beschreibung: Ein `PneumaticElement` ist ein Bestandteil eines `PneumaticDesign` und repräsentiert die unterste Ebene im hydraulischen Aufbau. Hydraulische Elemente einer Maschine erben von `MachineElement`. Jedes `Electr(on)icalElement` besitzt eine Assoziation zu einem `MechanicalElement` und in den Spezialfällen `P_Sensor` und `P_Actor` zu `E_Sensor_Input` und `E_Actor_Output`, um die E/E/PE-Sicht in Beziehung zu setzen. Durch Definition des Attributs muss ein neues Constraint vergeben werden.
Abstrakt: Nein
Generalisierung: MachineElement
Applied Stereotype: PneumaticElement
Constraints: [1]

```
--PneumaticElement should have a mechanicalelement,
a e_sensor or a e_actor associated
self.mechanicalelement<>null or self.e_sensor<>null
or self.e_actor<>null
```

P_Sensor

Beschreibung: Ein `P_Sensor` ist ein spezielles `PneumaticElement` und dient der Modellierung von hydraulischen Sensoren. Er spezialisiert hierfür `PneumaticElement`. Er definiert die Assoziation mit einem `Machineelement` aus `PneumaticElement` für `E_Sensor_Input` neu, um die E/E/PE-Sicht in Beziehung zu setzen. Das notwendige Constraint ist bereits in `PneumaticElement` beschrieben.
Abstrakt: Nein
Generalisierung: PneumaticElement
Applied Stereotype: PneumaticElement

H_Actor

Beschreibung: Ein `P_Actor` ist ein spezielles `PneumaticElement` und dient der Modellierung von hydraulischen Aktoren. Er spezialisiert hierfür `PneumaticElement`. Er definiert die Assoziation mit einem `Machineelement` aus `PneumaticElement` für `E_Actor_Output` neu, um die E/E/PE-Sicht in Beziehung zu setzen. Das notwendige Constraint ist bereits in `PneumaticElement` beschrieben.
Abstrakt: Nein
Generalisierung: PneumaticElement
Applied Stereotype: PneumaticElement

Pneumatic-Profil

Gibt die Stereotypen des Konzepts für die pneumatische Konstruktion in Abbildung A.21 wieder.

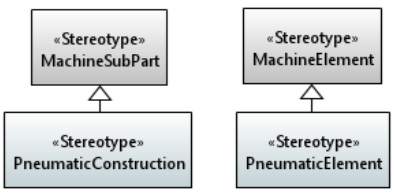


Abbildung A.21: PneumaticView-Profil

PneumaticConstruction

Beschreibung: Eine `PneumaticConstruction` ist eine Spezialisierung des Stereotyps `MachineSubPart`. Der `PneumaticConstruction`-Stereotyp wird benötigt, um rekursive Teilkonstruktionen der pneumatischen Konstruktion zu erstellen.
Abstrakt: Nein
Generalisierung: MachineSubPart
Erweiterung: Class, Property
Constraints: [1]

```
if not self.base_Class->isEmpty() then
      --PneumaticConstruction stereotype can only be
      applied on any class specialized from Pneumat-
      icConstruction from Pneumatic-Library
      self.base_Class->asSet()-
      >closure(general).name-
      >includes('PneumaticConstruction')
else
      -- PneumaticConstruction stereotype can only be
      applied on any property whose type is special-
      ized from PneumaticConstruction from Pneumatic-
      Library
      self.base_Property.type->asSet()-
      >closure(general).name-
      >includes('PneumaticConstruction')
endif
```

PneumaticElement

Beschreibung: Ein `PneumaticElement` ist eine Spezialisierung des Stereotyps `MachineElement`. Der `PneumaticElement`-Stereotyp wird benötigt, um pneumatische Elemente von anderen Arten von Elementen in Maschinenkonstruktionen zu unterscheiden.
Abstrakt: Nein
Generalisierung: MachineElement
Erweiterung: Class, Property
Constraints: [1]

```
if not self.base_Class->isEmpty() then
      --PneumaticElement stereotype can only be ap-
      plied on any class specialized from Pneu-
      maticElement from Pneumatic-Library
      self.base_Class->asSet()-
      >closure(general).name-
      >includes('PneumaticElement')
else
      -- PneumaticElement stereotype can only be ap-
      plied on any property whose type is specialized
      from PneumaticElement from Pneumatic-Library
      self.base_Property.type->asSet()-
      >closure(general).name-
      >includes('PneumaticElement')
endif
```

A.9.9 InformaticView

Informatic-Bibliothek

Software

Beschreibung: Eine `Software` spezialisiert `SubPart` und repräsentiert über UML-Komponenten die Sichtweise des Informatikers für ein entsprechendes Maschinen- bzw. Softwareteil. Untergeordnete Softwareteile und Softwareelemente sind direkt über eine Komposition verbunden und definieren die Attribute `machinesubpartchild` und `machineelement` neu. Es ist nicht zwingend erforderlich, dass ein Softwareteil aus weiteren Softwareteilen besteht. Allerdings muss mindestens ein untergeordnetes Teil oder Element zugeordnet werden. Durch Definition der Attribute muss ein neues Constraint vergeben werden.
Abstrakt: Nein
Generalisierung: SubPart
Applied Stereotype: Component, InformaticPart
Constraints: [1]

```
--Software should have an softwarechild or an soft-
wareelement
self.attributes-
>select(a | a.name.startsWith('softwarechild'))-
>notEmpty() or
self.attributes-
>select(b | b.name.startsWith('softwareelement'))-
>notEmpty()
```

SoftwareElement

Beschreibung: Ein `SoftwareElement` ist ein Bestandteil einer `Software` und repräsentiert die unterste Ebene der UML-Komponenten. Softwareelemente einer Maschine erben von `MachineElement`. Jedes `SoftwareElement` besitzt eine Assoziation zu einem `Controller`, um die E/E/PE-Sicht in Beziehung zu setzen und dessen Informationen beim Design zu nutzen. Durch Definition des Attributs muss ein neues Constraint vergeben werden.
Abstrakt: Nein
Generalisierung: MachineElement
Applied Stereotype: Component, InformaticElement
Constraints: [1]

```
--SoftwareElement should have a controller associ-
ated
self.controller<>null
```

Informatic-Profil

Gibt die Stereotypen des Konzepts für das informationstechnische Design in Abbildung A.22 wieder.

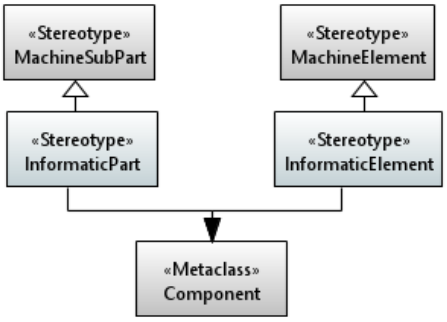


Abbildung A.22: InformaticView-Profil

InformaticPart

Beschreibung: Ein `InformaticPart` ist eine Spezialisierung des Stereotyps `MachineSubPart`. Der `InformaticPart`-Stereotyp wird benötigt, um rekursive Softwareteile zu erstellen. Er erweitert hierfür das Konzept der Komponenten, um Schnittstellen zu definieren, Funktionen zu implementieren, Konfigurationen bereitzustellen und Abhängigkeiten oder Ressourcen zu modellieren.
Abstrakt: Nein
Generalisierung: MachineSubPart
Erweiterung: Component
Erweiterung: Class, Property
Constraints: [1]

```
if not self.base_Class->isEmpty() then
     --InformaticPart stereotype can only be applied
     on any class specialized from Software from In-
     formatic-Library
     self.base_Class->asSet()-
     >closure(general).name->includes('Software')
else
     --InformaticPart stereotype can only be applied
     on any property whose type is specialized from
     Software from Informatic-Library
     self.base_Property.type->asSet()-
     >closure(general).name->includes('Software')
endif
```

InformaticElement

Beschreibung: Ein `InformaticElement` ist eine Spezialisierung des Stereotyps `MachineElement`. Der `InformaticElement`-Stereotyp wird benötigt, um die unterste Ebene der Softwareelemente von anderen Arten von Elementen in Maschinenkonstruktionen zu unterscheiden. Er erweitert hierfür das Konzept der Komponenten, um Schnittstellen zu definieren, Funktionen zu implementieren, Konfigurationen bereitzustellen und Abhängigkeiten oder Ressourcen zu modellieren.
Abstrakt: Nein
Generalisierung: MachineElement
Erweiterung: Component
Erweiterung: Class, Property
Constraints: [1]

```
if not self.base_Class->isEmpty() then
```

```
        --InformaticElement stereotype can only be ap-
        plied on any class specialized from SoftwareEl-
        ement from Informatic-Library
        self.base_Class->asSet()-
        >closure(general).name-
        >includes('SoftwareElement')
    else
        --InformaticElement stereotype can only be ap-
        plied on any property whose type is specialized
        from SoftwareElement from Informatic-Library
        self.base_Property.type->asSet()-
        >closure(general).name-
        >includes('SoftwareElement')
    endif
```

A.9.10 ISO 12100 & ISO 13849

ISO 12100 & ISO 13849-Bibliothek

MachineryRiskItem

Beschreibung: Eine `MachineryRiskItem` spezialisiert `AbstractRisk` und repräsentiert ein Risikoitem zur Spezifikation des Zusammenhangs einer Gefährdung mit Gefährdungssituationen und Effekten. `MachineryRiskItem` hat Kompositionen zu `MachineHazard`, `HazardousSituation` und `MachineryEffect` und definiert dabei die Attribute `riskIndex` für `score`, `hazard` für `harmPotential`, `machineryEffect` für `harm` und `hazardousSituation` für `trigger` neu. Zusätzlich werden `PLr` und `previousRiskIndex` definiert. Durch Definition der Attribute muss ein neues Constraint vergeben werden.
Abstrakt: Nein
Generalisierung: AbstractRisk
Applied Stereotype: MachineryRiskItem
Constraints: [1]

```
--MachineryRiskItem should have a hazard, a hazard-
ousSituation and a machineryEffect
self.hazard<>null and self.attributes-
>select(a | a.name.
startsWith('hazardousSituation'))->notEmpty() and
self.attributes-
>select(b | b.name.
startsWith('machineryEffect'))->notEmpty()
```

MachineHazard

Beschreibung: Ein `MachineHazard` spezialisiert `Hazard` und repräsentiert eine abstrakte Maschinengefährdung. Um die Abhängigkeit zu Maschinenteilen, die die Gefährdung auslösen, herzustellen wird eine Assoziation `hazardArea` zu `MachinePart` hergestellt. Durch Definition dieses Attributes muss ein neues Constraint vergeben werden.
Abstrakt: Ja
Generalisierung: Hazard
Applied Stereotype: Situation
Constraints: [1]

```
--MachineHazard should have a hazardArea
self.attributes->select(a | a.name.
startsWith('hazardArea'))->notEmpty()
```

MechanicalHazard

Beschreibung: Ein `MechanicalHazard` spezialisiert `MachineHazard` und repräsentiert eine mechanische Gefährdung. Weitere Informationen zu gängigen Gefährdungen sind unter Anhang A.5 zu finden.
Abstrakt: Nein
Generalisierung: MachineHazard
Applied Stereotype: Situation

ElectricalHazard

Beschreibung: Ein `ElectricalHazard` spezialisiert `MachineHazard` und repräsentiert eine elektrische Gefährdung. Weitere Informationen zu gängigen Gefährdungen sind unter Anhang A.5 zu finden.
Abstrakt: Nein
Generalisierung: MachineHazard
Applied Stereotype: Situation

ThermalHazard

Beschreibung: Ein `ThermalHazard` spezialisiert `MachineHazard` und repräsentiert eine thermische Gefährdung. Weitere Informationen zu gängigen Gefährdungen sind unter Anhang A.5 zu finden.
Abstrakt: Nein
Generalisierung: MachineHazard
Applied Stereotype: Situation

HazardThroughNoise

Beschreibung: Ein `HazardThroughNoise` spezialisiert `MachineHazard` und repräsentiert eine Gefährdung aufgrund von Lärm. Weitere Informationen zu gängigen Gefährdungen sind unter Anhang A.5 zu finden.
Abstrakt: Nein
Generalisierung: MachineHazard
Applied Stereotype: Situation

HazardThroughVibration

Beschreibung: Ein `HazardThroughVibration` spezialisiert `MachineHazard` und repräsentiert eine Gefährdung aufgrund von Vibration. Weitere Informationen zu gängigen Gefährdungen sind unter Anhang A.5 zu finden.
Abstrakt: Nein
Generalisierung: MachineHazard
Applied Stereotype: Situation

HazardThroughRadiation

Beschreibung: Ein `HazardThroughRadiation` spezialisiert `MachineHazard` und repräsentiert eine Gefährdung durch Strahlung. Weitere Informationen zu gängigen Gefährdungen sind unter Anhang A.5 zu finden.
Abstrakt: Nein

Generalisierung: MachineHazard
Applied Stereotype: Situation

HazardThroughMaterialAndSubstances

Beschreibung: Ein `HazardThroughMaterialAndSubstances` spezialisiert `MachineHazard` und repräsentiert Gefährdungen durch Materialien und Substanzen. Weitere Informationen zu gängigen Gefährdungen sind unter Anhang A.5 zu finden.
Abstrakt: Nein
Generalisierung: MachineHazard
Applied Stereotype: Situation

ErgonomicHazard

Beschreibung: Ein `ErgonomicHazard` spezialisiert `MachineHazard` und repräsentiert eine ergonomische Gefährdung. Weitere Informationen zu gängigen Gefährdungen sind unter Anhang A.5 zu finden.
Abstrakt: Nein
Generalisierung: MachineHazard
Applied Stereotype: Situation

OperatingEnvironmentHazard

Beschreibung: Ein `OperatingEnvironmentHazard` spezialisiert `MachineHazard` und repräsentiert eine Gefährdung im Zusammengang mit der Einsatzumgebung. Weitere Informationen zu gängigen Gefährdungen sind unter Anhang A.5 zu finden.
Abstrakt: Nein
Generalisierung: MachineHazard
Applied Stereotype: Situation

CombinedHazard

Beschreibung: Ein `CombinedHazard` spezialisiert `MachineHazard` und repräsentiert eine Kombination von Gefährdungen, beispielsweise bei Defekten oder Fehlfunktionen an der Steuerung. Weitere Informationen zu gängigen Gefährdungen sind unter Anhang A.5 zu finden.
Abstrakt: Nein
Generalisierung: MachineHazard
Applied Stereotype: Situation

HazardousSituation

Beschreibung: Eine `HazardousSituation` spezialisiert `DysfunctionalEvent` und `Scenario` und repräsentiert dabei eine gefahrbringende Situation mit einem gefährdenden Event und mindestens einer der Situation ausgesetzten Person. `HazardousSituation` hat daher eine Komposition zu `HazardousEvent` und eine Assoziation zu `ExposedActor` sowie definiert dabei die Attribute `hazardousEvent` für `scenarioStep` und `avoidance` für `likelihood` neu. Zusätzlich wird `avoidance` und `premitigationAvoidance` mit dem ValueType `Avoidance` definiert. Durch Definition der Attribute muss ein neues Constraint vergeben werden.
Abstrakt: Nein

Generalisierung: DysfunctionalEvent, Scenario
Applied Stereotype: HazardousSituation
Constraints: [1]

```
--HazardousSituation should have an exposedActor, a
hazardousEvent and an avoidance
self.attributes-
>select(a | a.name.startsWith('exposedActor'))-
>notEmpty() and
self.hazardousEvent<>null and self.avoidance<>null
```

ExposedActor

Beschreibung: Ein `ExposedActor` repräsentiert eine der Gefährdungssituation ausgesetzte Person.
Abstrakt: Nein
Generalisierung: ExposedActor
Applied Stereotype: ExposedActor

HazardousEvent

Beschreibung: Ein `HazardousEvent` spezialisiert `AbstractEvent` und repräsentiert dabei ein gefährdendes Event mit mindestens einer Ursache und möglichen Betriebsarten. `HazardousEvent` hat daher eine Komposition zu `MachineContextCause` und eine Assoziation zu `OperationMode` sowie definiert dabei das Attribut `frequencyOfExposure`, mit dem ValueType `FrequencyOfExposure` für `likelihood`, neu. Durch Definition der Attribute muss ein neues Constraint vergeben werden.
Abstrakt: Nein
Generalisierung: AbstractEvent
Applied Stereotype: HazardousEvent
Constraints: [1]

```
--HazardousEvent should have a machinecontextcause
and a frequencyOfExposure
self.attributes->select(a | a.name.
startsWith('machinecontextcause'))->notEmpty() and
self.frequencyOfExposure<>null
```

OperationMode

Beschreibung: Ein `OperationMode` spezialisiert `AbstractEvent` und repräsentiert dabei ein Betriebsmodus oder eine Betriebsart.
Abstrakt: Nein
Generalisierung: AbstractEvent, OperationMode
Applied Stereotype: OperationMode

MachineContextCause

Beschreibung: Ein `MachineContextCause` spezialisiert `AbstractCause` und repräsentiert dabei eine Ursache, die zu einem gefährdenden Event und zur gefahrbringenden Situation führt. `MachineContextCause` definiert hierfür das Attribut `occurrence` für `occurrence` und `premitigationOccurrence` für `premitigationOccurrence` mit dem ValueType `Occurence` neu. Zusätzlich wird `lifePhase` mit dem ValueType `LifePhase` definiert. Durch Definition der Attribute muss ein neues Constraint vergeben werden.
Abstrakt: Nein

Generalisierung: AbstractCause
Applied Stereotype: Situation
Constraints: [1]

```
--MachineContextCause should have an occurrence and
a lifePhase
self.occurence<>null and self.attributes-
>select(a | a.name.startsWith('lifePhase'))-
>notEmpty()
```

MachineryEffect

Beschreibung: Ein `MachineryEffect` spezialisiert `AbstractEffect` und repräsentiert dabei eine Auswirkung einer gefahrbringenden Situation. `MachineryEffect` definiert hierfür das Attribut `severity` für `severity` und `premitigationSeverity` für `premitigationSeverity` mit dem ValueType `Severity` neu. Durch Definition der Attribute muss ein neues Constraint vergeben werden.
Abstrakt: Nein
Generalisierung: AbstractEffect
Applied Stereotype: Situation
Constraints: [1]

```
--MachineryEffect should have a severity
self.severity<>null
```

Avoidance

Beschreibung: `Avoidance` ist ein ValueType der die Möglichkeit zur Vermeidung bzw. Begrenzung einer Gefährdungssituation definiert. Er besitzt die Werte `-`, `P1` und `P2`, wobei `-` bedeutet, dass die Gefährdungssituation nicht mehr eintritt. Entsprechend eines anderen Risikographen können hier ebenso andere Metriken gewählt werden.
Applied Stereotype: Enumeration, ValueType

FrequencyOfExposure

Beschreibung: `FrequencyOfExposure` ist ein ValueType der die Häufigkeit bzw. Dauer der Exposition im Gefährdungsereignis in der Ursache definiert. Er besitzt die Werte `-`, `F1` und `F2` wobei `-` bedeutet, dass die Häufigkeit bzw. Dauer nicht mehr relevant ist. Entsprechend eines anderen Risikographen können hier ebenso andere Metriken gewählt werden.
Applied Stereotype: Enumeration, ValueType

Occurrence

Beschreibung: `Occurence` ist ein ValueType der die Eintrittswahrscheinlichkeit eines Ereignisses in der Ursache definiert. Er besitzt die Werte `-`, `O1` und `O2`, wobei `-` bedeutet, dass die Eintrittswahrscheinlichkeit der Ursache nicht mehr relevant ist. Entsprechend eines anderen Risikographen können hier ebenso andere Metriken gewählt werden.
Applied Stereotype: Enumeration, ValueType

Severity

Beschreibung: `Severity` ist ein ValueType der die Schwere der Verletzungen und Auswirkungen im Effekt definiert. Er besitzt die Werte `-`, `S1` und `S2`, wobei `-` bedeutet, dass der Effekt nicht mehr auftritt. Entsprechend eines anderen Risikographen können hier ebenso andere Metriken gewählt werden.

Applied Stereotype: Enumeration, ValueType

MeasureAgainstCCF

Beschreibung: `MeasureAgainstCCF` ist ein ValueType der Maßnahmen gegen CCF anhand der Norm ISO13849 und unter Anhang A.6.6 beschriebene Maßnahmen definiert. Der ValueType wird für die SBBM verwendet. Die Werte verhalten sich dabei analog zu A.6.6 Verfahren zur Punktevergabe & Quantifizierung für Maßnahmen gegen CCF Tabelle A**.5**.
Applied Stereotype: Enumeration, ValueType

PL

Beschreibung: `PL` ist ein ValueType der das Performance Level mit den Werten `a` bis `e` definiert, um die Einstufung bei der Risikobeurteilung und Nachweis der Risikominderung vorzunehmen. Der ValueType wird für die SBBM verwendet.
Applied Stereotype: Enumeration, ValueType

Category

Beschreibung: `Category` ist ein ValueType der die Kategorie für die SBBM definiert. Die Kategorie teilt sich in die Werte `catB`, `cat1`, `cat2`, `cat3` und `cat4`.
Applied Stereotype: Enumeration, ValueType

LifePhase

Beschreibung: `LifePhase` ist ein ValueType der die relevanten Lebensphasen bei der Risikobeurteilung zur Verfügung stellt. Hierbei sind nach ISO 12100 die Lebensphasen Transport, Montage, Installation, in Betrieb nehmen, Betrieb, Einlernen, Instandhaltung, Fehlersuche und Beseitigung, Reinigung, Wartung, außer Betrieb nehmen, Demontage und Entsorgung berücksichtigt.
Applied Stereotype: Enumeration, ValueType

ISO 12100 & ISO 13849-Profil

DependabilityRequirement

Beschreibung: Ein `DependabilityRequirement` ist eine Spezialisierung des SysML Stereotyps `AbstractRequirement` und `Block`. Er stellt den übergeordneten Stereotypen für die domänenspezifischen Anforderungen dar.
Abstrakt: Ja
Generalisierung: AbstractRequirement

MachineLimitRequirement

Beschreibung: Ein `MachineLimitRequirement` ist eine Spezialisierung aus dem Stereotyp `DependabilityRequirement` zur Modellierung der Grenzen der Maschine.
Abstrakt: Nein
Generalisierung: DependabilityRequirement

RiskReductionMeasure

Beschreibung: Ein `RiskReductionMeasure` ist eine Spezialisierung aus dem Stereotyp `DependabilityRequirement` zur Modellierung der risikomindernden Maßnahmen.

Abstrakt: Nein
Generalisierung: DependabilityRequirement

FunctionalSafetyRequirement

Beschreibung: Ein `FunctionalSafetyRequirement` ist eine Spezialisierung aus dem Stereotyp `DependabilityRequirement` zur Modellierung der Anforderungen an die funktionale Sicherheit.
Abstrakt: Nein
Generalisierung: DependabilityRequirement

TechnicalSafetyRequirement

Beschreibung: Ein `TechnicalSafetyRequirement` ist eine Spezialisierung aus dem Stereotyp `DependabilityRequirement` zur Modellierung der technischen Anforderungen an die funktionale Sicherheit.
Abstrakt: Nein
Generalisierung: DependabilityRequirement

HardwareRequirement

Beschreibung: Ein `HardwareRequirement` ist eine Spezialisierung aus dem Stereotyp `DependabilityRequirement` zur Modellierung der Hardwareanforderungen an die funktionale Sicherheit.
Abstrakt: Nein
Generalisierung: DependabilityRequirement

SoftwareRequirement

Beschreibung: Ein `SoftwareRequirement` ist eine Spezialisierung aus dem Stereotyp `DependabilityRequirement` zur Modellierung der Softwareanforderungen an die funktionale Sicherheit.
Abstrakt: Nein
Generalisierung: DependabilityRequirement

UserInformationRequirement

Beschreibung: Ein `UserInformationRequirement` ist eine Spezialisierung aus dem Stereotyp `DependabilityRequirement` zur Modellierung der Anforderungen zur Benutzerinformation in Form von Hinweisen und Kundendokumentation.
Abstrakt: Nein
Generalisierung: DependabilityRequirement

PLDecompose

Beschreibung: Ein `PLDecompose` ist eine Spezialisierung aus dem SysML Stereotyp `DeriveReqt` und eine Erweiterung von `Abstraction` zur Unterteilung unterschiedlicher Sicherheitsanforderungen, die einen Nachweis der Risikominderung nach PL benötigen.
Abstrakt: Nein
Generalisierung: DeriveReqt
Erweiterung: Abstraction

Failure

Beschreibung: `Failure` ist eine Spezialisierung aus dem Stereotyp `FailureMode`, um verschiedene Fehlerarten im Maschinenbau zu beschreiben.
Abstrakt: Nein
Generalisierung: FailureMode

CCF

Beschreibung: `CCF` ist eine Spezialisierung aus dem Stereotyp `Failure`, um CCF zu modellieren.
Abstrakt: Nein
Generalisierung: Failure

DangerousFailure

Beschreibung: `DangerousFailure` ist eine Spezialisierung aus dem Stereotyp `Failure`, um potenzielle gefährliche Fehler in der Modellierung zu berücksichtigen.
Abstrakt: Nein
Generalisierung: Failure

SystematicFailure

Beschreibung: `DangerousFailure` ist eine Spezialisierung aus dem Stereotyp `Failure`, um potenzielle bekannte systematische Fehler in der Modellierung zu berücksichtigen.
Abstrakt: Nein
Generalisierung: Failure

FaultExclusion

Beschreibung: `FaultExclusion` ist eine Spezialisierung aus dem Stereotyp `Fault`, um Fehlerausschlüsse von Komponenten in der Modellierung zu berücksichtigen.
Abstrakt: Nein
Generalisierung: Fault

PermanentFault

Beschreibung: `PermanentFault` ist eine Spezialisierung aus dem Stereotyp `Fault`, um bestimmte permanente Fehler in der Modellierung zu berücksichtigen.
Abstrakt: Nein
Generalisierung: Fault

MachineControlSystem

Beschreibung: Ein `MachineControlSystem` ist eine Spezialisierung aus dem SysML Stereotyp Block und eine Erweiterung von `Class`, um Steuerungssysteme einzubinden, sofern die Framework-Erweiterung ohne Stakeholder-spezifische Sichten genutzt wird.
Abstrakt: Nein
Generalisierung: Block
Erweiterung: Class

SafetyRelatedPartOfAControlSystem

Beschreibung: Ein `SafetyRelatedPartOfAControlSystem` ist eine Spezialisierung aus dem Stereotyp `MachineControlSystem`, um sicherheitsrelevante Teile eines Steuerungssystems einzubinden, sofern die Framework-Erweiterung ohne Stakeholder-spezifische Sichten genutzt wird.
Abstrakt: Nein
Generalisierung: MachineControlSystem

HazardousEvent

Beschreibung: Ein `HazardousEvent` ist eine Spezialisierung aus Stereotyp `Situation`, um das Gefährdungsereignis darzustellen.
Abstrakt: Nein
Generalisierung: Situation
Constraints: [1]

```
--HazardousEvent stereotype can only be applied on
any class specialized from HazardousEvent from ISO
12100 & ISO 13849-Library
self.base_Class->asSet()->closure(general).name-
>includes('HazardousEvent')
```

HazardousSituation

Beschreibung: Eine `HazardousSituation` ist eine Spezialisierung aus Stereotyp `Situation`, um die Gefährdungssituation darzustellen.
Abstrakt: Nein
Generalisierung: Situation
Constraints: [1]

```
--HazardousSituation stereotype can only be applied
on any class specialized from HazardousSituation
from ISO 12100 & ISO 13849-Library
self.base_Class->asSet()->closure(general).name-
>includes('HazardousSituation')
```

MachineryRiskItem

Beschreibung: Ein `MachineryRiskItem` ist eine Spezialisierung aus dem SysML Stereotyp `Block` und eine Erweiterung von `Class`, um das Risikoitem dazustellen, sofern die Framework-Erweiterung ohne Stakeholder-spezifische Sichten genutzt wird.
Abstrakt: Nein
Generalisierung: Block
Erweiterung: Class
Constraints: [1]

```
--MachineryRiskItem stereotype can only be applied
on any class specialized from MachineryRiskItem from
ISO 12100 & ISO 13849-Library
self.base_Class->asSet()->closure(general).name-
>includes('MachineryRiskItem')
```

OperationMode

Beschreibung: Ein `OperationMode` ist eine Erweiterung von `UseCase`, um Betriebsarten und -modi in Abhängigkeit mit der Risikobeurteilung einzubinden.
Abstrakt: Nein
Erweiterung: UseCase

Constraints: [1]
```
--OperationMode stereotype can only be applied on
any class specialized from OperationMode from ISO
12100 & ISO 13849-Library
self.base_Class->asSet()->closure(general).name-
>includes('OperationMode')
```

ExposedActor

Beschreibung: Ein `ExposedActor` ist eine Erweiterung von `Actor`, um einer Situation ausgesetzte Beteiligte abhängig der Risikobeurteilung zu beschreiben.
Abstrakt: Nein
Erweiterung: UseCase
Constraints: [1]
```
--ExposedActor stereotype can only be applied on any
class specialized from ExposedActor from ISO 12100 &
ISO 13849-Library
self.base_Class->asSet()->closure(general).name-
>includes('ExposedActor')
```

A.9.11 DFT

DFT-Bibliothek

DFTree

Beschreibung: Ein `DFTree` spezialisiert `FTATree` und repräsentiert den neuen dynamischen Fehlerbaum. `DFTree` hat eine Komposition zu `DFTTopEvent`, die `topEvent` neu definiert. Zusätzlich definiert `probabilityOfFailure` die maximale Ausfallrate für `probability` neu und erstellt das Attribut `failureExposureTime` zur Angabe der Laufzeit, bis zu dem die maximale Ausfallrate unterschritten bleiben muss.
Abstrakt: Nein
Generalisierung: FTATree
Applied Stereotype: DFTree

DFTTopEvent

Beschreibung: Ein `DFTTopEvent` spezialisiert `TopEvent` und repräsentiert das neue dynamische Top-Event. `DFTTopEvent` definiert zusätzlich zum vorhandenen `probability` aus `TopEvent` das abgeleitete Attribut `meanTimeToFailure`.
Abstrakt: Nein
Generalisierung: TopEvent
Applied Stereotype: DFTTopEvent

PAND

Beschreibung: Ein `PAND` spezialisiert `AND` und repräsentiert damit ein PAND-Gate. Zusätzlich wird ein Constraint vergeben, um sicherzustellen, dass mindestens zwei Eingangsereignisse und ein Ausgangsereignis verbunden sind.
Abstrakt: Nein
Generalisierung: AND
Applied Stereotype: PAND
Constraints: [1]
```
--PAND should have at least two sourceEvent and one
targetEvent
```

```
self.attributes-
>select(a | a.name.startsWith('sourceEvent'))-
>size()>= 2 and self.targetEvent<>null
```

POR

Beschreibung: Ein `POR` spezialisiert `OR` und repräsentiert damit ein POR-Gate. Zusätzlich wird ein Constraint vergeben, um sicherzustellen, dass mindestens zwei Eingangsereignisse und ein Ausgangsereignis verbunden sind.
Abstrakt: Nein
Generalisierung: OR
Applied Stereotype: POR
Constraints: [1]

```
--POR should have at least two sourceEvent and one
targetEvent
self.attributes-
>select(a | a.name.startsWith('sourceEvent'))-
>size()>= 2 and self.targetEvent<>null
```

SEQ

Beschreibung: Ein `SEQ` spezialisiert `SEQ` und repräsentiert damit ein SEQ-Gate für DFTs. Zusätzlich wird ein Constraint vergeben, um sicherzustellen, dass mindestens zwei Eingangsereignisse und ein Ausgangsereignis verbunden sind.
Abstrakt: Nein
Generalisierung: SEQ
Applied Stereotype: SEQ
Constraints: [1]

```
--SEQ should have at least two sourceEvent and one
targetEvent
self.attributes-
>select(a | a.name.startsWith('sourceEvent'))-
>size()>= 2 and self.targetEvent<>null
```

SPARE

Beschreibung: Ein `SPARE` spezialisiert `Gate` und repräsentiert damit ein Spare-Gate. Die Attribute `primaryComponent` für die primäre Komponente und `spareComponent` für die Ersatzkomponenten bilden einen Subset aus `sourceEvent`. Das Attribut `spareType` mit dem ValueType `SPAREType` weist einen entsprechenden Spare-Typ zu. Zusätzlich wird ein Constraint vergeben, um sicherzustellen, dass ein auslösendes Ereignis und ein abhängiges Ereignis verbunden sind.
Abstrakt: Nein
Generalisierung: Gate
Applied Stereotype: SPARE
Constraints: [1]

```
--SPARE should have one primaryComponent, one spare-
Component, one spareType and one targetEvent
self.primaryComponent<>null and self.attributes-
>select(a | a.name.startsWith('spareComponent'))-
>notEmpty() and self.targetEvent<>null and
self.spareType<>null
```

SPAREType

Beschreibung: `SPAREType` ist ein ValueType der die Art eines SPARE-Gates angibt. Er besitzt die Werte `CSP`, `WSP` und `HSP`, die Cold Standby Spare (CSP), Warm Standby Spare (WSP) und Hot Standby Spare (HSP) repräsentieren.
Applied Stereotype: Enumeration, ValueType

FDEP

Beschreibung: Ein `FDEP` spezialisiert `Gate` und repräsentiert damit ein FDEP-Gate. Das Attribut `triggerEvent` definiert `sourceEvent` neu, um das auslösende Ereignis zu verbinden, und das Attribut `dependentEvent` wird definiert, um abhängige Ereignisse zu verbinden. Zusätzlich wird ein Constraint vergeben, um sicherzustellen, dass ein auslösendes Ereignis und ein abhängiges Ereignis verbunden sind.
Abstrakt: Nein
Generalisierung: Gate
Applied Stereotype: FDEP
Constraints: [1]

```
--FDEP should have one triggerEvent and one de-
pendentEvent
self.triggerEvent<>null and self.attributes-
>select(a | a.name.startsWith('dependentEvent'))-
>notEmpty()
```

DynamicBE

Beschreibung: Ein `DynamicBE` spezialisiert `BasicEvent` und repräsentiert damit ein dynamisches Basisereignis. Die Komposition zu Distribution definiert `probability` für `distribution` neu. Zusätzlich werden die Attribute `dormancy` und `coverageFactor` definiert. `DynamicBE` ist ebenso zu einem `MachineElement` assoziiert, um die Abhängigkeit zu den Stakeholdern von MMBSEFE herzustellen. Zusätzlich wird ein Constraint vergeben, um sicherzustellen, dass alle notwendigen Objekte verbunden und Werte zur Berechnung angegeben sind.
Abstrakt: Nein
Generalisierung: Gate
Applied Stereotype: DynamicBE
Constraints: [1]

```
--DynamicBE should have a distribution, a sourceGate
and a machineelement
self.distribution<>null and self.sourceGate<>null
and self.machineelement<>null
```

UnrepairableBE

Beschreibung: Ein `UnrepairableBE` spezialisiert `DynamicBE` und repräsentiert damit ein dynamisches nicht reparierbares Basisereignis. Es wird kein zusätzliches Attribut benötigt.
Abstrakt: Nein
Generalisierung: DynamicBE
Applied Stereotype: UnrepairableBE

RepairableBE

Beschreibung: Ein `RepairableBE` spezialisiert `DynamicBE` und repräsentiert damit ein dynamisches reparierbares Basisereignis. Zur Darstellung der Reparaturrate wird das Attribut `repairRate` definiert. Zusätzlich wird ein Constraint vergeben, um sicherzustellen, dass die Reparaturrate angegeben ist.
Abstrakt: Nein
Generalisierung: DynamicBE
Applied Stereotype: RepairableBE
Constraints: [1]

```
--RepairableBE should have a repairRate
self.repairRate<>null
```

LatentBE

Beschreibung: Ein `LatentBE` spezialisiert `DynamicBE` und repräsentiert damit ein dynamisches latentes Basisereignis. Zur Darstellung des Inspektionsintervalls wird das Attribut `inspectionInterval` definiert. Zusätzlich wird ein Constraint vergeben, um sicherzustellen, dass der Inspektionsintervall angegeben ist.
Abstrakt: Nein
Generalisierung: DynamicBE
Applied Stereotype: LatentBE
Constraints: [1]

```
--LatentBE should have an inspectionInterval
self.inspectionInterval<>null
```

Distribution

Beschreibung: Eine `Distribution` repräsentiert damit die abstrakte Klasse der Verteilungsfunktionen. Hierfür wird zusätzlich das Attribut `distributionType` definiert, dessen Datentyp sich aus `AperiodicPattern` ableitet, über den zusätzlich eine der Verteilungsfunktionen zugewiesen werden kann.
Abstrakt: Ja
Applied Stereotype: Distribution

DistributionType

Beschreibung: `DistributionType` ist eine aus `AperiodicPattern` spezialisierter Datentyp zur Spezifikation des Verteilungstyps. Der Datentyp nutzt hierbei das Attribut `distribution` aus `AperiodicPattern`, um eine Verteilungsfunktion zuzuweisen.
Abstrakt: Nein
Generalisierung: AperiodicPattern
Applied Stereotype: DataType, TupleType

Exponential

Beschreibung: Ein `Exponential` spezialisiert `NFP_CommonType` und `Distribution`, um exponentielle Verteilungsfunktionen zu modellieren. Hierfür wird zusätzlich das Attribut `failureLambda` definiert und für den Fall der internen Berechnung die Operation `exp(mean)` aus MARTE angegeben. Zusätzlich wird ein Constraint vergeben, um sicherzustellen, dass `failureLambda` angegeben ist.
Abstrakt: Nein

Generalisierung: NFP_CommonType, Distribution
Applied Stereotype: Distribution, TupleType
Constraints: [1]
```
--Exponential should have a failureLambda
self.failureLambda<>null
```

Weibull

Beschreibung: Ein `Weibull` spezialisiert `NFP_CommonType` und `Distribution`, um Weibull-Verteilungsfunktionen zu modellieren. Hierfür werden zusätzlich die Attribute `failureRate` und `shape` definiert und für den Fall der internen Berechnung die Operation `weibull(mean, shape)` angegeben. Zusätzlich wird ein Constraint vergeben, um sicherzustellen, dass `failureRate` und `shape` angegeben sind.
Abstrakt: Nein
Generalisierung: NFP_CommonType, Distribution
Applied Stereotype: Distribution, TupleType
Constraints: [1]
```
--Weibull should have a failureRate and a shape
self.failureRate<>null and self.shape<>null
```

LogNormal

Beschreibung: Ein `LogNormal` spezialisiert `NFP_CommonType` und `Distribution`, um Normalverteilungsfunktionen zu modellieren. Hierfür werden zusätzlich die Attribute `failureMean` und `standardDeviation` definiert und für den Fall der internen Berechnung die Operation `normal(mean, standardDev)` aus MARTE angegeben. Zusätzlich wird ein Constraint vergeben, um sicherzustellen, dass `failureMean` und `standardDeviation` angegeben sind.
Abstrakt: Nein
Generalisierung: NFP_CommonType, Distribution
Applied Stereotype: Distribution, TupleType
Constraints: [1]
```
--LogNormal should have a failureMean and a stand-
ardDeviation
self.failureMean<>null and
self.standardDeviation<>null
```

DFT-Profil

DFTree

Beschreibung: Ein `DFTree` ist eine Spezialisierung des Stereotyps `Tree` für den dynamischen Fehlerbaum.
Abstrakt: Nein
Generalisierung: Tree
Erweiterung: Class
Constraints: [1]
```
--DFTree stereotype can only be applied on any class
specialized from DFTree from DFT-Library
self.base_Class->asSet()->closure(general).name-
>includes('DFTree')
```

DFTTopEvent

Beschreibung: Ein `DFTTopEvent` ist eine Spezialisierung des Stereotyps `TopEvent` für das dynamische Top-Event.

Abstrakt: Nein
Generalisierung: TopEvent
Erweiterung: Class, Property
Constraints: [1]

```
if not self.base_Class->isEmpty() then
      --DFTTopEvent stereotype can only be applied on
      any class specialized from DFTTopEvent from
      DFT-Library
      self.base_Class->asSet()-
      >closure(general).name->includes('DFTTopEvent')
else
      --DFTTopEvent stereotype can only be applied on
      any property whose type is specialized from
      DFTTopEvent from DFT-Library
      self.base_Property.type->asSet()-
      >closure(general).name->includes('DFTTopEvent')
endif
```

PAND

Beschreibung: Ein PAND ist eine Spezialisierung des Stereotyps AND für PAND-Gates.
Abstrakt: Nein
Generalisierung: AND
Erweiterung: Class, Property
Constraints: [1]

```
if not self.base_Class->isEmpty() then
      --PAND stereotype can only be applied on any
      class specialized from PAND from DFT-Library
      self.base_Class->asSet()-
      >closure(general).name->includes('PAND')
else
      --PAND stereotype can only be applied on any
      property whose type is specialized from PAND
      from DFT-Library
      self.base_Property.type->asSet()-
      >closure(general).name->includes('PAND')
endif
```

POR

Beschreibung: Ein POR ist eine Spezialisierung des Stereotyps OR für POR-Gates.
Abstrakt: Nein
Generalisierung: OR
Erweiterung: Class, Property
Constraints: [1]

```
if not self.base_Class->isEmpty() then
      --POR stereotype can only be applied on any
      class specialized from POR from DFT-Library
      self.base_Class->asSet()-
      >closure(general).name->includes('POR')
else
      --POR stereotype can only be applied on any
      property whose type is specialized from POR
      from DFT-Library
      self.base_Property.type->asSet()-
      >closure(general).name->includes('POR')
endif
```

SEQ

Beschreibung: Ein `SEQ` ist eine Spezialisierung des Stereotyps `SEQ` für SEQ-Gates.
Abstrakt: Nein
Generalisierung: SEQ
Erweiterung: Class, Property
Constraints: [1]

```
if not self.base_Class->isEmpty() then
      --SEQ stereotype can only be applied on any
      class specialized from SEQ from DFT-Library
      self.base_Class->asSet()-
      >closure(general).name->includes('SEQ')
else
      --SEQ stereotype can only be applied on any
      property whose type is specialized from SEQ
      from DFT-Library
      self.base_Property.type->asSet()-
      >closure(general).name->includes('SEQ')
endif
```

SPARE

Beschreibung: Ein `SPARE` ist eine Spezialisierung des Stereotyps `SPARE` für SPARE-Gates.
Abstrakt: Nein
Generalisierung: SPARE
Erweiterung: Class, Property
Constraints: [1]

```
if not self.base_Class->isEmpty() then
      --SPARE stereotype can only be applied on any
      class specialized from SPARE from DFT-Library
      self.base_Class->asSet()-
      >closure(general).name->includes('SPARE')
else
      --SPARE stereotype can only be applied on any
      property whose type is specialized from SPARE
      from DFT-Library
      self.base_Property.type->asSet()-
      >closure(general).name->includes('SPARE')
endif
```

FDEP

Beschreibung: Ein `FDEP` ist eine Spezialisierung des Stereotyps `FDEP` für FDEP-Gates.
Abstrakt: Nein
Generalisierung: FDEP
Erweiterung: Class, Property
Constraints: [1]

```
if not self.base_Class->isEmpty() then
      --FDEP stereotype can only be applied on any
      class specialized from FDEP from DFT-Library
      self.base_Class->asSet()-
      >closure(general).name->includes('FDEP')
else
      --FDEP stereotype can only be applied on any
      property whose type is specialized from FDEP
      from DFT-Library
      self.base_Property.type->asSet()-
      >closure(general).name->includes('FDEP')
```

```
endif
```

DynamicBE

Beschreibung: Ein `DynamicBE` ist eine Spezialisierung des Stereotyps `BasicEvent` für das dynamische Basisereignis.
Abstrakt: Nein
Generalisierung: BasicEvent
Erweiterung: Class, Property

UnrepairableBE

Beschreibung: Ein `UnrepairableBE` ist eine Spezialisierung des Stereotyps `DynamicBE` für das dynamische Basisereignis nicht reparierbarer Komponenten.
Abstrakt: Nein
Generalisierung: DynamicBE
Erweiterung: Class, Property
Constraints: [1]

```
if not self.base_Class->isEmpty() then
      --UnrepairableBE stereotype can only be applied
      on any class specialized from UnrepairableBE
      from DFT-Library
      self.base_Class->asSet()-
      >closure(general).name-
      >includes('UnrepairableBE')
else
      --UnrepairableBE stereotype can only be applied
      on any property whose type is specialized from
      UnrepairableBE from DFT-Library
      self.base_Property.type->asSet()-
      >closure(general).name-
      >includes('UnrepairableBE')
endif
```

RepairableBE

Beschreibung: Ein `RepairableBE` ist eine Spezialisierung des Stereotyps `DynamicBE` für das dynamische Basisereignis reparierbarer Komponenten.
Abstrakt: Nein
Generalisierung: DynamicBE
Erweiterung: Class, Property
Constraints: [1]

```
if not self.base_Class->isEmpty() then
      --RepairableBE stereotype can only be applied
      on any class spe-cialized from RepairableBE
      from DFT-Library
      self.base_Class->asSet()-
      >closure(general).name-
      >includes('RepairableBE')
else
      --RepairableBE stereotype can only be applied
      on any property whose type is specialized from
      RepairableBE from DFT-Library
      self.base_Property.type->asSet()-
      >closure(general).name-
      >includes('RepairableBE')
```

```
endif
```

LatentBE

Beschreibung: Ein `LatentBE` ist eine Spezialisierung des Stereotyps `DynamicBE` für das dynamische Basisereignis latenter, auf einem Inspektionsintervall beruhender Komponenten.
Abstrakt: Nein
Generalisierung: DynamicBE
Erweiterung: Class, Property
Constraints: [1]

```
if not self.base_Class->isEmpty() then
      --LatentBE stereotype can only be applied on
      any class special-ized from LatentBE from DFT-
      Library
      self.base_Class->asSet()-
      >closure(general).name-
      >includes('LatentBE')
else
      --LatentBE stereotype can only be applied on
      any property whose type is specialized from
      LatentBE from DFT-Library
      self.base_Property.type->asSet()-
      >closure(general).name-
      >includes('LatentBE')
endif
```

Distribution

Beschreibung: Eine `Distribution` ist eine Spezialisierung des Stereotyps `Situation` und erweitert zusätzlich die Metaklasse `DataType` für verschiedene Verteilungsfunktionen.
Abstrakt: Nein
Generalisierung: Situation
Erweiterung: DataType

A.9.12 SBBM

SBBM-Bibliothek

13849Function

Beschreibung: Ein `13849Function` spezialisiert `DysfunctionalEvent` und repräsentiert eine abstrakte Funktion oder Teilfunktion. Hierfür wird `probabilityPFHd` aus likelihood von `AbstractEvent` neu definiert.
Abstrakt: Ja
Generalisierung: DysfunctionalEvent
Applied Stereotype: Function

13849Block

Beschreibung: Ein `13849Block` spezialisiert `DysfunctionalEvent` und repräsentiert einen abstrakten Block in der SBBM. Hierfür wird `meanTimeFailureMTTFd` aus `likelihood` von `AbstractEvent` neu definiert. Zusätzlich definiert ein `13849Block` die Attribute `diagnosticCoverage` und `missionTime`. Über eine Assoziation mit einem

`MachineElement` und des Attributes `machineelement` lässt sich ein Maschinenelement zu einem Block assoziieren.
Abstrakt: Ja
Generalisierung: DysfunctionalEvent
Applied Stereotype: 13849Block

13849Event

Beschreibung: Ein `13849Event` spezialisiert `AbstractEvent` und repräsentiert ein abstraktes Event in der SBBM.
Abstrakt: Ja
Generalisierung: AbstractEvent
Applied Stereotype: 13849Event

TriggeringEvent

Beschreibung: Ein `TriggeringEvent` spezialisiert `13849Event` und repräsentiert das auslösende Ereignis einer Sicherheitsfunktion in der SBBM.
Abstrakt: Nein
Generalisierung: 13849Event
Applied Stereotype: Trigger

ActuatorEvent

Beschreibung: Ein `ActuatorEvent` spezialisiert `13849Event` und repräsentiert das Reaktionsereignis einer Sicherheitsfunktion in der SBBM.
Abstrakt: Nein
Generalisierung: 13849Event
Applied Stereotype: Actuator

SafetyFunction

Beschreibung: Eine `SafetyFunction` spezialisiert `Scenario` und `13849Function`, um Sicherheitsfunktionen in der SBBM darzustellen. Hierfür wird `probabilityPFHdResult` aus `probabilityPFHd` neu definiert. Zusätzlich definiert eine `SafetyFunction` das Attribut `performanceLevelRequired` mit dem Datentyp PL aus der ISO 1210 & ISO 13849-Bibliothek. Über Kompositionen wird ein `TriggeringEvent` und ein `ActuatorEvent` sowie mindestens ein Subsystem verbunden. Zusätzlich wird ein Constraint vergeben, um sicherzustellen, dass alle notwendigen Werte angegeben sind.
Abstrakt: Nein
Generalisierung: Scenario, 13849Function
Applied Stereotype: Function
Constraints: [1]

```
--SafetyFunction should have a performanceLevelRe-
quired, a triggeringEvent, an actuatorEvent and a
subsystem
self.performanceLevelRequired<>null and
self.actuatorEvent<>null and
self.triggeringEvent<>null and self.attributes-
>select(a | a.name.startsWith('subsystem'))-
>notEmpty()
```

Subsystem

Beschreibung: Ein `Subsystem` spezialisiert `13849Function` und `13849Block`, um Teilfunktionen bzw. Subsysteme in der SBBM darzustellen. Hierfür werden zusätzlich die Attribute `category` und `measureAgainstCCF` mit den Datentypen aus der ISO 12100 & ISO 13849-Bibliothek definiert. Über eine Komposition wird `Channel` mit bis zu zwei Channels verbunden. Im Falle eines gekapselten Subsystems wird kein Channel benötigt. In diesem Fall soll `Input-SB`, `Logic-SB` oder `Output-SB` verwendet werden, um über deren Constraints sicherzustellen, dass ein `Machineelement` verbunden ist. Zusätzlich wird ein Constraint vergeben, um sicherzustellen, dass die `category` angegeben ist.
Abstrakt: Nein
Generalisierung: 13849Function, 13849Block
Applied Stereotype: Function
Constraints: [1]

```
--Subsystem should have a category
self.category<>null
```

Input-SB

Beschreibung: Ein `Input-SB` spezialisiert `Subsystem`, um meist gekapselte Eingangskomponenten mit vom Hersteller angegebenen PFH_D und Kategorie in der SBBM darzustellen. Das Attribut `machineelement` mit dem Typ `MachineElement` wird durch `machineelement` mit dem Typ `E_Sensor_Input` neu definiert. Zusätzlich wird ein Constraint vergeben, um sicherzustellen, dass ein `machineelement` angegeben ist. Sollte es sich um ein gekapseltes Subsystem handeln, ist `Subsystem` zu verwenden.
Abstrakt: Nein
Generalisierung: Subsystem
Applied Stereotype: Input
Constraints: [1]

```
--Input-SB should have a machineelement
self.machineelement<>null
```

Logic-SB

Beschreibung: Ein `Logic-SB` spezialisiert `Subsystem`, um meist gekapselte Eingangskomponenten mit vom Hersteller angegebenen PFH_D und Kategorie in der SBBM darzustellen. Das Attribut `machineelement` mit dem Typ `MachineElement` wird durch `machineelement` mit dem Typ `E_Controller` neu definiert. Zusätzlich wird ein Constraint vergeben, um sicherzustellen, dass ein `machineelement` angegeben ist. Sollte es sich um ein gekapseltes Subsystem handeln, ist `Subsystem` zu verwenden.
Abstrakt: Nein
Generalisierung: Subsystem
Applied Stereotype: Logic
Constraints: [1]

```
--Logic-SB should have a machineelement
self.machineelement<>null
```

Output-SB

Beschreibung: Ein `Output-SB` spezialisiert `Subsystem`, um meist gekapselte Eingangskomponenten mit vom Hersteller angegebenen PFH_D und Kategorie in der SBBM darzustellen. Das Attribut `machineelement` mit dem Typ `MachineElement` wird durch

machineelement mit dem Typ E_Actor_Output neu definiert. Zusätzlich wird ein Constraint vergeben, um sicherzustellen, dass ein machineelement angegeben ist. Sollte es sich um ein gekapseltes Subsystem handeln, ist Subsystem zu verwenden.
Abstrakt: Nein
Generalisierung: Subsystem
Applied Stereotype: Output
Constraints: [1]

```
--Output-SB should have a machineelement
self.machineelement<>null
```

Channel

Beschreibung: Ein Channel dient der Definition von Kanälen in der SBBM. Jeder Channel besitzt eine Komposition zu mindestens einem BaseBlock. Es wird ein Constraint vergeben, um sicherzustellen, dass ein BaseBlock angegeben ist.
Abstrakt: Nein
Applied Stereotype: Channel
Constraints: [1]

```
--Channel should have a block
self.attributes-
>select(a | a.name.startsWith('block'))->notEmpty()
```

BaseBlock

Beschreibung: Ein BaseBlock dient der Definition von Blöcken in der SBBM und spezialisiert hierfür 13849Block. Jeder BaseBlock kann eine Komposition zu mehreren BaseElement besitzen. Falls dies nicht der Fall ist, soll Input-B, Logic-B oder Output-B verwendet werden, um über deren Constraints sicherzustellen, dass ein Machineelement verbunden ist.
Abstrakt: Nein
Generalisierung: 13849Block
Applied Stereotype: 13849Block

Input-B

Beschreibung: Ein Input-B spezialisiert BaseBlock, um Eingangs-Blöcke in der SBBM darzustellen. Das Attribut machineelement mit dem Typ MachineElement wird durch machineelement mit dem Typ E_Sensor_Input neu definiert. Zusätzlich wird ein Constraint vergeben, um sicherzustellen, dass ein machineelement angegeben ist. Sollte sich der SBBM-Block weiter in SBBM-Elemente gliedern, ist BaseBlock zu verwenden.
Abstrakt: Nein
Generalisierung: BaseBlock
Applied Stereotype: Input
Constraints: [1]

```
--Input-B should have a machineelement
self.machineelement<>null
```

Testequipment-B

Beschreibung: Ein Testequipment-B spezialisiert Input-B, um Testequipment-Blöcke in der SBBM darzustellen.
Abstrakt: Nein
Generalisierung: Input-B

Applied Stereotype: Testequipment

Logic-B

Beschreibung: Ein `Logic-B` spezialisiert `BaseBlock`, um Logik-Blöcke in der SBBM darzustellen. Das Attribut `machineelement` mit dem Typ `MachineElement` wird durch `machineelement` mit dem Typ `Controller` neu definiert. Zusätzlich wird ein Constraint vergeben, um sicherzustellen, dass ein `machineelement` angegeben ist. Sollte sich der SBBM-Block weiter in SBBM-Elemente gliedern, ist `BaseBlock` zu verwenden.
Abstrakt: Nein
Generalisierung: BaseBlock
Applied Stereotype: Logic
Constraints: [1]

```
--Logic-B should have a machineelement
self.machineelement<>null
```

Output-B

Beschreibung: Ein `Output-B` spezialisiert `BaseBlock`, um Ausgangs-Blöcke in der SBBM darzustellen. Das Attribut `machineelement` mit dem Typ `MachineElement` wird durch `machineelement` mit dem Typ `E_Actor_Output` neu definiert. Zusätzlich wird ein Constraint vergeben, um sicherzustellen, dass ein `machineelement` angegeben ist. Sollte sich der SBBM-Block weiter in SBBM-Elemente gliedern, ist `BaseBlock` zu verwenden.
Abstrakt: Nein
Generalisierung: BaseBlock
Applied Stereotype: Output
Constraints: [1]

```
--Output-B should have a machineelement
self.machineelement<>null
```

OutputTest-B

Beschreibung: Ein `OutputTest-B` spezialisiert `Output-B`, um Ausgangs-Blöcke von Testequipment- in der SBBM darzustellen.
Abstrakt: Nein
Generalisierung: Output-B
Applied Stereotype: OutputTest

BaseElement

Beschreibung: Ein `BaseElement` dient der Definition von Elementen, für die zweite Ebene von Blöcken, in der SBBM und spezialisiert hierfür `13849Block`. Jedes `BaseElement` kann eine Komposition zu mehreren `BaseElement` besitzen. Zusätzlich wird ein Constraint vergeben, um sicherzustellen, dass ein `machineelement` angegeben ist.
Abstrakt: Nein
Generalisierung: 13849Block
Constraints: [1]

```
--Element should have a machineelement
self.machineelement<>null
```

Applied Stereotype: 13849Block

Input-E

Beschreibung: Ein `Input-E` spezialisiert `BaseBlock`, um Eingangs-Elemente in der SBBM darzustellen. Das Attribut `machineelement` mit dem Typ `MachineElement` wird durch `machineelement` mit dem Typ `E_Sensor_Input` neu definiert.
Abstrakt: Nein
Generalisierung: BaseElement
Applied Stereotype: Input

Logic-E

Beschreibung: Ein `Logic-E` spezialisiert `BaseElement`, um Logik-Elemente in der SBBM darzustellen. Das Attribut `machineelement` mit dem Typ `MachineElement` wird durch `machineelement` mit dem Typ `Controller` neu definiert.
Abstrakt: Nein
Generalisierung: BaseElement
Applied Stereotype: Logic

Output-E

Beschreibung: Ein `Output-E` spezialisiert `BaseElement`, um Ausgangs-Elemente in der SBBM darzustellen. Das Attribut `machineelement` mit dem Typ `MachineElement` wird durch machineelement mit dem Typ `E_Actor_Output` neu definiert.
Abstrakt: Nein
Generalisierung: BaseElement
Applied Stereotype: Output

SBBM-Profil

Function

Beschreibung: Eine `Function` ist eine Spezialisierung des Stereotyps `Situation`, um den Stereotyp für Funktionen und Teilfunktionen darzustellen.
Abstrakt: Nein
Generalisierung: Situation
Erweiterung: Class
Constraints: [1]

```
--Function stereotype can only be applied on any
class specialized from SafetyFunction or 13849Func-
tion from SBBM-Library
self.base_Class->asSet()->closure(general).name-
>includes('SafetyFunction') or
self.base_Class->asSet()->closure(general).name-
>includes('13849Function')
```

13849Block

Beschreibung: Ein `13849Block` ist eine Spezialisierung des Stereotyps `Situation` und eine Erweiterung der Metaklasse `Property`, um den Stereotyp für alle Blöcke in einer Sicherheitsfunktion darzustellen.
Abstrakt: Nein
Generalisierung: Situation
Erweiterung: Property

13849Event

Beschreibung: Ein `13849Event` ist eine Spezialisierung des Stereotyps `Situation` und eine Erweiterung der Metaklasse `Property`, um den Stereotyp für alle SBBM-Events dazustellen.
Abstrakt: Nein
Generalisierung: Situation
Erweiterung: Property

Trigger

Beschreibung: Ein `Trigger` ist eine Spezialisierung des Stereotyps `13849Event`, um den Stereotyp für alle auslösenden Ereignisse dazustellen.
Abstrakt: Nein
Generalisierung: Situation
Erweiterung: Class, Property
Constraints: [1]

```
if not self.base_Class->isEmpty() then
      --Trigger stereotype can only be applied on any
      class specialized from TriggerEvent from SBBM-
      Library
      self.base_Class->asSet()-
      >closure(general).name-
      >includes('TriggerEvent')
else
      --Trigger stereotype can only be applied on any
      property whose type is specialized from Trig-
      gerEvent from SBBM-Library
      self.base_Property.type->asSet()-
      >closure(general).name-
      >includes('TriggerEvent')
endif
```

Actuator

Beschreibung: Ein `Actuator` ist eine Spezialisierung des Stereotyps `13849Event`, um den Stereotyp für alle Reaktionsereignisse dazustellen.
Abstrakt: Nein
Generalisierung: Situation
Erweiterung: Class, Property
Constraints: [1]

```
if not self.base_Class->isEmpty() then
      --Actuator stereotype can only be applied on
      any class special-ized from ActuatorEvent from
      SBBM-Library
      self.base_Class->asSet()-
      >closure(general).name-
      >includes('ActuatorEvent')
else
      --Actuator stereotype can only be applied on
      any property whose type is specialized from Ac-
      tuatorEvent from SBBM-Library
      self.base_Property.type->asSet()-
      >closure(general).name-
      >includes('ActuatorEvent')
endif
```

Subsystem

Beschreibung: Eine `Subsystem` ist eine Spezialisierung des Stereotyps `Situation`, um den Stereotyp einer Teilfunktion bzw. eines Subsystems darzustellen.
Abstrakt: Nein
Generalisierung: Situation
Erweiterung: Class, Property
Constraints: [1]

```
if not self.base_Class->isEmpty() then
      --Subsystem stereotype can only be applied on
      any class specialized from Subsystem from SBBM-
      Library
      self.base_Class->asSet()-
      >closure(general).name->includes('Subsystem')
else
      --Subsystem stereotype can only be applied on
      any property whose type is specialized from
      Subsystem from SBBM-Library
      self.base_Property.type->asSet()-
      >closure(general).name->includes('Subsystem')
endif
```

Input

Beschreibung: Ein `Input` ist eine Spezialisierung des Stereotyps `13849Block`, um den Stereotyp eines Eingangs-Blocks der SBBM darzustellen.
Abstrakt: Nein
Generalisierung: 13849Block
Erweiterung: Class, Property
Constraints: [1]

```
if not self.base_Class->isEmpty() then
      --Input stereotype can only be applied on any
      class specialized from Input-SB, Input-B or In-
      put-E from SBBM-Library
      self.base_Class->asSet()-
      >closure(general).name->includes('Input-SB') or
      self.base_Class->asSet()-
      >closure(general).name->includes('Input-B') or
      self.base_Class->asSet()-
      >closure(general).name->includes('Input-E')
else
      --Input stereotype can only be applied on any
      property whose type is specialized from Input-
      SB, Input-B or Input-E from SBBM-Library
      self.base_Property.type->asSet()-
      >closure(general).name->includes('Input-SB') or
      self.base_Property.type->asSet()-
      >closure(general).name->includes('Input-B') or
      self.base_Property.type->asSet()-
      >closure(general).name->includes('Input-E')
endif
```

Testequipment

Beschreibung: Ein `Testequipment` ist eine Spezialisierung des Stereotyps `13849Block`, um den Stereotyp eines Testequipment-Eingangs-Blocks der SBBM darzustellen.
Abstrakt: Nein

Generalisierung: 13849Block
Erweiterung: Class, Property
Constraints: [1]

```
if not self.base_Class->isEmpty() then
      --Testequipment stereotype can only be applied
      on any class specialized from Testequipment-B
      from SBBM-Library
      self.base_Class->asSet()-
      >closure(general).name-
      >includes('Testequipment-B')
else
      --Testequipment stereotype can only be applied
      on any property whose type is specialized from
      Testequipment-B from SBBM-Library
      self.base_Property.type->asSet()-
      >closure(general).name-
      >includes('Testequipment-B')
endif
```

Logic

Beschreibung: Eine `Logic` ist eine Spezialisierung des Stereotyps `13849Block` und Erweiterung der Metaklasse `StateMachine`, um den Stereotyp eines Logik-Blocks der SBBM darzustellen.
Abstrakt: Nein
Generalisierung: 13849Block
Erweiterung: StateMachine, Class, Property
Constraints: [1]

```
if not self.base_Class->isEmpty() then
      --Logic stereotype can only be applied on any
      class specialized from Logic-SB, Logic-B or
      Logic-E from SBBM-Library
      self.base_Class->asSet()-
      >closure(general).name->includes('Logic-SB') or
      self.base_Class->asSet()-
      >closure(general).name->includes('Logic-B') or
      self.base_Class->asSet()-
      >closure(general).name->includes('Logic-E')
else
      --Logic stereotype can only be applied on any
      property whose type is specialized from Logic-
      SB, Logic-B or Logic-E from SBBM-Library
      self.base_Property.type->asSet()-
      >closure(general).name->includes('Logic-SB') or
      self.base_Property.type->asSet()-
      >closure(general).name->includes('Logic-B') or
      self.base_Property.type->asSet()-
      >closure(general).name->includes('Logic-E')
endif
```

Output

Beschreibung: Ein `Output` ist eine Spezialisierung des Stereotyps `13849Block`, um den Stereotyp eines Ausgangs-Blocks der SBBM darzustellen.
Abstrakt: Nein
Generalisierung: 13849Block

Erweiterung: Class, Property

Constraints: [1]

```
if not self.base_Class->isEmpty() then
      --Output stereotype can only be applied on any
      class specialized from Output-SB, Output-B or
      Output-E from SBBM-Library
      self.base_Class->asSet()-
      >closure(general).name->includes('Output-SB')or
      self.base_Class->asSet()-
      >closure(general).name->includes('Output-B') or
      self.base_Class->asSet()-
      >closure(general).name->includes('Output-E')
else
      --Output stereotype can only be applied on any
      property whose type is specialized from Output-
      SB, Output-B or Output-E from SBBM-Library
      self.base_Property.type->asSet()-
      >closure(general).name->includes('Output-SB')or
      self.base_Property.type->asSet()-
      >closure(general).name->includes('Output-B') or
      self.base_Property.type->asSet()-
      >closure(general).name->includes('Output-E')
endif
```

OutputTest

Beschreibung: Ein `OutputTest` ist eine Spezialisierung des Stereotyps `13849Block`, um den Stereotyp eines Testequipment-Ausgangs-Blocks der SBBM darzustellen.

Abstrakt: Nein

Generalisierung: 13849Block

Erweiterung: Class, Property

Constraints: [1]

```
if not self.base_Class->isEmpty() then
      --OutputTest stereotype can only be applied on
      any class specialized from OutputTest-B from
      SBBM-Library
      self.base_Class->asSet()-
      >closure(general).name-
      >includes('OutputTest-B')
else
      --OutputTest stereotype can only be applied on
      any property whose type is specialized from
      OutputTest-B from SBBM-Library
      self.base_Property.type->asSet()-
      >closure(general).name-
      >includes('OutputTest-B')
endif
```

Channel

Beschreibung: Ein `OutputTest` ist eine Spezialisierung des Stereotyps `13849Block`, um den Stereotyp eines Testequipment-Ausgangs-Blocks der SBBM darzustellen.

Abstrakt: Nein

Erweiterung: Class, Property

Constraints: [1]

```
if not self.base_Class->isEmpty() then
```

```
        --Channel stereotype can only be applied on any
        class specialized from Channel-B from SBBM-Li-
        brary
        self.base_Class->asSet()-
        >closure(general).name->includes('Channel')
else
        --Channel stereotype can only be applied on any
        property whose type is specialized from Chan-
        nel-B from SBBM-Library
        self.base_Property.type-
        >asSet()->closure(general).name-
        >includes('Channel')
endif
```

Connector

Beschreibung: Ein `Connector` ist eine Erweiterung der Metaklasse `Dependency`, um den Stereotyp für die Verbindung von SBBM-Blocks darzustellen.
Abstrakt: Nein
Erweiterung: Dependency
Constraints: [1]

```
--supplier and client of Connector must be a
13849Block
13849Block.allInstances().base_Class-
>includesAll(self.base_Dependency.client) and
13849Block.allInstances().base_Class-
>includesAll(self.base_Dependency.supplier)
```

MonitorTestConnector

Beschreibung: Ein `MonitorTestConnector` ist eine Spezialisierung des Stereotyps `Connector`, um den Stereotyp für Verbindungen zum Testkanal darzustellen.
Abstrakt: Nein
Generalisierung: Connector
Erweiterung: Dependency

A.10 Modulbeschreibung der Transformatoren

A.10.1 SBBM in SISTEMA

Im Folgenden finden sich die Struktur und der Code für die verschiedenen implementierten Module des Transformators für die SBBM zu SISTEMA.

Analysemodellexport und Transformation

Es wurden sieben Hauptfunktionen für den Export und die Transformation entworfen, wobei die erste Funktion den XMI-Export und einen Import als zeilenbasierte Textdatei in LabVIEW, gleich Abbildung 4.8 für DFTs, implementiert. Die Transformation findet in den restlichen sechs, in Abbildung A.23 dargestellten, VIs statt.

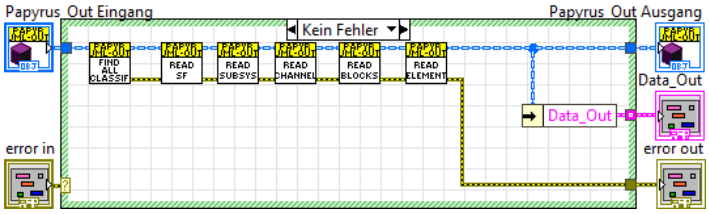


Abbildung A.23: XMI-Input-Transformation aller Elemente - G-Code

Zunächst werden, wie zuvor beim DFT-Transformator, die Klassifikator-Namen mit zugehörigen XMI-IDs und parallel die `Violated`-Abhängigkeiten eingelesen, um beim erneuten Import in Papyrus abhängige Anforderungen zu markieren. Abbildung A.24 illustriert diese Implementierung.

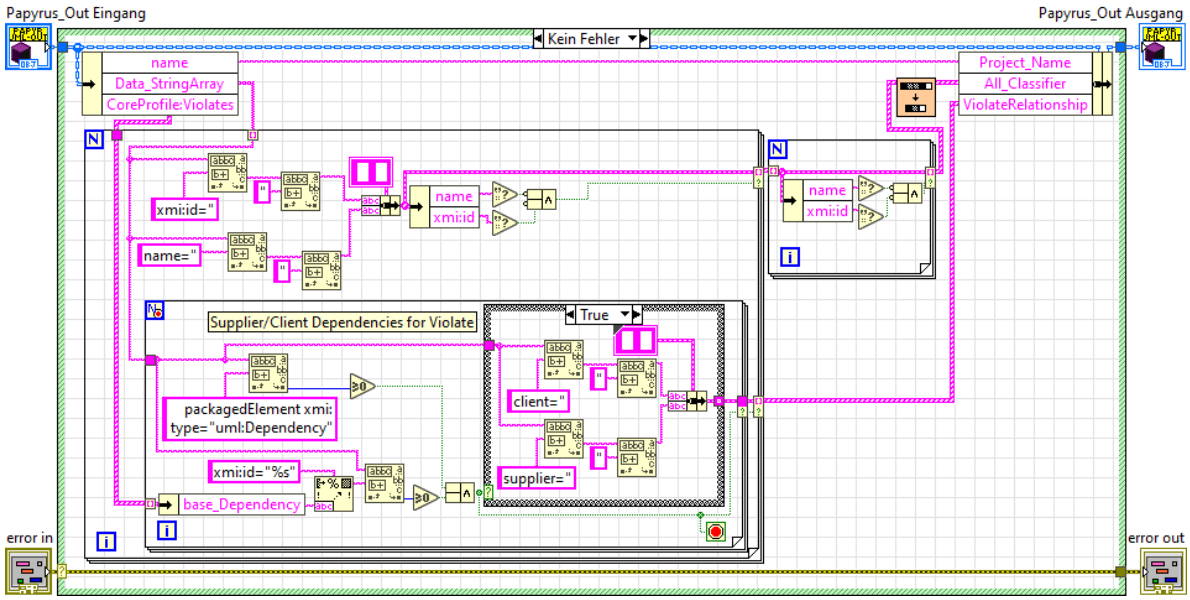


Abbildung A.24: XMI-Input-Transformation Violated-Abhängigkeiten - G-Code

Zunächst werden nachfolgend die Sicherheitsfunktionen, wie in Abbildung A.25 gezeigt, mit auslösendem Ereignis, Reaktionsereignis, PL_r und Subsystem-IDs ausgelesen und in ein für den SISTEMA-Input vorbereitendes Format gebracht.

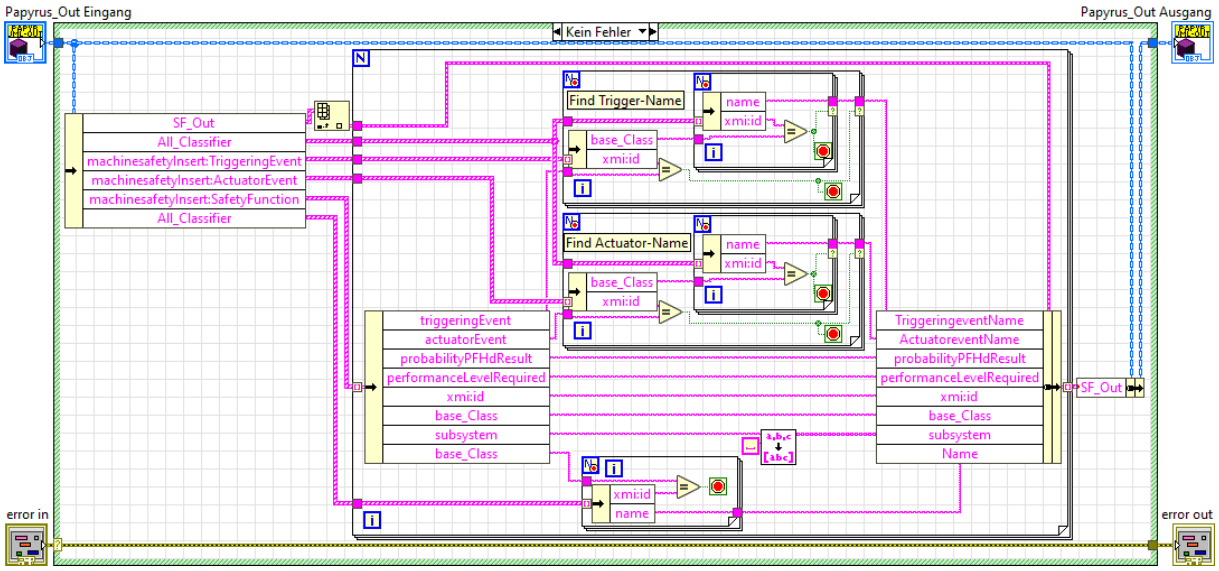


Abbildung A.25: XMI-Input-Transformation Funktionen bestimmen und lesen - G-Code

Ähnlich der Implementierung für Sicherheitsfunktionen in Abbildung A.25 bildet sich die Umsetzung für Subsysteme, Channel, Blöcke und Elemente ab, wobei für Subsysteme, Blöcke und Elemente zuvor, wie in Abbildung A.26 gezeigt, eine Typunterscheidung getroffen werden muss, je nachdem, ob es sich um einen Input, eine Logic oder ein Output handelt.

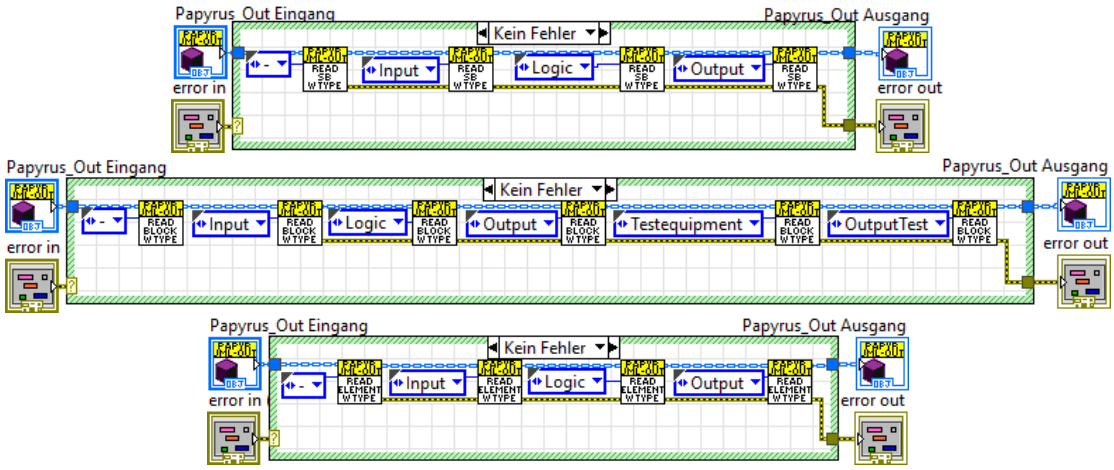


Abbildung A.26: Typen von SISTEMA-Blöcken bestimmen & auslesen - G-Code Subsystemen (oben), Blöcken (mittig) & Elemente (unten)

Wie Abbildung A.26 zeigt, handelt es sich jeweils um ein typisiertes VI, das für jeden Anwendungsfall eines Subsystems, eines Blocks oder eines Elements ausgeführt wird.

Analyseinputformat

Um die verschiedenen Datensätze an SISTEMA zu übertragen, wurde das in Abbildung A.27 umgesetzte VI mit dem SubVI in Abbildung A.28 entworfen. Zunächst setzt das VI die Datensätze nacheinander zusammen und bringt sie mit den Felder-Definitionen zusammen.

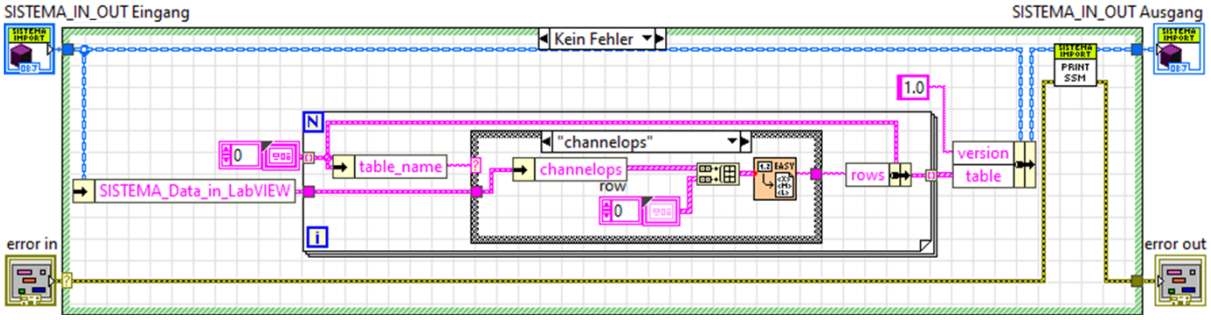


Abbildung A.27: SISTEMA-Input SSM-Dateizusammensetzung - G-Code

Auf Basis des fertigen Formats wandelt das VI in Abbildung A.28 das erstellte Cluster in einen XML-String, erstellt den Pfad für die *.ssm-Datei, fügt die Deklaration der XML-Spezifikation an und erstellt die Datei.

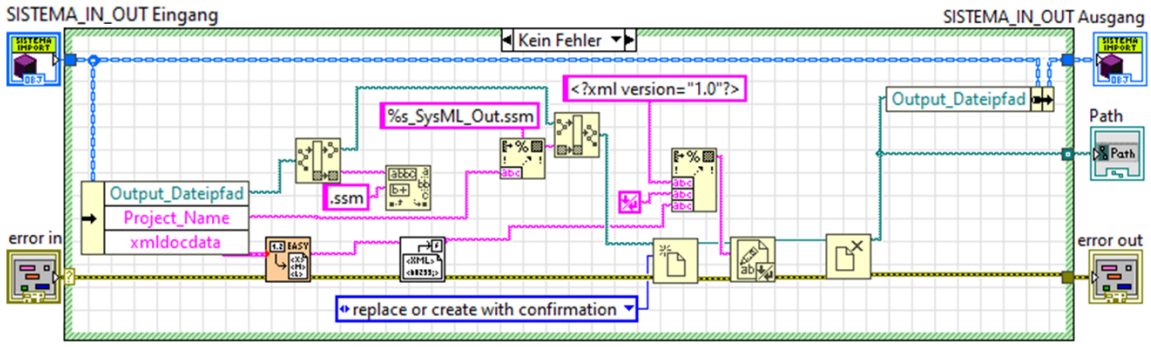


Abbildung A.28: SISTEMA-Input SSM-Datei erstellen - G-Code

Analyseausführung

Die Umsetzung zur Analyseausführung besteht lediglich aus folgender in Abbildung A.29 dargestellten Hauptfunktion.

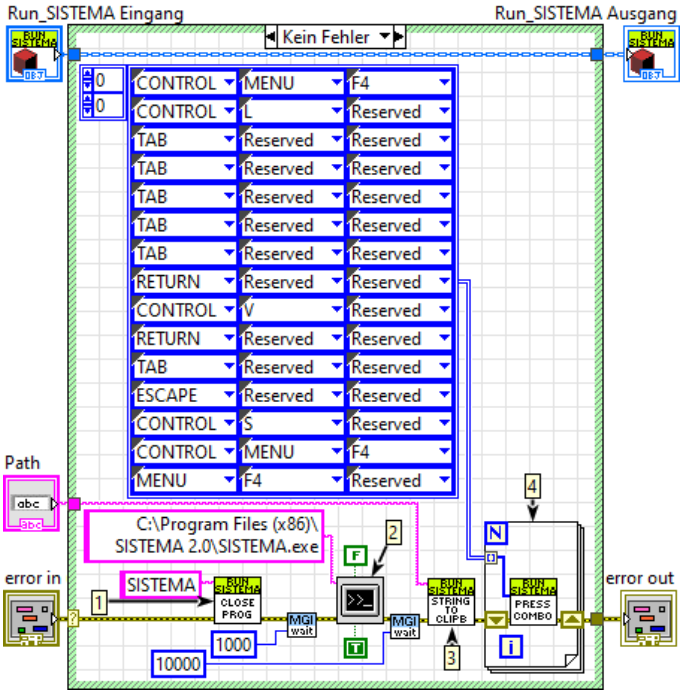


Abbildung A.29: SISTEMA-Input SSM-Datei erstellen - G-Code

Das VI nutzt hauptsächlich die Call Library Function Node von LabVIEW, um über Windows-Befehle SISTEMA auszuführen. Das erste SubVI (1) geht sicher, dass SISTEMA geschlossen ist (1). Nach einer Wartezeit von einer Sekunde wird SISTEMA erneut gestartet (2) und nach zehn Sekunden der Pfad zum zuvor erstellen SISTEMA-Projekt in die Zwischenablage kopiert (4). Zuletzt steuern Tastaturbefehle SISTEMA, indem die Tastaturkombinationen für alle Projekte schließen, Projekt öffnen, auf Pfadeingabe navigieren, Aus Zwischenablage einfügen, Projekt öffnen Fenster schließen, Projekt speichern, alle Projekte schließen und SISTEMA schließen nacheinander ausgeführt werden.

Analyseergebnisverarbeitung

Nachdem die *.ssm-Datei in SISTEMA geöffnet und gespeichert wurde, stehen die Ergebnisdaten zur Verfügung. Diese können über denselben Mechanismus ausgelesen werden, wie die Daten zuvor übertragen wurden. Dies hat den Vorteil, dass nicht nur die Ergebnisse, sondern auch aktualisierte Daten aller Bestandteile zur Verfügung stehen, um gegen die Eingabedatensätze zu überprüfen. Das Modul besteht aus dem Einlesen der Ergebnisdaten im VI in Abbildung A.30 und der Interpretation der Ergebnisdaten sowie Umwandlung ins Zwischenformat in Abbildung A.31.

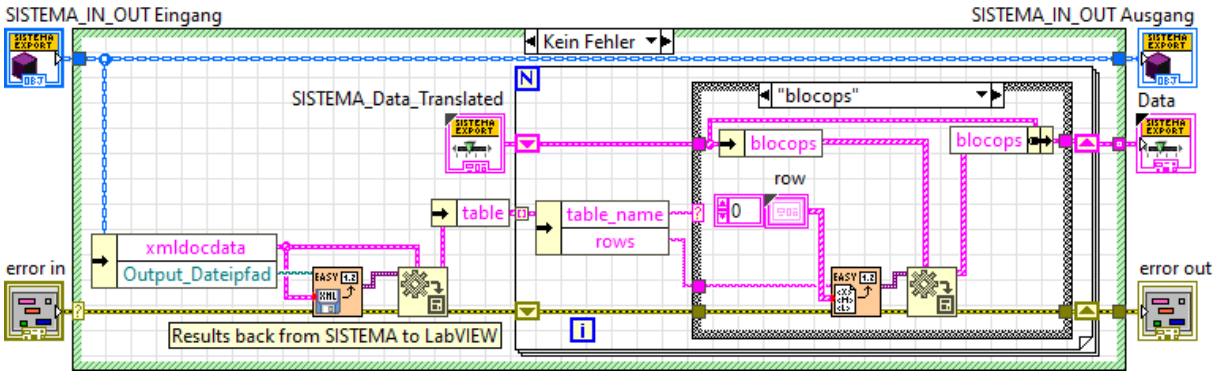


Abbildung A.30: SISTEMA-Output SSM-Datei lesen - G-Code

Die vier VI-Funktionen in Abbildung A.31 ordnen die Ergebnisdaten den hinterlegten Elementen aus Papyrus zu, angefangen mit den Elementen, über Blöcke und Subsysteme bis zu den Sicherheitsfunktionen. Die Implementierung für Blöcke findet sich in Abbildung A.32.

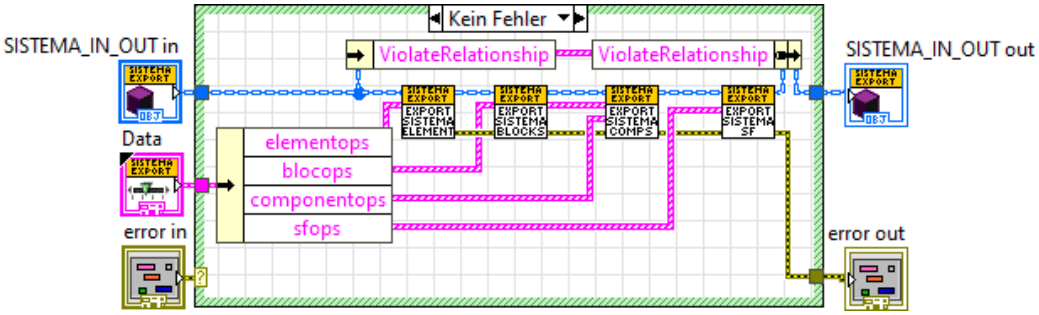


Abbildung A.31: SISTEMA-Output-Transformation - G-Code

Innerhalb der Implementierung zur Rücktransformation der Blöcke in Abbildung A.32 werden im oberen Teil die Ergebnisse zurückgelesen und das Ergebnis nach den Vorgaben der SBBM interpretiert. Zusätzlich werden die zurückerhaltenden Werte mit den zuvor in SISTEMA eingegebenen Werten verglichen. Sofern Abweichungen entstehen, wird dies als Benutzerinformation angezeigt und übernommen. Der untere Teil des VIs stellt für Subsysteme Informationen bereit, die eine Neuberechnung von Subsystemen zum Vergleich mit den von SISTEMA berechneten Ergebnisdaten ermöglicht.

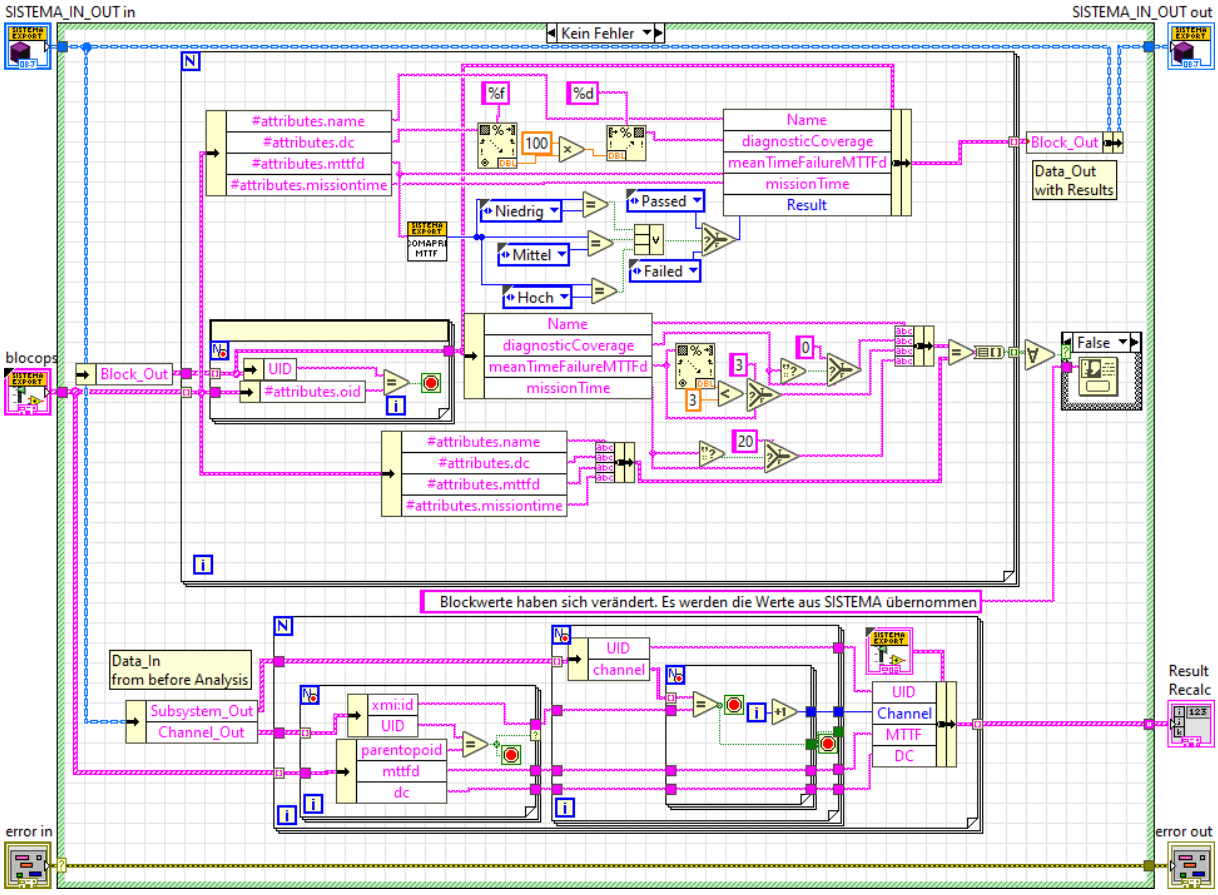


Abbildung A.32: SISTEMA-Output-Transformation Blöcke - G-Code

Die Kontrolle und teilweise Neuberechnung innerhalb von LabVIEW während der Rücktransformation ist neben einer hohen Informationssicherheit auch darin begründet, wie SISTEMA die Ergebnisdaten speichert. Innerhalb von SISTEMA können Zwischenergebnisse von Blöcken und Subsystemen zwar eingesehen werden, doch hat sich während der Entwicklung des ersten Entwurfs des Transformators ebenso herausgestellt, dass diese nicht oder nur teilweise ins XML-Format mitübernommen werden. Um diese aber eindeutig und lückenlos in die gemeinsame Datenbasis in Papyrus zu übernehmen, wurde diese Ergebnisdatenvalidierung eingeführt.

Analyseergebnisreimport

Das Analyseergebnisreimport-Modul besteht aus den sieben in Abbildung A.33 dargestellten Schritten. Der erste und letzte Schritt umfasst hier wieder das Einlesen und Schreiben der aktuellen *.uml-Datei.

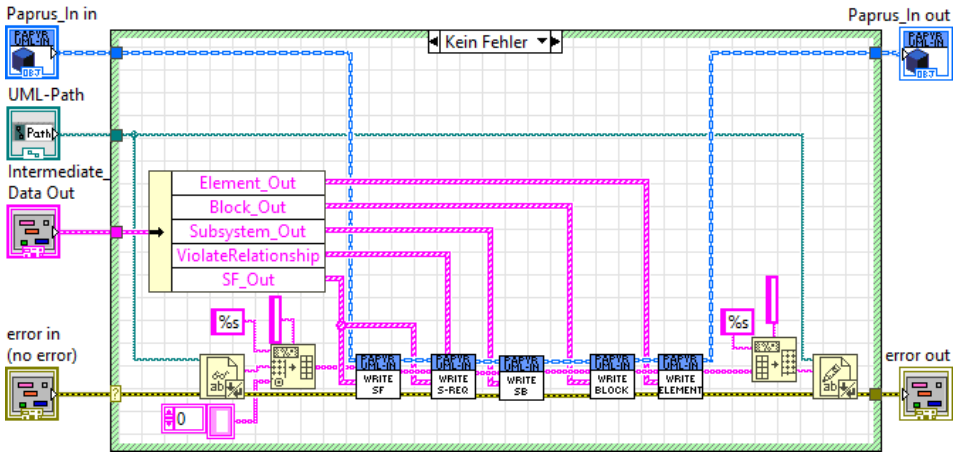


Abbildung A.33: Ergebnisdaten-Papyrus-Import SBBM - G-Code

Im zweiten bis sechsten Schritt werden in den entsprechenden VIs die Ergebnisdaten in den Stereotypen `FunctionalSafetyRequirement`, `SafetyFunction`, `Subsystem`, `Block` und `Element` und untergeordneten Stereotypen überschrieben. Die Implementierung zu `SafetyFunction` findet sich in Abbildung A.34.

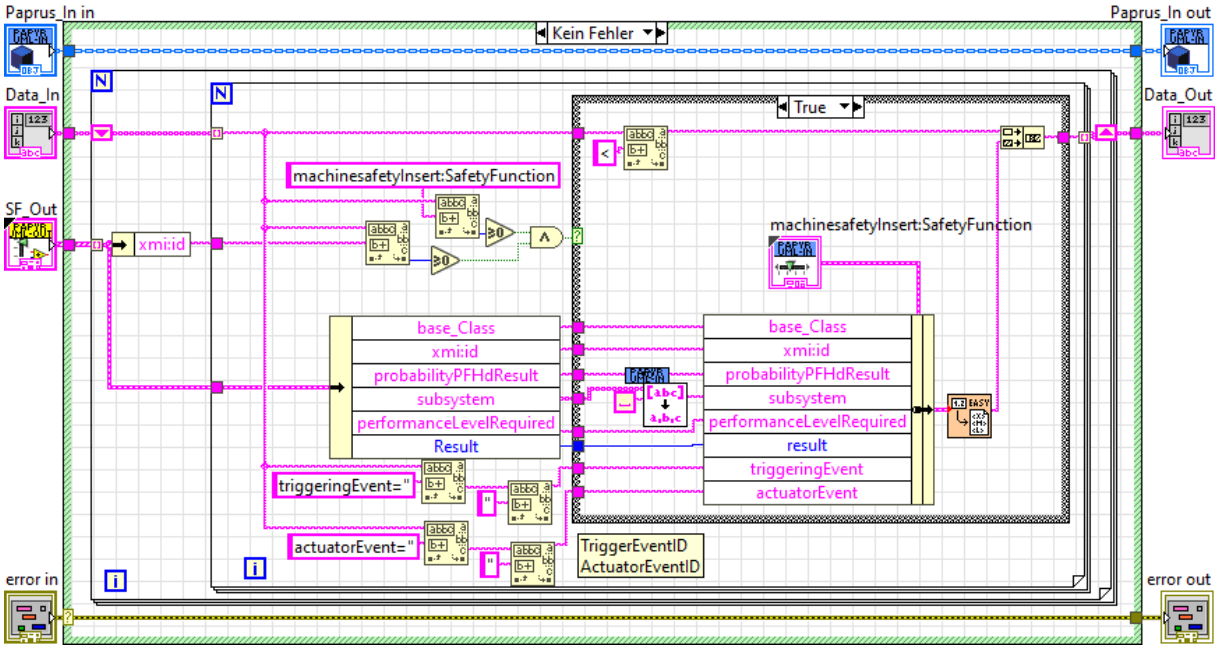


Abbildung A.34: Ergebnisdaten-Papyrus-SafetyFunction-Parser - G-Code

A.10.2 Risikographen in LabVIEW

Im Folgenden finden sich die Struktur und der Code für die verschiedenen implementierten Module des Transformators für Risikoitems von Risikographen in LabVIEW.

Analysemodellexport und Transformation

Es wurden drei Hauptfunktionen für den Export und die Transformation entworfen, wobei die erste Funktion den XMI-Export und einen Import als zeilenbasierte Textdatei in LabVIEW, gleich Abbildung 4.8 für DFTs, implementiert. Die zweite Funktion ist in Abbildung A.35 und die dritte in Abbildung A.37 dargestellt. Zunächst liest Abbildung A.35 wie bereits bei DFTs und SBBM alle Klassifikatoren ein, um die Namen zu den XMI-IDs zu erhalten. Aufgrund der Umsetzung ohne Anwendungsprofil werden die Daten aus dem zeilenbasierten String-Array gelesen. Hierfür werden für eine bessere Performance in der zweiten Funktion in Abbildung A.35 ungenutzte Daten gelöscht. Die dritte bis sechste Funktion liest dann basierend auf den Stereotypen `RisikoItem`, `HazardousSituation`, `HazardousEvent` und `Situation` die Daten ein.

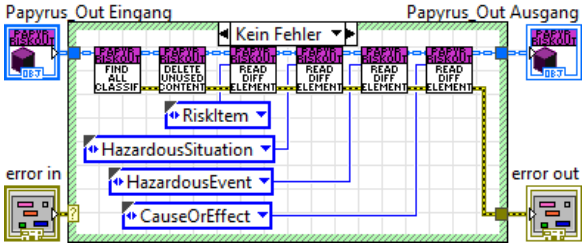


Abbildung A.35: XMI-Input-Transformation Risikodaten lesen - G-Code

Abbildung A.36 zeigt diese Funktion genauer. Für jedes Element wird basierend auf dem Typ die Stelle im String-Array gesucht, ausgelesen und mit allen für den Analyseergebnisreimport relevanten Daten in einem für die Weiterverarbeitung definierten Zwischenformat abgelegt.

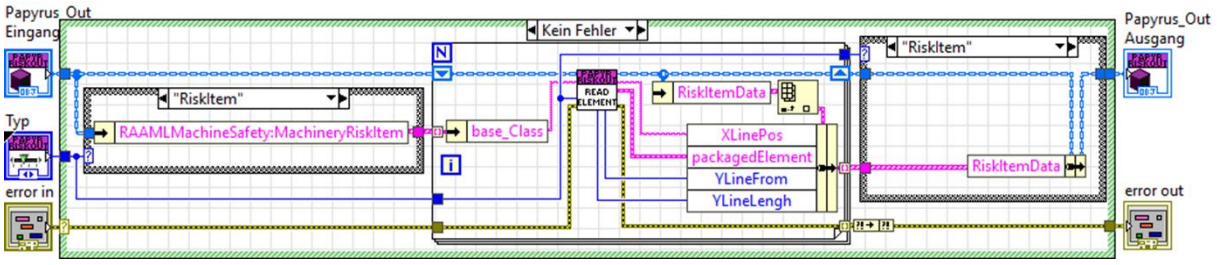


Abbildung A.36: XMI-Input-Transformation Risikoelement lesen - G-Code

Nachdem die Daten in diesem Zwischenformat vorliegen, kann der Risikograph für jedes Risikoitem korreliert werden. In Abbildung A.37 zeigt sich hierfür die erste Ebene. Da ein Risikoitem zunächst aus einer Gefährdungssituation und einem Effekt besteht, werden diese mit den

entsprechenden Werten für `avoidance` und `severity` ausgelesen. Innerhalb einer Gefährdungssituation schachteln sich zusätzlich die Gefährdungsereignisse und Ursachen, die jeweils die Werte für `frquencyOfExposure` und `occurence` liefern.

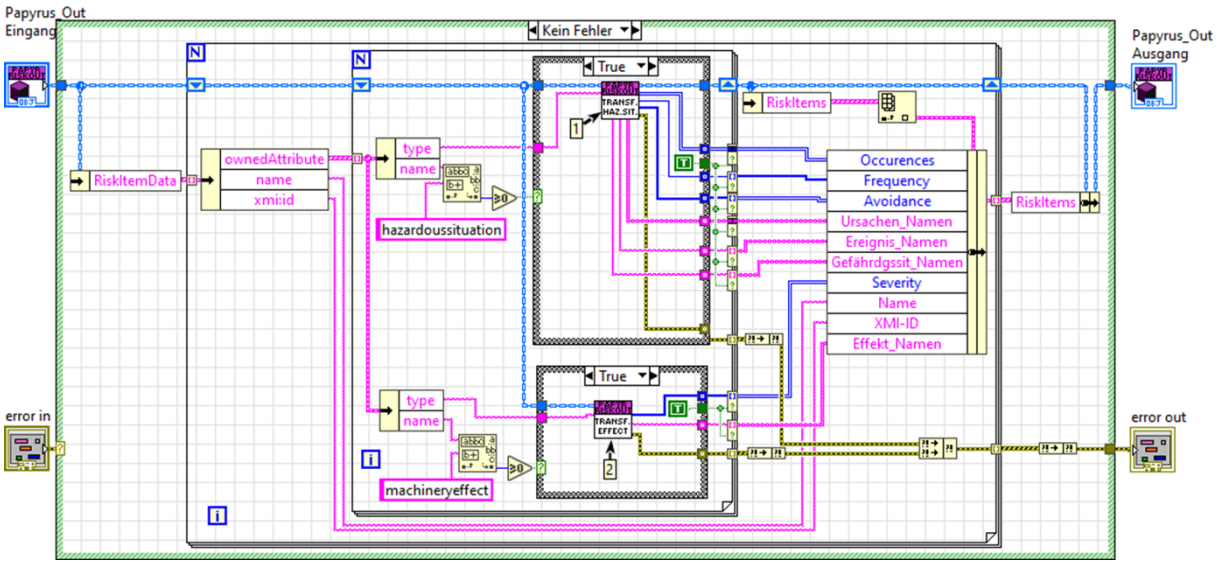


Abbildung A.37: XMI-Input-Transformation Risikoitem Korrelation - G-Code

Als zusätzliche Information finden sich ebenso die Namen für Gefährdungssituationen, Ursachen, Ereignisse und Effekte, die jedoch nicht weiterverwendet werden. Dies liefert allerdings die Möglichkeit einer weiteren Verarbeitung, beispielsweise zur Anbindung eines externen Werkzeugs, wie WEKA-Software.

Analyseausführung

Die Analyseausführung für die Risikobeurteilung durch Risikographen ist bereits vollständig in Abschnitt 4.2.4 beschrieben.

Analyseergebnisreimport

Das Analyseergebnisreimport-Modul besteht aus den vier in Abbildung A.33 dargestellten Schritten. Der zweite und vierte Schritt umfasst hier wieder das Einlesen und Schreiben der aktuellen *.uml-Datei.

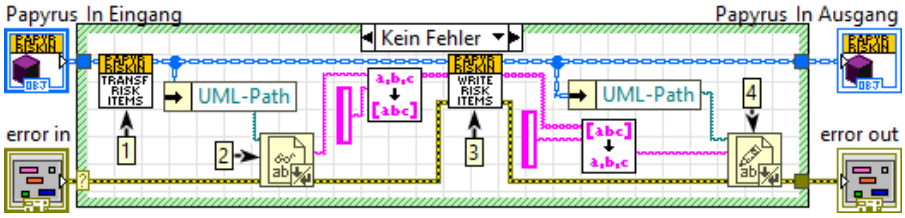


Abbildung A.38: Ergebnisdaten-Papyrus-Import Risikograph - G-Code

Da lediglich das Risikoitem aktualisiert werden muss, umfasst der erste Schritt das Ersetzen der Werte für die Attribute `riskIndex`, `previousRiskIndex` und `PLr` im Zwischenformat,

dargestellt in Abbildung A.39, und der dritte Schritt das Einfügen ins zeilenbasierte String-Array, dargestellt in Abbildung A.40.

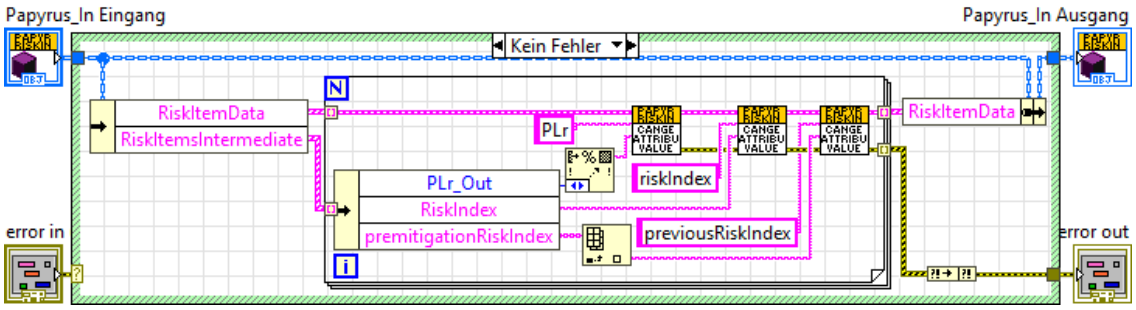


Abbildung A.39: Ergebnisdaten-Papyrus-Import Attribute in Zwischenformat - G-Code

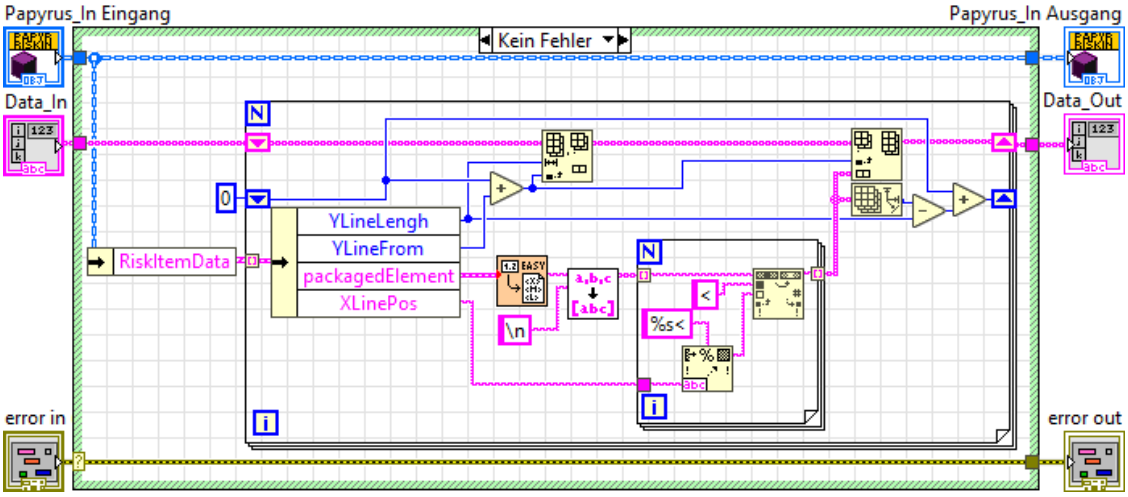


Abbildung A.40: Ergebnisdaten-Papyrus-Risikoitem-Parser - G-Code

Der in Abbildung A.40 dargestellte G-Code nutzt hierfür die bei der Transformation zusätzlich im Zwischenformat abgelegten Werte zur Position und Länge des Elements. Dies ermöglicht es, lediglich die entsprechenden Zeilen zu ersetzen.

A.11 Modelle der Evaluation

A.11.1 Weitere Risikoitems

Risikoitem Schneiden

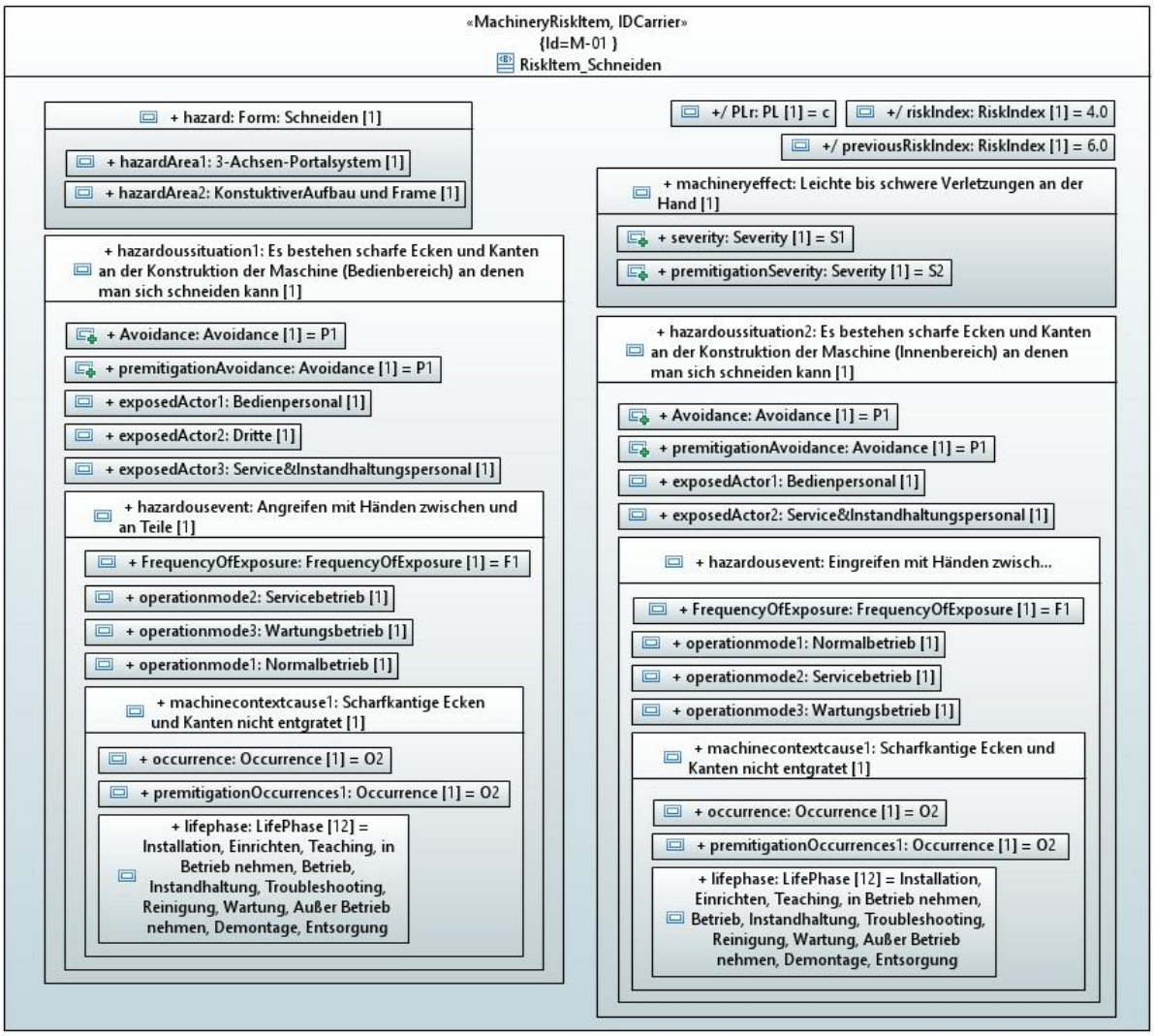


Abbildung A.41: Risikoitem Schneiden

Risikoitem statische Aufladung

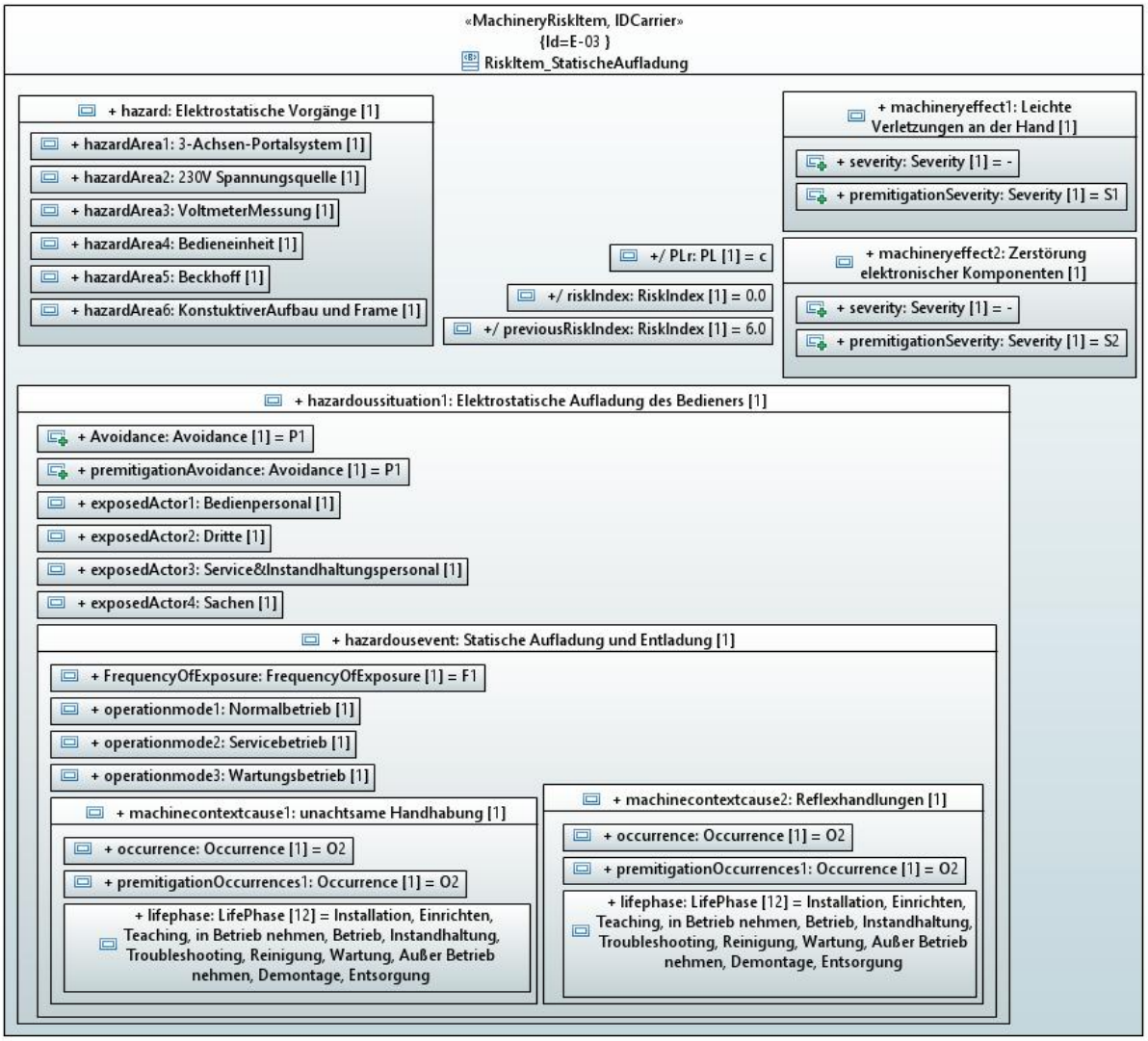


Abbildung A.42: Risikoitem statische Aufladung

Thermische Strahlung

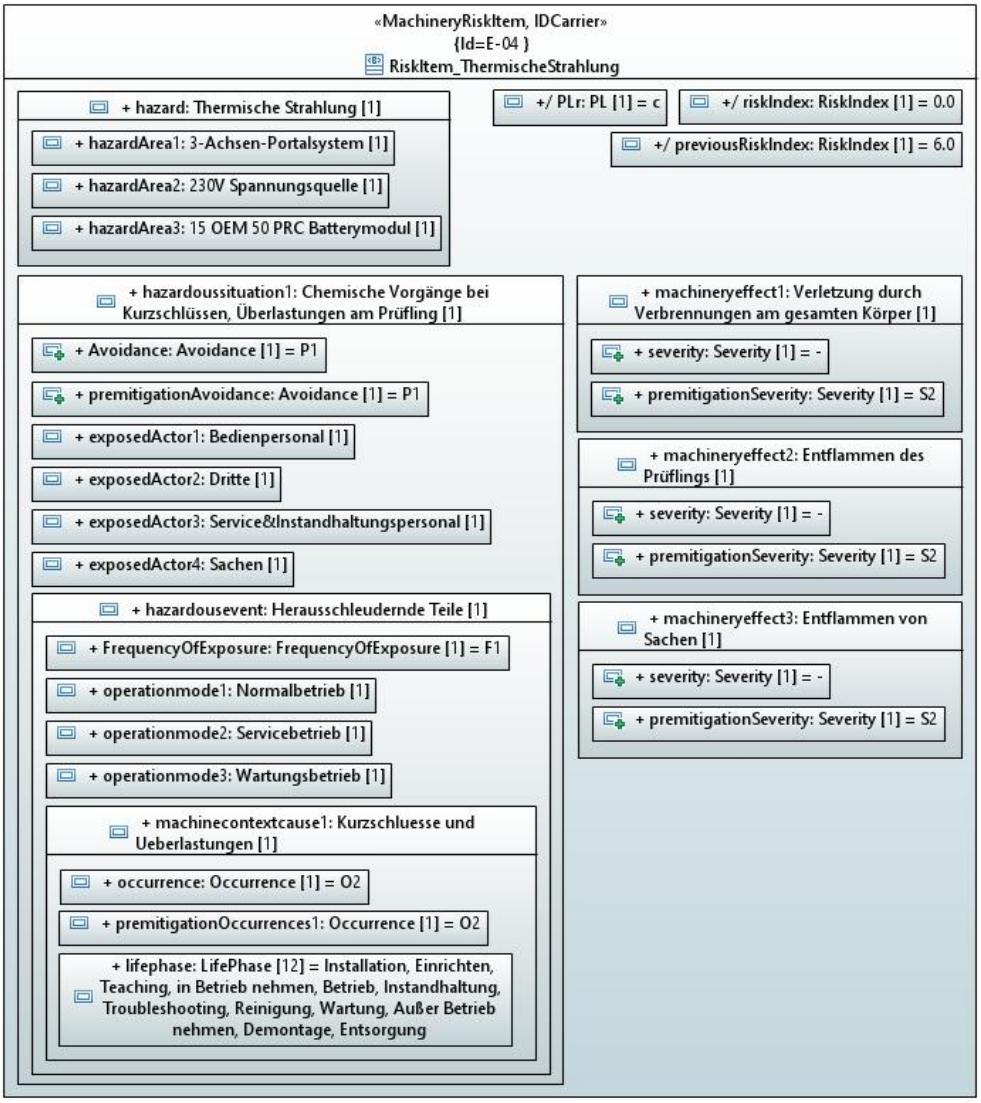


Abbildung A.43: Bewertung des Risikos von thermischer Strahlung

Verbrennung

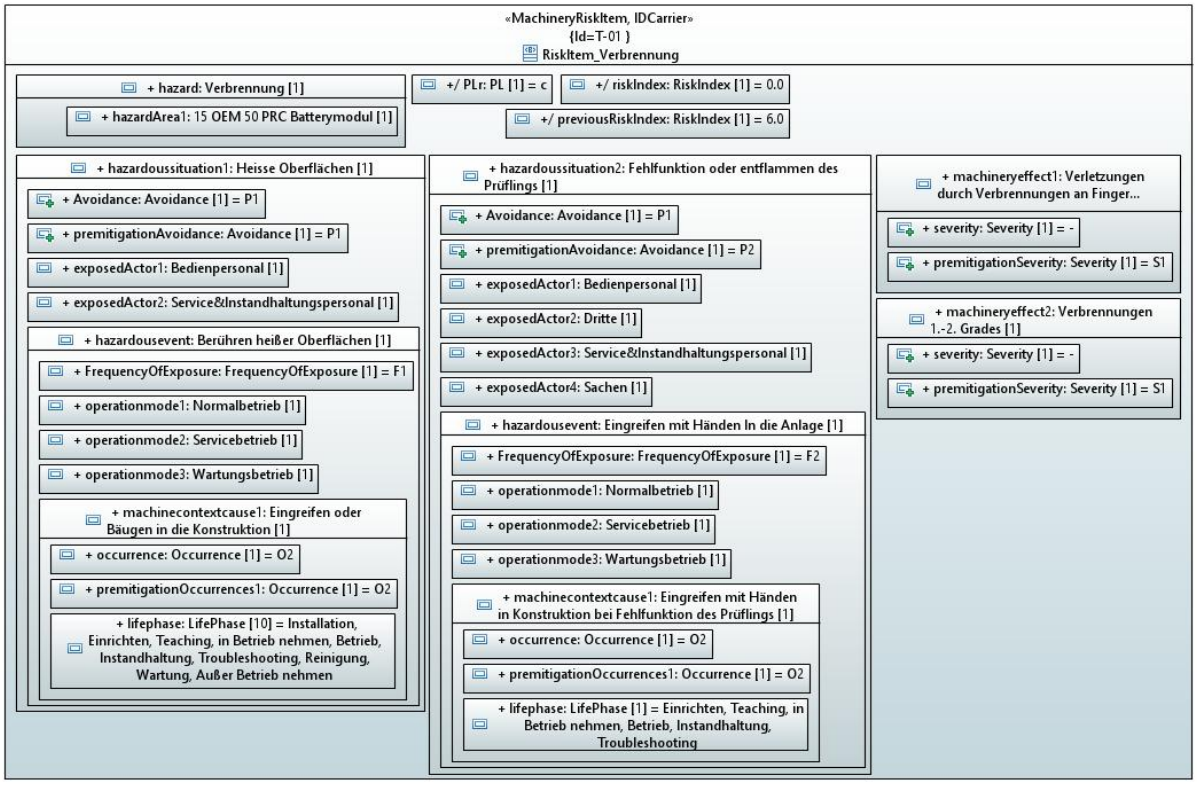


Abbildung A.44: Bewertung des Risikos von Verbrennungen

Unerwarteter Anlauf

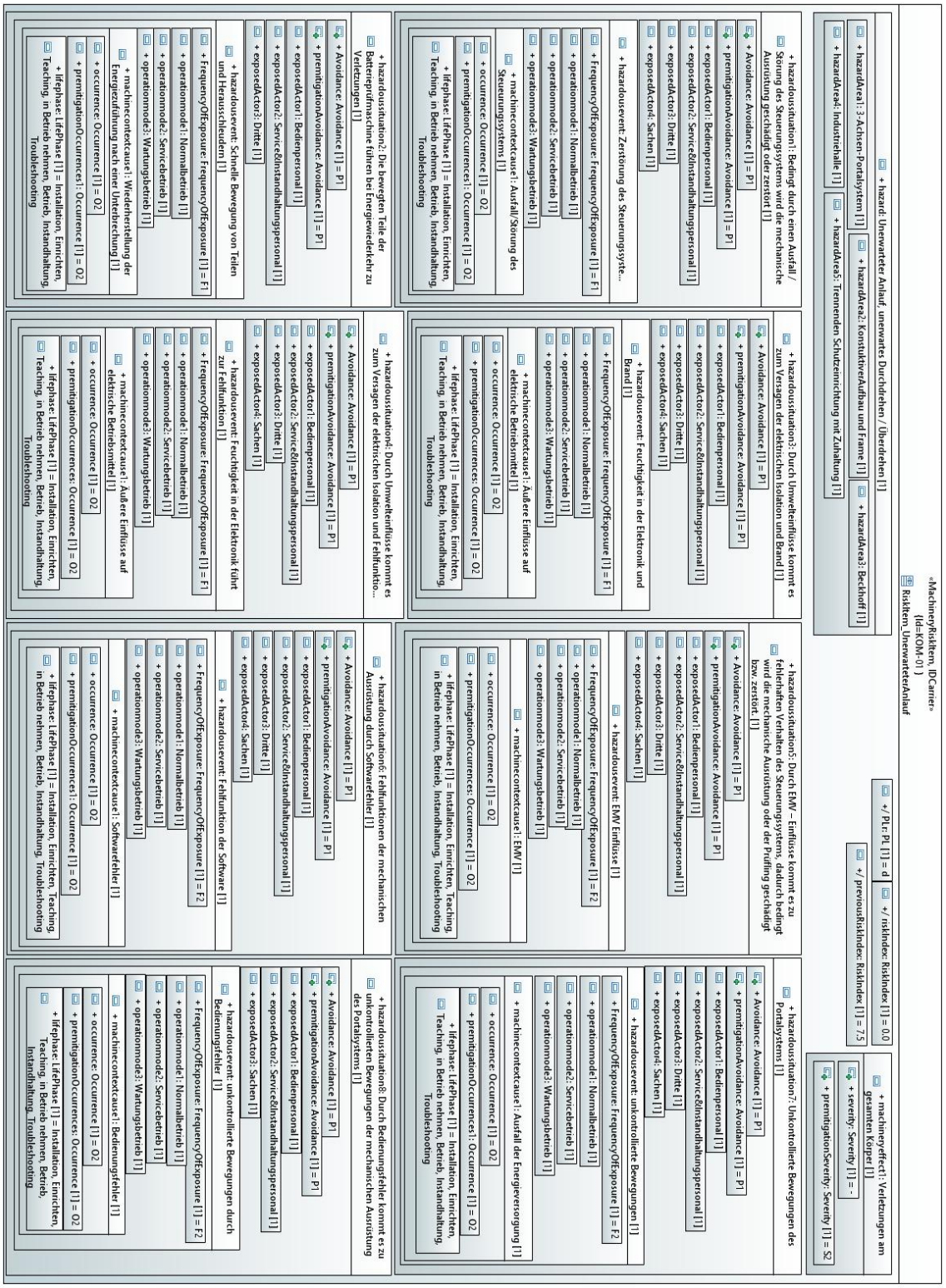


Abbildung A.45: Bewertung des Risikos eines unerwarteten Anlaufs

A.11.2 Sicherheitsbezogenes Eingangs-Ausgangs-Interface

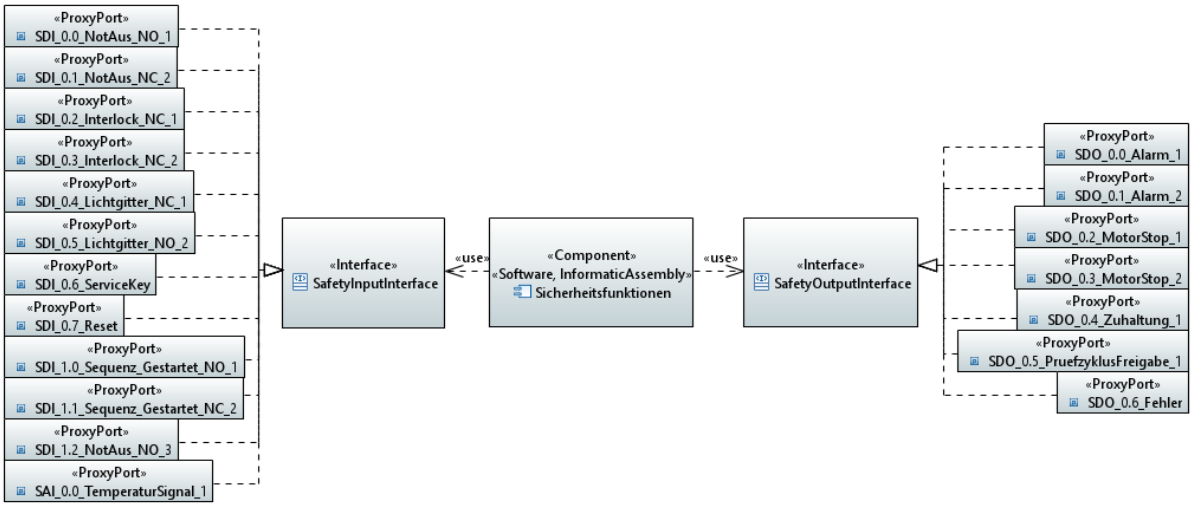


Abbildung A.46: Eingangs- und Ausgangsinterface elektrische Signale

A.11.3 Entwicklungen zur Sicherheitsfunktion „Nothalt“

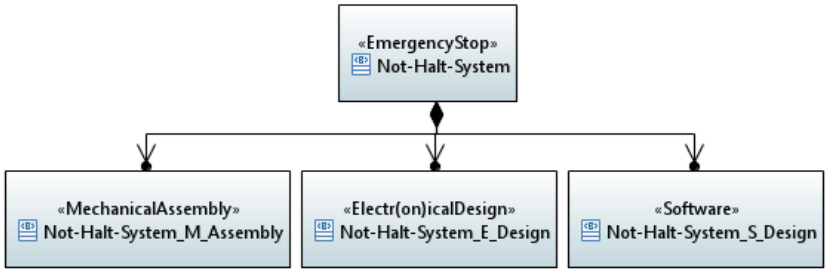


Abbildung A.47: Modellierung des Nothalt-Systems – Stakeholder-Aufteilung

Mechanische Sicht

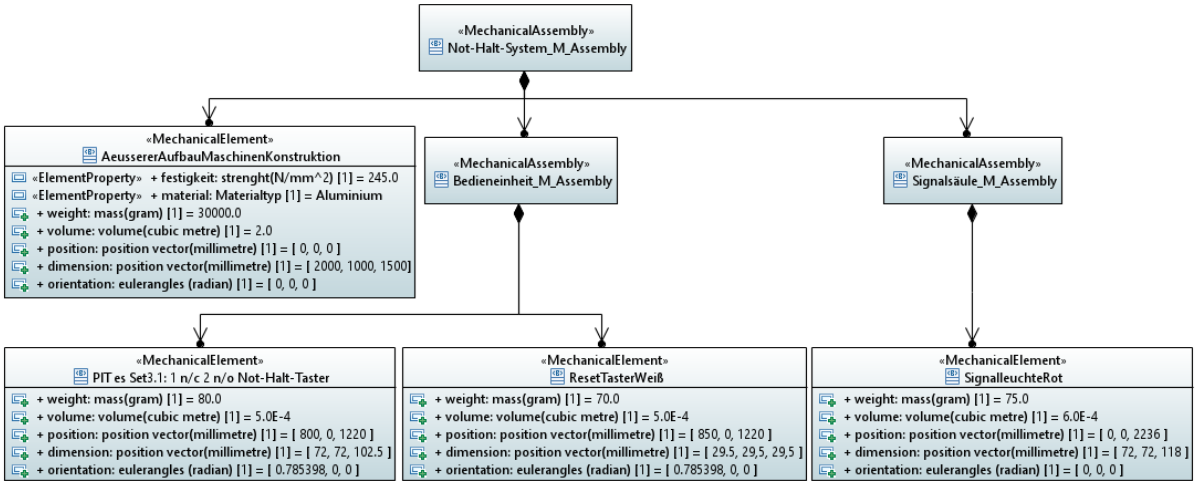


Abbildung A.48: Modellierung des Nothalt-Systems – BDD des Maschinenbauers

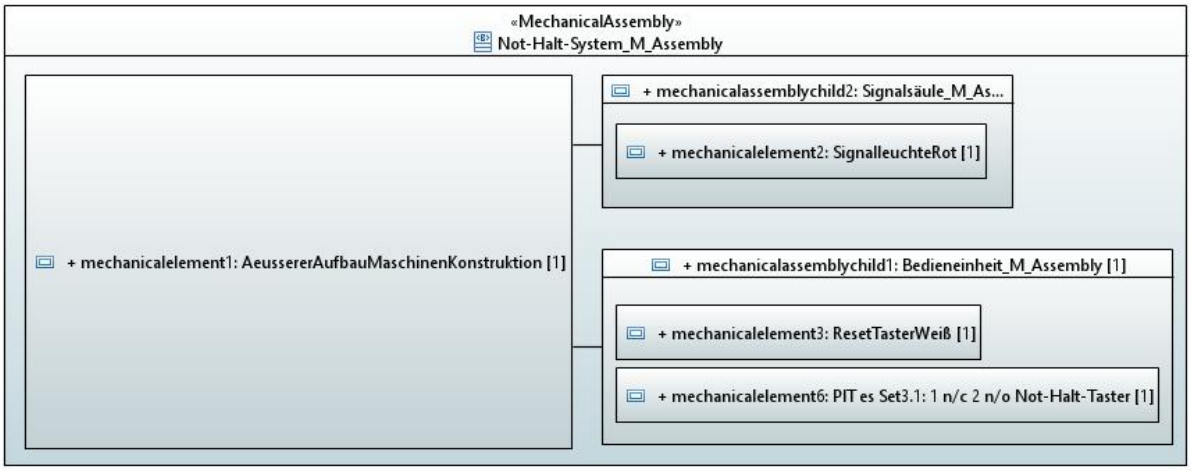


Abbildung A.49: Modellierung des Nothalt-Systems – IBD des Maschinenbauers

Elektronische Sicht

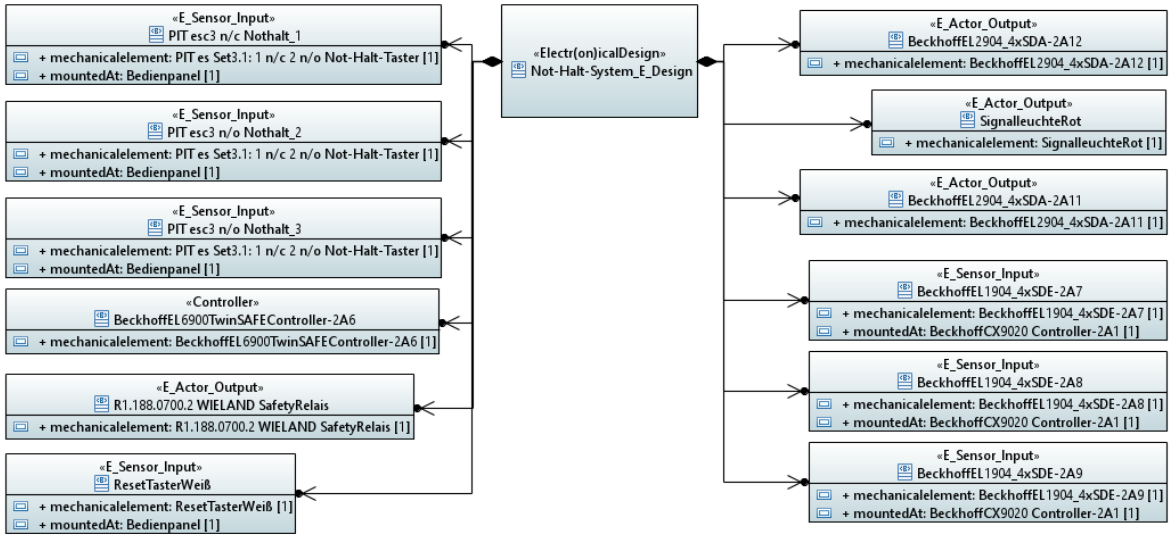


Abbildung A.50: Modellierung des Nothalt-Systems – BDD des Elektroingenieurs

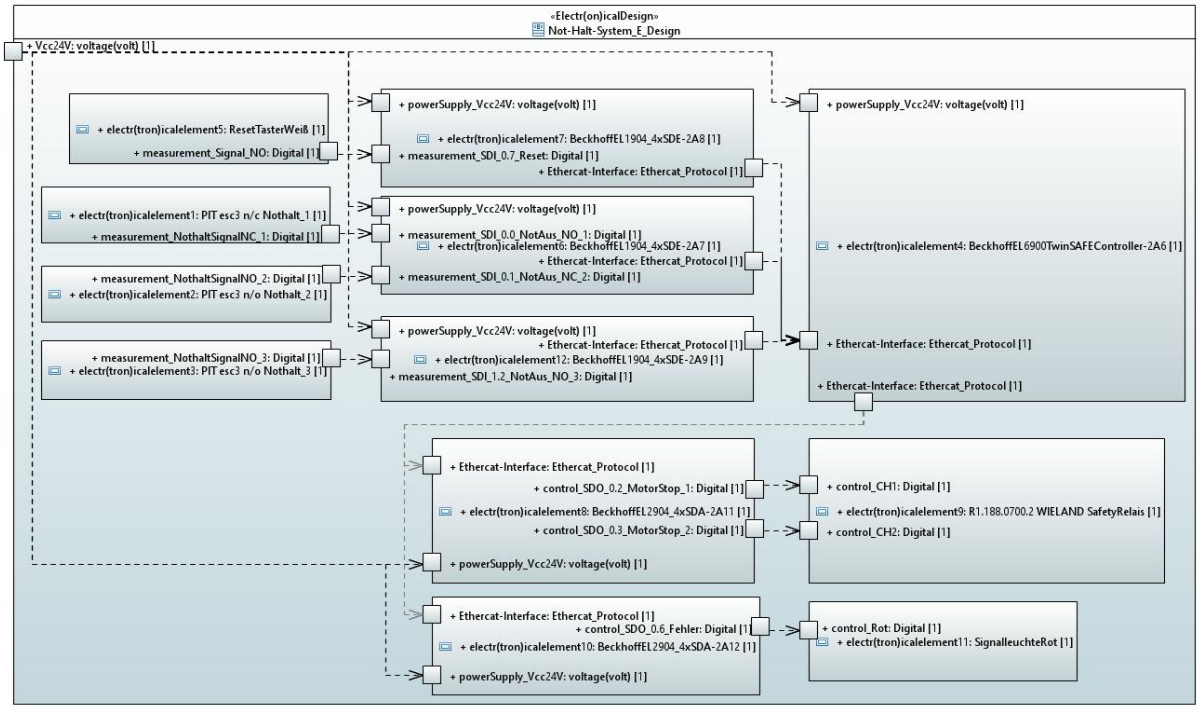


Abbildung A.51: Modellierung des Nothalt-Systems – IBD des Elektroingenieurs

Informationstechnische Sicht

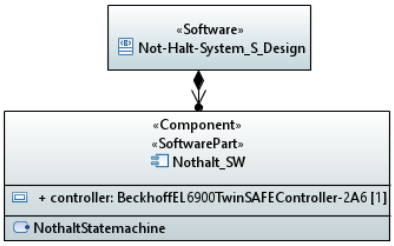


Abbildung A.52: Modellierung des Nothalt-Systems – BDD des Informatikers

Statemaschine ist in Komponente

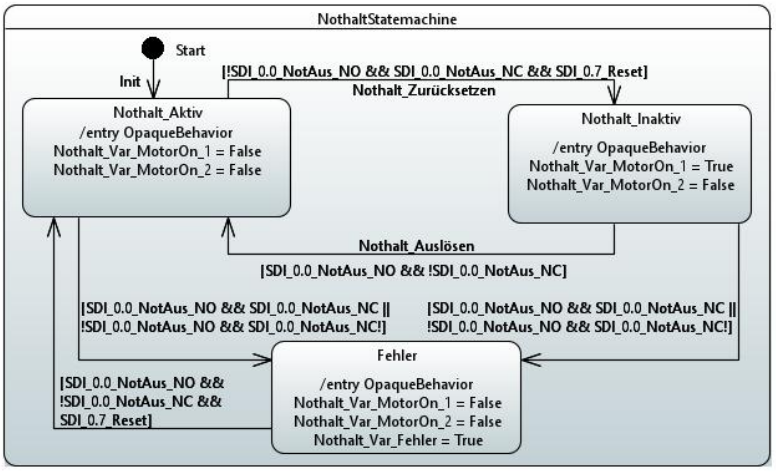


Abbildung A.53: Modellierung des Nothalt-Systems – Zustandsdiagramm

DFT

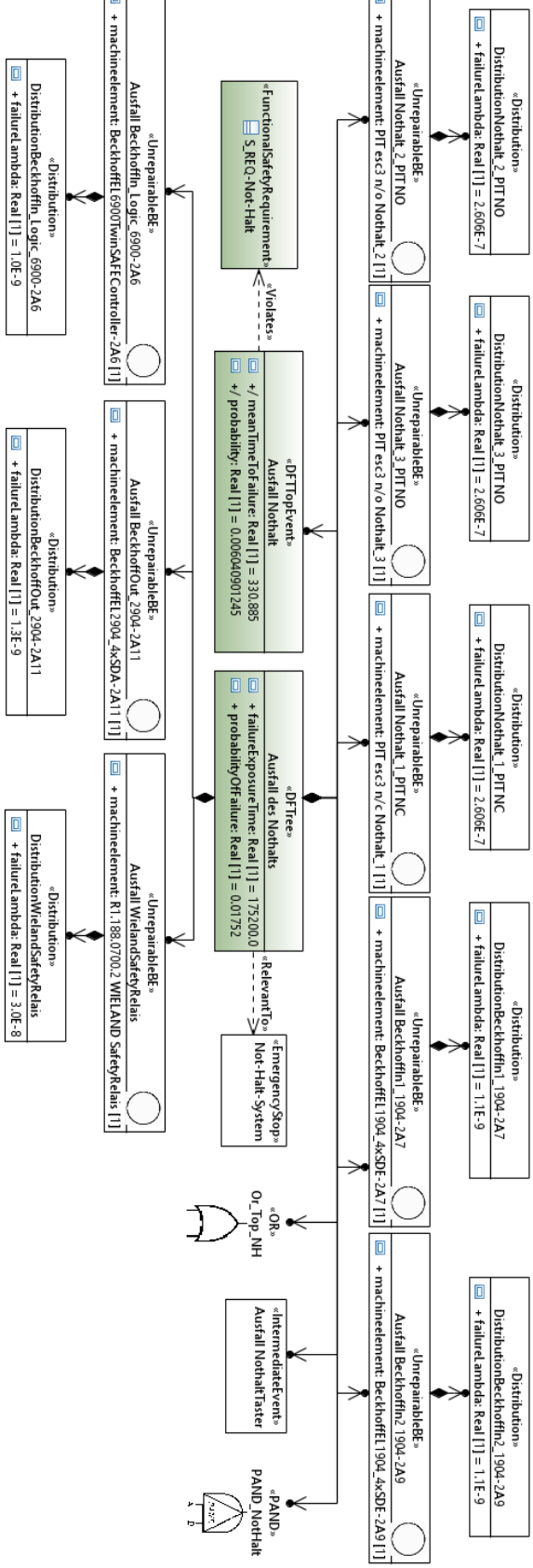


Abbildung A.54: Bewertung des Nothalt-Systems – BDD des DFTs

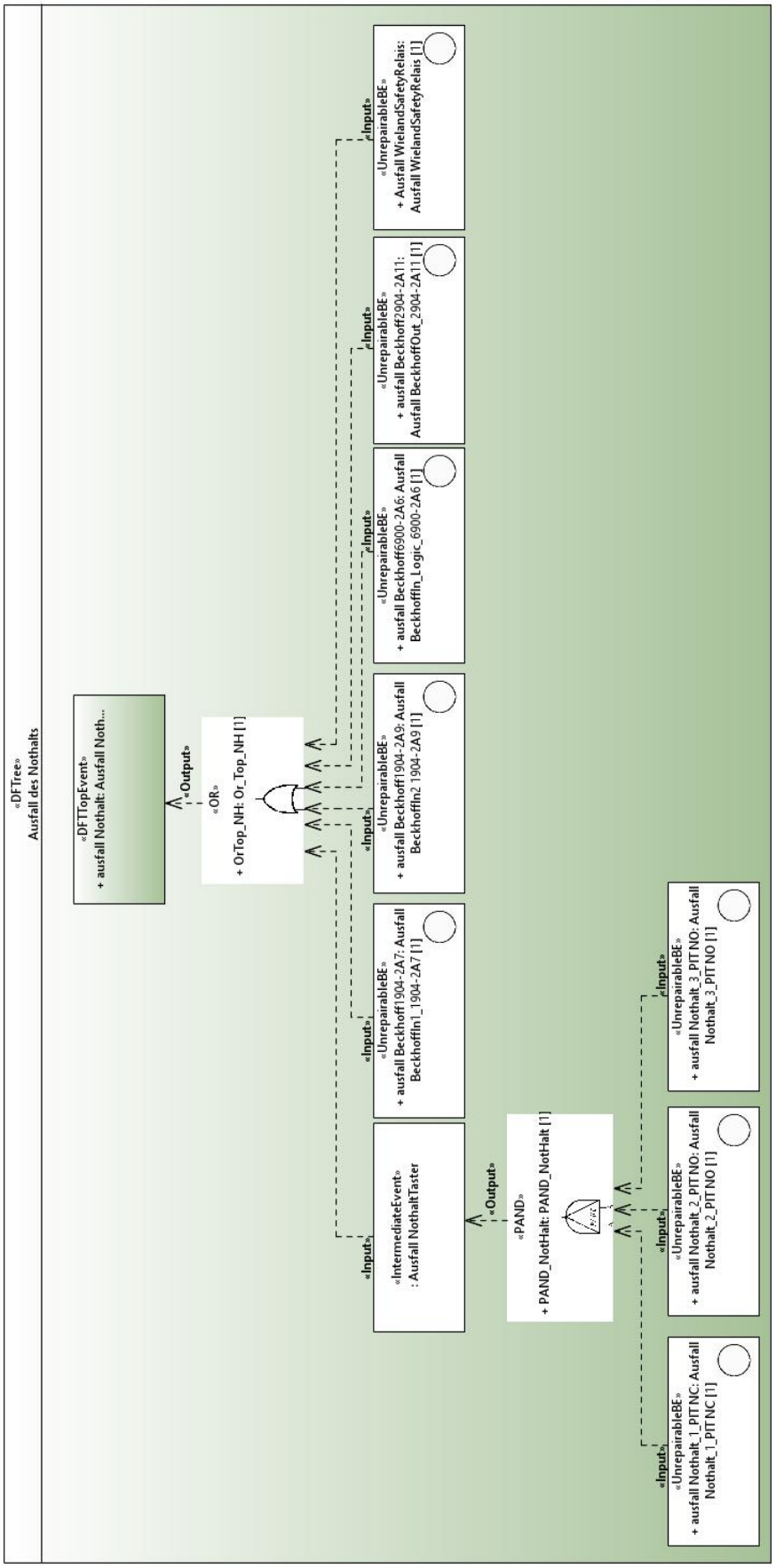


Abbildung A.55: Bewertung des Nothalt-Systems – IBD des DFTs

SBBM

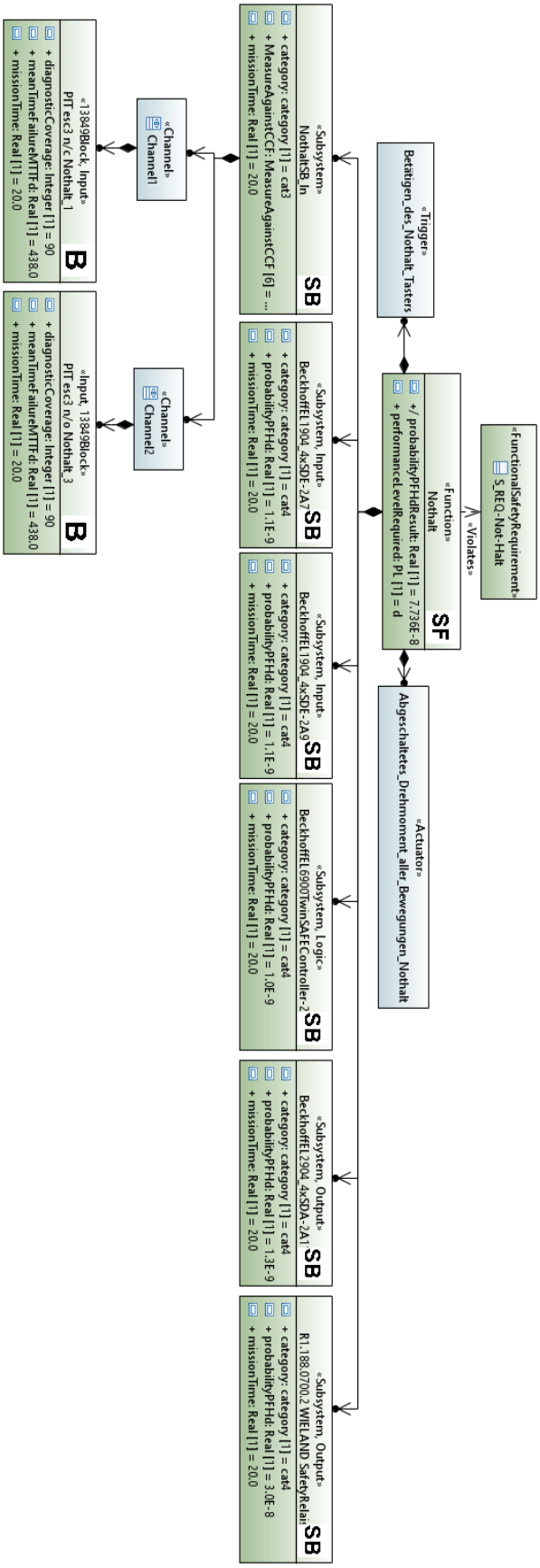


Abbildung A.56: Bewertung des Nothalt-Systems – BDD der SBBM

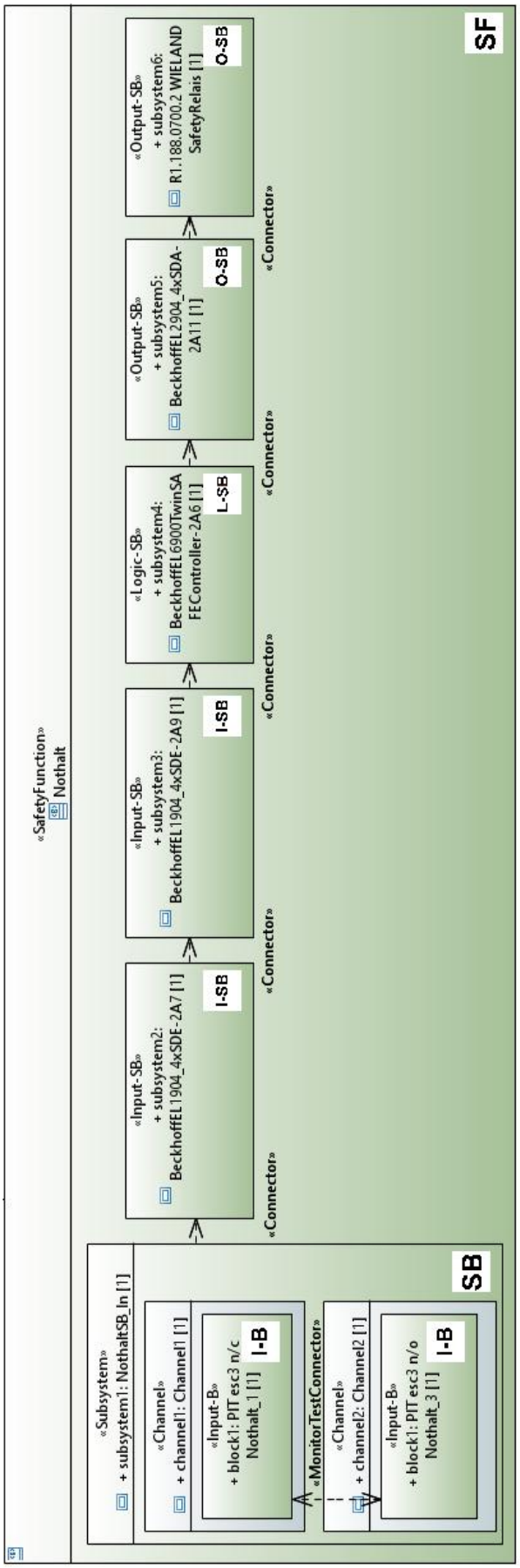


Abbildung A.57: Bewertung des Nothalt-Systems – IBD der SBBM

A.11.4 Weiterentwicklung zur Sicherheitsfunktion Prüflingsüberhitzungserkennung

Mechanische Sicht

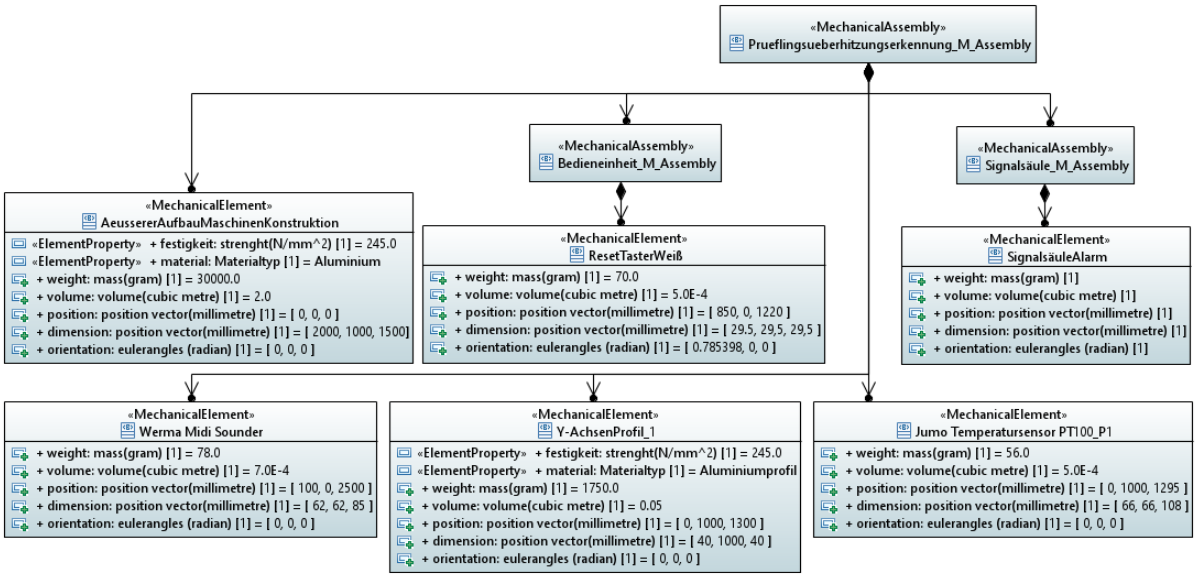


Abbildung A.58: Ext. Prüflingsüberhitzungserkennung – BDD des Maschinenbauers

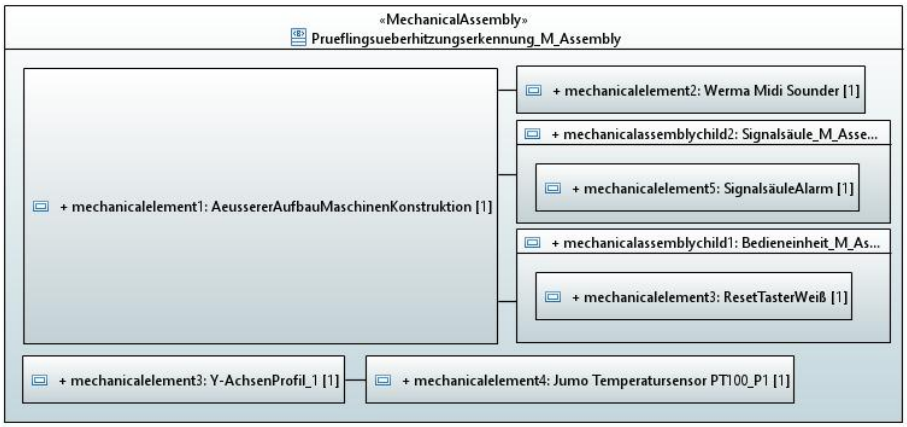


Abbildung A.59: Ext. Prüflingsüberhitzungserkennung – IBD des Maschinenbauers

Elektronische Sicht

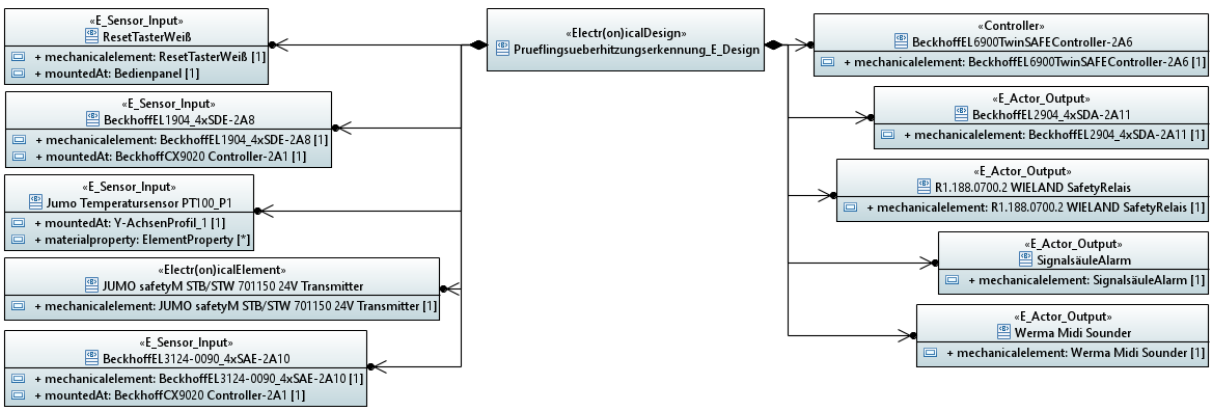


Abbildung A.60: Ext. Prüflingsüberhitzungserkennung – BDD des Elektroingenieurs

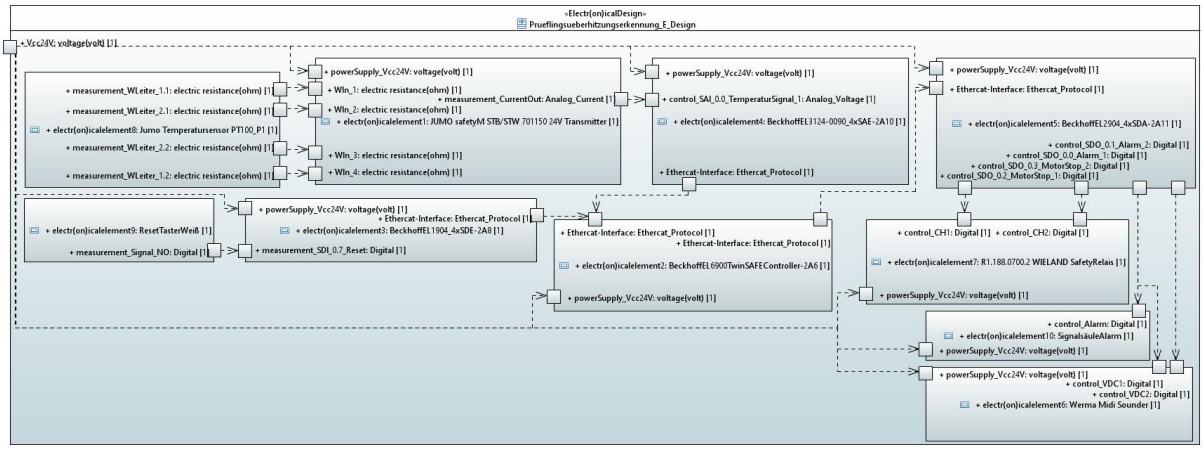


Abbildung A.61: Ext. Prüflingsüberhitzungserkennung – IBD des Elektroingenieurs

Informationstechnische Sicht

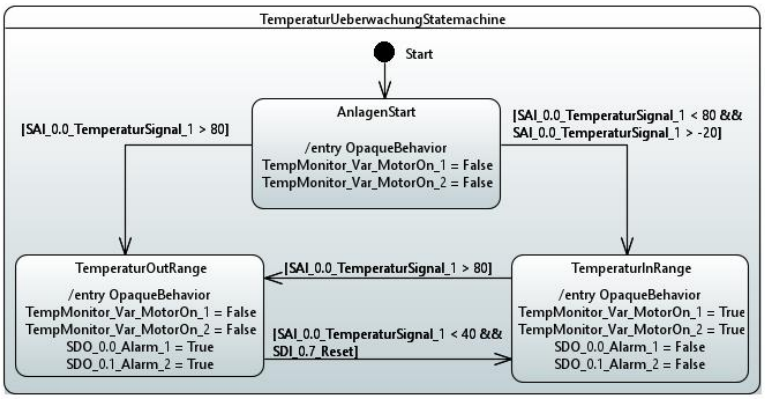


Abbildung A.62: Ext. Prüflingsüberhitzungserkennung – Zustandsdiagramm

DFT

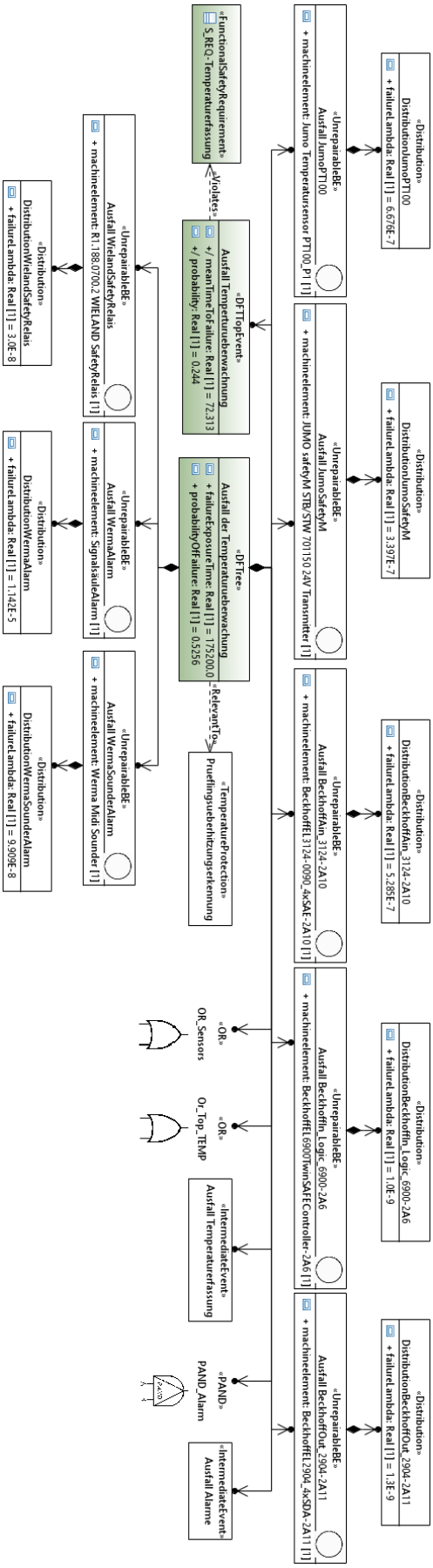


Abbildung A.63: Ext. Prüflingsüberhitzungserkennung – BDD des DFTs

SBBM

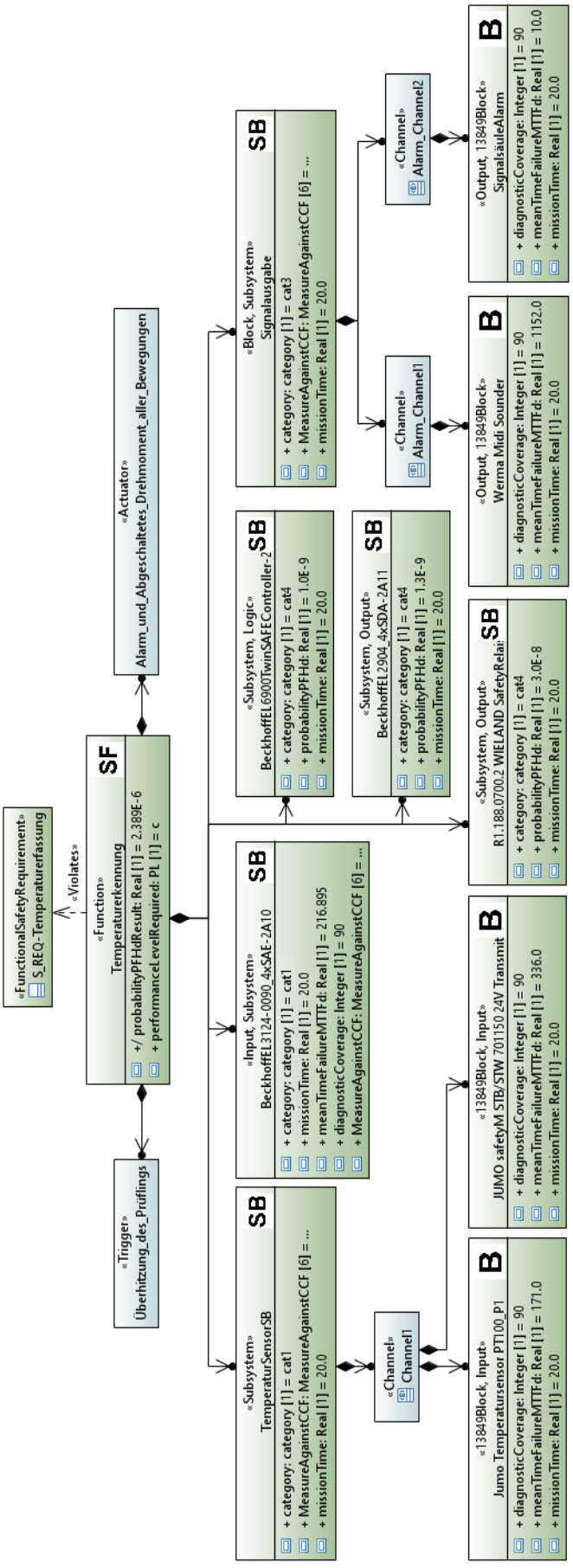


Abbildung A.64: Ext. Prüflingsüberhitzungserkennung – BDD der SBBM

A.11.5 Weitere Modelle

Übergeordnetes Zustandsdiagramm

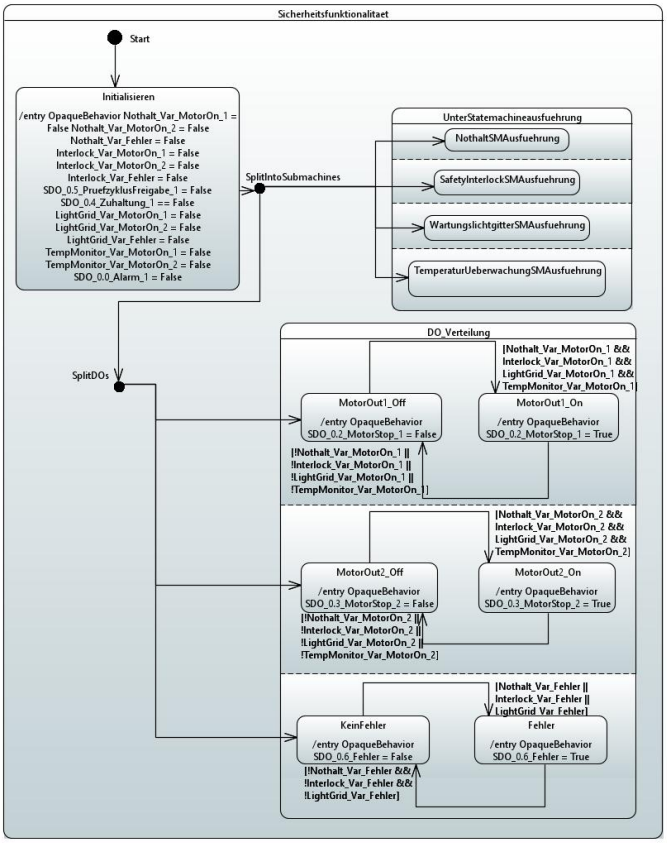


Abbildung A.65: Modellierung des übergeordneten Zustandsdiagramms

Zuhaltung

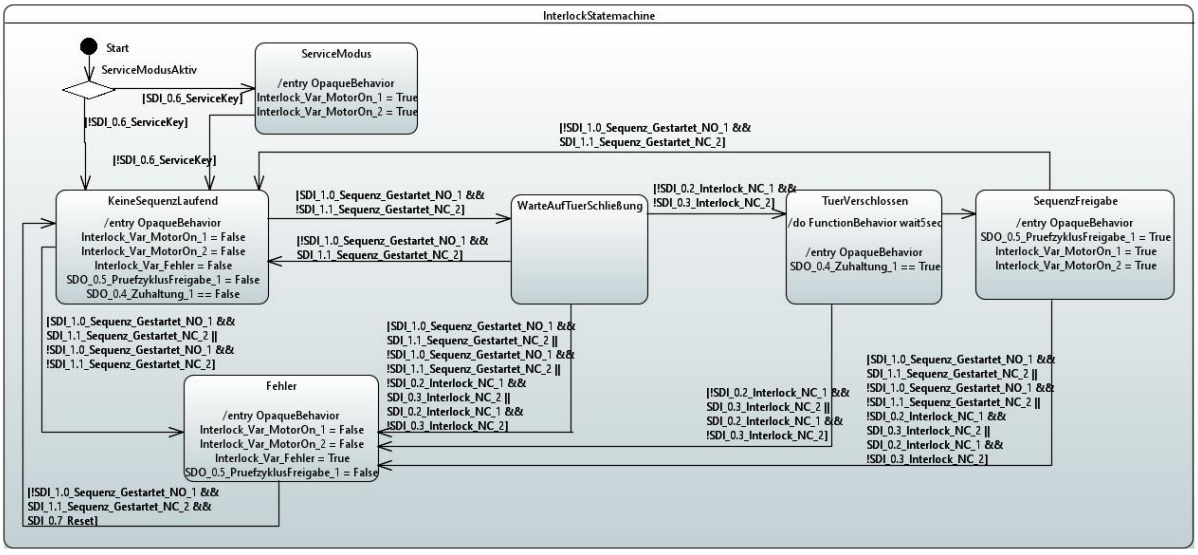


Abbildung A.66: Zuhaltung – Zustandsdiagramm

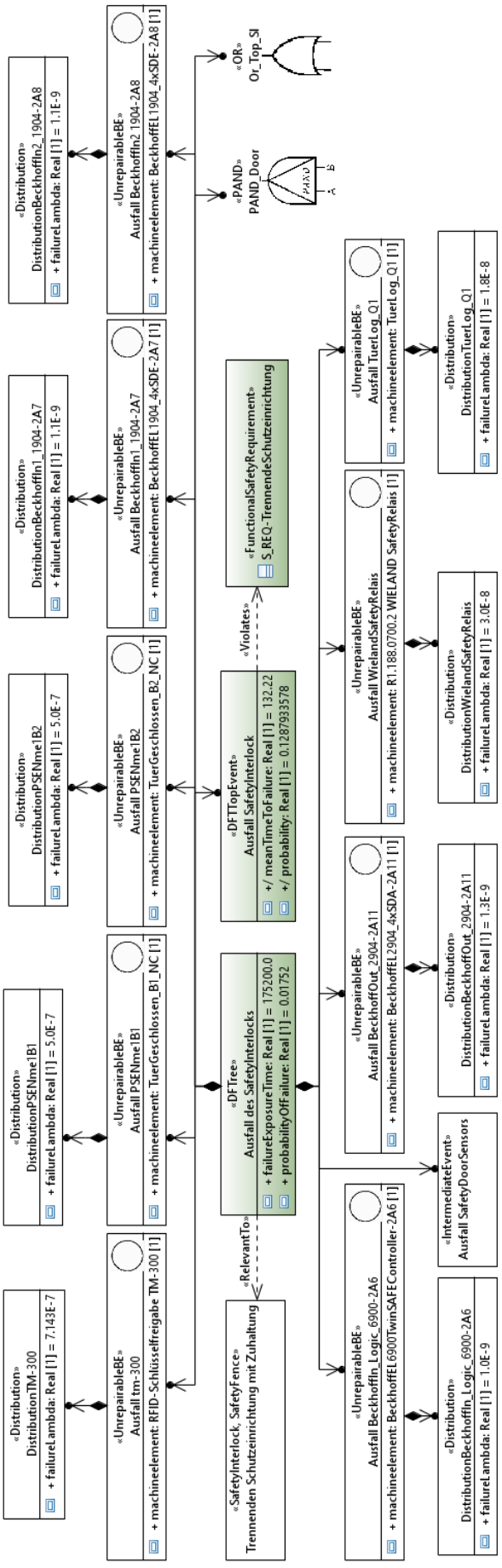


Abbildung A.67: Zuhaltung – BDD des DFTs

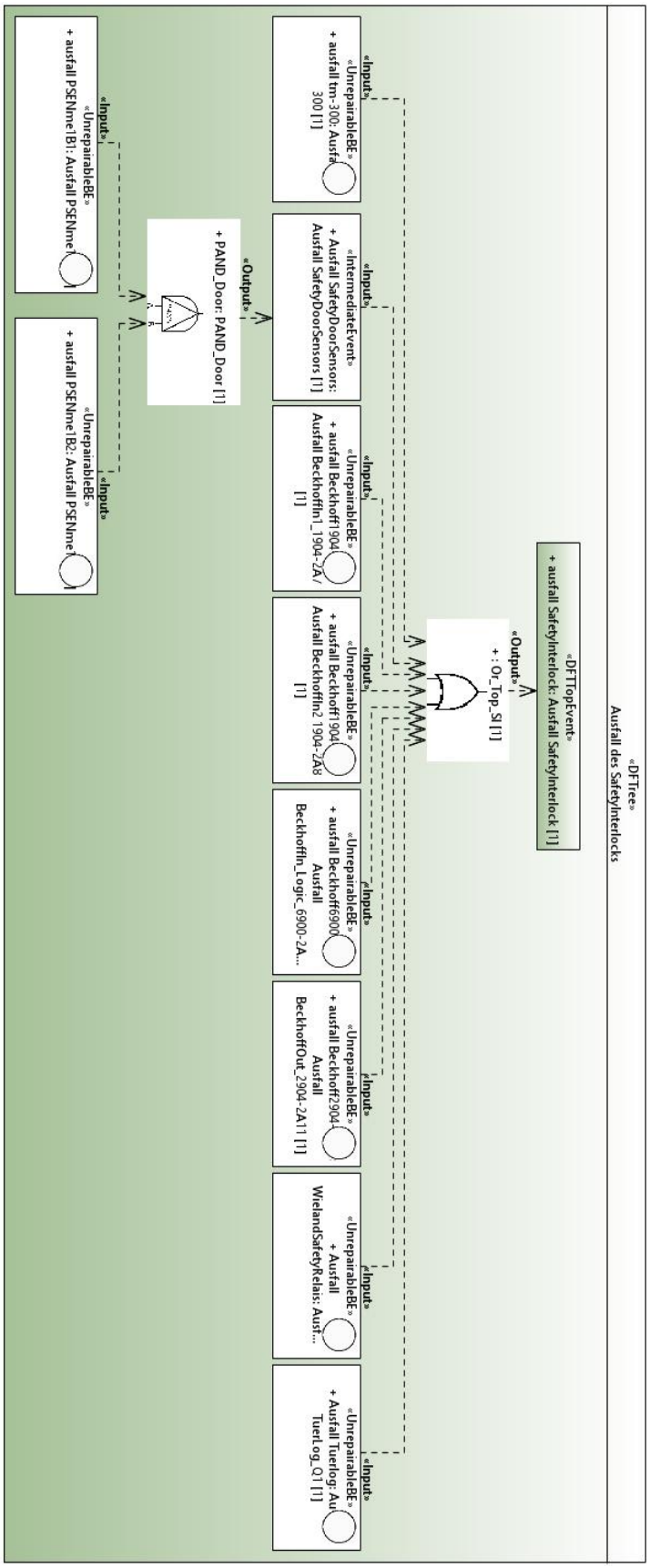


Abbildung A.68: Zuhaltung – IBD des DFTs

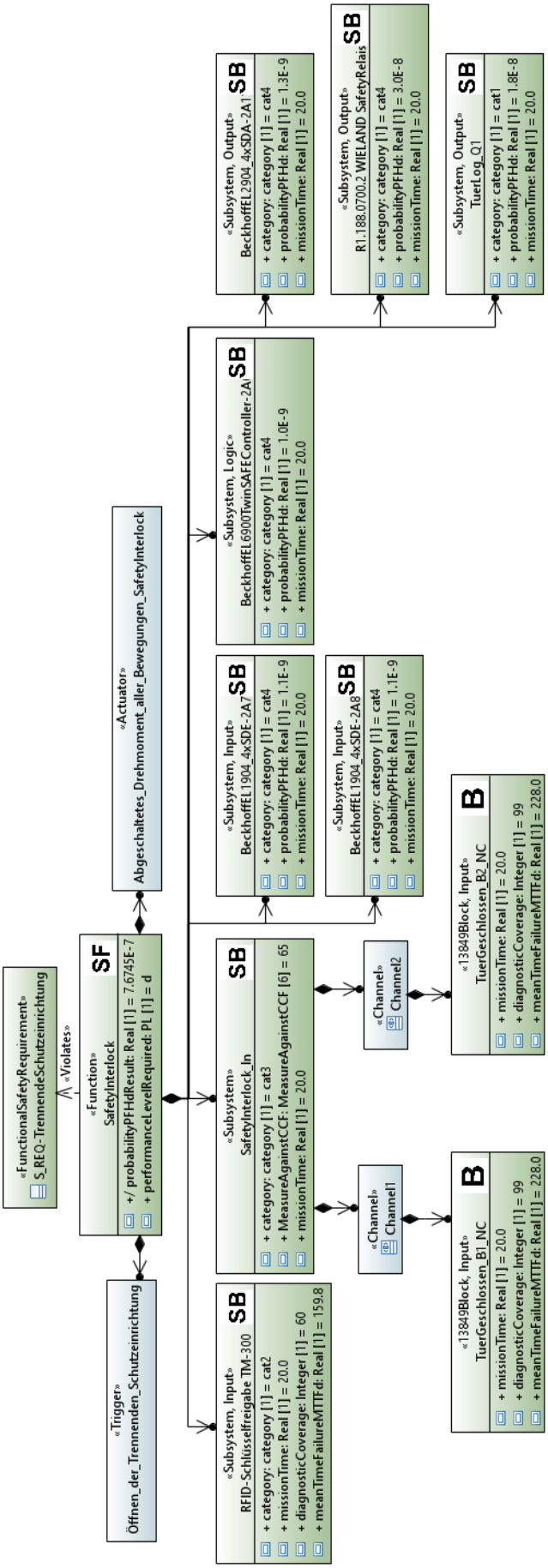


Abbildung A.69: Zuhaltung – BDD der SBBM

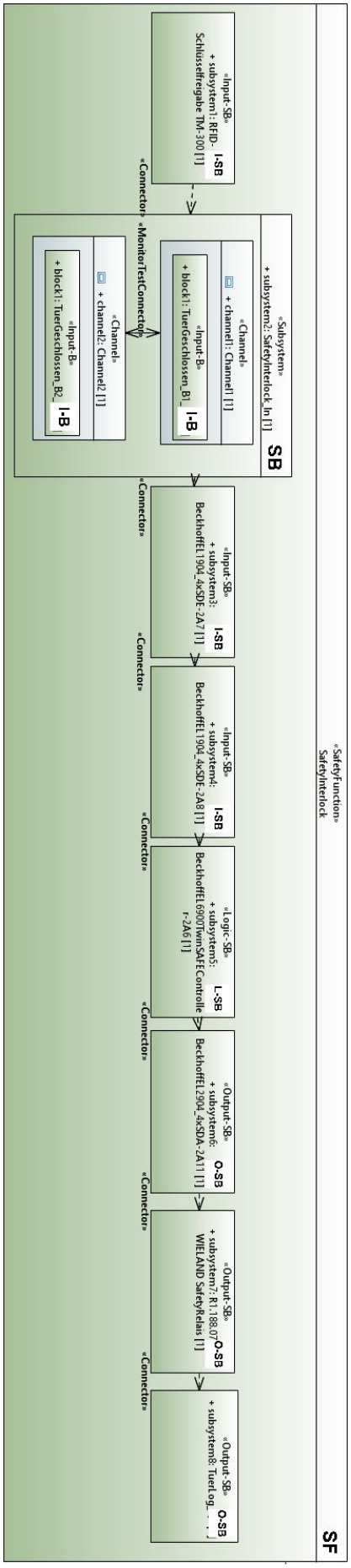


Abbildung A.70: Zuhaltung – IBD der SBBM

Wartungslichtgitter

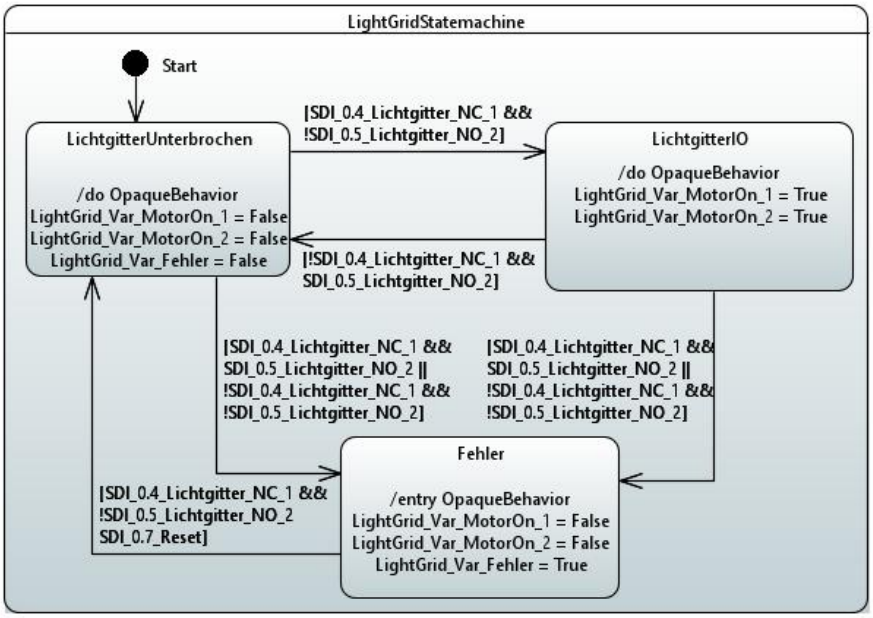


Abbildung A.71: Modellierung des Wartungslichtgitters – Zustandsdiagramm

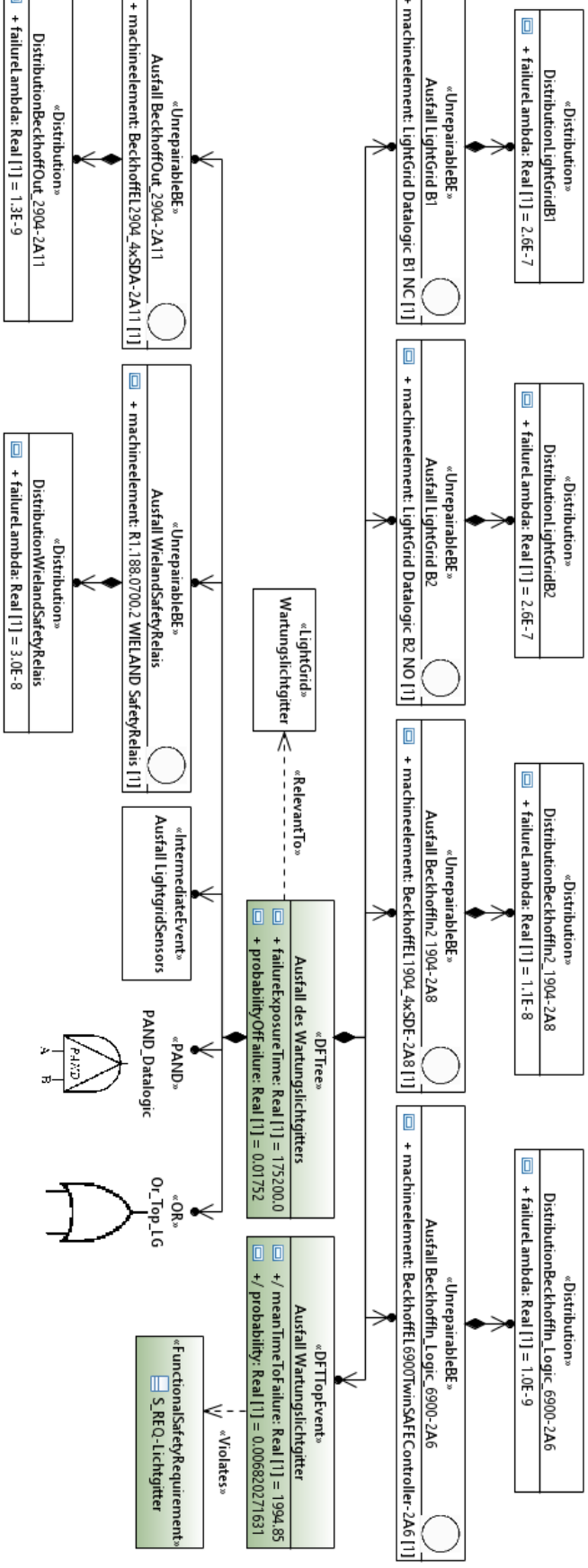


Abbildung A.72: Bewertung des Wartungslichtgitters – BDD des DFTs

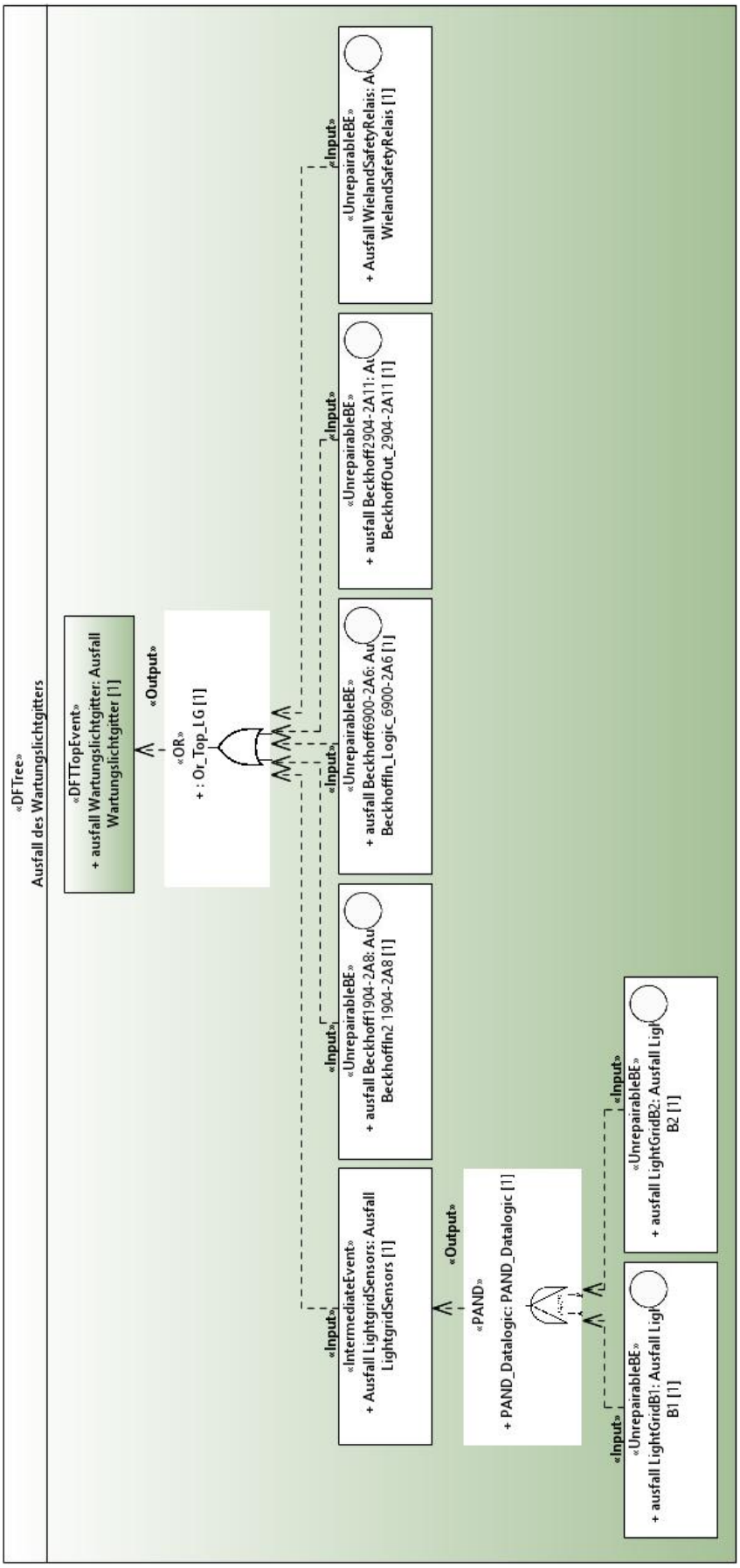


Abbildung A.73: Bewertung des Wartungslichtgitters – IBD des DFTs

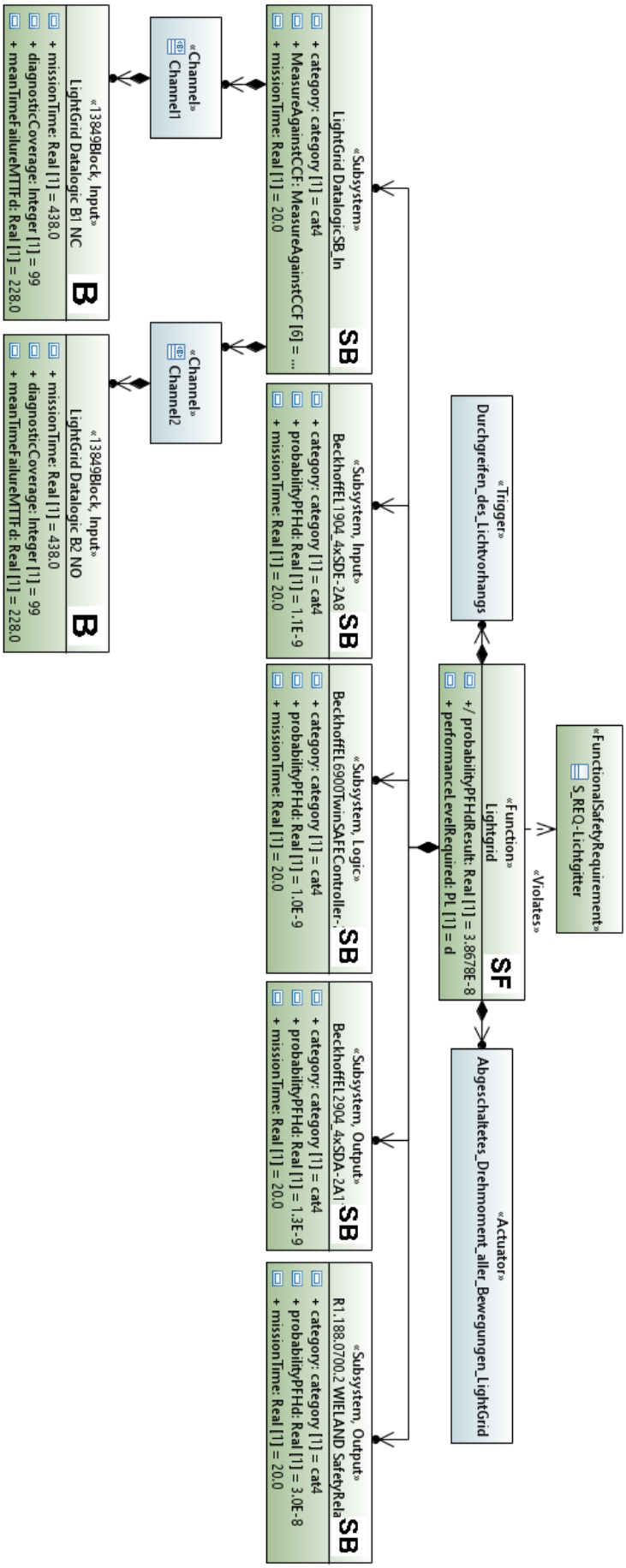


Abbildung A.74: Bewertung des Wartungslichtgitters – BDD der SBBM

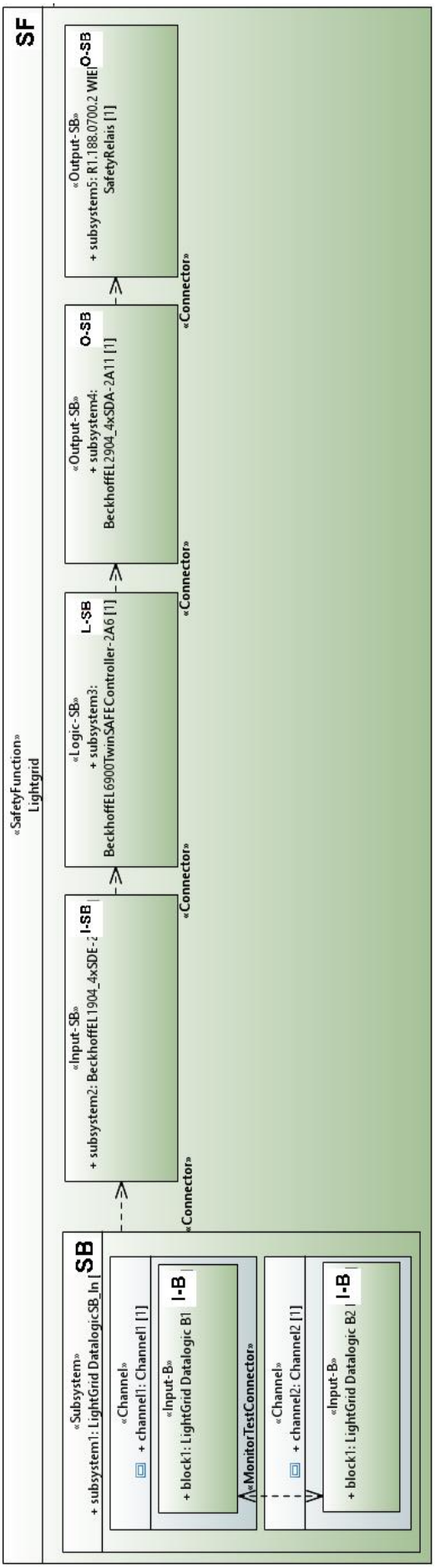


Abbildung A.75: Bewertung des Wartungslichtgitters – IBD der SBBM

A.12 Interviews zur Evaluation

Für die qualitative Evaluation wurden Fragenkataloge vorbereitet und darauf basierend Interviews durchgeführt. Im Folgenden werden die ausführlichen Antworten der interviewten Stakeholder beschrieben. Die Audiodaten und sonstigen Dokumentationen wurden transkribiert und werden so ausführlich dargestellt. Da von den Stakeholdern keine schriftliche Einwilligung zur Nutzung personenbezogener Daten und vom Unternehmen ProNES keine Einwilligung zur Nutzung unternehmensbezogener oder projektbezogener Daten erteilt wurde, wurden alle entsprechenden Daten nach [298] entfernt (Kennzeichnung durch „…“) oder anonymisiert. Hierbei handelte es sich lediglich um Ereignisse aus der Vergangenheit umgesetzter Projekte, persönliche Ereignisse oder anderweitige betriebsinterne Abläufe, die keine Relevanz zur Fallstudie hatten.

A.12.1 Fragenkatalog zur Erwartungshaltung der Stakeholder

1. Können Sie mir beschreiben, wie Sie normalerweise Ihre konstruktiven Modelle, Schaltpläne, Software und Risikobeurteilung erstellen/entwickeln?
2. Wie sieht Ihre Zusammenarbeit mit anderen Teammitgliedern aus?
3. Wie kommunizieren Sie während des Entwicklungsprozesses mit anderen Teammitgliedern? Wie sieht die Motivation aus, sich in ein Framework/Modell einzuarbeiten?
4. Haben Sie Erfahrungen mit MBSE-Frameworks?
5. Was würden Sie sich für ein solches Framework wünschen/ Was wäre ihrer Meinung nach notwendig, um ein solches Framework erfolgreich einzuführen?
6. Sehen Sie spezielle Vorteile oder Nachteile für die Einführung einer solchen Innovation in der Firma? Wie würde es Ihnen bei der Entwicklung helfen? Gibt es evtl. spezielle Informationen, die Sie von anderen Stakeholdern erhalten möchten, um Ihre Arbeit zu verbessern?

A.12.2 Protokoll zur Erwartungshaltung des Maschinenbauers

Zu Beginn des Projektes hatte der Maschinenbauer ebenso die Rolle des Sicherheitsingenieurs inne, die während des Projektes von der unter Sicherheitsingenieur aufgeführten Person übernommen wurde. Aus Übersichtsgründen wurden diese Aussagen im Protokoll belassen.

1. **Können Sie mir beschreiben, wie Sie normalerweise Ihre konstruktiven Modelle erstellen/entwickeln?**

   „Also, ich arbeite hauptsächlich mit AutoCAD, um die konstruktiven Modelle unserer Maschinen zu erstellen. Das Werkzeug wird dir ja bestimmt etwas sagen…Es ermöglicht es

mir, detaillierte 3D-Modelle der Maschine zu erstellen. Hierbei mach ich mir meist schon in der ersten Konstruktion Gedanken, ob das Konzept so sicher ist. Teilweise habe ich selbst die Sicherheitsbetrachtungen in der WEKA-Software gemacht, allerdings mit wenig Erfahrung. Aber ich weiß gar nicht, wo ich anfangen soll. Ich kann nicht anfangen, mein System zu modellieren und gleichzeitig wissen, welche Maßnahmen ich dann dafür jetzt einsetzen muss. Weil, wie du schon meint, dafür müsste ich die ganzen Normen in der WEKA-Software, die ja im Grunde eigentlich nur ein unterstützendes Tool ist, kennen, wo ein ganzes Unternehmen sich nur damit beschäftigt, diese Normen da einzuarbeiten und dir da entsprechende Hilfestellungen zu geben. Das kann ich kaum machen…Die meisten Firmen leben davon, dass sie eine Gefährdungsbeurteilung für eine Maschine anpassen. Wir machen immer wieder was Neues; komplett unterschiedliche Maschinen für komplett unterschiedliche Einsatzgebiete. Als Inverkehrbringer der Anlage sind wir neben den direkten Gefahren der Anlage ebenso für die Gefahren verantwortlich, die vom Prüfling ausgehen, was schwer einzuschätzen ist, ohne detaillierte Kenntnisse der Maschinenelemente und Normierung. Bei diesen Batterien zum Beispiel soll ich mich um die Sicherheitstechnik kümmern und ich habe nur Fragezeichen im Kopf…Oft habe ich dann von vorherigen Betrachtungen kopiert, soweit möglich…Ich kann dir auch gern ein paar Risikobeurteilungen von Nemko zeigen, die frühere Maschinen betrachtet haben…Meine Arbeit beginnt oft isoliert, aber sobald ich eine vorläufige Idee habe, teile ich das mit dem Team…"

2. **Wie sieht Ihre Zusammenarbeit mit anderen Teammitgliedern aus?**

   „…Ich arbeite das grobe Konzept einer neuen Maschine meist mit dem Elektroingenieur und Informatiker in gemeinsamen Meetings aus und darauf aufbauend nutze ich CAD, um meine Modelle zu entwickeln. Rückfragen geschehen dann mündlich, per Teams-Chat oder per E-Mail. Das ist schnell, effektiv und komfortabel. Oft schicken wir Dokumente hin und her…Da ich am Anfang der Kette stehe, kommen erstmal alle auf mich zu. Eine integrierte Plattform, die es uns ermöglicht, Modelle zu teilen und Feedback zu geben, so wie du es beschreibst, könnte bestimmt irgendwie die Effizienz und das Verständnis verbessern…Kann ich mir aber noch nicht so vorstellen. Die direkte Kommunikation finde ich aber gut. Meiner Meinung nach könnte man hier auch AutoCAD einsetzen. Hier gibt es Möglichkeiten, Schaltpläne darzustellen, Programmcode einzubinden und das kann dann als Grundlage für die Sicherheit dienen…"

3. **Wie kommunizieren Sie während des Entwicklungsprozesses mit anderen Teammitgliedern? Wie sieht die Motivation aus, sich in ein Framework/Modell einzuarbeiten?**

„…Wir kommunizieren größtenteils digital – E-Mail, Teams, Telefonate, Anforderungsdokumente…Aber, um ganz ehrlich zu sein, möchte ich keinen Mehraufwand. Ich sitze am Anfang der Kette und möchte nicht noch mehr Daten pflegen müssen…“

4. **Haben Sie Erfahrungen mit MBSE-Frameworks?**
   „Ich habe keine direkte Erfahrung mit MBSE-Frameworks. Meine Arbeit mit AutoCAD ist eher konventionell…Obwohl ich gestehen muss, dass ich mir unsicher bin, wie viel Aufwand die Implementierung eines solchen Systems bedeuten würde.“

5. **Was würden Sie sich für ein solches Framework wünschen/ Was wäre Ihrer Meinung nach notwendig, um ein solches Framework erfolgreich einzuführen?**
   „Ich würde mir ein Framework wünschen, das eine nahtlose Integration von mechanischen Modellen ermöglicht und gleichzeitig einfach zu bedienen ist und wenig bis keinen Mehraufwand für mich bedeutet. Das Ganze sollte offen und erweiterbar sein, um spezifische Anforderungen des Maschinenbaus und unsere Projekte zu erfüllen…“

6. **Sehen Sie spezielle Vorteile oder Nachteile für die Einführung einer solchen Innovation in der Firma? Wie würde es Ihnen bei der Entwicklung helfen? Gibt es evtl. spezielle Informationen, die Sie von anderen Stakeholdern erhalten möchten, um Ihre Arbeit zu verbessern?**
   „Die Einführung eines MBSE-Frameworks sehe ich vor allem als Chance, die Effizienz unserer Entwicklungsprozesse zu steigern. So wie ich es rausgehört habe, willst du uns eine verbesserte Kommunikation verkaufen und das Verständnis zwischen uns allen verbessern. Ich bin da noch etwas kritisch. Allein den Aufwand für Schulungen und Anpassung unserer Arbeitsweisen. Aber, wenn dann würde ich mir da detaillierte Rückmeldungen zu den Anforderungen vom Elektroingenieur oder Informatiker wünschen…Vielleicht kann ich die Konstruktionen dann besser auf die Bedürfnisse der Elektrotechnik und der Softwareentwicklung abstimmen…Man könnte auch einen Vorteil darin sehen, wenn sich nicht nur Entwickler untereinander, sondern auch externe und nicht technische Stakeholder zurechtfinden können. Das könnte häufige Fehlkommunikationen mit Kunden oder fehlerhafte Einkäufe verringern, auch wenn ich das auch in AutoCAD sehe.“

A.12.3 Protokoll zur Erwartungshaltung des Elektroingenieurs

1. **Können Sie mir beschreiben, wie Sie normalerweise Ihre Schaltpläne erstellen/entwickeln?**

„Ich nutze vorrangig EPLAN für die Entwicklung meiner Schaltpläne…Meine Arbeit an den Schaltplänen beginnt, sobald ich die ersten Entwürfe und Modelle vom Maschinenbauer erhalte…So einen richtigen Einblick in seine Konstruktion, wie du sie bereits genannt hast, fällt mir eher schwer.“

2. **Wie sieht Ihre Zusammenarbeit mit anderen Teammitgliedern aus?**
   „Die Koordination mit dem Maschinenbauer und Informatiker ist entscheidend für die Erstellung der Schaltpläne. Wir haben in den Projekten mehr oder weniger regelmäßige Meetings, um den Fortschritt zu besprechen. Ich will schon sichergehen, dass alle Aspekte der Maschinenentwicklung abgedeckt sind. Allerdings finde ich es aktuell nicht so perfekt. Mir fällt es schwer, meine Schaltpläne aktuell zu halten. Ein gemeinsamer Datenpool könnte helfen…Ich denke aber auch, man könnte ein Werkzeug nutzen, um Informationen effizienter untereinander zu teilen. Ich habe manchmal sogar das Gefühl, dass Daten mehrfach produziert werden…

3. **Wie kommunizieren Sie während des Entwicklungsprozesses mit anderen Teammitgliedern? Wie sieht die Motivation aus, sich in ein Framework/Modell einzuarbeiten?**
   „Wir können definitiv unsere Effizienz steigern. Manchmal werden Änderungen in den Schaltplänen gemacht, die bei der Inbetriebnahme auffallen. Ich fände es super, wenn diese besser kommuniziert werden, dass ich die Änderungen mit wenig Mehraufwand einpflegen kann.“

4. **Haben Sie Erfahrungen mit MBSE-Frameworks?**
   „Bislang habe ich persönlich noch nicht direkt mit MBSE-Frameworks gearbeitet. Ich sehe jedoch das Potenzial, dass das die Übersichtlichkeit und Nachverfolgbarkeit verbessern könnte. Die größte Herausforderung für mich wäre die Einarbeitungszeit in ein solches Framework. Wir haben im Grunde nie Zeit für irgendetwas. Da halte ich die Einführung diese Frameworks für zu langwierig. Wenn ich die Zeit bekommen würde, fände ich das toll. Aktuell bin ich aber der einzige EPLAN-Entwickler in der Firma…Oft betreue ich eine Vielzahl von unterschiedlichen Projekten gleichzeitig und bekomme kaum Zeit für Weiterbildungen eingeräumt. Bei ProNES ist nie viel Zeit über…“

5. **Was würden Sie sich für ein solches Framework wünschen/ Was wäre Ihrer Meinung nach notwendig, um ein solches Framework erfolgreich einzuführen?**
   „Aus meiner Perspektive müsste ein solches Framework leicht zugängliche Schnittstellen für die elektrische Planung bieten. Eine Integration in EPLAN oder die Möglichkeit,

Schaltpläne direkt im Framework zu generieren und zu bearbeiten, wäre ideal. Die Förderung einer engen Zusammenarbeit durch gemeinsame Bearbeitungsmöglichkeiten und Echtzeit-Feedback wäre ebenfalls von Vorteil. Nichtsdestotrotz, ohne solide Einführungsstrategie, wie schrittweise Schulungen und Workshops, halte ich das für recht aussichtslos.“

6. **Sehen Sie spezielle Vorteile oder Nachteile für die Einführung einer solchen Innovation in der Firma? Wie würde es Ihnen bei der Entwicklung helfen? Gibt es evtl. spezielle Informationen, die Sie von anderen Stakeholdern erhalten möchten, um Ihre Arbeit zu verbessern?**
   „Wichtig wäre für mich, frühzeitig Einblick in die mechanischen Konzepte und Softwareanforderungen zu bekommen, um die elektrotechnische Planung effektiver gestalten zu können.“

A.12.4 Protokoll zur Erwartungshaltung des Informatikers

1. **Können Sie mir beschreiben, wie Sie normalerweise Ihre Software erstellen/entwickeln?**
   „Meine Arbeit konzentriert sich auf die Programmierung von SPSen, vor allem mit der TwinSAFE Plattform von Beckhoff…Von meiner Seite aus, sobald ich eine Übersicht über den mechanischen Aufbau und die elektrischen Komponenten habe, beginne ich mit der Entwicklung der Software. Manchmal steht sogar schon die ganze Maschine, bis ich anfange. Da ich in mehreren Projekten arbeite, komme ich da sonst gar nicht richtig mit…TwinSAFE kann die relevanten Controllerkomponenten direkt einbinden und die SPS wird dann mit so graphischen Bausteinen programmiert. Eine enge Abstimmung mit dem Maschinenbauer und Elektroingenieur ist, sag ich mal, dabei essentiell. Aktuell kann ich sonst gar nicht nachvollziehen, was ich wie programmieren muss…Oft muss ich für die nötigen Informationen aber mehrfach nachfragen oder bekomme sie viel zu spät…“
2. **Wie sieht Ihre Zusammenarbeit mit anderen Teammitgliedern aus?**
   „Ich bekomme meine Informationen meist eher nur von technischen Spezifikationen…Ich bin auf klare Informationen vom Maschinenbauer und Elektroingenieur angewiesen, um die Software entsprechend entwickeln zu können. Wenn mir der Sicherheitsingenieur ebenso zusätzliche Vorgaben über notwendige Konzepte gibt, fände ich das super. Aktuell nutzen wir hier hauptsächlich digitale Kommunikationswege wie E-Mail und Teams, aber

ich sehe definitiv Raum für Verbesserungen… Vielleicht können wir dann Missverständnisse reduzieren und die Entwicklung beschleunigen.“

3. **Wie kommunizieren Sie während des Entwicklungsprozesses mit anderen Teammitgliedern? Wie sieht die Motivation aus, sich in ein Framework/Modell einzuarbeiten?**
   „Für die direkte Kommunikation bevorzuge ich den Teams-Chat, da er schnelle Antworten ermöglicht. Manchmal treffen wir uns einmal die Woche, manchmal einmal am Tag, je nachdem welche Phase und welche Dringlichkeit. Ich entwickle aber auch gern mal für mich allein, wenn ich alle Informationen habe. Missverständnisse lassen sich aber eh nie vermeiden, besonders wenn es um technische Details geht, mach ich lieber ein Meeting mehr als zu wenig, wenn Zeit ist. Vielleicht wäre eine visuellere Kommunikationsform hilfreich? Ein Bild sagt ja mehr als 1000 Worte.“

4. **Haben Sie Erfahrungen mit MBSE-Frameworks?**
   „Ich habe in der Tat etwas Erfahrung mit MBSE, hauptsächlich aber von der Uni…Die Verwendung von SysML zur Modellierung von Systemen hat definitiv seine Vorteile, besonders wenn es um die Kommunikation und Dokumentation komplexer Systemanforderungen geht. Ich würde es begrüßen, wenn wir ein solches Framework nutzen könnten, um unsere Arbeit zu vereinheitlichen und zu vereinfachen…Wenn du mir jetzt dort Vorgaben geben kannst, die ich einfach nur nutzen muss oder die mir bereits alle relevanten Informationen geben, fänd ich das super. Die Herausforderung sehe ich in der Anpassung unserer aktuellen Prozesse und der Schulung des Teams.“

5. **Was würden Sie sich für ein solches Framework wünschen/ Was wäre Ihrer Meinung nach notwendig, um ein solches Framework erfolgreich einzuführen?**
   „…Ich halte Kompatibilität und Interoperabilität mit bestehenden Werkzeuge, wie unseren, für unerlässlich. Ein Framework, das eine Brücke zwischen der Softwareentwicklung, dem mechanischen Design und der Elektrokonstruktion schlagen kann, würde den Entwicklungsprozess stark vereinfachen. Darüber hinaus wäre eine zentrale Datenbank oder Repository, die eine einfache Versionierung und den Austausch von Modellen ermöglicht, sehr nützlich. Ich weiß nicht, ob das dem entsprichst, was du planst…“

6. **Sehen Sie spezielle Vorteile oder Nachteile für die Einführung einer solchen Innovation in der Firma? Wie würde es Ihnen bei der Entwicklung helfen? Gibt es evtl. spezielle Informationen, die Sie von anderen Stakeholdern erhalten möchten, um Ihre Arbeit zu verbessern?**

„Für die Softwareentwicklung sehe ich einen großen Vorteil in der klaren Definition und Nachvollziehbarkeit von Systemanforderungen, die ein MBSE-Framework ermöglichen würde…Ich denke aber, das ganze könnte komplex und fehleranfällig werden, aber ich bin gespannt…Ich sehe einen klaren Vorteil darin, wenn ich ein Feedback bekomme, ob mein modellierter Programmcode mit den Anforderungen ans System, den Sicherheitsvorgaben, der Risikobeurteilung und Risikominderung übereinstimmt…Um meine Arbeit zu verbessern, wären daher genaue Informationen über die Hardware-Spezifikationen und die funktionalen Sicherheitsanforderungen direkt zu Beginn des Entwicklungsprozesses hilfreich…“

A.12.5 Protokoll zur Erwartungshaltung des Sicherheitsingenieurs

1. **Können Sie mir beschreiben, wie Sie normalerweise Ihre Risikobeurteilung erstellen/entwickeln?**

   „Ich bin zwar erst neu bei ProNES, doch kümmere ich mich auch bereits schon bei meinem vorherigen Arbeitgeber hauptsächlich um die Bewertung der funktionalen Sicherheit. Soweit ich weiß, verwenden wir bei ProNES die WEKA-Software für die Risikobeurteilung und SISTEMA für den Nachweis der Risikominderung. Auch wenn ich mich bisher nur weniger mit WEKA auseinandergesetzt habe, glaube ich, dass es hier bei der Anwendung wenige Unterschiede gibt. In meiner alten Firma habe ich diese Risikobeurteilungen teilweise in tabellarischer Form ausgefüllt und berechnet. Zu den aktuellen Vorgehensweisen und so kann ich noch nicht so viel sagen, aber eine der Hauptherausforderungen für mich sehe ich in einer Liste oder sowas wie eine Funktionsbeschreibung anhand der Einzelteile, die mir das Rahmenkonzept und die geplante Umsetzung einer Maschine darstellt…Sonst kann ich gar keine Risikobeurteilung machen. SISTEMA kann ich ohne genaue Komponenten und deren Sicherheitsfunktionen eigentlich überhaupt nicht einsetzen bzw. ergibt für mich dann garkeinen Sinn.“

2. **Wie sieht Ihre Zusammenarbeit mit anderen Teammitgliedern aus?**

   „Meine Zusammenarbeit ist vorwiegend reaktiv. Ich werde meist dann hinzugezogen, wenn meine Kollegen Unterstützung bei der Bewertung der Maschine brauchen, also wie schon gesagt, der Risikobeurteilung…Die Dokumentation des Ganzen ist ja eh nur für unsere Unterlagen…Mehr als Dokumente produziere ich ja gar nicht. Aber ich denke, wenn ich früher eingebunden werde, könnten potenzielle Risiken schneller erkannt werden. Heißt

eben auch eine höhere Gesamtsicherheit und geringere Kosten…Aktuell sehe ich dafür aber keine große Möglichkeit. Ich bin gespannt, was du dafür bereithältst"

3. **Wie kommunizieren Sie während des Entwicklungsprozesses mit anderen Teammitgliedern? Wie sieht die Motivation aus, sich in ein Framework/Modell einzuarbeiten?**
   „Da meine Ergebnisse sowieso Dokumente darstellen, sehe ich keinen negativen Mehraufwand, diese aus Modellen zu erstellen. Im Gegenteil. Ich finde das klingt spannend, auch wenn ich mir noch nicht so richtig vorstellen kann, wie das am Ende aussieht. Es gibt zu viele Normen und Standards, nach die man sich teilweise richten kann und teilweise richten muss. Da gibt es die ISO 12100 aber auch spezielle Gerätenormen, die man für Brenner, Walzen oder Batterien nutzt. Diese Informationen sollten auch kommuniziert werden…"

4. **Haben Sie Erfahrungen mit MBSE-Frameworks?**
   „Ich muss gestehen, dass ich MBSE heute zum ersten Mal höre. Ich kann dazu nicht viel sagen, außer dass ich es mir noch nicht so ganz vorstellen kann…Es klingt aber so, als ob es die Qualität und Sicherheit der Maschinen verbessern könnte. Die Normen sprechen ja auch von Nachvollziehbarkeit von den Anforderungen ans System bis zur Außerbetriebnahme eines Systems…Da könnte das bestimmt hilfreich sein."

5. **Was würden Sie sich für ein solches Framework wünschen/ Was wäre Ihrer Meinung nach notwendig, um ein solches Framework erfolgreich einzuführen?**
   „Für die erfolgreiche Einführung wäre es aus meiner Perspektive entscheidend, dass meine Sicherheitsbetrachtungen integriert und visualisiert werden. Ich habe das nun so verstanden, dass beispielsweise meine Gefährdungen einer Maschine direkt mit den gefährdeten Maschinenteilen verknüpft sind. Dazu muss das dann auch den Normen entsprechen. Es sollte also erlauben, Risikobeurteilungen irgendwie mit dem Schaltplan und CAD-Zeichnungen zu verbinden…Wenn ich das weiterspinne, wäre das dann ein Programm, das auch für Teammitglieder ohne Erfahrungen in MBSE anwendbar ist…Ich habe mich mal mit dem Maschinenbauer und dem Informatiker unterhalten und fände Schulungen und Unterstützung bei den ersten Entwürfen auch sehr wichtig."

6. **Sehen Sie spezielle Vorteile oder Nachteile für die Einführung einer solchen Innovation in der Firma? Wie würde es Ihnen bei der Entwicklung helfen? Gibt es evtl. spezielle Informationen, die Sie von anderen Stakeholdern erhalten möchten, um Ihre Arbeit zu verbessern?**

„Ich denke, die Integration von Sicherheitsanalysen in den Entwicklungsprozess kann bestimmt einiges vereinfachen und so zur besseren Einhaltung von Sicherheitsstandards führen. Allerdings sind die Vielzahl von Standards für Gerätegruppen und einzelne Geräte eine Menge und bestimmt schwer händelbar. Ich bin auch sehr gespannt, wie es am Ende aussieht.“

A.12.6 Fragenkatalog zur Zufriedenheit der Stakeholder

1. Ich habe Ihnen mein Projekt gezeigt. Wie denken Sie darüber? Hat es Ihre Erwartungen getroffen?
2. Welche Vorteile sehen Sie nun in dieser engen Verbindung zwischen den Modellen? Wie würde sich das Framework auf Ihre Zusammenarbeit im Team ausgewirken?
3. Welche Verbesserungsmöglichkeiten oder Herausforderungen sehen Sie bei der Anwendung des Frameworks?
4. Es wurde damals angegeben, dass es wichtig ist, dass das Framework ebenfalls bei der Kommunikation mit Kunden helfen soll. Trifft dies Ihrer Meinung nach zu?
5. Es wurde damals angegeben, dass die Ressourcen zur Einführung des Frameworks zu knapp sind. Sehen Sie das immer noch so, sollte das Framework außerhalb dieses Forschungsprojektes breitflächig in der Firma eingesetzt werden?
6. Könnte das Framework ebenso bei der Beschaffung von Materialien, also dem Einkauf von Maschinenteilen helfen?
7. Haben Sie noch weitere Anmerkungen oder Nachteile, die sie zu meinem Projekt beitragen möchten? Gibt es noch andere Bereiche/Teile, die verbessert werden könnten?
8. Sehen Sie das Framework als vollständigen Ersatz für die bisherige mündliche Kommunikation?

A.12.7 Protokoll zur Zufriedenheit des Maschinenbauers

1. **Ich habe Ihnen mein Projekt gezeigt. Wie denken Sie darüber? Hat es Ihre Erwartungen getroffen?**

   „Das Projekt sieht auf den ersten Blick interessant aus…Mir leuchtet aber nicht ein, was mein Vorteil ist, dieses Framework vor meinem CAD zu setzen. Computer Aided Design ist eigentlich alles. Du kannst einen digitalen Zwilling in AutoCAD erstellen, wenn du möchtest…Es gibt definitiv Potenzial hier, aber ich sehe auch Raum für Verbesserungen…Mir scheint du erfindest hier das Rad neu…Meiner Meinung nach kann man das alles in CAD machen und es wäre weniger fehleranfällig. Die Integration der verschiedenen Modelle und die daraus resultierende Verbesserung der Kommunikation und Zusammenarbeit sind

klar Vorteile, aber ich denke nicht, dass es in der Praxis anwendbar ist…Vielleicht wenn du es noch verfeinerst."

2. **Welche Vorteile sehen Sie nun in dieser engen Verbindung zwischen den Modellen? Wie würde sich das Framework auf Ihre Zusammenarbeit im Team auswirken?**
   „Ich verstehe, was du damit bezwecken willst, aber ich sehe nur Mehraufwand für mich…Als Vorteil ja vielleicht eine bessere Kommunikation im Team oder zum Kunden…Vielleicht kann man auch Verantwortlichkeiten damit klären, aber ich denke trotzdem es ist einfacher, alles in AutoCAD einzubinden."

3. **Welche Verbesserungsmöglichkeiten oder Herausforderungen sehen Sie bei der Anwendung des Frameworks?**
   „Mir scheint das ganze etwas überladen mit Informationen…Für mich sind die Informationen vom Elektroingenieur und Informatiker eher überflüssig. Kann man die ausblenden? Oder vielleicht kann man für ein Projekt oder im Teams entscheiden, welche Aspekte genutzt werden sollen, um die Arbeitsprozesse nicht unnötig zu verkomplizieren…Nur so ein Vorschlag…Ich hoffe, du nimmst mir die scharfe Kritik nicht übel, aber, wenn ich es richtig verstanden habe, ist es ja auch meine Aufgabe…"

4. **Es wurde damals angegeben, dass es wichtig ist, dass das Framework ebenfalls bei der Kommunikation mit Kunden helfen soll. Trifft dies Ihrer Meinung nach zu?**
   „Trotz allem, ja. Man kann es sicher dafür verwenden, aber im Grunde geht das auch mit CAD. Oft will der Kunde aber nur Ergebnisse und ist mit zu vielen Details eher überfordert…Manchmal weiß er gar nicht, was er will oder versteht nicht, warum manche Sachen so nicht umgesetzt werden können…Dann könnte dein Framework wirklich helfen Kundenanforderungen genauer zu erfassen und Lösungen mit ihm zu ermitteln…Ich würde lieber bei CAD bleiben"

5. **Es wurde damals angegeben, dass die Ressourcen zur Einführung des Frameworks zu knapp sind. Sehen Sie das immer noch so, sollte das Framework außerhalb dieses Forschungsprojektes breitflächig in der Firma eingesetzt werden?**
   „Wenn du mich jetzt fragst, ob ich die Ressourcen habe, mich da einzuarbeiten, dann sag ich Nein. Persönlich sehe ich keinen Nutzen und wie damals erwähnt, haben wir keine Ressourcen."

6. **Könnte das Framework ebenso bei der Beschaffung von Materialien, also dem Einkauf von Maschinenteilen helfen?**
   „Ich denke schon, aber AutoCAD auch und das ist genauer."

7. **Haben Sie noch weitere Anmerkungen oder Nachteile, die sie zu meinem Projekt beitragen möchten? Gibt es noch andere Bereiche/Teile, die verbessert werden könnten?**
   „Dein Projekt hat sicher seine Stärken, vor allem in der Theorie. In der Praxis sehe ich mich da aber noch nicht drin…Vielleicht wenn du es noch weiter aufbaust oder mir auf lange Sicht noch ein paar Sachen erklärst. Es wäre von Vorteil, wenn das Framework anpassbarer wäre…Ähnlich CAD muss es unseren Anforderungen projektabhängig gerecht werden. Ich finde auch die Abhängigkeit von deinem Framework ist gefährlich, sollte irgendwas nicht funktionieren…Wer wartet das?…Am Ende haben da doch AutoCAD, AutoDesk, EPLAN und Co mehr Knowhow und Ressourcen…Wir brauchen Software, die auch funktioniert…Also, ob das nun aktuell zutrifft oder nicht, eine robuste fehler- und ausfallsichere Anwendung wäre mir wichtig.“

8. **Sehen Sie das Framework als vollständigen Ersatz für die bisherige mündliche Kommunikation?**
   „Nein…Besonders am Anfang des Konzepts einer neuen Maschine oder zur Klärung komplexer Probleme gehört das dazu…Oft verstehen die Kollegen erst in Meetings wie was gemeint ist…“

A.12.8 Protokoll zur Zufriedenheit des Elektroingenieurs

1. **Ich habe Ihnen mein Projekt gezeigt. Wie denken Sie darüber? Hat es Ihre Erwartungen getroffen?**
   „Dein Projekt hat mich positiv überrascht. Vor allem die enge Verbindung meiner Modelle zum CAD vom Maschinenbauer finde ich toll. Dadurch weiß ich genau, welche Entwicklungen ich vornehmen muss und wie sich alles verhalten soll. Das ganze ermöglicht scheinbar eine recht präzise und effizientere Entwicklung, scheint das ganze zur Nachvollziehbarkeit von Vor-Ort-Änderungen von Teamkollegen zu helfen. Jetzt muss es nur noch getan werden. Ich sehe, wie meine elektrotechnischen Komponenten direkt mit den mechanischen Entwicklungen interagieren müssen. Ich schätze, ich kann damit präzisere Entscheidungen treffen. Ob die Unterteilung in Sensoren, Controller und Aktoren ausreicht, glaube ich nicht, aber das stört mich weniger…Wenn ich es richtig verstehe, könnte man das ja im Nachhinein ergänzen…“

2. **Welche Vorteile sehen Sie nun in dieser engen Verbindung zwischen den Modellen? Wie würde sich das Framework auf Ihre Zusammenarbeit im Team auswirken?**

„Ich denke, wenn ich das alles richtig verstanden habe und mich da einarbeiten kann, könnte es die Entwicklungseffizienz verbessern und auch bei Inbetriebnahmen die Änderungen nachvollziehen und bequatschen. Jetzt können wir bestimmt einfacher Funktionen auch selbst entwickeln…Irgendwie ist es bestimmt auch mal interessant, den aktuellen Stand des Projektes in Sichtweise der anderen Stakeholder zu sehen…Ob das wirklich bei der Koordination hilft, weiß ich nicht. Dazu müsste ich mehr Zeit zur Einarbeitung haben."

3. **Welche Verbesserungsmöglichkeiten oder Herausforderungen sehen Sie bei der Anwendung des Frameworks?**
   „Auch nochmal an dieser Stelle. Die größte Hürde sehe ich in der Einarbeitungszeit…Kannst du Schulungsmaterialien oder einen Workshop anbieten? Dann verstehe ich vielleicht auch besser…Aber hauptsächlich muss das wohl von der Chefebene kommen. Da wir ja nie Zeit haben…Außerdem verstehe ich noch nicht ganz, ob diese Modelle alle richtig sind. Kann man das nicht auch extern über LabVIEW überprüfen. Da haben wir das meiste Knowhow"
4. **Es wurde damals angegeben, dass es wichtig ist, dass das Framework ebenfalls bei der Kommunikation mit Kunden helfen soll. Trifft dies Ihrer Meinung nach zu?**
   „Ich denke ja…Das Framework kann helfen, die Kommunikation mit Kunden zu verbessern…Da alles auf einem Blick in Zusammenhang gebracht werden kann, versteht der Kunde bestimmt die Aufwände besser…Vielleicht würde er dann auch mehr zahlen…"
5. **Es wurde damals angegeben, dass die Ressourcen zur Einführung des Frameworks zu knapp sind. Sehen Sie das immer noch so, sollte das Framework außerhalb dieses Forschungsprojektes breitflächig in der Firma eingesetzt werden?**
   „Ich denke jetzt schon, dass sich die Investition in das Framework mittel- oder langfristig auszahlen könnte…Es scheint den Entwurf effizienter zu machen…Vielleicht kann es auch den einen oder anderen Fehler vermeiden, aber wer bezahlt den Aufwand zur Einarbeitung?…Unser Chef schaut da eher aufs Geld…Vielleicht, wenn man es dem Kunden über eine verbesserte Qualität verkauft. Ich denke aber, es kommt eher nicht dazu."
6. **Könnte das Framework ebenso bei der Beschaffung von Materialien, also dem Einkauf von Maschinenteilen helfen?**
   „Bestimmt, aber das solltest du eher unseren Einkauf fragen…EPLAN hat auch so ne Option. Würde ich bevorzugen"
7. **Haben Sie noch weitere Anmerkungen oder Nachteile, die sie zu meinem Projekt beitragen möchten? Gibt es noch andere Bereiche/Teile, die verbessert werden könnten?**

„Mir gefällt die Richtung. Mir scheint das Framework schon flexibel, aber ohne das da automatisch Sachen in EPLAN geladen werden oder zurückübertragen werden, macht es wenig Sinn…Kannst du das noch umsetzen?…Und hast du noch mehr Anwendungsbeispiele?…Ansonsten, wie gesagt, der Zeitaufwand, den bezahlt uns keiner.“

8. **Sehen Sie das Framework als vollständigen Ersatz für die bisherige mündliche Kommunikation?**
   „Nein, sollte es gar nicht. Es ist zwar super, so ne Plattform zu haben, aber oft spielen auch emotionale Entscheidungen eine Rolle, um Ideen zu finden oder Konflikte zu lösen…Hat ja auch was von Teambildung.“

A.12.9 Protokoll zur Zufriedenheit des Informatikers

1. **Ich habe Ihnen mein Projekt gezeigt. Wie denken Sie darüber? Hat es Ihre Erwartungen getroffen?**
   „Ich finde es cool, was du da aufgebaut hast. Ich habe mich schon etwas mit dem Maschinenbauer darüber unterhalten, bin aber, so wie du es mir gezeigt hast, anderer Meinung. Es integriert sich gut in meine bisherige Arbeitsweise, und ich bekomme all diese Informationen, die ich bisher mühsam erfragen musste…Ob es wirklich einsetzbar ist, kann ich dir so nicht sagen. Dazu muss ich damit arbeiten. Aber das mit den Schnittstellen direkt im Modell und das Switchen der Views finde ich toll…Ich glaube die anderen Stakeholder sind eher das Problem. Es dauert sicher etwas, sich einzuarbeiten…Ich muss zugeben, ich hatte anfangs eher geringe Erwartungen, aber ich bin zufrieden.“

2. **Welche Vorteile sehen Sie nun in dieser engen Verbindung zwischen den Modellen? Wie würde sich das Framework auf Ihre Zusammenarbeit im Team auswirken?**
   „Direkt zu wissen, wie die Hardwareschnittstellen aussehen, ist super. Durch die Verknüpfung mit dem Schaltplan sind mir meine Verantwortlichkeiten und wichtigen Teile viel klarer. Ich denke auch, wenn jetzt Änderungen in den anderen Modellen gemacht werden, sehe ich das direkt und kann die Änderungen bei mir implementieren…Eventuell kann das die Teamarbeit beschleunigen, aber das müsste sich noch zeigen. Es minimiert auf jeden Fall Missverständnisse im gesamten Team.“

3. **Welche Verbesserungsmöglichkeiten oder Herausforderungen sehen Sie bei der Anwendung des Frameworks?**
   „Ehrlich gesagt, sehe ich in dem Framework großes Potenzial und bisher nur wenige Probleme für mich…Wenn du aber von Verbesserungsmöglichkeiten sprichst, wäre ich für

die Integration eines Transformators oder Codegenerators zu TwinSAFE. Würde bestimmt die praktische Anwendung des Frameworks vereinfachen. Abgesehen davon sehe ich meine Sichtweise schon gut vertreten.“

4. **Es wurde damals angegeben, dass es wichtig ist, dass das Framework ebenfalls bei der Kommunikation mit Kunden helfen soll. Trifft dies Ihrer Meinung nach zu?**
   „Ich weiß nicht, eher nicht…Als ob sich bisher einer meine Software angeschaut hat…Die meisten Kunden wollen nur, dass es funktioniert…Großartig irgendwelche Modelle anschauen machen die eh nicht…Aber für mich ist das Ganze halt schon sinnvoll…Vielleicht gibt es auch Kunden, die das wollen…Wenn du nur von der Möglichkeit sprichst, dann ja…Wer will nicht alles auf einen Blick haben…“

5. **Es wurde damals angegeben, dass die Ressourcen zur Einführung des Frameworks zu knapp sind. Sehen Sie das immer noch so, sollte das Framework außerhalb dieses Forschungsprojektes breitflächig in der Firma eingesetzt werden?**
   „…Was die Ressourcen betrifft, so sehe ich die Vorteile eher langfristig. Initial mag es aufwendig sein, aber die Vorteile überwiegen deutlich…Auch sehe ich bei dir noch Löcher. Ohne Übersetzer zur Anbindung unserer Werkzeuge finde ich auch einen größeren Mehraufwand als verbesserte Effizienz.“

6. **Könnte das Framework ebenso bei der Beschaffung von Materialien, also dem Einkauf von Maschinenteilen helfen?**
   „Ich sehe Potenzial spezifische Teile und Materialien zu identifizieren, aber ich denke der Einkauf hat da schon seine eigenen Tools…Aber so richtig kann ich dir die Frage nicht beantworten.“

7. **Haben Sie noch weitere Anmerkungen oder Nachteile, die sie zu meinem Projekt beitragen möchten? Gibt es noch andere Bereiche/Teile, die verbessert werden könnten?**
   „…Vielleicht könnte man eine Art vereinfachte Einführung oder eine intuitive Benutzeroberfläche entwickeln, die den Einstieg erleichtert…Übersetzer zu unseren Tools könnte ich mir selbst entwickeln, wenn ich die Zeit bekomme. Sehe ich also als geringere Herausforderung.“

8. **Sehen Sie das Framework als vollständigen Ersatz für die bisherige mündliche Kommunikation?**
   „Auch, wenn das Framework die Kommunikation im Team und bestimmt auch nach außen verbessern kann, haben direkte Gespräche ein direktes Feedback. Diese Nuancen sehe ich durch eine reine digitale Kommunikation nicht…Es ist aber vollständig genug, um hier

Missverständnisse zu vermeiden und unsere Meetings zu ergänzen…Mir gefällt dieser diversitäre Ansatz.“

A.12.10 Protokolle zur Zufriedenheit des Sicherheitsingenieurs

1. **Ich habe Ihnen mein Projekt gezeigt. Wie denken Sie darüber? Hat es Ihre Erwartungen getroffen?**
   „Das vorgestellte Projekt hat mich überzeugt. Ich sehe erstmal alles, was ich für die Analyse und Beurteilung brauche…Alles ist verknüpft, auch, wenn ich das noch nicht ganz durchblicke, scheint mir das ganze ziemlich cool zu sein. Ich mag zwar meine tabellarische Form, aber ich denke, so kann man es bestimmt auch gut umsetzen…Wenn man sich aber erstmal damit auskennt, kann ich mir vorstellen, dass es die Entwicklung effizienter machen kann, vor allem in der Koordination im Team. Die direkte Kopplung meiner Analysewerkzeuge mit dem Framework erspart mir viel manuelle Übertragungsarbeit und macht den gesamten Prozess reibungsloser…Diese, wie nennst du sie nochmal, Anwendungsprofile scheinen mir das Ganze gut zu integrieren. Diese Drop-Down-Menus machen die Einstellungen scheinbar sehr einfach…Und dass dann die dynamischen Fehlerbäume und Blockdiagramme genauso aussehen, wie ich es kenne, und das ganze intern überprüft wird, wie hieß das nochmal, OCL und CSS oder so, macht die Gewöhnung definitiv einfacher. Kannst du so ein Anwendungsprofil auch für Risikographen entwickeln?“

2. **Welche Vorteile sehen Sie nun in dieser engen Verbindung zwischen den Modellen? Wie würde sich das Framework auf Ihre Zusammenarbeit im Team auswirken?**
   „Alles in einem Werkzeug finde ich erstmal klasse…Du hast ja die Standards ISO 13849 und ISO 12100 umgesetzt. Das ist für den Anfang gut. Wenn ich es richtig verstehe, kann ich bereits entwickelte Analysebausteine bereits wiederverwenden…Das ist ein enormer Gewinn. So ein Katalog macht uns dann das Leben mit der Zeit einfacher, vor allem, wenn es um die standardisierten Vorgehensweisen nach der Norm geht, scheint das effizient zu sein. Bezüglich der Teamarbeit hat das Framework bestimmt einen positiven Effekt. Allein die einfachere Kombination und Koordination bei der Identifizierung und Bewertung von Gefährdungen scheint klasse. Ich denke nicht, dass ich das alles brauch. Ich bin da eher traditionell veranlagt, aber ich sehe Potenzial.“

3. **Welche Verbesserungsmöglichkeiten oder Herausforderungen sehen Sie bei der Anwendung des Frameworks?**

„Eine Erweiterung um weitere relevante Normen umzusetzen. Dann bekommt man, so wie ich es verstehe, bestimmt mehr Möglichkeiten, verschiedene Risikobewertungsmethoden zu erstellen…Würde das Ganze noch verbessern und wäre dann noch ähnlicher zu WEKA.“

4. **Es wurde damals angegeben, dass es wichtig ist, dass das Framework ebenfalls bei der Kommunikation mit Kunden helfen soll. Trifft dies Ihrer Meinung nach zu?**
   „Ich finde das super…Es ermöglicht eine klarere Kommunikation der Risikobewertung gegenüber Kunden…Vielleicht sogar besser als die tabellarische Form. Das zeigt nicht nur unsere Kompetenz, sondern erleichtert auch die Abstimmung von Sicherheitsmaßnahmen…“
5. **Es wurde damals angegeben, dass die Ressourcen zur Einführung des Frameworks zu knapp sind. Sehen Sie das immer noch so, sollte das Framework außerhalb dieses Forschungsprojektes breitflächig in der Firma eingesetzt werden?**
   Keine definierbare Antwort
6. **Könnte das Framework ebenso bei der Beschaffung von Materialien, also dem Einkauf von Maschinenteilen helfen?**
   Keine definierbare Antwort
7. **Haben Sie noch weitere Anmerkungen oder Nachteile, die sie zu meinem Projekt beitragen möchten? Gibt es noch andere Bereiche/Teile, die verbessert werden könnten?**
   „Ich respektiere die Anstrengungen und den Aufwand, den du in dieses Projekt gesteckt hast. Meine Werkzeuge sind angebunden. Was will ich mehr…Die gerätespezifischen Sicherheitsstandards und –richtlinien haben wir ja schon drüber gesprochen.“
8. **Sehen Sie das Framework als vollständigen Ersatz für die bisherige mündliche Kommunikation?**
   Das Framework ist doch eher eine Ergänzung, als ein Ersatz für die mündliche Kommunikation. Besonders bei der Bewertung von Sicherheitsrisiken, Gefährdungssituationen, -ereignissen und -auswirkungen ist der direkte Dialog unerlässlich…Das Framework erleichtert zwar die Dokumentation und den Austausch spezifischer Informationen, aber zur Interpretation von Daten oder die Entwicklung der Lösungen brauch ich Meetings…Aber auf jeden Fall macht dein Ansatz das ganze bis zu einem gewissen Punkt effektiver.“