

**Forschungsbericht
Institut für Automatisierungstechnik
und Softwaresysteme**

Hrsg.: Prof. Dr.-Ing. Dr. h. c. Michael Weyrich

Behrang Ashtari Talkhestani

Methodik zur Synchronisierung der Modelle des Digitalen Zwillings automatisierter Systeme

Band 1/2020

Universität Stuttgart

Methodik zur Synchronisierung der Modelle des Digitalen Zwillings automatisierter Systeme

Von der Graduate School of Excellence
advanced Manufacturing Engineering der Universität Stuttgart
zur Erlangung der Würde eines Doktor-Ingenieurs (Dr.-Ing.)
genehmigte Abhandlung

Vorgelegt von
Behrang Ashtari Talkhestani
aus Borojerd (Iran)

Hauptberichter:	Prof. Dr.-Ing. Dr. h. c. Michael Weyrich
Mitberichter:	Prof. Dr.-Ing. Thomas Bauernhansl
Tag der Einreichung:	03.03.2020
Tag der mündlichen Prüfung:	30.07.2020

Institut für Automatisierungstechnik und Softwaresysteme
der Universität Stuttgart

2020

IAS-Forschungsberichte

Band 1/2020

Behrang Ashtari Talkhestani

**Methodik zur Synchronisierung der Modelle des
Digitalen Zwillings automatisierter Systeme**

D 93 (Diss. Universität Stuttgart)

Shaker Verlag
Düren 2020

Bibliografische Information der Deutschen Nationalbibliothek

Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über <http://dnb.d-nb.de> abrufbar.

Zugl.: Stuttgart, Univ., Diss., 2020

Copyright Shaker Verlag 2020

Alle Rechte, auch das des auszugsweisen Nachdruckes, der auszugsweisen oder vollständigen Wiedergabe, der Speicherung in Datenverarbeitungsanlagen und der Übersetzung, vorbehalten.

Printed in Germany.

ISBN 978-3-8440-7572-4

ISSN 1610-4781

Shaker Verlag GmbH • Am Langen Graben 15a • 52353 Düren

Telefon: 02421 / 99 0 11 - 0 • Telefax: 02421 / 99 0 11 - 9

Internet: www.shaker.de • E-Mail: info@shaker.de

Die vorliegende Arbeit entstand während meiner Tätigkeit als Doktorand der Graduate School of Excellence advanced Manufacturing Engineering (GSaME) der Universität Stuttgart am Institut für Automatisierungstechnik und Softwaresysteme (IAS) und einem Unternehmen der Automatisierungstechnik.

Mein besonderer Dank gilt meinem Doktorvater und Leiter des Instituts, Herrn Prof. Dr.-Ing. Dr. h. c. Michael Weyrich, für die Übernahme des Hauptberichts, die vielen konstruktiven Diskussionen und wertvollen Hinweise und Anregungen bei der Erstellung dieser Arbeit.

Herr Prof. Dr.-Ing. Thomas Bauernhansl danke ich für das Interesse an meiner Arbeit und die Übernahme des Mitberichts. Ein Dank gilt auch Herrn Prof. Dr.-Ing. Oliver Riedel für die Mitarbeit im Thesis-Committee und die fachlichen Anregungen.

Ohne die Unterstützung des Unternehmens wäre diese Arbeit nicht möglich gewesen. Daher möchte ich mich in Besonderem Maße bei Herrn Dr.-Ing. Wolfgang Schlögl bedanken, welcher mich mit der Aufgabenstellung betraute, stets förderte und mir sämtliche Türen öffnete um das Thema voranzubringen. Allen anderen, die in diesem Umfeld mit mir zusammenarbeiteten und wertvolle Diskussionspartner darstellten, gilt ebenfalls ein herzliches Dankeschön.

Zudem gilt mein Dank dem wissenschaftlichen Umfeld der GSaME, welche mir viele Blicke über den Tellerrand hinaus ermöglichte und mir ein „zweites Zuhause“ bot. Stellvertretend für alle Wegbegleiter möchte ich mich hier bei Herrn Hans-Friedrich Jacobi bedanken.

Ein besonderer Dank gilt Herrn Dr.-Ing. Nasser Jazdi für seine fortwährende und bedingungslose Unterstützung während der Arbeit, für die gemeinsamen Veröffentlichungen sowie die weisen Ratschlägen, die diese Promotion erst möglich gemacht haben. Des Weiteren gilt mein herzlicher Dank allen Kolleginnen und Kollegen am IAS für die gute Zusammenarbeit und die vielen guten und konstruktiven Diskussionen. Ein ebenfalls gebührender Dank gilt den Studierenden, die im Rahmen ihrer Master-, Studien-, Forschungs- und Bachelorarbeiten einen Beitrag zum Gelingen dieser Arbeit geleistet haben.

Schließlich bin ich meiner Familie, insbesondere meiner Frau Samira Maleki-Pilevar, von Herzen dankbar für ihre bedingungslose Unterstützung und ihr geduldiges Verständnis. Ihre Liebe und Zuversicht gaben mir die notwendige Kraft, die vorliegende Arbeit fertig zu stellen.

Stuttgart, im August 2020

Behrang Ashtari Talkhestani

Inhaltsverzeichnis

Abbildungsverzeichnis.....	iii
Tabellenverzeichnis.....	v
Abkürzungsverzeichnis.....	vi
Begriffsverzeichnis	viii
Zusammenfassung.....	x
Abstract.....	xi
1 Einleitung.....	1
1.1 Mehrwerte des Digitalen Zwillings während des gesamten Lebenszyklus	1
1.2 Herausforderungen bei der Bereitstellung des Digitalen Zwillings	3
1.3 Forschungsanforderungen.....	4
1.4 Zielsetzung der Arbeit	5
1.5 Abgrenzung zu ähnlichen Themenstellungen.....	6
1.6 Aufbau der Arbeit	7
2 Grundlagen automatisierter Systeme.....	10
2.1 Grundstruktur automatisierter Systeme	10
2.1.1 SPS-gesteuerte automatisierte Systeme	12
2.1.2 Entwicklungsvorgehen bei automatisierter Systeme	12
2.2 Domänenübergreifende Zusammenarbeit während des Engineering-Prozesses automatisierter Systeme.....	15
2.3 Ansätze zur Modelldurchgängigkeit und Verknüpfung von Tools	18
2.3.1 Proprietäre Punkt-zu-Punkt-Schnittstellen zwischen Tools	18
2.3.2 Standardisierte neutrale Austauschformate	19
2.3.3 Strukturierte Daten.....	21
2.3.4 Semantische Technologien zur Modellierung der Daten	21
2.4 Zusammenfassung	22
3 Stand der Wissenschaft und Technik.....	23
3.1 Digitaler Zwilling	23
3.1.1 Digitaler Zwilling im cyber-physischen Produktionssystem	25
3.1.2 Architektur des Digitalen Zwillings	26
3.2 Ansätze zur Integration der domänenübergreifenden Modelle	30
3.2.1 AutomationML-basierte Ansätze.....	31
3.2.2 SysML-basierte Ansätze	33
3.2.3 Ontologie-basierte Ansätze.....	35
3.2.4 Toolset-basierte Ansätze.....	36
3.3 Ansätze zur Synchronisierung der Modelle eines automatisierten Systems	37
3.3.1 3D-Laserscan-basierte Ansätze.....	37

3.3.2	Netzwerkscan-basierte Ansätze	39
3.3.3	Kommunikationsnetzwerkscan-basierte Ansätze	40
3.3.4	Änderungsdokumentationsanalyse-basierte Ansätze.....	41
3.4	Ansätze zur Modellgenerierung aus einem bestehenden automatisierten System	42
3.4.1	Generierung eines 3D-CAD-Modells	42
3.4.2	Generierung von Systemtopologien.....	45
3.4.3	Generierung eines Simulationsmodells	45
3.5	Zusammenfassende Bewertung und Schlussfolgerungen für die Konzeption	46
4	Lösungsansatz zur domänenübergreifenden Synchronisierung der Modelle des Digitalen Zwillings	51
4.1	Grundlegende Konzeptentscheidungen und Ansatz	51
4.1.1	Ankerpunkte innerhalb des Digitalen Zwillings	53
4.1.2	Voraussetzungen zur Synchronisierung der Ankerpunkte.....	57
4.2	Ankerpunktmethode zur Synchronisierung der Modelle des Digitalen Zwillings	61
4.2.1	Gesamtprozessablauf der Ankerpunktmethode	62
4.2.2	Phase 1: Änderungsdetektion.....	65
4.2.3	Phase 2: Domänenübergreifenden Abhängigkeitsdetektion	70
4.2.4	Phase 3: Modellanpassung	77
5	Realisierung der Ankerpunktmethode mithilfe eines Assistenzsystems	87
5.1	Gesamtüberblick über Softwarekomponenten des Assistenzsystems	87
5.2	Komponente „Grafische Benutzeroberfläche“	89
5.3	Komponente „Wrapper“	90
5.4	Komponente „Metamodell der SPS-Steuerungssoftware“	92
5.5	Komponente „Ankerpunktmodellierung und Analyse“.....	94
5.6	Komponente „Change-Request-Modell-Generierung“	97
6	Evaluierung.....	101
6.1	Szenario 1: Modulares Produktionssystem	101
6.1.1	Digitaler Zwilling des modularen Produktionssystems	104
6.1.2	Bewertung der Evaluierungsergebnisse.....	106
6.2	Szenario 2: Intelligentes Lager in der flexiblen Produktionsanlage.....	108
6.2.1	Digitaler Zwilling des intelligenten Lagers	110
6.2.2	Bewertung der Evaluierungsergebnisse.....	112
6.3	Erfüllung der Forschungsanforderungen und Zielsetzung der Arbeit	115
7	Schlussbetrachtung	117
7.1	Zusammenfassung der Ergebnisse und Bewertung	117
7.2	Ausblick.....	118
	Literaturverzeichnis.....	120

Abbildungsverzeichnis

Abbildung 1: Erhöhung der Verfügbarkeit eines automatisierten Systems durch Nutzung des Digitalen Zwillings	2
Abbildung 2: Aufbau der Arbeit	8
Abbildung 3: Grundstruktur mechatronischer Systeme nach der VDI-Richtlinie 2206 [53]	11
Abbildung 4: Das V-Modell für mechatronische Entwicklungsvorgehen nach [53]	13
Abbildung 5: Engineering-Vorgehensmodell und dessen domänenübergreifende Aktivitäten nach [61]	14
Abbildung 6: Daten und Informationsaustausch während des Engineerings eines automatisierten Systems nach [46]	16
Abbildung 7: Tools während des funktionalen Engineerings in Anlehnung an [61]	17
Abbildung 8: Modellaustausch zwischen Tools über proprietäre Punkt-zu-Punkt-Schnittstellen im Vergleich zum Modellaustausch über zentrale Austauschformate	19
Abbildung 9: Grundstruktur von AutomationML nach [165]	20
Abbildung 10: Der Digitale Zwilling und seine Eigenschaften im cyber-physischen Produktionssystem nach [6]	26
Abbildung 11: Architektur des Digitalen Zwillings in Anlehnung an [6]	28
Abbildung 12: Integriertes Engineering mit der AML-Integrationsplattform, AutomationML Hub nach [139]	31
Abbildung 13: Semantischer Modellierungsansatzes mit System- und Charakteristik-Metamodell nach [107]	32
Abbildung 14: Integrierter fachübergreifende Engineering-Workflow nach [104]	34
Abbildung 15: Prozessablauf des CAD-Modellerkennungsprozesses nach [124]	38
Abbildung 16: Prozessüberwachungsmethode nach [29]	40
Abbildung 17: Methode zur Synchronisierung des elektrischen schematischen Modells des Systems nach [128]	41
Abbildung 18: Reverse-Engineering-Prozess nach [131]	44
Abbildung 19: Digitaler Zwilling eines automatisierten Systems und dessen Modelle innerhalb des Digitalen Zwillings	52
Abbildung 20: Granularität des Assets innerhalb der Betriebsmittel in einem automatisierten System in Anlehnung an [166] und [140]	54
Abbildung 21: Ankerpunkte einer mechatronischen Komponente innerhalb des Digitalen Zwillings eines automatisierten Systems	56
Abbildung 22: Kapselung von domänenübergreifenden wiederverwendbaren Modellen eines Assets innerhalb einer Komponentenbibliothek	58
Abbildung 23: Signalname eines Assets gemäß der Namenskonvention	60
Abbildung 24: Prozessschritte der Ankerpunktmethode	63
Abbildung 25: Änderungsdetektion anhand des Formalisierungsprozesses der SPS-Steuerungssoftware in der Ankerpunktmethode	68
Abbildung 26: Abstraktionsmodell der formalen Beschreibung der SPS-Steuerungssoftware ..	70
Abbildung 27: Vorgehen zur Erkennung des Klassifizierungslabels "Neue Funktionsgruppe wurde dem System hinzugefügt" im Entscheidungsbaum	76
Abbildung 28: Prozessablauf des Lösungsansatzes basierend auf dem Überschreiben der Modelle	79
Abbildung 29: Prozessablauf des Lösungsansatzes basierend auf der Basismodell-Versionierung	81
Abbildung 30: Prozessablauf des Lösungsansatzes basierend auf Versionierung des Gesamtsystemmodells	81

Abbildung 31: Referenzierung der Ankerpunkte des Gesamtsystemmodells auf das Basismodell nach Versionierungsregeln	82
Abbildung 32: Vereinfachte Darstellung des Prozessverlaufs des Engineering-Change-Managements	83
Abbildung 33: Prozessablauf der Modellanpassung in der Ankerpunktmethode basierend auf der Erstellung generischer Change-Request-Modelle	85
Abbildung 34: Gesamtübersicht über die Softwarekomponenten des Assistenzsystems	88
Abbildung 35: Grafische Benutzeroberfläche des Assistenzsystems	90
Abbildung 36: Metamodellierungsansatz zur Erstellung des Metamodells einer SPS-Steuerungssoftware	93
Abbildung 37: Java-Klassen der Komponente "Ankerpunktmodellierung und Analyse"	95
Abbildung 38: Java-Klassen der Komponente "Change-Request-Modell-Generierung"	99
Abbildung 39: Change-Request-Modell innerhalb von Teamcenter-PLM-Plattform	100
Abbildung 40: Modulares Produktionssystem und dessen Assets	102
Abbildung 41: Engineering-Tools und erstellter Digitaler Zwilling vom modularen Produktionssystem	105
Abbildung 42: Evaluierungsszenario anhand des modularen Produktionssystems	106
Abbildung 43: intelligentes Lager in der flexiblen Produktionsanlage	110
Abbildung 44: Evaluierungsszenario anhand des iLagers in der flexiblen Produktionsanlage	111
Abbildung 45: Abfolge der Prozessschritte und die durchschnittliche Zeitdauer für die einzelnen Schritte der Rekonfiguration ohne und mit Verwendung des Digitalen Zwillings	113
Abbildung 46: Zeitdauer- und Prozessvergleich der beiden Rekonfigurationsprozesse mit und ohne synchronisierten Digitalen Zwilling mittels der Ankerpunktmethode	114

Tabellenverzeichnis

Tabelle 1: Zusammenfassung und Bewertung der Forschungsansätze zur Integration domänenübergreifender Modelle des Digitalen Zwillings	47
Tabelle 2: Zusammenfassung und Bewertung der bestehenden Ansätze zur Synchronisierung von Modellen eines bestehenden automatisierten Systems	48
Tabelle 3: Zusammenfassung und Bewertung der Ansätze zur Modellgenerierung aus einem bestehenden automatisierten System	49
Tabelle 4: Die Regeltabelle zur domänenübergreifenden Änderungsdetektion aus SPS- Steuerungssoftware	74
Tabelle 5: Bearbeitungsstationen sowie die Steuerungs- und Laststromversorgungs- Komponenten des MPS	103
Tabelle 6: Tools und erstellte Modelle innerhalb des digitalen Zwillings des MPS	104
Tabelle 7: Durchführung der Synchronisierung des Digitalen Zwillings mit dem Assistenzsystem	107
Tabelle 8: Bearbeitungsstationen sowie die Steuerungs- und Laststromversorgung-Komponenten des iLagers	109
Tabelle 9: Tools und die erstellte Modelle innerhalb des Digitalen Zwillings des iLagers	111
Tabelle 10: Quantitative und qualitative Bewertung der Evaluierungsergebnisse	115

Abkürzungsverzeichnis

API	Application Programming Interface
ARENA2036	Active Research Environment for the Next Generation of Automobiles 2036
AutomationML	Automation Markup Language
BoM	Bill of Material
BoP	Bill of Process
CAD	Computer-Aided Design
CPS	Cyber-Physical System
CPU	Central Processing Unit
CRM	Change Request Model
DI	Digital Input
DIN	Deutsches Institut für Normung
DO	Digitale Output
DZ	Digitaler Zwilling
E/A	Ein-/Ausgabe
ECAD	Electronic and Electrical Computer-Aided Design
EMF	Eclipse Modeling Framework
FUP	Funktionsplan
GUI	Graphical User Interface
IAS	Institut für Automatisierungstechnik und Softwaresysteme
ID	Identifikator
iLager	intelligentes Lager
MPS	Modulares Produktionssystem
OPC UA	Open Connectivity Unified Architecture
OWL	Web Ontology Language
PLM	Product-Lifecycle-Management
PROFIBUS	Process Field Bus
Profinet	Process Field Network
RDF	Resource Description Framework

SOA	S ervice- O riented A rchitecture
SoP	S tart o f P roduction
SPS	S peicherprogrammierbare S teuerung
SQL	S tructured Q uery L anguage
STEP	S tandard for the E xchange of P roduct model data
SysML	S ystems M odeling L anguage
UML	U nified M odeling L anguage
VDI	V erein D eutscher I ngenieure
VPN	V irtual P rivate N etwork
W3C	W orld W ide W eb Consortium
WLAN	W ireless L ocal A rea N etwork
XML	E xtensible M arkup L anguage
3D	3 - D imensional
4GD	4 th G eneration D esign

Begriffsverzeichnis

Aktor: Einheit zur Umsetzung von Stellinformation tragenden Signalen geringer Leistung in leistungsbefähigte Signale einer zur Prozessbeeinflussung notwendigen Energieform [1].

Ankerpunkt: Komponentenmodelle eines Assets auf der Ebene mechatronischer Komponenten im Gesamtsystemmodell eines automatisierten Systems innerhalb der Domäne Mechanik, Elektrik und Software.

Architektur: Kombination von Elementen eines Modells aufbauend auf Prinzipien und Regeln zum Zweck seiner Konstruktion, Weiterentwicklung und Nutzung [2].

Asset: Gegenstand, der einen Wert für eine Organisation hat [3]. Ein Asset kann hierbei das gesamte automatisierte System oder eine Einheit aus vielen Sensoren und Aktoren bis hin zu einer Schraube im System sein.

Automatisiertes System: Technischen Einrichtungen, die für die Automatisierung des technischen Prozesses erforderlich sind [4].

Co-Simulation: Technik, die eine globale Simulation eines gekoppelten Systems über die Zusammensetzung von Simulatoren ermöglicht [5].

Cyber-physisches System: System, das reale (physische) Objekte und Prozesse verknüpft mit informationsverarbeitenden (virtuellen) Objekten und Prozessen über offene, teilweise globale und jederzeit miteinander verbundene Informationsnetze [3].

Digitaler Zwilling: Virtuelles Abbild eines physischen Assets in einem cyber-physischen System, das in der Lage ist, seine statischen und dynamischen Eigenschaften zu reflektieren. Ein Digitaler Zwilling enthält und mappt domänenübergreifenden Modelle eines physischen Assets [6].

Domäne: Abgrenzbarer Anwendungsbereich, in dem typische Technologien mit vergleichbaren Methoden für gleiche Interessengruppen eingesetzt werden [7].

Engineering-Prozess: Reihe von Arbeitsschritten, die von einem Ingenieur zur Erstellung einer spezifischen Kundenlösung durchgeführt werden, von der Angebots-, Analyse-, Entwurf- und Implementierungsphase bis hin zur Inbetriebnahme und Übergabe der Anlage an den Kunden.

Inkonsistenz: Fehler oder Widersprüche in Modellen. Inkonsistenzen können innerhalb eines einzelnen Modells oder zwischen mehreren Modellen auftreten [8].

Komplexität: Im Bereich der Informationstechnologie durch die Vielzahl und Heterogenität der Elemente und deren Veränderungsdynamik in einem System charakterisiert [9].

Mechatronische Komponente: Ein Asset, welches aus mechanischen, elektrischen und informationstechnischen Elementen besteht [10].

Metamodell: Beschreibt die Struktur eines Modells und beschreibt so den Modelltyp. Es definiert die verfügbaren Modellelemente und deren mögliche Verknüpfungen [8].

Modell: Schlüssige, ausreichend detaillierte Abstraktion von Aspekten in einem Anwendungsbereich [3].

Rekonfiguration: Umfasst die technische Sicht des Prozesses der Veränderung eines bereits entwickelten und operativ eingesetzten Systems, um es an neue Anforderungen anzupassen, Funktionalität zu erweitern, Fehler zu beseitigen oder die Qualitätseigenschaften zu verbessern [11]

Rekonfigurierbare Systeme: Systeme die so konzipiert sind, dass eine schnelle Strukturänderung möglich ist, um zugige Anpassung der Produktkapazitäten und Funktionalität gemäß geänderte Marktanforderungen durchzuführen [12].

Semantik: Beziehung zwischen Zeichen und ihrer Bedeutung. Zeichen können grafischer Natur (z. B. Verkehrszeichen) oder alphanumerische Einheiten (z. B. Wörter) sein, die Dinge benennen [13].

Sensor: Einheit zur Erfassung von physikalischen Prozessgrößen und deren Umwandlung in elektrische oder optische Messsignale mit dem Zweck der Verarbeitung dieser Messsignale durch einen Rechner [14].

Service orientierte Architektur: Paradigma für die Strukturierung und Nutzung verteilter Funktionalität, die von unterschiedlichen Besitzern verantwortet wird [15].

Speicherprogrammierbare Steuerung (SPS): Ein Industrie-Computersteuerungssystem, das den Zustand von Sensoren mittels Eingangssignalen kontinuierlich überwacht und Entscheidungen basierend auf einer kundenspezifischen Steuerungssoftware zur Steuerung des Zustands von Aktoren trifft.

Synchronisierung: Systematisches Verfahren für die Erkennung der Systemänderungen und ihrer relevanten Einflüsse sowie die Aktualisierung der Modelle.

Zusammenfassung

Die von den Märkten ausgehenden Anforderungen bezüglich des ständigen Herausbringens von marktgerechten Innovationen zwingt Industrieunternehmen, einerseits zur zunehmenden Verwendung hochautomatisierter Produktionssysteme und andererseits diese während ihres Lebenszyklus oft rekonfigurieren zu müssen. Dies erfordert, dass flexible automatisierte Systeme zur Verfügung stehen, die bei Bedarf effizient rekonfiguriert werden können.

Im Kontext der cyber-physischen Systeme kann der Digitale Zwilling eines automatisierten Systems diese Herausforderung angehen, um Systeme einfach und schnell zu rekonfigurieren. So können Rekonfigurations-Szenarien in einer simulativen Umgebung realisiert und getestet werden. Die reale Wiederinbetriebnahme des Fertigungssystems benötigt daher weniger Zeit und ermöglicht eine höhere Systemverfügbarkeit.

Voraussetzung für den Einsatz des Digitalen Zwillings eines automatisierten Systems ist allerdings, dass ein ständig aktuelles Anlagenmodell der mechatronischen Bestandteile einer realen Anlage während der verschiedenen Phasen ihres Lebenszyklus existiert. Daraus leitet sich das Ziel dieser Forschung ab, eine Methodik zu erstellen, die eine domänenübergreifende Synchronisierung der Modelle des Digitalen Zwillings mit einem realen, automatisierten System ermöglicht. Damit wird ein Beitrag zur Steigerung der Effizienz des gesamten Engineering-Prozesses ab der Inbetriebnahme geleistet.

Die in der vorliegenden Arbeit konzipierte Methodik erlaubt es, die Abweichungen zwischen den Modellen und der Realität während des Betriebs zu ermitteln. Im Mittelpunkt steht dabei die entwickelte Ankerpunktmethodik, die in einem Assistenzsystem mündet. Diese detektiert die Änderungen und deren domänenübergreifende Abhängigkeiten in der realen Welt mit Hilfe einer regelbasierten Analyse der aktuellen SPS-Steuerungssoftware eines automatisierten Systems. Die Ankerpunktmethodik selbst beinhaltet eine automatisierte Modellanpassung unter Berücksichtigung der Nachvollziehbarkeit von Änderungen am Gesamtsystemmodell. Dadurch wird die Aufrechterhaltung der Konsistenz zwischen den domänenübergreifenden Modellen des Digitalen Zwillings gewährleistet. Diese Methode kommt innerhalb des Assistenzsystems auf der Grundlage der automatisierten Generierung von Change-Request-Modellen für die betroffenen Komponentenmodelle des Digitalen Zwillings zum Einsatz.

Die Validierung der Methodik (Ankerpunktmethodik) erfolgt auf der Basis eines modularen Produktionssystems am Institut für Automatisierungstechnik und Softwaresysteme (IAS) und eines intelligenten Lagers in einer flexiblen Produktionsanlage auf dem Forschungscampus ARENA2036 und wird anhand der formulierten Forschungsanforderungen evaluiert.

Abstract

Nowadays, industry is increasingly faced with the challenge of delivering customer-specific products in ever shorter production times. Shortened product life cycles and increasing market demands require continuous improvement and reconfiguration of current manufacturing system. This requires flexible automated manufacturing systems that can be efficiently reconfigured as required.

In the context of Cyber-Physical Systems, a Digital Twin of a manufacturing system can address this challenge to reconfigure systems quickly and effortlessly by implementing and testing reconfiguration scenarios in a simulated environment. The real recommissioning of the automation system therefore requires less time and allows for higher system availability. However, a prerequisite for the usability of the Digital Twin of an automation system is the availability of an up-to-date plant model of the mechatronic components of the real plant during the different phases of its life cycle. Hence, the goal of the research is defined to create a systematic approach that enables a cross-domain synchronization of the models of the Digital Twin with a real automation system. This will contribute to increasing the efficiency of the engineering process of the system after commissioning.

The methodology proposed in this thesis enables the detection of deviations between the models and the reality of the system during operation. The main focus is the developed Anchor-Point-Method, which leads to an assistance system that detects the changes and their cross-domain dependencies in the real world by means of a rule-based analysis of the current PLC control software of an automated system. The Anchor-Point-Method itself includes an automated model adaptation considering the tracking of changes in the overall system model. This ensures that consistency is maintained between the digital twin's cross-domain models. This method is used within the assistance system based on the automated generation of change request models for the relevant component models of the Digital Twin.

The validation of the methodology (Anchor-Point-Method) is carried out on a modular production system at the Institute of Industrial Automation and Software Engineering (IAS) and an intelligent warehouse in a flexible production system on the research campus ARENA2036 and is evaluated according to the formulated research requirements

1 Einleitung

In diesem einleitenden Kapitel wird zunächst die Motivation für das in dieser Arbeit vorgestellte Forschungsthema erläutert und die aus der Motivation und Ausgangssituation abgeleiteten Herausforderungen, Forschungsanforderungen und Ziele beschrieben. Im Anschluss daran erfolgt im letzten Abschnitt dieses Kapitels die Skizzierung des thematischen Aufbaus der Arbeit.

1.1 Mehrwerte des Digitalen Zwillings während des gesamten Lebenszyklus

Die Produktion in Europa steht durch strukturelle und nachfrageorientierte Veränderungen in der Weltwirtschaft unter starkem Druck. Die gegenwärtigen Industriebranchen stehen zunehmend vor der Herausforderung, immer mehr kundenspezifische Produkte in kürzeren Produktionszeiten liefern zu müssen [16][17]. Dies erfordert ein flexibles automatisiertes System, das bei Bedarf effizient neu konfiguriert werden kann [18][19]. Die Automobilindustrie ist hier als Beispiel für die Verwendung von fortschrittlichen, automatisierten Systemen hinsichtlich diskreter Fertigung mit hohem Automatisierungsgrad zu nennen. Die Automobilhersteller nehmen heute an einem globalen Wettbewerb teil, der durch zunehmende Anpassungsanforderungen und die Notwendigkeit einer größeren Produktvielfalt gekennzeichnet ist [20]. Diese Unternehmen müssen sich derzeit den stagnierenden oder sogar abnehmenden Märkten stellen [21]. Aufgrund des daraus resultierenden verstärkten Wettbewerbs um überlebenswichtige Marktanteile sind die Automobilhersteller gezwungen, an einem Innovationsrennen teilzunehmen, das durch eine steigende Anzahl von Produktvarianten mit zahlreichen Produktderivaten gekennzeichnet ist. Darüber hinaus sinkt die Dauer der Innovations- und Modellzyklen stetig. Um diesen Anforderungen nachkommen zu können, wird eine erhöhte Rekonfigurierbarkeit der betreffenden automatisierten Systeme angestrebt [22]. Die Rekonfiguration umfasst dabei die technischen Prozessveränderungen eines bereits entwickelten und operativ eingesetzten Systems, um dieses an die neuen Anforderungen anzupassen, die Funktionalität zu erweitern, Fehler zu beseitigen oder die Qualitätseigenschaften zu verbessern [23][11].

Als Folge der auftretenden industriellen Herausforderungen sowie motiviert durch die Notwendigkeit, einer höheren Effizienz in der Produktion sicherzustellen, entstand das Konzept „Industrie 4.0“ [24][25]. Im Rahmen dieses Konzepts können automatisierte Systeme mit Konnektivitätsmerkmalen, unter Einsatz von Internet-Technologien und Kommunikationsschnittstellen [26], ausgestattet werden und so konzipiert und betrieben werden, dass sie von intelligenten Diensten profitieren können. Diese Dienste greifen in der Regel auf verfügbare Informationen des Produktionssystems zu und nutzen diese, um Hilfsfunktionen während des Betriebs bereitzustellen und die Systemleistung zu optimieren [27].

Hierbei ist es notwendig, Konzepte der Industrie 4.0 zu nutzen, um ein dynamisch rekonfigurierbares Produktionssystem zu erzielen, das zeitnah auf aktuelle Marktanforderungen und veränderte Anforderungen reagieren kann. Eines der Industrie 4.0-Konzepte ist der Digitale Zwilling [28]. Der Digitale Zwilling stellt ein virtuelles Abbild eines automatisierten Systems dar, das in der Lage ist, seine statischen und dynamischen Eigenschaften zu reflektieren [6]. Dabei kann ein Digitaler Zwilling für ein automatisiertes System die Herausforderung bewältigen, das System einfach und schnell rekonfigurierbar zu machen, indem es für die Umsetzung und den Test von Rekonfigurationsszenarien in einer simulativen Umgebung verwendet wird [29][30]. Dementsprechend können Prozessschritte, die zur Rekonfiguration einer bestehenden Fertigungsautomatisierung erforderlich sind, mithilfe eines Digitalen Zwillings beschrieben und getestet werden. Für diesen Anwendungsfall werden die neuen, produktbezogenen Prozessanforderungen an das Produktionssystem auf Basis der aktuell verfügbaren Ressourcen des Digitalen Zwillings untersucht, wobei verschiedene Prozessszenarien getestet werden. Hier können alle Steuereinheiten offline programmiert werden und für jedes Szenario wird die virtuelle Inbetriebnahme des Systems getestet. Auf diese Weise wird die Anlaufphase (Ramp-Up-Prozess) verkürzt und das System in kürzerer Zeit wieder in Betrieb genommen, was zu einer höheren Verfügbarkeit sowie einer höheren Zuverlässigkeit bei geringerem Ausfallrisiko führt [30]. Um diese Vorteile nutzen zu können, muss der Digitale Zwilling über den gesamten Lebenszyklus des automatisierten Systems breitstellen, vgl. Abbildung 1.

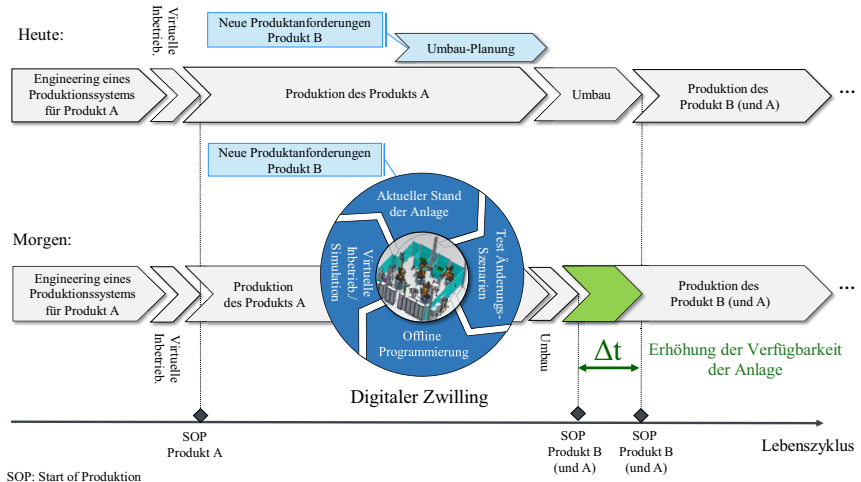


Abbildung 1: Erhöhung der Verfügbarkeit eines automatisierten Systems durch Nutzung des Digitalen Zwillings

1.2 Herausforderungen bei der Breitstellung des Digitalen Zwillings

Der Engineering-Prozess für ein automatisiertes System ist ein domänenübergreifender Prozess, bei dem Systemingenieure aus verschiedenen Ingenieurdomänen des Maschinenbaus, der Elektrotechnik und der Steuerungstechnik zusammenarbeiten müssen [31]. Hierbei handelt es sich meist um einen sequentiellen Prozess, der mit der Mechanik beginnt, mit dem Elektro- und Softwaredesign weitergeht und mit dem Testen in einer domänenübergreifenden Simulation in der digitalen Welt endet [32][30]. Im Kontext der Automobilindustrie beginnt dieser Engineering-Prozess etwa 1,5 Jahre vor dem Start der Serienproduktion [17]. Die dabei erzeugten bzw. verwendeten Daten und Modelle sowie deren Relationen während des Prozesses bilden zusammen den Digitalen Zwilling für die automatisierte Fertigungsanlage. Letztendlich wird, basierend auf diesem Digitalen Zwilling, die Anlage aufgebaut und in Betrieb genommen. Ab dem Start der Betriebsphase müssen oft mehrere Änderungen an der Anlage vorgenommen werden, beispielsweise hervorgerufen durch das Altern von Komponenten, das Einsetzen von Ersatzteilen, durch Optimierungen des Prozessablaufs sowie einer Reparatur [33]. Um diese Änderungen in der Fertigungsanlage zurückverfolgen und den Digitalen Zwilling mit dieser synchronisieren zu können, müssen drei wesentliche Herausforderungen bewältigt werden.

(H1) Unvollständige Dokumentation von Änderungen während des Betriebs sowie der Inbetriebnahme eines automatisierten Systems

Es liegt keine bzw. nur eine unvollständige Dokumentation von Änderungen an dem automatisierten System seit der Inbetriebnahme vor [29]. Da Änderungen im gesamten Produktionsprozess in kurzer Zeit in den realen Fertigungssystemen umgesetzt werden müssen [34], werden sie in der Praxis durch Experten der jeweiligen Domäne - Mechanik, Elektrik, Software - durchgeführt und meist unvollständig, widersprüchlich und auch teilweise inkorrekt dokumentiert [35]. Unzureichende Dokumentationen sind insbesondere typisch für Anlagen, die Stück für Stück über einen Zeitraum von vielen Jahren modernisiert worden sind [36].

(H2) Domänenübergreifende Abhängigkeiten der Änderungen

Die verdeckten Abhängigkeiten der Änderungen in den verschiedenen Domänen - Mechanik, Elektrik und Software - werden nicht erkannt und damit auch nicht beachtet [37]. Steigende Produktvielfalt und kurze Lebenszyklen führen zu einer zunehmenden Komplexität im Produktionsprozess [38], sodass komplexere und hoch automatisierte Fertigungssysteme gebaut werden. Die Komplexität wird im Bereich der Informationstechnologie durch die Vielzahl und Heterogenität der Elemente und deren Veränderungsdynamik in einem System charakterisiert [9]. Diese Komplexität steht ebenso im Zusammenhang mit dem zunehmenden Einsatz und der Verknüpfung von Mechanik und Elektronik sowie Software in Form von mechatronischen Anlagenkomponenten. Die Änderungen, die kurzfristig an einer Fertigungsanlage durchgeführt

werden müssen, werden in der Praxis durch Experten der jeweiligen Domänen mit unterschiedlicher beruflicher Qualifikation, technischem Sprachverständnis und Engineering-Tools durchgeführt [32]. Vor diesem Hintergrund ist es sehr schwierig, die domänenübergreifenden Einflüsse der Änderungen und Abhängigkeiten in der realen Welt zu erkennen.

(H3) Heterogenität und hohe Komplexität der Modelle des Digitalen Zwillings

Derzeit sind Heterogenität und hohe Komplexität zwischen den Modellen und deren Relationen im Digitalen Zwilling zu beobachten [39], [40].

Darüber hinaus kann in Großanlagen jede Komponente selbst von verschiedenen Ingenieuren in deren jeweiligen Fachdomänen unterschiedlich modelliert werden [41]. Mit zunehmender Komplexität der zu entwerfenden Objekte sowie der ständig wachsenden Zahl an beteiligten Domänen und Personen während des Engineering-Prozesses hinsichtlich einer automatisierten Fertigungsanlage steigt der Bedarf an Konsistenz zwischen den entstehenden Modellen, die von Ingenieuren mit unterschiedlichen Fachkenntnissen und Tools entwickelt werden [42]. Zusammenfassend ist eine manuelle Anpassung der Modelle des Digitalen Zwillings vorzunehmen, was fachübergreifendes Wissen und einen hohen Aufwand erfordern.

1.3 Forschungsanforderungen

Im Folgenden werden die Forschungsanforderungen ausgehend von den vorgenannten Herausforderungen bei der Synchronisierung der domänenübergreifenden Modelle des Digitalen Zwillings identifiziert und formuliert. Im Rahmen dieser Arbeit muss ein Konzept zum Erfüllen der jeweiligen Forschungsanforderungen entworfen und realisiert werden.

Im Hinblick auf die erste Herausforderung, unvollständige Dokumentation von Änderungen während des Betriebs sowie der Inbetriebnahme eines automatisierten Systems, ergibt sich folgende Anforderung:

(A1) Detektion der Änderungen bis zur Feld-Ebene in automatisierten Systemen

Die fehlenden Informationen hinsichtlich der Änderungen in der realen Welt bedingen die Entwicklung einer Methode, die die sich einstellenden Änderungsszenarien in der realen Welt automatisiert detektieren kann. Diese Methode muss die Änderungen bis zur Feld-Ebene (Ein-/Ausgangsebene und Sensor-/Aktorebene) in Fertigungssystemen identifizieren, weil eine domänenübergreifende Simulation mittels Digitalem Zwilling ohne eindeutige Informationen von den aktuellen Sensoren und Aktoren in der Anlage nicht möglich ist.

Aus der zweiten Herausforderung, den domänenübergreifenden Abhängigkeiten der Änderungen im automatisierten System, lässt sich die folgende Herausforderung definieren:

(A2) Detektion der domänenübergreifenden Abhängigkeiten der Änderungen

Die zu entwickelnde Methode muss die verdeckten Abhängigkeiten von Änderungen in verschiedenen Domänen erkennen, um die automatisierte Übertragung der domänenübergreifenden Abhängigkeiten von Änderungen innerhalb des gesamten Systems zu ermöglichen. Darüber hinaus muss diese Methode eine einfache Anpassungsfähigkeit an unterschiedlichen, diskreten automatisierten Systemen mit verschiedenen Charakteristika im Hinblick auf die domänenübergreifende Abhängigkeitsanalyse zulassen.

Schließlich lässt sich unter Berücksichtigung der dritten Herausforderung, nämlich der Heterogenität und hohen Komplexität der Modelle des Digitalen Zwillings, die folgende Anforderung ableiten:

(A3) Anpassung der domänenübergreifenden Modelle des Digitalen Zwillings

Aufgrund der hohen Komplexität und Heterogenität der Modelle des Digitalen Zwillings und der erforderlichen domänenübergreifenden Fachkenntnis in verschiedenen Domänen zur Modellanpassung besteht der Bedarf an einer Methode zur automatisierten Anpassung der domänenübergreifenden Modelle des Digitalen Zwillings. Darüber hinaus muss diese Methode mit den vorhandenen Technologien zur Integration von domänenübergreifenden Modellen des Digitalen Zwillings während des Engineering-Prozesses verträglich sein.

1.4 Zielsetzung der Arbeit

Die Problematik des hohen manuellen Aufwands bei der Synchronisierung der Modelle des Digitalen Zwillings für ein automatisiertes System ab der Inbetriebnahme bildet den Ausgangspunkt dieser Arbeit.

Die zu konzipierenden Methoden sollen letztendlich einen Beitrag zur Steigerung der Effizienz des Engineering-Prozesses ab der Inbetriebnahme leisten. Die zu integrierenden Methoden sollen eine Methodik ergeben, die eine domänenübergreifende Synchronisierung der Modelle des Digitalen Zwillings mit einem realen automatisierten System automatisiert ermöglicht. Dadurch kann die Verlagerung des menschlichen Arbeitsaufwands auf kreative, wertschöpfende Tätigkeit erreicht werden.

Innerhalb dieses Zielrahmens wird folgenden Aspekten ein besonderer Stellenwert beigemessen:

(Z1) Aktive Unterstützung der Ingenieure bei der Rekonfiguration eines automatisierten Systems

Das in dieser Arbeit entstehende Lösungskonzept soll eine Reduktion des manuellen Aufwands bei der Synchronisierung des Digitalen Zwillings für das bestehende automatisierte System im Rahmen der Rekonfiguration des Systems ermöglichen. Dazu muss eine automatisierte Methodik

zur Änderungsdetektion sowie zur Analyse der domänenübergreifenden Abhängigkeiten zwischen Komponenten des Systems und den notwendigen Änderungen an den domänenübergreifenden Modellen des Digitalen Zwillings entwickelt werden. So können die Ingenieure bei der Rekonfiguration unterstützt werden. Die zu entwickelnden Konzepte sollen besonders die Bedürfnisse des Ingenieurs berücksichtigen.

(Z2) Verträglichkeit mit den bestehenden Technologien zur Erstellung eines Digitalen Zwillings

Jedes neuartige Konzept, das konsequent die Abkehr von bisherigen Techniken und Methoden erfordert, wird zwangsläufig anfangs mit Akzeptanzschwierigkeiten zu kämpfen haben. Aus diesem Grund sollen in der vorliegenden Ausarbeitung die bestehenden Technologien zur Erstellung des Digitalen Zwillings eines automatisierten Systems in der Industrie weitgehend berücksichtigt und in die vorgeschlagene neue Methodik möglichst nahtlos integriert werden.

Die vorliegende Arbeit grenzt an weitere wichtigen Themenbereiche, die aber bewusst nicht weiter vertieft werden. Diese verwandten Bereiche werden im Folgenden kurz genannt und aufgeführt.

1.5 Abgrenzung zu ähnlichen Themenstellungen

Im Rahmen dieser Arbeit wird das Konzept der Synchronisierung von Modellen eines automatisierten Systems in der diskreten Fertigung ab dem Zeitpunkt der Inbetriebnahme des Systems vorgestellt. Hierbei müssen folgende Randbedingungen geklärt werden, um die Abgrenzung zu erleichtern.

Automatisierte Systeme in der diskreten Fertigung

Automatisierungssysteme weisen viele Merkmale auf, in denen sie sich unterscheiden können. Auf dem Gebiet der Automatisierungstechnik können Automatisierungssysteme in die zwei Hauptgebiete Anlagenautomatisierung und Produktautomatisierung unterteilt werden. Die Produktautomatisierung umfasst Automatisierungssysteme, in denen der technische Prozess in einem Gerät oder einer einzelnen Maschine abläuft [43]. Zur Produktautomatisierung zählen beispielsweise die Automatisierungssysteme wie Smartphones, intelligente Heimgeräte usw. Demgegenüber lassen sich die Prozessstypen in der Anlagenautomatisierung in kontinuierliche- und diskrete Fertigung sowie Chargen bzw. Batchproduktion gliedern [43]. Die Produktion von Chemikalien kann beispielsweise als eine kontinuierliche Produktion betrachtet werden. Hierbei wird das „Produkt“ ununterbrochen in einer durchgehenden Ausgangsmenge hergestellt. In der diskreten Fertigung werden die Produkte jeweils einzeln verarbeitet, und die Produkte sind dabei immer zählbar. Hierbei umfasst der Begriff *Fertigung* alle Verfahren zur Herstellung geometrisch bestimmter Festkörper [44]. Als Beispiel für die diskrete Fertigung kann der Karosseriebau in der Automobilindustrie mit den erstellten Karosserien genannt werden. Die weiteren Prozessstypen

sind die Chargen- bzw. Batchproduktion, eine Kombination aus kontinuierlicher und diskreter Fertigung. Heute werden diskrete Fertigungsanlagen im Vergleich zu kontinuierlichen Fertigungsanlagen häufiger modifiziert, insbesondere aufgrund von Änderungen im Durchsatz und den Produktanforderungen [45], [46]. Aus diesem Grund stellen sich die eingangs genannten Herausforderungen überwiegend im Betrieb von diskreten Produktionsanlagen. Der Ansatz in dieser Arbeit zur Synchronisierung des Digitalen Zwillings mit dem realen Produktionssystem, um die Verfügbarkeit der Anlage während der Rekonfiguration zu erhöhen, wird sich daher auf den Prozesstyp der diskreten Fertigung konzentrieren. In Kontext dieser Arbeit werden die Automatisierungssysteme in der diskreten Fertigung allgemein als automatisiertes System bezeichnet. Die Automatisierungssysteme mit den Prozesstypen der kontinuierlichen Fertigung und Chargen- bzw. Batchproduktion werden hier nicht berücksichtigt.

Entstehung eines Digitalen Zwillings während des Engineering-Prozesses

In dieser Arbeit wird davon ausgegangen, dass während des initialen Engineering-Prozesses ein Digitaler Zwilling zu einem automatisierten System erstellt wird. Der Schwerpunkt liegt auf der Synchronisierung der bestehenden Modelle des Systems ab dem Zeitpunkt der Inbetriebnahme des Systems. Eine automatisierte Generierung eines Digitalen Zwillings aus einem bestehenden automatisierten System, ohne Beachtung von dessen domänenübergreifender Modelle vor der Inbetriebnahme, wird in dieser Arbeit nicht fokussiert.

1.6 Aufbau der Arbeit

Die vorliegende Arbeit gliedert sich in 8 Kapitel, vgl. Abbildung 2. Im zweiten Kapitel werden einerseits die grundlegenden Methoden und Vorgehensmodelle für die Entwicklung eines automatisierten Systems beschrieben und andererseits die Bedeutung der Durchgängigkeit zwischen den zahlreichen Engineering-Tools und deren Modellen während des Engineering-Prozesses erläutert.

Im dritten Kapitel werden die vier wesentlichen Aspekte des Standes der Wissenschaft und Technik hinsichtlich der Erfüllung der Forschungsanforderungen untersucht. Hierbei werden die Definition und Bestandteile des Digitalen Zwillings im cyber-physischen Produktionssystem, bestehende Methoden sowie Frameworks für die Integration domänenübergreifender Modelle innerhalb eines Digitalen Zwillings, die Ansätze zur automatisierten Modellgenerierung aus dem bestehenden automatisierten System und bekannte Ansätze zur Synchronisierung von Modellen innerhalb des Digitalen Zwillings unter Berücksichtigung der Forschungsanforderungen untersucht. Anschließend werden diese bestehenden Ansätze anhand der im ersten Kapitel definierten Forschungsanforderungen bewertet und eine Forschungslücke, mit dem Inhalt: "Fehlende Methodik, die domänenübergreifende Änderungen und deren Abhängigkeiten vom Gesamtsystem nach der Inbetriebnahme eines automatisierten Systems erkennt und diese in den Modellen des digitalen Zwillings anpasst", identifiziert.

Im vierten Kapitel wird als Lösungsansatz zur Schließung der Forschungslücke das Konzept der „Ankerpunktmethode“ vorgeschlagen. Dieses Konzept weist drei Phasen auf: (1) Automatisierte Detektion der Änderungen in einem automatisierten System durch Steuerungssoftwareanalyse, (2) automatisierte Detektion der domainübergreifenden Abhängigkeiten der Änderungen zwischen der mechanischen, elektrischen und Softwaredomänen durch regelbasierte Abhängigkeitsdetektion und anschließend (3) automatisierte Anpassung der domänenübergreifenden Modelle innerhalb des Digitalen Zwillings.

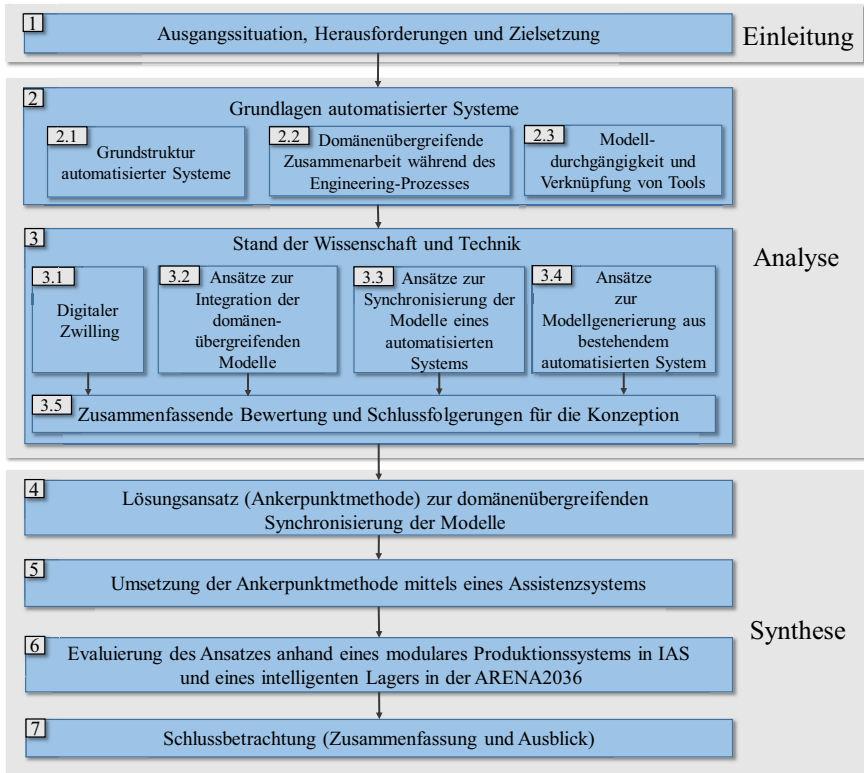


Abbildung 2: Aufbau der Arbeit

Mit Blick auf eine prototypische Umsetzung der Ankerpunktmethode wird im fünften Kapitel ein Assistenzsystem vorgestellt. Dieses Assistenzsystem unterstützt die Realisierung der drei Phasen des Konzepts, die Änderungsdetektion, die Abhängigkeitsdetektion sowie das Änderungsmanagement und stellt die Synchronisierung zwischen einem realen, automatisierten System sowie den entsprechenden Modellen innerhalb seines Digitalen Zwillings sicher. In diesem Kapitel werden die Softwarekomponenten und die Funktionalität des Assistenzsystems ausführlich beschrieben.

Im sechsten Kapitel erfolgt die Bewertung der Ankerpunktmethode anhand von zwei Evaluationsszenarien. Zunächst wird der physische und digitale Aufbau von zwei Evaluationsszenarien vorgestellt. Im Anschluss daran folgt der Ablauf der Prüfung. Das Konzept zur Synchronisierung der Modelle des Digitalen Zwillings wird durch eine quantitative Bewertung mithilfe der Messgröße "Zeit" und eine qualitative Bewertung durch die nachweisbare Erfüllung der zuvor festgelegten Anforderungen validiert. In diesem Kapitel wird am Ende wieder auf die Ausgangssituation der Arbeit hingewiesen.

Im abschließenden Kapitel werden die wesentlichen Aspekte dieser Arbeit zusammengefasst. Dabei kommen sowohl die erworbenen Erfahrungen zur Sprache als auch zukünftige Anwendungen und Erweiterungen.

2 Grundlagen automatisierter Systeme

Als automatisiertes System werden diejenigen technischen Einrichtungen bezeichnet, die für die Automatisierung des technischen Prozesses erforderlich sind [7]. Das beinhaltet das technische System, das Rechner- und Kommunikationssystem sowie die Menschen zur Leitung und Bedienung des auf dem technischen System ablaufenden technischen Prozesses [4].

Ein automatisiertes System in der diskreten Fertigung besteht aus mechanischen, elektrischen und Softwareteilen, die alle eng miteinander verwoben sind. Sie stellen somit eine besondere Klasse mechatronischer Systeme dar [47]–[49] und bestehen aus mechatronischen Komponenten wie Sensoren und Aktoren. Daher können für die Entwicklung von diesen automatisierten Systemen die Entwicklungsmethoden für mechatronische Systeme angewendet werden [47].

Die Entwicklung eines mechatronischen Systems ist ein domänenübergreifender Engineering-Prozess. Der Einfluss der Mehrdimensionalität auf die Entwicklung, den Umgang mit den Entwicklern und deren Tools ist von besonderer Bedeutung für die Synchronisierung der entwickelten Modelle und den weiteren Arbeitsablauf. In diesem Kapitel wird die Grundstruktur mechatronischer Systeme als ein automatisiertes System und ein mechatronisches Entwicklungsvorgehen während des Engineering-Prozesses nach aktuellen gültigen Normen sowie Richtlinien aus der Fachliteratur untersucht und Begrifflichkeiten geklärt. Dabei spielen die Phasen, in die das Engineering unterteilt ist, eine wesentliche Rolle. Von besonderem Interesse für die vorliegende Arbeit ist die Beantwortung der Frage: Welche Beteiligten in den unterschiedlichen Engineering-Phasen bringen welche Arbeitsergebnisse hervor? Dabei wird auf die Besonderheiten des domänenübergreifenden Engineerings in der Domäne Mechanik, Elektrik sowie Software und deren zahlreichen Tools eingegangen, um einen Überblick über die notwendige Durchgängigkeit sowie Verknüpfung der erstellten domänenübergreifenden Modelle des Systems zu gewinnen. Abschließend wird der Stand der Wissenschaft und Technik zur Bereitstellung einer domänenübergreifenden Modelldurchgängigkeit zwischen den Tools während des Engineering-Prozesses aus der Fachliteratur untersucht und diskutiert.

2.1 Grundstruktur automatisierter Systeme

Die Entwicklungsmethoden von mechatronischen Systemen können auch auf die Entwicklung von automatisierten Systemen in der diskreten Fertigung angewendet werden [47]. Deshalb wird hier die Mechatronik sowie deren Grundstruktur erläutert. Der Begriff Mechatronik wurde zum ersten Mal im Jahr 1969 verwendet [50][51]. Dieser Begriff setzt sich aus den Worten Mechanik und Elektronik zusammen und kennzeichnete zunächst nur die elektrotechnische und elektronische Funktionserweiterung von mechanischen Komponenten sowie Geräten [51]. Mit dem Aufkommen der Mikroelektronik und besonders der Mikroprozessortechnik ist die Software als weiterer Bestandteil der Mechatronik hinzugekommen [52]. In der VDI-Richtlinie 2206 wird

die Mechatronik als „das synergetische Zusammenwirken der Fachdomänen Maschinenbau, Elektrotechnik und Informationstechnik beim Entwurf und der Herstellung industrieller Erzeugnisse sowie bei der Prozessgestaltung“ definiert [53]. Ein Ziel der Mechatronik ist es, das Verhalten eines technischen Systems zu verbessern, indem mit Hilfe von Sensoren Informationen über die Umgebung, aber auch über das System selbst erfasst werden [53]. Ein mechatronisches System lässt sich dementsprechend als ein System beschreiben, das aus einem Grundsystem, Sensoren, Aktoren und einer Informationsverarbeitung besteht, siehe dazu Abbildung 3 [46]. Das Grundsystem ist dabei der Produktionsprozess, der eine mechanische Ausprägung aufweist [54]. Die Sensoren erfassen die messbaren Kenngrößen des Grundsystems und melden diese an die Informationsverarbeitung in Form von Messsignalen. Das Gesamtsystem kann eine Mensch-Maschine-Schnittstelle besitzen, die die Verbindung zwischen dem Menschen und der Informationsverarbeitung herstellt. Darüber hinaus kann die Informationsverarbeitung des mechatronischen Systems mit Hilfe eines Kommunikationssystems mit weiteren Informationsverarbeitungssystemen verbunden werden, Dies ermöglicht es, Systeme hierarchisch in Subsysteme aufzuteilen oder Systeme zu vernetzen [8]. Zudem können mit Hilfe von Peripheriemodulen die von der Informationsverarbeitung berechneten Werte (Stellsignale) gesetzt werden, sodass die Aktoren die notwendigen Eingriffe in das Grundsystem durchführen. Die Aktoren wandeln die Energie der Leistungsversorgung in Abhängigkeit von den erhaltenen Stellsignalen in Ausgangssignale um und beeinflussen damit aktiv das Grundsystem. Die Aktoren wandeln die Energie der Leistungsversorgung in Abhängigkeit von den erhaltenen Stellsignalen in Ausgangssignale um und beeinflussen damit aktiv das Grundsystem.

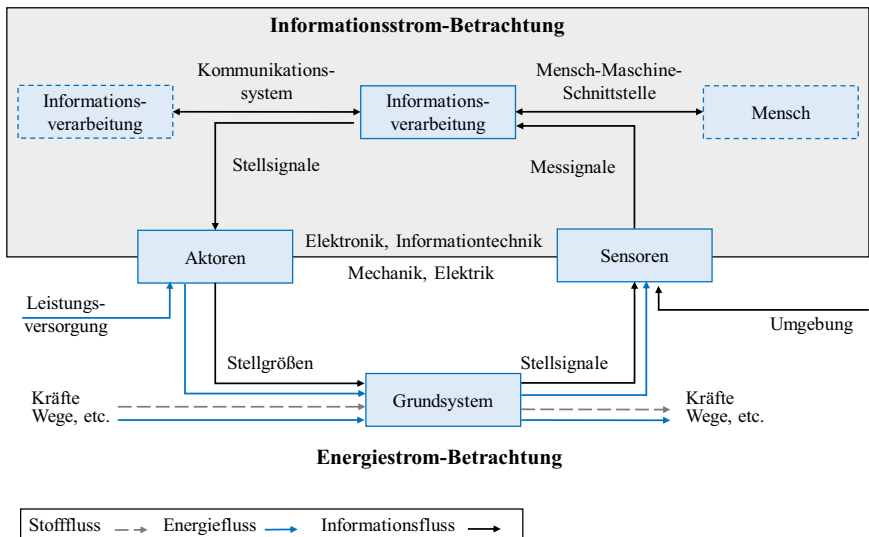


Abbildung 3: Grundstruktur mechatronischer Systeme nach der VDI-Richtlinie 2206 [53]

Dieser Aufbau kann auch auf SPS-gesteuerte, automatisierte Systeme in der diskreten Fertigung übertragen werden. Das Grundsystem repräsentiert dabei die mechanische Ausprägung der

Maschine und des Materialsystems. Die Sensoren messen Stellgrößen des Grundsystems und melden diese in Form von Signalen an die Informationsverarbeitung. Diese wird im Falle der diskreten Fertigung oft durch eine SPS umgesetzt.

2.1.1 SPS-gesteuerte automatisierte Systeme

In einem SPS-gesteuerten, automatisierten System ist eine SPS ein Industrie-Computersteuerungssystem, das den Zustand von Sensoren mittels Eingangssignalen kontinuierlich überwacht und Entscheidungen basierend auf einer anwenderspezifischen Steuerungssoftware zur Steuerung des Zustands von Aktoren trifft. Für die Programmierung einer SPS wurden proprietäre Programmiersprachen entwickelt. Zur Standardisierung der Programmiersprachen für SPS liegt die Norm IEC61131-2 [55] vor, in der es bereits fünf Standardprogrammiersprachen gibt. In der Norm IEC61131-2 werden die Anweisungsliste (AWL), der Kontaktplan (KOP), der Funktionsplan (FUP), der Ablaufplan (AS) und ein strukturierter Text (ST) für die SPS-Programmierung empfohlen. Dabei sind die Sprachen AWL und ST textbasiert, die anderen Sprachen grafisch. Diese Sprachen müssen grundsätzlich die Ein- und Ausgangssignale der Sensoren und Aktoren im System sowie die Funktionsblöcke zur Modellierung der gewünschten Funktionsabläufe der Aktoren und Datenblöcke zur Speicherung der Betriebsdaten des Systems umfassen. Die Betriebsdaten in den Datenblöcken werden über die Funktionsblöcke in der Steuerungssoftware angefordert. Die Verwendung dieser Sprachen hängt von der Komplexität und den Anforderungen des Fertigungsprozesses in der diskreten Fertigung ab.

2.1.2 Entwicklungsvorgehen bei automatisierter Systeme

Die Entwicklung automatisierter Systeme in der diskreten Fertigung entspricht der Entwicklungsmethode mechatronischer Systeme [47].

Der Entwicklungsprozess eines mechatronischen Systems ist eine domänenübergreifende Herausforderung zwischen Mechanik, Elektrik und Software [32]. Zwar erfüllen die Ingenieure in jedem der beteiligten Domänen aufgrund ihrer Expertise spezifische Aufgaben, doch müssen die erstellten Teillösungen aller Domänen eine gemeinsame Lösung schaffen, um die Funktionalität des Systems zu gewährleisten [53]. Als Vorgehensmodell wird die Zusammenstellung von Methoden und Elementen des Projektmanagements zu Prozessen und Phasen eines standardisierten Projektablaufs definiert [56]. Der Nutzen von Vorgehensmodellen für die domänenübergreifende Zusammenarbeit liegt nach [57] in der klaren Zuweisung von Tätigkeiten und der Festlegung der Abfolge dieser Tätigkeiten. Im Bereich der Mechatronik gibt es zahlreiche Vorgehensmodelle, die den Entwicklungsprozess strukturieren. Diese sind zum Teil

in den VDI-Richtlinien 2206 [53] und 2222 [58] als auch in wissenschaftlichen Arbeiten wie [46], [59] veröffentlicht.

Der Verein Deutscher Ingenieure (VDI) schlägt in der Richtlinie-2206 [53] ein Vorgehensmodell für die Entwicklung mechatronische Systeme basierend auf dem V-Modell vor. Das "V" im V-Modell ergibt sich aus der Form der grafischen Darstellung des Modells. Es kombiniert einen Top-Down-Ansatz hinsichtlich des Entwurfs mit einem Bottom-Up-Ansatz in Bezug auf die Verifizierungsphase. Das in Abbildung 4 dargestellte Vorgehensmodell kann als Makrozyklus mehrmals durchgeführt werden, wobei die Ergebnisse kontinuierlich verfeinert werden [46]. In diesem Vorgehensmodell werden die Produktanforderungen als Basis für die Entwicklung betrachtet. Der Systementwurf beginnt mit der Festlegung der Anforderungen, die das spätere Produkt erfüllen muss. Die Systementwurfsphase ist wesentlich durch die funktionale Betrachtung des Systems und dessen Zerlegung in Teilfunktionen geprägt. Der anschließende domänenspezifische Entwurf baut auf dem im Systemdesign erstellten Funktionsmodell auf. In dieser Phase werden in den einzelnen Domänen Mechanik, Elektrik und Software konkrete Lösungen für die Realisierung der Funktionen entwickelt. Dabei werden mehrere Modelle in verschiedenen Domänen entworfen. Bei der Systemintegration-Phase erfolgt eine Integration der Ergebnisse aus den jeweiligen Domänen zum Gesamtsystem, um die Zusammenhänge untersuchen zu können. In der letzten Phase des Makrozyklus wird durch die Eigenschaftsabsicherung überprüft und sichergestellt, ob die von der Systemintegration entwickelte Lösung den ursprünglichen Anforderungen entspricht. In diesem Vorgehensmodell wird die Modellbildung und -Analyse während des gesamten Prozesses durchgeführt.

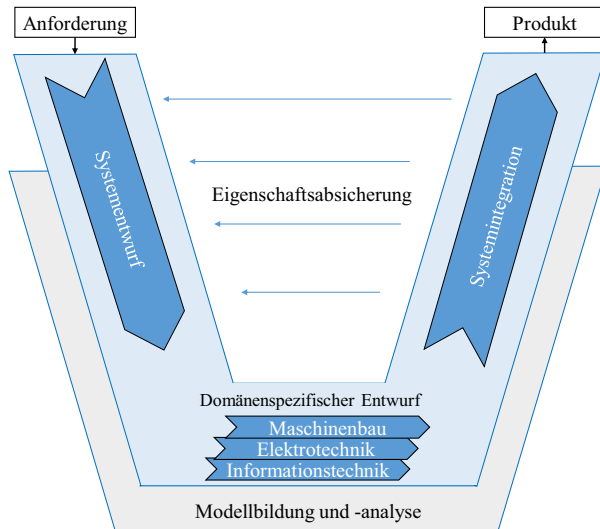


Abbildung 4: Das V-Modell für mechatronische Entwicklungsvorgehen nach [53]

Dabei werden die beschriebenen Phasen durch die Abbildung und Analyse der Systemeigenschaften mit verschiedenen Tools, deren Modellen und Simulationen, unterstützt. Neben diesem Vorgehensmodelle werden in [60] unterschiedliche Engineering-Vorgehensmodelle miteinander verglichen. Hierbei wurde festgestellt, dass obwohl in der Literatur verschiedene Engineering-Prozesse beschrieben werden, sie in der Regel ähnlich sind, wobei je nach Anwendungsbedarf entsprechend unterschiedliche Aspekte hervorgehoben werden [60] [61]. Lüder leitet in [60] aus dem Vergleich zahlreicher Engineering-Ansätze und Vorgehensmodelle ein mechatronisch orientiertes, allgemeines Engineering-Vorgehen ab. Dieses Engineering-Vorgehen wird in [61] erneut erläutert, in dem die domänenübergreifenden Aktivitäten in den jeweiligen Phasen detailliert vorgestellt sind, siehe Abbildung 5.

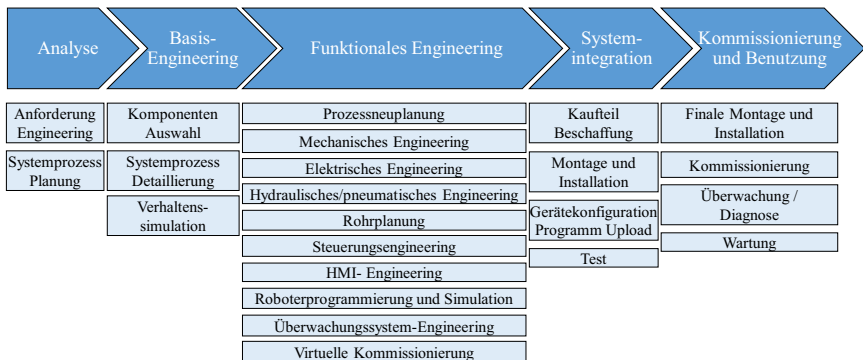


Abbildung 5: Engineering-Vorgehensmodell und dessen domänenübergreifende Aktivitäten nach [61]

In diesem Engineering-Vorgehensmodell wird im Detail auf die Phase des funktionalen Engineerings eingegangen, das durch das domänenübergreifende Planen und Konstruieren des Produktionssystems geprägt ist, vgl. [61]. Dies ist die Phase, in der die größte Möglichkeit der Parallelisierung und domänenübergreifenden Zusammenführung der Tätigkeiten gegeben ist, da hier unterschiedliche Tools Planungsschritte ausführen [62]. Im Rahmen der funktionalen Engineering-Phase werden alle zuvor erstellten Ergebnisse detailliert, d. h. der grundlegende Entwurf des vorgesehenen technischen Systems kristallisiert sich heraus. Das technische System wird vollständig entwickelt, sodass alle für den Aufbau des technischen Systems notwendigen Informationen zur Verfügung stehen [61]. Dabei wird die funktionale Engineering-Phase nach [61] in die verschiedenen Aktivitäten, in der mechanischen-, elektrischen- und Software-Domäne unterteilt, wie Abbildung 5 aufzeigt.

Auf die unterschiedlichen, am Engineering beteiligten Domänen während des funktionalen Engineerings wird in Kapitel 2.2 der vorliegenden Arbeit detaillierter eingegangen.

2.2 Domänenübergreifende Zusammenarbeit während des Engineering-Prozesses automatisierter Systeme

Ein automatisiertes System in der diskreten Fertigung besteht aus mechanischen, elektrischen und Software Teilen, die alle eng miteinander verwoben sind, und stellen somit eine besondere Klasse mechatronischer Systeme dar [47]–[49]. Nach der VDI-Richtlinie 2206 erfolgt die Erstellung eines mechatronischen Systems im Zusammenspiel von mehreren Domänen [53][30][32]. Der Begriff *Domäne* bezeichnet dabei ein Spezialgebiet auf dem die Mitarbeiter der Domäne besondere Kompetenzen haben [57]. Der Engineering-Prozess für die Entwicklung einer Fertigungsanlage ist heute durch die starke Integration unterschiedlicher Domänen gekennzeichnet, wodurch eine domänenübergreifende Herausforderung zwischen Mechanik, Elektrik und Software-Domäne zu bewältigen ist [53], [63], [64]. Jede dieser Domänen bringt ihr jeweiliges Wissen und ihre Kompetenz in das Engineering ein. In ihrem Spezialgebiet müssen dabei von den Domänenexperten entlang des Engineering-Prozesses verschiedene Teilaufgaben erfüllt werden [65]. Zusätzlich zu den genannten Domänen sind auch der technische Vertrieb und das Projektmanagement am Engineering beteiligt, jedoch zeichnen sich diese beiden betrieblichen Funktionen nicht durch einen besonderen technischen Fokus aus, sondern durch ein allgemeines technisches Verständnis sowie einem Aufgabenspektrum, das insbesondere im wirtschaftlichen und organisatorischen Bereich liegt [57]. Obwohl jeder Ingenieur die spezifischen Aufgaben der verschiedenen Domänen aufgrund seiner Fachkenntnisse erfüllt, müssen die domänenbezogenen Teillösungen in ein gemeinsames Konzept umgesetzt werden, um sicherzustellen, dass das System betriebsfähig ist [53]. Das Zusammenführen der fachspezifischen Lösungen, die alle ihren eigenen Charakter sowie Merkmale haben und durch Ingenieure mit unterschiedlichen Fachkenntnissen erstellen werden, führt jedoch zu einem hohen Komplexitätsgrad und dadurch u.a. zu mehreren Herausforderungen. Eine der größten Herausforderungen besteht darin, dass die Entwicklung mechatronischer Systeme in der Frühphase traditionell einem sequentiellen Entwicklungsprozess folgt, der mit dem mechanischen Design beginnt [32] [66] und mit dem Elektro- und Softwaredesign fortgesetzt wird [67]. Schon beim sequentiellen Engineering eines Systems müssen im Laufe des Prozesses zahlreiche Informationen zwischen den Domänen ausgetauscht werden [46].

Zu Beginn eines Engineering-Prozesses werden üblicherweise Kundenanforderungen gesammelt. Die mechanische Konstrukteure gehen aufgrund einer impliziten Abstraktion des Problems sofort zur Definition von prinzipiellen Lösungen über [68]. Erst nach der Definition eines Gesamtkonzeptes wird dieses im Mechanical Computer Aided Design-System, abgekürzt MCAD-System, modelliert. Dazu stehen dem mechanischen Konstrukteur Hilfsmittel zur Beschreibung von Geometrien zur Verfügung [68]. Die Geometriebeschreibungen werden in sogenannten Parts oder Bauteilen gespeichert. Ein Zusammenschluss mehrerer Parts zu einer Baugruppe wird auch als Assembly bezeichnet. [68] Die gesamten Parts und Assembly des Fertigungssystems sowie deren Aufbaustruktur wird als Stückliste bezeichnet. Zwischen den

Domänen Mechanik und Elektrik wird im einfachsten Fall eine Stückliste relevanter Bauteile ausgetauscht [46], [57]. Diese enthält im optimalen Fall eindeutige Betriebsmittelkennzeichen, um eine Zuordnung der Komponenten in der domänenübergreifenden Planung zu gewährleisten [69]. Diese Bauteile werden in einem Elektro-CAD-Programm miteinander verbunden [46]. Hier entwerfen die Elektroingenieure die Infrastruktur des Systems in einem Elektro-CAD-System (ECAD) und ergänzen die von der Abteilung Maschinenbau spezifizierten Sensoren und Aktoren [66]. In dieser Phase kann jeder Sensor und jeden Aktor direkt auf Symbole im Schaltplan abgebildet werden. Gegebenenfalls werden durch den Elektroplaner Steuerungskomponenten hinzugefügt, Signallisten definiert und in der Stückliste ergänzt. Aus der Domäne der Elektronik werden die Signale von Sensoren und Aktoren als Signalliste oder auch als Hardware-Adressliste der Steuerung sowie Steuerungskomponenten an die Software-Domain übergeben [69]. Basierend auf diesen Information und aus einer detaillierten Ablaufbeschreibung, die vom Maschinenbau als führendes Domain entworfen wird, erstellen die Softwaretechniker das Steuerungsprogramm für das automatisierte System [70][46]. Im letzten Schritt in der Software-Domain wird der Steuerungscode der SPS mit den Verhaltensmodellen der automatisierten Systeme sowie den Modellen der mechatronischen Komponenten in der sogenannten virtuellen Inbetriebnahme getestet [71], [30]. Ziel der virtuellen Inbetriebnahme ist unter anderem die Validierung der entwickelten Steuerungssoftware. Durch die Kopplung der verschiedenen Modelle kann der Zusammenhang zwischen einem Sensorsignal, der Verarbeitung durch die Software, das resultierende Steuersignal sowie die Reaktion durch den Aktor virtuell getestet und gegen die Anforderungen geprüft werden.

Wie bereits beschrieben, muss in den sequentiellen Schritten des Engineerings eine große Menge an Informationen und Modellen zwischen den Domänen und ihren Ingenieuren ausgetauscht werden. Zusammengefasst lässt sich der Austausch von Informationen, wie die Stückliste, die Signalliste oder die Prozessbeschreibung, zwischen den beteiligten Domänen in Anlehnung an [46] darstellen, vgl. Abbildung 6.

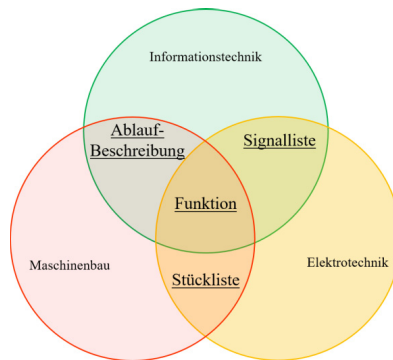


Abbildung 6: Daten und Informationsaustausch während des Engineerings eines automatisierten Systems nach [46]

Zur Generierung dieser Modelle müssen verschiedene Tools eingesetzt werden. In [61] wurden detailliert die häufig verwendeten Tools der unterschiedlichen Hersteller ohne bestimmte Systematik detailliert aufgeführt, um deren Vielfältigkeit während des funktionalen Engineerings zu verdeutlichen, vgl. Abbildung 7.

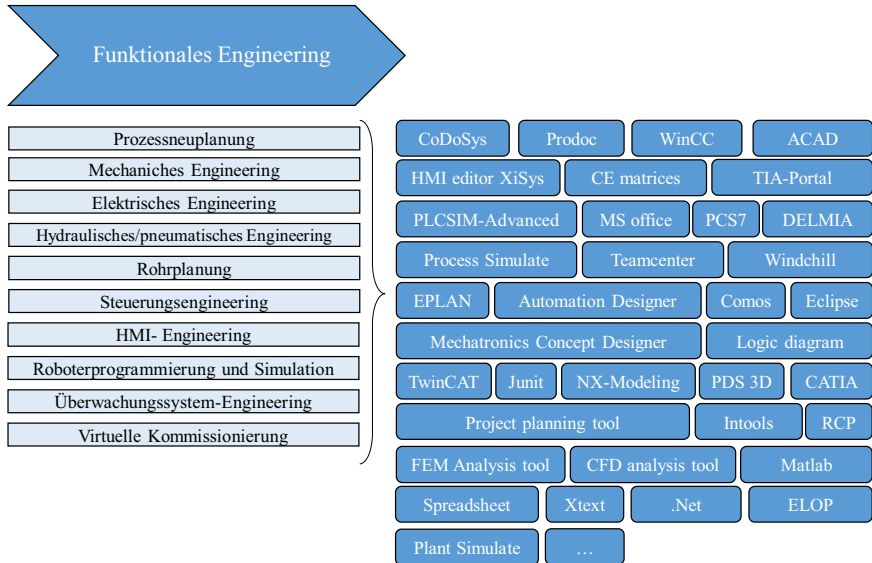


Abbildung 7: Tools während des funktionalen Engineerings in Anlehnung an [61]

Wie in Abbildung 7 erkennbar, existieren für jede Domäne spezielle Tools, wie MCAD, ECAD und Steuerungssoftware-Programmierung-Tools, die die Entwicklungsarbeit unterstützen [68].

Zur Überprüfung des gesamten Systems, dessen Teilsysteme mit verschiedenen Tools entwickelt wurden, müssen die erstellten Modelle zusammengeführt und analysiert werden. Zusammengefasst gilt es, während des Engineering-Prozesses eines mechatronischen Systems drei grundlegende Aspekte zu berücksichtigen:

- Zunächst müssen verschiedene Modelle und Informationen während des Engineerings ausgetauscht werden.
- Diese Modelle werden mit den unterschiedlichen Tools in den drei Domänen mit unterschiedlichen Sichtweisen erstellt.
- Eine Modelldurchgängigkeit zwischen den Tools ermöglicht eine hohe Effizienz während des Engineering-Prozesses durch eine Parallelisierung der Aufgaben und eine domänenübergreifende Zusammenarbeit.

Ausgehend von diesen Aspekten kann abgeleitet werden, dass die Verknüpfung von Tools und deren Modellen notwendig ist, um die Modelldurchgängigkeit zu gewährleisten und die Engineering-Effizienz zu erhöhen.

Wie sich die Tools und deren erstellten Modellen während des Engineerings verknüpfen lassen und wie Informationen zwischen diese Modelle überführt werden können, wird im folgenden Kapitel untersucht.

2.3 Ansätze zur Modelldurchgängigkeit und Verknüpfung von Tools

In Kapitel 2.2 wurde erläutert, dass bei der Konstruktion und Verifizierung eines automatisierten Systems in der diskreten Fertigung während des Engineerings zahlreiche Tools auf dem Stand der Technik wie MCAD, ECAD, SPS-Programmierungsumgebung und Simulationstools für die virtuelle Inbetriebnahme zur Verfügung stehen, um verschiedene Systemaspekte zu berücksichtigen und zu simulieren. Der CIMdata Marktanalysebericht, ein weltweit tätiges PLM-Beratungs- und Forschungsunternehmen für strategisches Management, berichtet in [72] und [73], dass die Engineering-Tools einen Jahresumsatz von 39 Milliarden Euro mit einer jährlichen Wachstumsrate von 7% haben. Diese Zahlen sind besonders beachtenswert, da sie darauf hinweisen, dass es bereits eine Vielzahl von Tools in der Industrie gibt, insbesondere im Bereich des Engineerings. Entscheidend ist hier die Verknüpfung und Zusammenarbeit dieser Tools. In diesem Zusammenhang gewinnt das Thema der Modelldurchgängigkeit zwischen den Tools eine sehr große Bedeutung. Zu diesem Zweck stehen mehrere Lösungen gemäß dem Stand der Wissenschaft und Technik zur Verfügung. Eine Lösung stellen die proprietären Punkt-zu-Punkt-Schnittstellen zwischen den Tools dar. Frank in [68] und Bellalouna in [74] berichten über weitere Ansätze zum Modellaustausch. Diese können zusammengefasst in drei weitere Ansätze kategorisiert werden: Ansätze auf Basis strukturierter Daten, Ansätze auf Basis der neutralen Austauschformate und Ansätze auf Basis semantischer Technologien. In den folgenden Unterkapiteln werden diese Ansätze vorgestellt und beschrieben.

2.3.1 Proprietäre Punkt-zu-Punkt-Schnittstellen zwischen Tools

In diesem Ansatz werden die einzelnen Tools mit bilateralen Schnittstellen verbunden [75]. Dafür muss paarweise ein Konverter für je zwei Tools erzeugt werden [68]. Ein Konverter ermöglicht die Übersetzung von einem Datenmodell in ein anderes. Je nach Qualität der Quelldatei und des Konverters kann die Qualität des Ergebnisses der Zieldatei gleichwohl sehr unterschiedlich sein [76]. Dieser Ansatz führt schnell zu ersten Ergebnissen und wird deshalb gern ausgewählt [57].

Allerdings steigt mit der Anzahl der angeschlossenen Tools auch die Anzahl der notwendigen Konverter stark an.

Deshalb ist dieser Ansatz trotz des Einsatzpotenzials zwischen den Tools aufgrund der hohen Anzahl der benötigten Schnittstellen sowie des hohen Aufwandes bei der Pflege der Schnittstellen für komplexe mechatronische Systeme nicht geeignet [77], [57], [78].

2.3.2 Standardisierte neutrale Austauschformate

Ein anderer Ansatz ist die Verwendung eines systemneutralen, zentralen Modells, das Daten und Informationen aller beteiligten Tools aufnimmt und damit ein Abbild aller Aspekte des Gesamtsystems darstellen kann [75][77][78][57].

Dabei ist es einerseits eine Herausforderung, die zentrale Datenhaltung so zu gestalten, dass alle domänenspezifische Modelle in ihr abgelegt werden können, andererseits muss das Format einfach bleiben, um die Schnittstellen und die Interpretation der Modelle beherrschbar zu halten [57]. Abbildung 8 veranschaulicht die Unterschiede zwischen den beiden Ansätzen "proprietäre Punkt-zu-Punkt-Schnittstellen zwischen Tools" und "standardisierte neutrale Austauschformate" für den Modellaustausch. Zum Zweck der systemneutralen, zentralen Datenhaltung gibt es bereits Ansätze, die nun vorgestellt und bewertet werden. Für den neutralen Datenaustausch zwischen den Tools wurden bereits mehrere Datenformate festgelegt. IGES (Initial Graphics Exchange Specification), JT (Jupiter Tessellation) oder DXF (Drawing Interchange File Format) zeichnen sich in erster Linie durch die Speicherung von geometrischen CAD-Daten und die Verknüpfung verschiedener CAD-Konstruktionstools aus. Diese sind aber auf geometrische Modellen beschränkt [57].

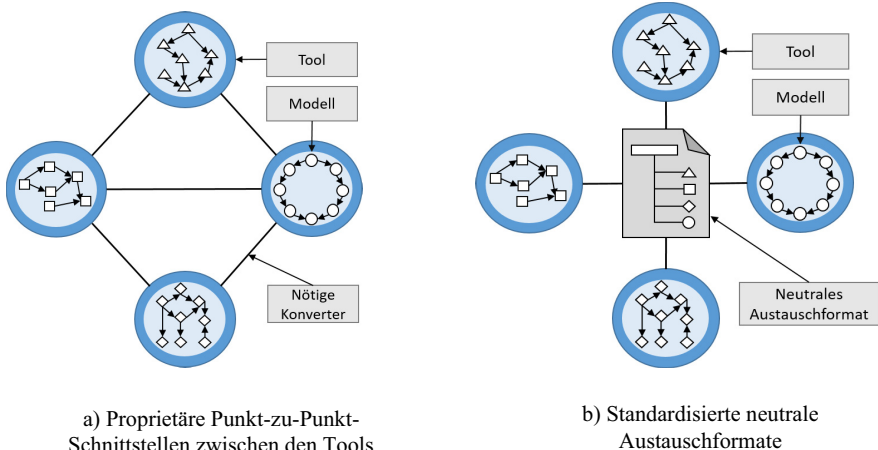


Abbildung 8: Modellaustausch zwischen Tools über proprietäre Punkt-zu-Punkt-Schnittstellen im Vergleich zum Modellaustausch über zentrale Austauschformate

Das Dateiformat STEP (Standard for the Exchange of Product model data) nach ISO 10303-1 erweitert die Geometriedarstellung um funktionale Aspekte [79] und befasst sich mit der Kollaboration von Produktdesigns und der Produktentwicklung [80]. Das MechaSTEP-Datenmodell wurde basierend auf den im STEP-Datenmodell definierten Methoden und Konzepten entwickelt und unterstützt die Simulation mechatronischer Systeme [74]. Als weitere neutrale Datenaustauschformate können PLCopenXML, AutomationML [78] und PLMXML [81] genannt werden. Bei AutomationML handelt es sich um ein sehr verbreitetes neutrales Austauschformat.

AutomationML ist ein XML (eXtensible Markup Language)-basiertes Format, das speziell mit dem Fokus auf das Engineering in der Automatisierungstechnik entwickelt wurde und das Ziel hat, komplette Fertigungsanlagen abbilden zu können [78]. Eine XML-Datei enthält Modelle sowie Daten, die ihr Format beschreiben, sogenannte Metadaten. Aus den in XML-Dateien enthaltenen Informationen können verschiedene Tools die ausgetauschten Daten extrahieren [82]. In AutomationML werden die Anlagenmodelle mit CAEX (Computer Aided Engineering Exchange) als Top-Level-Format abgebildet, das aus Beziehungen, Schnittstellen und Referenzen zwischen den Objekten in der Systemtopologie besteht.

Abbildung 9 fasst sämtliche AutomationML-Modelle in einer Übersicht zusammen. Die Geometrie und Kinematik wurden über COLLADA und das Verhalten mit Hilfe von PLCopenXML[78] abgebildet. Andererseits können die Betriebsdaten einer Anlage über OPC-UA mit der CAEX-Architektur von AutomationML [83] verknüpft werden. OPC UA ist eins der wichtigsten Kommunikationsprotokolle für Industrie 4.0-Anwendungen in der Automatisierungstechnik [84]. Als XML-basiertes Datenaustauschformat gehört AutomationML zur Gruppe der offenen Formate wie PLMXML und STEP AP242 [56].

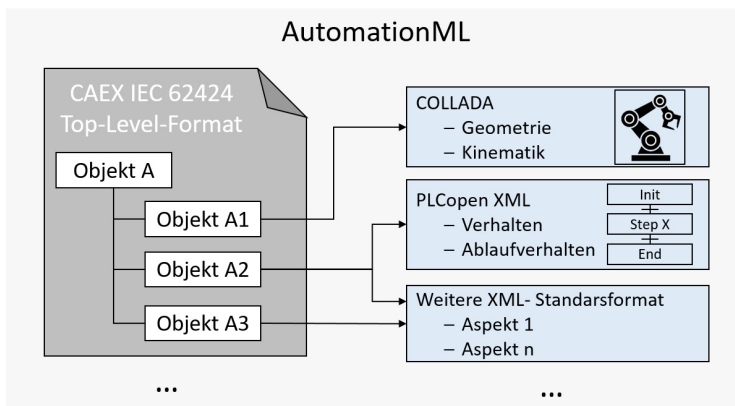


Abbildung 9: Grundstruktur von AutomationML nach [165]

2.3.3 Strukturierte Daten

Dieser Ansatz lässt sich in zwei Umsetzungsarten unterteilen: Data-Warehouse und Single IT-Systeme; *„Ein Data-Warehouse ist eine physische Datenbank, deren Inhalt sich aus Daten unterschiedlicher Quellen zusammensetzt. Die Daten werden von verschiedenen operationalen Tools in das Data Warehouse geladen, um eine integrierte Datensicht vor allem zu Analyse Zwecken zu ermöglichen.“* [85] [74].

Der Ansatz des Data-Warehouse integriert verschiedene Datenbanken auf physischer Ebene in einem globalen Schema, sodass die Daten und Modelle der einzelnen Tools erhalten bleiben [68]. Bei diesem Ansatz definieren Datenbanken die Semantik der Daten implizit in ihrer Struktur [68]. Hierzu besitzt die Datenbank eine eindeutige Struktur, in welcher die Daten abgespeichert werden. Diese Datenbank kann unterschiedliche Ausprägungen besitzen, beispielsweise eine hierarchische, netzartige, relationale oder objektorientierte Struktur [86]. Die Zusammenführung aller Daten der verschiedenen Tools in einem gemeinsamen Data-Warehouse bedingt ein gemeinsames Datenbankschema bzw. Datenmodell für alle Tools. Die Erstellung und die Pflege derartiger Datenmodelle aufgrund der Änderungen der Datentypen und -umfänge sind realistisch sehr schwer umsetzbar [74]. Deswegen konnte der Data Warehouse-Ansatz zur Verknüpfung von Tools und deren Modellen bis heute in der Praxis keine breite Akzeptanz erzielen [74]. Um dieser Herausforderung zu begegnen, die Daten zu pflegen, zu integrieren und eine konsistente Datenkonsistenz zwischen verschiedenen Modellen zu erreichen, können alle Modelle entlang des Lebenszyklus eines Systems auf einem umfassenden Single-IT-System aufgesetzt werden, das ebenfalls datenbankbasiert ist [86]. Dieser Ansatz beruht gleichermaßen auf einem strukturierten Datenaustausch in verschiedenen Domänen, der es ermöglicht, heterogene Modelle in ein einziges IT-System zu integrieren und zu verwalten. Derartige Single-IT-Systeme sollten in der Lage sein, die zahlreichen Tools und deren erstellte Modelle im Rahmen des Engineerings, Managements und Services durch ihre umfangreichen Funktionalitäten, Workflows und Schnittstellen zu integrieren [74]. So können verschiedene Modelle verschiedener Tools mit unterschiedlichen Strukturen, die zu derselben mechatronischen Komponente gehören, miteinander verknüpft werden. Als Technologiebeispiele für die Anwendung des Single IT-Systems können angeführt werden: Product Lifecycle Management Systeme und PLM-Systeme, bei denen Modelle aus fachübergreifenden Domänen zusammen strukturiert und referenziert werden [74]. Die mit den Tools erstellten Modelle werden im PLM-System in semantische Strukturen, den sogenannten Stücklisten von Produkt, Prozessen und Equipment etc. abgelegt, die miteinander verlinkt sind.

2.3.4 Semantische Technologien zur Modellierung der Daten

Semantische Technologien können für den Datenaustausch zwischen Tools und Maschinen eingesetzt werden, um deren Interoperabilität zu unterstützen. Draht beschreibt Interoperabilität in [87] wie folgt: „Interoperabilität zwischen Tools verfolgt das Ziel, Konsistenz zwischen den

Daten einer Toolkette computergestützt, systematisch und wiederholt herstellen zu können.“ Um dies zu erreichen, sollten nicht nur die Daten ausgetauscht werden, sondern unter anderem auch deren Syntax (Struktur) und Semantik (Bedeutung) [74]. Dies kann durch Hinzufügen von globalen Identifikationsmerkmalen zu Daten [88] oder durch einen objektorientierten Ansatz durch Identifizieren von Objekten und deren Attributen sowie Beziehungen zu anderen Objekten [89] erreicht werden. Mit diesem Ansatz für den Austausch von Modellen kann ein semantisches Netz bzw. eine Ontologie verwendet werden, um sowohl Modell-interne als auch Modell-zu-Modell-Relationen zu modellieren. In der Ontologie wird die Semantik der Daten explizit definiert und diese verlinkt [68]. Zu diesem Zweck können beispielsweise die Ontologie-Sprachen RDF und OWL verwendet werden, um Informationen und deren Abhängigkeiten strukturiert zu beschreiben sowie Regeln und Schlussfolgerungen zu formulieren [74]. Ein weiteres Vorgehen ist der objektorientierte Ansatz, bei dem Objekte eines beliebigen Domänenmodells sowie deren Verknüpfungen zu Objekten innerhalb und außerhalb einer bestimmten Domäne modelliert werden können, indem deren Daten in Form von Objektattributen gekapselt werden [68].

Trotz der Attraktivität des Datenaustausches zwischen Tools und Maschinen auf Basis semantischer Technologien ist der Aufwand für die Modellierung, Beschaffung und Pflege semantischer Daten aufgrund der hohen Komplexität gängiger Tools, wie Ontologie-Editoren, im semantischen Technologieumfeld heute sehr hoch. [90][74].

2.4 Zusammenfassung

In diesem Kapitel wurde ein automatisiertes System bei der diskreten Fertigung als mechatronisches System betrachtet. In diesem Zusammenhang wurden zunächst die Grundstruktur und Komponenten eines mechatronischen Systems beschrieben. Im Folgenden wurden unterschiedliche Vorgehensmodelle für das Engineering eines mechatronischen Systems und deren Unterschiede diskutiert. Dabei wurde das Vorgehensmodell der VDI-Richtlinie 2206 für das funktionale Engineering ausführlich erörtert. Ausgehend von diesem Vorgehensmodell wurde das Thema des domänenübergreifenden Engineerings in den Domänen, Mechanik, Elektrik und Software behandelt. Hierbei wurden ausführlich die Prozessschritte, Abhängigkeiten und die zahlreichen Tools, die dabei zum Einsatz kommen, erläutert; Es wurde festgestellt, dass im Engineering automatisierter Systeme eine große Menge von Daten und Modellen, die von unterschiedlichen Tools erstellt werden, zwischen den Domänen und Ingenieuren ausgetauscht werden müssen. Abschließend wurde der Stand der Wissenschaft und Technik zur Bereitstellung einer domänenübergreifenden Modelldurchgängigkeit zwischen den Tools während des Engineering-Prozesses aus der Fachliteratur untersucht und beschrieben. Basierend auf diesen beschriebenen Grundlagen im Engineering automatisierter Systeme wird im nächsten Kapitel der Stand der Wissenschaft und Technik hinsichtlich der Synchronisierung der Modelle, die während des Engineerings durch die Tools erstellt wurden, beleuchtet.

3 Stand der Wissenschaft und Technik

In diesem Kapitel wird der Digitale Zwilling erläutert und Stellung bezogen zu den relevanten wissenschaftlichen Veröffentlichungen sowie zum Stand der Wissenschaft und Technik. Dabei wird näher auf den Digitalen Zwilling in Fertigungssystemen, auf die bestehenden Ansätze zur Synchronisierung eines Digitalen Zwillings mit dem automatisierten System und auf die existierenden Ansätze zur automatischen Generierung eines Digitalen Zwillings mit Bezug zu einem bestehenden System eingegangen. Zunächst wird im Unterkapitel 3.1 der Begriff des Digitalen Zwillings in cyber-physischen Systemen, seine Eigenschaften und Funktionalität anhand der Fachliteratur analysiert und die Gemeinsamkeiten dieser Definitionen verdeutlicht. Darauf aufbauend werden verschiedene Architekturen für die Konzeption und Realisierung der Funktionalitäten untersucht. Anschließend wird eine begründete Architektur, die alle in der Literatur genannten Funktionalitäten des Digitalen Zwillings erfüllen kann, detailliert vorgestellt und deren Komponenten beschrieben. Im Folgenden werden drei Schwerpunkte für die Untersuchung des Standes der Wissenschaft und Technik definiert. Diese Definitionen sollen helfen, die Forschungslücke zur Bewältigung der Herausforderungen bei der Synchronisation des Digitalen Zwillings zu identifizieren. Erstens werden im Unterkapitel 3.2 die existierenden Ansätze für die Integration von domänenübergreifenden Modellen in einen Digitalen Zwilling im Engineering betrachtet sowie deren realisierte Frameworks und Technologien beschrieben. Im Rahmen der Modellsynchronisation sind deren zusätzliche Methoden zur Modellanpassung und Konsistenzprüfung innerhalb des Digitalen Zwillings von großer Bedeutung. Daraufhin werden im Unterkapitel 3.3 die veröffentlichten Ansätze und Technologien für die Synchronisierung der Modelle des Digitalen Zwillings eines bestehenden automatisierten Systems erörtert. Im Vordergrund stehen deren Methoden zur automatisierten Änderungsdetektion und domänenübergreifenden Abhängigkeitsdetektion sowie zur Modellanpassung innerhalb des Digitalen Zwillings. Die Ansätze zur automatisierten, domänenübergreifenden Modellgenerierung von derzeitigen, automatisierten Systemen werden im Unterkapitel 3.4 untersucht. Bei diesen Ansätzen wird davon ausgegangen, dass keine bezüglich der Engineering-Phase vorliegen und die aktuellen Modelle des Systems erstellt werden müssen. Abschließend werden in Unterkapitel 3.5 die vorgestellten Ansätze im Stand der Wissenschaft und Technik zusammengefasst und eine Bewertung dieser Ansätze im Hinblick auf die Schließung der Forschungslücke sowie auf die Formulierung der Zielsetzung skizziert.

3.1 Digitaler Zwilling

Das Konzept und die Verwendung des Begriffes "Zwilling" stammt aus dem Apollo-Programm der NASA (National Aeronautics and Space Administration). Das Programm schloss den Bau mindestens zweier identischer Raumfahrzeuge ein, um die Bedingungen der Raumfahrzeuge während der Mission permanent widerspiegeln zu können. Das Raumfahrzeug auf dem Boden

wurde als "Zwilling" bezeichnet. Der Zwilling wurde intensiv für das Training während der Flugvorbereitung eingesetzt. Während der Mission wurde es verwendet, um alternative Prozesse und Gegebenheiten am Bodenmodell zu simulieren, wobei die verfügbaren realen Flugdaten verwendet wurden, um die Flugbedingungen so genau wie möglich abzubilden und Astronauten im Orbit in kritischen Situationen zu unterstützen [91], [92]. Der Begriff "Digitaler Zwilling" wurde erstmals im Jahre 2012 in die integrierte Technologie-Roadmap der NASA aufgenommen. Nach [91] ist ein Digitaler Zwilling ein integriertes Simulationsmodell eines Fahrzeugs oder Systems, das die verfügbaren physikalischen Modelle, Sensoraktualisierungen, Flottenhistorie usw. verwendet, um die Lebensdauer des entsprechenden fliegenden Zwillings zu spiegeln. Von [93] wurde der Digitale Zwilling als aktualisiertes, digitales Modell eines Produkts oder Produktionssystems vorgestellt, das eine umfassende physikalische und funktionale Beschreibung einer Komponente oder eines Systems über den gesamten Lebenszyklus hinweg enthält. In [94] wurden mehrere Definitionen zum Digitalen Zwilling verglichen und bewertet. Daraus wird ersichtlich, dass es eine Vielzahl von Definitionen des Digitalen Zwillings gibt. Aber in den meisten der überprüften Definitionen ist ein Digitaler Zwilling eine möglichst realistische, gleichwertige digitale Repräsentation eines physischen Anlagegegenstands (Asset), die immer mit ihm synchronisiert ist. Darüber hinaus ist es möglich, eine Simulation an der digitalen Repräsentation durchzuführen, um das Verhalten des physischen Anlagegegenstands vorherzusagen und zu analysieren. Die Autoren in [94] geben ihre eigene Definition des Digitalen Zwillings vor. Diese basiert auf der Gemeinsamkeit der Definitionen, erweitert durch die Fähigkeit der digitalen Repräsentation, den physikalischen Anlagegegenstand selbstständig zu verändern und zu kontrollieren. Anhand einer Analyse der Literaturen über den Charakter und die Funktionalität des Digitalen Zwillings erläutern die Autoren in [6] die Ähnlichkeiten der gefundenen Interpretationen als eine Definition des Digitalen Zwillings wie folgt:

- Ein Digitaler Zwilling muss eine digitale Repräsentation eines physischen Anlagegegenstands sein, einschließlich möglichst realistischer Modelle und aller verfügbaren Daten über das physische Asset.
- Die Daten müssen alle Betriebsdaten enthalten, die während des Betriebs erfasst werden sowie alle organisatorischen und technischen Informationen, die während des Engineerings des Anlagegegenstands erstellt werden.
- Ein Digitaler Zwilling muss immer mit dem physischen Asset synchronisiert sein.
- Es muss möglich sein, das Verhalten des physischen Assets innerhalb des Digitalen Zwillings zu simulieren.

Im nächsten Abschnitt werden der beschriebene Charakter und die Funktionalität des Digitalen Zwillings in einem cyber-physischen Produktionssystem näher erörtert.

3.1.1 Digitaler Zwilling im cyber-physischen Produktionssystem

Ein cyber-physisches System, CPS, ist eine Integration von Berechnungsprozessen und physikalischen Prozessen durch Kommunikationsinfrastrukturen [95], die eine adaptive Überwachung und Steuerung der physischen Assets ermöglicht [96]–[98]. Eingebettete Computer und Netzwerke überwachen und steuern die physikalischen Prozesse in der Regel mit Feedbackschleifen, bei denen die physikalischen Prozesse die Berechnungen beeinflussen und umgekehrt [95]. Der Digitale Zwilling ist eine virtuelle Repräsentation eines physischen Assets in einem cyber-physischen Produktionssystem (CPPS), das in der Lage ist, dessen statische und dynamische Eigenschaften zu reflektieren. Er enthält und bildet verschiedene Modelle eines physischen Assets ab, von denen einige ausführbar sind, sogenannte Simulationsmodelle. Aber nicht alle Modelle sind ausführbar, deshalb ist der Digitale Zwilling mehr als nur eine Simulation eines physischen Assets. In diesem Zusammenhang kann ein Asset eine Entität sein, die bereits in der realen Welt existiert, oder eine Abbildung einer zukünftigen Entität, die erstellt werden soll [6]. Im Rahmen des Konzepts Industrie 4.0 enthält ein CPPS verschiedene CPSs, d.h. ein physisches Asset mit eigener Intelligenz und Kommunikationsfähigkeit. Der Digitale Zwilling innerhalb des CPPS ist eine Zusammenstellung aus vielen einzelnen Digitalen Zwillingen verschiedener CPS. Diese Digitalen Zwillinge kommunizieren miteinander und können Daten und Informationen austauschen [6]. Mit Hilfe des Digitalen Zwillings des automatisierten Systems können verschiedene Szenarien zur Rekonfiguration des Produktionssystems unter domänenübergreifenden Aspekten wie Zuverlässigkeit, Energieverbrauch, Prozesskonsistenz, Ergonomie, Logistik usw. simuliert und getestet werden. Diese Anwendungsfälle zeigen, dass der Digitale Zwilling während der Betriebsphase und nicht nur während des Engineering-Prozesses Anwendungsvorteile hat. Um diese nutzen zu können, sollte der Digitale Zwilling über den gesamten Lebenszyklus des automatisierten Systems zur Verfügung gestellt werden [71]. Weitere Anwendungsfälle eines Digitalen Zwillings sind u. a. Ergonomie-Simulationen, Energiesimulationen, Optimierungen des Prozessablaufs, Digitale Mock-Ups, Virtual/Augmented Reality und die virtuelle Inbetriebnahme. Um diese zu ermöglichen, muss der Digitale Zwilling über den gesamten Lebenszyklus des Produkts und des Produktionssystems hinweg bereitgestellt werden.

In Abbildung 10 werden die drei Charakteristiken eines Digitalen Zwillings in einem cyber-physischen Produktionssystem präsentiert. Diese basieren auf den Ergebnissen einer Literaturrecherche zur Definition des Digitalen Zwillings in [6] dargestellt. Im Cyber Layer besteht jedes physikalische Asset aus seinen Modellen und Daten. Diese Modelle und seine Daten bilden zusammen ein digitales Abbild des Assets. Damit dieses digitale Abbild der Definition eines Digitalen Zwillings genügt, müssen folgende drei Charakteristika erfüllt sein: Synchronisierung mit dem realen Asset, aktive Datenerfassung und Simulationsfähigkeit.

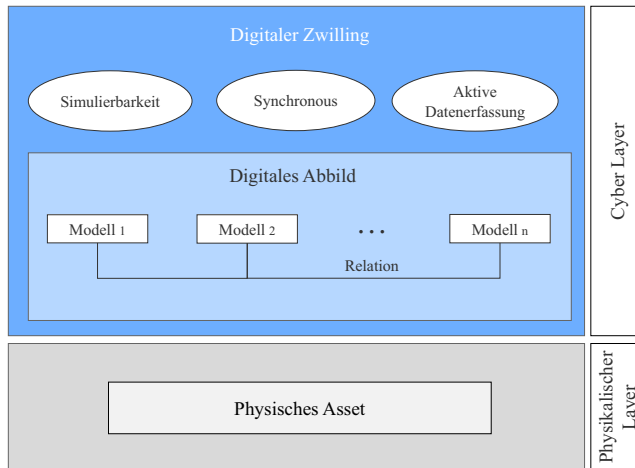


Abbildung 10: Der Digitale Zwilling und seine Eigenschaften im cyber-physischen Produktionssystem nach [6]

Im nächsten Abschnitt wird ein Überblick über die konkreten Bestandteile des Digitalen Zwillings innerhalb seiner Architektur gegeben. Dafür werden zunächst die vordefinierten Architekturen des Digitalen Zwillings untersucht und diese, basierend auf die Erfüllung der Eigenschaften eines Digitalen Zwillings, entschärft.

Ausgehend von den einschlägigen Literaturquellen wird in diesem Zusammenhang eine allgemeingültige Architektur vorgestellt.

3.1.2 Architektur des Digitalen Zwillings

Bisher wurden in der Literatur nur wenige Architekturen für Digitale Zwillinge vorgestellt. Malakuti und Grüner benennen vier architektonische Aspekte eines Digitalen Zwillings: Interne Struktur, Schnittstellen, Integration und Laufzeitumgebung [99]. Der Aspekt der internen Struktur und des Inhalts befasst sich mit folgenden Anforderungen:

- Dem Metamodell für den Digitalen Zwilling mit seinen internen Modellen und deren Beziehungen.
- Potenziale zur Anbindung an bestehende Tools.
- Möglichkeiten zur Modularisierung der Inhalte des Digitalen Zwillings und zur Erweiterung der Inhalte.
- Standards, die bei der Festlegung des Inhalts des Digitalen Zwillings zu verwenden sind, um den Informationsaustausch zu realisieren.

- Mögliche Umsetzung vorhandener Informationen in diese Standards.

Der Aspekt der APIs und der Nutzung beleuchtet die Schnittstellen des Digitalen Zwillings sowie die Gesichtspunkte der Kommunikationsarchitektur von Digitalen Zwillingen wie Cloud-to-Device-Kommunikation oder Zugriffsberechtigung auf Informationen des Digitalen Zwillings.

Der Aspekt der Integration betrachtet:

- Einen Identifizierungsmechanismus zur eindeutigen Identifizierung des realen Assets
- Mechanismen zur Identifizierung von neuen realen Assets in einem Gesamtsystem zur Verbindung mit ihrem Digitalen Zwilling und zur Synchronisation des Digitalen Zwillings mit dem realen Asset.
- Einen Mechanismus zum Kombinieren mehrerer Digitaler Zwillinge zu einem Digitalen Zwilling bezüglich eines Gesamtsystems.

Darüber hinaus wird unter dem Aspekt der Laufzeitumgebung gefordert, dass Möglichkeiten zur Speicherung und Verteilung der Inhalte des Digitalen Zwillings, zum Austausch und zur Synchronisation der Inhalte über die Unternehmensgrenzen hinweg, zum sicheren Zugriff auf die Inhalte und zur Kommunikation von Informationen vom Digitalen Zwilling an das reale Asset geschaffen werden. Dies ist jedoch nur eine Diskussion hinsichtlich einer Sammlung von Anforderungen, ohne dass konkrete Umsetzungsvorschläge vorliegen. Wie die einzelnen Aspekte konzeptionell aussehen müssen oder wie sie realisiert werden können, wird nicht diskutiert. Es wird nur auf die Plattform „Industrie 4.0 Administration Shell“ verwiesen, die auch einige dieser Aspekte aufgreift, aber nicht alle diskutiert [100]. Haag und Anderl definieren den Digitalen Zwilling als eine digitale Repräsentation der Eigenschaften, Zustände und Verhaltensweisen eines realen Assets, die durch Modelle und Daten realisiert wird [101]. Allerdings ist auch hier keine konkrete Architektur angegeben. Dasselbe gilt für Boschert und Rosen, welche ebenfalls keine konkrete Architektur des Digitalen Zwillings definieren, sondern sich auf die Definition von relevanten Modellen und Daten beschränken und die konkrete Gestaltung des Digitalen Zwillings dem Entwickler überlassen [42]. Außer [6] erfüllt keine der in der Literatur vorgestellten Ansätze alle im Abschnitt 3.1.1 genannten Aspekte des Digitalen Zwillings. Die meisten von ihnen behandeln auch nur einige dieser Aspekte, aber keine außer der in [6] vorgeschlagene Architektur definiert ein konkretes Konzept und eine Realisierung. Die meisten Architekturen behandeln nur einige dieser Aspekte und in keinem der Ansätze - außer die vorgeschlagene Architektur in [6] - wird ein konkretes Konzept für eine Realisierung beleuchtet. Aus diesem Grund wird in dieser Arbeit die Synchronisierung der Modelle innerhalb des Digitalen Zwillings nach der vorgeschlagenen Architektur in [6] betrachtet.

Die Bestandteile eines Digitalen Zwillings (DZ) sind in Abbildung 11 nach [6] dargestellt. Ein Digitaler Zwilling besteht aus Modellen und zugehörigen Schnittstellen zu Tools, dem Modellversionsmanagement, den Betriebsdaten der physischen Anlage, der Organisation und den

technischen Daten des realen Assets, den Informationen über seine Relationen zu anderen DZs in der realen Welt, einer eindeutigen Identifizierungsnummer und einer Schnittstelle zur Kommunikation mit anderen DZs sowie einer Schnittstelle zur Kommunikation mit der realen Welt. Im Folgenden werden diese jeweils erläutert.

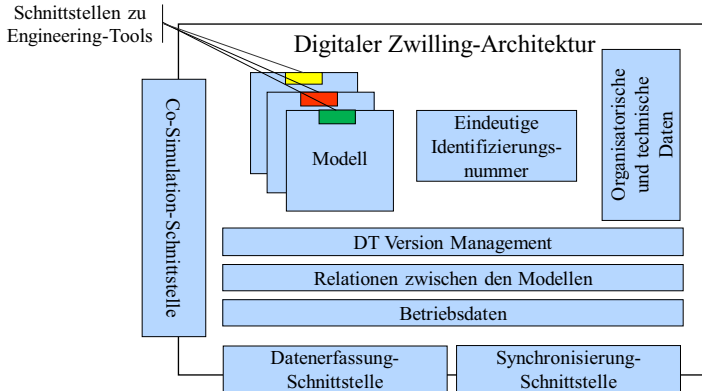


Abbildung 11: Architektur des Digitalen Zwillings in Anlehnung an [6]

Eindeutige Identifizierungsnummer (ID): Der Digitale Zwilling eines physischen Assets muss durch eine Identifikationsnummer in seiner Architektur eindeutig identifizierbar sein [99]. Dementsprechend ist eine eindeutige Identifikationsnummer innerhalb der Digitalen Zwillings-Architektur zwingend notwendig. Mit Hilfe dieser eindeutigen ID werden die Bestandteile des Digitalen Zwillings in einer Datenbank gespeichert. Diese können während des Engineerings oder der Rekonfiguration jederzeit aufgerufen werden.

Modelle und entsprechende Schnittstellen zu Tools: Der Digitale Zwilling beinhaltet alle Modelle und (Betriebs-) Daten des Realen Assets sowie eine Schnittstelle zwischen den Tools und deren Modelle [42]. Der Digitale Zwilling kapselt die domänenübergreifenden Modelle eines Assets, beispielsweise das CAD-Modell, das Elektroschaltplanmodell, das Softwaremodell und die Funktionsmodelle oder Simulationsmodelle [42] [94] [99] [101]. Jedes dieser Modelle wird während des Entwicklungsprozesses des Digitalen Zwillings mit einem speziellen Tool erstellt. Ein wichtiges Merkmal sind daher Schnittstellen zwischen den Tools und deren Modellen. Die Schnittstellen zu den Tools werden für die Interaktion mit den Modellen genutzt. So können beispielsweise die Modelle während des gesamten Lebenszyklus aktualisiert, angepasst oder domänenspezifisch mit Hilfe verschiedener Eingaben simuliert werden [6].

Betriebsdaten: Um das Verhalten und den aktuellen Zustand des Systems genau wiedergeben zu können, muss der Digitale Zwilling die aktuellen Betriebsdaten des Systems enthalten [101] [102]. Dazu gehören Sensordaten, die kontinuierlich aufgezeichnet werden, sowie Steuerdaten, die den

aktuellen Zustand der realen Komponente bestimmen, die auch über den gesamten Lebenszyklus aufgezeichnet werden. Auch neue Aufträge und andere Geschäftsdaten lassen sich so speichern.

Schnittstelle zur Datenerfassung: Ein Digitaler Zwilling erfordert eine Schnittstelle in seiner Architektur, um mit dem realen Asset zu kommunizieren und die aktuellen Betriebsdaten des Systems zu erfassen [99]. Über diese Schnittstelle können die Betriebsdaten vom Digitalen Zwilling übertragen und aufgezeichnet werden [6]. Darüber hinaus ist eine der Funktionalitäten des Digitalen Zwillings, die Möglichkeit zur Ausführung einer parallelen Simulation zum realen Asset während des Betriebs. Daher müssen die Betriebsdaten der Sensoren und Aktoren des realen Assets über diese Schnittstelle innerhalb des Digitalen Zwillings übertragen werden, sodass jede Form der Simulation dynamisch durchgeführt werden kann.

Organisatorische und technische Daten: Der Digitale Zwilling beinhaltet die organisatorischen Daten des realen Assets [99]. Diese Daten enthalten Informationen über die reale Anlage, wie z.B. wann sie hergestellt, von wem sie hergestellt, entworfen und entwickelt sowie wann und von wem sie in Betrieb genommen wurde usw. Zusätzlich werden alle während des Lebenszyklus der physischen Anlage erstellten Dokumentationen gespeichert. Dazu gehören Dokumente, die während der Entwicklungsphase erstellt werden, wie z.B. Pflichtenhefte, Entwurfslayouts usw. sowie Dokumentationen während des Betriebs wie z.B. Wartungsberichte.

Relationen (zu anderen Digitalen Zwillingen): In einem Digitalen Zwilling werden die aktuellen und alten domänenübergreifenden Modelle gespeichert, die mit anderen Modellen anderer Digitaler Zwillinge in Beziehung stehen [99], [101], z.B. Instanz, Vererbung, Parent-Child-Relationen usw. Damit das gesamte System konsistent bleibt, müssen diese Relationen in der Architektur des Digitalen Zwillings gespeichert werden, da sonst das gesamte System, bestehend aus vielen Digitalen Zwillingen, die erforderliche Konsistenz nicht sicherstellen kann.

Version-Management: Der Digitale Zwilling sollte nicht nur aktuelle Modelle enthalten, sondern auch alle alten Modelle aus dem gesamten Lebenszyklus [30], [42], [102]. Dies unterstützt ein effizientes Engineering bei der Rekonfiguration und die Erweiterbarkeit über den gesamten Lebenszyklus. Die Versionsverwaltung des Digitalen Zwillings greift auf alle gespeicherten Versionen der Modelle und deren Beziehungen zu. So kann die alte Version auf Wunsch des Ingenieurs unter Berücksichtigung der Gegebenheiten während des Engineerings oder der Rekonfiguration jederzeit abgerufen und auf die benötigte Version umgestellt werden.

Co-Simulation-Schnittstelle: Digitale Zwillinge müssen Informationen untereinander austauschen [99]. Außerdem ist Simulation ein wichtiger Aspekt des Digitalen Zwillings [42] und eine Co-Simulationsfunktionalität zwischen den heterogenen Modellen der Digitalen Zwillinge notwendig [103], um eine domänenübergreifende Simulation des gesamten Systems durchführen zu können. Dadurch ist innerhalb der Architektur des Digitalen Zwillings eine Schnittstelle zur Co-Simulation zwischen den Digitalen Zwillingen erforderlich [103]. So kann beispielsweise

durch einen Datenaustausch eine domänenübergreifende Co-Simulation ermöglicht und der Prozessablauf des gesamten realen Produktionssystems simuliert werden.

Schnittstelle zur Synchronisierung von Modellen und deren Relationen: Ein physisches Asset und seine physischen Abhängigkeiten zu anderen Assets (wie Verkabelung, physische Fixierungsposition usw.) können während seines Lebenszyklus sehr oft geändert werden [70]. Daher ist eine Schnittstelle in der Architektur des Digitalen Zwillings notwendig, um domänenübergreifende Modelle und deren Abhängigkeiten in einem Digitalen Zwilling synchronisieren zu können [6].

In diesem Abschnitt wurden die Komponenten der Digitalen Zwilling-Architektur im Detail beschrieben. Um die domänenübergreifenden Modelle und ihre Relationen innerhalb eines Digitalen Zwillings zu erstellen, wurden im Stand der Wissenschaft und Technik diverse Ansätze diskutiert und durch zahlreicher Frameworks umgesetzt. Diese Ansätze, ihre implementierten Frameworks und ihre Funktionalitäten wie automatisierte Modellanpassung innerhalb der Modelle, automatisierte Konsistenzprüfungen, Modellversionierungsfähigkeit usw. sind von großer Bedeutung im Zusammenhang mit der Synchronisierung der Modelle des Digitalen Zwillings eines automatisierten Systems über dessen Betriebsdauer. Bei diesen Ansätzen muss untersucht werden, welche ihrer Funktionalitäten die Herausforderungen bei der Synchronisierung der Modelle des Digitalen Zwillings nach der Inbetriebnahmezeit des Systems beheben können. Im nächsten Unterkapitel werden die bestehenden Ansätze zur Erstellung domänenübergreifender Modelle des Digitalen Zwillings während des Engineering-Prozesses, deren Funktionalitäten und Technologien im Detail analysiert und anhand der Kriterien zur Synchronisierung der Modelle des Digitalen Zwillings bewertet.

3.2 Ansätze zur Integration der domänenübergreifenden Modelle

Wie schon in Abschnitt 3.1.2 beschrieben, ist die Integration der domänenübergreifenden Modelle, die mit verschiedenen Tools erstellt werden, eine Voraussetzung zur Konzipierung des Digitalen Zwillings während des Engineering-Prozesses. In diesem Zusammenhang wurden verschiedene Ansätze zur Integration der domänenübergreifenden Modelle im Stand der Wissenschaft vorgestellt, die in zahlreichen Frameworks mit verschiedenen Technologien wie SysML, UML, AutomationML, Ontologien, etc. realisiert wurden. [104] wartet mit einem Überblick über viele Ansätze auf. Hierbei sind, unter Berücksichtigung der in Kapitel 1 definierten Anforderungen an die automatisierte Anpassung der geänderten domänenübergreifenden Modelle innerhalb des Digitalen Zwillings, die Technologien zur Konsistenzprüfung zwischen den Modellen sowie deren Versionierung und Wiederverwendbarkeit von großer Bedeutung. Dieses Unterkapitel gibt einen Überblick über

diese Ansätze und ihre Technologien zur fachübergreifenden Modellintegration innerhalb des Digitalen Zwillings.

3.2.1 AutomationML-basierte Ansätze

Biffel et al. und Winkler et al. stellen in [62], [105] einen Ansatz vor, der auf dem neutralen Austauschformat AutomationML (AML) in Kombination mit Model-Driven Engineering (MDE) basiert. In diesem Ansatz wird für die Verknüpfung domänenübergreifender Modelle eine Integrationsplattform, AML.hub, vorgestellt. Der AML.hub ist eine Integrationsplattform als Single-IT-System, das das Mapping und Merging von Modellen aus verschiedenen Domänen unterstützt sowie Analyse-Tools für die Visualisierung, Konsistenzprüfung und das Versionsmanagement der Modellen bereitstellt [105]. In diesem Ansatz übernimmt die Integrationsplattform Informationen aus verschiedenen fachübergreifenden AML-Dateien, extrahiert AML-Datenelemente und erstellt einen Instanzhierarchiebaum mit Versionierungsfunktionalität, vgl. Abbildung 12.

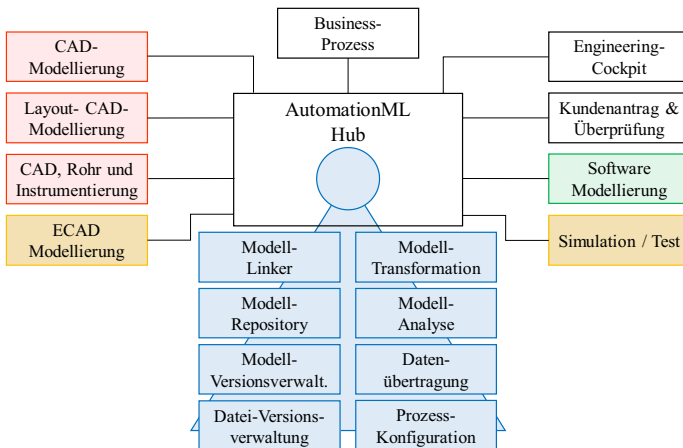


Abbildung 12: Integriertes Engineering mit der AML-Integrationsplattform, AutomationML Hub nach [139]

Darüber hinaus bietet dieser Ansatz eine Methode zur Konsistenzprüfung mit Schwerpunkt auf der Fehlererkennung [3]. Der Ansatz verwendet den AutomationML-Analyzer als Tool-Support, um Abfragen zu generieren, die automatisch Inkonsistenzen des verknüpften Engineering-Modells erkennen und an die Prüfer zur manuellen Korrektur berichten. Diese Vorgehensweise erfordert AML als gemeinsames Format für das gesamte Engineering sowie für das Verknüpfen von Tools und deren Modelle. Ein weiterer wichtiger Aspekt dieses Ansatzes ist die Möglichkeit der Wiederverwendung von Modellen. Die Wiederverwendung von Modellen während des

Engineering-Prozesses zielt darauf ab, die Zeit der Modellierung zu verkürzen [106]. Computer Aided Engineering Exchange (CAEX) wird als Top-Level-Struktur verwendet und dient für den Ansatz, der die Modellierung physikalischer und logischer Prozesse von Anlagenkomponenten als gekapselte Prototyp-Klassen ermöglicht. Ein wichtiger Vorteil dieser Kapselung besteht darin, dass sie die Wiederverwendung bestehender Komponenten durch Kopieren bestehender Prototyp-Klassen erleichtert.

Zudem haben Hildebrandt et al. und Scholz et al. in [107], [108] einen semantischen Modellierungsansatz für die Integration der domänenübergreifenden Modelle vorgeschlagen, um ein gemeinsames Informationsmodell des Systems zu schaffen. Sie haben ihren Ansatz in AutomationML umgesetzt. Der vorgeschlagene semantische Modellierungsansatz besteht aus zwei Hauptmetamodellen, einerseits aus einem System-Metamodell, das Informationen über das Funktions-, Struktur- und Verhaltensmodell der Systeme und mögliche Interaktionen zwischen diesen modelliert, andererseits aus einem Charakteristik-Metamodell, das zur Modellierung der Merkmale verwendet wird [107], vgl. Abbildung 13. Hierbei werden die Objekte innerhalb des Charakteristik-Metamodells nach Eigenschaft, Parameter und Zustand klassifiziert. Außerdem werden externe Bibliotheken (z.B. ecl@ss) zur Beschreibung von Objektscharakteristiken verwendet, um eine gemeinsame Semantik zwischen Systemmodellen zu ermöglichen.

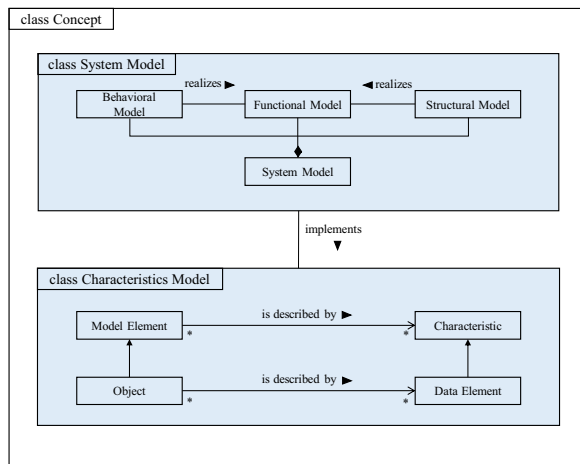


Abbildung 13: Semantischer Modellierungsansatzes mit System- und Charakteristik-Metamodell nach [107]

Die Versionierung steht nicht im Fokus der vorgeschlagenen Methode. Im Hinblick auf die Inkonsistenzprüfung bietet [109] jedoch eine Methode, die auf der systematischen Prüfung von fachübergreifenden Engineering-Relationen innerhalb wiederverwendbaren Komponenten basiert. Engineering-Relationen beziehen sich auf die inneren domänenübergreifenden Abhängigkeiten innerhalb einer wiederverwendbaren Komponente während des Engineering-

Prozesses und können in verschiedene Klassen hinsichtlich ihres technischen Charakters eingeteilt werden.

Hierbei sind die wiederverwendbaren Komponenten nach der funktionalen Zerlegung strukturiert, da jedes funktionale Subsystem eine wiederverwendbare Komponente enthält. Bei diesem Ansatz besteht ein Strukturmodell aus verschiedenen Varianten von wiederverwendbaren Komponenten, um ein breites Spektrum von Kundenanforderungen zu erfüllen [109].

3.2.2 SysML-basierte Ansätze

Thramboulidis stellt in [110], [111] ein sogenanntes 3+1 SysML-View-Modell vor, das sich mit der Integration der domänenübergreifenden Modelle von mechatronischen Systemen befasst, basierend auf vier System Views, der mechatronische-, mechanische-, elektrische- und Software-View. Der Grundgedanke dabei ist, dass ein SysML-Modell, speziell der mechatronische View, die Integration von domänenübergreifenden Modellen aus den anderen 3 Views ermöglicht. Dieses Modell deckt das Gesamtsystemmodell und alle Beziehungen zwischen mechanischen, elektrischen, Software- und Systemanforderungen ab. Die Systems Modeling Language (SysML) ist eine grafische, auf UML basierende, standardisierte Modellierungssprache und wurde entwickelt, um die verschiedenen Modellierungssprachen zu vereinheitlichen, die derzeit von Systemingenieuren verwendet werden. Sie unterstützt die Spezifikation, Analyse, Design, Verifikation und Validierung komplexer Systeme, die Hardware, Software, Informationen, Prozesse, Personal und Einrichtungen umfassen können [112].

Jede der anderen drei Views dient dazu, das System aus der Perspektive der entsprechenden Domain, d.h. Mechanik, Elektronik und Software, als Modell zu beschreiben. Hierbei wurde für die Modellausführung und Analyse der Modelle des mechatronischen Systems spezifische Tools jeder Domäne eingesetzt. Der vorgeschlagene Integrationsansatz bietet eine Methode zur Erhaltung der Konsistenz zwischen den Modellen des Systems. Hierbei wird die Konsistenz zwischen den Domänen durch das SysML-Modell gewährleistet und die Rückverfolgbarkeit zwischen verschiedenen Objekten, Anforderungen sowie domänenübergreifenden Abhängigkeiten unterstützt. Durch die drei Ansichten der zu modifizierenden Objekten wird die Aktualisierung vereinfacht [9]. Anders als bei der Konsistenzprüfungsmöglichkeit stehen die Wiederverwendung, Versions- und Variantenverwaltung von Modellen während des Engineering-Prozesses und die Modellintegration nicht im Fokus dieses Ansatzes.

Des Weiteren präsentiert Kernschmidt in [113] und [114] einen SysML-basierten Ansatz „SysML4Mechatronics“, der zur Erstellung eines zentralen Systemmodells verwendet wird. In diesem werden die verschiedenen fachspezifischen Modelle (d. h. mechanische, elektrische und Software-Modelle) und die domänenübergreifenden Abhängigkeiten sowie Einflüsse zwischen den Modellen explizit aufzeigt. Um die Zusammenstellung und Wiederverwendung von Modellen zu erleichtern, bietet der Ansatz eine funktionsorientierte Modularisierung. In diesem Ansatz

können die Module entsprechend der Domäne individuell modelliert und zu funktionalen domänenübergreifenden Modulen kombiniert werden. Darauf basierend präsentieren Kernschmidt et al und Li et al in [115], [104] einen fachübergreifenden Engineering-Workflow, der auf einer Kombination aus der SysML4Mechatronics-Modellierungssprache, auf einer Inkonsistenzmanagement-Software und einem PLM-System aufbaut, vgl. Abbildung 14. In [116] definiert Finkelstein das Inkonsistenzmanagement als den Prozess, bei dem Inkonsistenzen zwischen Modellen so behandelt werden, dass sie die Ziele der betroffenen Interessengruppen unterstützen. Die in diesem Engineering-Workflow entwickelte Inkonsistenzmanagement-Software funktioniert auf Basis einer regelbasierten Konsistenzprüfung im Sinne der Inkompatibilität geänderter Komponenten während des Engineerings. Dabei werden die Regeln aus fachspezifischen Vorgaben, Dokumenten und Expertenerfahrung (z.B. Kompatibilität zwischen den Komponenten) abgeleitet [104]. Gemäß dem in [104] beschriebenen Engineering-Workflow erfolgt die Versionsverwaltung über das in den gesamten Workflow integrierte PLM-System, das als Datenbank für das Informationsmanagement dient. Ebenfalls schlagen Abid et al in [117] einen domänenübergreifenden Modellintegrationsansatz vor, der auf der Integration von Modellen mechatronischer Systeme in ein PLM-System mittels SysML-Diagrammen basiert.

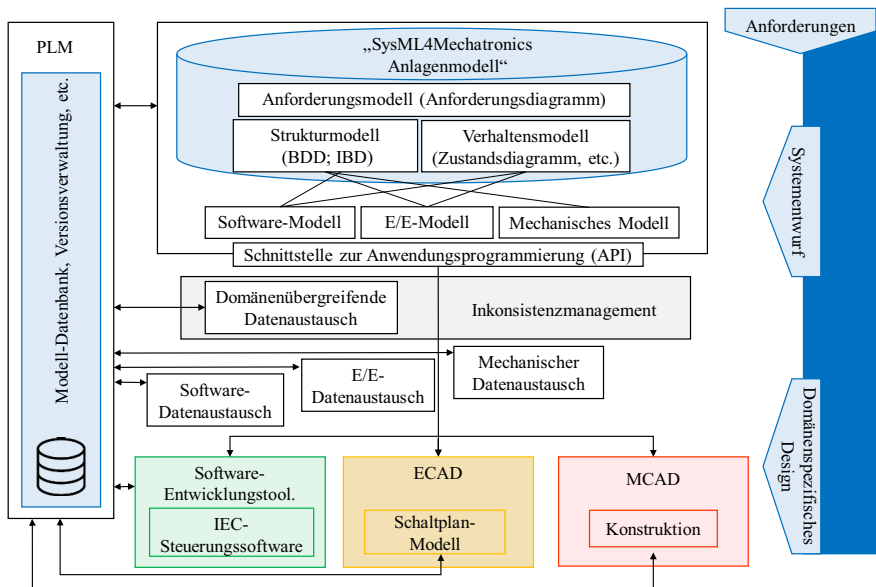


Abbildung 14: Integrierter fachübergreifender Engineering-Workflow nach [104]

Das PLM-System dient hier dazu, den Austausch der verschiedenen Modelle zu koordinieren und zu integrieren, die von den verschiedenen Designern und Tools erstellt wurden. Darüber hinaus ermöglicht das PLM-System auch die Wiederverwendung und Reversion der Modelle. Der vorgestellte Ansatz basiert auf der Nutzung der SysML-Anforderungsdiagramme, SysML-

Strukturdiagramme und SysML-Verhaltensdiagramme zur Modellierung eines zentralen Struktur-Modells des Systems in einem PLM-System. Bei diesem Ansatz wird eine strukturierte Modellierung des mechatronischen Produktionssystems, des PLM-Systems selbst und des im Produktionssystem im PLM-System produzierten Produkts durch die Verwendung von SysML-Strukturdiagrammen wie Block-Definition-Diagrams (BDD) und Internal-Block-Diagrams (IBD) ermöglicht. Hier bestehen die Produkt- und Produktionssystem-Struktur im PLM-System aus sogenannten Parts, die eine Komponente oder eine Baugruppe von Komponenten aus verschiedenen Domänen sein können. Darüber hinaus können diese strukturierten Parts auch in CAD-Modellen (3D-Modelle, Software-Modelle, Skizzen-Modelle, elektrische Schaltungsmodelle, Schaltplane-Modell und mehr) und Sekundärdaten wie dem Exportformat von Modellen referenziert werden.

Schließlich wird unter Verwendung der SysML-Anforderungsdiagramme und SysML-Verhaltensdiagramme eine strukturierte Modellierung von Kundenanforderungen und Produktentwicklungsprozessen im PLM-System bereitgestellt.

3.2.3 Ontologie-basierte Ansätze

Abramovici et al. schlagen in [118] [119] einen Ansatz - Product-Service-System (PSS) Lifecycle Management - vor, der die Integration von Engineering-Modellen, Engineering-Tools sowie die zugehörigen Services eines Systems während des gesamten Lebenszyklus anhand einer Ontologie in cyber-physische Systeme ermöglicht. Durch die Zusammenführung und Integration von Modellen des Systems und der Services fokussiert sich dieser Ansatz auf intelligente Systeme, die sich einfach rekonfigurieren lassen. Für die Integration von domänenspezifischen Modellen des Systems wird bei diesem Ansatz empfohlen, eine Ontologie zu verwenden, die aus zwei Schichten besteht. Die erste Schicht wird als "Top-Level-Ontologie" bezeichnet und beschreibt die domänenübergreifenden Modelle des Produkts, die Beziehungen zwischen diesen und den Modellen von Services, Anforderungen, Business, Ressourcen, Funktionen, technische Daten sowie Supportinformationen und einigen weiteren. Die zweite Schicht enthält weitere Ontologien, die für die Ausführung intelligenter Produkt Lifecycle Management-Methoden von Bedeutung sind. (z.B. adaptives Änderungsmanagement). Darüber hinaus schlägt dieser Ansatz drei wissensbasierte Assistenten für die Modellintegration, das Änderungsmanagement und die Konsistenzprüfung vor: Intelligente Änderungsassistenten, die die Parameteränderungen (Prozessänderungen) des realen Systems anhand der Konsistenzprüfung zwischen den Modellen verwaltet; Feedback-Assistenz, die Wissen aus dem Betrieb generiert und ein Feedback zur gesamten Ontologie liefern, um die Optimierung des bestehenden Produkts zu unterstützen; und Entscheidungshilfe-Assistenz, die den strategischen Entscheidungsprozess durch die Erfassung verschiedener Datenquellen von Anbieter, Kunde und Lieferant begünstigt. Um die Konsistenz bei der Modellintegration zwischen verschiedenen Tools und domänenspezifischen Systemen zu gewährleisten, wird von der gleichen Forschungsgruppe in [119] ein semantisches integriertes

Informationsmodell vorgestellt, das durch eine Ontologie realisiert wird. Diese Ontologie besteht aus fünf Elementen: Smart-Produkt, Smart-Produktinstanz, Prozess, Ressourcen und Organisation. Smart-Produkt, das verschiedene Aspekte des Systems beschreibt und enthält, einschließlich der Anforderung, Funktion, Gerätestruktur, Geometrie usw.; jede Instanz des Smart-Produkts ist direkt mit dem realen Asset verbunden und bildet das Konzept der Smart-Produktinstanz.

Prozesse stellen fachübergreifende Aktivitäten des Engineering-Prozesses dar, in den Ressourcen alles beschreiben, was während eines Prozesses verwendet wird; Eine Ressource kann eine Personalressource (z.B. Mitarbeiter), eine Informationsressource (z.B. Dokumente) oder eine physische Ressource (z.B. Werkzeuge, Ersatzteile) sein. Die Organisation wird verwendet, um die Zusammenarbeit verschiedener Bereiche im unternehmensübergreifenden Netzwerk zu modellieren.

3.2.4 Toolset-basierte Ansätze

Harrison et al. stellen in [120][121] einen systematischen Ansatz „VueOne-Engineering Toolset“ vor, der den gesamten Lebenszyklus der verteilten komponentenbasierten Automatisierung vom Design über den Betrieb bis hin zur Rekonfiguration unterstützt. VueOne kann in drei Fällen die Modellintegration und den Datenaustausch über den Systemlebenszyklus bereitstellen: Innerhalb der verschiedenen Lebenszyklusphasen des Systems, zwischen digitalen Modellen und realen Assets sowie zwischen domänenübergreifenden Domänen und spezifischen Ingenieuren, Tools und Organisationen. Ein wesentliches Merkmal der Entwicklungsumgebung von VueOne ist ein gekapseltes Komponentendatenmodell, das sich über den gesamten Systemlebenszyklus erstreckt und sowohl die multidimensionalen Systemmodelle als auch die Betriebsdaten gekapselt beinhaltet. Das bedeutet, dass die Daten innerhalb einer Softwarekomponente integriert sind, die auf einer vom Benutzer definierten Granularitätsstufe existieren kann. Das gekapselte Komponentendatenmodell im VueOne-Ansatz umfasst verschiedene Domänen, z.B. die mechanische und Softwaredomäne. Das Konzept des Komponentendatenmodells in VueOne erleichtert die Wiederverwendbarkeit von Modellen, indem es einen wiederverwendbaren Datensatz für ein Subsystem eines Systems definiert. Während der Entwurfsphase wirkt das VueOne-Komponentendatenmodell als ein gemeinsames Repository von designbezogenen Modellen. Im Rahmen des Versions- und Modellmanagements können Komponentendatenmodelle mit dem VueOne-Komponenten-Editor bearbeitet werden, in dem Bibliotheken auch für bereitgestellte, wiederverwendbare Komponenten enthalten sind. Die Komponentendatenmodelle werden schließlich durch ein Systemdatenmodell ergänzt, das die Systemarchitektur definiert und das Konsistenzmanagement erleichtert.

Basierend auf diesem Ansatz wird in [121] ein Framework vorgestellt. Dieses Framework besteht aus einem Modul, „VueOne deployment module“, zur Verwaltung der Konsistenz zwischen dem digitalen Softwaremodell und dem realen Steuerungssystem. Dieses Modul ermöglicht die

automatisierte Generierung von Steuerungssoftware für reale Steuerungssysteme unter Verwendung des aktuellen Softwaremodells im VueOne-Komponentenmodell sowie das Speichern von Steuerungsbetriebsdaten im VueOne-Komponentenmodell über OPC-UA-basierte Runtime-Schnittstellen. Ein Mapper-Modul ermöglicht die Zuordnung zwischen Funktionsblöcken, I/O- und Memory-Adressen sowie die Speicherung und Versionsverwaltung von Informationen.

Im Anschluss an die Beschreibung der bestehenden Ansätze zur Integration von domänenübergreifenden Modellen des Digitalen Zwillings in Unterkapitel 3.2 wird in Kapitel 3.5 eine zusammenfassende Bewertung der Ansätze im Hinblick auf die Erfüllung der Forschungsanforderungen vorgestellt.

3.3 Ansätze zur Synchronisierung der Modelle eines automatisierten Systems

Im Stand der Wissenschaft und Technik gibt es zahlreiche Ansätze, die zwei Aspekte einbeziehen,

- (1) die automatisierte Änderungserkennung aus dem realen System und
- (2) die automatisierte Anpassung der Modelle eines bestehenden Systems, die während des Engineering-Prozesses entwickelt wurden.

Diese Ansätze werden als Synchronisationsansätze bezeichnet. Im folgenden Kapitel sind die Ansätze nach den eingesetzten Technologien in vier Kategorien eingeteilt, Ansätze mittels 3D-Laserscans, Ansätze mittels Netzwerkscans, Ansätze auf Basis der Kommunikationsnetzanalyse und Ansätze mittels Analyse der Änderungsdokumentation, die während des Betriebs des Systems durch Fachexperten erstellt wurde. Im Weiteren werden die bestehenden Ansätze, ihre Technologien und Potenziale für die Synchronisierung von den Engineering-Modellen in den Domänen, Mechanik, Elektrik und Software umfassend beschrieben.

3.3.1 3D-Laserscan-basierte Ansätze

Ein weit verbreiteter Ansatz zur Erstellung einer realistischen Visualisierung des aktuellen Zustands des Produktionssystems ist das 3D-Laserscanning. Mit einem 3D-Laserscan kann ein aktuelles, mechanisches Modell des Systems erstellt werden, auf dessen Basis die Dauer und das Risiko des Rekonfigurationsprozesses reduziert werden kann [122]. Die Ausrüstung, die für die Laserscans verwendet werden, sendet einen Laserstrahl aus und erfasst dessen Reflexionen, um den Abstand zu messen. Es werden Millionen von Reflexionen in wenigen Minuten erfasst. Um die gesamte Layoutstruktur des automatisierten Systems zu modellieren, wird der Laserscan an mehreren Stellen durchgeführt und durch einen automatisierten Erfassungsprozess kombiniert, woraus dann eine sogenannte Punktwolke des gesamten Produktionssystems resultiert.

Lindskog stellt in [123] eine Methode zur Synchronisation des Layoutplanmodells des Produktionssystems basierend auf einem 3D-Laserscan vor. Der vorgeschlagene Ansatz kann bei der Synchronisation der mechanischen Modelle innerhalb des digitalen Zwillings angewendet werden. Bei diesem Ansatz wird die eigentliche 3D-Punktwolke manuell mit den alten 2D- und 3D-CAD-Modellen des Layouts verglichen, um die Änderungen im System zu erkennen. Ziel der Methode ist die Erstellung eines 3D-Simulationsmodells des Systemlayouts durch Kombination des 2D- oder 3D-CAD-Modells mit der genauen Punktwolke. Dieses Simulationsmodell kann verwendet werden, um die Auswirkungen einer Installation auf die Anlage, auf den verfügbaren physischen Raum und auf die Arbeitsplatzgestaltung zu analysieren. Erdős stellt zwei Methoden zur Synchronisation der 3D-CAD-Modelle vor. Bei beiden Methoden geht es darum, die aus dem 3D-Laserscan resultierende Punktwolke zu nutzen, um das mechanische Modell des digitalen Zwillings zu aktualisieren. Bei der in [124] veranschaulichten Methode geht es jedoch darum, die Punktwolke mit CAD-Modellen abzustimmen, während die in [125] vorgestellte Methode überwiegend von der Punktwolke abhängig ist, ohne die Beteiligung der alten CAD-Modelle.

Abbildung 15 zeigt den Prozessablauf des CAD-Modellerkennungsprozesses gemäß [124]. In [124] wird zuerst das alte CAD-Modell in das STL-Format umgewandelt. Dann werden die STL-Modelle in Face Adjacency Graphs (FAG) zerlegt, wobei verschiedene Entitäten durch verschiedene FAGs repräsentiert werden. Die geometrischen Merkmale dieser Einheiten werden dann basierend auf den topologischen Eigenschaften manuell und rechnerisch definiert. Dann wird ein Feature-Graph definiert, der die externen Verbindungen und Interaktionen dieser Entitäten definiert.

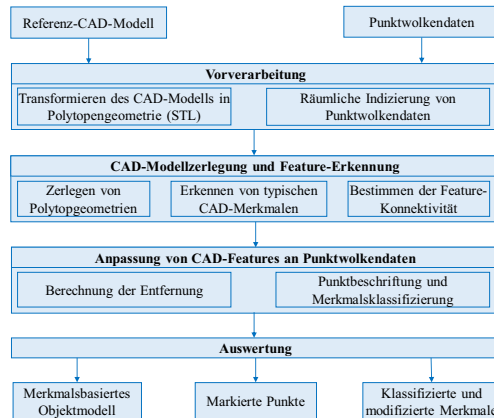


Abbildung 15: Prozessablauf des CAD-Modellerkennungsprozesses nach [124]

Zur Verkürzung der Bearbeitungszeit, wird eine parallele Berechnung durchgeführt, um die Merkmale verschiedener Einheiten gleichzeitig zu berechnen. Die Abgleichalgorithmen werden entwickelt, um die Punkte in der Punktwolke an die Merkmale der zerlegten CAD-Modelle

anzupassen. Dieser Algorithmus wird basierend auf der Messung der Entfernung jedes Punktes zur Oberfläche des CAD-Modells der Entität entwickelt. Der Punkt gehört zu dieser Entität, wenn die gemessene Entfernung innerhalb eines bestimmten Bereichs liegt. Hierfür sind manuelle Überprüfungen erforderlich, um die Genauigkeit des Ergebnisses zu gewährleisten.

Außerdem wird in [125] eine weitere Methode zur Erkennung von Objekten vorgestellt, die ausschließlich auf der Punktwolke basiert. Zunächst wird die Punktwolke gesammelt und in einem Standardformat gespeichert. Anschließend werden diese Punkte aufgeteilt und in Voxeln verwaltet. Ein Voxel kann als die Pixel in einer dreidimensionalen Umgebung betrachtet werden, mit denen ein Raum gleichmäßig in kleine Einheiten aufgeteilt werden kann. Die Punktwolke liegt in der 3D-Umgebung, und jeder Voxel erfasst die unterschiedliche Dichte der Punkte. Hierbei wird eine Filterung durchgeführt, um Voxel mit niedriger Punktdichte herauszufiltern. Topologien und Verbindungen verschiedener Objekte können basierend auf den Verbindungs-Voxeln definiert werden. Um hierbei den genauen Typ des Objekts zu erkennen, wird eine manuelle Erkennung durch Ingenieure mit Hintergrundwissen über das Produktionssystem durchgeführt. Die Plananpassung in Kombination mit der manuellen Erkennung (Ingenieure mit Hintergrundwissen über das Produktionssystem werden den Erkennungsprozess überprüfen) wird verwendet, um den Typ des Objekts zu erkennen. Die in [124] beschriebene Methode kann die genaue Position des gescannten Systems erhalten, aber die in [125] vorgestellte Methode erwähnte nicht das Erhalten von Ortsinformationen des Objekts. Beide Methoden können kombiniert werden, um ein eindeutig visualisiertes 3D-Modell des Produktionssystems zu erhalten. Allerdings erlauben es beide Methoden nicht, komponentenbezogene Informationen zu erhalten.

3.3.2 Netzwerkscan-basierte Ansätze

Biesinger in [126] stellt ein Konzept zur Synchronisierung des Bill of Equipment (BOE) und der CAD-Geometrie von der Anlage innerhalb des Digitalen Zwillings vor. Die BOE des automatisierten Systems ist eine Struktur aller Geräte, einschließlich ihrer Baugruppen und Unterbaugruppen, die im System verwendet werden. Das vorgeschlagene Konzept kann auf die Synchronisation der Modelle der mechanischen Domäne innerhalb des Digitalen Zwillings angewendet werden. Dieser Ansatz verwendet einen Netzwerkscan, um Informationen über vorhandene Geräte im System mit IP-Adressen zu erhalten. Diese Geräte werden allgemein als "intelligente Geräte" oder "IoT-Geräte" bezeichnet. Bei diesem Ansatz besteht der erste Schritt darin, eine aktuelle BOE aus dem Gesamtsystem mittels eines Netzwerkscans zu erstellen. Im nächsten Schritt wird das aktuelle BOE mit der detaillierten Anlagenstruktur aus der Engineering-Phase verglichen und die neu hinzugefügten oder entfernten Geräte aktualisiert. Da die Equipments im Netzwerk keine Positionsinformationen bezüglich der Gesamtanlage im Produktionssystem zur Synchronisierung des Geometrie CAD-Modell enthalten, wird in diesem Konzept mit einem 3D-Laserscanner eine 3D-Punktwolke des gesamten Systems erzeugt. Im letzten Schritt wird die Punktwolke manuell mit dem synchronisierten BOE verglichen und darauf

basierend ein aktualisiertes Geometrie-CAD-Modell mit Positionsinformation erstellt. Mit diesem Ansatz können die verschiedenen Komponenten, die im automatisierten System ohne IP-Adresse vorhanden sind, nicht durch den Netzwerk-Scan erkannt werden, was auf die Einschränkungen des Verfahrens zur Erstellung eines genauen Topologie-Modells der Anlage hinweist.

3.3.3 Kommunikationsnetzwerkscan-basierte Ansätze

Zipper präsentiert in [29] den Ansatz „Prozessüberwachungsmethode“ zur Erkennung der Änderung und Synchronisierung des Verhaltens- und Prozessablaufmodells innerhalb des Digitalen Zwillings. Durch einen Vergleich der realen Betriebsdaten mit den simulierten Betriebsdaten aus der parallellaufenden Anlagensimulation werden die Änderungen bestimmt. Die vorgeschlagene Methode fokussiert die Synchronisierung der Anlagensimulation mit der Anlage, die die Überwachung industrieller Anlage im Betrieb ermöglicht [127]. Das Kommunikationsnetzwerk transportiert Signale von der Steuerung zu den Aktoren und von den Sensoren zur Steuerung. In dieser Methode werden die Betriebsdaten aus dem Kommunikationsnetzwerk des physikalischen Systems in der Betriebsphase gesammelt und mit den Signalen der Anlagensimulation verglichen, um Abweichungen zu erkennen. Wie in Abbildung 16 dargestellt, wird bei dieser Methode z.B. das Signal von der Steuerung (u) in die

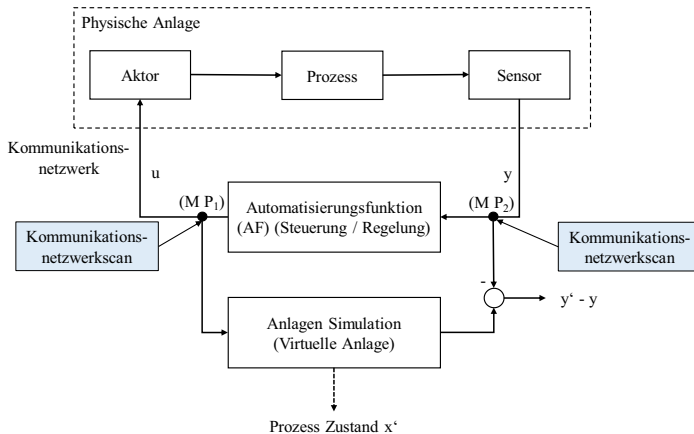


Abbildung 16: Prozessüberwachungsmethode nach [29]

Anlage am Messpunkt 1, MP1, erfasst und in Echtzeit in die Anlagensimulation eingebracht und aufgenommen. Durch das Signal von den Sensoren an die Steuerung in der Anlage (y) lässt sich die Reaktion der Anlage auf ein Steuersignal mit der Reaktion der Anlagensimulation auf dasselbe Signal vergleichen. Aus der Differenz lässt sich schließen, ob sich die reale Anlage geändert hat.

In diesem Konzept bildet der im Vergleich der physischen und simulierten Anlagen gefundene Unterschied die Grundlage für die drei weiteren Schritte ab: Die Identifizierung der Änderung,

die Diagnose des Änderungsgrundes und weitere Aktivitäten wie die Aktualisierung der Planungsmodelle und die Wartung [127]. Werden jedoch viele Sensoren und Aktoren dem System hinzugefügt oder entfernt, nimmt die Genauigkeit der Änderungserkennungsmethode ab.

3.3.4 Änderungsdokumentationsanalyse-basierte Ansätze

Koltun schlägt in [128] eine Methode zur Synchronisation des elektrischen Schaltplanmodells innerhalb des Digitalen Zwillings mittels automatisierter Änderungserkennung in Änderungsdokumenten von der realen Anlage vor. Dabei handelt es sich um eine Methode zur automatisierten Änderungserkennung aus den roten Änderungen in den Schaltplänen innerhalb der Änderungsdokumentation einer realen Anlage zur Synchronisation der elektrischen Schaltplanmodelle im Digitalen Zwillings. Der vorgeschlagene Ansatz kann für die elektrische Domäne des Digitalen Zwillings verwendet werden.

Wie in Abbildung 17 dargestellt, kann der Ansatz in zwei Schritten vorgenommen werden. Zunächst erfolgt das Erfassen von Daten aus den gescannten Dokumenten und anschließend die Modellierung und Darstellung der aus dem ersten Schritt gewonnenen Informationen. Hierbei wird die Bildverarbeitungstechnologie zur Informationsbeschaffung von Dokumenten eingesetzt. Anfangs wird das Rauschen des gescannten Dokuments basierend auf einem Kantenerkennungsverfahren in Abhängigkeit von der kritischen Kantengröße entfernt. Anschließend werden eine Symbolanalyse, Textanalyse und Verbindungsanalyse durchgeführt. In der Symbolanalyse wird eine geometrische Abgleichmethode angewendet, die eine Transformation der Dokumentmerkmale in die zuvor bekannten und in einer Datenbank gespeicherten Modellmerkmale der ECAD-Komponenten bewirkt. Die Verbindungsanalyse basiert auf einem Quadrat-Scan-Algorithmus, der zunächst die Richtung der Verbindung definiert und dann die Linie pixelweise über ein quadratisches Fenster verfolgt. Mit der Textanalyse sollen die Beschreibungen der Schaltpläne ausgelesen werden. Deshalb wird in diesem Ansatz für die Texterkennung eine Optical Character Recognition (OCR)-Technologie eingesetzt. Zuzüglich ist ein Retrieval-Management-System aufgebaut, um alle abgerufenen Informationen zu organisieren

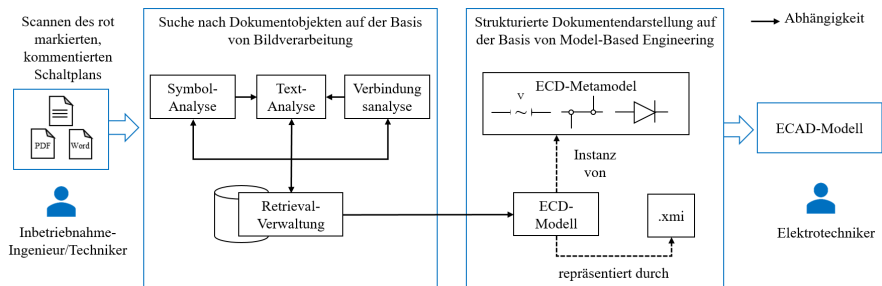


Abbildung 17: Methode zur Synchronisierung des elektrischen schematischen Modells des Systems nach [128]

und die Grenzen und Flächen der Analyse zu definieren. Für die Darstellung der abgerufenen Informationen wird der modellbasierte Ansatz verwendet. Dazu wird ein ECAD-Metamodell eingesetzt, das alle relativen Informationen aus Schaltplänen aufnimmt, die projektbezogene Informationen, blattbezogene Informationen (gemäß der Norm IEC 61082) und Schaltpläne enthält.

Im Rahmen der Synchronisierung des Digitalen Zwillings hat der vorgeschlagene Ansatz einige Einschränkungen. Zum einen kann der Ansatz nur für Schaltplänen und ECAD-Modellen in der elektrischen Domäne genutzt werden. Andererseits ist es mit dem vorgeschlagenen Ansatz nur möglich, mit gescannten, papierbasierten Schaltplänen umzugehen, für die der rote Bereich abgeändert wurde. Dementsprechend ist eine Änderungserkennung nicht möglich, wenn die Änderungen nicht dokumentiert werden.

3.4 Ansätze zur Modellgenerierung aus einem bestehenden automatisierten System

Neben Ansätzen zur Synchronisation der Modelle des Digitalen Zwillings sind auch die Ansätze zur automatisierten Modellgenerierung aus dem bestehenden System von Bedeutung. Diese Ansätze orientieren sich daran, dass kein hoher Aufwand für die Synchronisation von Modellen und deren Beziehungen innerhalb des erstellten Digitalen Zwillings während des Engineering-Prozesses erforderlich ist. Je nach den Anforderungen in den unterschiedlichen Phasen des Systemlebenszyklus können domänenspezifische Modelle wie 3D-CAD, System-Topologien und Simulationen aus dem System generiert werden. In diesem Kapitel werden diese Ansätze detailliert diskutiert.

3.4.1 Generierung eines 3D-CAD-Modells

Eine sehr verbreitete Methode zur Generierung eines aktuellen CAD-Modells aus einem bestehenden automatisierten System, insbesondere im Karosseriebau der Automobilindustrie, ist die Aufzeichnung der Punktwolkendaten des Systems. Anhand dieses Ansatzes schlägt Walla in [33] eine manuelle Methode zur Erzeugung eines CAD-Modells des Systems mit Technologien wie 3D-Tachymetrie und 3D-Laserscanning vor. Bei diesem Ansatz identifizieren die Fachexperten die einzelnen Komponenten des Systems durch manuelle Analyse der Punktwolkendaten. Im nächsten Schritt wird der Abstand zwischen den Komponenten mittels 3D-Tachymetrie-Technologien gemessen, um ein CAD-Modell des bestehenden Systems zu erstellen. Li schlägt in [129] aufgrund der Fortschritte in der GPU-Leistung und der Verfügbarkeit preiswerter hochauflösender Digitalkameras die Photogrammetrie als Alternative zum Laserscannen zur 3D-Modellerstellung vor. Der auf Photogrammetrie-Technologien basierende Ansatz charakterisiert weiterhin auch eine manuelle Bildanalyse, die von Fachexperten durchgeführt werden muss.

Son und Stark präsentieren in [130], [131] Ansätze zur automatisierten Generierung von 3D-CAD-Modellen durch systematischen Abgleich von systemgescannten Punktwolkendaten mit einer ausführlichen 3D-CAD-Datenbank von CAD-Equipment-Modellen. Son stellt in [130] einen Ansatz zur Generierung eines 3D-CAD-Modells von Anlagen aus gescannten Punktwolkendaten unter Verwendung einer CAD-Bibliothek vor und zeigt dessen Realisierung. Der Fokus dieses Ansatzes liegt auf der Generierung eines CAD-Modells des Gebäudes und des darin installierten Equipments im Bereich Prozessautomatisierung. Dennoch könnten der Ansatz und dessen Konzept auch für die Generierung des CAD-Modells von diskreten automatisierten Systemen geeignet sein. Um dieses Konzept anzuwenden, sind drei Kategorien von Vorkenntnissen über das System erforderlich: Szenenwissen, geometrisches Wissen und topologisches Wissen des Systems. Das Szenenwissen umfasst Informationen über alle vorhandenen Komponenten der Anlage, das geometrische Wissen über die geometrischen Merkmale dieser Komponenten und das topologische Wissen über die Zusammenhänge (Parent-Child-Relationen) dieser Komponenten sowie deren Sub-Komponenten. Das Szenenwissen kann aus Rohrleitungs- und Instrumentenfließschemata (R&I-Fließschema) des Systems abgeleitet werden. Das Schema enthält die Geräteliste, Rohrliste und Informationen über deren Funktionen und Zusammenhänge der Komponenten des Systems. Das geometrische Wissen ist aus einer 3D-CAD-Datenbank zu gewinnen. Dieses Wissen wird für einen Abgleich und eine Abfrage aus der 3D-Punktwolke verwendet. Das topologische Wissen kann aus einem computergestützten System abgeleitet werden, das das R&I-Fließschema automatisiert analysiert. Der Prozess der Generierung von CAD-Modellen besteht aus mehreren Schritten. Zunächst werden die 3D-Punktwolkendaten, in zweckmäßige Rohrleitungs- und Gerätesegmente unterteilt. Dieser Extraktionsprozess ist in [132] beschrieben. Der Autor präsentiert eine automatisierte Methode zur Berechnung von Krümmungen an bestimmten Stellen des Rohres sowie zu deren Abgleich mit dem geometrischen Vorwissen aus den R&I-Fließschema-Daten. Sobald die Rohrleitungen extrahiert sind, werden die übrigen Segmente, die die Geräte und Ventile repräsentieren, durch einen Gruppierungsprozess gruppiert. Anschließend werden die extrahierten segmentierten Daten aus Rohrleitungen, Geräten und deren Abhängigkeiten von den Punktwolkendaten und den Informationen aus den R&I-Fließschema-Daten zu einer sogenannten Topologie-Wissensrepräsentation zusammengeführt. Danach wird ein automatisierter Matching-Prozess zwischen dieser Topologie-Wissensrepräsentation und einer Geometrie-Wissensrepräsentation aus der 3D-Datenbank durchgeführt, die Höhe, Breite und Tiefe jeder Komponente des Systems enthält. Diesen Matching-Prozess beschreibt [133] detailliert. Die Kernidee dieses Matching-Prozesses besteht darin, die gemeinsamen Komponenten aus beiden Präsentationen zu finden. Während des Abgleichs wird die Ähnlichkeitsrate berechnet. Liegt dieser Wert unter einem bestimmten Prozentsatz, wird eine andere Komponente gewählt. Die am besten passenden Paare des obigen Prozesses zeigen eine erfolgreiche Übereinstimmung. Dieser Ansatz ist in der Praxis hilfreich, wenn keine Komponenten ohne detaillierte Änderungsdokumentation in das System

eingefügt oder entfernt werden. Ein aktuelles R&I-Fließschema aus dem System ist bei diesem Ansatz unerlässlich.

Stark stellt in [131] einen weiteren automatisierten Ansatz für den Reverse-Engineering-Prozess vor, um ein 3D-Assembly-Modell des bestehenden Produkts, wie z.B. Flugzeuge, Züge, Dampf- und Gasturbinen, große Generatoren sowie Industriekompressoren zu erstellen. Der vorgeschlagene Ansatz umfasst drei Schritte: 3D-Scan-Datensegmentierung, Part-Identifikation und Strukturidentifikation, vgl. Abbildung 18. Bei diesem Ansatz wird basierend auf dem 3D-Scanergebnis eine 3D-Scan-Datensegmentierung vorgenommen, um die Punktwolkendaten in mehrere „Parts“ zu trennen. Anschließend erfolgt in einer CAD-Datenbank eine Analyse, um die am besten passenden CAD-Modelle zu den segmentierten Punktwolkendaten zu finden. Damit kann jedem einzelnen Part des Systems ein individuelles CAD-Modell zugeordnet werden. Schließlich wird eine Analyse auf der Kontaktfläche der einzelnen Parts durchgeführt, um die Produktstruktur des sogenannten Assembly-Modells, zu erzeugen.

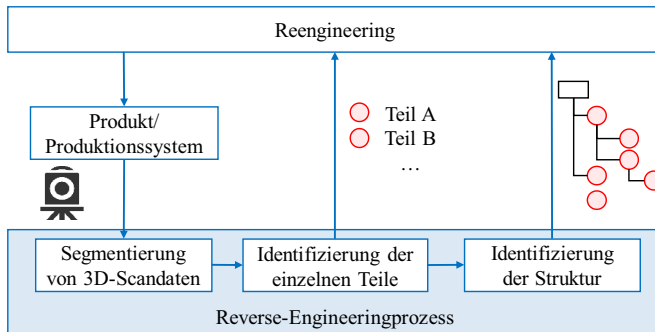


Abbildung 18: Reverse-Engineering-Prozess nach [131]

Hierbei wird die Segmentierung nach dem 3D-Scannen als erster Schritt durchgeführt. Die Herausforderung besteht jedoch darin, dass die 3D-Daten aufgrund der begrenzten Perspektive der Geräte ein unvollständiges Modell wiedergeben können. Hier schlägt der Autor die Computertomographie (CT-Scan) vor, um die unsichtbaren Geometrien für kleinere, skalierte Produkte zu erfassen. Jedoch ist auch der CT-Scan nicht in der Lage, alle Teile aus einem Assembly zu trennen, daher ist eine manuelle Trennung erforderlich [131]. Die Identifizierung der Parts erfolgt in diesem Ansatz über die CAD-Datenbank „CADENAS PARTsolutions“. Diese CAD-Datenbank bietet Funktionen für den automatisierten Abgleich von Punktwolkendaten mit 3D-Modellen durch eine geometrische Merkmalsuche. Schließlich wird die Produktstruktur bei diesem Ansatz durch ein zusätzliches Softwaresystem erstellt. Hierfür erfolgt zuerst das Laden des CAD-Modells in die Software, die eine Nachbarschaftsanalyse der angrenzenden Flächen durchführt und den Girvan-Newman-Algorithmus zur Erzeugung der Produktstruktur nutzt.

3.4.2 Generierung von Systemtopologien

Zur Erstellung eines Topologie-Modells der Steuergeräte und dezentralen Peripherien werden nach dem Stand der Technik verschiedene Tools verwendet. Diese Vorgehensweise kann zur Generierung von Topologien in den mechanischen und elektrischen Domänen des Digitalen Zwillings praktiziert werden. Der Netzwerkanalysator PRONETA ist eines der Tools zur Analyse von Profinet-Netzwerken und zum Testen von dezentralen Peripheriesystemen [134]. Profinet ist ein industrieller Technologiestandard für die Datenübertragung über Industrial Ethernet [135]. Die "Netzwerkanalyse" erstellt eine Topologie der am PROFINET angeschlossenen Geräte, die unter anderem aus dem Gerätenamen und den Netzwerkparametern besteht. Darüber hinaus kann dieses Tool ein Modell von den angeschlossenen Steuerperipheriemodulen an das System liefern. Wobei als Einschränkung bei der Anwendung des Tools, die Netzwerkgröße eine bestimmte Anzahl von Geräten nicht überschreiten darf [134]. Unter dem Einsatz von Netzwerkanalyse-Technologien stellt Biesinger in [136] einen Ansatz zur automatisierten Erstellung des Bill of Process (BOP) vor, der Roboter in einem automatisierten System, explizit in Karosseriebau der Automobilindustrie wirken lässt. Diese BOP kann zur Anlagenumbauplanung eingesetzt werden. Eine Roboter-Steuerungssoftware enthält Informationen über die von ihr gesteuerte Komponente, wie z.B. die Technologiesteuerung eines Greifers oder einer Schweißzange, sowie einige Informationen über aktuelle Fügepunkte, die in der Software definiert sind [136]. Der Ansatz basiert auf der Analyse der Robotersteuerungssoftware des bestehenden Systems unter Verwendung eines spezifischen Software-Parsers. Mit diesem Ansatz ist es möglich, die Position der Fügepunkte auf der Fahrzeugarchitektur, die Zykluszeiten für jeden einzelnen Prozess und schließlich das Strukturmodell der Prozessabläufe der Roboter im System zu generieren. Darüber hinaus schlägt der Autor vor, dass die Kombination dieses Ansatzes mit Steuerungskonfigurationstopologien, die aus dem Netzwerkscan erstellt werden können, eine Systemtopologie aus dem System in der mechanischen Domäne des Digitalen Zwillings liefern kann.

3.4.3 Generierung eines Simulationsmodells

Westkämper und Frank schlagen in [16] und [68] einen Ansatz zur automatisierten Generierung eines Simulationsmodells des System zur virtuellen Inbetriebnahme vor. Der Ansatz basiert auf der Analyse des aktuellen elektrischen Schaltplanmodells des Automatisierungssystems und der Nutzung einer Bibliothek, die die Verhaltensmodelle und CAD-Modelle der Komponenten beinhaltet. Das allgemeine Ziel der virtuellen Inbetriebnahme ist die Überprüfung der Steuerungssoftware des in Betrieb nehmenden Systems gegen die Verhaltensmodelle und 3D-CAD-Modelle der Komponenten, um die Reihenfolge und mögliche Kollisionen der Aktionen zu entdecken [16], [137]. Bei diesem Ansatz werden die Komponenten im Schaltplanmodell durch die Simulationsmodelle ersetzt. Die Informationen über die Verbindung zwischen den Komponenten im Simulationsmodell ergeben sich aus der Analyse der elektrischen

Verdrahtungen im Schaltplan. Zu diesem Zweck wurde ein Softwaresystem entwickelt, das die Verbindungen in elektrischen Schaltplänen in das Simulationsmodell überträgt.

Hierbei können auch die Geometrien und die kinematischen Modelle der Komponenten in der virtuellen Inbetriebnahme eingesetzt werden. Mit dem erzeugten Simulationsmodell kann die Steuerungssoftware jederzeit direkt verknüpft werden, um den Prozessablauf des Systems zu simulieren. Eigner stellt in [138] eine ähnliche Methode zur Generierung eines Simulationsmodells aus dem elektrischen Schaltplanmodell des Systems vor.

Nach der Diskussion der bestehenden Ansätze zur Integration domänenübergreifender Modelle des Digitalen Zwillings in Unterkapitel 3.2 sowie der bestehenden Ansätze zur automatisierten Synchronisierung und Neuerstellung des Digitalen Zwillings aus einem bestehenden automatisierten System in den Unterkapiteln 3.3 und 3.4 wird im Unterkapitel 3.5 eine zusammengefasste Bewertung der Ansätze hinsichtlich der Erfüllung des Forschungsbedarfs skizziert.

3.5 Zusammenfassende Bewertung und Schlussfolgerungen für die Konzeption

Im Hinblick auf die Synchronisierung der Modelle des Digitalen Zwillings nach der Inbetriebnahme eines automatisierten Systems wurden in dieser Arbeit, drei Herausforderungen verdeutlicht und von diesen anschließend drei Forschungsanforderungen abgeleitet. Diese Forschungsanforderungen lassen sich wie folgt zusammenfassen:

(A1) Detektion der Änderungsszenarien bis hin zur Feld-Ebene des automatisierten Systems.

(A2) Detektion der Abhängigkeiten von Änderungen am System.

(A3) Anpassung der Modelle innerhalb des Digitalen Zwillings.

Zur Identifizierung der Forschungslücke für die Synchronisierung der domänenübergreifenden Modelle des Digitalen Zwillings während des gesamten Lebenszyklus eines automatisierten Systems müssen die im Stand der Wissenschaft und Technik beschriebenen Ansätze und Technologien hinsichtlich dieser Anforderungen untersucht und bewertet werden.

Tabelle 1 fasst die in Unterkapitel 3.2 vorgestellten Forschungsansätze für die Integration von domänenübergreifenden Modellen des Digitalen Zwillings während des Engineering-Prozesses zusammen. Diese Tabelle gibt auch einen Überblick über die integrierten Funktionalitäten in diesen Forschungsansätzen und deren Framework, wie z.B. die Funktionalität für (1) domänenübergreifende Modellierung, (2) das Modellversionierung-Management, die Fähigkeit zum (3) Inkonsistenzmanagement zwischen den integrierten Modellen und (4) die Wiederverwendbarkeit der Modelle. Darüber hinaus wird die Modellierungssprache jedes dieser

Ansätze dargestellt. In dieser Tabelle sind all diese Ansätze anhand definierter Forschungsanforderungen dieser Arbeit bewertet.

Tabelle 1: Zusammenfassung und Bewertung der Forschungsansätze zur Integration domänenübergreifender Modelle des Digitalen Zwillings

Anforderungen Forschungsansätze	Integrierte Funktionalitäten in den Forschungsansätzen							
	(A1) Detektion der Änderungen bis hin zur Feld-Ebene	(A2) Detektion der Abhängigkeiten	(A3) Anpassung der Modelle	Domänenübergreifende Modellierung	Modellversionierung- Management	Inkonsistenz-Management	Wiederverwendbarkeit der Modelle	Modellierungssprache
Biffi et al. und Winkler et al. [105], [139]				+	+	+	+	AML
Hildebrandt et al., Scholz et al. und Schröck et al. [107]–[109]				+	-	+	+	AML
Thramboulidis [110], [111]				+	-	+	+	SysML
Kernschmidt et al. und Li et al. [104], [113]–[115]				+	+	+	+	SysML
Abid et al. [117]				+	+	-	+	SysML
Abramovici et al. [118], [119]				+	+	+	+	SDM- Ontologie
Harrison et al. und Konstantinov et al. [120], [121]				+	+	+	+	VueOne Engineering Toolset
















Erfüllt Teilweise erfüllt Nicht erfüllt




+ Vorhanden | - | Nicht vorhanden |

Wie aus Tabelle 1 hervorgeht, kann trotz aller Funktionalitäten dieser Ansätze, wie von Abramovici et al. [118] [119], Harrison et al. [120] und Konstantinov et al. [121], keiner allen Anforderungen zur Synchronisierung der bestehenden Modelle des Digitalen Zwillings ab der Inbetriebnahme eines automatisierten Systems gerecht werden.

In Tabelle 2 wird ein Überblick über die in Unterkapitel 3.3 vorgestellten Ansätze zur Synchronisierung von Modellen eines automatisierten Systems gegeben. Diese werden anhand der Forschungsanforderungen untersucht und bewertet. Die Ansätze von Koltun et al. [128] und Biesinger et al. [126] erfüllen zwar die Anforderung an eine automatisierte Anpassung der Modelle des Digitalen Zwillings, allerdings können die Anforderungen an eine domänenübergreifende Änderungs- und Abhängigkeitsdetektion im automatisierten System nicht erfüllt werden.

Tabelle 2: Zusammenfassung und Bewertung der bestehenden Ansätze zur Synchronisierung von Modellen eines bestehenden automatisierten Systems

























Anforderungen Forschungsansätze	(A1) Detektion der Änderungen bis hin zur Feld-Ebene	(A2) Detektion der Abhängigkeiten	(A3) Anpassung der Modelle	Synchronisiertes Modell im Digitalen Zwilling durch den Forschungsansatz
Lindskog et al. [123]				3D-CAD Modell des automatisierten Systems
Erdős et al. [124], [125]				3D-CAD Modell des automatisierten Systems
Zipper et al. [29], [127]				Verhaltens- und Prozessablaufmodell des automatisierten Systems
Koltun et al. [128]				ECAD-Modell des automatisierten Systems
Biesinger et al. [126]				Stückliste und 3D-CAD Modell des automatisierten Systems

 Erfüllt
  Teilweise erfüllt
  Nicht erfüllt

Die Ansätze zur Modellgenerierung aus einem bestehenden, automatisierten System im Unterkapitel 3.4 sind in der Tabelle 3 zusammengefasst. Da sich diese Ansätze auf die

automatisierte Generierung der domänenübergreifenden Modelle des Digitalen Zwillings und nicht auf die Synchronisierung der Modelle des Digitalen Zwillings konzentrieren, werden sie anschließend auf der Basis der von ihnen generierten Modelle aus dem System analysiert und beurteilt.

Tabelle 3: Zusammenfassung und Bewertung der Ansätze zur Modellgenerierung aus einem bestehenden automatisierten System

Anforderungen Forschungsansätze				Generiertes Modell im Digitalen Zwillings durch den Forschungsansatz
	(A1) Detektion der Änderungen bis hin zur Feld-Ebene	(A2) Detektion der Abhängigkeiten	(A3) Anpassung der Modelle	
Walla et al. [33]				3D-CAD Modell des automatisierten Systems
Li et al. [129]				3D-CAD Modell des automatisierten Systems
Stark et al. [131]				3D-CAD Modell des automatisierten Systems
PRNETA [134]				Steuerung-Netzwerk Topologie des automatisierten Systems
Eigner et al. [138]				Simulationsmodell des automatisierten Systems
Son et al. [130], [132]				3D-CAD Modell des automatisierten Systems
Biesinger et al. [136]				Prozessablauf-Stückliste und Steuerung-Netzwerk Topologie des automatisierten Systems
Westkämper et al., Frank et al. [16], [68]				Simulationsmodell des automatisierten Systems



Erfüllt



Teilweise erfüllt



Nicht erfüllt

Durch die Bewertung der aufgelisteten Ansätze nach den Forschungsanforderungen in Tabelle 3 ergibt sich, dass auch in diesen Ansätzen eine Methodik für die automatisierte Generierung von domänenübergreifenden Modellen des Digitalen Zwillings fehlt.

Nach der Untersuchung des Stands der Wissenschaft und Technik unter Berücksichtigung der Forschungsanforderungen lässt sich folgende Forschungslücke feststellen.

„Es fehlt eine Methode, die domänenübergreifende Änderungen und deren Abhängigkeiten im Gesamtsystem nach der Inbetriebnahme eines automatisierten Systems automatisch erkennt und die Systemmodelle unter Gewährleistung der Konsistenz der Modelle automatisch anpasst.“

Im nächsten Kapitel wird zur Schließung der o.g. Lücke ein Konzept ausführlich vorgestellt. Dieses Konzept erlaubt eine automatisierte Erkennung von domänenübergreifenden Änderungsszenarien in einem automatisierten System durch eine Steuerungssoftwareanalyse. Zusätzlich bietet es durch eine regelbasierte Analyse auch die Möglichkeit, die domänenübergreifenden Abhängigkeiten der Änderungen innerhalb der Modelle in der mechanischen, elektrischen und Softwaredomänen des Systems zu erkennen. Im Anschluss daran wird eine automatisierte Anpassung der domänenübergreifenden Modelle des Systems mittels eines Engineering-Change-Management-Prozesses erläutert.

4 Lösungsansatz zur domänenübergreifenden Synchronisierung der Modelle des Digitalen Zwillings

In diesem Kapitel wird ein systematisches Konzept vorgestellt, mit dessen Hilfe die Abweichungen zwischen den Modellen und der Realität eines automatisierten Systems während des Betriebs ermittelt sowie eine automatisierte Modellanpassung unter Berücksichtigung der Nachverfolgbarkeit von Änderungen am Gesamtsystemmodell ermöglicht. Basierend auf diesem Konzept wird eine Methodik beschrieben, welche die Änderungen und deren domänenübergreifende Abhängigkeiten in der realen Welt mit Hilfe einer regelbasierten Analyse der aktuellen SPS-Steuerungssoftware eines automatisierten Systems detektiert. Abschließend beinhaltet die Methodik eine automatisierte Modellanpassung unter Berücksichtigung der Nachverfolgbarkeit von Änderungen am Gesamtsystemmodell.

In diesem Zusammenhang werden zunächst im Unterkapitel 4.1 die grundlegenden Konzeptentscheidungen präsentiert. Hierfür wird der Begriff des Ankerpunktes innerhalb des Digitalen Zwillings eines automatisierten Systems definiert. Außerdem werden in diesem Unterkapitel die Voraussetzungen zur Synchronisierung dieser Ankerpunkte innerhalb des Digitalen Zwillings beschrieben. Das Unterkapitel 4.2 bietet einen Gesamtüberblick über die einzelnen Prozessschritte der Methodik in drei aufeinanderfolgenden Phasen, der (1) Änderungsdetektion, der (2) domänenübergreifende Abhängigkeitserkennung und (3) der Modellanpassung zur Synchronisierung des Digitalen Zwillings. In Phase eins wird die Methode der Modellabstraktion der SPS-Steuerungssoftware und des Modellvergleichs von der Steuerungssoftware zu verschiedenen Zeitpunkten eines Systems zur automatisierten Detektion von Änderungen in der Softwaredomäne ausführlich beschrieben. Phase zwei befasst sich mit der Methode zur automatisierten Analyse der detektierten Abweichungen zwischen der Steuerungssoftware in Bezug auf das Erkennen der auftretenden Änderungsszenarien im System und den relevanten Relationen der Modelle des Digitalen Zwillings in den Domänen Mechanik, Elektrik und Software. Abschließend wird in der dritten Phase eine Methode für die automatisierte Modellanpassung unter Berücksichtigung der Nachvollziehbarkeit von Änderungen am Gesamtsystemmodell für Systemingenieure und der Aufrechterhaltung der Konsistenz zwischen den domänenübergreifenden Modellen des Digitalen Zwillings vorgestellt. Diese Methode basiert auf der automatisierten Generierung von Change-Request-Modellen für die relevanten Komponentenmodelle des Digitalen Zwillings.

4.1 Grundlegende Konzeptentscheidungen und Ansatz

Ein Digitaler Zwillings beinhaltet die Modelle sämtlicher Assets des automatisierten Systems, sogenannte Gesamtsystemmodelle, in denen die Domänen Mechanik, Elektrik und Software

vorkommen und die immer mit dem realen System synchronisiert werden müssen. Ein Gesamtsystemmodell kann beispielsweise aus einem CAD-Modell und seinen Baugruppenrelationen in der mechanischen Domäne, aus elektrischen Schaltplanmodellen und einer Prozessablaufbeschreibung in der elektrischen Domäne sowie aus Steuerungs-Signalen, -Funktionsblöcken und -Datenblöcken in der Softwaredomäne bestehen, vgl. Abbildung 19.

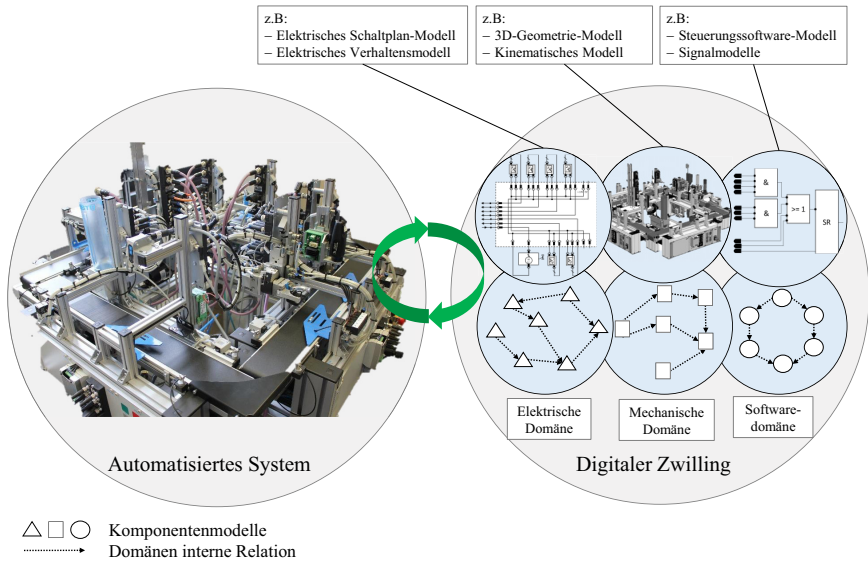


Abbildung 19: Digitaler Zwilling eines automatisierten Systems und dessen Modelle innerhalb des Digitalen Zwillings

In dieser Arbeit wurde die Forschungslücke identifiziert, die aufzeigt, dass eine Methode fehlt, die nach der Inbetriebnahme eines automatisierten Systems, automatisch domänenübergreifende Änderungen sowie deren Abhängigkeiten im System erkennt und das Gesamtsystemmodell in den Domänen, Mechanik, Elektrik und Software anpasst. Das Ergebnis dieser Forschungen wird daher sein, eine automatisierte domänenübergreifende Synchronisierung der Modelle des Digitalen Zwillings mit einem realen automatisierten System bereitzustellen und damit einen Beitrag zur Steigerung der Effizienz des Engineering-Prozesses eines automatisierten Systems ab der Inbetriebnahme liefern zu können.

Im Rahmen domänenübergreifender Änderungs- und Abhängigkeitsdetektion müssen zwei wesentliche Punkte bei der Erarbeitung eines Lösungsansatzes berücksichtigt werden. Erstens muss eine Datenquelle aus dem realen System zur Änderungsdetektion benutzt werden, die ausführliche Informationen über den aktuellen Stand des Systems bis hinunter zur Feld-Ebene enthält. Anderenfalls können die Änderungen des automatisierten Systems nicht vollständig erkannt werden. Zweitens ist domänenübergreifendes Fachwissen über häufig auftretende

Änderungen und deren Abhängigkeiten im automatisierten System während der Inbetriebnahme und des Betriebs einzubringen. Dieses Fachwissen kann genutzt werden, um die aktuelle Datenquelle aus einer Domäne automatisiert zu analysieren, um die Abhängigkeiten der Änderung auf andere Domänen im System zu erkennen. In Bezug auf den ersten Punkt kann die Steuerungssoftware des automatisierten Systems als geeignete Datenquelle betrachtet werden. Wie in Unterkapitel 2.1 diskutiert, bestehen automatisierte Systeme aus verschiedenen Sensoren und Aktoren, die alle zentral durch eine SPS gesteuert werden können. Nach jeder Änderung an den Sensoren oder Aktoren des Systems muss die SPS-Steuerungssoftware entsprechend aktualisiert werden. Infolgedessen kann die Steuerungssoftware einer SPS als eine ausführliche Datenquelle der aktuellen Komponenten des automatisierten Systems in der realen Welt eingeschätzt werden. Bezüglich der Beachtung des zweiten Punktes ist eine ausführliche Studie notwendig, um häufig auftretende Änderungen im realen System und deren Abhängigkeiten in den Domänen, Mechanik, Elektrik und Software zu präzisieren und zu kategorisieren. Außerdem sind die Modelle innerhalb des Digitalen Zwillings inhomogen und hoch komplex. Daher ist eine automatisierte Anpassung der Modelle des Digitalen Zwillings unter Wahrung der Konsistenz zwischen den Modellen und deren Relationen als eine große Herausforderung zu bezeichnen.

Das Konzept dieser Arbeit befasst sich mit der Synchronisierung der Komponentenmodelle eines Assets auf der Ebene mechatronischer Komponenten im Gesamtsystemmodell eines automatisierten Systems innerhalb der Domänen Mechanik, Elektrik und Software. Diese Komponentenmodelle werden in dieser Arbeit als Ankerpunkte des Assets innerhalb des Digitalen Zwillings bezeichnet. Die Steuerungssoftware eines automatisierten Systems beinhaltet über die in der Software verwendeten Ein- und Ausgangssignale Informationen zu allen mechatronischen Komponenten des Systems in seinem aktuellen Zustand. Wird eine Änderung am realen System durchgeführt, muss zeitgleich das Steuerungsprogramm angepasst werden, sodass die neue Anordnung der Komponenten Anwendung findet. Im Konzept dieser Arbeit werden die Komponentenmodelle, im Weiteren auch Ankerpunkt genannt, des geänderten Assets innerhalb der Steuerungssoftware gesucht und seine Relationen analysiert. Schließlich werden unter Verwendung der Informationen, die aus den erkannten Ankerpunkten von geänderten mechatronischen Komponenten gewonnen wurden, die Ankerpunkte des Assets automatisiert innerhalb der Modelle des Digitalen Zwillings unter Wahrung der Konsistenz angepasst.

Im nächsten Abschnitt erfolgen die Erörterung der Granularität eines Assets in einem automatisierten System und die Definition der Ankerpunkte eines Assets innerhalb des Digitalen Zwillings.

4.1.1 Ankerpunkte innerhalb des Digitalen Zwillings

Der Digitale Zwillings eines Assets besteht aus domänenübergreifenden Modellen mit domäneninternen und -übergreifenden Relationen sowie aus den Relationen zu Modellen anderer Komponenten im automatisierten System. In einem Digitalen Zwillings eines Assets werden alle

diese Modelle und Relationen unter einer eindeutigen ID zusammengefasst und referenziert. Die Bestandteile eines Digitalen Zwillings und dessen Architektur wurden in Abschnitt 3.1.2 detailliert erläutert. Ein Asset kann hierbei das gesamte automatisierte System oder eine Einheit aus vielen Sensoren und Aktoren bis hin zu einer Schraube im System sein. Die Granularität eines Assets wird während des Engineerings durch die Ingenieure entsprechend der Komplexität des Zielsystems bestimmt. Dementsprechend enthält der Digitale Zwilling eines Assets seine domänenübergreifenden Modelle und Relationen in verschiedener Granularität. Im Kontext der diskreten Fertigung stellen Assets alle Mittel dar, die zur Durchführung eines Prozesses oder allgemein zur Erfüllung einer Aufgabe in einem automatisierten System notwendig sind. Wie in Abbildung 20 gezeigt, lassen sich die Assets in die vier verschiedenen Klassen *Betriebsstätten*, *Betriebsmittel*, *Betriebshilfsmittel* und *Personal* untergliedern. Nach [140] bezeichnet die Klasse der Betriebsmittel, die im Fokus dieser Arbeit stehen, "alle Maschinen, Geräte und Komponenten, die der Ausführung von Herstellungs-, Handhabungs-, Transport-, Prüf- und Lagerschritten dienen". So können, entsprechend ihres funktionalen Umfangs, Bauteile (Part), mechanische Baugruppe, mechatronische Komponenten, Funktionsgruppen, Stationen und komplette Anlagen bzw. automatisierte Systeme als Betriebsmittel verstanden werden.

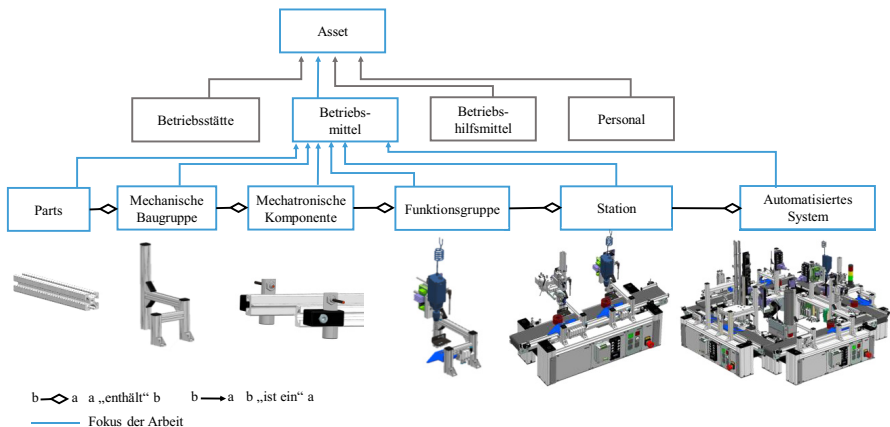


Abbildung 20: Granularität des Assets innerhalb der Betriebsmittel in einem automatisierten System in Anlehnung an [166] und [140]

Das kleinste Asset eines automatisierten Systems ist ein sogenannter Part, das aus nur einer mechanischen Einheit besteht, beispielsweise eine Schraube. Die mechanische Baugruppe ist eine Zusammenstellung aus vielen Parts ohne elektrische oder softwaretechnische Bestandteile. Eine Haltevorrichtung im System kann als mechanische Baugruppe dargestellt werden. Die mechatronischen Komponenten stellen die kleinsten Assets innerhalb eines automatisierten Systems dar, die aus den drei Domänen Mechanik, Elektrik und Software bestehen. In [10] wird eine mechatronische Komponente als ein Objekt vorgestellt, das aus mechanischen, elektrischen

und informationstechnischen Elementen besteht. Als mechatronische Komponenten werden zum Beispiel Aktoren und Sensoren bezeichnet. Diese Arbeit befasst sich jedoch mit der Synchronisierung eines Assets bis hinunter zur Feldebene eines automatisierten Systems. Folglich wird die kleinste Granularität des Assets im System, das nach dem Konzept dieser Arbeit innerhalb des Digitalen Zwillings synchronisiert werden muss, eine mechatronische Komponente sein. Funktionsgruppen können als Bündelung von mechatronischen Komponenten verstanden werden, die als eine Art Funktionseinheit der Erfüllung einer definierten Aufgabe dienen. Hardwareseitig sind Funktionsgruppen durch ein mechanisches und elektrisches Zusammenwirken definiert. Softwareseitig ist die gemeinsame Ansteuerung durch einen SPS-Programmbaustein kennzeichnend. Ein Beispiel für eine Funktionsgruppe ist eine Spannventilgruppe, die aus mechatronischen Spannern und Ventilen sowie aus mechanischen Komponenten, zum Beispiel Stellelementen zur Fixierung, besteht. Die Station oder Arbeitsstation (z.B. Ladestation, Schweißstation etc.) ist in dieser Hierarchie als eine Reihe von Funktionsgruppen aufzufassen, die innerhalb des Systems zur Erfüllung einer komplizierteren Aufgabe dienen. Zur Reduzierung der Komplexität der Anlagensteuerung werden die Stationen in der Regel möglichst unabhängig voneinander und der gesamten Steuerungssoftware der Anlage programmiert. Die miteinander verbundenen Stationen innerhalb einer Anlage werden als ein automatisiertes System bezeichnet.

Im Hinblick auf die Synchronisierung der Modelle des automatisierten Systems nach dem Zeitpunkt der Inbetriebnahme müssen die im System auftretenden Änderungen im Gesamtsystemmodell in verschiedenen Domänen synchronisiert werden. Wenn zum Beispiel eine mechatronische Komponente wie ein Sensor im automatisierten System nach der Inbetriebnahme angeschlossen wird, müssen die Komponentenmodelle des Sensors (3D-CAD-Modell, Schaltplanmodell, Signalmodell usw.) und ihre Relationen zu anderen mechatronischen Komponenten im System (mechanische Fixierung an anderen Komponenten, Verkabelung mit anderen Komponenten, Steuerungsverhalten usw.) innerhalb des Gesamtsystemmodells in jeder Domäne Mechanik, Elektrik und Software angepasst werden, um das Gesamtsystemmodell synchron zu halten.

*Diese Komponentenmodelle eines Assets, mit der kleinsten Granularität einer mechatronischen Komponente, in den unterschiedlichen Domänen innerhalb des Gesamtsystemmodells werden in Zukunft als **Ankerpunkte** eines Assets verstanden. Die **Ankerpunkte** eines Assets werden durch eine eindeutige ID in den Digitalen Zwilling des Assets referenziert bzw. verlinkt.*

Im Kontext der Synchronisierung der Modelle mit dem automatisierten System genügt es, die betroffenen Ankerpunkte und deren domänenübergreifenden Abhängigkeiten durch ein geändertes Asset zu detektieren und anzupassen. So kann der Digitale Zwilling des Systems bereitgestellt und synchronisiert werden.

Als Ankerpunkte einer mechatronischen Komponente in der Softwaredomäne können z.B. die Ein- und Ausgangssignale von Sensoren und Aktoren innerhalb der SPS-Steuerungssoftware betrachtet werden. In der elektrischen Domäne ist der Ankerpunkt einer mechatronischen Komponente ihr ECAD-Modell im Gesamtschaltplanmodell des Systems. Zusätzlich kann das funktionale Beschreibungsmodell des Verhaltens einer mechatronischen Komponente als Ankerpunkt der mechatronischen Komponente im funktionalen Gesamtsimulationsmodell des Systems erkannt werden. In der mechanischen Domäne wird das CAD-Modell der mechatronischen Komponente als Ankerpunkt im Gesamt-CAD-Modell des automatisierten Systems erkannt, vgl. Abbildung 21. Im Konzept dieser Arbeit erfolgt eine Fokussierung auf die Identifizierung und Analyse der Ankerpunkte der geänderten Assets innerhalb der Softwaredomäne. Dementsprechend wird das Nachgehen der Änderungen in den anderen Domänen mit dem Ziel der Synchronisierung des Digitalen Zwillings fortgesetzt.

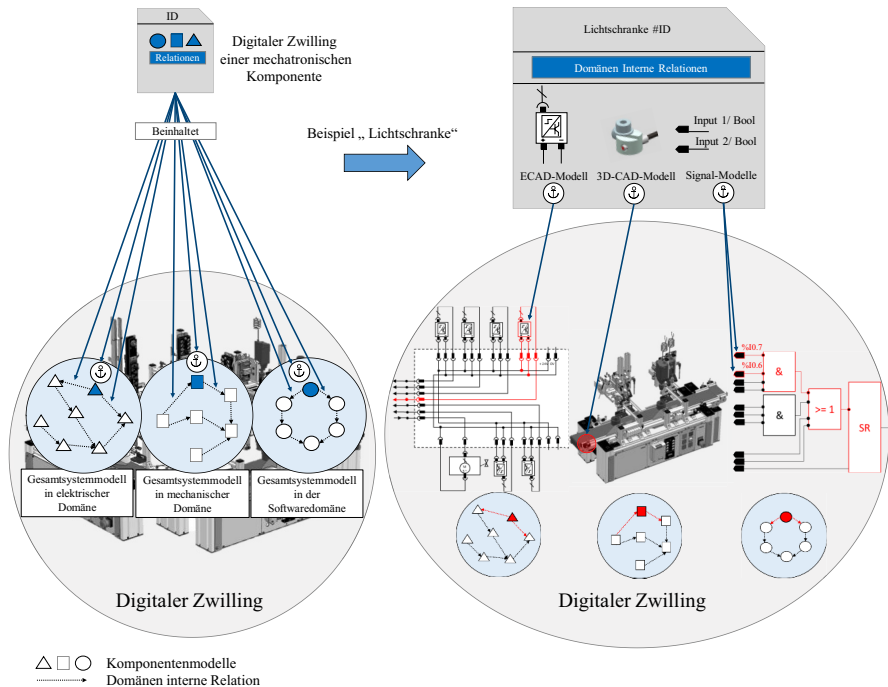


Abbildung 21: Ankerpunkte einer mechatronischen Komponente innerhalb des Digitalen Zwillings eines automatisierten Systems

Zur Erkennung der Ankerpunkte der geänderten Komponenten aus der modifizierten Steuerungssoftware eines automatisierten Systems sowie zur automatisierten Anpassung der Ankerpunkte innerhalb des Digitalen Zwillings müssen drei Voraussetzungen erfüllt werden:

1. **Vorhandensein einer Komponentenbibliothek:** Zunächst besteht die Voraussetzung, dass eine mechatronische Komponentenbibliothek existiert, die wiederverwendbare Modelle von verschiedenen Assets bereitstellt. Diese Komponentenbibliothek und deren domänenübergreifende, wiederverwendbare Modelle des Assets unter einer eindeutigen ID ermöglichen es, die Modelle des geänderten Assets in den Domänen Mechanik und Elektrik anhand seines Ankerpunktes in der Softwaredomäne zu identifizieren.
2. **Namenskonvention zur Benennung mechatronischer Komponenten:** Weiterhin besteht ein Bedarf an einer Namenskonvention für die Benennung der mechatronischen Komponenten innerhalb der Steuerungssoftware bei der anfänglichen Programmierung der Steuerungssoftware einer Anlage sowie bei Änderungen im Betrieb. Dies ermöglicht die Analyse der Steuerungssoftware und die Identifikation von Abhängigkeiten in anderen Domänen. Die Namenskonvention sollte jedoch nicht zu weit von den aktuellen Industrienormen entfernt sein, sodass die Umsetzung des Konzepts dieser Arbeit mit bestehenden Technologien und Normen verträglich ist.
3. **Versionierung der Steuerungssoftware:** Die dritte Voraussetzung ist die regelmäßige Speicherung und Versionierung der Steuerungssoftware des automatisierten Systems ab dem Zeitpunkt der Inbetriebnahme. Dies ermöglicht einen Vergleich der verschiedenen Versionen der Steuerungssoftware zur Änderungsdetektion.

Diese drei Voraussetzungen für eine automatisierte Synchronisierungsmöglichkeit der Ankerpunkte des Digitalen Zwillings werden im nächsten Abschnitt ausführlich beschrieben.

4.1.2 Voraussetzungen zur Synchronisierung der Ankerpunkte

Für eine automatisierte Detektion der Ankerpunkte der geänderten mechatronischen Komponente innerhalb der Steuerungssoftware sowie eine automatisierte Anpassung ihrer Ankerpunkte innerhalb des domänenübergreifenden Gesamtsystemmodells des Digitalen Zwillings müssen folgende drei Voraussetzungen erfüllt sein.

Voraussetzungen für eine Komponentenbibliothek

Der Einsatz von wiederverwendbaren Modellen während des Engineerings eines automatisierten Systems ist ein weit verbreiteter Ansatz in der Industrie. Die Wiederverwendung von Modellen zielt darauf ab, die Modellierungszeit während des Engineerings zu verkürzen und den Engineering-Prozess systematisch zu optimieren. Hierfür ist eine Komponentenmodellbibliothek erforderlich, die wiederverwendbare domänenübergreifende Modelle des Assets beinhaltet, die im automatisierten System verwendet werden. Diese mechatronische Komponentenbibliothek ist ein Pool von gekapselten Modulen, die auch für die Rekonfiguration eines Fertigungssystems verwendet werden können. Bei diesem Ansatz werden domänenübergreifende Modelle eines Assets, z.B. Funktionsmodelle in der Softwaredomäne, 3D-CAD-Modelle in der mechanischen

Domäne und Schaltplanmodelle in der elektrischen Domäne in einem gemeinsamen Modul gekapselt. Zur Wiederverwendung der Modelle eines Moduls müssen in der Regel nur die Modelle eines Moduls aus einer Komponentenbibliothek instanziiert werden. Über ihre Beziehungen zu anderen Modellen und ihre einstellbaren Parameter können diese für jeden spezifischen Anwendungsfall angepasst werden.

In dieser Arbeit wird davon ausgegangen, dass eine Komponentenbibliothek existiert, die die domänenübergreifenden Modelle von Assets unter einem eindeutigen Namen kapselt. Diese Bibliothek muss während des Engineering-Prozesses des Systems aufgebaut werden und kapselt außer den Modellen der Komponenten selbst ebenso Informationen wie eine eindeutige ID oder einen Namen. In Abbildung 22 ist eine Komponentenbibliothek dargestellt, die aus mehreren gekapselten Modulen mit ihren eindeutigen IDs besteht. Die domänenübergreifenden Modelle eines Assets werden in ihrem Modul über diese eindeutige ID miteinander verbunden.

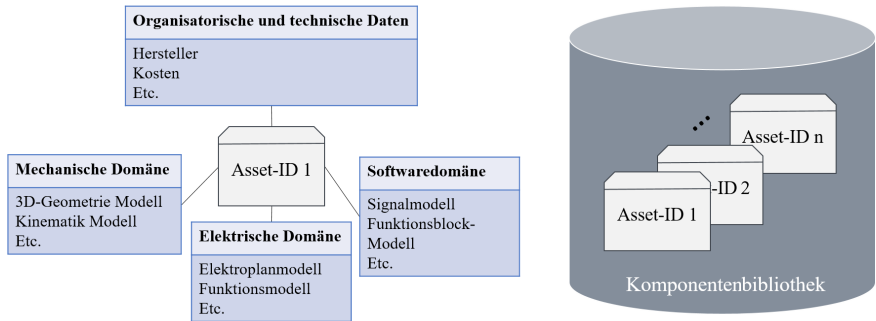


Abbildung 22: Kapselung von domänenübergreifenden wiederverwendbaren Modellen eines Assets innerhalb einer Komponentenbibliothek

Für die industrielle Umsetzung einer Komponentenbibliothek nach [68] und [141] kann eine domänenübergreifende Modularisierung in mechanischer, hydraulischer, elektrischer und softwaretechnischer Sicht mit EPLAN oder Mind8 Engineering Center, Automation Designer und TIA Portal von Siemens realisiert werden. Im Rahmen des Konzepts dieser Arbeit kann auf domänenübergreifende Modelle des geänderten Assets zugegriffen werden, indem der eindeutige Name des Assets in der Softwaredomäne erkannt und die Komponentenbibliothek verwendet wird. Daher wird im Lösungsansatz dieser Arbeit zur Synchronisierung der Modelle des Digitalen Zwillings die Vorteile einer Komponentenbibliothek vorausgesetzt und genutzt.

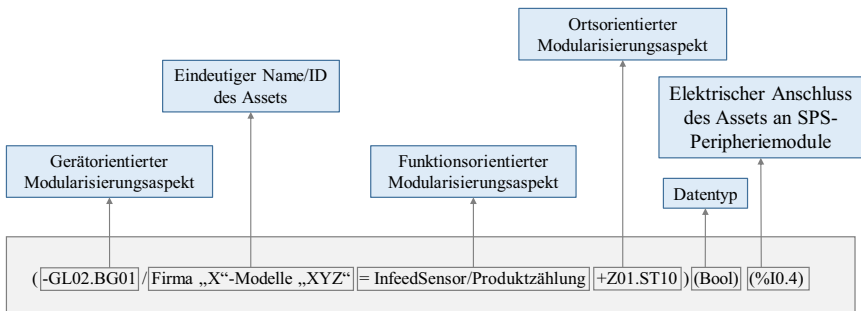
Voraussetzungen für eine Namenskonvention innerhalb der Steuerungssoftware

Zur automatisierten Detektion der Ankerpunkte der geänderten mechatronischen Komponenten durch die Analyse der aktuellen Steuerungssoftware des automatisierten Systems muss eine Namenskonvention zur Benennung der Ankerpunkte der Assets innerhalb der SPS-Steuerungssoftware definiert werden. Diese Namenskonvention muss während des Engineering-

Prozesses und bei jeder Änderung der SPS-Steuerungssoftware nach der Inbetriebnahme des Systems eingehalten werden. Eine Namenskonvention ist allgemein ein essenzieller Bestandteil des Engineering-Prozesses. Die Identifizierung der Modelle und deren Verknüpfung über die unterschiedlichen Domänen hinweg ist mit einer einheitlichen Namenskonvention erreichbar. In der Industrie werden verschiedene Namenskonventionen für die Benennung des Assets insbesondere für die Standardisierung in der Steuerungssoftware verwendet, die für eine Identifikation verwendet werden können. Die Namenskonvention für diese Arbeit muss jedoch drei wichtige Aspekte berücksichtigen: Erstens müssen alle Modelle eines Assets durch eine eindeutige ID oder einen Namen innerhalb seines Digitalen Zwillings gemeinsam referenziert werden. Dementsprechend müssen alle Modelle eines Assets in der Cyberwelt diese eindeutige ID in ihrem Namen tragen. Zum Beispiel muss der eindeutige Name eines Assets im Namen seiner Modelle innerhalb der SPS-Steuerungssoftware (Signale, Funktions- und Datenblöcke) angegeben werden. Zweitens müssen Informationen über den topologischen Ort des Assets im Namen der Ankerpunkte des Assets angegeben werden, um eine automatische Erkennung der Position des geänderten Assets im System zu ermöglichen. Drittens muss diese Namenskonvention einfach zu verwenden sein, damit die Ingenieure sie schnell und einfach während der Engineeringphase des automatisierten Systems sowie bei der Durchführung von Änderungen am System nutzen können. Dementsprechend können durch die Ankerpunkte sowie deren zu Grunde liegende Namenskonvention eindeutig das Ziel-Asset und der Standort des Assets im Gesamtsystem bestimmt und einfach umgesetzt werden. In dieser Arbeit wird als Voraussetzung zur Synchronisierung der Ankerpunkte innerhalb des Digitalen Zwillings davon ausgegangen, dass während des gesamten Lebenszyklus des automatisierten Systems eine Namenskonvention - basierend auf der DIN-Norm EN 81346 - und eine eindeutige ID des Assets verwendet werden. Die DIN-Norm EN 81346 bzw. die ISO-Norm IEC 81346 definieren eine Norm zur Standardisierung der Benennung mechatronischer Komponenten in automatisierten Systemen basierend auf der spezifischen Kennzeichnungen sowie einer Strukturierung der Assets im System unter drei Positionsaspekten: geräteorientiert, ortsorientiert und funktionsorientiert [142], [143]. Die Informationen über die geräte-, orts- und funktionsorientierte Position des Assets in der Gesamtsystemstruktur sind von Bedeutung, da durch die Erkennung dieser Informationen die Positionen der Ankerpunkte des Assets in allen Domänen Mechanik, Elektrik und Software unter Berücksichtigung der unterschiedlichen Perspektiven der Konstrukteure aus den jeweiligen Domänen während des Engineering-Prozesses identifiziert werden können. Wie bereits im Abschnitt 2.2 ausführlich beschrieben, ist der Engineering-Prozess eines automatisierten Systems ein sequentieller Prozess, der mit der mechanischen Domäne beginnt, mit der elektrischen Domäne fortgesetzt wird und mit der Softwaredomäne endet. Hier haben die Konstrukteure jeder dieser Domänen verschiedene Aspekte zu berücksichtigen, daher ist die Modularisierung der Modelle unterschiedlich. Die Konstruktion in der mechanischen Domäne modularisiert ein Modell - basierend auf der ortsorientierten Strukturierung der Assets - im System: Die Assets, die zusammen eine physische Einheit an einem Ort bilden, werden als eine Einheit, ein Modul,

definiert. Während des Engineering-Prozesses der elektrischen Domäne werden die Konstrukteure die Modelle der Assets im System geräteorientiert betrachten und mit ihren Tools geräteorientiert modellieren bzw. strukturieren. Geräteorientiert bedeutet, dass diese an eine gemeinsame Versorgungseinheit angeschlossenen Assets als eine Einheit betrachtet werden. In der Softwaredomäne ist die Modularisierung funktionsorientiert, d.h. die von Softwareentwicklern erstellten Modelle werden zu einer Einheit zusammengefasst, wenn sie die gleiche Funktionalität haben [144]. Alle diese geräte-, orts- und funktionsorientierten Informationen eines Assets im System müssen nach der DIN-Norm EN 81346 in die Namenskonvention aufgenommen werden. Durch Hinzufügen der eindeutigen ID des Assets zu dieser Namenskonvention können während der Analyse zusätzliche Informationen, wie die Namen der domänenübergreifenden, wiederverwendbaren Modelle des Assets in einer Komponentenbibliothek, extrahiert werden.

In Abbildung 23 ist beispielsweise der Signalname eines Assets unter Verwendung der Namenskonvention dargestellt. In EN 81346 wird eine Kennung definiert, die aus zwei Kennbuchstaben und einer eindeutigen, zweistelligen Nummerierung besteht, die mit der Anzahl der vorhandenen Objekte in dieser Klasse hochgezählt wird. Diese Kennbuchstaben setzen sich aus einer Hauptklasse und einer nachfolgenden Unterklasse zusammen.



GL02: Zweite Förderband, BG01: Erste Bewegungsmelder, Z01: Zone 1, ST10: Station 10

Abbildung 23: Signalname eines Assets gemäß der Namenskonvention

Die Hauptklasse gibt die weitgefasste Aufgabe des Objektes an, die durch die Unterklasse genauer definiert wird. Eine Lichtschranke hat beispielsweise die Aufgabe, die Anwesenheit eines Gegenstandes als elektrisches Signal darzustellen und wird daher der Hauptklasse B (Umwandlung einer Eingangsvariable) und der Unterklasse G (Art der Eingangsvariablen: Abstand, Länge, Stellung) zugeordnet. Die Lichtschranke wird daher durch die eindeutige Kodierung BG01 repräsentiert. Jede weitere Lichtschranke bekommt eine neue Nummer zugeordnet. Das jeweilige Kürzel wird dann zusätzlich noch mit einem der drei definierten Vorzeichen versehen, um den Betrachtungsaspekt festzulegen. Wie in Abbildung 23 zu sehen, wird zu Beginn der Benennung des Signals die geräteorientierte Modularisierung des Assets im

System genannt. Im Anschluss folgt, getrennt durch einen Querstrich, der eindeutige Name der mechatronischen Komponente und deren Funktionalität. Es folgt der funktionale Aspekt, durch einen Unterstrich getrennt. Dieser Teil gibt die Funktion der mechatronischen Komponente im gesamten System an. Durch einen weiteren Unterstrich getrennt, folgt abschließend der Ortsaspekt, der angibt, wo sich die Komponente räumlich befindet und wo sich eine Referenz zur Position im mechanischen Entwurf befindet. Diese Namenskonvention muss während des gesamten Lebenszyklus des Systems für Signale, Funktionsblöcke und Datenblöcke in der SPS-Steuerungssoftware eingehalten werden.

Voraussetzungen für die Versionierung der Steuerungssoftware

Eine weitere Voraussetzung zur Synchronisierung der Ankerpunkte innerhalb des Digitalen Zwillings ist die Speicherung und Versionierung der Steuerungssoftware des Systems zu regelmäßigen Zeitpunkten. Diese ermöglicht eine Detektion der Ankerpunkte der geänderten Assets anhand des Vergleichs der SPS-Steuerungssoftware des Systems zu verschiedenen Zeitpunkten. Nachdem die Steuerungssoftware des automatisierten Systems durch eine virtuelle Inbetriebnahme getestet und validiert wird, muss diese SPS-Steuerungssoftware in einem Repository gespeichert und mit der SPS-Steuerungssoftware zum Referenzzeitpunkt bezeichnet werden. Darüber hinaus muss die SPS-Steuerungssoftware des Systems während des Betriebs regelmäßig im Repository gespeichert werden und steht somit zu verschiedenen Zeiten zum Vergleich zur Verfügung. Das Repository der Steuerungssoftware sollte ein Archiv sein, das die Steuerungssoftware des automatisierten Systems nach jeder Änderung enthält. Die Steuerungssoftware zu verschiedenen Zeitpunkten können entweder manuell durch die Ingenieure nach jeder Änderung im System oder automatisch in periodischen Abständen im Repository gespeichert werden.

Unter Berücksichtigung der vorgestellten Voraussetzungen befasst sich das nächste Unterkapitel ausführlich mit der daraus abgeleiteten Ankerpunktmethodik und deren drei Phasen, (1) Änderungsdetektion, (2) domänenübergreifende Abhängigkeitserkennung und (3) Modellanpassung zur Synchronisierung der Ankerpunkte des Digitalen Zwillings.

4.2 Ankerpunktmethodik zur Synchronisierung der Modelle des Digitalen Zwillings

Die SPS-gesteuerten automatisierten Systeme bestehen aus verschiedenen Baugruppen, die alle zentral durch eine SPS gesteuert werden. Ein Ankerpunkt einer mechatronischen Komponente oder Funktionsgruppe innerhalb des Gesamtsystemmodells in der Softwaredomäne kann ein Funktionsblock, ein Datenblock oder ein Signal innerhalb der SPS-Steuerungssoftware sein. Diese Elemente stehen in einer direkten Verbindung zu den realen Assets im automatisierten System. Bei einer Änderung des Systems müssen die Ankerpunkte zwangsläufig angepasst werden.

Wie zuvor in Unterkapitel 2.2 diskutiert, wird im letzten Schritt des Engineering-Prozesses eines automatisierten Systems die SPS-Steuerungssoftware mit den Verhaltensmodellen gegen die Anforderungen getestet. Das Ziel dieser virtuellen Inbetriebnahme ist die Validierung der entwickelten Steuerungssoftware, indem das Zusammenspiel der Aktoren und Sensoren mit den erwarteten Aktionen abgeglichen wird. Nach der virtuellen Inbetriebnahme und der Validierung der Steuerungssoftware, im Folgenden als Steuerungssoftware zum Referenzzeitpunkt bezeichnet, wird der Digitale Zwilling mit allen seinen Modellen bereitgestellt. In Bezug auf den Digitalen Zwilling wird das automatisierte System aufgebaut und in Betrieb genommen. Jedoch werden ab dem Zeitpunkt der Inbetriebnahme viele Änderungen am realen System vorgenommen. Damit das System nach den Änderungen wieder in Betrieb genommen werden kann, muss die Steuerungssoftware der SPS an diese Änderung angepasst werden. Dementsprechend kann die Steuerungssoftware als Datenquelle zur Darstellung der aktuellen mechatronischen Komponenten des realen automatisierten Systems gesehen werden. Darauf basierend können durch einen Abgleich der Ankerpunkte zwischen der aktuellen Steuerungssoftware und der gespeicherten Steuerungssoftware zum Referenzzeitpunkt sowie durch eine Analyse der Differenz, die Änderungen im realen System erkannt werden.

Im folgenden Abschnitt wird der detaillierte Prozessablauf der Ankerpunktmethode erläutert.

4.2.1 Gesamtprozessablauf der Ankerpunktmethode

Die Ankerpunktmethode soll die Synchronisierung der Modelle des Digitalen Zwillings unterstützen oder ermöglichen. Zur Synchronisierung der Modelle des Digitalen Zwillings besteht der Gesamtprozessablauf der Ankerpunktmethode aus drei Phasen mit insgesamt sieben Prozessschritten. In Abbildung 24 wird ein Ablaufdiagramm der Gesamtprozessschritte der Ankerpunktmethode veranschaulicht.

Im Assistenzsystem werden die drei Phasen der Ankerpunktmethode,

- (1) automatisierte Änderungsdetektion,
- (2) domänenübergreifende Abhängigkeitsanalyse und
- (3) Modellanpassung ausgeführt.

In der ersten Phase werden durch das Assistenzsystem die beiden SPS-Steuerungssoftwares zu zwei unterschiedlichen Zeitpunkten mittels der Formalisierungsmethode modelliert, abstrahiert und verglichen. Auf Basis dieser Analyse werden die Ankerpunkte und deren Relationen der geänderten Komponente in der Softwaredomäne erkannt. Die nächste Phase ist die Durchführung einer regelbasierten Analyse, um die domänenübergreifenden Abhängigkeiten der Änderungen zu erkennen. Schließlich wird das Assistenzsystem in der Endphase die Änderungen mit Hilfe einer Komponentenbibliothek automatisiert in den Digitalen Zwilling einbringen.

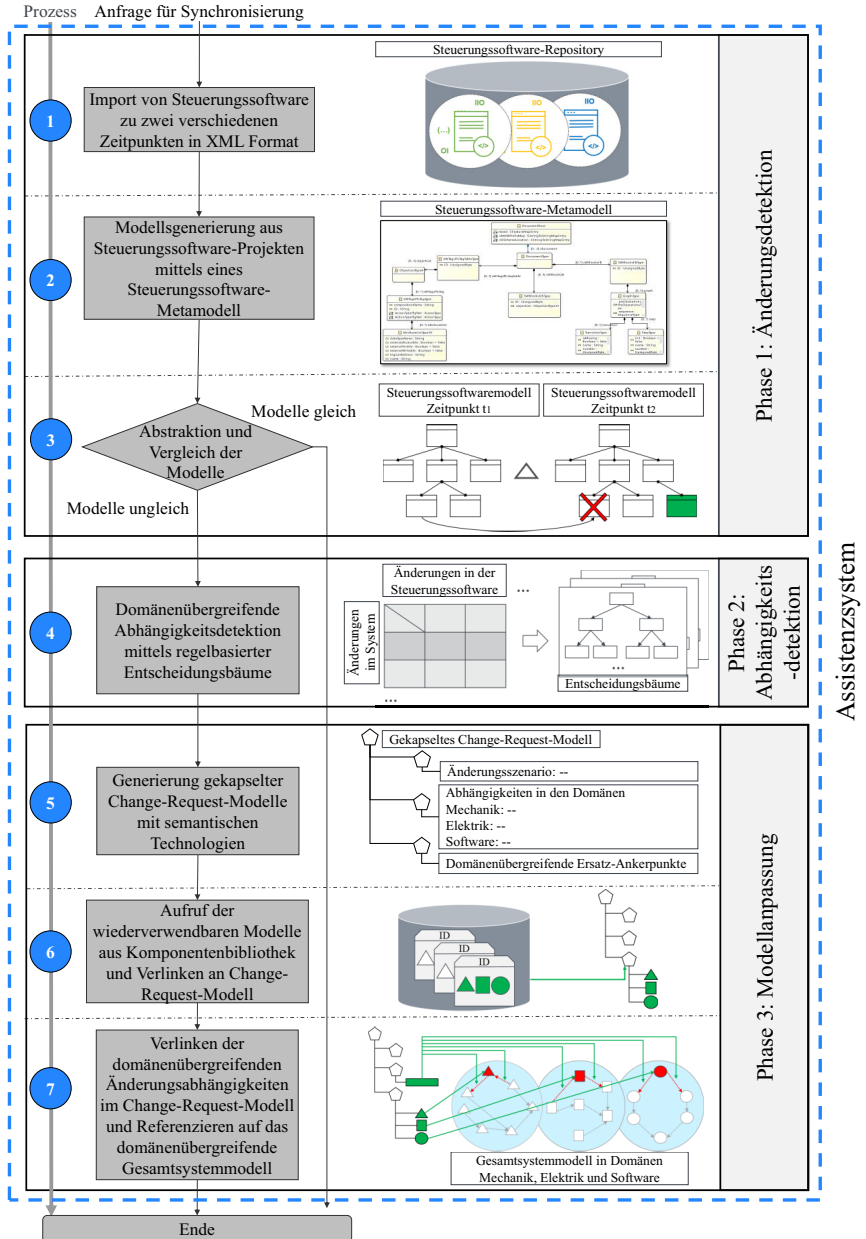


Abbildung 24: Prozessschritte der Ankerpunktmethode

Die erste Phase, die automatisierte Änderungsdetektion, besteht aus drei Schritten. Im ersten Schritt wird die SPS-Steuerungssoftware des Systems zu zwei verschiedenen Zeitpunkten aus einem Steuerungssoftware-Repository ausgelesen: Die Steuerungssoftware zum Referenzzeitpunkt und die aktuelle Steuerungssoftware des Systems.

Die zweiten und dritten Schritte des Ablaufdiagramms beschreiben die Formalisierung und Abstraktion der SPS-Steuerungssoftware basierend auf einem zentralen Steuerungssoftware-Metamodell, das einen Instanz-Instanz-Modellvergleich zwischen beiden Steuerungssoftware-Modellen ermöglicht. Im zweiten Steuerungssoftware-Modell wird nach den Ankerpunkten von modifizierten mechatronischen Komponenten im System gesucht, wie beispielsweise hinzugefügte, entfernte oder modifizierte Signale, Funktions- oder Datenblöcke. Die erkannten Änderungen werden für den nächsten Schritt als detektierte Ankerpunkte von geänderten Komponenten zur Verfügung gestellt. Der Prozess der automatischen Änderungserkennung wird im Abschnitt 4.2.2 näher beschrieben. Werden Änderungen innerhalb der verglichenen Modelle festgestellt, wird im vierten Schritt die zweite Phase (Abhängigkeitsdetektion) gestartet, was zu einer weiteren Erkennung von Änderungsszenarien und deren domänenübergreifenden Abhängigkeiten innerhalb des Gesamtsystems führt. Zu diesem Zweck müssen die Änderungsszenarien anhand eines domänenübergreifenden Expertenwissens in einer Regeltabelle zusammengefasst werden. In der Ankerpunktmethodik sammeln sich diese Regeln aus einer ausführlichen Untersuchung über häufig vorkommende Änderungen in den automatisierten Systemen. Mit diesen Regeln wird analysiert, welche geänderten Ankerpunkte und Relationen in der Softwaredomäne von welcher Bedeutung sind. Deshalb werden bei der Ankerpunktmethodik die Änderungen in der Steuerungssoftware gemäß den domänenübergreifenden Regeln kategorisiert, um die exakten Änderungsszenarien im System und den Einfluss von Änderungen auf mechanische und elektrische Domänen zu erkennen. Zur Erstellung und Kategorisierung dieser Regeln muss ein umfangreiches technisches Fachwissen über mögliche Änderungen und deren Auswirkungen in einem automatisierten System innerhalb der Domänen Mechanik, Elektrik und Software zusammengeführt werden. Die Regeltabelle enthält eine Reihe von Änderungsszenarien, die in einem automatisierten System auftreten und wie sie innerhalb der Steuerungssoftware erkannt werden können. Die Szenarien sind unter Angabe der Änderungsart und deren Verfolgung in der Steuerungssoftware zusammengefasst und kategorisiert. Diese Kategorisierungen müssen im Rahmen der Entscheidungsbäume entwickelt werden, um die detektierten, geänderten Ankerpunkte und deren Zusammenhänge im Steuerungssoftware-Modell regelbasiert zu analysieren. Diese Regeltabelle wird ausführlich im Abschnitt 4.2.3 beschrieben.

Nachdem Änderungen und Abhängigkeiten im realen System identifiziert wurden, sind die Modelle des entsprechenden Digitalen Zwillings anzupassen. Zu diesem Zweck beschreiben die Schritte fünf, sechs und sieben, welche die Phase drei „Modellanpassung“ bilden, eine automatisierte Generierung von gekapselten Change-Request-Modellen, die auf Engineering-Change-Management-Ansätzen mit semantischen Technologien basieren. Dieser Ansatz und der

Grund für die Entscheidung für diesen Ansatz wird im Abschnitt 4.2.4 detailliert beschrieben. Im fünften Schritt wird ein leeres Change-Request-Modell mit einer semantischen Technologie innerhalb der integrierten Modelle des Digitalen Zwillings erstellt. Zur Erstellung von Change-Request-Modellen müssen semantische Technologien benutzt werden, die auch zur Erstellung der Modelle des Digitalen Zwillings während des Engineerings verwendet wurden. Damit sind die erstellten Change-Request-Modelle für die fachübergreifenden Engineering-Tools lesbar. Auf diese Weise können die zuständigen Konstrukteure der jeweiligen Modelle innerhalb des Gesamtsystemmodells diese Change Request Modelle in ihren Tools ohne weiteres öffnen und einsetzen. Durch die Analyse der detektierten Ankerpunkte der geänderten Komponente aus dem vierten Schritt wird die eindeutige ID der modifizierten Komponente identifiziert. Mit dieser eindeutigen ID werden im sechsten Schritt alle wiederverwendbaren Modelle des Assets aus einer Komponentenbibliothek gelesen und in einem Change-Request-Modell gekapselt. Im letzten Schritt der Ankerpunktmethode, Schritt sieben, werden die domänenübergreifenden Abhängigkeiten der Änderungen im Gesamtsystemmodell in das Change-Request-Modell zusammengeführt. Dabei wird das gesamte, gekapselte Change-Request-Modell gemeinsam mit seinen Informationen und Modelle an den Ankerpunkten der geänderten Komponenten im Gesamtsystemmodell referenziert.

Basierend auf der Ankerpunktmethode ist ein Assistenzsystem zu realisieren, das ausführlich im Kapitel 5 beschrieben wird. Die gesamten Prozessschritte eins bis sieben der Ankerpunktmethode werden in einem Assistenzsystem automatisiert durchgeführt, was die Synchronisation der Modelle des Digitalen Zwillings ermöglicht. Letztendlich muss die Anpassung der Modelle des Digitalen Zwillings von den zuständigen Ingenieuren für die betroffenen Modelle unter Verwendung von den Ankerpunkten und Informationen innerhalb des Change-Request-Modell durchgeführt werden. Wie bereits beschrieben, werden diese Change-Request-Modelle mit semantischen Technologien direkt an die betroffenen Modelle referenziert. So sind die Systemingenieure in der Lage, die Änderungen direkt im Digitalen Zwilling zu übernehmen.

In den folgenden Abschnitten werden die oben beschriebenen Prozessschritte ausführlich erläutert.

4.2.2 Phase 1: Änderungsdetektion

In Bezug auf die erste Hauptanforderung dieser Forschung für die automatisierte Erkennung von Änderungsszenarien im automatisierten System während des Betriebs und der Umgang mit dem Mangel an einer Dokumentation von Änderungen in der realen Welt, enthält die Ankerpunktmethode einen Ansatz basierend auf der Analyse von SPS-Steuerungssoftware für die automatisierte Erfassung von Änderungen im System. In diesem Prozess werden die Unterschiede zwischen der aktuellen SPS-Steuerungssoftware des automatisierten Systems sowie der SPS-Steuerungssoftware zum Referenzpunkt automatisiert durch einen Vergleich ermittelt und analysiert, siehe die ersten drei Schritte der Ankerpunktmethode in Abbildung 24.

Ein SPS-gesteuertes, automatisiertes System besteht aus einer SPS als industrielles Mikrocontrollersystem, bei dem Hard- und Software speziell an die industrielle Umgebung angepasst sind. SPS als eingebettete Steuerungssysteme werden immer komplexer, da die Sicherheit von Systemen eine entscheidende Rolle für eine hohe Zuverlässigkeit spielt. Ein Nachteil bei der Arbeit mit SPSen ist die Vielfalt der Anbieter und Entwickler, die für die Integration und Erweiterung dieser verschiedenen Systeme existieren. IEC 61131 ist die internationale Norm für speicherprogrammierbare Steuerungen, die entwickelt wurde, um die Basis für eine standardisierte SPS-Programmierung zu schaffen und diese für eine effiziente SPS-Programmierung zu nutzen [55]. Der dritte Teil dieser Norm definiert fünf Programmiersprachen, die bereits in Abschnitt 2.1.1 beschrieben wurden. Die Verwendung eines definierten Standards für SPS-Programmiersprachen kann die Kompatibilität, Offenheit und Interoperabilität zwischen verschiedenen Geräten verbessern.

Ein Problem der IEC 61131 besteht darin, dass diese Norm kein standardisiertes Format für SPS-Steuerungssoftware, die mit zahlreichen SPS-Programmiertools auf dem Markt erstellt werden, vorschlägt. Zurzeit liegen nur herstellerspezifische Formate vor. Allerdings hat die SPS-Anwenderorganisation PLCopen ein Technisches Komitee gegründet, um ein XML-basiertes Standardformat für Projekte nach IEC 61131-10 zu definieren, wobei jedoch viele bestehende SPS-Programmiertools diese Norm noch nicht einsetzen. [145] [146]. Dementsprechend muss im Rahmen der Detektion der Änderungen im realen automatisierten System durch die Analyse der SPS-Steuerungssoftware ein unabhängiger Ansatz eines SPS-Steuerungssoftware-Formats zur Analyse eingesetzt werden. Darüber hinaus kann beispielsweise ein neues Steuerungssystem auf dem automatisierten System während der Betriebsphase installiert werden. Aufgrund dieser neuen Steuerung müssen die SPS-Programmiertools aktualisiert werden. Das neue SPS-Programmiertool fügt dem ursprünglichen SPS-Projekt ein neues Format hinzu. Dementsprechend sind die aktuelle SPS-Steuerungssoftware und die SPS-Steuerungssoftware zum Referenzzeitpunkt in verschiedenen Formaten aufgebaut. Bei der automatisierten Änderungserkennung muss ein weiterer, entscheidender Punkt berücksichtigt werden: Ein Vergleich und eine Analyse zwischen allen Elementen der aktuellen SPS-Steuerungssoftware und der SPS-Steuerungssoftware zum Referenzzeitpunkt sind nicht zwingend erforderlich. Es ist ausreichend, wenn nur die Ankerpunkte des Assets und deren Relationen in der Steuerungssoftware analysiert werden. Bezugnehmend auf die Änderungsdetektion aus dem realen automatisierten System mittels SPS-Steuerungssoftwareanalyse müssen die beiden vorstehend beschriebenen Punkte in Betracht gezogen werden. In der Ankerpunktmethode wird eine Formalisierungsmethode zur Überwindung der obigen Herausforderungen eingeführt.

Nachfolgend werden die Analysemöglichkeiten der SPS-Steuerungssoftware beschrieben und erläutert, weshalb der Formalisierungsansatz ein geeigneter Ansatz zur Bewältigung der Herausforderungen sein kann. Darüber hinaus werden die einzelnen Schritte des Formalisierungsansatzes innerhalb der Ankerpunktmethode ausführlich beschrieben.

4.2.2.1 Modellgenerierung aus der SPS-Steuerungssoftware

Nachdem zwei XML-Dateien aus beiden SPS-Steuerungssoftware erzeugt wurden, kann das XML-Parsing verwendet werden, um die Ankerpunkte und ihre Relationen zu erkennen. XML-Parsing ist ein Mechanismus für den Zugriff und die Bearbeitung eines in XML erstellten Dokuments. Dafür gibt es eine Vielzahl an Tools. Diese Tools stellen eine Fülle von Algorithmen für den Zugriff auf XML-Elemente bereit, die je nach Zweck des Parsens verwendet werden. Das Parsen der XML-Dateien, um die Ankerpunkte und ihre Relationen in den einzelnen SPS-Steuerungssoftwares zu erkennen und miteinander zu vergleichen, beinhaltet einige Herausforderungen. Die Wahl des richtigen Parsers für eine Anwendung ist eine kritische Entscheidung, da unsachgemäßes Parsen zu Leistungsverlusten und Produktivitätsverlusten führt. Im Rahmen dieser speziellen Anwendung hier gibt es in bestehenden Technologien keinen geeigneten XML-Parser für die Analyse der Ankerpunkte innerhalb der SPS-Steuerungssoftware. Die Herausforderung besteht demnach darin, dass nur für diese Anwendung ein spezieller XML-Parser erstellt werden muss, was einen damit verbundenen manuellen Aufwand erfordert. Eine weitere Herausforderung ergibt sich darin, dass es ein erheblicher Aufwand ist, XML-Dateien aus der SPS-Steuerungssoftware eines automatisierten Systems zu analysieren, das aus vielen Assets mit einem hohen Automatisierungsgrad besteht, wie es beispielweise in der Automobilindustrie der Fall ist. Denn diese XML-Dateien enthalten eine große Anzahl von Informationen, die für die Änderungsdetektionsanalyse nicht erforderlich sind.

Ein weiterer Ansatz zur Analyse dieser XML-Dateien ist andererseits die Formalisierung, worunter die Beschreibung von einem Programm durch eine formale Sprache zu verstehen ist. Eine formale Sprache entspricht einer eindeutig definierten Notation von Zeichen, Wörter, Symbole oder Phrasen. Younis erklärt in [147], dass in der Literatur zwei Hauptanwendungen bzw. Gründe für die Formalisierung der SPS-Steuerungssoftware angegeben werden. Der erste weist auf die Notwendigkeit der formalen Verifizierung und Validierung sowie Analyse bestehender Systeme aufgrund eines erhöhten Sicherheits- und Qualitätsbewusstseins hin. Der zweite Grund beruht auf dem ständigen Fortschritt in der Produktion und ihrer Automatisierung. Er betrifft die Änderung der bestehenden Steuerungssoftware, um den neuen Produktionsanforderungen gerecht zu werden. Ausgehend von den bestehenden Formalisierungsmethoden und unter Berücksichtigung des Konzepts dieser Arbeit können die beiden erzeugten XML-Dateien aus der Steuerungssoftware formal beschrieben und nur aus diesen formal beschriebenen Modellen die wesentlichen Elemente (die Ankerpunkte und ihre Relationen) untersucht werden. Im Vergleich zum Parsing-Ansatz hat der Formalisierungsansatz den Vorteil, dass er unabhängig von der XML-Spezifikation ist, die aus der SPS-Steuerungssoftware generiert wird. Im Formalisierungsansatz kann eine formale Beschreibung der Steuerungssoftware unter Verwendung vorhandener XML-Formalisierungsmethoden automatisiert erzeugt werden. In den folgenden Prozessschritten der Ankerpunktmethode kann das generierte Modell mit einer abstrahierten Methode verallgemeinert und für eine effiziente

Analyse zur Verfügung gestellt werden. Dies trägt dazu bei, den Analyseaufwand zu reduzieren. Der Formalisierungsansatz hat daher nicht die beiden Herausforderungen des XML-Parsing-Ansatzes: "Manueller Aufwand zur Erstellung eines XML-Parsing-Algorithmus in Abhängigkeit der XML-Spezifikation" und "hoher Aufwand zur Analyse der gesamten XML-Datei". Folglich wird für die Phase der Änderungsdetektion der Ankerpunktmethod der Formalisierungsansatz ausgewählt.

Die Abbildung 25 zeigt die Prozessschritte der Änderungsdetektion in der Ankerpunktmethod mittels Formalisierung. Diese Formalisierung besteht aus drei Hauptschritten, (1) Transformation der Elemente innerhalb der Steuerungssoftware in XML-Dateien, (2) Generierung einer zentralen, formalen Beschreibung aus diesen XML-Dateien und (3) Abstraktion und Vergleich des beschriebenen Modells mit ihren Ankerpunkten und Relationen, die für die Analyse erforderlich sind. Dabei wird zunächst die transformierte Steuerungssoftware im XML-Format durch eine Formalisierungsmethod in eine formale Beschreibung überführt. Diese enthält eine detaillierte Beschreibung aller Elemente und ihrer Variablen sowie interne Verknüpfungen innerhalb der SPS-Steuerungssoftware in Form von zwei Modellen. Allerdings sind viele Einzelheiten innerhalb dieser Steuerungssoftwaremodelle im Hinblick auf die Änderungserkennung durch Analyse der SPS-Steuerungssoftware irrelevant. Aufgrund der möglichen Änderungen, die im System auftreten können sowie deren Auswirkungen auf die Software, sind viele dieser Details nicht zwingend erforderlich, um die Veränderungen im realen System zu erkennen. Die hohe

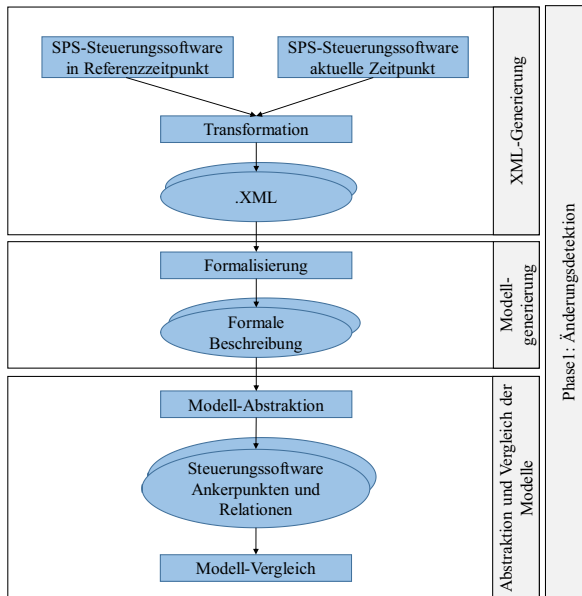


Abbildung 25: Änderungsdetektion anhand des Formalisierungsprozesses der SPS-Steuerungssoftware in der Ankerpunktmethod

Komplexität der Details kann einen automatisierten Vergleich zwischen den Modellen der SPS-Steuerungssoftware in einem komplizierten automatisierten System aufwendiger gestalten. Daher wird in diesem Ansatz eine Modellabstraktion vorgesehen.

Hierbei werden im nächsten Schritt die formal generierten Modelle für eine effiziente Analyse abstrahiert. Abstraktion bedeutet hier, dass ein Modell bestehend nur aus Ankerpunkten und deren Relationen innerhalb der Steuerungssoftware erstellt werden muss. Das Thema Modellabstraktion wird im folgenden Abschnitt näher beleuchtet.

4.2.2.2 Abstraktion und Vergleich der Modelle

Wie bereits erwähnt, erfordert eine effiziente Analyse des formalisierten Softwaremodells, dass es in die notwendigen Elemente zur Änderungserkennung zusammengefasst wird. Dementsprechend müssen die Bestandteile innerhalb des abstrakten Modells basierend auf den für die Analyse notwendigen Informationen ausgewählt werden. Dabei ist es von entscheidender Bedeutung zu erkennen, welche Elemente innerhalb der SPS-Steuerungssoftware sich durch welche Art von Änderung im realen System ändern. In diesem Zusammenhang müssen die domänenübergreifenden Änderungen, die im realen System auftreten, sowie deren Auswirkungen auf die SPS-Steuerungssoftware-Elemente untersucht und regelbasiert kategorisiert werden. Zu diesem Zweck wurde im Rahmen dieser Forschung eine Studie durchgeführt und eine domänenübergreifende Regeltabelle erarbeitet. Die Regeln für die regelbasierte Analyse werden in Abschnitt 4.2.3.1 erläutert. Gemäß den in Abschnitt 4.2.3.1 festgestellten domänenübergreifenden Regeln genügt es, nur die Signale, Funktionsblöcke und Datenblöcke innerhalb der SPS-Steuerungssoftware als Ankerpunkte der Assets in der Softwaredomäne und deren Innenrelationen im Rahmen der Änderungserfassung zu untersuchen. Die Innenrelationen sind die Modellierung der Interaktion zwischen Signalen und den Transitionen sowie Schritten von Funktionsblöcken und den gespeicherten Daten von Signalen und Funktionsblöcke in Datenblöcken. Die Regeln für die regelbasierte Analyse können jederzeit einfach erweitert und damit die notwendigen Elemente in der Modellabstraktion angepasst werden. Zusammengefasst bezeichnet die Modellabstraktion, dass von jeder SPS-Steuerungssoftware nur ein Modell generiert werden muss, das Informationen über Signale, Funktions- und Datenblöcke und deren Innenrelationen aus der formal beschriebenen SPS-Steuerungssoftware bereitstellt. Die Abstraktionsklassen der Signale entsprechen der in Abschnitt 4.1.2 beschriebenen Namenskonvention und beinhalten die geräte-, orts- und funktionsorientierten Strukturinformationen des Assets im System, basierend auf der DIN-Norm EN 81346. Außerdem die funktionale Beschreibung des Assets im System, die eindeutige ID und die Hardwareadresse des Assets sowie der Datentyp des Signals. Die Funktions- und Datenblockvariablen bestehen ebenfalls aus einem eindeutigen Namen der Funktionsgruppe. Abbildung 26 veranschaulicht eine einfache Darstellung des Abstraktionsmodells der SPS-Steuerungssoftware, bestehend aus den Ankerpunkten und deren Relationen. Nach der Erstellung von zwei Modellabstraktionen aus der

formal beschriebenen SPS-Steuerungssoftware müssen die Ankerpunkte und ihre Relationen innerhalb der jeweiligen Steuerungssoftware miteinander abgeglichen werden. Wenn die beiden Abstraktionsmodelle keine Unterschiede aufweisen, ist davon auszugehen, dass innerhalb der mechatronischen Komponenten des realen Systems keine Änderungen stattgefunden haben. Werden jedoch Änderungen durch den Modellvergleich festgestellt, werden die geänderten Ankerpunkten und deren Relationen für die regelbasierte Analyse zur domänenübergreifenden Abhängigkeitsdetektion im System zur Verfügung gestellt.

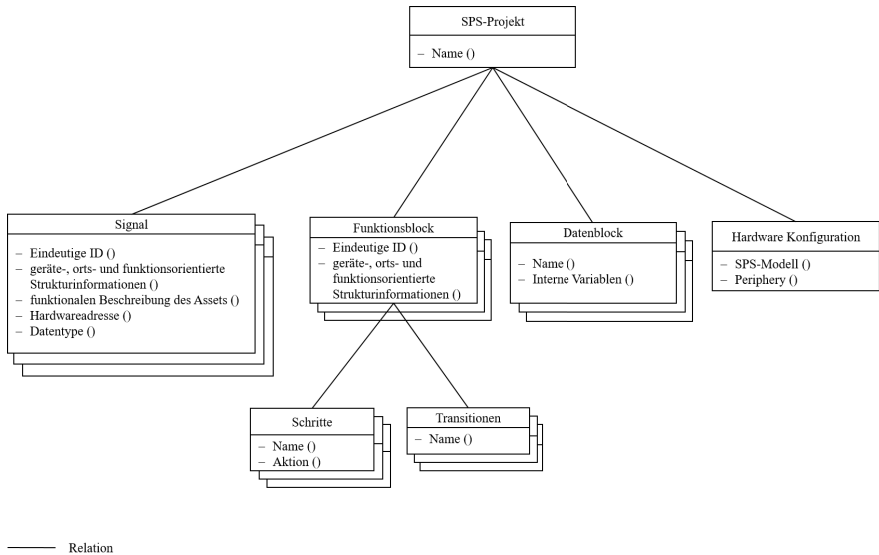


Abbildung 26: Abstraktionsmodell der formalen Beschreibung der SPS-Steuerungssoftware

Im nächsten Abschnitt wird die automatisierte domänenübergreifende Erkennung von Abhängigkeiten innerhalb der Ankerpunktmethode behandelt.

4.2.3 Phase 2: Domänenübergreifenden Abhängigkeitsdetektion

Nachdem die geänderten Ankerpunkte und ihre Relationen in der Softwaredomäne erkannt wurden, müssen sie für eine weitere Informationsgewinnung analysiert werden. Aus dieser Analyse müssen die folgenden drei Fragen für die domänenübergreifende Anpassung des Gesamtsystemmodells im nächsten Schritt beantwortet werden:

- Was sind die exakten Änderungsszenarien, die im realen System vorgekommen?
- Welche Ankerpunkte müssen dementsprechend in den drei Domänen Mechanik, Elektrik und Software im Gesamtsystemmodell modifiziert, gelöscht oder hinzugefügt werden?

- Welche internen Relationen zwischen den modifizierten, hinzugefügten oder gelöschten Ankerpunkten sowie den anderen Ankerpunkten des Gesamtsystemmodells müssen in den einzelnen Domänen angepasst werden, um das Gesamtsystemmodell konsistent zu halten?

In Bezug auf den logischen Aufbau der SPS-Steuerungssoftware sind die möglichen Änderungsszenarien, die im automatisierten System aus der Analyse der statischen Daten in der SPS-Steuerungssoftware erkannt werden können, beispielsweise

- das Hinzufügen oder Entfernen von mechatronischen Komponenten,
- das Hinzufügen oder Entfernen von Funktionsgruppen bestehend aus mehreren mechatronischen Komponenten mit gemeinsamer Funktionalität im System,
- die neue Verdrahtung von mechatronischen Komponenten,
- die Änderung der Modularisierungsposition von mechatronischen Komponenten,
- das Ändern der Schritte des Prozessverhaltens von Funktionsgruppen im System oder
- das Ändern des hardwaretechnischen Aufbaus der SPS (Hardwarekonfiguration).

Die Analyse der geänderten Ankerpunkte und ihrer internen Relationen in der SPS-Steuerungssoftware erlaubt es, diese Änderungsszenarien im realen System automatisiert zu identifizieren. Durch das Ermitteln der Antwort auf die erste Frage und die Einbeziehung eines domänenübergreifenden fachlichen Wissens lässt sich feststellen, wie das auftretende Änderungsszenario im Gesamtsystemmodell in den einzelnen Domänen angepasst werden muss. Daraus lässt sich ableiten, ob dem Gesamtsystemmodell zum Referenzzeitpunkt neue Ankerpunkte hinzugefügt oder einige bestehende Ankerpunkte aus dem Gesamtsystemmodell gelöscht werden müssen oder ob es ausreicht, nur die internen Relationen zwischen einigen der vorhandenen Ankerpunkte in der betroffenen Domäne anzupassen. Unter Berücksichtigung der zweiten Frage können, wie im Abschnitt 4.1.2 beschrieben, alle wiederverwendbaren Modelle eines Assets in den Domänen Mechanik und Elektrik aus einer Komponentenbibliothek abgerufen werden. Für den Fall, dass ein Asset dem System hinzugefügt wurde, müssen alle wiederverwendbaren Modelle des Assets aus der Komponentenbibliothek mit seiner ID, die bei der Analyse der Steuerungssoftware erkannt wurde, abgerufen werden. Dabei müssen die Variablen der domänenübergreifenden wiederverwendbaren Modelle angepasst und als Ankerpunkte des neuen Assets in das Gesamtsystemmodell eingefügt werden. Im Falle, dass ein Asset aus dem automatisierten System entfernt wurde, müssen seine Ankerpunkte innerhalb des Gesamtsystemmodells gelöscht werden. Zu diesem Zweck sind seine Ankerpunkte im Gesamtsystemmodell durch die eindeutige ID des betreffenden Assets, die aus der Analyse der Steuerungssoftware ermittelt wurde, direkt anzusprechen. In anderen Änderungsfällen müssen nur die Relationen der Ankerpunkte des geänderten Assets zu anderen Assets angepasst werden. Zur Beantwortung der dritten Frage müssen die domänenübergreifenden Relationen zwischen den

geänderten, hinzugefügten oder gelöschten Ankerpunkten und den anderen Ankerpunkten des Gesamtsystemmodells aus der Analyse der SPS-Steuerungssoftware erkannt werden. Die domänenübergreifenden Relationen können in drei Kategorien unterteilt werden: Parent-Child-, Instanz-Instanz- und Vererbungsrelation zwischen den Modellen.

Parent-Child Relation: Die Parent-Child-Relation bezieht sich auf die Modularisierung bzw. Gruppierung der Ankerpunkte der Assets im Gesamtsystemmodell in den einzelnen Domänen. Im Rahmen der Anpassung der Relationen ist es notwendig zu wissen, wo die Ankerpunkte des geänderten Assets im Gesamtsystemmodell angeordnet werden müssen. Wie im Abschnitt 4.1.2 beschrieben, modularisieren die Konstrukteure während des Engineering-Prozesses die Komponentenmodelle der Assets im domänenübergreifenden Gesamtsystemmodell unter unterschiedlichen Gesichtspunkten mit ihren Engineering-Tools. In diesem Zusammenhang ist bei der Analyse der SPS-Steuerungssoftware ein wesentlicher Aspekt zu beachten: Zur Anpassung der Parent-Child-Relation der geänderten Ankerpunkte im domänenübergreifenden Gesamtsystemmodell müssen deren geräte-, standort- und funktionsorientierte Modularisierungspositionen ermittelt werden.

Instanz-Instanz Relation: Eine andere Relation innerhalb des Gesamtsystemmodells ist die Instanz-Instanz-Relation. Als Instanz-Instanz-Relationen sind die Relationen zu bezeichnen, die zwischen Objektinstanzen innerhalb des Gesamtsystemmodells in den einzelnen Domänen bestehen. Dabei kann die Verbindung vom Eingangssignal eines Sensors zu den Transitionsbedingungen der Funktionsblöcke anderer Komponenten im System innerhalb der Softwaredomäne als Instanz-Instanz-Relation betrachtet werden oder die Verkabelungen zwischen dem schematischen Modell des Assets und den schematischen Modellen der anderen Assets des automatisierten Systems in die elektrische Domäne. Ebenso sind die kinematischen Verhaltensabhängigkeiten des Assets mit anderen Assets im System in der mechanischen Domäne zu reflektieren. Zur Synchronisierung der Modelle des Gesamtsystemmodells müssen besonders diese Instanz-Instanz-Relationen in allen Domänen konsistent aktualisiert werden, damit eine domänenübergreifende Simulation innerhalb des Digitalen Zwillings, beispielsweise zur virtuellen Inbetriebnahme bei einer Rekonfiguration, durchgeführt werden kann.

Vererbungsrelation: Zur Erklärung der Vererbungsrelation ist zunächst die Modellierung von Komponenten während des Engineering-Prozesses des Systems zu betrachten. Während des Engineering-Prozesses wird ein Modell eines Assets nur einmal mit einer eindeutigen ID erstellt. Dieses Modell kann während des Designprozesses sehr oft modifiziert werden, aber am Ende werden diese unter der eindeutigen ID bzw. dem Namen des so genannten Basismodells zusammen als Reversionen referenziert. Bei der Wiederverwendung dieses Basismodells in dem Gesamtsystemmodell können alle gleichen Modelle aus dem Basismodell vererbt (instanziiert) und müssen nicht immer wieder neu konstruiert werden. Diese, vom Basismodell vererbten Modelle, werden als Instanz-Modelle bezeichnet. Das führt zu einer hohen Effizienz im Engineering-Prozess, ist aber ein kritisches Thema bei Änderungen in den Modellen des

Gesamtsystemmodells. Die Relation zwischen den Basismodellen und vererbten Modellen muss während des Anpassungsprozesses der Modelle des Gesamtsystemmodells immer beibehalten werden. Wenn beispielsweise das Basismodell während der Anpassung gelöscht oder geändert wird, gehen die gesamten vererbten Modelle und deren Abhängigkeiten verloren und das Gesamtsystemmodell wird inkonsistent. Dies bedeutet, dass im Rahmen einer Modellanpassung innerhalb des Gesamtsystemmodells das Basismodell eines Assets nicht gelöscht werden darf. Für den Fall, dass ein Instanz-Modell eines Basismodells angepasst werden muss, ist das Basismodell entsprechend der notwendigen Änderung zu versionieren und das Instanz-Modell durch eine sogenannte referenzierende Regel bei der letzten Reversion unter dem Basismodell in Beziehung gesetzt werden. So gehen die anderen Vererbungsrelationen an Basismodell nicht verloren. Das Thema Modellanpassung wird in Abschnitt 4.2.4.2 ausführlicher behandelt.

4.2.3.1 Regeln zur Analyse der SPS-Steuerungssoftware-Abstraktionsmodelle

Mit der Analyse der SPS-Steuerungssoftware müssen im ersten Schritt die Änderungsszenarien aus dem realen System automatisiert erkannt werden. Eine automatisierte Erkennung von domänenübergreifenden Änderungsszenarien ist durch einen Vergleich der Ankerpunkte und Relationen aus dem aktuellen Abstraktionsmodell der SPS-Steuerungssoftware und dem Abstraktionsmodell der SPS-Steuerungssoftware zum Referenzzeitpunkt möglich. Dies erfordert den Aufbau einer Wissensbasis, damit die Änderungen in der SPS-Steuerungssoftware zur Verfolgung des domänenübergreifenden Änderungsszenarios aufgenommen werden können. Um eine Wissensbasis zu schaffen, muss eine Studie über mögliche Änderungen, die in automatisierten Systemen in der diskreten Fertigung auftreten können, durchgeführt werden. Dabei ist es notwendig, eine individuelle Regel zwischen jeweils diesen Änderungsszenarien im realen System und ihrem Einfluss auf die SPS-Steuerungssoftware zu definieren. Im Abschnitt 4.1.2 wurde eine Namenskonvention auf Basis der DIN EN 81346 zur Benennung der Ankerpunkte der Assets innerhalb der SPS-Steuerungssoftware eingeführt und im Detail beschrieben. Basierend auf dieser Namenskonvention, die während der Engineering- und Betriebsphase des Systems vorgenommen werden muss, liefert die Benennung der Signale in der Steuerungssoftware Informationen über die geräte-, orts- und funktionsorientierte Modularisierung des zugehörigen Assets im System. Außerdem liefert sie Informationen über die eindeutige ID des Assets, über die Hardwareadresse für die Verbindung des Assets am Steuerungssystem und die Funktionsbeschreibung des Assets im System. Funktionsblöcke und Datenblöcke innerhalb der SPS-Steuerungssoftware des Systems tragen in ihre Benennung auch einen eindeutigen Namen und eine Modularisierungsposition des zugehörigen Assets. Basierend auf dieser Benennungsstandardisierung und einer Studie hinsichtlich der automatisierten Systeme in der diskreten Fertigung wurde die erforderliche Wissensbasis für die automatisierte Analyse der Steuerungssoftware aufgebaut. Dabei wurden die erforderlichen Regeln für die Analyse auf Basis einer umfangreichen praxisorientierten Studie im Hinblick auf eventuell auftretende Änderungsszenarien im System und deren Einfluss auf die jeweiligen Ankerpunkte der Assets

sowie auf die internen Relationen in der SPS-Steuersoftware zusammengetragen. Die Tabelle 4 listet einige dieser Regeln auf.

Tabelle 4: Die Regeltabelle zur domänenübergreifenden Änderungsdetektion aus SPS-Steuerungssoftware

<div>Änderungen in der Steuerungssoftware</div> <div>Änderungsszenario im automatisierten System</div>	Neues I/O-Signal (mit neuer ID) in der Signalliste	Neue Positionsadresse im Namen eines vorhandenen I/O-Signals	Einsatz des neuen Signals in Schritten und/oder Transitionen eines bestehenden Funktionsblocks	Neuer Funktionsblock mit neuer ID	Gelöschte I/O-Signale mit eindeutiger ID in der Signalliste	Gelöschte Signale in Schritten und/oder Transitionen im bestehenden Funktionsblock	Gelöscht Funktionsblock/-blöcke	Neue/gelöschte Schritte und/oder Transitionen im bestehenden	Modifikation von Schritten und/oder Transitionen eines bestehenden	Änderung der Hardware-Adresse von vorhandenem I/O-Signal.	Neue Software-Version	Neues/gelöschtes Marker-Signal in der Signalliste auch in Funktionsblock
1. Hinzufügen einer mechatronischen Komponente	✓		✓									
2. Hinzufügen einer Funktionsgruppe	✓		(✓)	✓								
3. Entfernen einer mechatronischen Komponente					✓	✓						
4. Entfernen einer Funktionsgruppe					✓	(✓)	✓					
5. Optimierung einer Funktionsgruppe (1)			✓									
6. Optimierung einer Funktionsgruppe (2)						✓						
7. Optimierung einer Funktionsgruppe (3)												✓
8. Hinzufügen einer unbenutzten mechatronischen Komponente	✓											
9. Entfernen einer unbenutzten mechatronischen Komponente					✓							
10. Verhaltensänderung einer Funktionsgruppe (1)								✓				
11. Verhaltensänderung einer Funktionsgruppe (2)									✓			
12. Verdrahtungspositionsänderung einer mechatronischen Komponente										✓		
13. Ändern der Fixposition einer mechatronischen Komponente		✓										
14. Update/Downgrade der Software-Version											✓	

✓ muss geschehen

(✓) kann ebenfalls geschehen

Diese Tabelle kategorisiert eine Reihe von Änderungsszenarien, die bei der physischen Fertigungszelle auftreten, und wie sie anhand des Vergleichs von SPS-Steuerungssoftware identifiziert werden können. Mit diesen Regeln lassen sich die Unterschiede innerhalb der Variablen von Signalen, Funktions- und Datenblöcken sowie die Beziehungen zwischen Signalen mit Schritten und Transitionsbedingungen der Steuerungssoftware zu beiden Zeitpunkten analysieren. So ermöglichen die Regeln eine automatisierte Erkennung von Änderungsszenarien im realen System.

Neben der Erkennung von Änderungsszenarien müssen die Ankerpunkte der geänderten Assets im System und deren domänenübergreifende Relationen automatisiert ermittelt werden; dies kann auf Basis einer Analyse mithilfe des Entscheidungsbaums erfolgen.

4.2.3.2 Regelbasierter Ansatz mittels eines Entscheidungsbaums

Im vorherigen Abschnitt wurden mehrere Regeln für eine regelbasierte Analyse der abstrakten Modelle der SPS-Steuerungssoftware zum Referenzzeitpunkt und zum aktuellen Zeitpunkt vorgestellt. Zur automatisierten Durchführung der regelbasierten Analyse wird in der Ankerpunktmethod eine Analyse mittels einer Entscheidungsbaumklassifikation konzipiert. Die Anwendung des Entscheidungsbaums als Lernmethode für Data Mining zur automatisierten Klassifizierung und Analyse von Daten ist sehr weit verbreitet [148], [149]. Eine weitere Anwendung von Entscheidungsbäumen wäre die Datenobjektanalyse basierend auf der semantischen Beschreibung von Daten und Regeln, die Variablenauswahl und Klassifizierung. Ein Entscheidungsbaum ist ein strukturiertes Modell von Entscheidungen und deren möglichen Schlussfolgerungen, welches mit Hilfe der Klassifizierung von Datenobjekten und Regeln erstellt wurde, um Entscheidungsaufgaben zu lösen. Die Grundidee dieses Ansatzes ist es, eine komplexe Entscheidung in eine Vereinigung mehrerer einfacherer Entscheidungen zu zerlegen [150]. Jeder Entscheidungsbaum ist aufgebaut aus einem Wurzelknoten und beliebig vielen internen Knoten, die für die Attributprüfung in jedem Analyseschritt stehen, sowie von mindestens zwei Klassifizierungslabels oder Blättern. Die Blätter sind die Entscheidungen, die getroffen werden, nachdem alle Attribute berechnet wurden. Jeder Knoten repräsentiert eine logische Regel und jedes Blatt eine Antwort auf das Entscheidungsproblem. Die Pfade von der Wurzel bis zum Blatt stellen die Klassifizierungsregeln dar. Zur Bestimmung eines Klassifizierungslabels für ein einzelnes Datenobjekt in den Entscheidungsbäumen wird das Entscheidungsverfahren vom Wurzelknoten entlang des Baumes nach unten durchgeführt. Dabei wird bei jedem Knoten ein Attribut abgefragt und eine Entscheidungsfindung hinsichtlich der Selektion des nächsten Knotens getroffen. Dieser Prozess wird fortgesetzt, bis ein Klassifizierungslabel erzielt wird. Die Klassifizierung der Daten und Regeln eines Entscheidungsbaums können entweder automatisiert durch eine überwachte Lernmethode oder direkt durch den Anwender auf Basis vorhandener Regeln und fachspezifischen Kenntnissen, wie im Fall dieser Arbeit, für die Analyse der Daten erstellt werden. Abbildung 27 zeigt einen Ausschnitt aus dem Entscheidungsbaum, der

exemplarisch den Prozessablauf zur Erkennung des Änderungsszenarios "Neue Funktionsgruppe wurde dem System hinzugefügt" und dessen domänenübergreifende Abhängigkeiten klassifiziert.

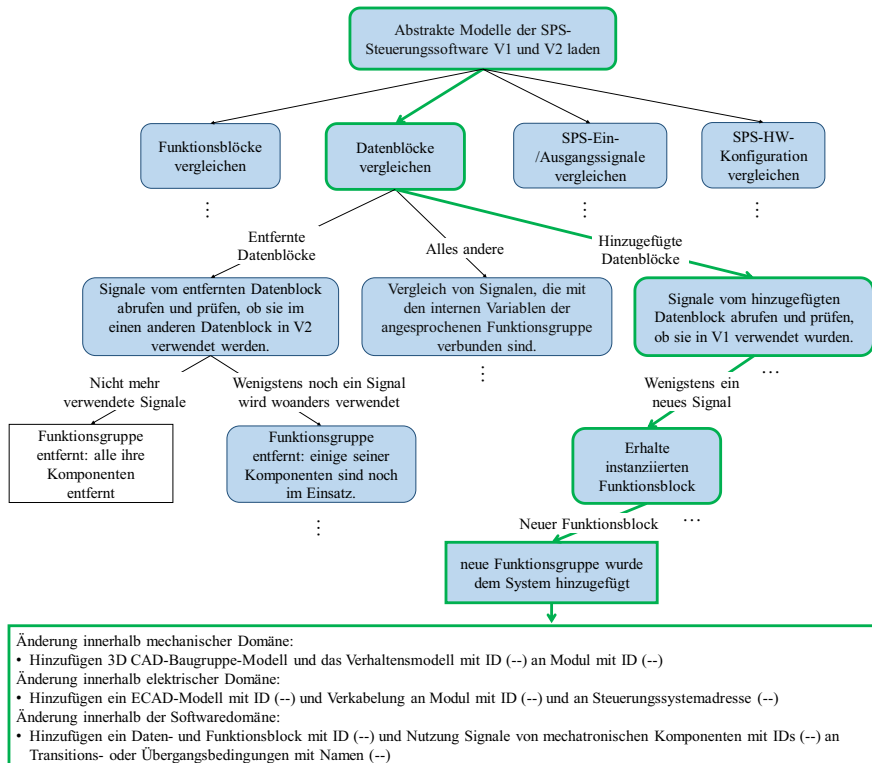


Abbildung 27: Vorgehen zur Erkennung des Klassifizierungslabls "Neue Funktionsgruppe wurde dem System hinzugefügt" im Entscheidungsbaum

Wie im Entscheidungsbaum zu sehen ist, werden die relevanten Attribute geänderter Ankerpunkte in den beiden abstrakten Modellen der SPS-Steuerungssoftware regelbasiert geprüft und entlang des Baumes nach unten analysiert. Um das Änderungsszenario zu erkennen, muss der Entscheidungsbaum die geänderten Ankerpunkte und deren Relationen nach den vordefinierten Regeln in der Regeltabelle analysieren. Jedoch müssen auch andere Variablen durch den Entscheidungsbaum untersucht werden, um domänenübergreifende Abhängigkeiten und notwendige Maßnahmen zur Anpassung der Modelle innerhalb des Gesamtsystemmodells zu identifizieren. Dazu werden im Entscheidungsbaum die Variablen innerhalb der Namen von Signalen, Funktions- und Datenblöcken nach weiteren Regeln analysiert. Beispielsweise werden nach der Erkennung des Änderungsszenarios "Neue Funktionsgruppe wurde dem System hinzugefügt" die Benennungsvariablen des Funktionsblocks und die im Funktionsblock

verwendeten Signale sowie die Benennungsvariablen der zugehörigen Datenblöcke regelbasiert im Entscheidungsbaum analysiert, um die Anpassungsmaßnahmen im Klassifizierungslabel am Schluss des Entscheidungsbaums zu erreichen. Basierend auf diesen Regeln werden entsprechend dem jeweiligen Änderungsszenario die Informationen innerhalb dem Namen der Ankerpunkte untersucht und Anpassungsmaßnahmen festgelegt.

Dementsprechend werden durch das Klassifizierungslabel im Entscheidungsbaum alle zuvor offenen Fragen vom Anfang des Abschnitts 4.2.3 geklärt. Der Entscheidungsprozess vom Wurzelknoten entlang des Baumes nach unten wird auf Basis von Attributprüfungen durchgeführt, um ein Änderungsszenario als Klassifizierungslabel zu erreichen. Bei der Entscheidungsfindung wird die eindeutige ID des geänderten Assets im System als entscheidende Variable übernommen. Darüber hinaus erkennt der Entscheidungsbaum weitere Relationen, wie beispielsweise Informationen über Verkabelungsänderungen in der elektrischen Domäne oder, Verhaltensänderungen von Assets in der mechanischen Domäne. Dafür führt der Entscheidungsbaum eine Analyse der anderen Informationen von Ankerpunkten, wie Hardwareadresse, Funktionsbeschreibung und Datentyp der Signale, sowie der internen Beziehungen zwischen Signalen, Funktionsblöcken und Datenblöcken in der Softwaredomäne durch. Nach dem Erkennen der Änderungen und ihrer domänenübergreifenden Abhängigkeiten müssen diese automatisiert in die Modelle des Digitalen Zwillings eingefügt und dabei die Konsistenz des Gesamtsystemmodells gewahrt werden.

Im nächsten Kapitel werden verschiedene Ansätze zur automatisierten Modellanpassung vorgestellt und deren Vor- und Nachteile diskutiert. Auf dieser Grundlage wird eine Methode vorgestellt, die auf einem Change-Management-Prozess basiert.

4.2.4 Phase 3: Modellanpassung

Aus der Änderungsdetektion und Abhängigkeitsdetektion in früheren Schritten der Ankerpunktmethode stehen bereits die Informationen über das geänderte Zielmodell bzw. den geänderten Zielankerpunkt (orts-, geräte- und funktionsorientierte Modularisierungsposition) sowie die Parent-Child Relation und die Instanz-Instanz-Relationen zwischen den Ankerpunkten im Gesamtsystemmodell zur Verfügung. Darüber hinaus werden die genauen Änderungsszenarien und die eindeutige ID der wiederverwendbaren Modelle der geänderten Assets bereitgestellt. Ausgehend von diesen gewonnenen Informationen aus der Analyse der Steuerungssoftware wird im vorliegenden Prozessschritt der Ankerpunktmethode ein Lösungsansatz zur automatisierten Modellanpassung innerhalb des Digitalen Zwillings vorgestellt. Dabei gibt es zwei wesentliche Punkte, die bei der automatisierten Anpassung der Modelle eines automatisierten Systems mit einem externen Softwaresystem zu beachten sind. Erstens müssen Änderungen unter Beibehaltung der Semantik der Modelle des Digitalen Zwillings vorgenommen werden; North definiert in [151] Semantik durch eine Kategorisierung von Zeichnen, Daten und Semantik in einer „Wissenstreppe“. Nach [151] werden Zeichen, wie

z.B. Buchstaben, Ziffern und Sonderzeichen durch Ordnungsregeln als Daten erkannt, d.h. sind Daten eine Integration von Zeichen mit einer Syntax, die noch nicht interpretiert sind. Zu Information bzw. Semantik werden diese Daten, wenn sie in einem Bedeutungskontext stehen. Die Tools erstellen ihre Modelle in einem Datenformat mit ihrer eigenen spezifischen Semantik. Ein Datenformat enthält dabei mindestens eine Semantik [68]. Die Tools, die diese spezifische Semantik erfassen können, sind in der Lage, die Modelle zu lesen und zu bearbeiten. Im Hinblick auf die Wiederverwendung der modifizierten Modelle des Digitalen Zwillings nach dem Anpassungsprozess für verschiedene Anwendungen wie Rekonfiguration, Simulation usw. durch die Tools ist es notwendig, die ursprüngliche Semantik der Modelle einzuhalten. Ein weiterer wichtiger Punkt ist, dass bei der Anpassung der Modelle keine Relationen im Gesamtsystemmodell verloren gehen dürfen, die sich im realen System nicht verändert haben. Vor diesem Hintergrund müssen die möglichen Lösungsansätze zur Modellanpassung des Gesamtsystemmodells des Digitalen Zwillings in der Lage sein, alle Modelle und Relationen bis ins kleinste Detail zu importieren und exportieren. Andernfalls können durch den Import und Export einige Relationen verloren gehen und damit das Gesamtsystemmodell inkonsistent werden.

4.2.4.1 Mögliche Ansätze zur Modellanpassung

Unter Berücksichtigung der oben genannten Punkte zur automatisierten Ausführung der Modellanpassung durch externe Software können drei Ansätze nach dem Stand der Wissenschaft und Technik in Betracht gezogen werden. Diese werden im Folgenden erläutert.

Überschreiben der Modelle

Ein möglicher Ansatz für die Aktualisierung der Modelle des Digitalen Zwillings ist die Überschreibung der Modelle mit Hilfe eines XML-Konverters und einer XML-Bearbeitungsmethode. Dieser Ansatz umfasst vier aufeinanderfolgende Schritte, die in Abbildung 28 dargestellt sind. Im ersten Schritt werden die Modelle des Digitalen Zwillings im XML-Format mittels eines XML-Konverter-Tools aus dem Datenbackbone importiert und dem Softwaresystem zur Modellanpassung zur Verfügung gestellt. Die Modelle des Digitalen Zwillings werden in einem Datenbackbone gespeichert. Der Datenbackbone kann ein Datenbanksystem mit verschiedenen Speicherprinzipien sein, wie z.B. relationale Datenbanken, Graphen-Datenbanken, dokumentorientierte Datenbanken, spaltenorientierte Datenbanken, hierarchische Datenbanken und Netzwerkdatenbanken oder es kann sich um einen unstrukturierten Speicher handeln. Im nächsten Schritt müssen die Adresse, Relationen und Attribute der betroffenen Ankerpunkte im Gesamtsystemmodell im XML-Format untersucht und durch die XML-Parsing-Methode ermittelt werden. Das XML-Parsing ist ein Mechanismus für den Zugriff und die Auswahl von Daten aus einem in XML erstellten Dokument. Im dritten Schritt muss eine Manipulation innerhalb des XML-Dokuments durchgeführt werden. Unter der Definition von XML-Manipulationsaktivitäten werden die Tätigkeiten wie Datenfilterung, Datenumstrukturierung, Datenextraktion,

Datenmodifikation, Dateneinfügung, Datenlöschung usw. zusammengefasst [152]. Schließlich wird in Schritte 4 der synchronisierte Digitale Zwilling ins Datenbackbone importiert.

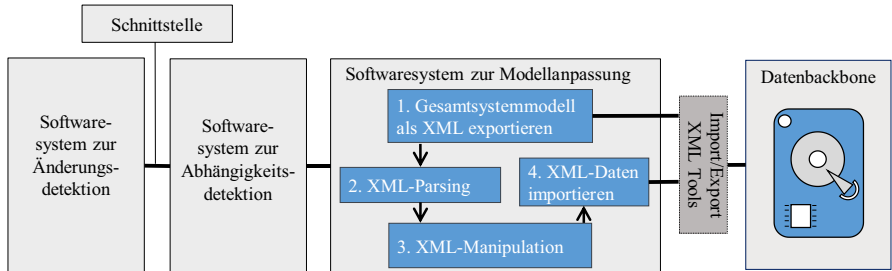


Abbildung 28: Prozessablauf des Lösungsansatzes basierend auf dem Überschreiben der Modelle

In diesem Ansatz können anstelle von XML-Parsing und XML-Manipulation die Formalisierungsansätze und Algorithmen zur Modellmanipulation in den Schritten zwei und drei verwendet werden, was im Kontext der Vermeidung der Herausforderungen diesen Ansatz nicht verändert. Um die wesentlichen Herausforderungen bei der Anwendung dieses Ansatzes zu erläutern, können die folgenden Punkte skizziert werden.

Keine Rückverfolgbarkeit für die Systemingenieure: Im besten Fall, wenn alle Änderungen mit vollständiger Detailtiefe aus der Analyse erkannt wurden, kann dieser Ansatz zu einem höheren Automatisierungsgrad bei der Anpassung der Modelle des Digitalen Zwillings führen. Allerdings fokussiert sich das in dieser Arbeit vorgestellte Konzept auf die Erkennung von Änderungen durch Analyse der SPS-Steuerungssoftware; da die durch die Analyse der SPS-Steuerungssoftware erfassten Änderungen keine Informationen wie die exakte Koordinaten der geänderten Fixierposition oder die exakte geänderte kinematische Bewegung des geänderten Assets etc. im automatisierten System enthalten, müssen noch einige zusätzliche Anpassungen im Rahmen der Synchronisierung durch die verantwortliche Ingenieure für jedes der betroffenen Modelle des Digitalen Zwillings vorgenommen werden. Dementsprechend kann der hohe Automatisierungsgrad in diesem Ansatz nicht vollständig ausgeschöpft werden. Daher fehlt diesem Ansatz eine Möglichkeit, die es den Ingenieuren ermöglicht die Änderungen nachzuvollziehen und bei Bedarf weitere Anpassungen an den Modellen vornehmen zu können.

Industrielle Einschränkung: Eine weitere Herausforderung bei der Anwendung des Überschreibungsansatzes ist die industrielle Einschränkung. Normalerweise erlauben die Unternehmen nicht, dass die während des Engineering-Prozesses erstellten Modelle überschrieben werden und dadurch die organisatorischen Relationen sowie die Relationen zu den verantwortlichen Ingenieuren für die einzelnen Modelle verloren gehen.

Ein weiterer Ansatz zur Anpassung der Modelle des Digitalen Zwillings besteht in der Versionierung der modifizierten Ankerpunkte im Gesamtsystemmodell.

Versionierung der Modelle

Wie in Abschnitt 2.2 beschrieben, sind während des Engineering-Prozesses verschiedene Fachexperten aus den verschiedenen Domänen an der Erstellung eines Basismodells für ein Asset beteiligt. Die Konstrukteure erstellen im Engineering unter Berücksichtigung verschiedener Aspekte wie Qualitäts- und Anwendungsparameter viele Modelle zu einem Asset. Diese Modelle werden alle unter einer eindeutigen ID des Assets als Wurzelknoten in einem Datenbackbone mit unterschiedlicher Versionierung abgelegt. Wenn beispielsweise eine Funktionsgruppe wie ein Bohrmodul, das aus vielen Sensoren und Aktoren besteht, als ein Asset betrachtet wird, gibt es viele verschiedene Kombinationen, mit denen das 3D-CAD-Modell des Bohrers zusammengesetzt werden kann. Jede dieser Kombinationen kann für eine bestimmte Anwendung bevorzugt werden. Sie sind jeweils eine Version des 3D-CAD-Modells des Bohrmoduls und werden unter der identischen eindeutigen ID als Wurzelknoten angeordnet. Diese Wurzelknoten und alle untergeordneten Versionsmodelle eines Assets werden als Basismodell für das funktionale Engineering eines automatisierten Systems verwendet. Beim funktionalen Engineering eines automatisierten Systems werden alle Ankerpunkte des Gesamtsystemmodells durch Vererbungsrelationen an den Wurzelknoten des Basismodells referenziert. Im Allgemeinen definieren die Ingenieure des automatisierten Systems eine Versionierungsregel, die angibt, welche der untergeordneten Versionen bei der Referenzierung eines Basismodells auf das Gesamtsystemmodell übernommen werden müssen. Durch die Anwendung einer festen Versionierungsregel, nach der die letzte Version aus dem Basismodell immer über eine Referenzierung aufgerufen wird, besteht die Möglichkeit, Änderungsszenarien mit einer neuen Version der Basismodelle im Gesamtsystemmodell zu synchronisieren. In diesem Ansatz der Basismodell-Versionierung, können alle Änderungsszenarien mit Ausnahme des Löschens eines Ankerpunktes aus dem Gesamtsystemmodell oder des Hinzufügens eines neuen Ankerpunktes im Gesamtsystemmodell angepasst werden.

Durch das Auslesen des Basismodells aus dem Datenbackbone und die Nutzung der detektierten Informationen aus den letzten Schritten der Ankerpunktmethode sowie durch XML-Manipulation wird die aktuelle Version synchronisiert mit dem Asset im realen System unter dem Basismodell untergeordnet. Schließlich wird das Basismodell in den Databackbone hochgeladen. Somit wird der Ankerpunkt, der sich auf das Basismodell bezieht, in seiner neuesten Version als die aktuelle Version übernommen, vgl. Abbildung 29.

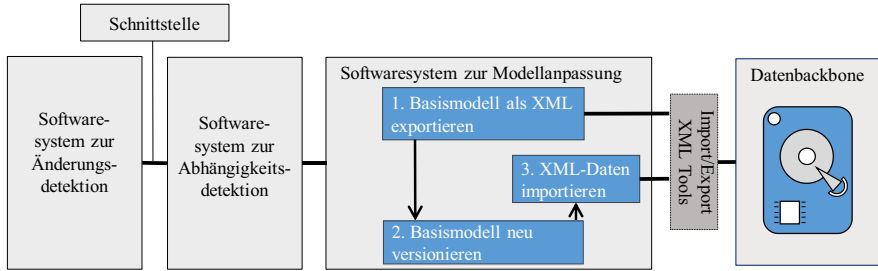


Abbildung 29: Prozessablauf des Lösungsansatzes basierend auf der Basismodell-Versionierung

Allerdings können einige in der Ankerpunktmethode erkannte Änderungsszenarien im Ansatz der Basismodell-Versionierung nicht synchronisiert werden. Beispielsweise müssen nach dem Änderungsszenario "Gelöschte mechatronische Komponente aus dem System" die entsprechenden Ankerpunkte des entfernten Assets im Gesamtsystemmodell gelöscht werden. Dies ist jedoch auf der Basis der Neuversionierung des Basismodells des Assets nicht möglich. Dazu muss das Basismodell aus dem Datenbackbone gelöscht werden; in der Folge verlieren jedoch alle anderen Ankerpunkte, die sich auf dieses Basismodell beziehen, ihre Vererbungsrelation und damit wird das Gesamtsystemmodell inkonsistent. Für die Fälle, in denen z.B. ein neuer Ankerpunkt zum Gesamtsystemmodell hinzugefügt wird, muss das Gesamtsystemmodell modifiziert werden; dieser Prozess ist nicht alleine durch eine Neuversionierung des Basismodells eines Assets möglich. Dementsprechend muss für derartige Änderungsszenarien ein anderer Prozessschritt für den Versionisierungsansatz durchgeführt werden, der dem Überschreiben der Modelle sehr ähnlich ist. Die hierfür notwendigen Prozessschritte sind in Abbildung 30 dargestellt.

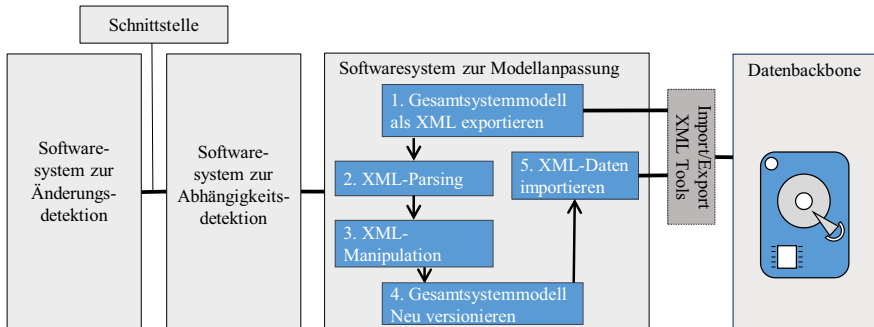


Abbildung 30: Prozessablauf des Lösungsansatzes basierend auf Versionierung des Gesamtsystemmodells

Zusammenfassend lassen sich die Herausforderungen für den Ansatz basierend auf einer Versionierung der Modelle des Digitalen Zwillings in zwei Schwerpunkte aufteilen.

Eine allgemeine Herausforderung für den Ansatz ist die **Rückverfolgbarkeit für Systemingenieure**; auch in diesem Ansatz muss eine zusätzliche Gesamtlösung entwickelt werden, damit die Ingenieure die Änderungen verfolgen und ggf. weitere Anpassungen an den Modellen durchführen können.

Versionierungsregel eines Basismodells: Im Versionierungsansatz liegt die hauptsächliche Herausforderung bei der Anpassung der bestehenden Regel im Gesamtsystemmodell eines automatisierten Systems mit einer hohen Anzahl an mechatronischen Komponenten. In einem komplexen automatisierten System kommt es häufig vor, dass verschiedene untergeordnete Versionen eines Basismodells in einem Gesamtsystemmodell aufgerufen werden, dies ist in Abbildung 31 dargestellt. Beispielsweise besteht die Möglichkeit, dass ein Bohrmodul als eine Funktionsgruppe in einer Anlage mehrfach mit unterschiedlichen Konfigurationen seiner Sensoren und Aktoren verwendet wird. In diesem Fall wird jeder Ingenieur während des Engineering-Prozesses mittels einer Versionierungsregel das notwendige Versionsmodell aus dem Basismodell des Bohrers im Gesamtsystemmodell aufrufen. Dieser Ansatz funktioniert nur, wenn es im Gesamtsystemmodell immer nur eine feste Versionierungsregel gibt. Diese Regel erlaubt es nicht, verschiedene Modellversionen des Basismodells eines Assets im Gesamtsystemmodell in Anspruch zu nehmen, was sehr häufig bei komplexen automatisierten Systemen vorkommt.

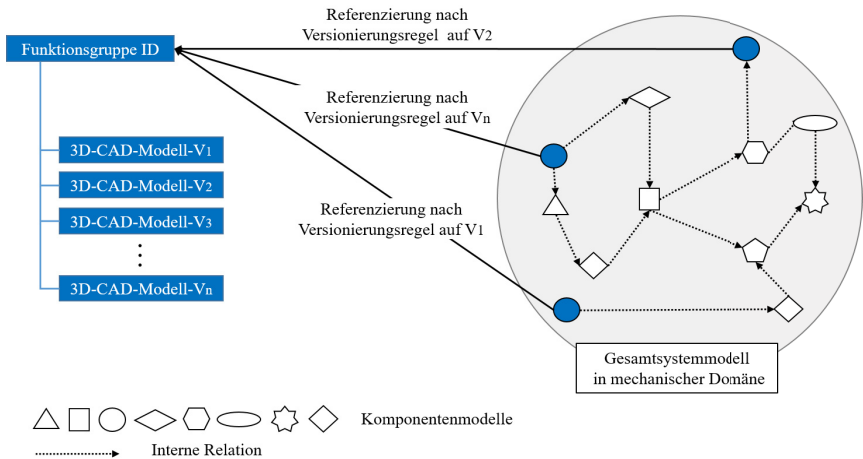


Abbildung 31: Referenzierung der Ankerpunkte des Gesamtsystemmodells auf das Basismodell nach Versionierungsregeln

Ein weiterer Ansatz basiert auf der automatisierten Durchführung des Engineering-Change-Prozesses unter Einsatz generischer Change-Request-Modelle (CRM) an den Ankerpunkten des Gesamtsystemmodells. Dieser ist dem Versionierungsansatz sehr ähnlich mit dem Unterschied,

dass der Ansatz die Rückverfolgbarkeit durch Systemingenieure ermöglicht und die Herausforderung bei der Anpassung von Versionierungsregeln überwindet.

Erstellung generischer Change-Request-Modelle an die entsprechenden Ankerpunkte des Gesamtsystemmodells

Das Konzept dieses Ansatzes basiert auf der automatisierten Durchführung der notwendigen Schritte zur Modelländerung in einem System unter Verwendung eines standardisierten Engineering-Change-Prozesses. Jarratt definiert in [153] Engineering-Change als eine Änderung an Modellen, die bereits während des Engineering-Prozesses freigegeben wurde; die Änderung kann eine beliebige Anzahl von Personen betreffen und beliebig lange dauern. Zur Durchführung von Engineering-Change wurden in der wissenschaftlichen Literatur auf verschiedenen Abstraktionsebenen viele verschiedene Prozessschritte vorgeschlagen [154]. Unter Berücksichtigung der Gemeinsamkeiten der vorgeschlagenen Ansätze zum Engineering-Change-Prozess in [154], [155], [156] kann Abbildung 32 als vereinfachte Darstellung dieses Prozesses betrachtet werden. Der Prozess beginnt mit dem Anlegen eines Change-Requests zu einem Modell im System. Nach der Genehmigung des Änderungsantrags durch eine Analyse der Abhängigkeiten und Anforderungen der Änderung werden die Änderungsschritte von einem Change-Committee geplant; im nächsten Schritt werden die geplanten Änderungen von einer Gruppe verantwortlicher Ingenieure, den Change-Builder, durchgeführt. Diese Ingenieure mit unterschiedlichen Domain-Kenntnissen sind mit den Versionsregeln des jeweiligen Modells vertraut. Schließlich wird das neue Modell vom Projekt-Manager in das System integriert. Basierend auf diesem Prozessablauf kann das Lösungskonzept "Erstellung generischer Change-Request-Modelle an die entsprechenden Ankerpunkte des Gesamtsystemmodells" zur automatisierten Modellanpassung des Digitalen Zwillings vorgeschlagen werden. In diesem Ansatz wird ein generisches Modell als Change-Request an die entsprechenden Ankerpunkte des Gesamtsystemmodells referenziert. Das Change-Request-Modell ist hier mehr als nur eine Abfrage, es ist ein gekapseltes Modell von domänenübergreifenden wiederverwendbaren Modellen und Informationen über Anpassungsaktivitäten für die Change-Builder, denen es bei

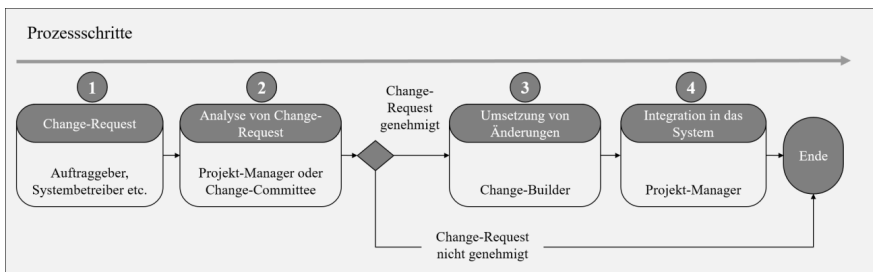


Abbildung 32: Vereinfachte Darstellung des Prozessverlaufs des Engineering-Change-Managements

der Synchronisierung des Digitalen Zwillings assistiert. Das Change-Request-Model kapselt die benötigten wiederverwendbaren Modelle in verschiedenen Formaten für jede Änderung. Dadurch kann der verantwortliche Ingenieur das wiederverwendbare Modell aus der Domäne Mechanik, Elektrik und Software direkt aus dem CRM in seine spezifischen Tools einlesen und an das Gesamtsystemmodell anfügen. Darüber hinaus stellt das Change-Request-Model alle notwendigen Informationen über die Abhängigkeiten zur Verfügung, die jeweils bei einer Änderung angepasst werden müssen, sodass auch Ingenieure ohne domänenübergreifendes Fachwissen die Änderungen anpassen können. Eine automatisierte Generierung von einem CRM durch ein externes Softwaresystem ermöglicht eine hochautomatisierte Ausführung des Prozessablaufs des Engineering-Change-Prozesses bis zur Systemintegration von Änderungen als letzten Schritt.

Dieser Ansatz überwindet die drei vorhergenannten Herausforderungen zur Anpassung der Modelle des Digitalen Zwillings, wie die Rückverfolgbarkeit von Änderungen für Systemingenieure, die Beibehaltung der Modellversionierungsregeln und industrielle Einschränkungen bei der Verarbeitung des Gesamtsystemmodells eines automatisierten Systems. Darüber hinaus bietet es einen hohen Automatisierungsgrad bei der Umsetzung eines effizienten Engineering-Change-Prozesses im Vergleich zum manuellen Modellanpassungsprozess. Zudem ist bei diesem Ansatz kein domänenübergreifendes technisches Wissen der Systemingenieure für die Umsetzung von Änderungen in den Modellen des Digitalen Zwillings erforderlich. Daher wird dieser Ansatz für die automatisierte Modellanpassung im Ankerpunktverfahren verwendet.

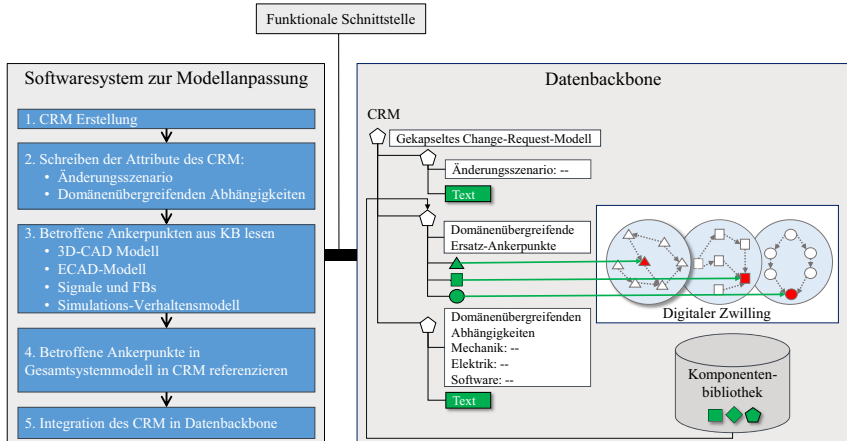
Der folgende Abschnitt beschreibt die Prozessschritte der Modellanpassung anhand generischen Change-Request-Modellen an den Ankerpunkten des Gesamtsystemmodells in Einzelschritten.

4.2.4.2 Modellanpassung basierend auf generischen Change-Request-Modellen an Ankerpunkten des Gesamtsystemmodells

Bei diesem Ansatz müssen zwei wesentliche Punkte bei der automatisierten Erstellung von Change-Request-Modellen an die Ankerpunkte des Gesamtsystemmodells beachtet werden. Erstens muss die Semantik des CRM für die unterschiedlichen Tools, mit denen die Modelle des Digitalen Zwillings erstellt wurden, auslesbar sein. Zudem können die für die Modellanpassung verantwortlichen Ingenieure, Change-Buildern, die Inhalte von CRM direkt in ihren spezifischen Tools einlesen und einsetzen. Dafür muss das CRM mit semantischen Technologien zur Integration der domänenübergreifenden Modelle des Digitalen Zwillings erstellt werden. Zweitens sind zur Gestaltung vom CRM in den integrierten Modellen des Digitalen Zwillings keine detaillierten Kenntnisse über die Speicherstruktur innerhalb des Datenbackbones erforderlich. Speicherstruktur bedeutet wie die Daten zusammen abgelegt, verwaltet und verknüpft werden. Dadurch kann die Entwicklung einer komplexen, spezifischen Schnittstelle zur Kommunikation zwischen zwei Softwaresystemen vermieden werden. Außerdem ermöglicht es eine Systemunabhängigkeit für Softwaresysteme zur Modellanpassung. Um die oben genannten

Punkte zu erfüllen, besteht eine Lösung darin, eine funktionale Schnittstelle zwischen dem Softwaresystem zur Modellanpassung und dem Datenbackbone aufzusetzen. In funktionalen Schnittstellen werden die Funktionen als Kommunikationskanäle zwischen zwei Softwaresystemen genutzt. Ein Softwaresystem ruft eine Funktion in einer anderen Software über eine Abfrage auf und erhält eine Antwort oder Aktion zurück. Es gibt viele verschiedene Ansätze zur Umsetzung dieser funktionalen Schnittstellen, die in Abschnitt 5.6 ausführlich beschrieben werden. Über die funktionale Schnittstelle kann das Softwaresystem für die Modellanpassung sein CRM ohne Kenntnis über die Speicherstruktur innerhalb des Datenbackbone erstellen, indem es einfach Funktionen durch vorhandene Bibliotheken aus dem Datenbackbone abruft. Da das CRM durch die gleichen Funktionen auf dem Datenbackbone erstellt wird, mit denen auch die Engineering-Tools ihre Modelle auf dem Datenbackbone erstellt haben, werden die Engineering-Modelle und das CRM die gleiche Semantik haben. Somit wird das CRM für die Engineering-Tools lesbar sein. Abbildung 33 zeigt den gesamten Workflow der Erstellung generischer Change-Request Modelle in fünf aufeinanderfolgenden Schritten.

Im ersten Schritt wird ein CRM erstellt, indem das Softwaresystem zur Modellanpassung eine Anfrage zur Modellerzeugungsfunktion des Datenbackbones sendet. Das Datenbackbone erzeugt automatisiert ein leeres generisches CRM und stellt dieses dem Softwaresystem zur Modellanpassung in Form einer Antwort zur Verfügung. Im zweiten Schritt nimmt das Softwaresystem zur Modellanpassung die Attribute des CRMs auf und schreibt Informationen aus den vorhergehenden Schritten der Ankerpunktumkehrung über die genau aufgetretenen Änderungsszenarien und damit verbundene domänenübergreifende Abhängigkeiten dieser



CRM: Change-Request-Modell KB: Komponentenbibliothek

Abbildung 33: Prozessablauf der Modellanpassung in der Ankerpunktumkehrung basierend auf der Erstellung generischer Change-Request-Modelle

Änderung in den Domänen Mechanik, Elektrik und Software ins CRM. Dies ermöglicht es, den Change-Buildern, diese Informationen direkt in ihren Tools zu verwenden.

Wenn dem Gesamtsystemmodell ein neuer Ankerpunkt hinzugefügt werden soll, werden im dritten Schritt die gesamten wiederverwendbaren Modelle wie 3D-CAD, elektrische Schaltpläne, Signal- und Funktionsblöcke, logisches Simulationsmodell usw. aus der bestehenden Komponentenbibliothek im Datenbackbone über die Funktionsschnittstelle aufgerufen. Im CRM wird direkt auf die wiederverwendbaren Modelle referenziert. Auf diese Weise wird bei der Umsetzung des Change-Management-Prozesses eine automatisierte Assistenz für die Change-Builder gewährleistet. Im vierten Schritt werden die Attribute des CRM an die betroffenen Ankerpunkte des Gesamtsystemmodells hinsichtlich ihrer geräte-, orts- und funktionsorientierte Modularisierungsposition angepasst. Im letzten Schritt wird das CRM über die Funktionsschnittstelle ins Datenbackbone integriert.

In diesem Kapitel wurde die Ankerpunktmethode als Lösungskonzept zur automatisierten domänenübergreifenden Änderungs- und Abhängigkeitsdetektion in einem automatisierten System ab dem Zeitpunkt der Inbetriebnahme vorgestellt. Diese Vorgehensweise basiert auf einer regelbasierten Analyse der aktuellen Steuerungssoftware des Systems und der Steuerungssoftware des Systems zum Referenzzeitpunkt. Darüber hinaus wurde in der Ankerpunktmethode ein Modellanpassungsansatz vorgeschlagen, der auf der automatisierten Generierung von Change-Request-Modellen an Ankerpunkten des Gesamtsystemmodells basiert.

Im nächsten Kapitel erfolgt die Veranschaulichung der Realisierung der drei Phasen der Ankerpunktmethode unter Zuhilfenahme eines Assistenzsystems. In diesem Zusammenhang werden die Implementierung und die Softwarearchitektur des Assistenzsystems ausführlich beschrieben.

5 Realisierung der Ankerpunktmethode mithilfe eines Assistenzsystems

Zur Realisierung der Ankerpunktmethode wurde in dieser Arbeit ein Assistenzsystem entwickelt. Das Assistenzsystem beinhaltet drei Softwaresysteme zur Umsetzung der Änderungsdetektion-, Abhängigkeitsdetektion- und der Modellanpassungsphase. Im Folgenden werden im ersten Unterkapitel die Architektur und die Komponenten des Assistenzsystems ausführlich beschrieben. Dabei wird ein Überblick über die gesamten Softwarekomponenten des Assistenzsystems zur Synchronisierung des Digitalen Zwillings gegeben. Das zweite Unterkapitel befasst sich mit der Vorstellung der grafischen Benutzeroberfläche des Assistenzsystems. Das dritte Unterkapitel stellt einen Ansatz zur Kopplung des Assistenzsystems mit einem externen Softwaresystem zur automatisierten Generierung von XML-Dateien aus den Elementen der SPS-Steuerungssoftware vor. Dadurch wird der erste Schritt des Prozessablaufs der Ankerpunktmethode umgesetzt. Zur Realisierung des zweiten Schrittes der Ankerpunktmethode wird im vierten Unterkapitel ein SPS-Steuerungssoftware-Metamodell zur Formalisierung der SPS-Steuerungssoftware und Erzeugung eines Instanz-Objektmodells vorgestellt. Darüber hinaus wird ein Ansatz zur automatisierten Generierung dieses Metamodells erklärt. Im darauffolgenden Unterkapitel werden die Komponenten des Assistenzsystems zur Selektion von Ankerpunkten aus der formalisierten SPS-Steuerungssoftware sowie die implementierten Entscheidungsbäume für die regelbasierte Analyse dieser Ankerpunkte erläutert, die die Schritte drei und vier der Ankerpunktmethode umsetzen. Das letzte Unterkapitel befasst sich mit der Softwarekomponente und dem Prozessablauf zur servicebasierten Generierung von Change-Request-Modellen im Digitalen Zwilling zur Umsetzung der Schritte fünf bis sieben des Lösungsansatzes.

5.1 Gesamtüberblick über Softwarekomponenten des Assistenzsystems

Ein Assistenzsystem ist ein Expertensystem, das entwickelt wird, um Benutzer mit einer bestimmten Aufgabe bei der Bewältigung vielschichtiger Problemen zu unterstützen, indem es Informationen oder Handlungsempfehlungen zur Verfügung stellt [157], [158]. Es verfügt über die Fähigkeit zur Speicherung von Wissen, möglicherweise regelbasiert [159], das bei der Verarbeitung von Benutzereingaben verwendet wird. Die Ausgabe des Assistenzsystems wird dann aus der Wissensbasis sowie den Eingabedaten abgeleitet, die den Benutzer bei einer Zielerreichung unterstützen. Ein Assistenzsystem kann auch Schnittstellen zu übergeordneten IT-Systemen haben [159]. Im Rahmen der Umsetzung der Ankerpunktmethode zur Synchronisierung der Modelle des Digitalen Zwillings wurde in dieser Arbeit ein Assistenzsystem konzipiert. Das entwickelte Assistenzsystem ermöglicht die automatisierte Durchführung der drei Hauptschritte der Ankerpunktmethode zur Änderungserkennung, Abhängigkeitserkennung und

Modellanpassung durch Zugriff auf die SPS-Steuerungssoftware eines automatisierten Systems zu zwei verschiedenen Zeitpunkten. Das Assistenzsystem wurde als Java-Applikation implementiert und kann auf dem Betriebssystem eines Computers ausgeführt werden. Abbildung 34 gibt eine Gesamtübersicht über die Komponenten und Schnittstellen des Assistenzsystems wieder.

Eine Komponente des Assistenzsystems ist die grafische Benutzeroberfläche, GUI. Über die GUI kann der Benutzer zwei SPS-Steuerungssoftwares von verschiedenen Zeitpunkten in das Assistenzsystem laden. Außerdem wird die GUI verwendet, um die Zugangsdaten der Benutzer für die Anmeldung am Datenbackbone und die Adresse des Datenbackbones zu erhalten. So kann sich das Assistenzsystem über seine Schnittstelle am Datenbackbone anmelden und automatisiert Change-Request-Modelle an den Ankerpunkten des Digitalen Zwillings generieren. Ein "Wrapper" wurde als weitere Komponente im Assistenzsystem implementiert, um externe Software zum Erzeugen von XML-Dateien aus der SPS-Steuerungssoftware aufzurufen. Zur Erstellung der SPS-Steuerungssoftware kommt in dieser Arbeit die Software TIA-Portal zum Einsatz. TIA-Portal-Openness ist eine integrierte API der TIA Portal-Software, die es ermöglicht, XML-Daten aus den im TIA Portal-Format gespeicherten Elementen der SPS-Steuerungssoftware zu generieren. Eine XML-Datei weist eine feste Struktur auf, die maschinell lesbar ist und dem Assistenzsystem eindeutig identifizierbare Elemente liefert. Da die Schnittstelle der TIA-Openness nur in der C#-Sprache vorliegt, wird der Wrapper verwendet, um die im Assistenzsystem geschriebenen Befehle an die TIA-Openness-Schnittstelle zur Generierung der

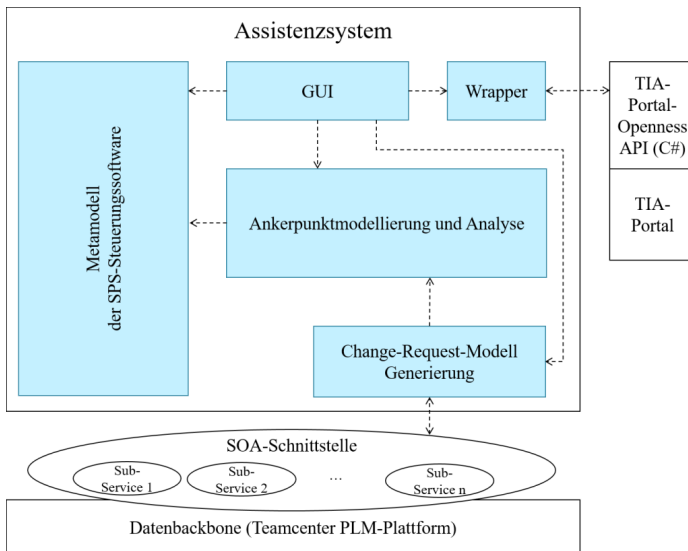


Abbildung 34: Gesamtübersicht über die Softwarekomponenten des Assistenzsystems

XML-Dateien aus der SPS-Steuerungssoftware von Java nach C# zu übersetzen. Für die Realisierung des Formalisierungskonzeptes der Ankerpunktmethode ist im Assistenzsystem die Komponente "Metamodell der SPS-Steuerungssoftware" implementiert, wodurch aus XML-Dateien der SPS-Steuerungssoftware zu zwei verschiedenen Zeitpunkten zwei standardisierte Instanz-Objektmodelle generiert werden können. Das Assistenzsystem enthält die weitere Komponente "Ankerpunktmodellierung und Analyse". Diese Komponente hat eine wesentliche Bedeutung für die Änderungserkennung durch Abstraktion der erzeugten Steuerungssoftware-Instanz-Objektmodelle sowie für die Analyse der in diesem Modell enthaltenen Ankerpunkte durch in ihr implementierte Entscheidungsbäume. Das Ankerpunktmodell basiert auf dem Abstraktionskonzept der Ankerpunktmethode, die nur die notwendigen Daten und Relationen zur regelbasierten Analyse enthält. Diese Komponente besteht aus der Java-Klasse "Compare", die alle Regeln des Entscheidungsbaums der Ankerpunktmethode aufweist. Sie dient zum Vergleich der beiden Ankerpunktmodelle der SPS-Steuerungssoftware, um die domänenübergreifenden Änderungen im realen automatisierten System abzuschließen. Das Ergebnis der erfassten Änderungen wird in die Änderungserkennungsergebnisse der Komponente "Anchor Point Modeling and Analysis" gespeichert. Hier kann einerseits die GUI auf die Ergebnisse der Änderungserkennung zugreifen und diese für die Benutzer visualisieren, andererseits werden deren Inhalte der implementierten Komponente "Change-Request-Modell-Generierung" zur Verfügung gestellt. Die Java-Klassen innerhalb dieser Komponente interagieren mit den Services der Teamcenter PLM-Plattform unter Verwendung der Teamcenter SOA-Client-Bibliothek und führen die Schritte der Ankerpunktmethode durch, um die Change-Request-Modelle zu generieren und sie an den Ankerpunkten des Gesamtsystemmodells im Datenbackbone zu referenzieren. Zur Umsetzung und Evaluierung der Ankerpunktmethode wurde in dieser Arbeit die Teamcenter-PLM-Plattform als Datenbackbone für die Integration und strukturierte Speicherung der Modelle des Digitalen Zwillings während des Engineerings ausgewählt. Wie in Abschnitt 2.3.3 ausführlich beschrieben wurde, ist die PLM-Plattform eine weit verbreitete Technologie in der Industrie zur strukturierten Modellierung domänenübergreifende Modelle des Digitalen Zwilling in einem zentralen Datenbanksystem.

5.2 Komponente „Grafische Benutzeroberfläche“

Abbildung 35 stellt die grafische Benutzeroberfläche des Assistenzsystems dar. Das Hauptfenster der GUI besteht aus zwei separaten Bereichen für die Benutzer. Auf der linken Seite werden zwei SPS-Steuerungssoftwares des Systems zu unterschiedlichen Zeitpunkten per Button "Upload" vom Benutzer angefordert. Durch Drücken des Buttons "Save Results" besteht die Möglichkeit, die Ergebnisse der Änderungserkennung als Dokument im lokalen Speicher des Systems zu archivieren. Dabei wird das Ablageverzeichnis auf dem lokalen System abgefragt. Durch Anklicken des Knopfes "Save Results" wird die Änderungserkennung im Assistenzsystem gestartet. Nach der Analyse der Änderungen und Abhängigkeiten können die erfassten

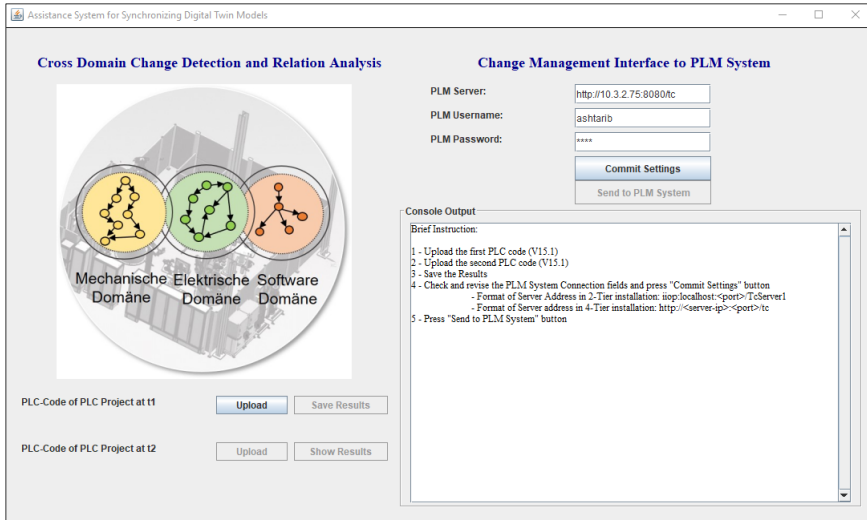


Abbildung 35: Grafische Benutzeroberfläche des Assistenzsystems

Änderungen unabhängig vom Modellanpassungsprozess innerhalb des Digitalen Zwillings durch den Button "Show Result" visualisiert werden. Durch Anklicken dieses Buttons werden die Änderungserkennungsergebnisse aus den Java-Klassen der Komponente "Anchor Point Modeling and Analysis" geladen und als tabellarisches Protokoll in der grafischen Benutzeroberfläche angezeigt. Die rechte Seite der GUI dient zur Anmeldung des Benutzers an die Benutzerverwaltung des Datenbackbones, in dem das Gesamtsystemmodell des realen Systems während des Engineerings gespeichert wurde.

Hierfür gibt es zunächst eine Eingabemaske, die korrekte Angaben zu Server, Nutzernamen und Passwort beinhalten muss. Durch Drücken des Buttons „Commit Settings“ werden die Angaben gespeichert. Ebenfalls steht ein Button "Send to PLM System" zur Verfügung, der über die servicebasierte Schnittstelle zwischen Assistenzsystem und Datenbackbone automatisiert CRMs generiert und diese an die Ankerpunkte des Gesamtsystemmodells referenziert. Durch Drücken des Buttons „Send to PLM System“ wird zunächst eine Überprüfung der in der Eingabemaske angegebenen Daten gestartet. Sind diese korrekt, wird die Übermittlung der Ergebnisse der Änderungserkennung an Teamcenter gestartet.

5.3 Komponente „Wrapper“

Im Rahmen der Erstellung eines formalisierten Modells der Steuerungssoftware und ihrer systematischen Analyse müssen in der Ankerpunktmethode die Elemente der SPS-Steuerungssoftware in einem eindeutigen Modell bereitgestellt werden. Dazu sind zunächst

verschiedene Elemente der Steuerungssoftware, wie Funktions- und Datenblöcke, Signale, Hardwarekonfiguration des Steuerungssystems, in XML-Dateien umzuwandeln. Jedoch haben die XML-Dateien der Elemente unterschiedliche Modellierungsspezifikationen. Um diese verschiedenen XML-Dateien und ihre Gemeinsamkeiten in einem Modell zusammenzuführen, ist ein generelles Metamodell für SPS-Steuerungssoftware erforderlich, das diese miteinander kombinieren und aus den Elementen der Steuerungssoftware ein Instanz-Objektmodell erzeugen kann. Ein industrieller Ansatz für den Export und Import von herstellereinspezifischer SPS-Steuerungssoftware in XML-Dateien ist die TIA-Portal-Openness. TIA-Portal-Openness ist eine öffentliche API für die Kommunikation zwischen den externen Anwenderprogrammen und TIA-Portal. Durch TIA-Portal-Openness kann die proprietäre Steuerungssoftware als XML-Dateien aus dem TIA-Portal exportiert oder importiert werden, sodass Projektdaten in externen Anwendungen verwendet werden können [160]. TIA-Portal-Openness stellt mehrere Dynamic Link-Library (DLLs) zur Verfügung, über die auf TIA-Portal zugegriffen werden kann. Diese DLLs sind jedoch auf das .NET-Framework ausgelegt und wurden in C# implementiert [160]. Da das Assistenzsystem jedoch in Java geschrieben wurde, muss für den Austausch zwischen dem Assistenzsystem und TIA-Portal-Openness in Hinblick auf die automatisierte XML-Datengenerierung eine Schnittstelle als Wrapper implementiert werden. Wrapper gelten als bekannte Werkzeuge, um Softwaresysteme mit minimalem Aufwand zu koppeln. Der Wrapper wird für die Übersetzung der Befehle von Java nach C# eingesetzt.

Hierzu wurde im Assistenzsystem die Java-Klasse „TiaOpenness“ implementiert. Die Klasse TiaOpenness im Assistenzsystem beinhaltet die Funktionen, die die Kommunikation zum TIA-Portal ermöglichen. Hierzu wird bereits im Vorfeld die API TIA-Portal-Openness in C# implementiert und die erzeugte DLL mithilfe des Wrappers „jni4net“ und dessen Proxy-Funktionen für Java zur Verfügung gestellt. Die Java-Klasse „TiaOpenness“ beinhaltet lediglich den Code zur Verwendung von „jni4net“ und den Aufruf der Proxy-Funktion „OpenAndExport“. Über den Proxy wird anschließend eine Anfrage an die DLL gestellt, um diese Funktion auszuführen. Der Export der formalisierten Dateien kann dann stattfinden. Nach Eingabe des Pfades der gewünschten SPS-Steuerungssoftware zur Analyse aus dem Steuerungssoftware-Repository in die GUI des Assistenzsystems wird dieser Pfad von der Klasse „TiaOpenness“ über jni4net an das C#-Programm weitergeleitet. TIA-Portal-Openness startet anschließend TIA-Portal im Hintergrund und öffnet die gewählte Steuerungssoftware. Schließlich folgt der Export der gewünschten XML-Dateien in einen softwareintern definierten Pfad und TIA-Portal wird geschlossen. Sobald dieser Prozess abgeschlossen ist, wird die Ordnerstruktur der oben genannten XML-Dateien aufgerufen und diese in das erstellte Metamodell der SPS-Steuerungssoftware im Assistenzsystem übertragen. Dadurch wird ein Instanz-Objektmodell der gesamten SPS-Steuerungssoftware erzeugt. Es folgt eine Abstraktion der relevanten Informationen oder Ankerpunkte aus dem Instanz-Objektmodell, die für die Änderungserkennung des Assistenzsystems erforderlich sind. Die Erstellung des generellen Metamodells der SPS-

Steuerungssoftware im Assistenzsystem und das Abstrahieren des Instanz-Objektmodells zu einem Ankerpunktmodell werden in den Unterkapiteln 5.4 und 5.5 näher beschrieben.

5.4 Komponente „Metamodell der SPS-Steuerungssoftware“

Wie im Konzept, Kapitel 4, erläutert, muss für die Analyse des Inhalts der Steuerungssoftware aus importierten XML-Dateien in das Assistenzsystem ein Instanz-Objektmodell erzeugt werden, das die Logik und den Inhalt der Software einheitlich repräsentiert. Demzufolge muss im Assistenzsystem ein Metamodell durch Java-Klassen implementiert werden, das automatisiert ein Instanz-Objektmodell aus den XML-Dateien instanziiert. Ein Metamodell ist selbst ein Modell und dient dazu, ein anderes Modell mit Hilfe einer Modellierungssprache zu beschreiben. Zu diesem Zweck wurde im Assistenzsystem ein Metamodell erstellt, das die generierten XML-Dateien aus einer in S7-Graph- oder FUP-Sprache geschriebenen SPS-Steuerungssoftware in ein Instanz-Objektmodell umwandeln kann. Dabei wurden die Sprachen S7-Graph und FUP ausgewählt, da sie weit verbreitete Sprachen für die SPS-Programmierung in automatisierten Systemen in der diskreten Fertigung sind. Allerdings ist die Vorgehensweise zum Erstellen eines Metamodells für andere Programmiersprachen im Assistenzsystem identisch und das Metamodell kann bei Bedarf um andere SPS-Programmiersprachen erweitert werden. Zur Erstellung dieses Metamodells wurde ein Metamodellierungsansatz mittels des Eclipse Modeling Framework (EMF)-Tools eingesetzt. Dieser Ansatz ermöglicht die automatisierte Metamodellgenerierung aus einem XML-Schema einer SPS-Steuerungssoftware unabhängig von der SPS-Programmiersprache und dem Projektformat.

Metamodellierungsansatz aus einem XML-Schema

Wie im Abschnitt 5.3 beschrieben, werden durch Tia-Portal und TIA-Portal-Openness als API die Elemente wie die Hardware-Konfiguration des Steuerungssystems, Signallisten, Funktions- und Datenblöcke etc. einer SPS-Steuerungssoftware in separaten und unterschiedlichen XML Datentypen exportiert.

Damit aus den XML-Dateien ein Instanz-Objektmodell automatisiert generiert wird, das die Zusammenhänge zwischen den Steuerungssoftware-Elementen in einem Schema zusammenfasst, ist ein allgemeines Metamodell notwendig. Zur Erstellung des Metamodells wird hier ein Metamodellierungsansatz zur Transformation der XML-Schema-Definition (XSD) der exportierten XML-Dateien in ein Metamodell unter Verwendung des EMF eingesetzt. Das EMF erstellt mithilfe einer UML ein Klassendiagramm, das Klassen und Attribute enthält, die die in den XML-Dateien enthaltenen Elemente widerspiegeln. Abbildung 36 zeigt den Prozessablauf des Ansatzes zur Erstellung des Metamodells mit verschiedenen Tools. Zur Erzeugung des Metamodells der SPS-Steuerungssoftware werden zunächst aus verschiedenen XML-Dateien mehrere XSD-Dateien - Schema-Dateien - erzeugt.

Eine XSD-Datei beschreibt die Daten und die Struktur in einer XML-Datei. Ein wichtiger Aspekt von XSD-Dateien ist die Verwendung von sogenannten XML-Namensräumen. Namensräume sind spezielle Schemaattribute, die ein Präfix für ein bestimmtes XML-Element definieren, um es von anderen gleichnamigen Elementen zu unterscheiden. Das jeweilige Schema kann mit Software Tools wie z.B. Liquid Studio generiert werden. Um sicherzustellen, dass das generierte Schema tatsächlich die Wertebereiche und die Datenstruktur aller exportierten XML-Dateien definiert, können die Software Tools wie OxygenML eine XML-Datei gegen ihr Schema validieren. Im nächsten Schritt werden die XSD-Dateien an das EMF übertragen, das anschließend eine hohe Anzahl an Java-Klassen und Beziehungen zwischen diesen, gemäß dem vorgegebenen Schema und den Gemeinsamkeiten, erzeugt. Das Eclipse Modeling Framework ist ein Tool für die Metamodellierung innerhalb des Eclipse Integrated Development Environments. Das erstellte Metamodell folgt dem objektorientierten Paradigma.

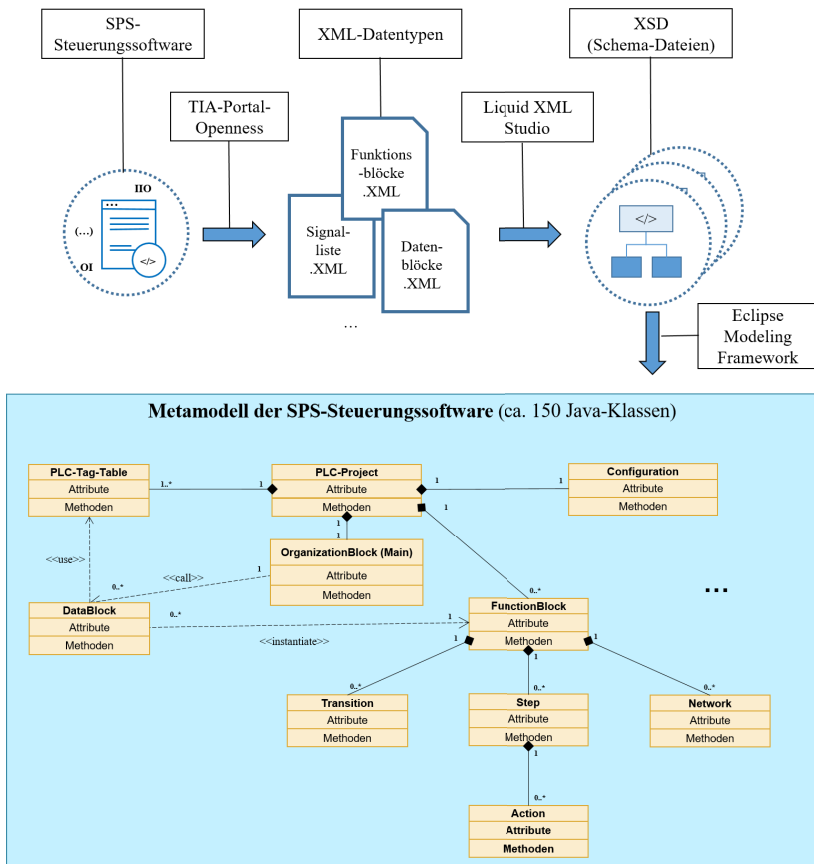


Abbildung 36: Metamodellierungsansatz zur Erstellung des Metamodells einer SPS-Steuerungssoftware

Hierbei werden Charakteristiken wie Assoziationen, Vererbung, Objekte und Attribute angewendet. Die in diesem Metamodell generierten Java-Klassen können auch für weitere Anwendungszwecke angepasst und angewendet werden, beispielsweise zur Realisierung des Abstraktionskonzeptes in der Ankerpunktmethode. Dies wird im nächsten Abschnitt näher beschrieben.

5.5 Komponente „Ankerpunktmodellierung und Analyse“

Eine weitere Komponente des Assistenzsystems ist die Komponente „Ankerpunktmodellierung und Analyse“. Auf der einen Seite bietet diese Komponente die Möglichkeit, eine Modellinstanz aus den Ankerpunkten innerhalb jeder der erzeugten SPS-Steuerungssoftware-Modellinstanzen aus dem vorherigen Schritt zu erzeugen. Dies ermöglicht einen schnellen Auswertungsprozess. Auf der anderen Seite befinden sich in dieser Komponente die implementierten Entscheidungsbäume, die für die Analyse der Ankerpunkte beider Modellinstanzen verwendet werden. Abbildung 37 zeigt ein vereinfachtes UML-Klassendiagramm der Klassen und deren Abhängigkeiten in der Komponente „Ankerpunktmodellierung und Analyse“. Die Java-Klasse "Compare" steht im Mittelpunkt der Komponente. Wenn der Upload beider SPS-Steuerungssoftwares in XML-Dateien durch TIA-Openness erfolgreich abgeschlossen ist, werden dedizierte Methoden aus der Klasse "Compare" aufgerufen. In Abbildung 37 sind die Methoden und Funktionen innerhalb der Java-Klasse "Compare" dargestellt. Die Funktion "loadComponents" innerhalb der Compare-Klasse wird durch Drücken der Button "Upload" auf der GUI aufgerufen und erhält den Dateipfad, der zu den XML-Dateien führt. Innerhalb von "loadComponents" werden die einzelnen Funktionen aufgerufen, die mit "open-" beginnen. Durch das Laden der XML-Dateien in das Metamodell der SPS-Steuerungssoftware wird aus den gesamten Elementen der Steuerungssoftware ein Instanz-Objektmodell erzeugt. Nach den "open-" Funktionen werden die "extract-" Funktionen innerhalb der Compare-Klasse aufgerufen. Somit werden die Ankerpunkte und ihre Attribute innerhalb des Instanz-Objektmodells selektiert und ein abstraktes Modell von Ankerpunkten extrahiert. Die Klassen "HardwareConfiguration", "FunctionBlock", "Datablock", "Component", "Network", "Step", "Transition", "Action" und "Component" beinhalten die Ankerpunkte der SPS-Steuerungssoftware. Die Klasse "HardwareConfiguration" beinhaltet das SPS-Modell, CPU-Modell und eine Auflistung der angeschlossenen Peripherie-Geräte. Die Klasse "FunctionBlock" beinhaltet die relevanten Informationen zu den Funktionsblöcken der SPS-Steuerungssoftware und damit zur Logik der Steuerung.

Zum einen werden allgemein Informationen, wie Name, Datenblockinstanzen und Programmiersprache (GRAPH oder FUP) des Funktionsblocks, gespeichert. Zusätzlich hierzu sind weitere Attribute der Steuerungslogik festgelegt, die je nach Programmiersprache verwendet oder leer gelassen werden.

Bei der Programmierung von Funktionsblöcken in der Sprache GRAPH, werden die Attribute Schritte und Transitionen festgelegt. Bei Funktionsblöcken in FUP, werden Attribute wie funktionsblockinterne Variablen und Konstanten sowie Netzwerke verwendet. Die Klasse "DataBlock" enthält relevante Informationen zu den Datenblöcken des Programms. Dazu gehören die Namen der Datenblöcke (nach der Namenskonvention), die instanziierten Funktionsblöcke auf den Datenblöcken sowie Informationen über die Verknüpfung der Signale der SPS-

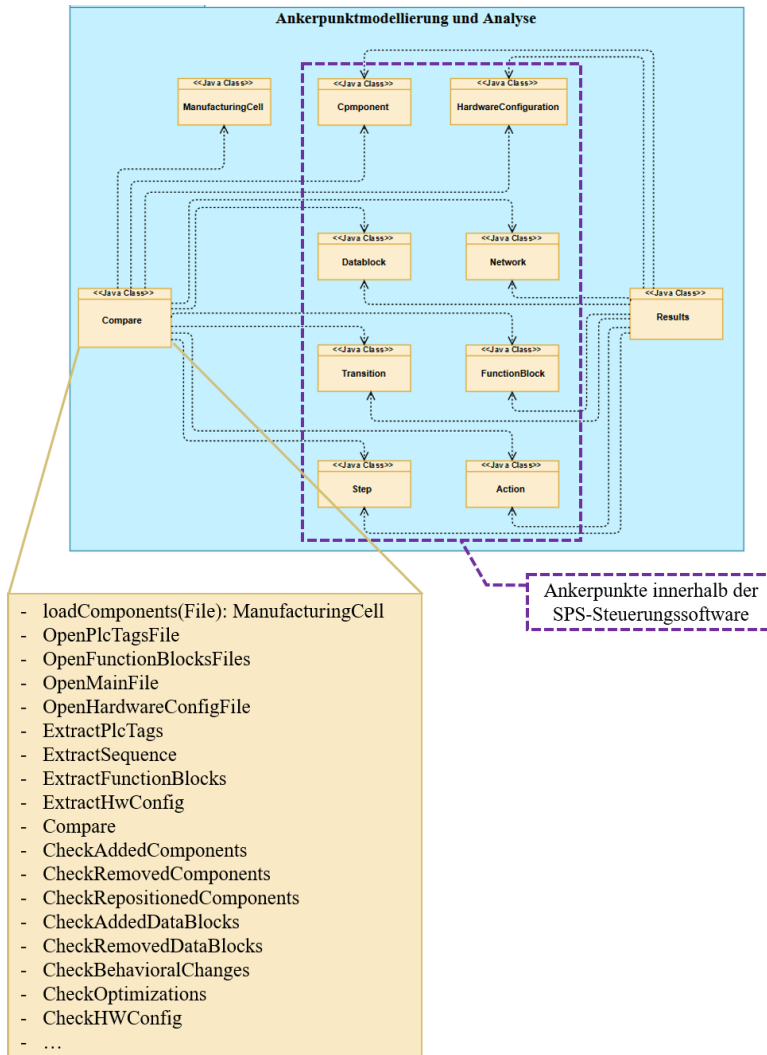


Abbildung 37: Java-Klassen der Komponente "Ankerpunktmodellierung und Analyse"

Steuerungssoftware mit den funktionsblockinternen Variablen. Die Information zu den Verknüpfungen wurde dem Aufruf des Datenblocks mit Attributen/Signalen in dem Organisationsblock „Main“ entnommen. Die Klasse "Component" stellt die verwendeten mechatronischen Komponenten und ihre zugehörigen Signale dar. Aufgrund der Anwendung der Namenskonvention wird der Name der Signale unterteilt. In diese Unterteilung werden die Attribute ID, Name der Komponente und Position der Klasse eingetragen. Ebenfalls wird aufgrund der Namenskonvention gespeichert, an welchen Hardware-Adressen die Signale der Komponente angeschlossen sind und in welchen Datenblöcken die Signale ihren Einsatz finden. Bei Funktionsblöcken in der Sprache FUP werden sogenannte "Netzwerke" innerhalb des Funktionsblocks angelegt, die auch innerhalb der formalisierten XML-Dateien zu finden sind. Diese Netzwerke erlauben eine logische Unterteilung der Funktionsblöcke in kleinere funktionale Einheiten. Daher wurde eine Klasse "Network" festgelegt, welche die Netzwerke repräsentiert. Die Logik innerhalb der Netzwerke wird durch Variablen, Konstanten und funktionale Blöcke (AND, OR, RS etc.) dargestellt, die wiederum durch sogenannte "Wires" miteinander verknüpft sind. Neben einer Auflistung der zuvor genannten Attribute, werden in der Klasse "Network" Matrizen definiert, welche die verschiedenen Arten von Verbindungen abbilden. Es wird detailliert gespeichert, welches "Wire" welche Komponenten miteinander verbindet, um die Steuerungscode-Logik darstellen zu können. Darüber hinaus wird gespeichert, welche Datenblöcke auf diese Logik innerhalb der Funktionsbausteine zugreifen. Die drei Klassen "Step", "Transition", "Action" werden ausschließlich bei Funktionsblöcken in der Sprache GRAPH verwendet. Die Klasse "Step" stellt einen Schritt innerhalb des Funktionsblocks dar. Gespeichert werden lediglich der Name des Schritts und der Name des Funktionsblocks, in dem dieser verwendet wird. Die Klasse "Transition" beinhaltet die Transitionen eines Funktionsblocks. Außerdem hier werden der Name der Transition und der Name des zugehörigen Funktionsblocks gespeichert. Die letzte Klasse, die zur Darstellung eines Funktionsblocks in GRAPH benötigt wird ist "Action". Action ist der Begriff für die Steuerung der Signale innerhalb eines Schritts. Dementsprechend wird der Name des Schritts, der Qualifier, der Token und der Status gespeichert. Zusätzlich wird auch der Name des Funktionsblocks referenziert.

Sobald die oben erwähnte Klasse aus beiden SPS-Steuerungssoftware befüllt wurden, wird die Funktion "compare" durch Drücken des Buttons "Save Results" auf der GUI aufgerufen und startet den Änderungsdetektionsprozess. Hierbei werden die Ankerpunkte von den beiden SPS-Steuerungssoftwares miteinander verglichen und analysiert. Der Änderungsdetektionsprozess wird unter Zuhilfenahme der implementierten Entscheidungsbäume innerhalb der "Check" Funktionen ausgeführt. Hierbei werden die Regeln innerhalb der Regeltabelle aus Abschnitt 4.2.3 durch if-else-Anweisungen in den "Check" Funktionen umgesetzt. Dabei realisiert jede "Check"-Funktion einen Entscheidungsbaum für die Detektion ein bestimmtes Änderungsszenario. Schließlich wird nach der Ausführung der "check"-Funktionen jedes erkannte Änderungsszenario in die Java-Klasse "Results" geschrieben und auf entsprechende geänderte Ankerpunkte dieses Änderungsszenarios referenziert. Dafür werden innerhalb der Klassen „HardwareConfiguration“,

„FunctionBlock“, „Datablock“, „Component“, „Network“, „Step“, „Transition“, „Action“, oder „Component“ Variablen angelegt, die für eine Verwendung der Klasse als Instanz zur Speicherung der Ergebnisse nach der Änderungserkennung verwendet werden. Durch Drücken des Buttons "Show Results" auf der GUI wird diese Klasse aufgerufen und ihr Inhalt dem Benutzer angezeigt.

Im nächsten Schritt muss der Inhalt der Result-Klasse mit wiederverwendbaren Modellen des modifizierten Ankerpunktes in einem Change-Request-Model gekapselt und über eine funktionale Schnittstelle auf dem Datenbackbone übertragen werden. Der Ablauf dieses Prozesses erfolgt in der Komponente "Change-Request-Model-Generierung", die im nächsten Unterkapitel im Detail vorgestellt wird.

5.6 Komponente „Change-Request-Modell-Generierung“

Im Zusammenhang mit dem Einsatz einer funktionalen Schnittstelle auf einem Datenbackbone, in dem das Gesamtsystemmodell des Digitalen Zwillings enthalten ist, wird im Assistenzsystem eine SOA-basierte Schnittstelle auf der Teamcenter PLM-Plattform verwendet. Wie im Abschnitt 2.3.3 im Zusammenhang mit verschiedenen Ansätzen zur Durchgängigkeit von Modellen und Toolverknüpfungen ausführlich erläutert, ist der Einsatz einer PLM-Plattform als einzelne IT-Systeme ein verbreiteter Ansatz zur strukturierten Modellierung und Integration der domänenübergreifenden Modelle innerhalb des Digitalen Zwillings eines automatisierten Systems. Die Auswahl der Teamcenter-PLM-Plattform als Datenbackbone zur Realisierung des Konzeptes dieser Arbeit erfolgt nach den folgenden Kriterien:

Erstens, die Teamcenter PLM-Plattform bietet die Möglichkeit, domänenübergreifende Tools zu verknüpfen, die zur Realisierung des Digitalen Zwillings eines automatisierten Systems erforderlich sind. Tools wie Line Designer, Automation Designer, Mechatronics Concept Designer, TIA-Portal und Process Simulate etc. sind bereits auf dem Teamcenter integriert. Durch diese Tools können alle Modelle des Digitalen Zwillings in den Domänen Mechanik, Elektrik und Software strukturiert auf Teamcenter erstellt und verlinkt werden.

Zweitens kann über die Teamcenter PLM-Plattform die Realisierung einer Komponentenbibliothek ermöglicht werden, die als Voraussetzung der Ankerpunktmethodik definiert wurde.

Das Datenbanksystem von Teamcenter kann verwendet werden, um die domänenübergreifenden wiederverwendbaren Modelle eines Assets unter einem eindeutigen Namen zu speichern. CAD-Modell, Schaltplanmodell, Funktionsmodell, Simulationsmodell usw. eines Assets lassen sich unter einem eindeutigen Namen in Teamcenter speichern und jederzeit vom Assistenzsystem anfragen.

Drittens bietet das Teamcenter eine SOA-basierte Schnittstelle zur Integration von externen Tools und deren Softwaresystemen. Hierzu beinhaltet Teamcenter ausführlich dokumentierte SOA-Bibliotheken in Java, die eine direkte Implementierung einer funktionalen Schnittstelle zwischen dem Assistenzsystem und dem Teamcenter zur automatisierten Generierung von Change-Request-Modellen und den Ankerpunkten des Digitalen Zwillings ermöglicht.

Gemäß [15] ist SOA *„ein Paradigma für die Strukturierung und Nutzung verteilter Funktionalität, die von unterschiedlichen Besitzern verantwortet wird“*. Durch die Anwendung von SOA kann die Generierung der Change-Request-Modelle ohne Bedarf an ausführlichen Kenntnissen über die Softwarearchitektur des Datenbackbones und nur durch Interaktion mit den Services von Teamcenter aus durchgeführt werden. Durch die SOA stellt der Teamcenter-Server seine Funktionen als Services zur Verfügung. Ein Service ist ein Mechanismus, um den Zugriff auf eine oder mehrere Funktionen in einem Softwaresystem zu ermöglichen, sofern der Zugriff über eine vorgeschriebene Schnittstelle bereitgestellt wird und in Übereinstimmung mit den in der Servicebeschreibung angegebenen Bedingungen und Richtlinien erfolgt [161] [15]. Das heißt, dass die einzelnen internen Methoden bzw. Klassen so geschrieben werden, dass sie mit Hilfe von Bibliotheken von außen auch erreichbar sind. Dafür reicht eine Verbindung mit dem Teamcenter-Server aus, um die Methoden oder Funktionen zu aktivieren bzw. abzurufen. Im Falle eines fehlenden Service für eine spezielle Anwendung, können mit Hilfe von „Business Modeller IDE“ (BMIDE) neue Funktionen in Teamcenter sehr schnell implementiert, geprüft, als Bibliotheken verpackt und bei Clients (Drittanwendungen) installiert werden. Das Teamcenter stellt die Bibliotheken in Java zur Verfügung. Dadurch lassen sie sich innerhalb des Assistenzsystems verwenden.

Eine weitere Komponente des Assistenzsystems ist die Komponente „Change-Request-Model-Generierung“. Abbildung 38 veranschaulicht die implementierten Java-Klassen und verwendeten SOA-Bibliotheken innerhalb der Komponente "Change-Request-Model-Generierung" des Assistenzsystems sowie den Ablauf mit Abfragen und Antworten zwischen den Teamcenter-Diensten und dem Assistenzsystem zur Generierung eines CRMs. Im ersten Schritt übermittelt das Assistenzsystem den Benutzernamen und das Passwort des Nutzers über die Komponente "Session starten" an den "Session Service" der PLM-Plattform und stellt eine Verbindung her. Die Komponente "Generische Model Generierung" übergibt dann eine Anfrage an Teamcenter zur Erstellung der erforderlichen Anzahl von Instanz-Objektmodellen, die als CRMs eingesetzt werden. Die Anzahl an CRMs ergibt sich aus der Gesamtzahl der detektierten Änderungen. Die erstellten Instanz-Objektmodelle sind generisch angelegt wie die Modelle des Digitalen Zwillings auf der PLM-Plattform. Denn sowohl die Engineering-Tools als auch das Assistenzsystem erstellen ihre Modelle in Teamcenter mit den gleichen funktionalen Schnittstellen und Strukturierungstechnologien. Dadurch sind die CRMs mit den Engineering-Tools lesbar.

Ausgehend von den erkannten Änderungen und domänenübergreifenden Abhängigkeiten werden im nächsten Schritt die Attribute der CRMs im Teamcenter über die Komponente "Attribute schreiben" geschrieben. Die Komponente "Attribute schreiben" bezieht die Informationen aus der Java-Klasse "Results" innerhalb der Komponente "Ankerpunktmodellierung und Analyse" im Assistenzsystem. Dabei werden in jedem CRM die Informationen wie das genaue Änderungsszenario, die orts-, geräte- und funktionsorientierte Modularisierungsposition der modifizierten Ankerpunkte im Gesamtsystemmodell sowie Informationen über die Anpassungsbedarfe an die elektrische Verkabelung, die Softwarelogik und die mechanische Kinematik geschrieben. In der Folge werden die Liste der erstellten CRMs und Item-IDs ihrer Instanz-Objektmodelle sowie die Liste der wiederverwendbaren Modelle der geänderten Assets und Item-IDs ihrer Instanz-Objektmodelle durch die Komponente "Liste der CRMs laden" und

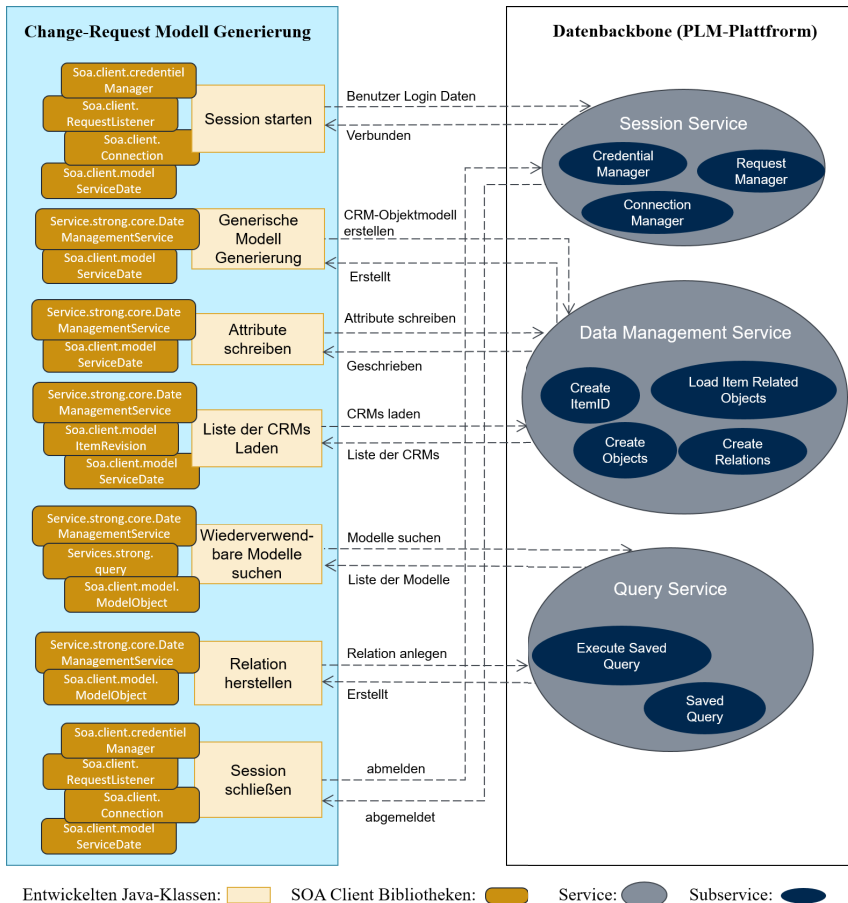


Abbildung 38: Java-Klassen der Komponente "Change-Request-Modell-Generierung"

"Wiederverwendbare Modelle suchen" aus Teamcenter aufgerufen und erstellt. Die Komponente "Relation herstellen" stellt eine Referenz zwischen der Items-ID des CRM und den entsprechenden Item-IDs von den wiederverwendbaren Modellen des geänderten Assets unter Verwendung der vorhandenen Listen her. Damit weiß der Systemingenieur, der Änderung am Gesamtsystemmodell anpassen möchte, welche wiederverwendbaren Modelle anstelle von vorhandenen Ankerpunkten ersetzt werden müssen. Schließlich, nachdem alle CRMs angelegt sind, meldet sich das Assistenzsystem von der Teamcenter PLM-Plattform über die Komponente "Session schließen" ab.

Abbildung 39 zeigt ein erstelltes CRM innerhalb von Teamcenter. Dieses beinhaltet eine Beschreibung des vorkommenden Änderungsszenarios. In diesem Fall wurde "Eine Neue Funktionsgruppe "Bohrer" in das automatisierte System hinzugefügt. Dazu gehört auch die orts-, geräte- und funktionsorientierte Modularisierungsposition des Ankerpunktes innerhalb der Struktur des Gesamtsystemmodells. Darüber hinaus enthält das CRM Informationen über die Funktionalität der Komponente im System, die elektrische Verkabelung und die modifizierten Elemente der SPS-Steuerungssoftware. Des Weiteren werden die wiederverwendbaren Modelle der neuen Funktionsgruppe "Bohrer", wie 3D-CAD-Modell, elektrisches Schaltplanmodell, Funktionsmodul und Signale sowie das Verhaltensmodell auf das CRM übertragen. Dem Systemingenieur wird es so ermöglicht, die wiederverwendbaren Modelle in den entsprechenden Engineering-Tools aufzurufen, ihre Attribute anhand der Notizen im CRM anzupassen und sie an der richtigen Strukturposition dem Gesamtsystemmodell hinzuzufügen.

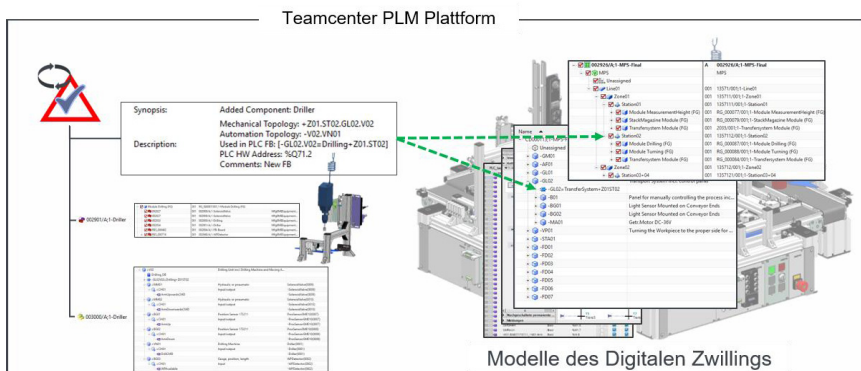


Abbildung 39: Change-Request-Modell innerhalb von Teamcenter-PLM-Plattform

Für die Evaluierung der Ankerpunktmethode wurden in dieser Arbeit zusätzlich zum Assistenzsystem zwei Evaluierungsszenarien entworfen und realisiert, wobei ein Szenario auf einem modularen, automatisierten System in einer Forschungsumgebung basiert und ein weiteres ein Beispiel für eine Industrieanlage darstellt. Im nächsten Kapitel wird die Realisierung dieser Anwendungsszenarien ausführlich beschrieben und die Evaluierungsergebnisse diskutiert.

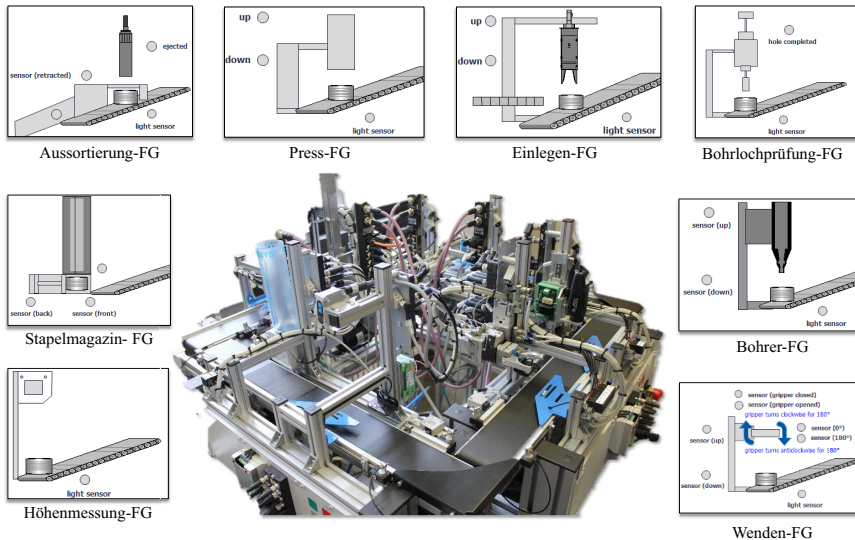
6 Evaluierung

Um die Erfüllung der definierten Forschungsanforderungen sowie die quantifizierbare Zielsetzung dieser Arbeit - Erhöhung der Verfügbarkeit des automatisierten Systems bei der Rekonfiguration durch den Einsatz des Digitalen Zwillings - mit Hilfe der Ankerpunktmethodik evaluieren zu können, sind zwei automatisierte Systeme sowohl physisch als auch digital zu entwerfen und zu implementieren. Eines dieser automatisierten Systeme ist ein modulares Produktionssystem, das am Institut für Automatisierungstechnik und Softwaresysteme implementiert ist. Das modulare Produktionssystem stellt ein realistisches Beispiel für ein automatisiertes System in der diskreten Fertigung mit modularen Stationen dar. Im ersten Unterkapitel erfolgt Beschreibung der Entwicklung des modularen Produktionssystems im Detail aus der Perspektive der realen Hardwarestruktur und des Digitalen Zwillings. Danach wird das darauf aufbauende Evaluationsszenario vorgestellt. Ausgehend vom diesen Evaluationsszenario wird das Assistenzsystem im Rahmen der Synchronisierung der Modelle des Digitalen Zwillings über den gesamten Lebenszyklus des automatisierten Systems geprüft. Um die Adaptierbarkeit des Konzeptes und Assistenzsystems in der Industrie demonstrieren zu können, wurde ein zweites System, ein *intelligentes Lager* in einer flexiblen Fertigungsanlage auf dem Forschungscampus ARENA2036, Active Research Environment for the Next Generation of Automobile, eingerichtet und für die Evaluierung herangezogen. Ausgehend von diesem Anwendungsszenario musste das Assistenzsystem im Rahmen der Synchronisation der Modelle des Digitalen Zwillings des automatisierten Systems ab der Inbetriebnahme geprüft werden.

Das zweite Unterkapitel befasst sich mit der Beschreibung des intelligenten Lagers und dem darauf aufbauenden Evaluationsszenario sowie der Auswertung der Evaluationsergebnisse. Schließlich wird im letzten Unterkapitel die vorgestellte Ankerpunktmethodik und das daraus realisierte Assistenzsystem in Bezug auf die Erfüllung der Anforderungen und die Ausgangssituation dieser Forschung qualitativ und quantitativ evaluiert.

6.1 Szenario 1: Modulares Produktionssystem

Das Modulare Produktionssystem, MPS, ist ein automatisiertes System in der diskreten Fertigung, das aus zwölf Hauptfunktionsgruppen (Modulen) besteht, die zusammen ein Produkt fertigen. Diese Funktionsgruppen werden insgesamt von einer zentralen SPS gesteuert. In diesem System wird ein Kunststoffwerkstück (als Produkt) in verschiedenen Stationen des Systems verarbeitet. Abbildung 40 veranschaulicht das MPS und seine Funktionsgruppen: Stapelmagazin-, Höhemessung-, Wenden-, Bohrer-, Bohrlochprüfung-, Einlegen-, Press- und Aussortierung-Funktionsgruppe. Zusätzlich zu diesen Funktionsgruppen besteht das System aus vier Förderbandfunktionsgruppen bestehend aus Motor, Gurt und mehreren Lichtsensoren, die das Werkstück zu den Arbeitsstationen transportieren. Das Gesamtsystem umfasst 59 Sensoren und 31 Aktoren. Der Bearbeitungsprozess fängt im MPS mit der Übergabe des Werkstücks durch das



FG: Funktionsgruppe

Abbildung 40: Modulares Produktionssystem und dessen Assets

Stapelmagazin auf dem Förderband an. Die Funktionsgruppe „Stapelmagazin“ hat die Aufgabe, die Werkstücke zu lagern und sie zum Transferband auszuschieben. Befindet sich das Werkstück auf das Förderband, wird es von den Sensoren erkannt und das Förderband transportiert es zur nächsten Funktionsgruppe "Höhenmessung". In dieser Funktionsgruppe wird mithilfe der Sensoren überprüft, ob das Werkstück die richtige Höhe hat. In den nächsten Bearbeitungsstationen wird zunächst das Werkstück von der Funktionsgruppe "Wenden" angehoben und von ihren Greifern gewendet, dann wird durch die Funktionsgruppe "Bohren" auf der Oberseite des Werkstücks ein Loch gebohrt. Nach dem Bohren des Lochs wird das Werkstück in der nächsten Station durch die Funktionsgruppe "Bohrlochprüfung" überprüft. Wenn das Werkstück die richtige Position erreicht hat und die Lichtsensoren der Funktionsgruppe auslöst, fährt der Prüfstift der Funktionsgruppe nach unten. Wenn das Werkstück richtig gebohrt ist, kann der Prüfstift die untere Endposition erreichen. In der nächsten Station wird durch die Funktionsgruppe "Einlegen" eine Mutter in das Bohrloch eingesteckt. Die Funktionsgruppe "Pressen" ist zuständig für das Einpressen der Mutter in das Loch, falls sie nicht vollständig in das Loch eingesetzt ist. Den letzten Arbeitsschritt führt die Funktionsgruppe "Aussortierung" durch. Sie hat die Funktion, das Werkstück bei Bedarf über ihre Sensoren auszusortieren. Wird ein gebohrtes Werkstück ohne Mutter erkannt, gilt das Werkstück als defekt, es sei denn, es wird als Endprodukt an das Ende des Transferbandes transportiert.

Die Bearbeitungsstationen, deren Sensoren und Aktoren sowie die Steuerungs- und Laststromversorgungs-Komponente des MPS sind in Tabelle 5 aufgelistet.

Tabelle 5: Bearbeitungsstationen sowie die Steuerungs- und Laststromversorgungs-Komponenten des MPS

Anzahl von mechatronischen Komponenten Bearbeitungsstationen in MPS	Sensoren	Aktoren
Funktionsgruppe Höhemessung	1 Lichtschranke/ 1 Abstandssensor	-
Funktionsgruppe Stapelmagazin	1 Lichtschranke 2 Endlagenschalter	1 Hubzylinder
Funktionsgruppe Aussortieren	2 Lichtschranken 1 Abstandssensor 1 Endlagenschalter	1 Hubzylinder
Funktionsgruppe Pressen	1 Lichtschranke 2 Endlagenschalter	1 Hubzylinder
Funktionsgruppe Einlegen	2 Lichtschranken 5 Endlagenschalter	1 Hubzylinder 1 Greifer
Funktionsgruppe Bohrlochprüfung	1 Lichtschranke 1 Endlagenschalter	1 Hubzylinder
Funktionsgruppe Bohren	1 Lichtschranke 2 Endlagenschalter	1 Hubzylinder 1 Bohrer
Funktionsgruppe Wenden	1 Lichtschranke 6 Endlagenschalter	3 Hubzylinder
Funktionsgruppe Förderband 1	2 Lichtschranken 5 Steuerknöpfe	1 Motor 4 Anzeigen
Funktionsgruppe Förderband 2	2 Lichtschranken 5 Steuerknöpfe	1 Motor 4 Anzeigen
Funktionsgruppe Förderband 3	2 Lichtschranken 5 Steuerknöpfe	1 Motor 4 Anzeigen
Funktionsgruppe Förderband 4	2 Lichtschranken 5 Steuerknöpfe	1 Motor 4 Anzeigen
Steuerungskomponente	SPS-S7 1500-CPU 1516 mit OPC-UA, HMI IO: PROFINET, PROFIBUS und ASi 2 Input-, 2 Output-Module	
Laststromversorgung	24 V/8 A Stromversorgung für die SPS mit Kommunikation 24V/10 A Stromversorgung für die - Funktionsgruppen 24V/3 A Stromversorgung für ASi	

Die SPS-Steuerungssoftware für die beschriebene Verarbeitung wurde auf dem Siemens TIA-Portal geschrieben und innerhalb des Digitalen Zwillings des Systems durch eine virtuelle Inbetriebnahme getestet. Basierend darauf wurde das System in Betrieb genommen. Diese Steuerungssoftware wurde als SPS-Steuerungssoftware zum Referenzzeitpunkt in einem Steuerungssoftware-Repository gespeichert.

6.1.1 Digitaler Zwilling des modularen Produktionssystems

Die Grundlagen für die Erstellung eines Digitalen Zwillings von einem automatisierten System in der diskreten Fertigung wurden in Unterkapitel 2.1.2 ausführlich beschrieben. Im Rahmen dieser Arbeit wurden verschiedene Tools verwendet, um die domänenübergreifenden Modelle des Digitalen Zwillings des MPS auf dem Datenbackbone zu erstellen. Tabelle 6 listet alle Tools und die damit erstellten Modelle innerhalb des Digitalen Zwillings des MPS auf.

Tabelle 6: Tools und erstellte Modelle innerhalb des digitalen Zwillings des MPS

Tools	Erstellte Modelle im Digitalen Zwilling
NX-Modeling	3D-CAD-Modell der einzelnen Assets des MPS
Line Designer	3D-CAD Modell des gesamten MPS
Automation Designer	Elektrisches Modell des MPS Signalmodell der mechatronischen Komponenten Funktionsblockmodelle der Funktionsgruppen
TIA-Portal	Steuerungs-Ablaufmodell
PLCSIM Advanced	Modell der CPU von SPS
Visual Studio	Verhaltens- und Prozessablaufs-modell des MPS
Teamcenter PLM-Plattform	Integrierte domänenübergreifende Modelle des MPS

Abbildung 41 gibt einen allgemeinen Überblick über die Tools und die Modelle, die zur Erstellung dieser Modelle des MPS verwendet wurden. Zu Beginn wird im Tool NX-Modeling für jede mechatronische Komponente des Systems ein 3D-CAD-Modell erstellt und unter einer eindeutigen ID auf dem Datenbackbone, der Teamcenter PLM-Plattform, gespeichert. Diese 3D-Modelle werden dann vom Tool Line Designer aus dem Datenbackbone instanziiert und eine Zusammenstellung der Funktionsgruppen und ihrer mechanischen Abhängigkeiten im System erstellt. Der fertige 3D-CAD-Entwurf des MPS in der mechanischen Domäne wird als Gesamtsystemmodell über eine direkte Schnittstelle zwischen Line Designer und Teamcenter auf dem Datenbackbone gespeichert. Als nächstes wird das Gesamtsystemmodell in der mechanischen Domäne aus Teamcenter durch das Tool Automation Designer ausgelesen und auf den erstellten 3D-CAD-Modellen ein elektrisches Modell referenziert. Dabei werden die Signalliste (Ein- und Ausgangssignale von Komponenten), Steuerungs-Funktionsblöcke der Funktionsgruppen, geeignete Controller und Peripheriegeräte für das System sowie das elektrische Schaltplanmodell jeder Komponente über eine direkte Schnittstelle zwischen Automation Designer, EPLAN-Tool und TIA Portal erstellt. Ein elektrisches Schaltplanmodell für das MPS wurde in dieser Arbeit nicht realisiert. Nach der Erstellung der elektrischen Modelle werden die Signal- und die Funktionsblockmodelle über eine direkte Schnittstelle vom Automation Designer zum TIA-Portal übertragen. Anschließend wird im TIA-Portal das gesamte SPS-Steuerungsprogramm für den Prozessablauf des Systems erstellt. Abschließend wird im

Rahmen einer virtuellen Inbetriebnahme das endgültige SPS-Steuerungsprogramm getestet. Hier wird die SPS-Steuerungssoftware im TIA-Portal mit einer virtuellen SPS, erstellt mit dem Tool PLCSIM-Advanced, und das Verhalten der Funktionsgruppen des MPS, erstellt in C#, im Rahmen einer Software-in-the-Loop-Simulation, SIL-Simulation, getestet. Abschließend wurde die validierte SPS-Steuerungssoftware durch virtuelle Inbetriebnahme als Software zum Referenzzeitpunkt auf dem Steuerungsrepository gespeichert. Alle mit den Tools NX-Modeling, Line Designer, Automation Designer und TIA-Portal erstellten Modelle werden als Digitaler Zwilling zum Referenzzeitpunkt unter Verwendung der 4th-Generation-Design-Technology, 4GD-Technologie, zusammengeführt und auf der Teamcenter PLM-Plattform gespeichert. Neben der Entwicklung des Digitalen Zwillings des MPS wurde während des Engineering-Prozesses eine Komponentenmodellbibliothek von wiederverwendbaren Cross-Domain-Modellen der Komponenten des MPS auf Teamcenter erstellt. Dazu wurden die wiederverwendbaren mechanischen, elektrischen und Softwaremodelle aller Sensoren und Aktoren sowie aller Funktionsgruppen des MPS unter einem identischen eindeutigen Namen in der PLM-Plattform abgelegt.

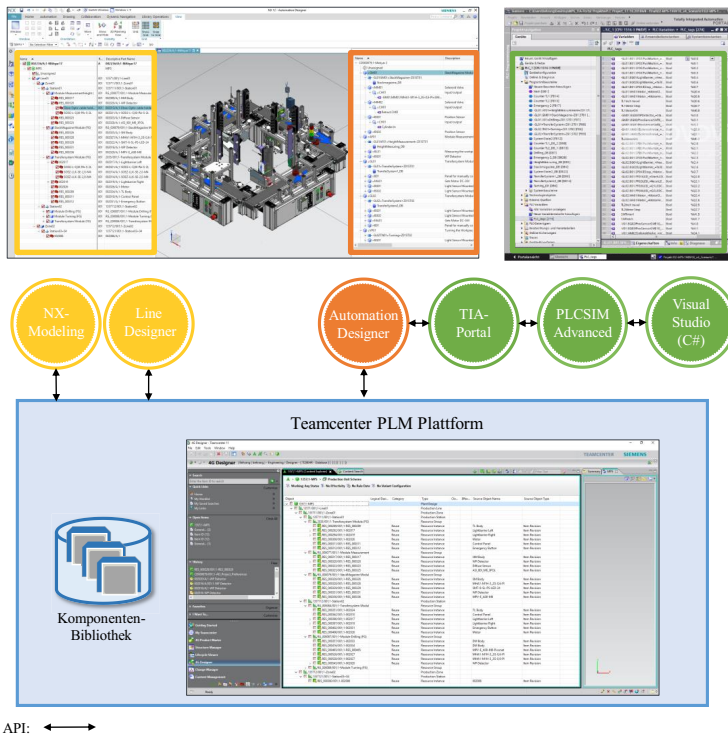


Abbildung 41: Engineering-Tools und erstellter Digitaler Zwilling vom modularen Produktionssystem

Arbeitsspeicher installiert. Die entsprechend gespeicherte Steuerungssoftware nach dem Ausführen dieser Änderungsgruppen und die Steuerungssoftware zum Referenzzeitpunkt werden im nächsten Schritt aus dem Steuerungssoftware-Repository aufgerufen und in das Assistenzsystem geladen. Wie in Tabelle 7 zu sehen ist, konnten alle Änderungen und deren domänenübergreifende Abhängigkeiten in den einzelnen Änderungsgruppen vom Assistenzsystem in kurzer Zeit korrekt erkannt und korrekte CRMs auf dem Digitalen Zwilling erstellt werden.

Tabelle 7: Durchführung der Synchronisierung des Digitalen Zwillings mit dem Assistenzsystem

Änderungsszenarien	Anzahl der Änderungen	Korrekte Detektion der Änderungen und Erstellung von CRMs	Dauer
Änderungsgruppe 1: <ul style="list-style-type: none"> Hinzufügen einer Bohrer-FG 	5	✓	<45 Sec
Änderungsgruppe 2: <ul style="list-style-type: none"> Hinzufügen einer Bohrer-FG und Lichtschrank Entfernen von zwei Lichtschranken Verhaltensänderung in Komponenten einer Förderband-FG 	9	✓	<45 Sec
Änderungsgruppe 3: <ul style="list-style-type: none"> Hinzufügen eines Motors Entfernen einer Lichtschranke Verhaltensänderungen in den Komponenten der Bohrer-, Wenden- und Förderband-FG 	14	✓	<55 Sec
Änderungsgruppe 4: <ul style="list-style-type: none"> Hinzufügen von zwei Förderbändern- und einer Stapelmagazin-FG Entfernen der Press- und Bohrlochprüfung-FG Verkabelungsänderung von zwei Lichtschranken und einer Höhemessung-FG Verhaltensänderung in Komponenten der Einlage-, Aussortierung- und Bohrer-FG 	32	✓	<77 Sec

Nach der Synchronisierung der Modelle des Digitalen Zwillings des MPS mit dem Assistenzsystem wurde im Rahmen einer Studienarbeit eine Fallstudie zur Evaluierung der Ausgangssituation dieser Forschung durchgeführt [162]. Die Fallstudie befasst sich mit der Rekonfiguration des MPS zur Erfüllung neuer Produktanforderungen unter Verwendung der synchronisierten Modelle des Digitalen Zwillings mittels Ankerpunktmethode. Der gesamte Rekonfigurationsprozess wurde in ca. einer Woche durch den Studenten durchgeführt. Dieser Zeitraum umfasst die Anpassung der domänenübergreifenden Modelle des Digitalen Zwillings des MPS und die Durchführung von What-If-Simulationen zur Erfüllung neuer

Produktanforderungen sowie die automatisierte Erstellung der SPS-Steuerungssoftware für das MPS mit Engineering-Tools.

In einer weiteren Fallstudie wurde am IAS ein vergleichbares modulares Produktionssystem mit einer ähnlichen Anzahl von Komponenten ohne den Einsatz des Digitalen Zwillings rekonfiguriert. Der Prozess wurde mit der Identifizierung bestehender Komponenten und der Identifizierung notwendiger Modifikationen und Nebenwirkungen durch manuelle Tests gestartet. In der Folge wurden die notwendigen Ersatzteile auf ihre Eigenschaften hin untersucht, ausgewählt und bestellt. anschließend wurde das System physikalisch assembliert. Danach wurde das CAD-Modell des Systems erstellt und das Gesamtsystem anhand der aktuellen Komponenten und des Soll-Ablaufs simuliert. Schließlich wurde die SPS-Steuerungssoftware direkt auf das System geschrieben und das System in Betrieb genommen. Die gesamte Rekonfiguration erfolgte in diesem Fall durch zwei parallele Studentenarbeiten mit Unterstützung einer wissenschaftlichen Mitarbeiterin am IAS in insgesamt ca. 12 Monaten.

Eine quantitative Bewertung der beiden Ansätze verdeutlicht, dass die Rekonfiguration eines automatisierten Systems mit synchronisierten Modellen mittels der Ankerpunktmethodologie eine höhere Verfügbarkeit der Anlage und einen effizienten Rekonfigurationsprozess gewährleistet. Darüber hinaus minimiert die Rekonfiguration mit diesem Ansatz das Produktionsrisiko nach der Inbetriebnahme des Systems, da bereits mehrere Fertigungsszenarien in einer What-If-Simulation unter Verwendung des Digitalen Zwillings getestet wurden und die Testergebnisse bereits vollständig sind.

Zusätzlich zur Erstellung des MPS und seines Digitalen Zwillings zur Konzeptevaluierung wurden während dieser Arbeiten ein intelligentes Lager und sein Digitaler Zwilling in der ARENA2036 entworfen und realisiert. Aufgrund der sehr industrienahen Umgebung in der ARENA2036 in Form von Modifikationen des realen Automatisierungssystems im laufenden Betrieb durch domänenübergreifende Fachexperten sowie der Prüfung der Adaptierbarkeit des Assistenzsystems mittels eines industrienahen automatisierten Systems wurde das intelligente Lager in Kooperation mit diversen Industriepartnern in einer flexiblen Produktionsanlage errichtet. Im Unterkapitel 6.2 wird der Aufbau des Systems sowohl in realer als auch in digitaler Form beschrieben, das Evaluierungsszenario vorgestellt und Evaluierungsergebnisse anhand des intelligenten Lagers präsentiert.

6.2 Szenario 2: Intelligentes Lager in der flexiblen Produktionsanlage

Auf dem Forschungscampus ARENA2036, Active Research Environment for the Next Generation of Automobile, forschen die Institute der Universität Stuttgart gemeinsam mit Industriepartnern an der wandlungsfähigen Fabrik der Zukunft im Jahr 2036, dem 150. Jahrestag des Automobils. Im Fokus steht die nächste Generation von Automobilen, das Design einer agilen

und flexiblen Produktion und eines intelligenten Leichtbaus [163]. Zur Evaluierung der Ausgangssituation dieser Arbeit sowie der Forschungsanforderungen anhand eines industrienahen Einsatzes wird im Rahmen dieser Arbeit ein flexibles intelligentes Lager, iLager, für eine flexible Fertigungsanlage in der ARENA2036 entworfen und aufgebaut. Die flexible Produktionsanlage besteht aus vier automatisierten Systemen mit dezentraler Steuerung (Schweißmaschine, mobiler Roboter, intelligentes Lager und Steuerschrank zur Kopfsteuerung), die aus vier Blechteilen durch Kommunikation über WLAN ein Modellauto herstellen.

Die automatisierten Systeme sind nicht in einem festen, konventionellen Liniengestänge angeordnet, sondern werden von einem mobilen Roboter als fahrerloses Transportfahrzeug verbunden. Dies ermöglicht einen variablen und leicht modifizierbaren Fertigungsprozess. Das iLager wird als Vorratslager für die vorgefertigten Blechteile verwendet, welche dem mobilen Roboter zur Verfügung gestellt werden. Die Blecheinzelteile werden zum besseren Handling in unterschiedlichen Werkstückträgern zusammengefasst. Die Position des iLagers ist dank seines fahrbaren Aufbaus flexibel und der Roboter kann durch Kommunikation per WLAN dessen Position erkennen. Das iLager besteht aus 37 Sensoren, 25 Aktoren, einer SPS und einem WLAN-Modul zur Kommunikation zwischen den dezentralen Peripheriekomponenten mit dem Kopfsteuerschrank sowie dessen Batterie.

Die Bearbeitungsstationen und deren Sensoren und Aktoren sowie die Steuerung und Laststromversorgung des iLagers sind in Tabelle 8 aufgelistet.

Tabelle 8: Bearbeitungsstationen sowie die Steuerungs- und Laststromversorgungs-Komponenten des iLagers

Anzahl von mechatronischen Komponenten Bearbeitungsstationen in iLager	Sensoren	Aktoren
Funktionsgruppe Ablage	16 Lichtschranken	12 Hubmagnete 1 Anzeige
Funktionsgruppe Ausgabe	16 Lichtschranken 5 RFID-Scanner	12 Hubmagnete
Steuerungskomponente:	SPS-ET.200-SP iWLAN Client SCALANCE W734-1 IO: PROFINET 5 Input-, 4 Output-Module	
Laststromversorgung	Batterie 24V/12Ah Hutschienen-Netzteil 24V/10A	

Abbildung 43 stellt die Bestandteile der flexiblen Produktionsanlage dar. Abgesehen vom realen Aufbau des iLagers wurde auch sein Digitaler Zwilling mit Engineering-Tools realisiert. Diese Tools wurden auf der installierten Teamcenter-PLM-Plattform auf dem Server der ARENA2036 integriert.

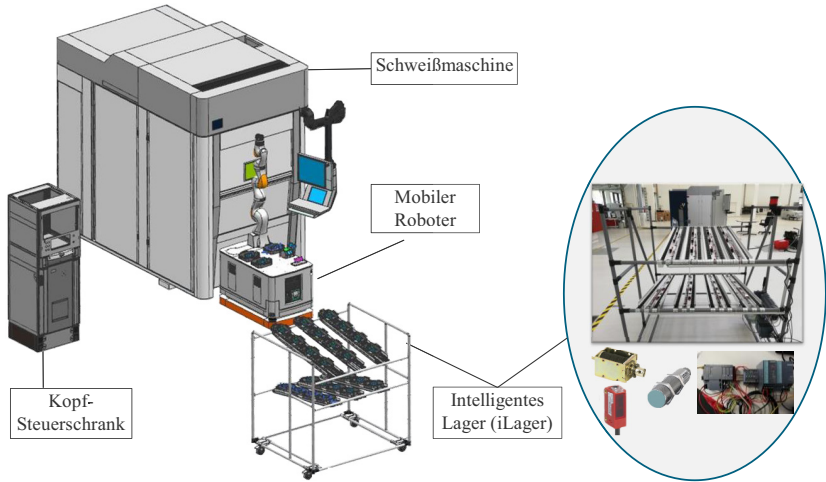


Abbildung 43: intelligentes Lager in der flexiblen Produktionsanlage

Der nächste Abschnitt beschreibt diesen Engineering-Prozess sowie dessen Anwendung im Detail.

6.2.1 Digitaler Zwilling des intelligenten Lagers

Der Digitale Zwilling des iLagers wurde auf der installierten Teamcenter PLM-Plattform als Datenbackbone auf dem Server der ARENA2036 erstellt. Tabelle 9 listet alle Tools und die damit erstellen Modelle innerhalb des Digitalen Zwillings vom intelligenten Lager auf. Hier wurden die Modelle des iLagers mit NX-Modeling, Line Designer, Automation Designer, TIA-Portal, PLCSIM Advanced in ähnlicher Reihenfolge wie der Engineering-Prozess des Digitalen Zwillings des MPS erstellt. Des Weiteren wurden die Digitalen Zwillinge der anderen automatisierten Systeme innerhalb der flexiblen Produktionsanlage wie Schweißmaschine, mobiler Roboter und Steuerschrank im Teamcenter auf dem Server der ARENA2036 durch die Industriepartner mit den gleichen Engineering-Tools erstellt. Abschließend wurde der gesamte Modellauto-Produktionsprozess in einer virtuellen Inbetriebnahme mit den Tools Process Simulate, PLCSIM Advanced und TIA Portal simuliert und validiert.

Nach der Validierung des Gesamtsystems durch eine virtuelle Inbetriebnahme, wurde die SPS-Steuerungssoftware des intelligenten Lagers auf einem Steuerungssoftware-Repository mit dem Referenzzeitpunkt-Label gespeichert.

Tabelle 9: Tools und die erstellte Modelle innerhalb des Digitalen Zwillings des iLagers

Tools	Erstelltes Modell im Digitalen Zwilling
NX-Modeling	3D-CAD-Modell der einzelnen Assets des iLagers
Line Designer	3D-CAD Modell des gesamten iLagers
Automation Designer	Elektrisches Modell des iLagers Signalmodell der mechatronischen Komponenten Funktionsblockmodelle der Funktionsgruppen
TIA-Portal	Steuerung-Ablaufmodell
PLCSIM Advanced	Modell der CPU von SPS
Visual Studio	Verhaltens- und Prozessablaufs-modell des iLagers
Teamcenter PLM-Plattform	Integrierte domänenübergreifende Modelle des iLagers

Abbildung 44 zeigt das Anwendungsszenario des iLagers mit seinem Digitalen Zwilling zur Evaluierung der Forschungsanforderungen. Während des Betriebs der flexiblen Produktionsanlage musste das iLager durch Industriepartner ständig angepasst oder optimiert werden, z.B. durch Wartung, Austausch von Komponenten, Optimierung des gesamten Prozessablaufs, etc. Nach diesen Änderungen wurde die Steuerungssoftware regelmäßig mit einem Zeitlabel auf dem Repository gespeichert. Dabei musste das entwickelte Assistenzsystem Änderungen im System automatisch erkennen, indem es zwei SPS-Steuerungssoftwares aus dem

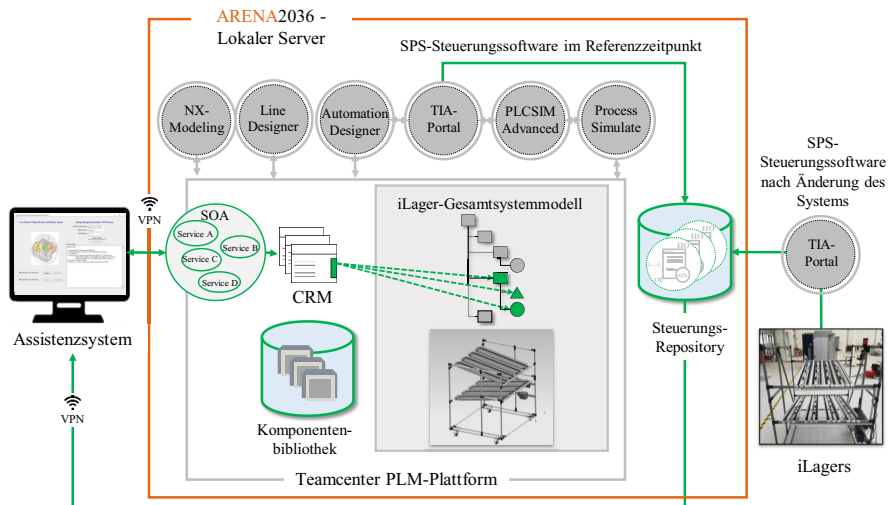


Abbildung 44: Evaluierungsszenario anhand des iLagers in der flexiblen Produktionsanlage

Steuerungsrepository aufrief und analysierte. So konnten CRMs an die veränderten Ankerpunkte des Digitalen Zwillings vom iLager im Teamcenter erstellt werden.

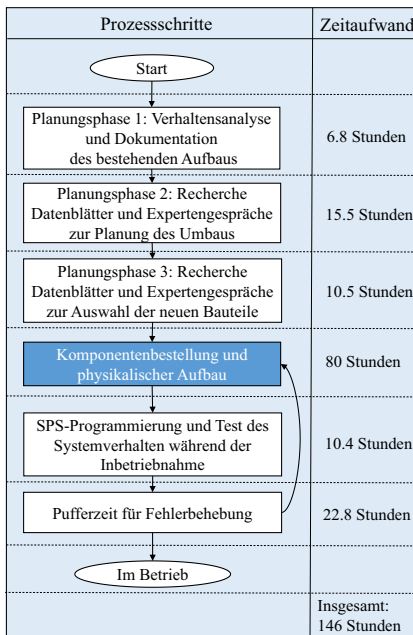
6.2.2 Bewertung der Evaluierungsergebnisse

Zur Evaluierung der Ankerpunktmethode wurde in einer Fallstudie im Rahmen einer studentischen Arbeit in der ARENA2036 die Rekonfiguration des iLagers auf zwei verschiedene Arten durchgeführt [164]. Der erste Ansatz umfasst die Synchronisation der Modelle des Digitalen Zwillings mit dem entwickelten Assistenzsystem und die Rekonfiguration des iLagers unter Verwendung des Digitalen Zwillings. Der zweite Ansatz umfasst die Rekonfiguration des iLagers nach einem heute gängigen Umbauprozess in der Industrie ohne Unterstützung des Digitalen Zwillings. Im zweiten Ansatz können daher die Potenziale des Digitalen Zwillings nicht zur Rekonfiguration herangezogen werden, wie Model-based-System-Engineering, domänenübergreifende, wiederverwendbare Komponentenbibliotheken, What-If-Simulationen, virtuelle Inbetriebnahme und die daraus resultierenden Vorteile.

Zur Ermittlung des Ablaufs und der Dauer des Rekonfigurationsprozesses in diesem Ansatz wurde in der ARENA2036 eine Umfrage durchgeführt. Hierbei wurde zur Erhebung der Vergleichsdaten ein Fragebogen entworfen und von *elf unterschiedliche Fachexperten* aus den Gebieten des Anlagenbaus und -umbaus ausgefüllt. Die Befragten arbeiten als Entwicklungsingenieure, Projektmanager und SPS-Programmierer mit professioneller Erfahrung mit realen automatisierten Systemen. Des Weiteren wurden wissenschaftliche Mitarbeiter befragt, die auf dem Bereich der Rekonfiguration forschen. Bei den Experten handelt es sich um Mitarbeiter der Unternehmen: Bär Automation GmbH, Kuka AG, Siemens AG, Siemens Industry Software GmbH, Schunk GmbH & Co. KG, das Fraunhofer IPA und das IAS der Universität Stuttgart. Die Umfrage wurde ausschließlich in direktem Bezug auf das iLager durchgeführt. Den Befragten wurde der reale Aufbau des iLagers gezeigt und anschließend die grundlegende Aufgabe des Systems beschrieben. Es wurde auf die vorhandenen Modelle hingewiesen und die aufgetretenen Änderungen im System erwähnt, ohne deren Inhalt exakt zu benennen. Mit diesem Vorwissen wurden die Umfrageteilnehmer gebeten, den Fragebogen beantworten. Hierbei wurden die Arbeitsschritte der am häufigsten genannten Methoden und deren Reihenfolge kombiniert und durchschnittliche Bearbeitungsdauer der Einzelschritte berechnet. Abbildung 45 skizziert die Abfolge der Prozessschritte und die durchschnittliche Zeitdauer für die einzelnen Schritte der Rekonfiguration ohne Verwendung des Digitalen Zwillings sowie die Ergebnisse der tatsächlichen Rekonfiguration des iLagers unter Verwendung des synchronisierten Digitalen Zwillings anhand der Ankerpunktmethode.

Der Prozess ohne Verwendung des Digitalen Zwillings beginnt mit der Erforschung des Verhaltens des Systems und der Erkennung seiner Komponenten. Es folgt eine Expertenbesprechung über die erforderlichen Modifikationen und Komponenten im System, um die Kundenanforderungen zu erfüllen. Anschließend werden die entsprechenden Komponenten

1. Rekonfigurationsprozesse des iLagers ohne Digitalen Zwilling
(Anzahl der befragten Fachexperten = 11)



2. Rekonfigurationsprozesse des iLagers mit synchronisiertem Digitalen Zwilling durch Ankerpunktmethode

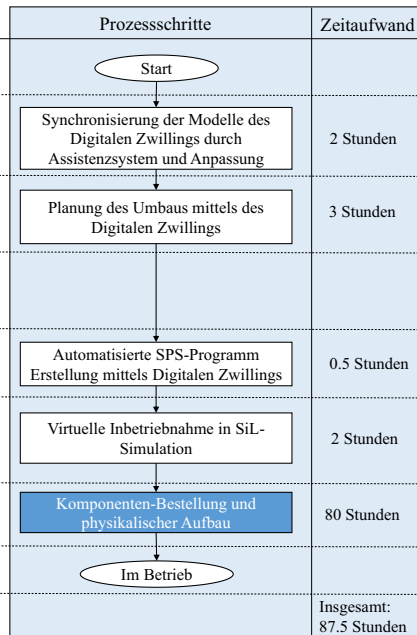


Abbildung 45: Abfolge der Prozessschritte und die durchschnittliche Zeitdauer für die einzelnen Schritte der Rekonfiguration ohne und mit Verwendung des Digitalen Zwillings

bestellt und das iLager umgebaut. Das SPS-Programm wird im nächsten Schritt basierend auf dem Soll-Prozess direkt auf das System geschrieben und das System wird für diverse Szenarien mit manuellen Tests in einem Prüfzeitraum geprüft. Wenn im System keine Fehler auftreten, wird das iLager in Betrieb genommen. Gemäß den Antworten auf die Umfrage kann es in der Praxis vorkommen, dass während der Pufferzeit im System ein Fehler erkannt wird und der Umbauprozess entsprechend wiederholt wird.

Im Rekonfigurationsansatz unter Verwendung des Digitalen Zwillings wird im ersten Schritt mit Hilfe des Assistenzsystems die Änderungserkennung durchgeführt. In ca. 2 Minuten hat das Assistenzsystem die Analyse der SPS-Steuerungssoftwares zum Referenzzeitpunkt und des Ist-Zustands erfolgreich abgeschlossen und alle auftretenden Änderungen im iLager und deren Abhängigkeiten während des Betriebs korrekt erkannt. Darüber hinaus wurden in diesem Zeitraum korrekte CRMs an die Ankerpunkte des iLagers auf der PLM-Plattform gesendet. Die Erkennung der exakten mechanischen Lokalisierungen der Änderungen ist jedoch eine Herausforderung, die mit der Ankerpunktmethode allein nicht automatisiert zu bewältigen ist und

zwangsläufig die Zusammenarbeit mit einem Systemingenieur erfordert. Allerdings kann die Anpassung der genauen Position der Änderungen in den Modellen je nach Umfang in wenigen Stunden erfolgen. Im Falle des iLagers konnten die Änderungen in etwas weniger als zwei Stunden eingepflegt werden. Die weiteren Schritte zur Rekonfiguration des iLagers in diesem Ansatz sind sehr ähnlich wie die Rekonfigurationsschritte des MPS unter Verwendung seines Digitalen Zwillings im Abschnitt 6.1.

Die Ergebnisse der Umfrage zur Rekonfiguration ohne Digitaler Zwilling und die ermittelten Zeiten für eine Rekonfiguration mit einem synchronisierten Digitalen Zwilling anhand der Ankerpunktmethode dienen als Grundlage für eine Bewertung der Mehrwerte durch die Ankerpunktmethode. Abbildung 46 stellt die zeitliche Auswertung beider Rekonfigurationsprozesse gegenüber.

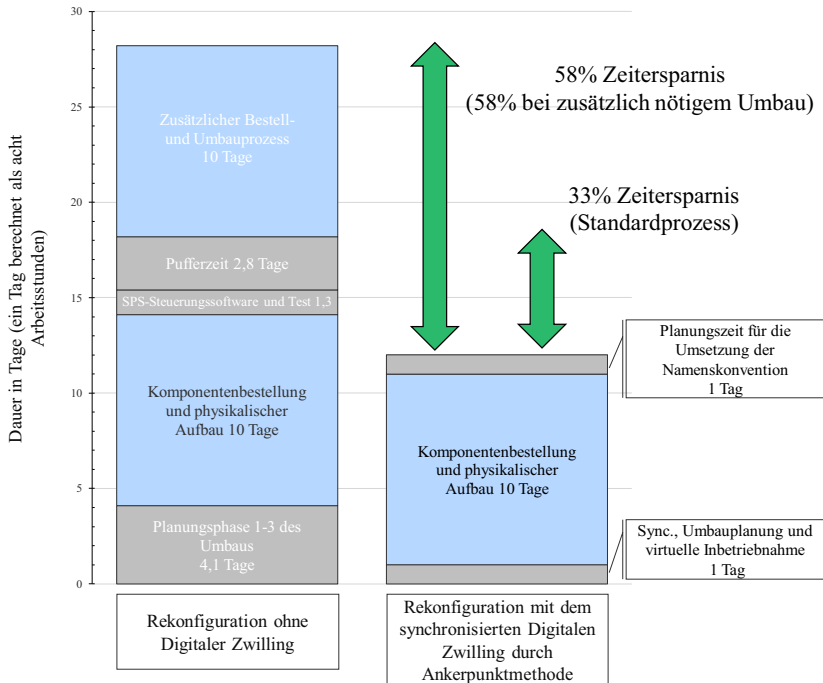


Abbildung 46: Zeitdauer- und Prozessvergleich der beiden Rekonfigurationsprozesse mit und ohne synchronisierten Digitalen Zwilling mittels der Ankerpunktmethode

Bei der Rekonfiguration ohne Verwendung des Digitalen Zwillings ist aufgrund der fehlenden Simulation zu erwarten, dass die Rekonfiguration beim ersten Versuch nicht vollständig erfolgreich sein wird. Aus diesem Grund wird für den Umbauprozess ohne den Einsatz eines Digitalen Zwillings mit einem zusätzlichen Bestellprozess und einer neuen Umbauphase für 10

Tage gerechnet. Darüber hinaus wurde zur korrekten Bewertung beider Ansätze der zusätzliche Zeitaufwand für die Planung der Ankerpunkte (Berücksichtigung der Namenskonvention) im Engineering-Prozess in die Gesamtdauer des Umbaus einbezogen. In Hinblick auf die quantitative Bewertung, welche auf den Evaluierungsergebnissen basiert, konnte durch die Ankerpunktmethod und den Einsatz eines Digitalen Zwillings der Umbauprozess des intelligenten Lagers um etwa 33 Prozent im Standard-Prozess und 58 Prozent bei einem zusätzlich notwendigen Umbau verkürzt werden. Auch in Bezug auf die Verfügbarkeit des Lagers wurde eine Verkürzung des Stillstands erzielt. Der physikalische Testbetrieb der Anlage kann entfallen, da dieser bereits virtuell durchgeführt wurde. Im Rahmen einer qualitativen Bewertung ist zu vermerken, dass durch die Vollständigkeit der Testergebnisse durch die Simulation mehrerer Fertigungsszenarien eine Rekonfiguration unter Verwendung des Digitalen Zwillings das Produktionsrisiko nach der Inbetriebnahme reduziert. Tabelle 10 fasst die qualitative und quantitative Bewertung des Evaluierungsszenarios zusammen.

Tabelle 10: Quantitative und qualitative Bewertung der Evaluierungsergebnisse

Evaluierungsszenario \ Evaluierungsergebnisse	Quantitative Bewertung	Qualitative Bewertung
Rekonfiguration des iLagers mit dem synchronisierten Digitalen Zwilling durch die Ankerpunktmethod im Vergleich zur Rekonfiguration ohne Nutzung des Digitalen Zwillings	✓ Höhere Verfügbarkeit der Anlage	✓ Minimierung von Produktionsrisiken
	<ul style="list-style-type: none"> • 33% Zeitersparnis beim Standardprozess • 58% Zeitersparnis bei zusätzlich notwendigem Umbau 	<ul style="list-style-type: none"> • Vollständigkeit der Testergebnisse durch Simulation mehrerer Fertigungsszenarien

6.3 Erfüllung der Forschungsanforderungen und Zielsetzung der Arbeit

Nach der Umsetzung der Ankerpunktmethod mittels eines Assistenzsystems und seinem Einsatz in zwei Evaluationsszenarien, einem modularen Produktionssystem und einem intelligenten Lager in der ARENA2036, gilt es nun zu prüfen, ob die in Unterkapitel 1.3 gestellten Anforderungen erfüllt sind.

(A1) Detektion der Änderungen bis zur Feld-Ebene in automatisierten Systemen

Die Evaluierungsergebnisse in den Kapiteln 6.1 und 6.2 haben gezeigt, dass die Ankerpunktmethod, die in einem Assistenzsystem mündete, in beiden Evaluierungsszenarien

alle Änderungen bis zu den Sensoren und Aktoren detektieren konnte. Durch den Vergleich der statischen Elemente der SPS-Steuerungssoftware vom Referenzzeitpunkt und der aktuellsten Softwareversion basierend auf einer formalisierten Methode lassen sich die Ankerpunkte der geänderten Komponente in der Softwaredomäne erfassen. Durch den erfolgreichen Einsatz der Ankerpunktmethod in zwei Evaluationsszenarien wurde demonstriert, dass die automatisierte Erkennung von Änderungen bis in die Feldebene in automatisierten Systemen erfüllt ist.

(A2) Detektion der domänenübergreifenden Abhängigkeiten der Änderungen

Die zur Evaluierung erstellten Änderungsszenarien spiegeln realistische Änderungen in der Inbetriebnahme und Betriebsphase automatisierter Systeme wieder. In diesen Evaluierungsszenarien wurden alle domänenübergreifenden Abhängigkeiten der Änderungen mit Hilfe der Ankerpunktmethod korrekt identifiziert. Die Ankerpunktmethod ermöglicht eine generische regelbasierte Analyse der erfassten Ankerpunkte und ihrer Relationen in der Softwaredomäne sowie die Identifizierung von verdeckten domänenübergreifenden Abhängigkeiten der Änderungen in den Domänen Mechanik und Elektrik. Die einfache Anpassungsfähigkeit an verschiedene diskrete automatisierte Systeme mit unterschiedlichen Charakteristika in Bezug auf die regelbasierte Abhängigkeitsanalyse wurde in dieser Arbeit mit zwei verschiedenen Automatisierungssystemen erfolgreich nachgewiesen.

(A3) Anpassung der domänenübergreifenden Modelle des Digitalen Zwillings

Im Rahmen der Ankerpunktmethod wurde eine automatisierte Erstellung generischer Change-Request-Modelle zur Anpassung der Modelle des Digitalen Zwillings ohne Bedarf an domänenübergreifendem Fachwissen über die komplexen und heterogenen Modelle eingeführt. Dieses Konzept ermöglicht eine hochgradig automatisierte Anpassung der geänderten Ankerpunkte des Digitalen Zwillings unter Verwendung einer wiederverwendbaren Komponentenbibliothek sowie einer funktionalen Schnittstelle zwischen dem Softwaresystem zur Modellanpassung und dem Datenbackbone des Gesamtsystemmodells. In zwei Evaluierungsszenarien wurde dieses Konzept mit einer servicebasierten Schnittstelle zwischen einem Assistenzsystem und dem Datenbackbone des Digitalen Zwillings erfolgreich validiert.

Folglich entspricht das entwickelte Konzept allen drei Anforderungen. Der konkrete Einsatz des Konzepts in den beiden Evaluationsszenarien hat zudem bewiesen, dass das Konzept auf bestehende Technologien, die in der Industrie zur Erstellung Digitaler Zwillinge eingesetzt werden, integrierbar und adaptierbar ist. Darüber hinaus wurde gezeigt, dass dieses Konzept die Ingenieure bei der Rekonfiguration eines automatisierten Systems unterstützt und insbesondere die Bedürfnisse der Systemingenieure berücksichtigt. Dementsprechend wurden die in Unterkapitel 1.4 definierten Ziele der Arbeit, (1) die aktive Unterstützung der Ingenieure bei der Rekonfiguration einer Anlage und (2) die Kompatibilität des Konzepts mit bestehenden Technologien erfüllt.

7 Schlussbetrachtung

7.1 Zusammenfassung der Ergebnisse und Bewertung

Heutzutage steht die Industrie zunehmend vor der Herausforderung kundenspezifische Produkte in stetig sinkenden Produktionszeiten zu liefern. Verkürzte Produktlebenszyklen und steigende Marktanforderungen erfordern eine ständige Verbesserung und Rekonfiguration der aktuellen Produktion. Hierfür werden flexible automatisierte Produktionssysteme benötigt, die bei Bedarf effizient rekonfiguriert werden können. Effizient rekonfigurierbare Systeme bieten eine hohe Anlagenverfügbarkeit und ermöglichen die Herstellung kundenspezifischer Produkte. Durch den Einsatz eines Digitalen Zwillings während der Rekonfiguration kann einerseits die Rekonfigurationszeit reduziert und damit die Anlagenverfügbarkeit erhöht werden, andererseits können die Produktionsrisiken minimiert werden, indem verschiedene Testszenarien bei What-If-Simulationen innerhalb des Digitalen Zwillings ausgeführt werden.

Nach dem Stand der Wissenschaft und Technik überwiegt heute die Erstellung des Digitalen Zwillings für ein automatisiertes System während des Engineering-Prozesses. Jedoch kommt es nach der Inbetriebnahme des automatisierten Systems – u.a. auch marktbedingt - oft zu vielen Änderungen im realen System, die nicht im ursprünglich erstellten Digitalen Zwilling angepasst werden. Diese führen zu einer Abweichung zwischen dem realen System und dessen Digitalen Zwilling. Daher fehlt eine Methodik, die automatisiert domänenübergreifende Änderungen im realen automatisierten System erfasst und entsprechend die Modelle des Digitalen Zwillings anpasst, ohne dass das Gesamtsystemmodell seine Konsistenz verliert.

In dieser Arbeit wird eine systematische Ankerpunktmethode konzipiert, die eine automatisierte Änderungsdetektion und Modellanpassung innerhalb des Digitalen Zwillings in drei Phasen ermöglicht. Um die Änderungen in der Steuerungssoftware erkennen zu können, findet in der ersten Phase ein Vergleich zwischen der SPS-Steuerungssoftware des automatisierten Systems zum Referenzzeitpunkt und zum aktuellen Zeitpunkt unter Verwendung einer Formalisierungsmethode statt. In der zweiten Phase wird eine regelbasierte Analyse der erfassten Änderungen in der Steuerungssoftware durchgeführt, um die domänenübergreifenden Abhängigkeiten der Änderungen sowie das auftretende Änderungsszenario im exakt System zu identifizieren. In der dritten Phase erfolgt die Referenzierung der domänenübergreifenden Modelle und ihrer Anpassungsparameter an den geänderten Ankerpunkten des Gesamtsystemmodells des Digitalen Zwillings durch die Verwendung einer Komponentenbibliothek und die Erstellung von gekapselten Change-Request-Modellen. Schließlich muss ein Systemingenieur die Änderungen im Gesamtsystemmodell des Digitalen Zwillings unter Verwendung der Informationen und Modelle innerhalb des Change-Request-Models ausführen.

Die Realisierung der drei Phasen der Ankerpunktmethode geschieht durch ein Assistenzsystem. Das Assistenzsystem lässt sich auf einem Betriebssystem auf einem Computer installieren. Es besteht aus verschiedenen Komponenten hinsichtlich der Analyse der SPS-Steuerungssoftware und zur Erkennung der domänenübergreifenden Änderungen sowie Abhängigkeiten und aus einer funktionalen Schnittstelle zur Erstellung und Referenzierung der Change-Request-Modelle an den geänderten Ankerpunkten des Digitalen Zwillings innerhalb eines PLM-Systems als Datenbackbone.

Um den Forschungsbedarf und die Ziele dieser Arbeit sowie die Adaptierbarkeit des Assistenzsystems in unterschiedlichen Anwendungsbereichen evaluieren zu können, wurden am IAS und in der ARENA2036 zwei forschungs- und industriennahe Evaluierungsszenarien physisch und digital konzipiert. Vor diesem Hintergrund erfolgte die Evaluierung des Assistenzsystems. Die qualitative und quantitative Bewertung der Evaluierungsergebnisse ergab, dass mit der Ankerpunktmethode alle Änderungsszenarien, die die Steuerungssoftware des Systems beeinflussen, automatisiert erkannt und auf die Modelle des Digitalen Zwillings referenziert wurden. Darüber hinaus konnte gezeigt werden, dass sich bei der Rekonfiguration eines automatisierten Systems mit einem synchronisierten Digitalen Zwilling unter Verwendung der Ankerpunktmethode die Verfügbarkeit einer Anlage um 58 Prozent erhöht und sich damit auch die Produktionsrisiken verringern.

7.2 Ausblick

Die hier durchgeführten Forschungsarbeiten führten darüber hinaus zu weiteren Erkenntnissen, die als Ausgangspunkte für neue Forschungsthemen dienen können:

Detektion allein mechanischer oder elektrischer Änderungen: Mit der Ankerpunktmethode lassen sich Änderungen feststellen, die einen Einfluss auf die Steuerungssoftware haben. Ausschließlich mechanische oder elektrische Änderungen sind mit der Ankerpunktmethode nicht zu erkennen. Die Anwendung der Ankerpunktmethode - neben anderen Technologien wie dem 3D-Laserscan oder der Analyse der Änderungsdokumentation in der realen Anlage - ermöglicht eine mehrdimensionale Synchronisierung der Modelle des Digitalen Zwillings. Eine weitere Quelle für die Änderungserkennung innerhalb des Systems sind die historischen und aktuellen Betriebsdaten der Sensoren und Aktoren des Systems. Diese Daten können für die folgenden Forschungen als Datenquelle dienen. Durch die Analyse dieser Betriebsdaten werden genaue Prozessabläufe und kinematische Bewegungen der Assets ermittelt. Zusammengefasst lässt sich die Ankerpunktmethode mit den anderen Methoden zur Änderungsdetektion zukünftig kombinieren.

Erweiterung der Regel zur Änderungsdetektion: Bei der Ankerpunktmethode erfolgte die Definition einer limitierten Anzahl von Regeln für die Analyse der Änderungen innerhalb der Steuerungssoftware auf der Grundlage einer praktischen Untersuchung und die Validierung ihres

generischen Charakters anhand von zwei verschiedenen Evaluierungsszenarien. Die Ergebnisse dieser Forschungen lassen es nicht zu, zu behaupten, dass alle möglichen Änderungen für alle Anwendungsszenarien durch diese Regeln abgedeckt sind. Jedoch ist eine Erweiterung dieser definierten Regeln sehr einfach möglich. In Zukunft können diese Regeln anhand einer kontinuierlichen Analyse der Elemente der Steuerungssoftware mithilfe maschinellen Lernens selbstständig erstellt werden. Dadurch ist es möglich, ein Assistenzsystem zu entwickeln, das seine Regeln ohne die Voraussetzung eines menschlichen, domänenübergreifenden Wissens erweitern kann.

Erweiterung der Methode zur Modellanpassung: Aufgrund der Einschränkungen der heutigen Technologien in der Industrie ist eine vollautomatische Anpassung der Modelle des Digitalen Zwillings durch die Ankerpunktmethode nicht durchführbar. Die Ankerpunktmethode kann jedoch den Systemingenieuren bei der Anpassung der Modelle des Digitalen Zwillings mit der Erstellung der Change-Request-Modelle durch den hohen Automatisierungsgrad unterstützen und die Forderung nach einer domänenübergreifenden Fachkenntnis für den Einsatz verschiedener Tools bei Ingenieuren abmildern. Die Weiterentwicklung der Technologien im Rahmen der Modellspeicherung und der generischen Modellierung in der Industrie, beispielsweise durch den Einsatz von neuen Technologien wie grafikbasierte Datenbanken, wird es in Zukunft möglich sein, weitere Ansätze für die vollautomatische Anpassung der Modelle des Digitalen Zwillings zu entwickeln. Somit ist zukünftig statt von einer assistierenden Anpassung der Modelle des Digitalen Zwillings von einer vollautomatischen Synchronisierung auszugehen.

Literaturverzeichnis

- [1] N. Jazdi, “Universelle Fernservice-Infrastruktur für eingebettete Systeme, Dissertation,” Universität Stuttgart, 2003.
- [2] DIN SPEC 91345, *Deutsches Institut für Normung, Referenzarchitekturmodell Industrie 4.0 (RAMI4.0)*. Berlin: Beuth Verlag, 2016.
- [3] VDI/VDE-Gesellschaft Mess- und Automatisierungstechnik, “VDI/VDE-GMA 7.21: Industrie 4.0 – Begriffe/Terms,” 2017.
- [4] R. Lauber and P. Göhner, *Prozessautomatisierung 1*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999.
- [5] C. Gomes, C. Thule, D. Broman, P. G. Larsen, and H. Vangheluwe, “Co-Simulation: A Survey,” *ACM Comput. Surv.*, vol. 51, no. 3, pp. 1–33, May 2018.
- [6] B. Ashtari Talkhestani *et al.*, “An architecture of an Intelligent Digital Twin in a Cyber-Physical Production System,” - *Autom.*, vol. 67, no. 9, pp. 762–782, Sep. 2019.
- [7] C. R. Maga, “Adaptierbares Wiederverwendungskonzept für die Entwicklung von automatisierten Systemen, Dissertation,” Universität Stuttgart, 2013.
- [8] M. Rauscher, “Agentenbasierte Konsistenzprüfung heterogener Modelle in der Automatisierungstechnik, Dissertation,” Universität Stuttgart, 2015.
- [9] I. Hanschke, “Beherrschen der IT-Komplexität mithilfe von EAM,” *Wirtschaftsinformatik Manag.*, vol. 3, no. 4, pp. 66–71, 2011.
- [10] M. Weyrich, F. Steden, J. Wolf, and M. Scharf, “Identification of mechatronic units based on an example of a flexible customized multi lathe machine tool,” *IEEE Int. Conf. Emerg. Technol. Fact. Autom. ETFA*, pp. 7–10, 2011.
- [11] J. Matevska, *Rekonfiguration komponentenbasierter Softwaresysteme zur Laufzeit*. Wiesbaden: Vieweg+Teubner Verlag, 2010.
- [12] H. A. El Maraghy, “Flexible and reconfigurable manufacturing systems paradigms,” in *Flexible Services and Manufacturing Journal*, 2006, vol. 17, no. 4 SPECIAL ISSUE, pp. 261–276.
- [13] C. Diedrich, T. Hadlich, and M. Thron, “Semantik durch Merkmale für Industrie 4.0,” in *Handbuch Industrie 4.0 Bd.2*, Springer Berlin Heidelberg, 2017, pp. 417–432.
- [14] M. Wedel, “Bewertung der Zuverlässigkeit von Automatisierungssystemen in frühen Entwicklungsphasen, Dissertation,” Universität Stuttgart, 2011.
- [15] C. M. MacKenzie, K. Laskey, F. McCabe, P. F. Brown, and R. Metz, “Reference Model for Service Oriented Architecture 1.0. OASIS Standard,” *OASIS Open*, 2006. [Online]. Available: http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=soa-rm.
- [16] E. Westkämper, T. Baudisch, W. Schlögl, and G. Frank, “Automatic Model Generation for Virtual Commissioning of Specialized Production Machines,” *Softwaretechnik-Trends*, vol. 32, no. 2, pp. 82–83, May 2012.
- [17] J. Kiefer, T. Breckle, R. Stetter, M. Manns, S. Allegretti, and T. Breckle, “Digital assembly planning using graph-based design languages,” *Procedia CIRP*, vol. 72, pp. 802–807, 2018.
- [18] M. Weyrich, I. R. Wior, D. Bchennati, and A. Fay, “Flexibilisierung von Automatisierungssystemen - Systematisierung der Flexibilitätsanforderungen von Industrie 4.0,” *wt Werkstattstechnik online*, 2014. .
- [19] P. Marks, M. Weyrich, X. L. Hoang, and A. Fay, “Agent-based adaptation of automated manufacturing machines,” in *IEEE International Conference on Emerging Technologies and Factory Automation, ETFA*, 2017, pp. 1–8.
- [20] F. S. Fogliatto and G. J. C. da Silveira, Eds., *Mass Customization*. London: Springer London, 2011.

- [21] G. Schuh, A. Kampker, and B. Franzkoch, "Anlaufmanagement Kosten senken – Anlaufzeit verkürzen – Qualität sichern," *Werkstattstech. online*, vol. 95, no. 5, pp. 405–409, 2005.
- [22] L. Ribeiro and M. Bjorkman, "Transitioning From Standard Automation Solutions to Cyber-Physical Production Systems: An Assessment of Critical Conceptual and Technical Challenges," *IEEE Syst. J.*, vol. 12, pp. 1–13, 2017.
- [23] H. Xuan Luu, A. Fay, P. Marks, and M. Weyrich, "Systematization approach for the adaptation of manufacturing machines," in *IEEE International Conference on Emerging Technologies and Factory Automation, ETFA*, 2016, vol. 2016-Novem.
- [24] M. Weyrich *et al.*, "Flexibles Management einer dezentralen Automatisierungsverbundanlage als Beispiel für Industrie 4.0," in *VDI-Kongress Automation (VDI KA 2014)*, 2014.
- [25] G. Schuh, R. Anderl, J. Gausemeier, and M. Hompel ten, "Industrie 4.0 maturity index," *Acatech Stud.*, p. 62, 2017.
- [26] A. Faul, N. Jazdi, and M. Weyrich, "Approach to interconnect existing industrial automation systems with the Industrial Internet," in *IEEE International Conference on Emerging Technologies and Factory Automation, ETFA*, 2016, vol. 2016-Novem, pp. 1–4.
- [27] J. Qin, Y. Liu, and R. Grosvenor, "A Categorical Framework of Manufacturing for Industry 4.0 and beyond," in *Procedia CIRP*, 2016.
- [28] T. H. J. Uhlemann, C. Lehmann, and R. Steinhilper, "The Digital Twin: Realizing the Cyber-Physical Production System for Industry 4.0," in *Procedia CIRP*, 2017.
- [29] H. Zipper, F. Auris, A. Strahilov, and M. Paul, "Keeping the digital twin up-to-date — Process monitoring to identify changes in a plant," in *2018 IEEE International Conference on Industrial Technology (ICIT)*, 2018, pp. 1592–1597.
- [30] B. Ashtari Talkhestani, N. Jazdi, W. Schlögl, and M. Weyrich, "A concept in synchronization of virtual production system with real factory based on anchor-point method," *Procedia CIRP*, vol. 67, no. July, pp. 13–17, 2018.
- [31] M. Prösser, P. Moore, X. Chen, C.-B. Wong, and U. Schmidt, "A new approach towards systems integration within the mechatronic engineering design process of manufacturing systems," *Int. J. Comput. Integr. Manuf.*, vol. 26, no. 8, pp. 806–815, Aug. 2013.
- [32] T. Helbig, S. Erler, E. Westkämper, and J. Hoos, "Modelling dependencies to improve the cross-domain collaboration in the engineering process of special purpose machinery," *Procedia CIRP*, vol. 41, pp. 393–398, 2016.
- [33] W. Walla and J. Kiefer, "Life Cycle Engineering – Integration of New Products on Existing Production Systems in Automotive," in *Glocalized Solutions for Sustainability in Manufacturing*, vol. 82, no. 3, Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 207–212.
- [34] J. Kiefer, "Mechatronikorientierte Planung automatisierter Fertigungszellen im Bereich Karosserierohbau, Dissertation," Universität des Saarlandes, 2007.
- [35] M. Strube, "Modellgestützte Modernisierungsplanung industrieller Automatisierungslösungen, Dissertation," Helmut-Schmidt-Universität, 2014.
- [36] G. Bytomski, "Customised plant modernisation for high productivity," 2009. [Online]. Available: <http://www.millennium-steel.com/wp-content/uploads/articles/pdf/2010/pp73-79MS10.pdf>. [Accessed: 22-Feb-2020].
- [37] J. Kiefer, S. Allegretti, and T. Breckle, "Quality- and Lifecycle-oriented Production Engineering in Automotive Industry," *Procedia CIRP*, vol. 62, pp. 446–451, 2017.
- [38] J. Jäger, O. Schöllhammer, M. Lickefett, and T. Bauernhansl, "Advanced Complexity Management Strategic Recommendations of Handling the 'industrie 4.0' Complexity for Small and Medium Enterprises," in *Procedia CIRP*, 2016, vol. 57, pp. 116–121.
- [39] S. Boschert, C. Heinrich, and R. Rosen, "Next Generation Digital Twin," *Proc. TMCE 2018*, no. May, 2018.
- [40] Q. Qi, F. Tao, Y. Zuo, and D. Zhao, "Digital Twin Service towards Smart Manufacturing," *Procedia CIRP*, vol. 72, pp. 237–242, 2018.

- [41] F. Elezi, T. G. Maier, and U. Lindemann, "Engineering change management challenges and management cybernetics," in *2013 IEEE International Systems Conference (SysCon)*, 2013, pp. 567–573.
- [42] S. Boschert and R. Rosen, "Digital Twin—The Simulation Aspect," in *Mechatronic Futures*, Cham: Springer International Publishing, 2016, pp. 59–74.
- [43] T. Heimbold, *Einführung in die Automatisierungstechnik - Automatisierungssysteme, Komponenten, Projektierung und Planung*. München : Fachbuchverl. Leipzig im Carl-Hanser-Verl, 2015.
- [44] DIN 8580, *Deutsches Institut für Normung: Fertigungsverfahren*. Berlin: Beuth Verlag, 2003.
- [45] H. Mersch, D. Behnen, D. Schmitz, U. Eppe, C. Brecher, and M. Jarke, "Gemeinsamkeiten und Unterschiede der Prozess- und Fertigungstechnik," - *Autom.*, vol. 59, no. 1, pp. 7–17, Jan. 2011.
- [46] A. Scholz, "Unterstützung des Engineerings von fertigungstechnischen Produktionssystemen mit Hilfe von Maschinenfunktionen-Methode, Modell und Beschreibungsmittel für ein funktionsorientiertes Planen, Dissertation," Helmut-Schmidt-Universität, 2018.
- [47] B. Vogel-Heuser, A. Fay, I. Schaefer, and M. Tichy, "Evolution of software in automated production systems: Challenges and research directions," *J. Syst. Softw.*, vol. 110, pp. 54–84, Dec. 2015.
- [48] M. Bonfé and C. Fantuzzi, "Design and verification of industrial logic controllers with UML and statecharts," in *IEEE Conference on Control Applications - Proceedings*, 2003, vol. 2, pp. 1029–1034.
- [49] G. Rzevski, "On conceptual design of intelligent mechatronic systems," *Mechatronics*, vol. 13, no. 10 SPEC. Elsevier Ltd, pp. 1029–1044, 2003.
- [50] F. Harashima, M. Tomizuka, and T. Fukuda, "Mechatronics - 'What Is It, Why, and How?'" An editorial," *IEEE/ASME Trans. Mechatronics*, vol. 1, no. 1, pp. 1–4, 1996.
- [51] M. Eigner, "Überblick Disziplin-spezifische und -übergreifende Vorgehensmodelle," in *Modellbasierte virtuelle Produktentwicklung*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 15–52.
- [52] Matthias Gehrke, "Entwurf mechatronischer Systeme auf Basis von Funktionshierarchien und Systemstrukturen, Dissertation," Universität Paderborn, 2005.
- [53] VDI-Gesellschaft Entwicklung Konstruktion Vertrieb, "VDI-Richtlinie 2206– Entwicklungsmethodik für mechatronische Systeme," 2004.
- [54] J. Bathelt, "Entwicklungsmethodik für SPS-gesteuerte mechatronische Systeme, Dissertation," Eidgenössische Technische Hochschule Zürich, 2007.
- [55] M. Tiegelkamp and K. H. John, *SPS-Programmierung mit IEC 61131-3*. Springer Berlin Heidelberg, 2009.
- [56] M. Eigner, D. Roubanov, and R. Zafirov, *Modellbasierte virtuelle Produktentwicklung*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014.
- [57] T. B. Helbig, "Methode zur Verbesserung der domänenübergreifenden Zusammenarbeit während des Engineering-Prozesses im Sondermaschinenbau, Dissertation," Universität Stuttgart, 2017.
- [58] VDI-Gesellschaft Entwicklung Konstruktion Vertrieb, "VDI-Richtlinie 2222- Methodisches Entwickeln von Lösungsprinzipien," 1997.
- [59] K. Ehrlenspiel and H. Meerkamm, *Integrierte Produktentwicklung: Denkabläufe, Methodeneinsatz, Zusammenarbeit*. Münschen: Carl Hanser Verlag GmbH & Co. KG, 2013.
- [60] A. Lüder, M. Foehr, L. Hundt, M. Hoffmann, Y. Langer, and S. Frank, "Aggregation of engineering processes regarding the mechatronic approach," in *IEEE International Conference on Emerging Technologies and Factory Automation, ETFA*, 2011.
- [61] N. Schmidt, A. Luder, H. Steininger, and S. Biffl, "Analyzing requirements on software tools according to the functional engineering phase in the technical systems engineering process," in *Proceedings of the 2014 IEEE Emerging Technology and Factory Automation (ETFA)*,

- 2014, pp. 1–8.
- [62] S. Biffl, E. Maetzler, M. Wimmer, A. Lueder, and N. Schmidt, “Linking and versioning support for AutomationML: A model-driven engineering perspective,” in *Proceeding - 2015 IEEE International Conference on Industrial Informatics, INDIN 2015*, 2015, pp. 499–506.
 - [63] V. Voß, “Wiederverwendbare Simulationsmodelle für die domänen-und disziplinübergreifende Produktentwicklung, Dissertation,” Universität Stuttgart, 2012.
 - [64] A. Lindworsky, *Teilautomatische Generierung von Simulationsmodellen für den entwicklungsbegleitenden Steuerungstest*, vol. 249. Herbert Utz Verlag, 2011.
 - [65] T. Jäger, “Methodik zur Identifikation von technischen Abhängigkeiten anhand von Engineeringprozessen automatisierter Anlagen, Dissertation,” Helmut-Schmidt-Universität, 2014.
 - [66] G. Frank, E. Westkämper, W. Schlögl, and M. Lenord, “System Design of PLC-Controlled Specialized Production Machines,” Springer, Berlin, Heidelberg, 2013, pp. 613–622.
 - [67] A. Strahilov and H. Hämmerle, “Engineering workflow and software tool chains of automated production systems,” in *Multi-Disciplinary Engineering for Cyber-Physical Production Systems: Data Models and Software Solutions for Handling Complex Engineering Projects*, Springer International Publishing, 2017, pp. 207–234.
 - [68] G. Frank, “Durchgängiges mechatronisches Engineering für Sondermaschinen, Dissertation,” Universität Stuttgart, 2016.
 - [69] T. Helbig, E. Westkämper, and J. Hoos, “Identifying automation components in modular manufacturing systems: A method for modeling dependencies of manufacturing systems,” *19th IEEE Int. Conf. Emerg. Technol. Fact. Autom. ETFA 2014*, 2014.
 - [70] B. Ashtari Talkhestani, N. Jazdi, W. Schloegl, and M. Weyrich, “Consistency check to synchronize the Digital Twin of manufacturing automation based on anchor points,” *Procedia CIRP*, vol. 72, pp. 159–, 2018.
 - [71] B. Ashtari Talkhestani, W. Schlögl, and M. Weyrich, “Synchronisierung von digitalen Modellen,” *atp Ed.*, vol. 59, no. 07–08, p. 62, Sep. 2017.
 - [72] CIMdata, “The CIMdata 2019 PLM Market Analysis Report Series,” 2019. [Online]. Available: https://www.cimdata.com/images/Research/PLM_MAR_Datasheet_Ltr.pdf. [Accessed: 21-Feb-2020].
 - [73] “CIMdata Publishes Global CAM Market Analysis Report - CIMdata.” [Online]. Available: <https://www.cimdata.com/en/news/item/12142-cimdata-publishes-global-cam-market-analysis-report>. [Accessed: 19-Feb-2020].
 - [74] F. Bellalouna, “Integrationsplattform für eine interdisziplinäre Entwicklung mechatronischer Produkte, Dissertation,” Ruhr-Universität Bochum, 2009.
 - [75] M. Litto, I. Korajda, C. Mangold, R. Angerbauer, W. Hils, and M. Lerche, *Baukastenbasiertes Engineering mit Föederal - Ein Leitfaden für Maschinen- und Anlagenbauer*. 2004.
 - [76] S. Sindermann, “Schnittstellen und Datenaustauschformate,” in *Modellbasierte virtuelle Produktentwicklung*, Springer Berlin Heidelberg, 2014, pp. 327–347.
 - [77] S. Vornholt, I. Geist, and Y. Li, “Categorisation of data management solutions for heterogeneous data in collaborative virtual engineering,” in *Proceedings of the First International Workshop on Digital Engineering - IWDE '10*, 2010, pp. 9–16.
 - [78] R. Drath, *Datenaustausch in der Anlagenplanung mit AutomationML*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010.
 - [79] M. Weyrich, “CAD/CAM: the ultimate step,” *Am. Mach.*, vol. 145, no. 5, pp. 77–78, 2001.
 - [80] J. Cai, M. Weyrich, and U. Berger, “Ontological machining process data modelling for powertrain production in extended enterprise,” *J. Adv. Manuf. Syst.*, vol. 04, no. 01, pp. 69–82, Jun. 2005.
 - [81] S. Herbst and A. Hoffmann, “Product Lifecycle Management (PLM) mit Siemens Teamcenter,” in *Product Lifecycle Management (PLM) mit Siemens Teamcenter*, München: Carl Hanser Verlag GmbH & Co. KG, 2018, pp. I–XII.

- [82] B. Lercher, "Konzeption und System einer Integrationsplattform zur Entwicklung von Werkzeugmaschinen, Dissertation," Technische Universität München, 2008.
- [83] M. Schleipen and O. Sauer, "Use of dynamic product and process information in a production monitoring and control system by means of CAEX and OPC UA," *Proc. 3rd Int. Conf. Chang. Agil. Reconfigurable Virtual Prod.*, pp. 662–671, 2009.
- [84] R. Schiekofer, A. Scholz, and M. Weyrich, "REST based OPC UA for the IIoT," in *IEEE International Conference on Emerging Technologies and Factory Automation, ETFA*, 2018, vol. 2018-Septe, pp. 274–281.
- [85] A. Bauer and H. Günzel, *Data-Warehouse-Systeme: Architektur, Entwicklung, Anwendung*, 4th ed. Dpunkt.verlag, 2013.
- [86] A. Kemper and A. Eickler, *Datenbanksysteme: eine Einführung*, 4. Auflage. München: Oldenbourg Verlag, 2001.
- [87] R. Drath, A. Fay, and M. Barth, "Interoperabilität von engineering-werkzeugen: Konzepte und empfehlungen für den datenaustausch zwischen engineering-werkzeugen," *At-Automatisierungstechnik*, vol. 59, no. 7, pp. 451–460, Jun. 2011.
- [88] U. Epplé, "Merkmale als grundlage der interoperabilität technischer systeme," *At-Automatisierungstechnik*, vol. 59, no. 7, pp. 440–450, Jul. 2011.
- [89] K. Reichenberger, *Kompendium semantische Netze*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010.
- [90] K. Tochtermann and H. Maurer, "Semantic Web — Geschichte und Ausblick einer Vision," in *Semantic Web*, Springer Berlin Heidelberg, 2007, pp. 1–6.
- [91] M. Shafto *et al.*, "Modeling, Simulation, Information Technology & Processing Roadmap: Technology Area 11," *Natl. Aeronaut. Sp. Adm. Washington, DC, United States Am.*, 2012.
- [92] R. Rosen, G. Von Wichert, G. Lo, and K. D. Bettenhausen, "About The Importance of Autonomy and Digital Twins for the Future of Manufacturing," *IFAC-PapersOnLine*, vol. 48, no. 3, pp. 567–572, 2015.
- [93] G. PANG *et al.*, "Multi-View 3D object recognition from a point cloud and change detection," *Pub. No.: US 2015/0254499 A1*, 2014.
- [94] W. Kritzinger, M. Karner, G. Traar, J. Henjes, and W. Sihn, "Digital Twin in manufacturing: A categorical literature review and classification," *IFAC-PapersOnLine*, vol. 51, no. 11, pp. 1016–1022, Jan. 2018.
- [95] E. A. Lee, "Cyber Physical Systems: Design Challenges," in *2008 11th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC)*, 2008, pp. 363–369.
- [96] P. Hehenberger, B. Vogel-Heuser, D. Bradley, B. Eynard, T. Tomiyama, and S. Achiche, "Design, modelling, simulation and integration of cyber physical systems: Methods and applications," *Comput. Ind.*, vol. 82, pp. 273–289, Oct. 2016.
- [97] M. Weyrich *et al.*, "Evaluation Model for Assessment of Cyber-Physical Production Systems," 2017, pp. 169–199.
- [98] N. Jazdi, "Cyber physical systems in the context of Industry 4.0," in *2014 IEEE International Conference on Automation, Quality and Testing, Robotics*, 2014, pp. 1–4.
- [99] S. Malakuti and S. Grüner, "Architectural aspects of digital twins in IIoT systems," in *Proceedings of the 12th European Conference on Software Architecture Companion Proceedings - ECSA '18*, 2018, pp. 1–2.
- [100] BMWi, "Details of the Asset Administration Shell- Part 1 - The exchange of information between partners in the value chain of Industrie 4.0, Version 1.0," 2018.
- [101] S. Haag and R. Anderl, "Digital twin – Proof of concept," *Manuf. Lett.*, vol. 15, pp. 64–66, Jan. 2018.
- [102] F. Tao *et al.*, "Digital twin-driven product design framework," *Int. J. Prod. Res.*, vol. 57, no. 12, pp. 3935–3953, Jun. 2018.
- [103] S. Yun, J. H. Park, and W. T. Kim, "Data-centric middleware based digital twin platform for

- dependable cyber-physical systems,” in *International Conference on Ubiquitous and Future Networks, ICUFN*, 2017, pp. 922–926.
- [104] H. Li *et al.*, “Application of a multi-disciplinary design approach in a mechatronic engineering toolchain,” - *Autom.*, vol. 67, no. 3, pp. 246–269, Apr. 2019.
 - [105] D. Winkler, F. Ekaputra, and S. Biffl, “AutomationML review support in multi-disciplinary engineering environments,” in *IEEE International Conference on Emerging Technologies and Factory Automation, ETFA*, 2016, vol. 2016-Novem.
 - [106] M. Weyrich, P. Klein, and F. Steden, “Reuse of modules for mechatronic modeling and evaluation of manufacturing systems in the conceptual design and basic engineering phase,” in *IFAC Proceedings Volumes (IFAC-PapersOnline)*, 2014, vol. 19, pp. 3450–3455.
 - [107] C. Hildebrandt *et al.*, “Semantic modeling for collaboration and cooperation of systems in the production domain,” in *IEEE International Conference on Emerging Technologies and Factory Automation, ETFA*, 2018, pp. 1–8.
 - [108] A. Scholz, C. Hildebrandt, and A. Fay, “Functional modelling in production engineering workflows,” in *IEEE International Conference on Automation Science and Engineering*, 2018, vol. 2017-Augus, pp. 695–700.
 - [109] S. Schröck, F. Zimmer, A. Fay, and T. Jäger, “Systematic reuse of interdisciplinary components supported by engineering relations,” in *IFAC-PapersOnLine*, 2015, vol. 28, no. 3, pp. 1545–1552.
 - [110] K. Thramboulidis, “The 3+1 SysML View-Model in Model Integrated Mechatronics,” *J. Softw. Eng. Appl.*, vol. 03, no. 02, pp. 109–118, Apr. 2010.
 - [111] K. Thramboulidis, “Overcoming Mechatronic Design Challenges: the 3+1 SysML-view Model,” *J. Comput. Sci. Technol.*, vol. 1, no. 1, pp. 6–14, 2013.
 - [112] Object Management Group (OMG), *OMG Systems Modeling Language (OMG SysML™)*, Version 1.4, no. 1.4. 2015, p. 320.
 - [113] K. Kernschmidt, S. Feldmann, and B. Vogel-Heuser, “A model-based framework for increasing the interdisciplinary design of mechatronic production systems,” *J. Eng. Des.*, vol. 29, no. 11, pp. 617–643, Nov. 2018.
 - [114] K. E. T. Kernschmidt, “Interdisciplinary structural modeling of mechatronic production systems using SysML4Mechatronics, Ph.D. Thesis,” Technische Universität München, 2019.
 - [115] H. Li, M. Sollfrank, M. Zou, D. Ryashentseva, M. Seitz, and B. Vogel-Heuser, “Consistent automated production systems modeling in a multi-disciplinary engineering workflow,” in *Proceedings: IECON 2018 - 44th Annual Conference of the IEEE Industrial Electronics Society*, 2018, pp. 2971–2978.
 - [116] A. Finkelstein, G. Spanoudakis, and D. Till, “Managing interference,” in *Joint proceedings of the second international software architecture workshop (ISAW-2) and international workshop on multiple perspectives in software development (Viewpoints '96) on SIGSOFT '96 workshops -*, 1996, pp. 172–174.
 - [117] H. Abid, P. Pernelle, D. Noterman, J. P. Campagne, and C. Ben Amar, “SysML approach for the integration of mechatronics system within PLM systems,” *Int. J. Comput. Integr. Manuf.*, vol. 28, no. 9, pp. 972–987, Sep. 2015.
 - [118] M. Abramovici, Y. Aidi, and H. B. Dang, “Knowledge-Based Lifecycle Management Approach for Product Service Systems (PSS),” 2013, pp. 239–248.
 - [119] M. Abramovici, J. C. Göbel, and H. B. Dang, “Semantic data management for the development and continuous reconfiguration of smart products and systems,” *CIRP Ann. - Manuf. Technol.*, vol. 65, no. 1, pp. 185–188, 2016.
 - [120] R. Harrison, D. Vera, and B. Ahmad, “Engineering Methods and Tools for Cyber-Physical Automation Systems,” *Proc. IEEE*, vol. 104, no. 5, pp. 973–985, May 2016.
 - [121] S. Konstantinov, M. Ahmad, K. Ananthanarayan, and R. Harrison, “The Cyber-physical E-machine Manufacturing System: Virtual Engineering for Complete Lifecycle Support,” in *Procedia CIRP*, 2017, vol. 63, pp. 119–124.

- [122] E. Lindskog, J. Berglund, J. Vallhagen, and B. Johansson, "Layout Planning and Geometry Analysis Using 3D Laser Scanning in Production System Redesign," in *Procedia CIRP*, 2016, vol. 44, pp. 126–131.
- [123] E. Lindskog, J. Vallhagen, and B. Johansson, "Production system redesign using realistic visualisation," *Int. J. Prod. Res.*, vol. 55, no. 3, pp. 858–869, Feb. 2017.
- [124] G. Erdős, T. Nakano, and J. Váncza, "Adapting CAD models of complex engineering objects to measured point cloud data," *CIRP Ann.*, vol. 63, no. 1, pp. 157–160, 2014.
- [125] G. Erdős, T. Nakano, G. Horváth, Y. Nonaka, and J. Váncza, "Recognition of complex engineering objects from large-scale point clouds," *CIRP Ann.*, vol. 64, no. 1, pp. 165–168, 2015.
- [126] F. Biesinger, D. Meike, B. Kras, and M. Weyrich, "A Case Study for a Digital Twin of Body-in-White Production Systems General Concept for Automated Updating of Planning Projects in the Digital Factory," in *2018 IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2018, vol. 2018-Sept, pp. 19–26.
- [127] H. Zipper and C. Diedrich, "Echtzeit-Prozessmonitoring auf Basis standardisierter Simulationsmodelle und Anlagenbeschreibungen," in *AUTOMATION Congress, VDI-Berichte Nr. 2330*, 2018, pp. 1131–1141.
- [128] G. Koltun, F. Mäurer, A. Knoll, E. Trunzer, and B. Vogel-Heuser, "Information retrieval from redlined circuit diagrams and its model-based representation for automated engineering," in *Proceedings: IECON 2018 - 44th Annual Conference of the IEEE Industrial Electronics Society*, 2018, pp. 3114–3119.
- [129] J. Li, J. Berglund, F. Auris, A. Hanna, J. Vallhagen, and K. Åkesson, "Evaluation of Photogrammetry for Use in Industrial Production Systems," in *IEEE International Conference on Automation Science and Engineering*, 2018, vol. 2018-Augus, pp. 414–420.
- [130] H. Son, C. Kim, and C. Kim, "3D reconstruction of as-built industrial instrumentation models from laser-scan data and a 3D CAD database based on prior knowledge," *Autom. Constr.*, vol. 49, pp. 193–200, 2015.
- [131] R. Stark, H. Grosser, and P. Müller, "Product analysis automation for digital MRO based on intelligent 3D data acquisition," *CIRP Ann. - Manuf. Technol.*, vol. 62, no. 1, pp. 123–126, 2013.
- [132] H. Son, C. Kim, and C. Kim, "Fully Automated As-Built 3D Pipeline Extraction Method from Laser-Scanned Data Based on Curvature Computation," *J. Comput. Civ. Eng.*, vol. 29, no. 4, 2014.
- [133] E. Paquet, M. Rioux, A. Murching, T. Naveen, and Ali Tabatabai, "Description of shape information for 2-D and 3-D objects," *Signal Process. Image Commun.*, vol. 16, no. 1–2, pp. 103–122, Sep. 2000.
- [134] "PRONETA User Manual: PROFINET Network Analyzer-Configuration and Diagnostic Tool." [Online]. Available: https://support.industry.siemens.com/cs/attachments/67460624/67460624_PRONETA_Documentation_V2_6_en.pdf. [Accessed: 22-Feb-2020].
- [135] "Easy PROFINET implementation." [Online]. Available: <https://web.archive.org/web/20150402183456/http://w3app.siemens.com/mcms/infocenter/dokumentencenter/sc/ic/Documentsu20Brochures/E20001-A24-M116-X-7600.pdf>. [Accessed: 22-Feb-2020].
- [136] F. Biesinger, D. Meike, B. Kraß, and M. Weyrich, "A digital twin for production planning based on cyber-physical systems: A Case Study for a Cyber-Physical System-Based Creation of a Digital Twin," *Procedia CIRP*, vol. 79, pp. 355–360, 2019.
- [137] S. Stüb, A. Strahilov, and C. Diedrich, "Behaviour simulation for Virtual Commissioning using co-simulation," in *IEEE International Conference on Emerging Technologies and Factory Automation, ETFA*, 2015, vol. 2015-October.
- [138] M. Eigner, R. Zafirov, and T. Baudisch, "Information transfer from electrical design to

- simulation models in Modelica for virtual commissioning,” in *Proceedings of NordDesign*, 2012.
- [139] S. Biffl, R. Mordinyi, H. Steininger, and D. Winkler, “Integrationsplattform für anlagenmodellorientiertes Engineering,” in *Handbuch Industrie 4.0 Bd.2*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2017, pp. 189–212.
- [140] J. Kiefer, L. Ollinger, and M. Bergert, “Virtuelle Inbetriebnahme - Standardisierte Verhaltensmodellierung mechatronischer Betriebsmittel im automobilen Karosserierohbau,” *VDI Berichte*, no. 2067, pp. 129–132, 2009.
- [141] “Neue Wege zum Digitalen Zwilling durch mechatronisches Anlagen-Engineering.” [Online]. Available: https://www.siemens.de/Digital-Factory/download/EventDocs/1_Digitaler_Zwilling_durch_mechatronisches_Anlagen-Engineering.pdf. [Accessed: 26-Aug-2019].
- [142] DIN EN 81346-1, *Deutsches Institut für Normung: Industrielle Systeme, Anlagen und Ausrüstungen und Industrieprodukte - Strukturierungsprinzipien und Referenzkennzeichnung - Teil 1: Allgemeine Regeln*. Berlin: Beuth Verlag, 2010.
- [143] DIN EN 81346-2, *Deutsches Institut für Normung: Industrielle Systeme, Anlagen und Ausrüstungen und Industrieprodukte - Strukturierungsprinzipien und Referenzkennzeichnung - Teil 2: Klassifizierung von Objekten und Kennbuchstaben von Klasse*. Berlin: Beuth Verlag, 2010.
- [144] S. Feldmann and B. Vogel-Heuser, “Änderungsszenarien in der Automatisierungstechnik– Herausforderungen und interdisziplinäre Auswirkungen,” *Eng. von der Anforderung bis zum Betr.*, vol. 3, p. 95, 2013.
- [145] M. B. Younis and G. Frey, “Visualization of PLC programs using XML,” in *Proceedings of the American Control Conference*, 2004, vol. 4, pp. 3082–3087.
- [146] International Electrotechnical Commission, “IEC 61131-10 – Programmable controllers – Part 10: PLC open XML exchange format,” 2019.
- [147] M. B. Younis and G. Frey, “Formalization of PLC programs to sustain reliability,” in *2004 IEEE Conference on Robotics, Automation and Mechatronics*, 2004, pp. 613–618.
- [148] J. Chen, T. Wang, R. Abbey, and J. Pingnot, “A distributed decision tree algorithm and its implementation on big data platforms,” in *Proceedings - 3rd IEEE International Conference on Data Science and Advanced Analytics, DSAA 2016*, 2016, pp. 752–761.
- [149] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, vol. 27, no. 2. New York, NY: Springer New York, 2009.
- [150] S. R. Safavian and D. Landgrebe, “A Survey of Decision Tree Classifier Methodology,” *IEEE Trans. Syst. Man Cybern.*, vol. 21, no. 3, pp. 660–674, 1991.
- [151] K. North, *Wissensorientierte Unternehmensführung*. Wiesbaden: Gabler Verlag, 2011.
- [152] G. Tekli, R. Chbeir, and J. Fayolle, “A visual programming language for XML manipulation,” *J. Vis. Lang. Comput.*, vol. 24, no. 2, pp. 110–135, Apr. 2013.
- [153] T. Jarratt, J. Clarkson, and C. Eckert, “Engineering change,” in *Design process improvement*, London: Springer London, 2005, pp. 262–285.
- [154] B. Hamraz, N. H. M. Caldwell, and P. J. Clarkson, “A Holistic Categorization Framework for Literature on Engineering Change Management,” *Syst. Eng.*, vol. 16, no. 4, pp. 473–505, Dec. 2013.
- [155] T. A. W. Jarratt, C. M. Eckert, N. H. M. Caldwell, and P. J. Clarkson, “Engineering change: An overview and perspective on the literature,” *Res. Eng. Des.*, vol. 22, no. 2, pp. 103–124, 2011.
- [156] VDA (Verbund der Automobilindustrie) 4965, “Engineering Change Management Reference Process covering ECM Recommendation V2.0 Engineering Change Management Reference Process,” 2009.
- [157] Shu-Hsien Liao, “Expert system methodologies and applications—a decade review from 1995 to 2004,” *Expert Syst. Appl.*, vol. 28, no. 1, pp. 93–103, Jan. 2005.
- [158] L. Merkel, C. Berger, C. Schultz, S. Braunreuther, and G. Reinhart, “Application-specific

- design of assistance systems for manual work in production,” in *2017 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, 2017, vol. 2017-Decem, pp. 1189–1193.
- [159] F. Arevalo, T. Nguyen, and A. Schwung, “Assistance system for a bulk good system based on information fusion,” in *2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2017, pp. 1–8.
- [160] “TIA Portal Openness: Introduction and Demo Application.” [Online]. Available: https://support.industry.siemens.com/cs/attachments/108716692/108716692_TIA_Openness_GettingStartedAndDemo_V15_1_en.pdf. [Accessed: 22-Feb-2020].
- [161] S. Overhage and K. Turowski, “Serviceorientierte Architekturen — Konzept und methodische Herausforderungen,” in *Service-orientierte Architekturen*, Wiesbaden: Gabler Verlag, 2007, pp. 3–17.
- [162] M. Ghermezi, “Entwurf und Realisierung eines digitalen Zwillings einer automatisierten Fertigungszelle auf Basis einer Integrationsplattform, Masterarbeit 3004,” Institut für Automatisierungstechnik und Softwaresysteme, Universität Stuttgart, 2018.
- [163] T. Bauernhansl, “Automotive industry without conveyer belt and cycle – research campus ARENA2036,” 2015, pp. 1145–1154.
- [164] D. Braun, “Entwicklung eines intelligenten Lagers zur Evaluierung der Ankerpunktmethode in einem flexiblen Produktionssystem, Masterarbeit 3039,” Institut für Automatisierungstechnik und Softwaresysteme, Universität Stuttgart, 2019.
- [165] DIN EN 62714-1, *Deutsches Institut für Normung: Industrielle Systeme, Datenaustauschformat für Planungsdaten industrieller Automatisierungssysteme - Automation markup language - Teil 1: Architektur und allgemeine Festlegungen*. Berlin: Beuth Verlag, 2015.
- [166] J. Kiefer, T. Baer, and H. Bley, “Mechatronic-oriented Engineering of Manufacturing Systems Taking the Example of the Body Shop,” *13th CIRP Int. Conf. Life Cycle Eng.*, pp. 681–686, 2006.