HELMUT SCHMIDT UNIVERSITÄT

Universität der Bundeswehr Hamburg

Engineering agentenbasierter Automatisierung für Verteilnetz-Management-Systeme

Von der Fakultät für Maschinenbau
der Helmut-Schmidt-Universität / Universität der Bundeswehr Hamburg
zur Erlangung des akademischen Grades eines Doktor-Ingenieurs (Dr.-Ing.)
genehmigte

DISSERTATION

von Tobias Linnenberg

aus Gladbeck

Hamburg 2019

Gutachter	Univ.-Prof. Dr.-Ing. Alexander Fay Univ.-Prof. Dr.-Ing. Georg Frey
Tag der mündlichen Prüfung:	04.12.2019

Vorwort

Die vorliegende Arbeit entstand während meiner Zeit als wissenschaftlicher Mitarbeiter an der Professur für Automatisierungstechnik der Helmut-Schmidt-Universität / Universität der Bundeswehr Hamburg.

Da eine wissenschaftliche Arbeit nie das Werk einer einzelnen Person ist, möchte ich mich im Folgenden bei allen Menschen bedanken, die mich während der Entstehung meiner Dissertation unterstützt haben.

An erster Stelle möchte ich mich bei meinem Doktorvater, Herrn Professor Dr.-Ing. Alexander Fay, bedanken. Er ermöglichte mir durch die Mitarbeit an seiner Professur tiefe Einblicke in Lehre, Forschung und Industrie. Durch seine kontinuierliche Motivation, Forderung und Förderung hat er maßgeblich zum Gelingen dieser Arbeit beigetragen.

Ebenso bedanke ich mich für die Anfertigung des Zweitgutachtens bei Herrn Professor Dr.-Ing. Georg Frey. Für die Übernahme des Prüfungsvorsitzes gilt mein Dank Herrn Professor Dr.-Ing. Jens Wulfsberg.

Meinen ehemaligen Kollegen am Institut und auch den „Externen“danke ich für die freundschaftliche Unterstützung. Neben der fachlichen Zusammenarbeit werde ich mich an viele gemeinsame Unternehmungen in Hamburg erinnern.

Namentlich herauszustellen sind im Besonderen die Herren Christian Derksen, Sebastian Törsleff und Erik Wassermann, mit welchen ich im Rahmen des Projekts Agent.HyGrid viele Auf- und Abs erleben durfte und welche mich mit viel Geduld und gutem Rat durch die Erstellung dieser Arbeit begleiteten.

Meiner Frau Christine und meinen Eltern gilt mein besonderer Dank. Ohne eure bedingungslose Unterstützung und euren Rückhalt wäre dieses Projekt nicht möglich gewesen. Ihr habt mich immer wieder motiviert am Ball zu bleiben und mir ausreichend Raum und Zeit gegeben meine Arbeit abzuschließen.

Innsbruck, im Dezember 2019

Inhaltsverzeichnis

Vorwort **I**

Abbildungsverzeichnis **V**

Tabellenverzeichnis **VII**

Abkürzungsverzeichnis **VIII**

1. Einleitung **1**
1.1. Motivation und Rahmenbedingungen 1
1.2. Zielsetzung und Aufbau der Arbeit 3

2. Automation der Energieleittechnik heute **5**
2.1. Struktur der Energienetze 5
2.1.1. Elektrische Netze 5
2.1.2. Gasnetze 7
2.1.3. Wärmenetze 7
2.2. Kommunikationsprotokolle und -architekturen 8
2.2.1. IEC TR 62357 – Die Referenzarchitektur des IEC 10
2.2.2. IEC 60870 – Fernwirktechnische Kommunikationsprotokolle 11
2.2.3. IEC 61850 – Datenmodell und Kommunikationsprotokolle 12
2.2.4. IEC 61970 und IEC 61968 – Das Common Information Model 15
2.2.5. IEC 61968 – Das Interface Reference Model (IRM) 16
2.3. Sicherheit 17
2.3.1. Organisatorische Sicherheit 17
2.3.2. Funktionale Sicherheit 22
2.3.3. Kommunikationssicherheit 23
2.3.4. Smart-Meter-Gateway-Kommunikation 24
2.4. Evaluation der Kommunikationsprotokolle und Sicherheitsstandards 28
2.4.1. Kriterien 28
2.4.2. Analyse 30
2.5. Zwischenfazit 33

3. Agenten in der Energieleittechnik **34**
3.1. Systemstrukturen und Architekturen 35
3.1.1. Zentral-hierarchische Führungsstruktur 36
3.1.2. Dezentral-hierarchische Führungsstruktur 36
3.1.3. Dezentrale Führungsstruktur 37
3.2. Vorteile und Herausforderungen 37
3.3. Literatur über Agenten in der Energieleittechnik 38

3.4. Kompatibilität agentenbasierter Steuerungen 40
3.5. Zwischenfazit 41

4. Komponenten des Energie-Agenten 42
4.1. Definition und Struktur des Energie-Agenten 42
4.2. Kosten & Optionsmodell 48
4.3. Durchgängiges Engineering durch Wiederverwendung 54
4.4. Zwischenfazit 57

5. Vorgehensmodelle und Entwicklungsmethoden für Software-Agenten 59
5.1. Grundlagen der Softwareentwicklung - Projekt- und Softwarelebenszyklus . . 59
5.2. Grundlagen der Softwareentwicklung – Vorgehensmodelle 59
5.2.1. Beispiel eines Vorgehensmodells 59
5.2.2. Metamodell und Begriffsdefinition 60
5.2.3. Grundlegende Konzepte der Softwareentwicklung 61
5.2.4. Klassische Modelle 62
5.2.5. Agile Modelle 63
5.3. Spezifische Vorgehensmodelle 65
5.3.1. V-Modell XT 65
5.3.2. Scrum 68
5.3.3. Extreme Programming 71
5.4. Agentenspezifische Ansätze des Software Engineerings 76
5.4.1. Gaia 77
5.4.2. Tropos 79
5.4.3. O-MaSE 83
5.5. Evaluation der Vorgehensmodelle 85
5.5.1. Kriterien 85
5.5.2. Analyse 87
5.6. Zwischenfazit 94

6. Engineering agentenbasierter Automatisierungstechnik für Verteilnetze 96
6.1. Artefakte 96
6.1.1. Organisatorische Artefakte 96
6.1.2. Technische Artefakte 103
6.2. Phasen 108
6.2.1. Initiale Anforderungsermittlung 112
6.2.2. Vorplanung 112
6.2.3. Anforderungsermittlung 113
6.2.4. Basisplanung 114
6.2.5. Ausführungsplanung 114
6.2.6. Implementierung 115
6.2.7. Simulation 116
6.2.8. Testbedanwendung 117
6.2.9. Debriefing 118
6.2.10. Deployment 118
6.2.11. Lifecycle Management 119

6.3. Aktivitäten . . . 120
6.3.1. Aktivitäten zur Erarbeitung organisatorischer Artefakte . . . 120
6.3.2. Aktivitäten zur Erarbeitung technischer Artefakte . . . 123
6.4. Rollen . . . 126
6.4.1. Kunde . . . 126
6.4.2. Projektleiter . . . 127
6.4.3. Koordinator . . . 127
6.4.4. Entwickler . . . 128
6.4.5. Experte . . . 128
6.5. Werkzeuge . . . 128

7. Anwendung und Evaluation 130
7.1. Fallstudien . . . 130
7.1.1. Vorstellung der Anwendungsfälle . . . 131
7.1.2. Ergebnisse der Fallstudien . . . 136
7.1.3. Bewertung . . . 139
7.2. Experteninterviews . . . 142
7.2.1. Methodik . . . 142
7.2.2. Ergebnisse . . . 142
7.2.3. Bewertung . . . 144
7.3. Zwischenfazit . . . 146

8. Zusammenfassung, kritische Bewertung und Ausblick 147
8.1. Zusammenfassung . . . 147
8.2. Kritische Bewertung . . . 149
8.3. Ausblick . . . 150

A. Interviewleitfaden zur Evaluation des Ansatzes 152

B. Qualitative Ergebnisse der Experteninterviews 154

Literaturverzeichnis 159

Abbildungsverzeichnis

1.1. Entwicklung der Stromerzeugung aus erneuerbaren Energien [UBA18] 2

2.1. Referenzarchitektur nach IEC 62357 [IEC14] 10
2.2. Struktur der Telegramm Daten nach IEC 60870-5-103 11
2.3. Aufbau eines logischen Knoten nach IEC 61850-7-1 13
2.4. Abbildungsmöglichkeiten der IEC 61850 auf weitere Protokolle nach [USD+13] 15
2.5. Schnittstellen des Interface Reference Model [Usl09] 16
2.6. Phasen der Sicherheitskonzeption nach BSI 100-2 [BSI08b] 21
2.7. Einsatzumgebung des Smart-Meter-Gateway[BSI13b] 27
2.8. Bewertung der Standards in Hinblick auf ihre Relevanz [Hel16] 30
2.9. Bewertung der Standards in Hinblick auf ihren Aufwand [Hel16] 31
2.10. Bewertung der Qualitätsmerkmale in Hinblick auf ihre Relevanz [Hel16] . . . 32

3.1. (a) Zentral-hierarchische, (b) dezentral-hierarchische und (c) dezentrale Führungsstruktur, nach [BDL+13] 36
3.2. Anzahl an Publikationen zum Thema „„Software Agent“Energy“auf Google Scholar von 1997 bis 2017 39

4.1. Struktur, Schnittstellen und Prozesse eines Energie-Agenten des Reifegrades 3 und höher 48
4.2. Hauptelemente des Energieoptionsmodells und des Zustandsmodells 50
4.3. Wurzelklasse eines technischen Systems als Grundlage eines Energie-Agenten 52
4.4. Modell der Systemzustände des Energie-Agenten 53
4.5. Kostenmodell des Energie-Agenten 53
4.6. Exemplarische Darstellung eines austauschbaren *IO-Behaviours* für die Nutzung des Energie-Agenten in unterschiedlichen Entwicklungsphasen 57

5.1. Bestandteile eines Vorgehensmodells nach [BK13] 60
5.2. Metamodell der Softwareentwicklung, in Anlehnung an [NS99] 61
5.3. Wasserfallmodell nach W. Royce [Roy70] 63
5.4. Ergebnis der Studie zur Frage: Welche(r) der folgenden Entwicklungsprozesse findet/n bei Ihnen Anwendung? [KL14] 66
5.5. Ablaufmodell Scrum nach [Luk15] 69
5.6. Rollenmodell Scrum nach [Luk15] 70
5.7. Ablaufmodell Extreme Programming nach [Luk15] 72
5.8. Ablaufmodell Release Planning nach [Luk15] 73
5.9. Ablaufmodell Development nach [Luk15] 74
5.10. Artefaktmodell Gaia, nach [WJ05] 78

6.1. Vergleich der organisatorischen Artefaktmodelle von XP, SCRUM und 2DECS 97
6.2. Beispiel einer Storycard in 2DECS 99

6.3. Beispiel eines *Product Backlogs* 101
6.4. Vergleich der technischen Artefaktmodelle von GAIA, TROPOS, O-MaSE und 2DECS 103
6.5. Beispiel eines 2DECS-Interaktionsmodells 106
6.6. Beispiel eines Agentenmodells 107
6.7. Technisches Phasenmodell 110
6.8. Organisatorisches Phasenmodell 111
6.9. Ein- und Ausgangsartefakte der Initialen Anforderungsermittlung 112
6.10. Ein- und Ausgangsartefakte der Vorplanung 113
6.11. Ein- und Ausgangsartefakte der Anforderungsermittlung 113
6.12. Ein- und Ausgangsartefakte der Basisplanung 114
6.13. Ein- und Ausgangsartefakte der Ausführungsplanung 115
6.14. Ein- und Ausgangsartefakte der Implementierung 116
6.15. Ein- und Ausgangsartefakte der Simulation 117
6.16. Ein- und Ausgangsartefakte der Testbedanwendung 118
6.17. Ein- und Ausgangsartefakte des Debriefings 118
6.18. Ein- und Ausgangsartefakte des Deployments 119
6.19. Ein- und Ausgangsartefakte des Lifecycle Managements 119
6.20. Vergleich der Rollenmodelle von SCRUM, XP, V-Modell und 2DECS 127

7.1. Schematische Struktur des Agent.HyGrid - 2016 Projektes 132
7.2. Schematische Struktur des Agent.HyGrid - 2018 Projektes 133
7.3. Schematische Struktur des OEEX Mobile Power Projektes 135
7.4. Auszug aus der OEEX Mobile Power Sprint Planung in Form eines Sprint Backlogs 137
7.5. Auszug des Agent.HyGrid - 2018 Zielmodells 137
7.6. Auszug des Agent.HyGrid - 2018 Rollenmodells 138

Tabellenverzeichnis

2.1. Wichtige Standards der Smart-Grid-Kommunikation [Hel16] 9
2.2. Übersicht zum Aufbau der IEC 61850 12
2.3. Wichtige Standards der organisatorischen IT-Sicherheit [Hel16] 18
2.4. Übersicht der Schutzmaßnahmen nach IEC 62351 25
2.5. Standards aus dem Bereich Smart-Metering [Hel16] 26
2.6. Skala zur Bewertung der Relevanz [Hel16] 28
2.7. Skala zur Bewertung des notwendigen Aufwands [Hel16] 29
2.8. Skala zur Bewertung der Qualitätsmerkmale [Hel16] 29

4.1. Reifegrade der Energie-Agenten 46

5.1. Relevanz agentenorientierter Entwicklungsansätze, Stand: 26.07.2017, Quelle: scholar.google.com 76
5.2. Übersicht der Phasenabdeckung 88
5.3. Übersicht der Kundenorientierung 89
5.4. Übersicht der Prozesssteuerung 91
5.5. Übersicht des Detaillierungsgrades 92
5.6. Übersicht der Anpassbarkeit 93
5.7. Übersicht der Berücksichtigung gesetzlicher Anforderungen 94
5.8. Übersicht der Entwicklungsprozesse 94

6.1. Auszug der Sammlung an *User Stories* für das OEEX Mobile Battery Projekt 98
6.2. Beispiel von Acceptance Test Requirements 99
6.3. Organisatorische Aktivitäten des 2DECS Ansatzes und ihre Zusammenhänge 120
6.4. Technische Aktivitäten des 2DECS Ansatzes und ihre Zusammenhänge . . . 124

7.1. Übersicht der in den Fallstudien evaluierten Phasen 131
7.2. Ergebnisse der Fallstudien 140
7.3. Symbole zur Einordnung der befragten Experten 143
7.4. Quantitative Ergebnisse der Experteninterviews 143

A.1. Leitfaden für Experteninterviews zur Evaluation des 2DECS Konzeptes . . . 153

Abkürzungsverzeichnis

2DECS	Development approach for DEcentralized Control Systems
ACL	Agent Communication Language
ACSI	Abstract Communication Service Interface
API	Application Programming Interface
BDEW	Bundesverband der Energie- und Wasserwirtschaft e.V.
BHKW	Blockheizkraftwerk
BMWi	Bundesministerium für Wirtschaft und Energie
BDI	Belief, Desire, Intention
BSI	Bundesamt für Sicherheit in der Informationstechnik
CIM	Common Interface Modell
CDPSM	Common Distribution Power System Modell
DKE	Deutschen Komission Elektrotechnik Elektronik Informationstechnik
DMS	Distribution Management System
ENTSO-E	European Network of Transmission System Operators for Electricity
EOM	Energie- & Optionsmodell
EPRI	Electric Power Research Institute
EVU	Energieversorgungsunternehmen
FR	Foundational Requirement
GID	Generic Interface Definition
GMA	Gesellschaft Mess- und Automatisierungstechnik im VDI und VDE
GOOSE	Generic Object Oriented Substation Event
GUI	Graphical User Interface
HAN	Home Area Network
IA	Initiale Anforderungserhebung
ICCP	Inter Control Center Protocol
IDE	Integrated Development Environment
IEC	International Electrotechnical Commission
IED	Intelligentes Elektronisches Gerät (Device)
IoT	Internet of Things
IP	Internet Protocol
IRM	Interface Reference Model
ISA	International Society of Automation
ISMS	Informationssicherheits-Managementsystem
JADE	Java Agent DEvelopment Framework
LMN	Local Metrological Network
MAS	multi agent system
MMS	Manufacturing Messaging Specification
O-MaSe	Organization-based Multiagent Software Engineering
OEEX	Open Energy Exchange

OSI	Open Systems Interconnection Model
PDCA	Plan Do Check Ackt
RE	Requirement Enhancement
SCADA	Supervisory Control and Data Acquisition
SCL	System Configuration Language
SGAM	Smart Grid Architecture Models
SL	Security Level
SIL	Safety Integrity Level
SMGW	Smart Meter Gateway
TASE.2	Telecontrol Application Service Element Number 2
TCP	Transmission Control Protocol
UML	Unified Modeling Language
VDE	Verband der Elektrotechnik
VDI	Verein Deutscher Ingenieure
VDMA	Verband Deutscher Maschinen- und Anlagenbau e.V.
VPN	Virtual Private Network
WA	Weitere Anforderungserhebung
WAN	Wide Area Network

1. Einleitung

1.1. Motivation und Rahmenbedingungen

Bedingt durch den Klimawandel, sich verändernde geopolitische und gesellschaftliche Interessen sowie den daraus resultierenden Verschiebungen im Energiemarkt stehen die Energiesysteme weltweit vor fundamentalen Umbrüchen. Deutschland versucht in diesem wachsenden Markt eine Vorreiterrolle einzunehmen und sich an der technologischen Spitze zu etablieren. Das 2001 erlassene *Erneuerbare-Energien-Gesetz* läutete den nun als Energiewende bekannten Wandel des Strommixes von nuklearen und fossilen Brennstoffen hin zu erneuerbaren Energien ein. Zwar umfasst die Energiewende auch die Sektoren Wärme und Mobilität, doch fokussiert sich die öffentliche Wahrnehmung und damit auch ein Großteil der Anstrengungen auf die Erzeugung und den Verbrauch von Elektrizität. Durch eine anhaltende Förderung konnte der Anteil des aus Wasserkraft, Sonne, Wind und Biomasse gewonnenen Stromes seit dem Jahr 2000 von 36,2 TWh auf 217,9 TWh in 2017 gesteigert werden. Abbildung 1.1 zeigt die steigende Entwicklung der erzeugten Strommenge, welche lediglich durch die Wirtschaftskrise in 2008 und eine geänderte Förderlandschaft in 2016 kurzzeitig abgeflacht wurde. Die ersten Quartale des Jahres 2018 sind von einem weiteren starken Wachstum mit neuen Erzeugungsrekorden der erneuerbaren Energien geprägt. Aufgrund der Volatilität und der Dislozierung der Kraftwerke bedeutet dieses Wachstum jedoch auch eine erhöhte Anzahl an regelnden Eingriffen in die Erzeugungs-, Übertragungs- und Verteilkapazitäten. So müssen bei Überlastungen Kraftwerke heruntergeregelt oder Stromflüsse umgeleitet werden. Im entgegengesetzten Fall, dem Ausbleiben erwarteter Produktion, müssen alternative Quellen aktiviert werden. Bis heute lassen sich diese Eingriffe noch manuell aus zentralen Leitwarten bewerkstelligen. Setzt sich das Wachstum der Erzeugungskapazitäten und der damit gewonnenen Menge an Strom jedoch in der beobachteten Form fort, werden automatisierte Leitsysteme in Form sogenannter *Smart-Grids* unumgänglicher Teil der Infrastruktur werden. Um die Netzverfügbarkeit weiterhin auf dem aktuellen Niveau halten zu können überdenken Energieversorger, -Erzeuger und Netzbetreiber ihre aktuellen Führungsstrategien und verschieben ihren Blick vom bisher sehr Hardware-geprägten Netzausbau hin zu mehr Intelligenz im Netz. Eine Reihe an wissenschaftlich fundierten Beispielen für diese modernen Netzmanagementlösungen existiert bereits heute. Auch wenn diese einen unterschiedlichen Grad an Dezentralisierung aufweisen, teilen sie doch dieselben Ziele. Diese liegen in einer technischen Optimierung mit dem Ziel der Synchronisation von Erzeugung und Verbrauch als auch einer finanziellen Optimierung, einer Erhöhung des Nutzerkomforts oder einer Mischung mehrerer dieser Ziele. Die bekanntesten Beispiele sind die *PowerMatcher Suite* [KWK+05], welche vor kurzem durch den niederländischen Netzbetreiber *Alliander* erworben wurde und nun mit dem Ziel einer kommerziellen Nutzung weiterentwickelt wird. Zur selben Zeit wie *PowerMatcher* wurde durch Prof. Lehnhoff der *DEZENT* Ansatz entwickelt [WLHK06], welcher den Weg für das *E-Energy* Projekt *eTelligence* [GGFJ11] ebnete. Beide Konzepte nutzen einen marktbasierten Ansatz um die Erzeugung und den Verbrauch von Elektrizität zu nivel-

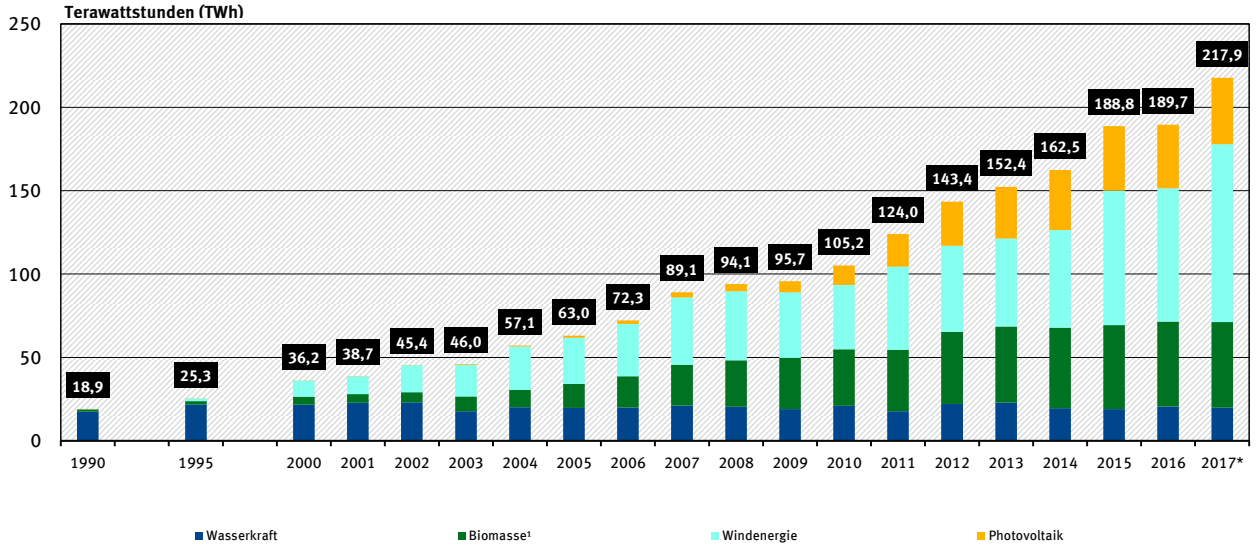

Abbildung 1.1.: Entwicklung der Stromerzeugung aus erneuerbaren Energien [UBA18]

lieren und somit die Integration Erneuerbarer Energien in Nieder- und Mittelspannungsebene zu erleichtern. Als grundlegende Technologie dienen ihnen dabei Software-Agenten.
Ein Agentensystem lässt sich als ein System aus einer Vielzahl verteilter, abgrenzbarer Software- und ggf. Hardware-Einheiten, den sogenannten Agenten, mit definierten Zielen beschreiben. Diese Agenten zeichnen sich insbesondere durch ihre Fähigkeit zur Kommunikation und Kollaboration aus. Ein individueller Agent besitzt üblicherweise nur beschränkte Informationen und kann somit lediglich Teilprobleme lösen. Hierfür besitzt ein Agent folgende individuelle Eigenschaften und Fähigkeiten, welche in der VDI/VDE-Richtlinie 2653 beschrieben sind [VDI10] :

- **Autonomie** – Kontrolle über seine Handlungen und Zustände;
- **Interaktion** – Fähigkeit, mit anderen Agenten zu interagieren;
- **Kapselung** – Möglichkeit des Agenten, Zugriff auf innere Teile wie seinen Zustand, sein Verhalten, seine Strategie und seine Ziele zu reglementieren;
- **Persistenz** – Fähigkeit, seinen inneren Zustand zu wahren;
- **Reaktivität** – Möglichkeit, auf seine Umwelt zu reagieren;
- **Proaktivität** – Potential, aus eigenem Antrieb Initiative zu ergreifen;
- **Zielorientierung** – Fähigkeit, sein Verhalten an vorgegebenen Zielen auszurichten.

Bedingt durch ihre verteilte Architektur sind Agentensysteme insbesondere zur Lösung komplexer, verteilter Probleme geeignet. Hierfür wird das Problem im Sinne des „Teile und Hersche“-Prinzips in Teilprobleme zerlegt. Die Agenten können dann so gestaltet werden, dass jeder einzelne Agent nur ein solches Teilproblem lösen muss. Im Kollektiv sind sie jedoch in der Lage das Gesamtproblem zu bewältigen.
Durch diese Herangehensweise eignen sich Agentensysteme zur Steuerung einer großen Zahl

verteilter und zusammenhängender, sich gegenseitg beeinflussender Anlagen. Sie bieten in diesem Kontext zudem eine erhöhte Ausfallsicherheit, da das Versagen einer einzelnen Komponente – bei entsprechendem Design des Systems – durch die verbleibenden Agenten automatisch kompensiert werden kann. Somit sind Agentensysteme zur Regelung von Elektrizitätserzeugern, welche auf erneuerbaren Energien basieren, besonders geeignet. Sie können eine große Menge an verteilten Anlagen überwachen und steuern, die Volatilität der Energiequellen kompensieren, einen Ausgleich zwischen Erzeugung und Verbrauch schaffen und darüber hinaus die Toleranz der Energienetzte gegenüber Ausfällen einzelner Komponenten erhöhen.
Da wissenschaftliche Ansätze moderner Netzmanagementlösungen bisher lediglich als Demonstratoren in gut überwachten und ausreichend definierten Testumgebungen entwickelt wurden, waren die Anforderungen an die eingesetzte Hard- und Software jedoch niedriger als es die gültigen Spezifikationen der Netzbetreiber vorsehen. Zwecks Implementierung der Testprojekte wurden daher durch die Netzbetreiber und Behörden entsprechende Ausnahmen erteilt, um so nicht zertifizierte Software in der öffentlichen Infrastruktur einsetzen zu können. Aus diesem Grund ist einer der Hauptvorwürfe, welcher von anderen Netzbetreibern geltend gemacht wird, der, dass die Art und Weise, wie die Software entwickelt wurde, als äußerst unstrukturiert und inkonsistent wahrgenommen wird. Die Implikationen, welche sich in Hinblick auf die Sicherheit und Interoperabilität des so entwickelten Systems daraus ergeben, werden daher regelmäßig als Argumente für konservativere Führungsstrategien vorgebracht. Diesen Argumenten kann man jedoch begegnen, indem im Rahmen der Entwicklung von Steuerungssoftware für Netzamanagementlösungen ein strukturierter Entwicklungsprozess genutzt wird, welcher sowohl die Spezifika des Energiesektors als auch die der agentenbasierten Steuerungslösungen berücksichtigt. Bindet man dabei den Kunden und alle anderen relevanten Stakeholder adäquat ein, kann man ihnen so Kontrolle und Sicherheit geben. Dabei sind sowohl Normen, also durch einen klar definierten Prozess mit möglichst breiter Beteiligung entstandene Dokumente, welche Anforderungen an Dienstleistungen, Produkte oder Vorgehensweisen definieren, als auch Standards, welche eher einer akzeptierten, als einer definierten Vorgehensweise entsprechen, zu berücksichtigen
Zusammenfassend lässt sich feststellen, das ein strukturierter Entwicklungsprozess für agentenbasierte Steuerungssysteme in der Energietechnik fehlt, welcher es erlaubt sich im Projektverlauf ändernde Anforderungen zu berücksichtigen und als Grundlage hierfür eine starke Integration des Kunden ermöglicht.

1.2. Zielsetzung und Aufbau der Arbeit

Aufbauend auf den vorangegangenen Ausführungen ist es das primäre Ziel dieser Arbeit einen Entwicklungsprozess für Agentensysteme zu definieren, welcher die beschriebenen Probleme adressiert und idealerweise eliminiert. Der Prozess soll dabei sowohl eine Integration des Kunden als auch eine Wahrung der im Bereich der Energiesysteme äußerst umfassenden Standards und Normen im Zuge der Entwicklung zentraler oder dezentraler Energieleittechnik auf Basis von Software-Agenten erlauben. Der Entwicklungsprozess soll dabei sowohl auf Ebene des Projektmanagements als auch des konkreten Entwurfs von Agentensystemen ausreichend umfassend definiert sein. Hiedruch soll eine durchgängige Nutzung der erarbeiteten Artefakte ermöglicht werden. Als disparate Anforderung muss jedoch auch eine hinreichende Flexibilität zwecks Anpassbarkeit an die Spezifika individueller Entwicklungsprojekte gege-

ben sein.
Die folgenden Problemfelder wurden daher als Schwerpunkte der Arbeit identifiziert:

- **Identifikation relevanter Standards und Normen in der Verteilnetzautomatisierung** - Hierbei soll ein besonderer Fokus auf Sicherheitsaspekte und aktuelle sowie zukunftsträchtige Kommunikationslösungen gelegt werden. Dies soll die Nutzbarkeit des Ansatzes in aktuellen und zukünftigen Entwicklungsprojekten ermöglichen.
- **Charakterisierung von Agenten in der Energiedomäne** - Die Spezifika von Agenten in Energiesystemen sollen beschrieben werden.
- **Definition eines abstrakten Energie-Agenten** - Es soll ein abstraktes Softwarekonzept geschaffen werden, welches die Beschreibung und Modellierung von Agenten in unterschiedlichen Energienetzen der Zukunft ermöglicht.
- **Untersuchung existierender Methoden** - Es sollen sowohl abstrakte als auch agentspezifische Methoden für die Softwareentwicklung untersucht werden. Hierfür sollen verbreitete Ansätze ausgewählt und auf Basis einheitlicher Kriterien evaluiert werden.
- **Synthese und Erweiterung geeigneter Ansätze** - Aufbauend auf den in den vorangegangenen Schritten definierten Anforderungen soll ein Entwicklungsansatz definiert werden, welcher bekannte Methoden integriert und, wo notwendig, erweitert.
- **Evaluation des Ansatzes** - Der Ansatz soll sowohl praktisch in Umsetzungsprojekten als auch in Experteninterviews evaluiert werden um seine Anwendbarkeit und den erzielten Nutzen zu untersuchen.

Im weiteren Verlauf der Arbeit wird in Kapitel 2 „Automation der Energieleittechnik heute“zunächst der aktuelle Aufbau der Energienetze und ihrer Leittechnik skizziert um so das grundlegende Problemfeld darzustellen. Im Anschluss werden Normen und Anforderungen aus den Bereichen Kommunikation und Sicherheit in Energienetzen dargestellt und in Hinblick auf ihre Relevanz analysiert. Das darauf folgende Kapitel 3 „Agenten in der Energieleittechnik“stellt agentenbasierte Energieleitsysteme und deren Grundlagen dar. Aufbauend darauf wird in Kapitel 4 „Komponenten des Energie-Agenten“ein abstraktes Agentenmodell eingeführt, welches den Anforderungen unterschiedlicher Energiesysteme gerecht wird. Die dem Modell zugrundeliegenden Konzepte werden erläutert und anhand von Beispielen dargestellt. Kapitel 5 „Vorgehensmodelle und Entwicklungsmethoden für Software-Agenten“gibt im Anschluss einen Überblick über Ansätze zum generellen Management von IT-Projekten wie auch zum konkreten Design und zur Implementierung von Software-Agenten. Die dargestellten Vorgehensmodelle werden am Ende des Kapitels 5 auf ihre Anwendbarkeit in der Energiedomäne und in Hinblick auf die Implementierung eines Agentensystems bewertet. Unter Berücksichtigung der in den vorhergehenden Kapiteln erarbeiteten Erkenntnisse und Konzepte werden in Kapitel 6 „Engineering agentenbasierter Energieleittechnik“das Artefakt-, Phasen-, Aktivitäts- und Rollenmodell des im Zuge dieser Arbeit entwickelten *Development approach for DEcentralised Control Systems* (2DECS) dargestellt. Die anschließende Evaluation in Kapitel 7 „Anwendung und Evaluation“baut auf drei realen Anwendungsfällen auf, welche zunächst eingeführt werden. Die praktische Anwendung in diesen Projekten wird dargestellt und durch Experteninterviews ergänzt. Das abschließende Kapitel 8 „Zusammenfassung, kritische Bewertung und Ausblick“zieht ein Fazit der Arbeit und zeigt potentielle Erweiterungen und weitere Anwendungsfelder des Ansatzes.

2. Automation der Energieleittechnik heute

Die wesentliche Grundlage dezentraler Entscheidungsfindung ist die Kommunikation und Verhandlung zwischen den beteiligten Entitäten. Zum besseren Verständnis der Entitäten sollen im Folgenden zunächst die unterschiedlichen Energienetze und ihre Spezifika kurz vorgestellt werden. Im Anschluss werden sowohl Kommunikationsprotokolle als auch Methoden der Absicherung von Überwachungs- und Steuerungs-Prozessen im Bereich der Energiewirtschaft dargestellt. Die in diesem Kontext relevanten Standards und Normen mit den darin enthaltenen Anforderungen werden beschrieben und evaluiert. Dieses Vorgehen soll die Grundlage für eine spätere Berücksichtigung der Methoden und Protokolle im Rahmen des Engineeringprozesses legen.

2.1. Struktur der Energienetze

Die folgende Beschreibung der drei primären Energienetzarten, namentlich für Elektrizität, Gas und Wärme, soll dem grundlegenden Verständnis der Problemstellung sowie der Schnittstellen und Ausgleichsmöglichkeiten der drei Energieformen beziehungsweise Träger untereinander dienen.

2.1.1. Elektrische Netze

Das zentraleuropäische Verbundnetz (vormals das „UCTE-Verbundnetz“[1]) gliedert sich in zwei Ebenen: Das Übertragungsnetz sowie das Verteilnetz.
Das Transport- beziehungsweise Übertragungsnetz dient dem Transport großer Strommengen über hohe Distanzen. Um die dabei auftretenden elektrischen Verluste gering zu halten, wird es im Bereich der Hoch- und Höchstspannung betrieben. Diese liegt bei 220 kV, 380 kV oder 400 kV und wird in der Regel direkt von Großerzeugern eingespeist. Das Übertragungsnetz erlaubt den Austausch elektrischer Energie zwischen entfernten Regionen und dient damit der Stabilisierung des Gesamtnetzes im Falle lokaler Störungen. Dieser Teil des Stromnetzes ist eng vermascht und wird gut überwacht.
Das lose vermaschte Verteilnetz wird im Bereich der Mittelspannung (10 kV bis 20 kV) betrieben. Dies erlaubt eine effiziente Verteilung der elektrischen Energie in lokalen Netzen, welche nur wenige Kilometer umfassen. Auf dieser Netzebene finden sich vermehrt dezentrale und volatile Stromerzeuger. Die hierdurch auftretenden Phänomene wie zum Beispiel Spannungsbandverletzungen können aufgrund einer geringen Verbreitung von Mess-, Steuer- und Regelungstechnik in dieser Netzebene nicht oder nur unzureichend erfasst und beeinflusst werden.

[1] Union for the Co-ordination of Transmission of Electricity - UCTE

Die ebenfalls dem Verteilnetz zugeordnete Niederspannungsebene (230 V oder 400 V) umfasst die letzten Meter der elektrischen Verteilung hin zu Privat- und kleinen Industriekunden. Auf dieser Ebene speisen vor allem private Solaranlagen und Blockheizkraftwerke ein. Die Verbindung zwischen den einzelnen Netzebenen bilden Transformatoren in leittechnisch gut bestückten Umspannwerken zwischen Hoch- und Mittelspannungsebene sowie mit rudimentären Schutzmechanismen versehene Transformatorenstationen zwischen der Mittel- und Niederspannungsebene.

Elektrische Energieerzeuger und Speicher

Auch heute noch bilden thermische Großkraftwerke die Grundlage der Energieversorgung und liefern laut Agora Energiewende [JSBG17] auch im Jahr 2018 noch 70% der elektrischen Energie in Deutschland. Es werden dabei unterschiedliche Kraftwerkstypen anhand der eingesetzten Primärenergieträger sowie der zeitlichen Verfügbarkeit differenziert. Eine geringe Regelung im engen Rahmen lassen dabei die nuklearen und fossilen Grundlastkraftwerke zu, welche typischerweise auf der Hoch- und Höchstspannungsebene einspeisen. Gasturbinen- und Pumpspeicherkraftwerke werden hingegen als ausgleichende Elemente zu Energiewandlungsanlagen genutzt, welche auf erneuerbaren Energien basieren. Sie lassen eine flexible Regelung über ein weites Spektrum zu und werden daher häufig in der Nähe von Wind- und Solarparks installiert. Letztere sind aufgrund ihrer Abhängigkeit von fluktuierenden Energiequellen nicht grundlastfähig. Biogas- und Wasserkraftwerke zählen zu den invariablen regenerativen Energieumwandlungsanlagen.
Um die Aufnahmekapazität erneuerbarer Energien zu erhöhen, werden elektrische Netze zunehmend mit modernen Speichertechnologien wie Lithium-Ionen-Akkus und Wasserstoffspeichern ausgestattet. Auch die Umwandlung von elektrischer Energie in Erdgas wird bereits im wirtschaftlichen Einsatz erprobt [VDI15]. Als kurzfristige Energiespeicher werden darüber hinaus schnell reagierende Schwungmassenspeicher und sogenannte „Super-Caps“- also sehr große Kondensatoren - in einzelnen Pilotprojekten untersucht.
Energieerzeuger stellen wirkungsvolle Ressourcen zur Netzstabilisierung bereit. So können diese - in den oben beschriebenen Grenzen - Wirk- und Blind-Leistung bereitstellen und reduzieren. Anlagen mit hohen rotierenden Massen, wie Dampf-, Gas- oder Windturbinen, stellen darüber hinaus eine wichtige Dämpfungskomponente des Netzes dar.
Die Regelung von Erzeugungsanlagen ist auch heute noch das Mittel der Wahl zur lokalen Spannungsregelung auf Hoch- und Mittelspannungsebene sowie der übergreifenden Frequenzregelung. Bedingt durch einen steigenden Anteil verteilter Elektrizitätserzeugung auf Basis erneuerbarer Energien sowie die steigende Attraktivität von elektrischen Kleinspeichern findet die Elektrizitätserzeugung jedoch zunehmend auf der Mittel- und Nieder-Spannungsebene statt. Gleichzeitig sinkt die elektrische Leistung der einzelnen Erzeugungsanlagen. Damit geht eine sinkende Auswirkung der einzelnen Anlage auf das Spannungsniveau und die Netzfrequenz einher. Es werden daher, wie in Kapitel 3 beschrieben, in Wissenschaft und Praxis Methoden untersucht wie diese Erzeugungsanlagen im Falle von Spannungs- oder Frequenz-Schwankungen eingesetzt werden können.

Elektrische Energieverbraucher

Wie in [BDL+13] beschrieben, ist „Ein elektrischer Energieverbraucher [...] ein Gerät oder ein System von Geräten, das einen Teil der elektrischen Energie in eine für den Endnutzer

nützliche Energieform umwandelt. Der Energiebedarf eines elektrischen Energieverbrauchers wird durch die auszuführenden Systemfunktionen bestimmt. Dieser ist abhängig von verschiedenen Einflüssen, aus denen sich mögliche Freiheitsgrade für die Optimierung zur Laufzeit ergeben."
Neben gegenseitigen Wechselwirkungen elektrischer Verbraucher, wie zum Beispiel der gegenseitigen Beeinflussung eines Backofens und einer Klimaanlage, stellen insbesondere die Anforderung der Nutzer an die ständige Verfügbarkeit ihrer Geräte die größte Einschränkung in Hinblick auf eine netzdienliche Nutzung im Sinne des Lastmanagements dar. Die Einschränkung von Nutzerkomfort kann im privaten Bereich aufgrund des geringen Preises für elektrische Energie und einem geringen Automatisierungsgrad der elektrischen Anlagen nur abstrakt, also in Form eines guten Gewissens oder spielerischer Ansätze, kompensiert werden. Im industriellen Kontext hingegen, wo eine automatisierte Anlagenführung bereits möglich ist und der Kostenfaktor Energie einen höheren Stellenwert besitzt, hat sich das sogenannte „Demand Side Management", welches eine Verschiebung von elektrischem Verbrauch in Phasen geringerer Netzauslastung vorsieht, bereits etabliert und bietet den Anwendern ökonomische Vorteile.

2.1.2. Gasnetze

Analog der Klassifikation elektrischer Energienetze gliedert sich die Gasversorgung in Übertragungs- und Verteilnetze. Die Übertragungsnetze werden mit einem Druck von bis zu 100 bar betrieben, welcher durch Verdichterstationen im Abstand von bis zu 200 km gewährleistet wird. Die verwendeten Rohrdurchmesser variieren und nehmen Maße von bis zu 140 cm an. Die angestrebte Strömungsgeschwindigkeit liegt dabei bei circa 10 m/s.
Das über 470.000 km umfassende Verteilnetz für Erdgas wird mit einem Druck zwischen vier bar und 16 bar betrieben und stellt eine großflächige Versorgung der kommunalen Gasnetze sicher.
Da die Speicherung von Gas, im Gegensatz zur Speicherung elektrischer Energie, nur mit geringen Verlusten verbunden ist, werden zur Pufferung von Spitzenlasten sowie als strategische Reserve knapp 20 Milliarden Kubikmeter Arbeitsgas in 46 Untertage-Gasspeichern vorgehalten. Dies entspricht in etwa 28% des deutschen Erdgasverbrauchs von 2014.
Die Abrechnung von Erdgas erfolgt volumetrisch und ist durchgängig von analoger Zählertechnik geprägt. Parallel hierzu ist die Überwachung und der Betrieb von Erdgasnetzen noch weitgehend manuell und entspricht dem Stand der Technik aus den 1980er Jahren. Eine Digitalisierung der Zählerinfrastruktur und Netzleittechnik ist nur im Rahmen einzelner Pilot- und Forschungs-Projekte sowie auf der Verteilnetzebene zu beobachten.

2.1.3. Wärmenetze

Man unterscheidet Wärmenetze in Nah- und Fernwärmenetze, wobei keine eindeutigen Differenzierungsmerkmale der beiden Kategorien existieren. In beiden Fällen handelt es sich um isolierte Rohrleitungssysteme, in welchen mittels eines Mediums Wärme transportiert und gespeichert werden kann. Nahwärmenetze nutzen hierfür meist Wasser, in älteren Fernwärmenetzen wird zum Teil noch Dampf eingesetzt. Die Übergabe der Wärme an den Verbraucher findet in der Regel mittels eines Wärmeübertragers statt, welche das lokale System des Verbrauchers von dem Netz hydraulisch trennt. Im Falle von Großverbrauchern kann hiervon abgesehen und diese direkt mit dem Fernwärmenetz gekoppelt werden.

Die Vorlauftemperaturen der Nah- und Fernwärmenetze liegen zwischen 200 Grad Celsius bei dampfbasierten Systemen und 50 Grad Celsius bei Niedrigtemperatursystemen. Die Mehrzahl der Systeme wird jedoch bei unter 100 Grad Celsius betrieben, um somit auf der einen Seite den Druck der Rohrleitungen, welcher durch die Verdunstung und Kondensation des Wassers beeinflusst wird, und die damit einhergehende Komplexität und Störanfälligkeit des Systems zu reduzieren, und auf der anderen Seite ausreichend hohe Temperaturen für die Warmwasserbereitung zur Verfügung zu stellen. Bei dieser muss die Warmwassertemperatur zur Vermeidung von Legionellen stets über 60 Grad Celsius liegen. In Niedertemperatursystemen wird dies über lokale Warmwasserwärmepumpen realisiert.
Die Erfassung und Abrechnung der gelieferten Wärmemengen erfolgt mittlerweile durchgängig mit digitalen Mess- und Zählsystemen. Diese sind mit Kommunikationseinrichtungen verbunden, welche eine automatisierte Abrechnung durch den Netzbetreiber ermöglichen.
Wärmenetze bieten ein gutes Potential, um Last- und Erzeugungsschwankungen im Bereich der Elektrizitäts- und Gasnetze auszugleichen. Mögliche Wärmequellen sind beispielsweise solarthermische-Anlagen, überschüssige Wärmeenergie aus Rechenzentren oder anderen energieintensiven Anwendungen wie stromgesteuerten und dezentralen Wärmepumpen. Die gute Speicherbarkeit von Wärme erlaubt es die Erzeugungskapazität auf einen Durchschnittsbetrieb auszulegen und einen Mehrbedarf in Starklastphasen aus Speichern zu bedienen. Bei der Übertragung und Speicherung von Wärme müssen jedoch stets die dabei anfallenden Verluste berücksichtigt werden. Selbes gillt in Hinblick auf die Umwandlung von Wärme in andere nutzbare Energieformen, welche in der Regel mit hohen Verlusten verbunden ist. Somit können die oben genannten Verlustenergien nie im vollen Umfang zurückggewonnen werden.

2.2. Kommunikationsprotokolle und -architekturen

Basis funktionierender Kommunikation ist eine gemeinsame Syntax sowie ein gemeinsames Verständnis der transportierten Inhalte. Diese Merkmale sowie weitere Anforderungen an einen erfolgreichen Informationsaustausch, sind für diverse Anwendungsfälle im Energiesektor in einer Vielzahl von Normen, Standards und Empfehlungen spezifiziert.
Auf Basis der *Smart-Grid Standardization Roadmap* der *International Electrotechnical Commission* (IEC) [IEC10], der *Normungsroadmap E-Energy* des deutschen *Verbandes der Elektrotechnik* (VDE) [VDE13] sowie der *Untersuchung des Normungsumfeldes zum BMWi Förderschwerpunkt E-Energy* des *Bundesministeriums für Wirtschaft und Energie* (BMWi) aus dem Jahr 2009 [Usl09] lassen sich die Trends nationaler wie auch internationaler Normungsorganisationen in Hinblick auf kommunikations- und sicherheitstechnische Anforderungen ableiten. Neben diesen drei Kernpublikationen existieren eine Vielzahl weiterer Initiativen und daraus erwachsener Veröffentlichungen einzelner Akteure aus Wirtschaft und Politik, mit dem Ziel, nationale und internationale Normen und Standards zu schaffen. Zwecks Reduktion der damit einhergehenden Komplexität und der besonderen Bedeutung der drei eingangs erwähnten Veröffentlichungen wird im Folgenden primär auf diese Bezug genommen.
Im Rahmen des Projektes *Agent.HyGrid* [DLNLS15] wurde, angeleitet durch den Verfasser dieser Dissertation, eine Übersicht relevanter Normen und Standards erarbeitet, welche Anforderungen an das Kommunikationsverhalten von Netzwerkkomponenten definieren. Diese sind in Tabelle 2.1 [Hel16] zusammengefasst und werden im Folgenden dargelegt.

Standard	Titel	Kurzbeschreibung
IEC TR 62357	Power system control and associated communications - Reference architecture for object models, services and protocols	Der Technische Report beschreibt alle existierenden Objektmodelle, Dienste und Protokolle des Technical Committee 57 (TC 57) der IEC und dokumentiert ihre Abhängigkeiten. Er liefert damit eine Übersicht aller vom IEC TC 57 verantworteten Standards.
IEC 61970 / DIN EN 61970	Schnittstelle der Anwendungsprotokolle von Energieverwaltungssystemen (EMS-API)	Beschreibt eine Programmierschnittstelle für Energiemanagementsysteme; Common Information Model - CIM (Definition von Objekten und Datenaustauschformaten) zur Nachbildung der Übertragung, Verteilung und Erzeugung elektrischer Energie. EMS-API = Energy Management System – Application Programming Interface
IEC 61968 / DIN EN 61968	Integration von Anwendungen in Anlagen der Elektrizitätsversorgung – Systemschnittstellen für Netzführung	Informationsaustausch zwischen Verteilnetzen; Schnittstellendefinition für die Integration von IT-Systemen im Verteilnetzmanagementumfeld. Beschreibt das sog. Interface Reference Model (IRM) für das CIM.
IEC 61850 / DIN EN 61850	Kommunikationsnetze und -systeme für die Automatisierung in der elektrischen Energieversorgung	Definiert ein Übertragungsprotokoll für Schaltanlagen in der Mittel- und Hochspannungsebene sowie die dazugehörige Schnittstelle und Konfigurationssprache.
IEC 61400-25	Communications for monitoring and control of wind power plants	Kann als Teil der IEC 61850 aufgefasst werden, der sich explizit der Realisierung der Kommunikation bei WKA widmet.
IEC 60870 / DIN EN 60870	Fernwirkeinrichtungen und – systeme, Teil 5: Übertragungsprotokolle, Teil 6: Fernwirkprotokolle	Die Normenreihe IEC 60870-5 definiert Übertragungsprotokolle, mit denen erreicht werden konnte, dass Geräte und Anlagen der Fernwirk- und Stationsleittechnik verschiedener Hersteller miteinander kommunizieren können. Die IEC 60870-6 beschreibt die Kommunikation (TASE.2 & MMS) zwischen Netzleitstellen.

Tabelle 2.1.: Wichtige Standards der Smart-Grid-Kommunikation [Hel16]

2.2.1. IEC TR 62357 – Die Referenzarchitektur des IEC

Die Architektur des *Technical Report (TR) 62357*, welcher durch das in der IEC für die Normung von Smart-Grid-Aktivitäten zuständige *Technical Committee 57* entwickelt wurde, gliedert sich in drei relevante Bereiche, welche in Abbildung 2.1 in einer vereinfachten Form dargestellt werden. Zum einen werden im oberen Teil der Grafik abstrakte Datenmodelle wie z.B. das Common Information Model (z.B. IEC 61970 - siehe Abschnitt 2.2.4) und im unteren Teil der Grafik technische Kommunikations- und Automatisierungsstandards für Feldgeräte und Stationen (z.B. IEC 60870 - Abschnitt 2.2.2 und IEC 61850 - Abschnitt 2.2.3) definiert. Zum anderen wird auf der rechten Seite der Grafik die Ebenen-übergreifende Normenreihe IEC 62351 (siehe Abschnitt 2.3.3) eingeführt, welche die Grundlagen und Spezifika der Cybersicherheit definiert. [ABBU12]
Die hier genannten Standards und Normen werden im Folgenden weiter diskutiert.

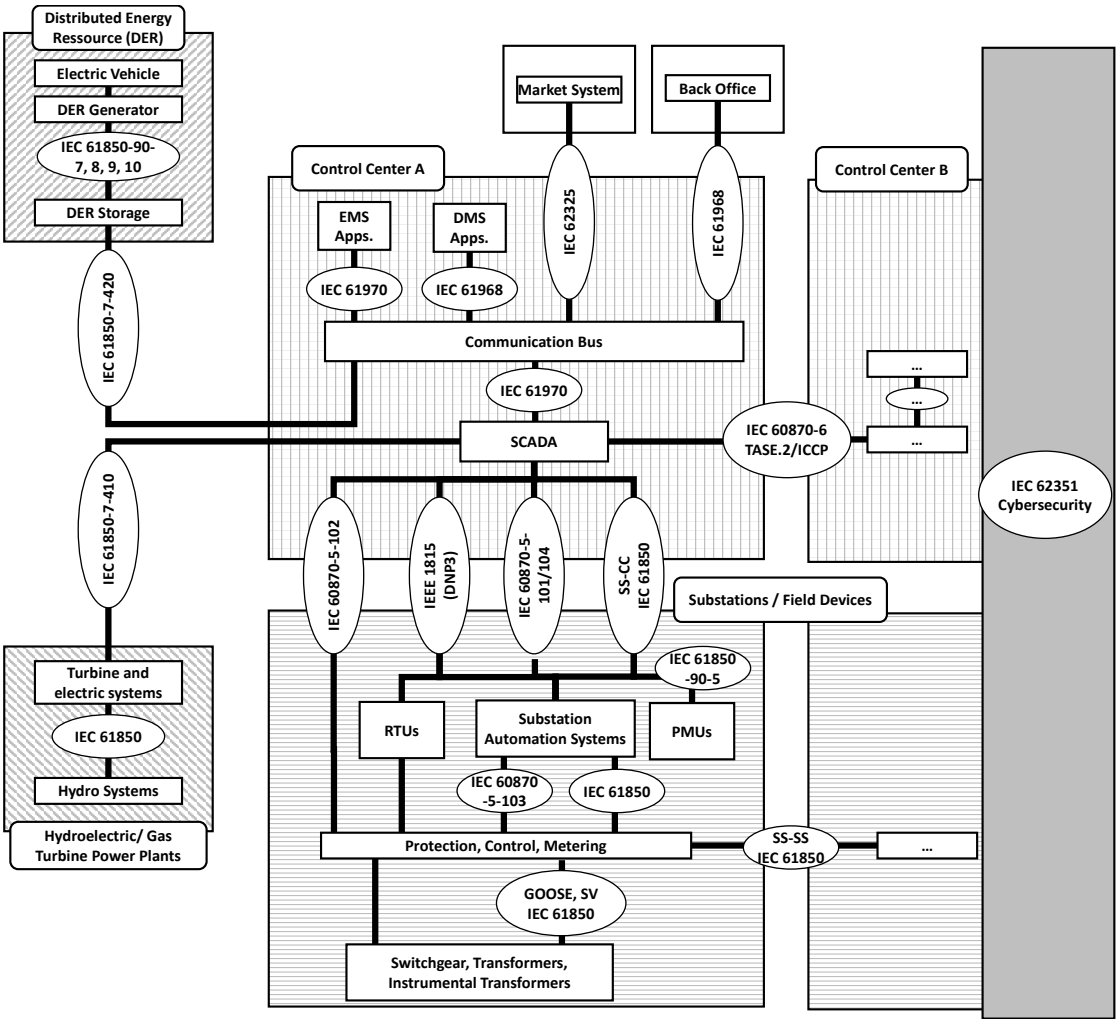

Abbildung 2.1.: Referenzarchitektur nach IEC 62357 [IEC14]

2.2.2. IEC 60870 – Fernwirktechnische Kommunikationsprotokolle

Anfang der 1990er Jahre wurde aufgrund der umfangreichen Erneuerung der Netzleittechnik auf den oberen Spannungsebenen und der damit einhergehenden Einführung digitaler Komponenten die Standardisierung der herstellerspezifischen, proprietären Protokolle unabdingbar. Es wurde daher die internationale Normenreihe IEC 60870 [Usl09] geschaffen, welche, neben der Definition von Anforderungen an fernwirktechnische Anlagen und deren Schnittstellen in den Teilen 1 bis 4, in Teil 5 auch die Übertragungsprotokolle und schlussendlich in Teil 6 die Kopplung von Leitstellen definiert. Neben den langen Investitionszyklen der Netzbetreiber machen auch die Erweiterung des Teils 5 um aktualisierte Begleitstandards für spezifische und im Rahmen der Digitalisierung der unteren Netzebenen erwachsende Szenarien die Normenreihe auch heute noch zu einem wichtigen Referenzwerk.
Die Datenübertragung erfolgt, wie in der Grundnorm IEC 60870-5 beschrieben, signalorientiert im Telegrammformat über eine serielle Schnittstelle mit einer Baudrate von 64 kbit/s. Jedes Telegramm beinhaltet in dieser Kommunikation zwischen Netzleitstellen und den untergeordneten Mess- und Fernwirkeinheiten stets nur einen Datenpunkt wie einen Alarm, Messwert oder Schaltbefehl. Diese Datenpunkte enthalten, wie in Abbildung 2.2 dargestellt, neben dem reinen Wert der Nachricht auch einen Zeitstempel, den Übertragungsgrund sowie eine dem Sender und Empfänger bekannte Informationsobjektadresse, welche der Identifikation der Nachricht dient. Die Adressierung der Objekte ist nicht standardisiert und erlaubt die Vergabe derselben Adressen an unterschiedliche Kommunikationspartner. Die hierdurch entstehenden Risiken sind erheblich [Usl09].
Um diese Risiken für schutztechnische Geräte zu mindern wurde in der IEC 60870-5-103 „Standard für die Informationsschnittstelle von Schutzeinrichtungen“eine Kennung spezifiziert, welche sich aus dem Funktionstyp des Schutzgeräts und einer vordefinierten Informationsnummer zusammensetzt. Auch wenn die Integration nicht standardisierter Funktionen durch diesen Ansatz erschwert wird, ist die gesteigerte Interoperabilität aufgrund der präzisen Gerätebeschreibung ein wichtiger Faktor, welcher zu einer hohen Verbreitung dieser Schnittstellenbeschreibung führt. [SKD+15]
Aufgrund der steigenden Bedeutung von Transmission Control Protocol / Internet Protocol

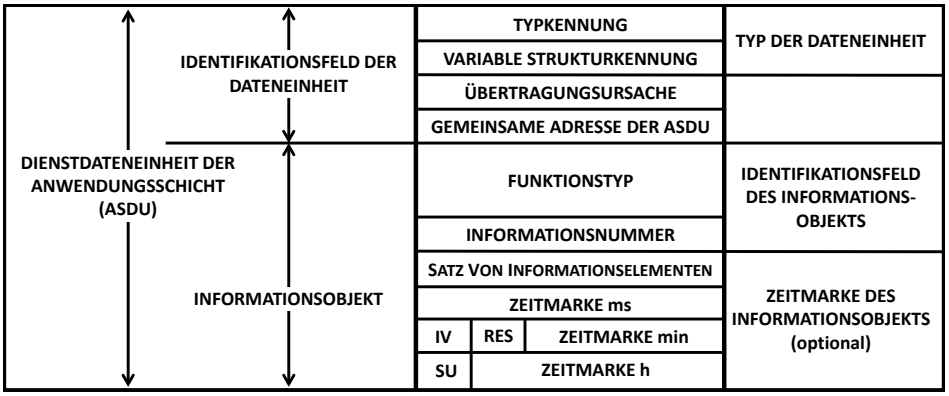

Abbildung 2.2.: Struktur der Telegramm Daten nach IEC 60870-5-103

(TCP/IP) basierten Wide Area Netzwerken (WAN) wurde die Normenreihe um den Teil 5-104 „Zugriff für IEC 60870-5-101 auf Netze mit genormten Transportprofilen“, dem sogenannten „104er-Protokoll“, erweitert. Auf Basis dieser Neuerung sowie der in der IEC 60870-5-101

Teil	Titel
Allgemein	
1	Einführung und Übersicht
2	Wörterbuch
3	Allgemeine Anforderungen
4	System- und Projektverwaltung
5	Kommunikationsanforderungen für Funktionen und Gerätemodelle
Konfiguration	
6	Sprache für die Beschreibung der Konfiguration der Geräte (SCL)
Datenmodell	
7-1	Kommunikationsprinzipien- und Modelle
7-2	Beschreibung der abstrakten Kommunikationsschnittstelle (ACSI)
Datenmodell	
7-3	Gemeinsame Datenklassen und Attributdefinitionen
7-4	Definition der logischen Knoten
7-410	Erweiterung des Modells für Wasserkraftwerke
7-420	Erweiterung des Modells für dezentrale Erzeuger
Abbildung auf reale Kommunikationsprotokolle	
8-1	Abbildung auf MMS und Ethernet sowie GOOSE und Sampled Values
9-1	Abbildung von Abtastwerten über serielle Punkt-zu-Punkt Verbindungen
9-2	Abbildung von Abtastwerten auf Ethernet

Tabelle 2.2.: Übersicht zum Aufbau der IEC 61850

„Anwendungsbezogene Norm für grundlegende Fernwirkaufgaben“definierten Interoperabilitätsliste ist es möglich Telegramme über öffentliche WANs und den TCP Port 2404 zu versenden. [BS14]
Die Kopplung von Netzleitstellen, welche in der IEC 60870-6 definiert ist, baut ebenfalls auf der WAN-basierten Kommunikation auf. Die hier definierten Protokolle [DIN15a] und Datenmodelle [DIN15b] werden in Europa unter dem Titel *Telecontrol Application Service Element Number 2* (TASE.2) und dem angelsächsischen *Inter Control Center Protocol* (ICCP) subsumiert. Die Kommunikation mit den Feldgeräten erfolgt dabei über Relais, welche die Signale aus SCADA oder Distribution Management Systemen (DMS) in TASE.2 Nachrichten übersetzen. Diese basieren auf dem in der ISO 9506 spezifizierten *Manufacturing Messaging Specification* (MMS) -Protokoll, welches ursprünglich für die Kopplung verteilter Automatisierungssysteme entwickelt wurde [Usl09]. Die definierten Protokolle umfassen dabei die im Open Systems Interconnection Model (OSI) definierten Schichten 5-7, also die reine Anwendungsebene. Es ist somit möglich den Nachrichtentransport frei zu gestalten und z.B. das oben erwähnte TCP/IP Protokoll zu nutzen [SKD+15].

2.2.3. IEC 61850 – Datenmodell und Kommunikationsprotokolle

Die bereits vor 10 Jahren für die Stationsautomatisierung eingeführte IEC 61850, welche unter dem Titel *Kommunikationsnetze und -systeme für die Automatisierung in der elektrischen Energieversorgung* firmiert, wurde an die sich wandelnden Erfordernisse eines Energiesystems im Umbruch angepasst. Dem Ziel folgend, die Interoperabilität intelligenter elektronischer Ge-

räte (IEDs) unterschiedlicher Hersteller zu steigern [ABBU12], fokussiert sich die Norm auf den Austausch von Messwerten, Statusinformationen, Schaltbefehlen und weiteren Parametern, welche zur Koordination einer großen Anzahl IEDs benötigt werden. Wichtig ist dabei, dass die spezifischen Gerätefunktionen der IEDs nicht detailliert definiert oder vorgegeben werden.
Um diese Abstraktion sicherstellen zu können, beschreibt die Norm ein XML-basiertes, objektorientiertes Datenmodell zur Modellierung technischer Anlagen in elektrischen Energienetzen sowie eine mögliche Umschlüsselung auf unterschiedliche Kommunikationsprotokolle. Neben diesen beiden Kernelementen werden weitere Anforderungen wie minimale Anforderungen an die zu nutzenden Automatisierungssysteme, eine standardisierte Konfigurationssprache sowie Konformitätstests beschrieben. Tabelle 2.2 gibt eine Übersicht über die unterschiedlichen Teile der Norm.
Im Rahmen der Modellierung werden virtuelle Repräsentanten (*logical devices*) realer Geräte geschaffen. Deren Teilfunktionen werden dann mittels genormter Kategorien in sogenannte logische Knoten (*logical nodes* - LN) unterteilt.Abschnitt 7-4 der IEC 61850 gibt eine Übersicht aller definierten Knoten und unterteilt diese in Kategorien wie zum Beispiel Messwandler, Schaltgeräte, Automatikfunktionen, Schutzfunktionen oder Überwachung und Steuerung [DIN14b]. Sollte eine gewünschte Funktion in diesem Abschnitt nicht beschreiben sein, so ist eine Erweiterung auf Basis der Abschnitte 13 und 14 möglich. Dabei sind die knotenspezifischen Eigenschaften der Geräte und ihrer Teilfunktionen in Anlehnung an Abschnitt 7-4 anhand von Attributen festzulegen. Diese bilden im Rahmen der IEC 61850 zentrale Prozessinformationen in Datenobjekten ab [Usl09].

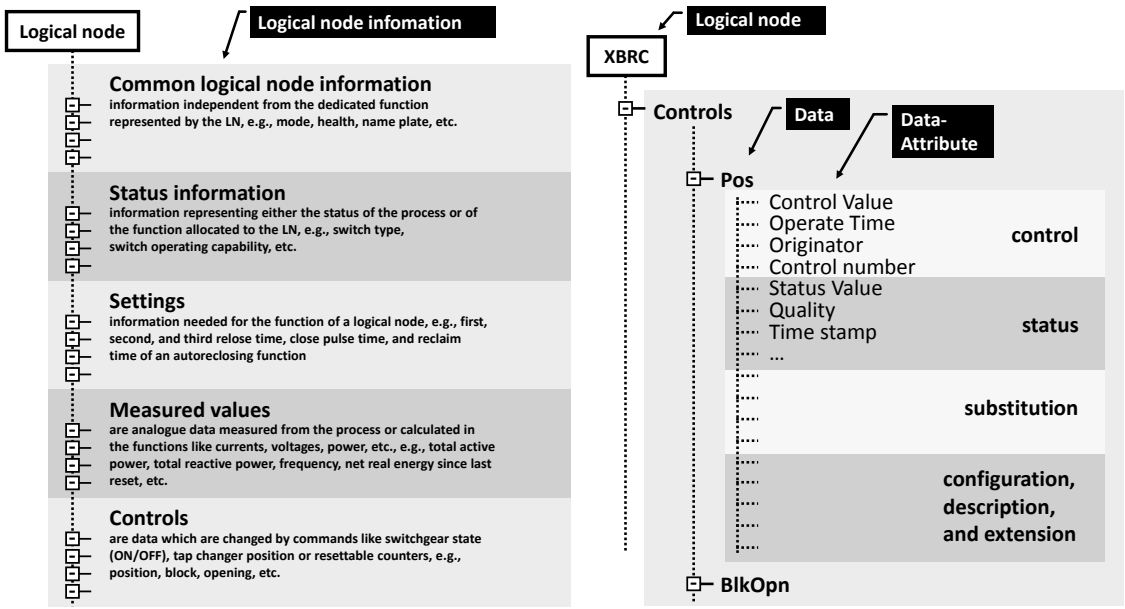

Abbildung 2.3.: Aufbau eines logischen Knoten nach IEC 61850-7-1

Abbildung 2.3 zeigt den daraus abgeleiteten Aufbau eines logischen Knotens. Neben der generischen Form auf der linken Seite zeigt die rechte Seite der Abbildung anhand des Beispiels eines Leistungsschalters (XBRC - Kategorie Schaltgeräte) den exemplarischen Aufbau der Information *Position* (Pos) aus dem Bereich *Controls*. [Usl09] beschreibt die hierfür als notwendig erachteten Attribute als:

- **Control Value** - kann die Werte „zu“ und „auf“ annehmen und beschreibt damit die eigentliche Steuerinformation
- **Operate Time** - die Zeit, zu welcher gesteuert werden soll
- **Originator** - Informationen zum Absender der Information
- **Status** - Zustandsinformationen des betrachteten Gerätes
- **Substitution** - die Möglichkeit einen Ersatzwert zu setzen
- **Configuration** - Konfigurationsparameter des Gerätes
- **Description** - eine Beschreibung des Gerätes
- **Extension** - die Erweiterungskennung des Gerätes

Als zentrale Herausforderung im Zuge der Modellierung wird die Auswahl des geeignetsten logischen Knoten gesehen, um die Funktionen des realen Gerätes im virtuellen Repräsentanten adäquat abzubilden. Vordefinierte Profile helfen dabei Ausgangswerte für Beziehungen zwischen logischen Knoten und die damit verbundenen Austauschvorgänge zu beschreiben. Die dafür geeigneten Transportprotokolle werden in Teil 7-2 mittels einer abstrakten Kommunikationsschnittstelle (*Abstract Communication Service Interface* - ACSI) generalisiert. Teil 8-1 ermöglicht dann eine Abbildung auf konkrete Implementierungen wie die Manufacturing Message Specification (MMS) zur Datenübertragung über Weitverkehrsnetze via TCP/IP, oder aber auch Generic Object Oriented Substation Event (GOOSE) und Sampled Measured Values (SV) zum lokalen *Ethernet* basierten Datenaustausch via *Multicast*. Die für den Bereich der Windkraft entwickelte und aus organisatorischen Gründen ausgegliederte IEC 61400 ist kompatibel zur IEC 61850 und enthält die Abbildung auf weitere Kommunikationsprotokolle, welche entsprechend übernommen werden können. Abbildung 2.4 zeigt das durch die IEC 61400-25-4 erweiterte Portfolio. Aufgrund des aktuellen Wandels hin zu Cloudlösungen und webbasierten Systemen stellen insbesondere die Nutzbarmachung von Webservices und die IEC 60870-5-101/104 einen wichtigen Baustein für eine zukunftssichere Implementierung agentenbasierter Systeme dar. Auch heute noch nicht existente oder beachtete Transportmedien und -protokolle können so aufgrund der Trennung des Datenmodells und Protokollebene mittels ACSI in zukünftigen Szenarien implementiert und genutzt werden [USD+13].
Die Kommunikation mittels IEC 61850 ist gegenüber den in den weiteren Abschnitten beschriebenen Standards und zugehörigen Protokollen aufgrund ihrer Objektorientierung besonders für komplexe Kommunikationsaufgaben geeignet. Wo bei signalorientierten Übertragungsverfahren lediglich Mess- und Sollwerte an spezifische Empfänger übertragen werden, sind die im Rahmen der IEC 61850 definierten *Multicast*-Mechanismen in der Lage beliebig viele Empfänger in Echtzeit zu erreichen und das, aufgrund der Entkopplung von Datenmodell und Kommunikationsprotokoll, mittels einer Vielzahl unterschiedlicher Kommunikationswege. Ergänzt wird all dies durch die in Abschnitt 6 der Norm definierte Konfigurationssprache (SCL), welche die Beschreibung einzelner Funktionen und kompletter Geräte standardisiert. Um eine spätere Plug-and-Play-Fähigkeit zu ermöglichen, wird die so beschriebene Gerätekonfiguration in den Systemen lokal vorgehalten [Koc15].

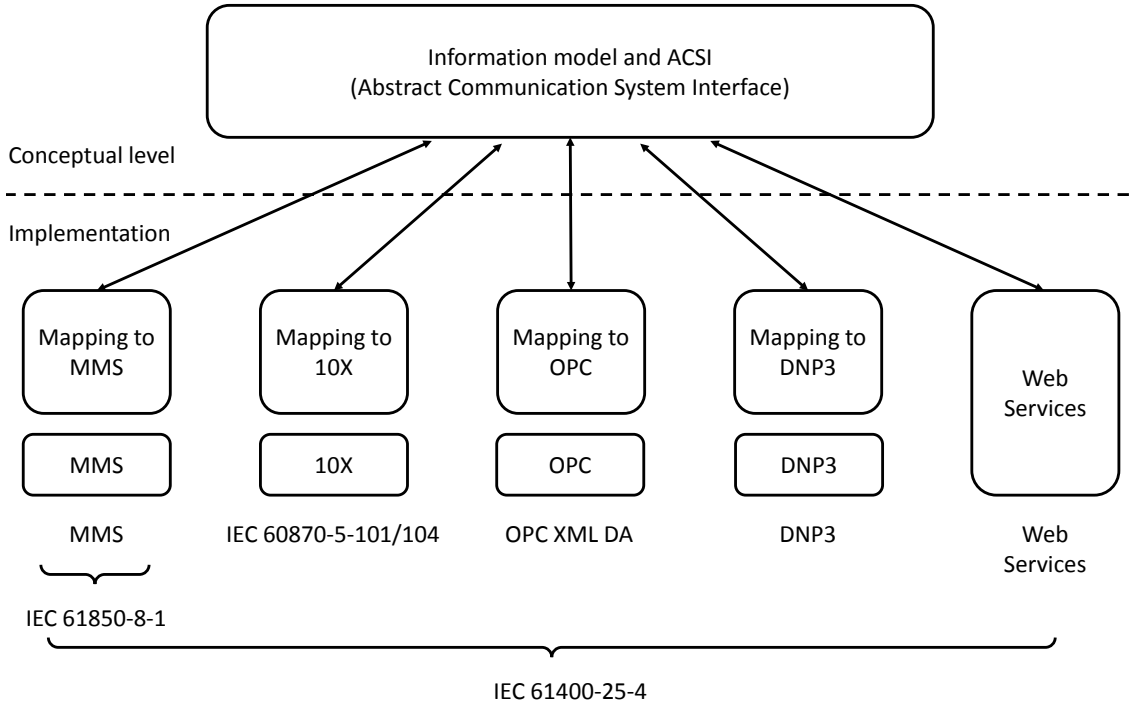

Abbildung 2.4.: Abbildungsmöglichkeiten der IEC 61850 auf weitere Protokolle nach [USD+13]

2.2.4. IEC 61970 und IEC 61968 – Das Common Information Model

Aufgrund sich mehrender Inkompatibilitäten proprietärer Datenmodelle und Systemschnittstellen in Leitsystemen der elektrischen Energieversorgung und -Übertragung entwickelte das *Electric Power Research Institute* (EPRI) Mitte der 1990er Jahre das *Common Interface Model* (CIM). In der Zwischenzeit wurde das Modell in die internationalen IEC Standards 61970/61968 überführt und als DIN EN veröffentlicht [ABBU12].
Um eine möglichst hohe Anzahl an Anwendungsfällen im Bereich der Energiewirtschaft abzubilden, wurde das CIM als Domänenontologie konzipiert und als UML-Modell in Form eines Klassendiagramms beschrieben. Es ist somit möglich unterschiedliche Anwendungsfälle wie z.B. Verteilnetztopologien oder Geschäftsprozesse damit abzubilden. Um diese in unterschiedlichen Implementierungen integrieren zu können, kann zwischen verschiedenen Serialisierungen wie XML oder RDF gewählt werden [UG07]. Parallel zur IEC 61850, erlauben sogenannte Profile eine Komplexitätsreduktion durch eine Vorauswahl relevanter Klassen und Objekte. Es werden dafür offizielle Profile wie das *CDPSM* oder das *ENTSO-E* zur Verfügung gestellt. Diese finden insbesondere im Rahmen des Austausches europäischer Topologiedaten der Verteil- und Übertragungsnetze Verwendung [USD+13]. Darüber hinaus ist es möglich individuelle Profile zu erstellen und dabei statische Zustandsdaten, wie Leitungen oder Transformatoren und auch dynamische Zustandsdaten, wie zum Beispiel Fahrpläne, abzubilden.
Die oben beschriebene Serialisierung wird durch eine Standardisierung der Objektschnittstellen unterstützt. Diese ist in der IEC 61970-401, der sogenannten *Generic Interface Definition*, definiert. Sie legt dabei fest, wie die, auf Basis einer gemeinsamen semantischen Basis beschriebenen, Daten ausgetauscht werden [USR+12]. Das hierfür wesentliche Datenmodell wird in den Standards IEC 61970-301 (CIM Basismodell), IEC 61968-11 (Erweiterung für die Verteilung) und der IEC 62325-301 (Erweiterung für Märkte) beschrieben.

2.2.5. IEC 61968 – Das Interface Reference Model (IRM)

Basierend auf dem in Abschnitt 2.2.4 beschriebenen CIM erlaubt die IEC 61968 den Entwurf einer zentralen Integrations-Plattform für Energieversorger und Netzbetreiber. Diese bietet eine einheitliche Syntax und Semantik und stärkt die Integration divergenter Anwendungen wie zum Beispiel Verteilnetzmanagementsystemen unterschiedlicher Betreiber und Hersteller. Das Interface Reference Model (IRM) beschreibt zur einfacheren Umsetzung explizite Anwendungsfälle und die hierfür notwendigen Schnittstellen. Besonderer Fokus liegt dabei auf dem Bereich Verteilnetzmanagement und umfasst im wesentlichen die „[...] Überwachung und Steuerung von Gerätschaften zur Stromversorgung, zum Prozessmanagement, zur Systemstabilität, zum Spannungsmanagement, Demand-Side-Management, Ausfallmanagement, Arbeitsmanagement, zur automatischen Datenintegration sowie Facility Management“ [DIN14c].
In Abbildung 2.5 werden die Schnittstellen der IEC 61968 grafisch dargestellt. Die für das Verteilnetzmanagement relevanten Schnittstellen sind links der gestrichelten Linie dargestellt. Hierzu zählen zum Beispiel die Netzbetriebsüberwachung, die Netzsteuerung, das Störungsmanagement, die Realzeit-Netzberechnung oder die Betriebsmittelanalyse im Teil 3, dem Bereich Network Operation (Netzbetrieb).
Die rechts der gestrichelten Linie in grau dargestellten Geschäftsfunktionen sind nicht unmittelbar am Verteilnetzmanagement beteiligt. Hierzu zählen Funktionen wie zum Beispiel der Einkauf von elektrischer Energie in Teil 10 des Standards. Aufgrund der Fokussierung auf unterschiedliche Geschäftsfunktionen ist es möglich, das IRM unabhängig von proprietären, herstellerspezifischen Lösungen zu nutzen [DIN14c].

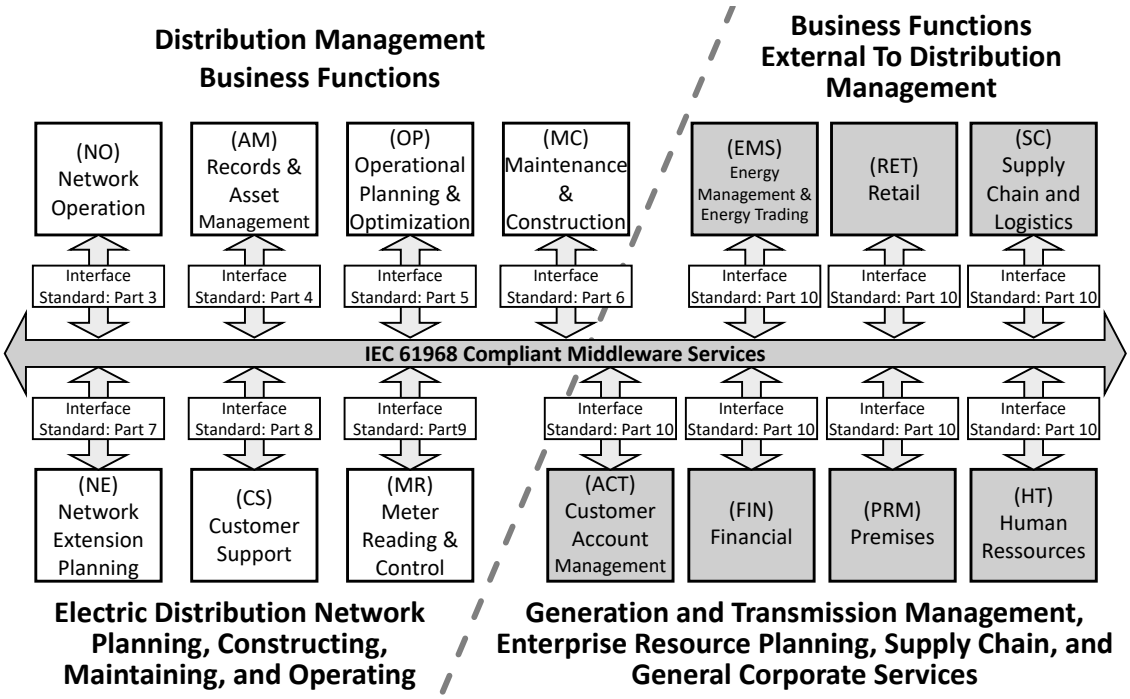

Abbildung 2.5.: Schnittstellen des Interface Reference Model [Usl09]

2.3. Sicherheit

Aufgrund der hohen Relevanz elektrischer Netze werden diese zur kritischen Infrastruktur gezählt. Die im Rahmen der Netzleitung und -überwachung eingesetzten Systeme unterliegen daher erhöhten Sicherheitsanforderungen. Neben der *Normungsroadmap IT-Sicherheit* [DIN14a], welche von der Deutschen Kommission Elektrotechnik Elektronik Informationstechnik (DKE) herausgegeben wird sowie dem vom Bundesamt für Sicherheit in der Informationstechnik (BSI) vertriebenen *ICS-Security Kompendium* [BSI13a] und der Studie *Status Quo der Security in Produktion und Automation* des Verbandes Deutscher Maschinen- und Anlagenbau e.V. (VDMA) [VDM13], existieren weitere Normen, welche Sicherheitsanforderungen an Leitsysteme im Energiebereich definieren. Tabelle 2.3 gibt einen Überblick über die im folgenden beschriebenen Standards und Normen. Hiervon ausgenommen sind die in Deutschland weniger gewichtigen Standards NERC CIP, IEEE 1686 und die VDI/VDE 2182.

2.3.1. Organisatorische Sicherheit

Grundlage sicherer IT-Systeme ist die Verankerung adäquater Instrumente im täglichen Betrieb und der Entwicklung dieser Systeme. Hierfür werden von Einrichtungen, wie dem BSI, dem Bundesverband der Energie- und Wasserwirtschaft e.V. (BDEW) und der DIN Konzepte vorgeschlagen, um diese organisatorischen Maßnahmen nachvollziehbar und nachprüfbar umzusetzen. Im Folgenden sollen diese näher ausgeführt werden.

DIN ISO / IEC 27000 - Informationssicherheits-Managementsysteme Auch wenn die DIN ISO/IEC 27000 Normenreihe nicht spezifisch für den Energiebereich entwickelt wurde, so dient sie doch als Grundlage vieler Maßnahmen in Hinblick auf die sichere Implementierung und den Betrieb von IT-Systemen im elektrischen Netzbetrieb sowie der Energieerzeugung. Im Rahmen von Ausschreibungen und der Auftragsvergabe wird durch Energieunternehmen regelmäßig eine ISO-27000-konforme Zertifizierung des Auftragnehmers gefordert. Grundsätzlich beschreibt die Normenreihe ein Informationssicherheits-Managementsystem (ISMS), welches auf eine ganzheitliche, prozessorientierte Perspektive der Verantwortlichen aufbaut. Es wird dabei ein „Plan, Do, Check, Act“(PDCA) Verbesserungsprozess zugrunde gelegt, welcher eine kontinuierliche Verbesserung der Sicherheit sicherstellen soll [DIN11c]. ISO/IEC 27001 definiert die hierfür notwendigen Anforderungen [DIN14d]. Im weiteren wird ein Umsetzungsleitfaden in den Normen ISO 27002 [DIN08] und ISO 27003 [DIN17] sowie ein Leitfaden zum Risikomanagement in ISO 27005 [DIN11d] gegeben.
In Hinblick auf den Energiebereich sind insbesondere die DIN ISO/IEC TR 27019 [DIN15d], welche auf der DIN ISO/IEC 27002 aufbaut und Anforderungen an das ISMS für Prozessleitsysteme und Automatisierungstechnik in der Energieversorgungsindustrie näher spezifiziert sowie die IEC 62443-3-3 [DIN15c], welche Anforderungen an die IT-Sicherheit von industriellen Prozessleitlösungen beschreibt, von Bedeutung. Beide beziehen sich auf die 27000er Normenreihe und erweitern diese um anwendungsspezifische Kriterien, welche im Folgenden weiter beschrieben werden sollen.

DIN ISO/IEC TR 27019 Die technische Richtlinie DIN ISO/IEC 27019 *Leitfaden für das Informationssicherheitsmanagement von Steuerungssystemen der Energieversorgung auf Grundlage der ISO/IEC 27002* ist eine branchenspezifische Erweiterung der ISO/IEC 27000

Standard	Titel	Kurzbeschreibung
ISO/IEC 27000 / DIN EN 27000	Informationstechnik – IT-Sicherheitsverfahren – Informationssicherheits-Managementsysteme (ISMS)	International anerkanntes Normenwerk zum Informationssicherheitsmanagement. Unternehmen können ihre Informationssicherheitsmaßnahmen und Prozesse nach ISO/IEC 27001 zertifizieren lassen.
ISO/IEC 27002	IT-Sicherheitsverfahren – Leitfaden für das ISMS	Sammlung von Vorschlägen für Kontrollmechanismen in ISMS mit Fokus auf Sicherheit gegen Angriffe.
BSI IT-Grundschutz	BSI-Standard 100-1: Managementsysteme für Informationssicherheit BSI-Standard 100-2: IT-Grundschutz-Vorgehensweise	Umfangreiche Sammlung von Dokumenten die ein Vorgehen zur Erkennung und Bewertung von Schwachstellen in IT-Umgebungen beschreiben. Deutsches Pendant zur ISO/IEC 27000.
BDEW Whitepaper	Anforderungen an sichere Steuerungs- und Telekommunikationssysteme	Beschreibt die grundsätzlichen Sicherheitsmaßnahmen für Steuerungs- und Telekommunikationssysteme mit Bezug zur ISO/IEC 27002.
DIN ISO/IEC TR 27019	Informationstechnik – Sicherheitsverfahren – Leitfaden für das Informationssicherheitsmanagement von Steuerungssystemen der Energieversorgung auf Basis der ISO/IEC 27002	Die Norm ISO/IEC 27002 enthält Umsetzungshinweise zu den in ISO/IEC 27001 geforderten Sicherheitsmaßnahmen. Durch die ISO/IEC TR 27019 werden diese um sektorspezifische Aspekte der Energiewirtschaft (insb. *Smart-Grid*) erweitert.
IEC 62443 / DIN IEC 62443	Industrielle Kommunikationsnetze – IT-Sicherheit für Netze und Systeme	Beschreibt ein ganzheitliches IT-Sicherheitskonzept basierend auf der Analyse der grundlegenden Sicherheitsanforderungen und für die jeweilige Aufgabe benötigtem Schutzniveau.
VDI / VDE - Richtlinie 2182	Informationssicherheit in der industriellen Automatisierung	Vorgehensmodell zur Umsetzung konkreter Schutzmaßnahmen mit praxisnahen Beispielen aus der Automatisierungstechnik. Der Fokus liegt auf den Abhängigkeiten zwischen Herstellern, System-Integratoren und Betreibern.
NERC CIP / 001 bis 009	North American Electric Reliability Council - Critical Infrastructure Protection	Zentrale Norm (verpflichtend) in Nordamerika mit dem Ziel kritische Infrastrukturen zu schützen und einen sicheren Betrieb zu gewährleisten
IEEE 1686	Standard for Intelligent Electronic Devices (IEDs) - Cyber Security Capabilities	Beschreibt grundlegende Sicherheitsanforderungen für IEDs in Schaltanlagen (Stationen). Ist auf diesen Bereich beschränkt und konnte in Europa bisher keine praktische Relevanz erreichen.

Tabelle 2.3.: Wichtige Standards der organisatorischen IT-Sicherheit [Hel16]

Normenreihe und fokussiert sich auf Aspekte der ISMS für Prozesssteuerungssysteme der Energieversorgung aller relevanten Domänen wie Strom, Gas und Wärme. Die Richtlinie stellt die formale Nachfolge des 2007 veröffentlichten BDEW-Whitepapers (siehe weiter unten in diesem Abschnitt) dar [DIN14a].
Die Richtlinie tangiert alle IT-gestützten Prozesssteuerungs-, Leit- und Automatisierungssysteme sowie die damit verbundenen Kommunikationskomponenten und Mess-, Zähl- und Schutzsysteme, inklusive der darin integrierten Soft- und Firmware im Bereich der Steuerung und Überwachung von Energie-Umwandlung, -Übertragung, -Verteilung und -Speicherung [DIN15d].
Die ISO/IEC 27019 erhebt nicht den Anspruch, die ISO/IEC 27002 abzulösen oder gänzlich zu ersetzen, viel mehr ist sie als Ergänzung gedacht, welche die Grundgedanken der 27002 fortführt - und wo für die Bedürfnisse des Energiesektors notwendig - modifiziert und ergänzt. Um diese Kohärenz zu unterstreichen, ist auch die Struktur der 27019 an die Grundlage der 27002 angelehnt. Die zusätzlichen Empfehlungen betreffen eine Vielzahl von Bereichen, welche von der „Organisation der Informationssicherheit“(Abschnitt 6) über das „Betriebs- und Kommunikationsmanagement“(Abschnitt 10) bis hin zu „Zugangskontrollen“(Abschnitt 11) und der „Beschaffung, Entwicklung und Wartung von Informationssystemen“(Abschnitt 12) reichen. Die dabei präsentierten Maßnahmen und die beschriebenen Methoden sind sehr unspezifisch. So wird beispielsweise zur Sicherung der Prozessdatenkommunikation (Punkt 10.6.3) die „Sicherstellung der Vertraulichkeit, Integrität und Verfügbarkeit“[DIN15d] gefordert, ohne weiter zu spezifizieren, in welchem Umfang oder mit welchen Mitteln dies erreicht werden kann. Einzige Anregung für ein weiteres Vorgehen ist ein Verweis auf die IEC 62351 [IEC07b].

IEC 62443 Industrielle Kommunikationsnetze - IT-Sicherheit für Netze und Systeme
Die IEC 62443 definiert elementare IT-Sicherheitsanforderungen und leitet daraus Systemanforderungen an automatisierungstechnische Anlagen ab. Der Fokus liegt dabei insbesondere auf kritischen Infrastrukturen wie der Energieversorgung und Verteilung. Die Normenreihe wird von der IEC in Zusammenarbeit mit der *International Society of Automation* (ISA) herausgegeben. Weitere Normungsorganisationen wie die CENELEC und auch die DKE planen Veröffentlichungen der auch als *ISA 99* bekannten Reihe.
Die in Abschnitt 62443-1-1 definierten elementaren Sicherheitsanforderungen (*Foundational Requirement* - FR) gliedern sich wie folgt [Hel16]:

- FR 1 – Identifikation und Authentifizierung: Sicherstellung, dass alle Benutzer (Personen, Software-Agenten und Geräte) sich erfolgreich identifizieren und authentifizieren.
- FR 2 – Nutzungskontrolle: Sicherstellung, dass authentifizierte Benutzer (Personen, Software-Prozesse oder Geräte) die ihnen zugeteilten Rechte nutzen können.
- FR 3 – Systemintegrität: Sicherstellung der Integrität, um unberechtigte Manipulationen zu verhindern.
- FR 4 – Vertraulichkeit der Daten: Sicherstellung der Vertraulichkeit der Information bei Übertragung und in der Datenablage, um unberechtigte Weiterverbreitung zu verhindern.

- FR 5 – Eingeschränkter Datenfluss: Segmentierung des Systems in Zonen und Kanäle, um unnötigen Datenverkehr einzuschränken.
- FR 6 – Rechtzeitige Reaktion auf Ereignisse: Ermöglichen einer zeitnahen Reaktion auf sicherheitskritische Ereignisse
- FR 7 – Ressourcenverfügbarkeit: Sicherstellung der Verfügbarkeit von Ressourcen und Daten für Berechtigte.

Mittels der in Abschnitt 62443-3-3 beschriebenen *Security Level* (SL), die den Schutz gegen Missbrauch und Angriffe kategorisieren, ist es möglich, notwendige oder vorhandene Schutzniveaus der einzelnen FRs für eine konkrete Anlage in klar definierter Form zu beschreiben. Die *Security Level* reichen dabei von einem ungewollten oder zufälligen Missbrauch (SL 1) bis hin zum Schutz gegen gewollten Missbrauch unter Verwendung hoch entwickelter Mittel und erweiterter Ressourcen (SL 4) [DIN15c]. Darüber hinaus werden die FRs um zusätzliche Anforderungen zu Systemanforderungen erweitert. Diese sind mittels Hauptanforderungen und bei Bedarf mehreren Erweiterungen, sogenannten *Requirement Enhancement*s - RE, beschrieben. In Abhängigkeit des geforderten SL müssen die definierten Hauptanforderungen und RE erfüllt werden.
Zwecks Einbettung der oben beschriebenen Aspekte in ein ganzheitliches IT-Sicherheitssystem sollen in naher Zukunft die Teile 62443-2-1 und 62443-2-2 veröffentlicht werden. Sie stützen sich maßgeblich auf die Inhalte der ISO/IEC 27000-Reihe und erweitern sie um automatisierungstechnische Aspekte. Da sie somit ein ähnliches Ziel wie die oben beschriebene ISO/IEC TR 27019 verfolgen, wird eine Harmonisierung angestrebt [Ise].

BSI IT-Grundschutz Das BSI definiert mit seinen Standards 100-1 bis 100-4 IT-Sicherheitsmaßnahmen für öffentliche Einrichtungen und Unternehmen aller Branchen. Die definierten Maßnahmen sind dabei bewusst branchenübergreifend und können entsprechend der individuellen Gegebenheiten erweitert werden. Die einzelnen Standards bauen zum Teil auf der in 2.3.1 beschriebenen ISO/IEC 27000-Normenreihe auf. So werden zum Beispiel im BSI Standard 100-1 *Managementsysteme für Informationssicherheit* (ISMS) die Empfehlungen der ISO 27000 zum PDCA-Zyklus aufgegriffen und stellenweise ergänzt [BSI08a]. Aufbauend darauf bietet der BSI-Standard 100-2 *IT-Grundschutz-Vorgehensweise* eine praxisnahe Schritt-für-Schritt-Anleitung zum Aufbau und Betrieb eines ISMS. Dem Grundsatz des PDCA-Zyklus folgend wird eine sogenannte Sicherheitskonzeption erarbeitet und die in Abbildung 2.6 dargestellten Phasen werden dabei durchlaufen.

In diesem Prozess müssen initiale Basisparameter definiert und organisatorische Grundlagen für die Einführung geschaffen werde. Dies umfasst die Definition der personellen Verantwortlichkeiten und die Ausarbeitung von Dokumenten, wie z.B. einer Leitlinie zur Informationssicherheit oder den Organisationsrichtlinien des Sicherheitsprozesses. Sind diese Eingangsvoraussetzungen geschaffen, wird in einem ersten Schritt der Geltungsbereich der Maßnahmen (der sogenannte Informationsverbund) festgelegt. So ist es möglich Maßnahmen für gesamte Organisationen oder auch nur einzelne Geschäftsprozesse zu etablieren. Die Struktur des hier definierten Bereichs wird im nächsten Schritt im Hinblick auf die Kommunikationsverbindungen, IT-Systeme und Geschäftsprozesse analysiert und dokumentiert. Die anschließende Risikoanalyse dient der Feststellung des Schutzbedarfes. Sie baut auf den im vorherigen Schritt

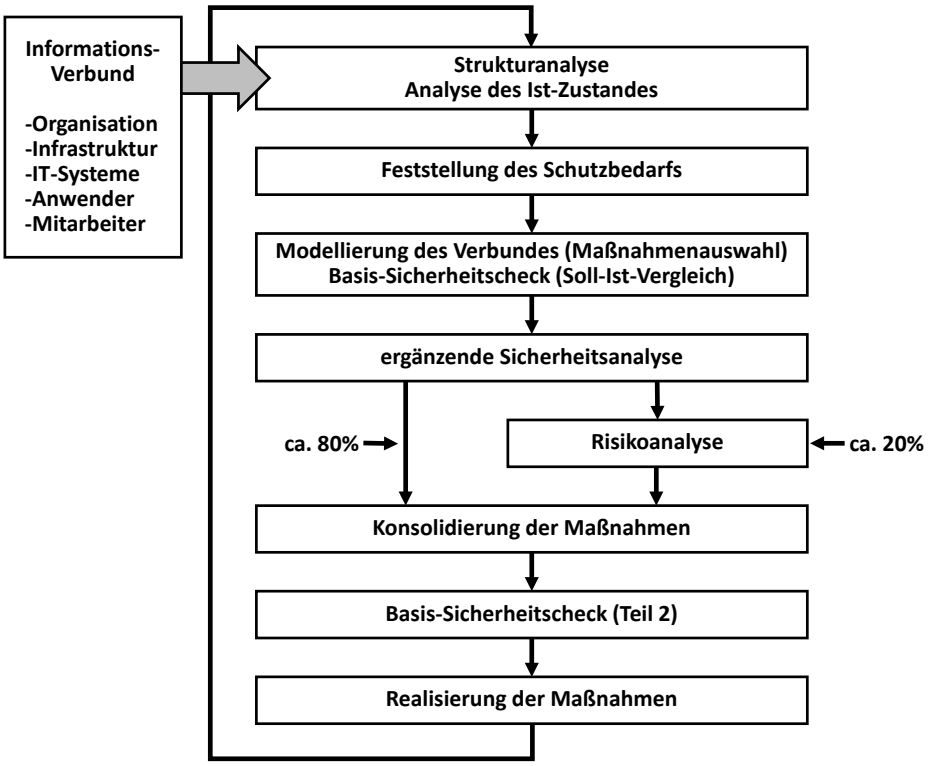

Abbildung 2.6.: Phasen der Sicherheitskonzeption nach BSI 100-2 [BSI08b]

erarbeiteten Parametern auf und leitet daraus die Risiken einer Sicherheitslücke ab. Auf Basis der hier identifizierten Risiken und einer Quantifizierung in die Schutzbedarfskategorien *normal*, *hoch* oder *sehr hoch* werden dann die konkreten Maßnahmen definiert [BSI08b].
Um diesen Prozess auch bei besonders schutzbedürftigen Applikationen anwenden zu können, werden im BSI Standard 100-3 *Risikoanalyse auf der Basis von IT-Grundschutz* weitergehende Verfahren zur Risikoanalyse definiert. Diese ergänzen die oben beschriebenen Maßnahmen und dienen der Konsolidierung und Überprüfung der erarbeiteten Ergebnisse um diese als Eingangsparameter für eine weitere Iteration nutzen zu können.
Das BSI hat bereits in der Konzeption seiner Standards die Möglichkeiten einer Zertifizierung vorgesehen. Aufgrund der hohen Überdeckung mit der ISO 27001 wird diese neben dem ISMS und den konkreten Sicherheitsmaßnahmen mit geprüft und zertifiziert [BSI08a].

BDEW-Whitepaper Das BDEW-Whitepaper *Anforderungen an sichere Steuerungs- und Telekommunikationssysteme* findet vermehrt Anwendung bei Neuentwicklungen von Steuerungs- und Kommunikationssystemen von Energieversorgungsunternehmen (EVUs) [BDE15]. Seit einer grundlegenden Überarbeitung im Jahre 2015 ist das Whitepaper in weiten Teilen an die zuvor beschriebene DIN ISO/IEC TR 27019 angelehnt. Am Anfang eines jeden Abschnittes werden die Gemeinsamkeiten aufgezeigt und dann eine nähere Beschreibung des Inhaltes gegeben. Dieser gliedert sich wie folgt: Der Abschnitt *Allgemeines* beschreibt Grundlagen des sicheren Systemdesigns wie Redundante Strukturen, Verschlüsselungstechniken, Integritätsprüfungen und Systemdokumentation. Der zweite Abschnitt *Basissystem* behandelt Konzepte der sicheren IT-Konfiguration und Systemhärtung. Dazu gehören neben einem Antivirenkonzept und einer adäquaten Nutzerauthentifizierung auch ein Whitelisting

von benötigten Diensten und Anwendungsfunktionalitäten. Der Abschnitt *Netze/Kommunikation* ergänzt diese lokale Perspektive um Vorgaben für vernetzte Systeme. Neben den oben genannten Faktoren Authentifizierung, Verschlüsselung und Integritätsprüfung wird ebenfalls eine durch Firewalls unterstützte Zonierung, also die horizontale und vertikale Segmentierung des Netzes, empfohlen. Der Abschnitt *Anwendung* fordert eine rollenbasierte Nutzerverwaltung, welche mit einer personenspezifischen Identifizierung und Authentifizierung verknüpft werden muss. Um die Integrität der Anwendung sicherzustellen werden Zeitstempel oder Sequenznummern in allen Kommunikationsvorgängen empfohlen, da dies Replay-Angriffe erschwert. Der für diese Arbeit relevante Abschnitt *Entwicklung, Test und Rollout* fordert den Einsatz von Entwicklungsstandards und Programmierrichtlinien sowie das dokumentierte Testen auf Staging-Systemen. Die Integrität des Quellcodes muss überprüfbar sein und mittels Change-Management-Direktiven auch ein Rollback auf vorherige Versionen ermöglicht werden. Parallel hierzu soll ein Prozess definiert werden, welcher einen strukturierten Umgang mit Sicherheitslücken definiert. Aufbauend hierauf beschreibt der letzte Abschnitt *Datensicherung/-wiederherstellung und Notfallplanung* das Management von Backups und den getesteten Wiederanlauf der Systeme.

2.3.2. Funktionale Sicherheit

Neben den Aspekten der *Security*, welche in Abschnitt 2.3 beschrieben wurden, sind auch Fragestellungen zur *Safety* im Rahmen der Konzeption und Entwicklung eines Leitsystems zu beachten. In diesem Kontext sind insbesondere die beiden folgenden Normen hervorzuheben: Die IEC 61508 beschreibt diese in Hinblick auf die Vermeidung von Fehlern, welche Konsequenzen für die Sicherheit von Personen und Umwelt nach sich ziehen, mit einem Fokus auf Steuerungssysteme. Die DIN EN 50160 behandelt Spannungsanforderungen im Elektrizitätsnetz, welche einer ordnungsgemäßen Funktion der am Netz angeschlossenen Geräte dienen.

IEC 61508 - Funktionale Sicherheit Die IEC 61508 ist eine sogenannte Sicherheits-Grundnorm. Sie dient allen technischen Komitees des IEC als Grundlage im Rahmen der Entwicklung anwendungsspezifischer Normen, welche elektrische, elektronische und programmierbare elektronische Systeme umfassen [DIN05]. Existiert für den Anwendungsbereich keine anwendungsfallspezfische Norm, so ist die IEC 61508 für „alle Typen sicherheitsbezogener elektrische, elektronische und programmierbare elektronische Systeme, einschließlich Schutzsysteme und Leit- und Steuerungssysteme“ [DIN11b] anzuwenden.
Ziel der Norm ist die Minimierung unvermeidlicher technischer Risiken auf Menschen und Umwelt sowie der Evaluierung wirtschaftlicher Schäden. Aufgrund der Unvermeidbarkeit von Fehlern soll deren Eintrittswahrscheinlichkeit gesenkt und mittels Sicherheitszielen transparent kommuniziert werden. Es werden dabei alle Phasen des Produkt-Lebenszyklus betrachtet und auf Quellen systematischer Fehler untersucht. Der daraus abgeleitete *Sicherheitslebenszyklus* wird im ersten Teil der Norm definiert und beschreibt Maßnahmen zum Management der funktionalen Sicherheit sowie der Verifikation und der Beurteilung der funktionalen Sicherheit für die einzelnen Phasen in den Abschnitten 7.2 bis 7.17. Auf Basis der sogenannten *Safety Integrity Level* (SIL) werden in der IEC 61508-1 vier Sicherheitsanforderungsstufen eingeführt. Diese helfen die Wahrscheinlichkeit eines gefährlichen Ausfalls zur kategorisieren. [DIN05].

DIN EN 50160 - Spannungshaltung Neben der Frequenzhaltung ist die Spannungshaltung eine der wichtigsten Aufgaben im Rahmen der Steuerung elektrischer Netze. Die grundlegenden Anforderungen an die Spannungsqualität sind in der DIN EN 50160 *Merkmale der Spannung in öffentlichen Elektrizitätsnetzen* definiert. Die Norm beschreibt die Merkmale der elektrischen Energie durch Angaben über den Verlauf der Netzwechselspannung. Im Sinne der Nutzung elektrischer Energie ist es wünschenswert, dass die Versorgungsspannung „eine konstante Frequenz, eine perfekte Sinus-Kurvenform und eine konstante Höhe aufweist“ [DIN11a]. Aufgrund der Komplexität der Spannungshaltung und der mannigfachen Rückwirkungen der Erzeuger, Verbraucher und Übertragungseinrichtungen untereinander wurden in der DIN EN 50160 Ober- und Untergrenzen für die unterschiedlichen Spannungsbänder (also Hoch-, Mittel- und Niederspannungsbereich) definiert. Es werden dabei Faktoren wie die Nennspannung, Grenzen für langsame und schnelle Spannungsänderungen, Frequenzbereiche, Grenzwerte für Flicker und Oberschwingungen sowie der Spannungsunsymmetrie definiert. Auch Aspekte wie Überspannungen, Spannungseinbrüche und Versorgungsunterbrechungen werden thematisiert und mittels Richtwerten für deren Auftrittswahrscheinlichkeiten begrenzt.
Um die in dieser Norm definierten Grenzwerte einhalten zu können, sind die dafür notwendigen Sensoren und Kommunikationsmöglichkeiten im Rahmen des Engineerings zu beachten. Die DIN EN 50160 ist daher ein wichtiger Baustein automatisierter Steuerungslösungen für elektrische Energiesysteme.

2.3.3. Kommunikationssicherheit

IEC 62351 - technische Sicherheitsmaßnahmen Die zu großen Teilen in Form einer technischen Spezifikation vorliegende Normenreihe IEC 62351 beschreibt protokollspezifische technische Maßnahmen, welche eine sichere Kommunikation ermöglichen sollen. Die zehn Teile der Norm werden im Folgenden kurz vorgestellt:
IEC 62351-1 beschreibt protokollunabhängige Sicherheitsbedrohungen, wie Datendiebstahl, -modifikation oder -nichtverfügbarkeit. Basierend auf diesen Bedrohungen werden Schutzziele wie Authentizität, Integrität, Verfügbarkeit etc. definiert, welche in den weiteren Teilen der Norm adressiert werden [IEC07b].
Der zweite Teil der Reihe umfasst ein Glossar der im Kontext der IT Sicherheit genutzten Begrifflichkeiten. Es handelt sich dabei hauptsächlich um Verweise auf andere Standards, in welchen diese ursprünglich definiert wurden.
Teil drei beschreibt Sicherheitsmaßnahmen für TCP/IP-basierte Protokolle wie TASE.2, IEC 61850 und IEC 60870-5-104. Der Fokus liegt dabei auf der *Transport Layer Security* basierten Absicherung der Transportschichten um die Schutzziele Authentizität, Vertraulichkeit und Integrität zu gewährleisten. Bemerkenswert ist, dass das Schutzziel Verfügbarkeit nicht berücksichtigt wird [Usl09]. Die IEC 52351-3 ist aktuell der einzige Teil der Reihe, welcher den Status *International Standard* erreicht hat.
IEC 52351-4 beschreibt die Absicherung MMS-basierter Kommunikation auf der Anwendungsebene. Insbesondere TASE.2 und die IEC 61850 profitieren von den hier empfohlenen Maßnahmen, welche die in Teil drei beschriebenen Schutzziele um den Faktor Verfügbarkeit erweitern.
Teil fünf beschreibt ein Signierungsverfahren für serielle Punkt-zu-Punkt-Verbindungen, wie den Protokollen der IEC 60870-5-101 bis 103. Aufgrund der eingeschränkten Eignung von Zertifikaten wird ein regelmäßig zu tauschender Schlüssel zur Signatur schützenswerter Ses-

sions empfohlen. Es werden damit die Schutzziele Authentizität und Verfügbarkeit abgedeckt. Wird, trotz der Bandbreitenbeschränkungen, die gesamte Kommunikation verschlüsselt, so sind auch die Schutzziele Vertraulichkeit und Integrität gewahrt. Werden jedoch nur kritische Teile des Nachrichtenaustauschs wie Steuerbefehle oder -Parameter gesichert, so werden die beiden letztgenannten Ziele nicht erreicht.
Ergänzend zur IEC 62351-4 werden im sechsten Teil Möglichkeiten der Absicherung IEC-61850-basierter Kommunikation, außerhalb der MMS-basierten Protokolle, aufgezeigt. Parallel hierzu werden die Echtzeitprotokolle GOOSE und SV betrachtet. Aufgrund der ressourcenarmen Feldgeräte und der Forderung nach geringen Latenzen wird empfohlen die Kommunikation nicht zu verschlüsseln. Statt dessen soll parallel der IEC 52351-5 eine digitale Signatur genutzt werden. Darüber hinaus wird eine Segmentierung in logische Teilnetze, sogenannte Virtual Local Area Networks (VLAN) dargestellt. Auch wird empfohlen alle Nachrichten mit einem Zeitstempel zu versehen und nach einer definierten Schwelle zu verwerfen [IEC07a]. Es werden somit die Authentizität, Verfügbarkeit sowie die Integrität der Kommunikation berücksichtigt.
Der siebte Teil der Normenreihe beschreibt Maßnahmen und Werkzeuge, welche die Verfügbarkeit des Gesamtsystems erhöhen sollen. Kern des Dokuments ist ein Datenmodell für das Netzwerk- und Systemmanagement, welches, neben der Überwachung der Netzwerkkomponenten während des Betriebs, insbesondere im Zuge der Modellierung eine Reflektion der notwendigen Infrastruktur zur Überwachung und Betrieb des Systems erleichtern soll [DIN14a].
Das in den vorangegangen Teilen etwas stiefmütterlich behandelte Schutzziel der Verbindlichkeit wird in der IEC 62351-8 aufgegriffen. Es wird hierfür ein rollenbasiertes Zugangskontrollsystem erläutert, welches neben einer strikten Beschränkung der Nutzerrechte eine Nachverfolgbarkeit von Nutzerhandlungen vorsieht. Dieses System soll dabei nicht nur auf menschliche Akteure beschränkt werden, sondern auch Software-Systeme umfassen [IEC11].
Teil neun beschreibt das Management von Zertifikaten und Sicherheitsschlüsseln. Mechanismen und Technologien zur Erstellung, zur Verteilung, zum Austausch sowie zur Tilgung werden aufgezeigt und diskutiert. Im Fokus des bisher als interner „Norm-Entwurf“ vorliegenden Dokumentes liegt dabei die Klassifizierung der Sensitivität der Informationen sowie die Authentifizierung [DIN14a].
Der als *Technischer Report* vorliegende zehnte Teil der Normenreihe gibt in Form einer Richtlinie Hinweise zum Umgang mit dem in den vorangegangenen Dokumenten vorgestellten IT-Sicherheitskonzept für die Energieversorgung. Es werden neben der Darstellung der allgemeinen Zusammenhänge auch Hinweise auf weiterreichende Normen gegeben, welche im Rahmen der Integration beachtet werden sollten.
Basierend auf [Usl09] stellt Tabelle 2.4 die in der IEC 62351 adressierten Schutzziele und die diesen zugrunde liegenden Maßnahmen und Schutztechnologien dar.

2.3.4. Smart-Meter-Gateway-Kommunikation

Tabelle 2.5 gibt eine Übersicht der aktuellen Normen und Standards im Bereich Smart-Metering-Kommunikation. In diesem Kontext muss zwischen Normen und Standards, welche Kommunikationsmittel und Datenmodelle beschreiben, und jenen, welche die dabei Anwendung findenden Schutzmaßnahmen definieren, unterschieden werden. So definieren die IEC 62056, die DIN 13757 sowie die *Smart-Meter-Language* Datenmodelle und Techniken für den Informationsaustausch zwischen *Smart-Metern* und den Messstellenbetreibern. In den BSI

Bereich	Schutzziel	Risiko / Gefahr	Schutzmaßnahme	Schutztechnologie	Ebene
TCP/IP	Authentizität	Maskierung / Spoofing	Zertifikate	Keine Angaben	Anwendungsebene
	Vertraulichkeit	Mithören	Verschlüsselung	TLS	Transportebene
	Integrität	Manipulation	Prüfwert	MAC	Anwendungsebene
MMS basierte Kommunikation	Authentizität	Maskierung / Spoofing	Zertifikate	ACSE-Authentifikation	Anwendungsebene
	Vertraulichkeit	Mithören	Verschlüsselung	TLS	Transportebene
	Integrität	Manipulation	Sicherheits-Log	Keine Angabe	Anwendungsebene
	Verfügbarkeit	Wiederholung	Zeitstempel oder Sequenznummern	Keine Angabe	Anwendungsebene
Serielle Kommunikation	Authentizität	Maskierung / Spoofing	Prüfwert	HMAC / GMAC	Anwendungsebene
	Verfügbarkeit	Wiederholung / Denial-of-Service	Sequenznummern	Keine Angabe	Anwendungsebene
61850 (v.a. GOOSE & SV)	Authentizität	Maskierung / Man-in-the-middle	Erweiterung der SCL-Zertifikate	Keine Angaben	Anwendungsebene
	Integrität	Manipulation	Prüfwert	Erweiterung der PDU um HMAC	Anwendungsebene
	Verfügbarkeit	Abstreitbarkeit	Zertifikate, Access Token	Role Based Access Control	Anwendungsebene

Tabelle 2.4.: Übersicht der Schutzmaßnahmen nach IEC 62351

Schutzprofilen BSI-CC-PP-0073 und BSI-CC-PP-0077 werden dann konkrete Maßnahmen zur Absicherung der Feldgeräte und deren Kommunikation angegeben. Auf diesen Schutzprofilen baut die BSI Richtlinie 03109 auf und definiert die Anforderungen an die Kommunikationseinheiten eines *Smart-Meter-Gateways*. Aufgrund der steigenden Relevanz dezentraler Erzeugung und der avisierten Verschiebung von Haushaltslasten gewinnt diese unterste Ebene des Energienetzes eine zunehmend Bedeutung im Engineering von Leitlösungen. Daher soll die BSI TR-03109 im Folgenden vorgestellt werden. *Smart-Meter* stellen als Nachfolger der klassischen analogen Zähler, primär ein Messmittel zur Verbrauchserfassung von Energie- und Stoffströmen dar. Im Zuge der Digitalisierung sollen sie jedoch, ergänzt um ein *Smart-*

Standard	Titel	Kurzbeschreibung
BSI TR-03109	Anforderungen an die Interoperabilität der Kommunikationseinheit eines intelligenten Messsystems	Umfasst die Anforderungen an die Funktionalität, Interoperabilität und Sicherheit, welche die Einzelkomponenten in einem *Smart-Metering-System* erfüllen müssen. Darüber hinaus werden Anforderungen zur Prüfung dieser Eigenschaften definiert.
BSI Schutzprofil BSI-CC-PP-0073	Schutzprofil für die Kommunikationseinheit eines intelligente Messsystems für Stoff- und Energiemengen (Smart-Meter-Gateway PP)	In diesem Schutzprofil werden erforderliche Mindestsicherheitsanforderungen an das *Smart-Meter-Gateway* festgelegt
BSI Schutzprofil BSI-CC-PP-0077	Schutzprofil für das Sicherheitsmodul der Kommunikationseinheit eines intelligenten Messsystems für Stoff- und Energiemengen (*Security Module PP*)	Das Sicherheitsmodul dient zum einen als Speicher für das Schlüsselmaterial und stellt zum anderen für das SMGW selbst die Kernroutinen für Signaturerstellung und -prüfung, Schlüsselgenerierung, Schlüsselaushandlung sowie Zufallszahlengenerierung bereit. Im Schutzprofil wird die Sicherheitsfunktionalität erfasst und festgeschrieben.
IEC 62056	Messung der elektrischen Energie – Zählerstandübertragung, Tarif- und Laststeuerung	DLMS (*Device Language Message Specification*) i.V.m. COSEM (*COmpanion Specification for Energy Metering*) ist ein Datenmodell der Anwendungsebene für die Ablesung und Übermittlung von Zählerständen. Im Rahmen der Norm werden ebenfalls Sicherheitsaspekte behandelt.
DIN EN 13757	Kommunikationssysteme für Zähler und deren Fernablesung Auch bekannt als: M-Bus bzw. Wireless M-Bus	Definiert in allgemeiner Weise den Datenaustausch und die Kommunikation für Zähler und deren Fernablesung. Liefert eine Übersicht der zu verwendenden Protokolle sowie eine Spezifikation für die COSEM-basierte Anwendungsschicht.
SML	Smart-Meter-Language	Deutscher Standard, der ein simples Kommunikationsprotokoll für *Smart-Meter* definiert. Das Protokoll ist auf Applikationsebene angesiedelt und damit unabhängig von der eingesetzten Transporttechnik. Normierung als IEC 62056-5-3-8 angestrebt

Tabelle 2.5.: Standards aus dem Bereich Smart-Metering [Hel16]

Meter-Gateway (SMGW) als kundenseitige Schnittstelle für die gesamte Kommunikation von Mess-, Steuer- und Preissignalen dienen [AD13]. Die Systemarchitektur dieser Schnittstelle sowie die essentiellen Sicherheitsmaßnahmen werden in der BSI TR-03109 sowie den ihr zugrundeliegenden Schutzprofilen BSI-CC-PP-0073 und BSI-CC-PP-0077, beschrieben. Es ist vorgesehen, dass das SMGW drei Netzwerke miteinander verbindet und dabei sowohl den Datenschutz als auch die Systemsicherheit gewährleistet. Abbildung 2.7 zeigt die dabei entstehenden Verbindungen. Auf lokaler Ebene sind dies zum einen das Heimnetzwerk, welches

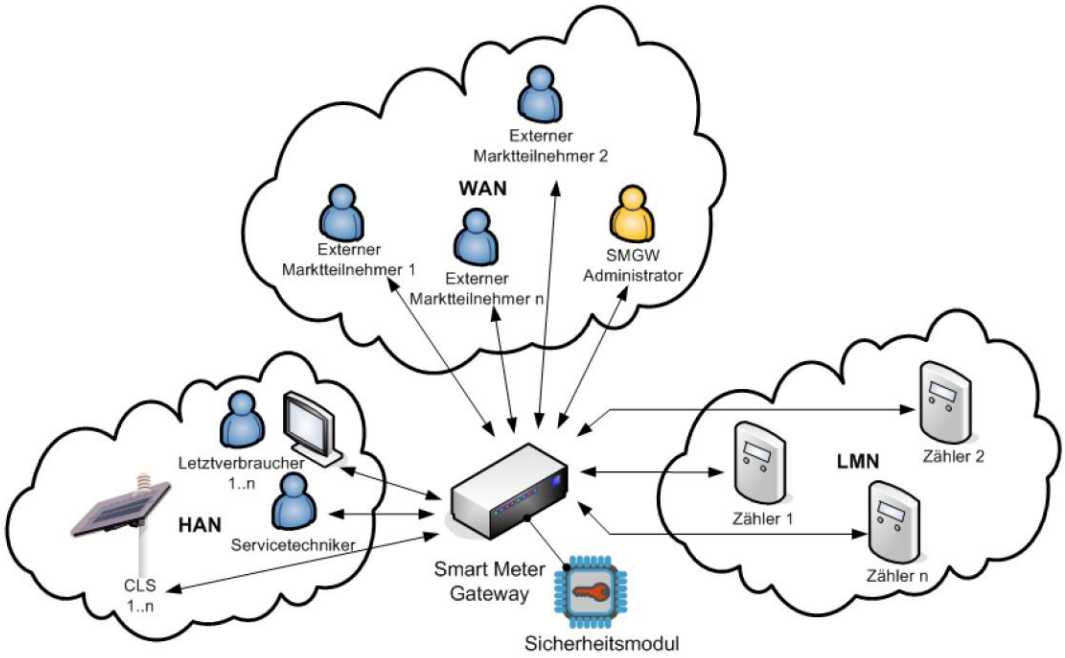

Abbildung 2.7.: Einsatzumgebung des Smart-Meter-Gateway[BSI13b]

Home Area Network (HAN) genannt wird. Über das HAN ist es dem Anschlussnutzer möglich Daten aus dem SMGW auszulesen. Außerdem werden über das HAN steuerbare Lasten (*Controllable Local Systems* CLS) angesteuert. Diese umfassen z.B. intelligente Hausgeräte sowie thermische und elektrische Speicher und Prosumer. Die CLS müssen gemäß §9 EEG durch den Netzbetreiber regel- oder abschaltbar sein. Auch externe Marktteilnehmer sollen, das Einverständnis des Nutzers vorausgesetzt, mittels SMGW relevante CLS steuern können. Hierfür agiert das SMGW als Proxy und erlaubt mittels beidseitig authentifizierter TLS-Verbindung die Nutzung proprietärer oder standardisierter Protokolle wie IEC 60870-5 oder IEC 61850.

Die zweite lokale Schnittstelle empfängt, verarbeitet und speichert Daten, welche durch die digitalen Messeinrichtungen im *Local Metrological Network* (LMN) bereitgestellt werden. Diese Daten umfassen neben den reinen Verbrauchswerten auch Informationen über eingespeiste Energiemengen und Parameter wie Spannung, Frequenz, Phasenwinkel etc.

Die *Wide Area Network* (WAN) Schnittstelle verbindet die beiden oben beschriebenen Netzwerke mit dem Weitverkehrsnetz und den daran angebundenen berechtigten externen Marktteilnehmern. Diese können nicht direkt auf das SMGW zugreifen. Alle Kommunikationsvorgänge werden stets durch das SMGW initialisiert. Lediglich der Administrator (in der Regel der zuständige Messstellenbetreiber) kann das Gerät über einen *Wake-Up Service* zu einem

Verbindungsaufbau auffordern. Sämtliche Dienste müssen über HTTP basierte *RESTful Webservices* als COSEM Objekte abgebildet und mittels TLS abgesichert werden [BSI13b].

2.4. Evaluation der Kommunikationsprotokolle und Sicherheitsstandards

Aufgrund der Schnelllebigkeit des Automatisierungsbereichs und der begleitenden Anforderungen wurden Experteninterviews geführt um den Status-Quo in Wissenschaft und Wirtschaft zu ermitteln. Diese Interviews führte Heller in seiner durch den Verfasser dieser Dissertation betreuten Masterarbeit am Institut für Automatisierungstechnik in 2016 durch [Hel16]. In diesen ermittelte er die Relevanz der oben beschriebenen Standards, Normen und Richtlinien, welche im Folgenden „Anforderungen“genannt werden. Diese Evaluation erfolgte aus Sicht von Branchenteilnehmer wie Verteilnetzbetreibern und Softwareherstellern. Die in Abs. 2.4.1 beschriebenen Kriterien dienten ihm dabei als Gewichtungsfaktoren. Aufgrund des geringen Umfangs der Grundgesamtheit (der Anzahl befragter Experten) seiner Studie sind die im Abschnitt 2.4.2 aus Hellers Resultaten abgeleiteten und quantitativ visualisierten Daten statistisch nicht relevant, sondern nur eine Tendenzaussage.

2.4.1. Kriterien

Es wurden durch Heller im wesentlichen drei Faktoren identifiziert [Hel16]: Die Relevanz einer Anforderung, der Aufwand um diese umzusetzen sowie die Steigerung der Produktqualität durch eine Berücksichtigung des Faktors. Die Faktoren und ihre Ausprägungen werden im Folgenden erläutert.

Bewertung	Bedeutung
1	Nicht relevant, kann vernachlässigt werden
2	Wenig relevant, Beachtung sinnvoll aber nicht notwendig
3	Mittlere Relevanz, die Beachtung ist wünschenswert
4	Hoch relevant, sollte berücksichtigt werden
5	Äußerst relevant oder gesetzlich vorgeschrieben. Obligatorisch.

Tabelle 2.6.: Skala zur Bewertung der Relevanz [Hel16]

Relevanz Die Relevanz der beschriebenen Anforderungen für Branchenteilnehmer ist offensichtlich das wesentliche Kriterium für eine zukünftige Berücksichtigung in Softwareprojekten der Energieleittechnik. Aufgrund des vergleichsweise geringen Alters der Anforderungen im Verhältnis zu den über Dekaden gewachsenen Strukturen der Energiewirtschaft war die Verankerung in der industriellen Praxis zu eruieren. Es wurden dabei die Relevanz für aktuelle Systeme und eine Einschätzung zur zukünftigen Entwicklung geprüft. Das in Tabelle 2.6 dargestellte Bewertungsschema wurde dabei eingesetzt.

Aufwand Der Aufwand eine Anforderung umzusetzen ist von hoher Relevanz, da dies die Wirtschaftlichkeit und die daraus resultierende Umsetzbarkeit eines Projektes wesentlich beeinflusst. Die Summe des Zeit- und Kostenaufwandes wird dabei als Aufwand interpretiert.

Es wurden im Rahmen der Studie sowohl der Zeitaufwand für die Umsetzung als auch die externen Kosten für Bibliotheken, Tests und Ähnliches geprüft. Folgendes Schema wurde zur Quantifizierung des Aufwandes angelegt:

Bewertung	Bedeutung
1	Kein Aufwand
2	Geringer Aufwand
3	Mittlerer Aufwand
4	Hoher Aufwand
5	Enormer Aufwand

Tabelle 2.7.: Skala zur Bewertung des notwendigen Aufwands [Hel16]

Qualität Im Kontext der Softwareentwicklung gibt es eine Vielzahl unterschiedlicher Qualitätsbegriffe. Es wurden daher, in Anlehnung an die ISO/IEC 9216, die folgenden fünf Qualitätsmerkmale definiert:

- **Funktionalität** : Interoperabilität, Angriffssicherheit
- **Zuverlässigkeit** : Fehlerfreiheit, Stabilität, Wiederherstellbarkeit
- **Benutzbarkeit** : Benutzerfreundlichkeit, (Fern-) Parametrierbarkeit
- **Effizienz** : Performance
- **Übertragbarkeit** : Auswirkungen auf bestehende Prozesse, Installierbarkeit

Diese werden in Abbildung 2.8 nach den ihnen zugeordneten Parametern getrennt dargestellt.

Bewertung	Bedeutung
1	Nicht relevant, kann vernachlässigt werden
2	Wenig relevant, Erfüllung sinnvoll aber nicht notwendig
3	Mittlere Relevanz, die Erfüllung ist wünschenswert
4	Hoch relevant, sollte erfüllt werden
5	Äußerst relevant, muss zwingend erfüllt werden

Tabelle 2.8.: Skala zur Bewertung der Qualitätsmerkmale [Hel16]

Die dahinter aufgezählten Ausprägungen wurden im Zuge der Interviews erfasst um der individuellen Interpretation der Studienteilnehmer auf die formalisierten Qualitätsbegriffe gerecht zu werden. Den beiden vorangegangenen Faktoren folgend, wurden die Anforderungen in Hinblick auf die Erfüllung einzelner Qualitätsmerkmale wie in Tabelle 2.8 dargestellt quantifiziert.

2.4.2. Analyse

Im Folgenden werden die in den Abschnitten 2.2 und 2.3 beschriebenen Standards und Normen auf Basis der identifizierten Faktoren bewertet. Die Bewertung basiert auf der von Heller 2016 durchgeführten Studie [Hel16]. Diese Studie umfasste fünf Experteninterviews in Form eines Leitfadeninterviews. Alle seine Interviewteilnehmer besitzen akademische Abschlüsse und langjährige Berufserfahrung in ihrem Fachgebiet. Vier der Interviewpartner stammen aus Unternehmen, welche Software für Energiesysteme entwickeln und vertreiben. Ein Interviewpartner ist im Netzmanagement eines Verteilnetzbetreibers tätig.

Relevanz Die Befragung zeigt, dass aus Sicht der Experten Sicherheitsaspekte mit der Digitalisierung an Gewicht gewinnen werden. Der Status Quo ist geprägt von Anforderungen an die Spannungsqualität - namentlich der DIN EN 50160 - sowie der seriellen Punkt-zu-Punkt-Kommunikation mittels IEC 60870-5. Alle Experten sind sich einig, dass diese beiden Faktoren im Rahmen aktueller Softwareprojekte Berücksichtigung finden müssen. Der BSI Grundschutz auf Basis der ISO 27000 sowie die technischen Sicherheitsmaßnahmen nach IEC 62351 werden ebenfalls als äußerst relevant erachtet - beide erhalten jedoch in aktuellen Szenarien eine Bewertung unter fünf Punkten.

Erst in zukünftigen Anwendungen werden durchgängig alle Anforderungen des Bereichs Si-

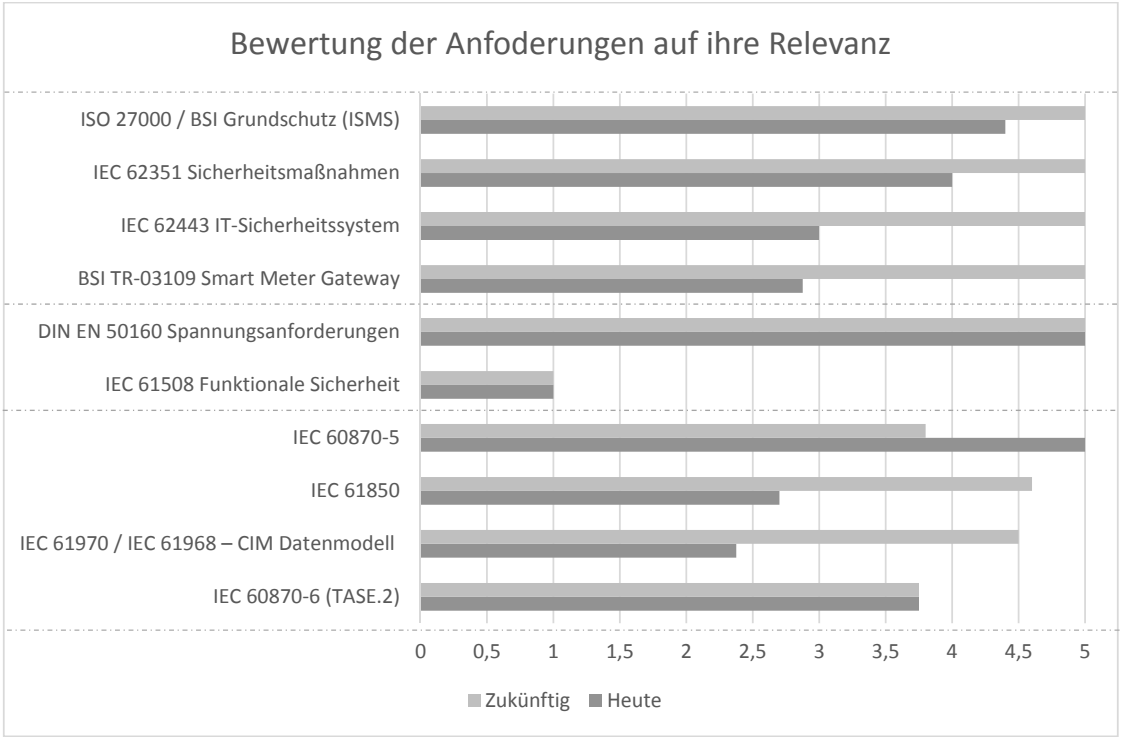

Abbildung 2.8.: Bewertung der Standards in Hinblick auf ihre Relevanz [Hel16]

cherheit (ISO 27000, IEC 62351, IEC 62443 und BSI TR-03109) mit der vollen Punktzahl bedacht. Die Spannungsanforderungen nach DIN EN 50160 verharren auf diesem Niveau. Im Feld der Kommunikationsstandards wird ein Umbruch erwartet - weg von der seriellen Kommunikation nach IEC 60870-5 hin zu flexibleren Informationsmodellen, welche der Kommunikation zugrunde gelegt werden können. Als Beispiele können hier die IEC 61850 oder

das CIM dienen. Die Kopplung der Leitstellen nach IEC 60870-6 wird nach Einschätzung der Experten auch in Zukunft mit TASE nach IEC60870-6 realisiert und erhält daher zukünftig wie auch heute eine Bewertung von 3,8 Punkten.

Aufwand Die befragten Experten bewerten die Aufwände zur Umsetzung der Sicherheitsanforderungen einhellig als hoch. Insbesondere die Etablierung und der Unterhalt eines ISMS nach ISO 27000 bzw. dem BSI Grundschutz wird hier als besonders aufwändig bewertet. Die Anforderungen an Geschäftsprozesse und Meldesysteme bedingen in der Regel Anpassungen in der Hard- und Software eines Unternehmens. So muss zum Beispiel Software vor ihrer Auslieferung auf gehärteten Insel-Systemen getestet werden, welche nicht mit der produktiven Umgebung verbunden sind. Ein vollautomatisches Deployment kann dabei den Aufwand vermeiden. Darüber hinaus soll es menschliche Fehler vermeiden, bedarf aber einer ausführlichen Planung, langwieriger Konfiguration und intensiver Tests, bevor es genutzt werden kann. Die vorgeschriebene Zertifizierung durch eine externe Stelle für ISO-27001-konforme Feldgeräte sowie die Sicherheitsmaßnahmen nach IEC 62351 werden als relevanter Kostenfaktor im Rahmen der Projektumsetzung betrachtet. Die Anforderungen der IEC 62443 sowie des *BSI Smart-Meter-Gateways* werden durch die Experten als leistbar bewertet. Insbesondere die Sicherheitsarchitekturen der *Smart-Meter* werden aufgrund der angestrebten hohen Verbreitung als ausgereift und wenig kostenintensiv betrachtet.

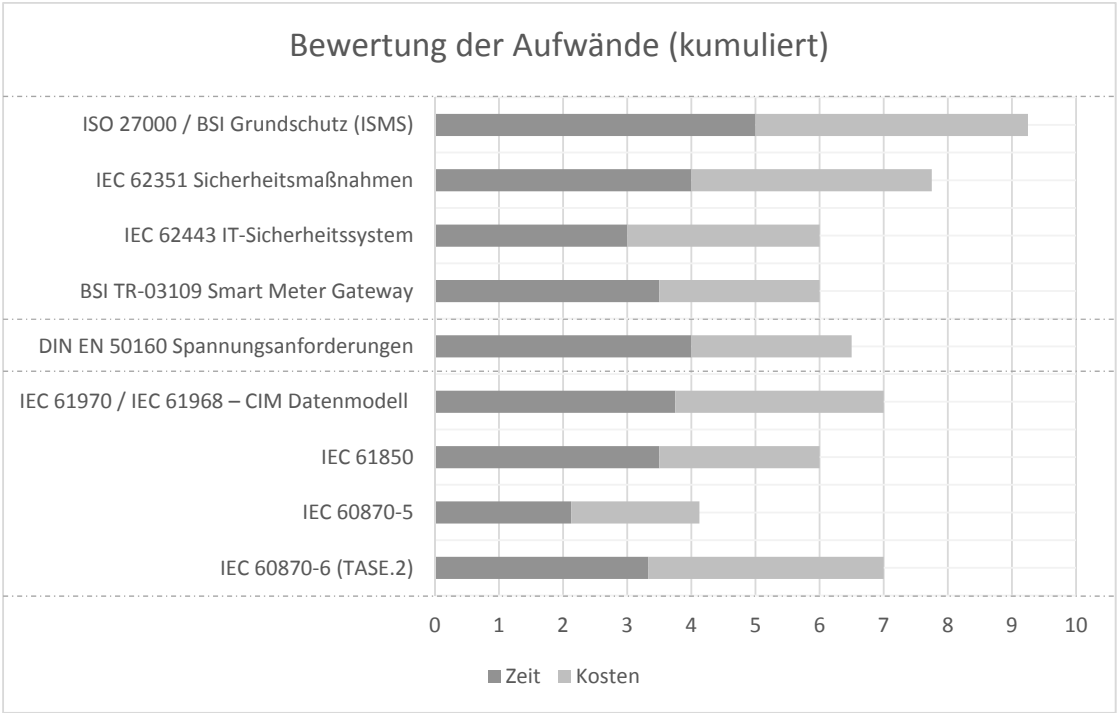

Abbildung 2.9.: Bewertung der Standards in Hinblick auf ihren Aufwand [Hel16]

Die Anforderungen im Bereich der funktionalen Sicherheit - namentlich der DIN EN 50160 - werden ebenfalls als lösbar erachtet. Zwar wird der initiale Aufwand zur Entwicklung von Steuerungslösungen als hoch bewertet, jedoch von einer hohen Wiederverwendbarkeit der Software ausgegangen.

Auch die zur Implementierung der IEC-61850- und CIM-basierten Kommunikation notwendigen Aufwände werden initial als hoch, aber langfristig gewinnbringend angesehen. Dieser Umstand ist durch die hohe Wiederverwendbarkeit der Informationsmodelle erklärbar. So wird von einer anwendungsfallspezifischen Skalierung des Implementierungsumfangs gesprochen, welche sich zum Beispiel in der Kombination vorhandener Informationsmodelle mit neuen (Teil-)Modellen manifestiert. Das bekannte und einfach strukturierte IEC-60870-5-Format wird unisono als äusserst kostengünstig bewertet.

Qualität Die entscheidenden Qualitätsmerkmale für Softwareprodukte im Bereich der Energietechnik sind nach Ansicht der Experten die Funktionalität, Zuverlässigkeit und Sicherheit der Lösung. Auch die Performanz und Konfigurierbarkeit der Software wird als wichtig erachtet, wenn auch in geringerem Maße. Aspekte wie Fern-Parametrierbarkeit, technische Interoperabilität oder Übertragbarkeit auf andere Systeme werden dabei als Grundvoraussetzungen betrachtet. Ob diese Bewertung aufgrund geringer Erfahrungen mit verteilten Systemen, mangels aktueller Verbreitung oder auf Basis der Annahme, dass solche Systeme stets dem Stand der Technik entsprechen werden, getroffen wurden, bleibt in Hellers Umfrage [Hel16] unklar.
Es kann jedoch aus den Aussagen der Experten abgeleitet werden, dass die Umsetzung von Sicherheitsmaßnahmen wie denen der IEC 62351 die Qualität des Produktes steigern. Auch die Berücksichtigung der DIN ISO/IEC 27019, des BDEW-Whitepapers oder der IEC 62443 sollten in Hinblick auf die Konformität mit etwaigen ISMS-Lösungen betrachtet werden. In Hinblick auf Performance, Interoperabilität und technische Anforderungen bieten die beschriebenen Lösungen nach IEC 60870-5 oder der IEC 61850 unterschiedliche Vor- und Nachteile und müssen in Hinblick auf den konkreten Anwendungsfall projektspezifisch bewertet werden.
Abbildung 2.10 zeigt eine detaillierte Auflistung der in Abschnitt 2.4.1 eingeführten Qualitätsmerkmale und deren Relevanz für Softwareprodukte im Bereich der Energiewirtschaft.

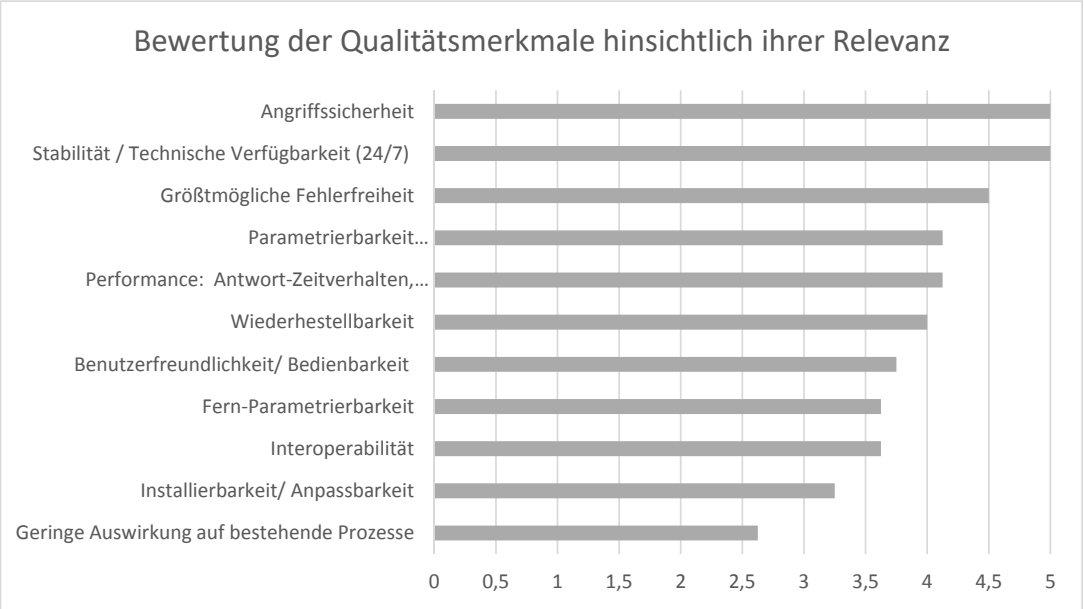

Abbildung 2.10.: Bewertung der Qualitätsmerkmale in Hinblick auf ihre Relevanz [Hel16]

2.5. Zwischenfazit

In diesem Kapitel wurden auf Basis von Normen und Standards die Anforderungen an die Sicherheit und Kommunikation von Softwarelösungen im Bereich der Energiewirtschaft beschrieben. Auf Basis eines initialen Überblicks über den aktuellen Stand der Technik und Regulierung wurden die Ergebnisse einer Expertenbefragung zur Relevanz, zu den erwarteten Aufwänden und Auswirkungen auf die Produktqualität präsentiert. Das Ergebnis dieser Befragung zeigte, dass der aktuelle Fokus der Experten auf Fragen der Sicherheit liegt. So werden die Normen ISO 27000 und die Sicherheitsmaßnahmen nach IEC 62351 als äußerst relevant und gleichzeitig als äußerst aufwendig in der Umsetzung betrachtet. Als relevante Qualitätsmerkmale werden übereinstimmend damit die Angriffssicherheit sowie Fragen der Stabilität und Verfügbarkeit bewertet. Nach Meinung der Experten werden auch die Interoperabilität und damit Protokolle und Datenmodelle wie die IEC 61850 oder CIM an Relevanz gewinnen. Diese werden jedoch in Hinblick auf die Implementierungsaufwände als weniger kritisch bewertet.
Ein Entwicklungsansatz für Softwarelösungen im Bereich der Energietechnik muss somit erlauben sicherheitsrelevante Normen und Standards zu integrieren und im Hinblick auf Kommunikationsprotokolle eine möglichst große Flexibilität bieten. Diese ließe sich durch eine abstrakte Beschreibung der Modellierungsschritte zur Auswahl und Erstellung der Kommunikationsprotokolle in einem generischen Metamodell erzielen.

3. Agenten in der Energieleittechnik

Bedingt durch die zunehmende Dezentralisierung der Energieerzeugung gewinnt das Konzept der Software-Agenten auch in diesem Bereich an Bedeutung. Im folgenden Abschnitt soll der Begriff des Agenten eingeführt, der aktuelle Stand der Forschung präsentiert und seine Anwendbarkeit in Energiesystemen der Zukunft diskutiert werden.
Die Definition dessen, was einen Agenten charakterisiert, ist einer Vielzahl an Randbedingungen unterworfen. Das wichtigste Differenzierungsmerkmal ist dabei die Wissenschaftsdomäne, in welcher die Agenten Anwendung finden sollen. Im Bereich der Automatisierungstechnik und den technischen Wissenschaften dient häufig die Beschreibung von Wooldridge und Jennings [WJ95] als Grundlage der weiteren Spezifikation von Agentensystemen. Diese besagt, dass ein Agent durch ein Softwareartefakt oder ein Computersystem repräsentiert wird. Dieses ist in eine Umgebung eingebettet, in welcher es autonom agieren kann. Dabei verfolgt der Agent individuelle Ziele und besitzt weitergehende, zum Teil optionale, Eigenschaften wie Persistenz und Kapselung der internen Prozesse. Auch kooperatives und kommunikatives Verhalten sowie Eigenschaften wie Lernfähigkeit und Mobilität werden in den technischen Domäne mit Agenten verknüpft [Woo02] . Im Bereich der Automatisierungstechnik wird diese Beschreibung häufig um die Notwendigkeit der Kopplung von Hard- und Software ergänzt [VDI10]. Diese Eigenschaften können in Abhängigkeit des konkreten Anwendungsfalls Berücksichtigung finden. Der durch die Untergruppe Energie des VDI-GMA-Fachausschusses 5.15 getroffenen Eingrenzung [BDL+13] folgend, sollen als minimale Eigenschaften zur Definition eines Agenten lediglich Autonomie, Kommunikations- und Kooperationsfähigkeit vorausgesetzt werden. Die allgemeingültige Definition eines Multi-Agentensystems wird im Folgenden ebenfalls berücksichtigt. Sie besagt, dass ein System, welches aus mehreren Agenten aufgebaut ist, als Multi-Agentensystem definiert wird. Mittels dieser Eigenschaften ist eine Modellierung und Implementierung der Agentensysteme in Anlehnung an die sozialen Eigenschaften menschlicher Beziehungs- und Kooperationsnetzwerke möglich. Die Fähigkeiten und Freiheitsgrade eines Agenten können dabei mit denen menschlicher Akteure in Produktions-, Handels- und Kooperationsbeziehungen verglichen werden. Somit wird eine intuitive Modellierung komplexer und heterogener Softwaresysteme möglich.
Dem Bild der Modellierung in Anlehnung menschlicher Verhaltensweisen folgend, können auch Agenten, ähnlich ihren menschlichen Vorbildern, ihre Entscheidungen auf Basis unterschiedlicher Faktoren treffen. Dabei werden in der Literatur insbesondere deliberative (abwägende) und reaktive Agenten unterschieden. Letztere reagieren, wie ihr Name bereits vermuten lässt, auf Basis einfacher Algorithmen auf Veränderungen in ihrer Umgebung oder auf Anfragen anderer Agenten. Im Gegensatz zu deliberativen Agenten ist eine Proaktivität dabei nicht vorgesehen. Deliberative Agenten durchlaufen komplexe interne Abwägungsprozesse, welche stets nach einer optimalen Strategie für die ihnen gegebenen Aufgaben suchen. Diese sind naturgemäß von ihrer Umgebung geprägt und unterscheiden sich daher stark in Abhängigkeit ihres Ausführungskontexts und ihren daraus abgeleiteten Implementierungen [PW04]. Eines der wichtigsten Konzepte ist dabei die Modellierung nach dem Belief, Desire,

Intention (BDI) Konzept, welches in [RGRG95] und [SBP+07] gut beschrieben wird. Die inhärente Komplexität von Agentensystemen verursacht jedoch ein zeitlich nicht-deterministisches Verhalten, welches insbesondere in prozessnahen Anwendungsfällen in der Regel vermieden werden sollte.
Diese Schwäche wird jedoch durch die Vorteile der Agentensysteme im Zuge der Modellierung und Implementierung verteilter Betriebsführungskonzepte kompensiert:

- Agentenbasierte Konzepte erlauben eine einfache Beschreibung verteilter Probleme sowie deren Lösung durch kooperative Akteure, welche sowohl individuelle als auch übergeordnete Ziele verfolgen. So können zum Beispiel die Ziele der lokalen Kostenoptimierung und der globalen Netzstabilität kombiniert werden.

- Die Entwicklung kann sich auf Agentenplattformen und -frameworks abstützen, welche die oben beschriebenen Eigenschaften der Agenten durch entsprechende Klassen und Schnittstellen unterstützen. Insbesondere Kommunikations- und Koordinationsprozesse können hierdurch wesentlich effizienter umgesetzt werden als in Systemen mit proprietären Schnittstellen.

- Zentrale Steuerungsprozesse können durch den Einsatz dezentraler, autonomer Agenten verschlankt werden. So können lokale Aspekte der Koordination in den Vor-Ort-Systemen gekapselt und bereits vorberechnet werden. Eine auf dieser lokalen Verarbeitung basierende bedarfsorientierte Kommunikation reduziert die Häufigkeit und den Umfang der notwendigen Kommunikation.

- Dynamische Veränderungen ihrer Umgebung können von Agenten zeitnah erfasst und ausgeglichen werden. Sie sind somit in der Lage der hohen Änderungsgeschwindigkeit des Strom- und Energienetzes stand zu halten und sich (idealerweise) ohne menschlichen Eingriff an neue Systemzustände anpassen.

- Agenten lassen sich auf kostengünstiger Hardware realisieren, was eine weite Verbreitung, insbesondere im Haushaltsbereich, begünstigt.

Aus den oben aufgeführten Vor- und Nachteilen von Multi-Agentensystemen lässt sich deren Eignung für eine nahtlose Restrukturierung des Energiesystems von seinem heutigen Zustand in ein noch stärker verteiltes und vernetztes System erschließen.

3.1. Systemstrukturen und Architekturen

Die Effizienz und Effektivität eines Steuerungssystems wird unter anderem von dessen Architektur beeinflusst. Es sollen daher im Folgenden bekannte Architekturen für Steuerungssysteme dargestellt und diskutiert werden. Der Definition von Rohbogner [RFH+12] folgend, soll dabei die Führungsstruktur vom Führungskonzept unterschieden werden. Nimmt man beide zusammen, ergeben sich die im Folgenden näher betrachteten Architekturen für Steuerungssystemen. Abbildung 3.1 illustriert die unterschiedlichen Architekturen anhand dreier plakativer Beispiele.
Die Führungsstrukturen sind dabei durch eine oder mehrere Führungseinheiten charakterisiert. Die Ziele der Führungseinheiten bilden die Basis des Führungskonzepts. Im Gegensatz

dazu wird die Organisation der Führungseinheiten mit ihrer Einteilung in verschiedene Ebenen der Informationsverarbeitung, Optimierung und Datenhaltung als Führungsstruktur bezeichnet. Im Folgenden sollen unterschiedliche Führungsstrukturen diskutiert werden, welche im Rahmen unterschiedlicher Führungskonzepte angewendet werden können.

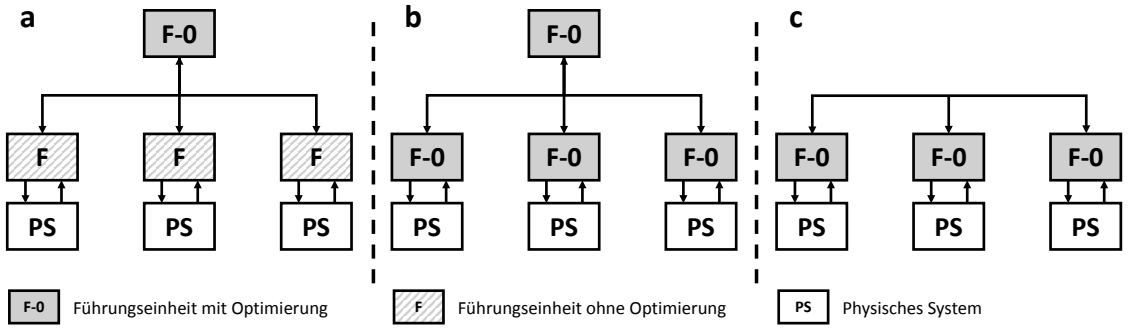

Abbildung 3.1.: (a) Zentral-hierarchische, (b) dezentral-hierarchische und (c) dezentrale Führungsstruktur, nach [BDL+13]

3.1.1. Zentral-hierarchische Führungsstruktur

Eine wichtige Form der Führungsstrukturen ist die Zentral-Hierarchische. Insbesondere die hohen Sicherheitsanforderungen im Kontext elektrischer Netze und die gut definierbaren Entscheidungsprozesse und Verantwortlichkeiten dieser Struktur führen zu einer weiten Verbreitung. Zentral-hierarchische Systeme besitzen mindestens zwei Ebenen. Führungsentscheidungen werden auf der höchsten Ebene getroffen und dann durch unterlagerte Ebenen umgesetzt. Häufig werden die unteren Ebenen dabei auf reine Sensor- und Aktortätigkeiten reduziert. Frühe Forschungs- und Implementierungsprojekte im Bereich der Erneuerbaren Energien nutzen diese Struktur. Beispielhaft dafür steht das *Kombikraftwerk 1* [Fra14], welches unterschiedliche Energieerzeuger vernetzt und durch einen zentralen Controller steuert. Die im Feld verteilten Agenten dienen in diesem Projekt lediglich der Erfassung von Daten und Umsetzung der zentral ermittelten Sollwerte.

3.1.2. Dezentral-hierarchische Führungsstruktur

Die dezentral-hierarchische Führungsstruktur erweitert die vorgenannte Form um eine verbesserte Informationsverarbeitung auf den unteren Ebenen der Führungseinheiten. Die dezentral verorteten Fähigkeiten können sowohl Algorithmen zur Optimierung als auch einen Teil der Datenhaltung umfassen. Die Kommunikation läuft jedoch stets über einen zentralen Knoten. Die hierdurch entstehende Schwachstelle eines „Single Point of Failure“ wird durch die Vorteile einer besseren Prüf- und Steuerbarkeit der Kommunikationsflüsse kompensiert. Auch Agentensysteme können in dieser Form umgesetzt werden. Ein Beispiel dafür ist das Projekt *DEMAPOS*. Hier verhandeln dezentrale Einheiten in einer hierarchischen Struktur über Strompreise [LWSF11]. So kann das System komplett dezentral geführt werden, bietet jedoch zu jedem Zeitpunkt die Möglichkeit durch einen zentralen Knotenpunkt Korrekturen einzubringen.

3.1.3. Dezentrale Führungsstruktur

Die komplett dezentrale Führungsstruktur verzichtet im Gegensatz zur dezentral-hierarchischen Struktur auf den oben beschriebenen zentralen Knoten. Alle Einheiten des Systems können somit unabhängig von ihrer Ebenen- oder Linienzugehörigkeit miteinander kommunizieren. Diese Struktur ist damit mit den grundlegenden Agentenkonzepten vergleichbar. Alle Agenten arbeiten dabei im Kollektiv zusammen. Individuelle Fähigkeiten und Wissensstände werden mit der Gemeinschaft geteilt um ein gemeinsames, höheres Ziel als den reinen individuellen Vorteil zu erreichen.

3.2. Vorteile und Herausforderungen

Die Anwendungsfälle von softwarebasierten Steuerungen im Bereich der Energieleittechnik sind vielfältig und reichen von einfacher Temperatursteuerung in der Gebäudeautomatisierung bis hin zur Detektion und Eindämmung harmonischer Schwingungen in Stromnetzen. Es wird daher eine Koexistenz unterschiedlicher Architekturen von Steuerungssystemen mit vielfältigen Führungsstrukturen geben, welche stets an die Anforderungen des individuellen Optimierungsziels angepasst werden müssen. Das folgenden Beispiel soll die Vorteile der oben dargestellten Strukturen verdeutlichen:
Bis vor wenigen Jahren war die Gliederung eines Niederspannungsnetzes von einer hohen Konstanz geprägt. Zentral hierarchische Strukturen konnten so Daten aus ihnen bekannten, darin eingebetteten Führungseinheiten sammeln und auswerten. Zentrale Optimierer waren aufgrund der Verfügbarkeit einer umfassenden Systembeschreibung sowie der geringen Komplexität des Problems in der Lage alle relevanten Entscheidungen zu treffen um ein globales Optimum zu erreichen. Einziger Nachteil war dabei der hohe Kommunikationsaufwand zwischen Sensoren, Aktoren und zentraler Steuerung.
In Hinblick auf die Herausforderungen der Energiewende wurden dann in einigen *Smart-Grid*-Projekten, welche in Abschnitt 3.3 beschrieben werden, dezentral hierarchische Strukturen zur Steuerung von Niederspannungsnetzen implementiert. Diese ermöglichen in den konkreten Implementierungen die dezentrale Speicherung und Vorverarbeitung von Daten, was zum einen die Datensicherheit erhöht, die notwendige Bandbreite reduziert und eine Abstützung der Entscheidungsprozesse auf abstrakte Führungsgrößen ermöglicht. Darüber hinaus ist eine höhere Anpassungsfähigkeit auf Änderungen der Regelstrecke erzielbar. Die Installation von Solaranlagen oder Ladesäulen für Elektrofahrzeuge im Niederspannungsnetz kann so, ohne Anpassungen an der Software des zentralen Entscheidungsknotens, umgesetzt werden. Das Netz wird somit „Plug & Play“fähig. Diese Vorteile werden jedoch durch eine potentiell höhere Latenz der Kommunikation erworben, welche sich durch die Vorverarbeitung der Daten begründet. Der Nachteil des zentralen Flaschenhalses und des „Single Point of Failure“bleibt weiterhin bestehen.
Diese letzte Einschränkung lässt sich durch Einsparung der zentralen Einheit umgehen. Da Nachrichten dann jedoch ungerichtet an eine Vielzahl von Führungseinheiten gesendet werden müssen, bedarf es wesentlich höherer Datenübertragungsraten in den genutzten Kommunikationsnetzen. Koordinationsprozesse, wie zum Beispiel Zeitsynchronisation oder auch die Formung von Koalitionen zur Bewältigung bestimmter Aufgaben wie der Änderung eines lokalen Spannungsniveaus im Strang des Niederspannungsnetzes, werden dann durch Broadcast-Nachrichten eingeleitet, welche auch Agenten erreichen (und somit beschäftigen),

die in diesen Prozessen keine Aufgabe erfüllen. Aufgrund der Robustheit gegenüber Ausfällen unterschiedlicher Komponenten des Steuerungssystems und der Änderung der Regelstrecke ist die dezentrale Struktur insbesondere für sich im Wandel befindende und zunehmend kleinskalige Systeme mit einem hohen Grad an Digitalisierung jedoch ideal.

3.3. Literatur über Agenten in der Energieleittechnik

Literatur zu agentenbasierten Ansätzen kann, in Ergänzung zu den oben diskutierten Führungsstrukturen, in die drei folgenden Kategorien unterteilt werden:

- Bilanzielle Optimierung, welche eine ökonomische Optimierung von Erzeugungs- bzw. Nachfrageverbünden behandelt und meist klassischen wirschaftlichen Optimierungsansätzen folgt.
- Netzorientierte Optimierung, welche Betriebsführungskonzepte für dezentral geprägte Netze entwickelt.
- Inhouse Optimierung, welche meist Teil eines der vorgenannten Ansätze ist und den Komfort des Endverbrauchers im Blick hat.

Alle drei Ansätze gewannen mit dem Aufkommen des „Agenten-Hypes“ um die Jahre 2000 / 2001 an Bedeutung. Seitdem zeigt die stark gewachsene Zahl an Publikationen, dass sich das Konzept des Agenten im Bereich der Energietechnik weiter etablieren konnte. Abbildung 3.2 zeigt die Anzahl an Publikationen der Jahre 1997 bis 2017, welche auf der Plattform *Google Scholar* für den Suchbegriff „„Software Agent“ Energy“ indiziert werden. Mögliche Ursachen für die Stagnation, beziehungsweise den Abfall der Publikation seit 2013 können nur vermutet werden. Denkbar wären dabei zum einen der gestiegene Reifegrad der Technologie, welche eine Verlagerung aus dem Bereich der Forschung hin zur Implementierung bedeuten würde. Des weiteren könnten Forschungskapazitäten auf neue Technologien und Konzepte wie Big-Data gelenkt werden und somit ein Absinken der Publikationszahlen erklären.
Eine kleine Auswahl der großen Fülle an Ansätzen und Publikationen soll im Weiteren dargestellt werden.
Als einer der ersten marktbasierten bilanziellen Ansätze setzte *PowerMatcher* [KWK+05] bereits in 2005 Maßstäbe in Hinblick auf Dezentralisierung und Autonomie. In *PowerMatcher* vertreten individuelle Anlagenagenten die mit ihnen verbundenen Energieerzeuger und -verbraucher auf einer gemeinsamen Auktionsplattform um so Angebot und Nachfrage von Elektrizität in Waage zu halten. Nach einer späteren Erweiterung kann *PowerMatcher* auch weitere Optimierungsziele, wie zum Beispiel koordiniertes Handeln von Anlagenverbünden wie virtuellen Kraftwerken, verfolgen. Der niederländische Netzbetreiber *Alliander* unterstützt die Weiterentwicklung von *PowerMatcher*, welches in Feldversuchen sein Potential beweisen konnte [KRB+11].
Parallel zu *PowerMatcher* wurde der *DEZENT*-Ansatz entwickelt und in 2006 veröffentlicht [WLHK06]. Wie auch in *PowerMatcher* vertreten in *DEZENT* virtuelle Agenten physikalische Anlagen auf digitalen Marktplätzen, um so einen möglichst guten Ausgleich von Erzeugung und Verbrauch zu erzielen. Die Besonderheit von *DEZENT* ist die Einrichtung sogenannter Bilanzgruppenmanager für einzelne Netzabschnitte wie Niederspannungsstränge und Mittelspannungsnetze. Der lokale Austausch von Energie soll dabei sowohl technische,

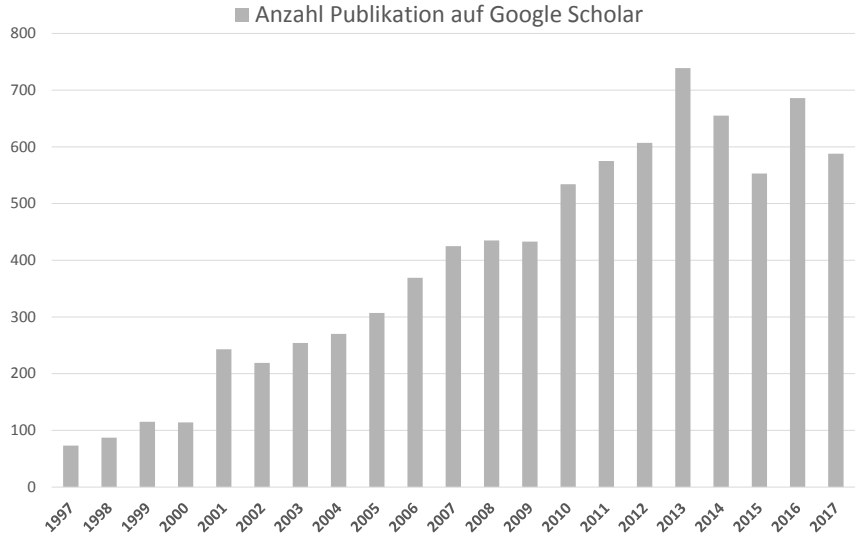

Abbildung 3.2.: Anzahl an Publikationen zum Thema „„Software Agent“Energy“auf Google Scholar von 1997 bis 2017

ökonomische als auch ökologische Optimierung ermöglichen [Leh10] [LKRW11].
Aufbauend auf den Erfahrungen mit *PowerMatcher* und *DEZENT* wurde in 2011 *DEMAPOS* vorgestellt [LWSF11]. Neben der Berücksichtigung lokaler Effekte wie der Spannungshaltung oder dem Management kleiner Bilanzkreise wurde hier erstmals auch die global beeinflussbare Frequenz mit in die Optimierung einbezogen. Es erfolgte dabei zunächst keine Vorausplanung, alle Austauschvorgänge wurden - wie auch bei *DEZENT* und *PowerMatcher* - für die jeweils aktuelle Periode (in Anlehnung an *DEZENT* mit einem Intervall von 500 ms) vereinbart. Eine spätere Erweiterung ermöglichte dann die Einbeziehung von Vorhersagen, was das System insbesondere für Stabilitätsuntersuchungen interessant machte [LWF13].
Die Vorteile der lokalen Regelung zur Spannungshaltung nutzte auch Richardot in einem Ansatz, welchen er 2009 in seiner Dissertation vorstellte [Ric06]. Hier beschrieb er einen dezentral hierarchischen Ansatz zur Blindleistungsregelung mittels verteilter Anlagen. Richardot sah hierfür, ähnlich wie in den oben beschriebenen Systemen, lokale Verbünde vor, welche durch individuelle Messpunkte die für ihre Abschnitte erforderliche Blindleistung ermitteln und mit den ihnen zur Verfügung stehenden Betriebsmitteln bereitstellen [RBRH09].
Troschel [Trö10] legt in seinem holonischen Energiemanagementsystem den Fokus auf eine erhöhte Zuverlässigkeit der klassischen Einsatzplanung von aggregierten Energieerzeugern, wie sie in virtuellen Kraftwerken zu finden sind. Dabei verknüpft er sowohl reaktive als auch prädiktive Einsatzplanung, um auch nach einem Ausfall einzelner Anlagen mittels dynamischer Reorganisation die vom Verbund eingegangenen Lieferverpflichtungen weiter erfüllen zu können.
Das in 2007 von Nestle vorgestellte *BEMI* [Nes07] setzt den Fokus ebenfalls auf die lokalen Regelung und damit auf den Netzanschluss als Schnittstelle zwischen öffentlichem Netz und Kunden. Das Agentensystem, welches um diese Schnittstelle herum entwickelt wurde, soll dabei zum einen die Hoheit des Kunden über seinen eigenen Stromverbrauch wahren, aber auch die Interessen des Netzbetriebes berücksichtigen. Es ist somit ein Hybridsystem zwischen Inhouse Management und Netzoptimierung.

Neben der Beeinflussung von Erzeugung und Verbrauch, wie es fast alle vorgenannten Ansätze anstreben, können auch Anlagen wie FACTS-Geräte oder Querregler genutzt werden, um die Leistungsflüsse in Netzen zu beeinflussen. Häger stellte 2011 einen dezentralen Ansatz vor, welcher mittels dieser Leistungsflussregler die Spannung und Impedanz eines Übertragungsnetzes beeinflusst, um so dessen Stabilität zu gewährleisten [HLR11].
Rohbogner stellte 2014 erstmals ein Agentensystem zur Spannungshaltung in Niederspannungsnetzen vor, welche komplett dezentral, also ohne zentrale *White-* und *Yellowpage Services*, funktioniert [RFB+14]. Um das Fehlen der zentralen Steuerungs- und Informationsorgane auszugleichen, legen die von ihm beschriebenen Agenten interne Adressbücher an und führen eine Lokalisierung im einzelnen Strang durch einen Vergleich der individuell gemessenen Spannungen durch.
Neben der Optimierung elektrischer Energienetze tritt auch die Integration anderer Energieträger in den Vordergrund. Die Optimierung von Fernwärmenetzen ist dabei ein wichtiger Innovationstreiber. Gulzar beschäftigte sich 2015 mit diesem Thema und der dezentralen Optimierung eines Netzwerkes mit einem zentralem Blockheizkraftwerk in Kombination mit dezentralen solarthermischen Anlagen und Wärmespeichern [GSV+15].
Wie dieser Überblick verdeutlicht, ermöglicht das Konzept der Software-Agenten Architekturen mit unterschiedlichen Führungskonzepten und Fürungsstrukturen zu entwickeln. Es können dabei sowohl Teilprobleme als auch gesamtheitliche Lösungen im Themenkomplex der bilanziellen-, netzorientierten- oder Inhouse-Optimierung betrachtet werden.

3.4. Kompatibilität agentenbasierter Steuerungen

Die im vorangegangenen Abschnitt dargestellte Vielfalt an Ansätzen und Implementierungen wird in der weiteren Entwicklung der Energienetze zu einer hohen Heterogenität der Leitsysteme führen. Diese bietet große Chancen, aber auch erhebliche Risiken. So ist die Angreifbarkeit als auch die Resilienz voneinander unabhängiger Steuerungen höher als es bei zentralen Elementen der Fall wäre. Doch können Interferenzen inkompatibler Systeme auch zu einer Verstärkung negativer Effekte führen. Das Endkundensegment kann hier als gutes Beispiel dienen. Die Einführung der mit dem Internet verbundenen Geräte und der Etablierung des sogenannten Internet of Things (IoT) erfolgt aktuell ohne eine Abstimmung der einzelnen Akteure. Proprietäre Protokolle, inkompatible *Smart-Home-Systeme* und Haushaltsgeräte führen zu einer Ablehnung durch den Nutzer. Um dies in der weiteren Energieversorgung zu vermeiden und so die Versorgungssicherheit nicht zu gefährden, müssen Methoden und Werkzeuge etabliert werden, welche die Elemente der zukünftigen Energieversorgungssysteme abbilden und so deren Kompatibilität verifizieren können.
Insbesondere agentenbasierte Simulationen eigenen sich für dieses Vorhaben, da sie die zu verifizierenden Szenarien in einem Agentenmodell, und somit dezentral, abbilden. Wie in der Einleitung dieses Kapitels beschrieben, ermöglicht die agentenbasierte Modellierung die an einem Problem beteiligten Einheiten, in Anlehnung an bekannte Muster sozialer Wesen beziehungsweise technischer Anlagen, abzubilden. Diese können sich dann gegenseitig sowie eine gegebenenfalls simulierte Umgebung, in welche sie eingebettet sind, beeinflussen [DVM03]. Dies ermöglicht das Schlussfolgern aus dem Verhalten der Simulation auf reales Systemverhalten und somit die Evaluation des dezentralen Steuerungssystems. Dies kann, im Gegensatz zu konventionellen Simulationsmethoden, in nahezu beliebigem Detaillierungsgrad geschehen, was auch die Abbildung komplexer dezentraler Energieversorgungsstrukturen

ermöglicht. Die Analyse und der Vergleich der beteiligten Komponenten im Kontext eines Gesamtsystems wird somit ermöglicht [Klü08].
Die damit verbundenen Aufwände sind mit denen zur Entwicklung eines Agentensystems vergleichbar, welches in der Realität eingesetzt werden soll. Fragen nach der notwendigen Detailtiefe und der damit verbundenen Abbildung einzelner Verhaltensweisen individueller Akteure stehen dabei am Anfang des Prozesses. Dabei kann, ähnlich der Ausrichtung der Steuerungssysteme, eine Unterscheidung in ökonomische und technische Simulationen vorgenommen werden. Ökonomisch orientierte Simulationen fußen dabei meist auf deliberativen Agentensystemen, während technische Simulationen aufgrund der Forderung nach geringen Latenzen meist reaktiv ausgelegt werden. Nach diesen initialen Schritten müssen dann die simulierten Komponenten und deren virtuelle Umgebung geschaffen werden [DU11].

3.5. Zwischenfazit

In diesem Kapitel wurde der Term Agent eingeführt und seine Bedeutung im Kontext der Informatik und Automatisierungstechnik dargestellt. Ein Agent wurde dabei in Anlehnung an die Arbeiten von Wooldridge und Jennings sowie in Übereinstimmung mit dem VDI-GMA-FA 5.15 als ein Softwareartefakt beschrieben, welches in eine (Hardware-)Umgebung eingebettet ist und dort autonom agieren kann um individuelle Ziele zu verfolgen. Zu diesem Zweck ist das Softwareartefakt darüber hinaus in der Lage mit anderen Agenten zu kommunizieren und zu kooperieren.
Es wurden im Anschluss drei typische Führungsstrukturen für Agentensysteme vorgestellt:

- Zentral-hierarchisch - welche insbesondere für kleine Systeme mit schneller Reaktionszeit geeignet ist
- Dezentral-hierarchisch - welche den aktuellen Implementierungen von Agentensystemen in der Energieleittechnik entspricht und eine Balance zwischen Skalierbarkeit und Reaktionszeit sicherstellt.
- Dezentral - welche eine sehr gute Skalierbarkeit auf Kosten der Latenz aufweist.

Diese dienten, neben der Ausrichtung auf wirtschaftliche oder technische Optimierung, der Differenzierung der diskutierten Implementierungen und wissenschaftlichen Arbeiten. Die präsentierte Auswahl relevanter Arbeiten, welche den aktuellen Stand der Forschung abbilden, veranschaulichte, dass Agentensystem sich grundsätzlich für die Steuerung zukünftiger Energienetze eignen. Sie müssen jedoch, insbesondere in der Implementierungs- und Verifikationsphase, methodisch untermauert werden, um Inkompatibilitäten und daraus resultierende Interferenzen zu vermeiden. Die Kompatibilität der in Energiesystemen eingesetzten Agenten kann durch die Definition einer allen Systemen gemeinsamen Softwarestruktur erhöht werden. Zu diesem Zweck wurde das Konzept des Energie-Agenten etabliert.

4. Komponenten des Energie-Agenten

Der Begriff des Energie-Agenten dient der Umschreibung eines Softwarekonzeptes, wie es in Energienetzen der Zukunft Anwendung finden kann. Das Konzept wurde durch Christian Derksen und den Autor dieser Dissertation entwickelt [DL13]. Es baut auf dem ersten Hauptsatz der Thermodynamik auf und erweitert diese rein physikalische Perspektive um wirtschaftliche Beschreibungsmittel. Eines der Kernelemente dieses Ansatzes ist die Nutzung eines Agentensystems, dessen Individuen - die Energie-Agenten - die Steuerung der Systemelemente verantworten. Bei den Energie-Agenten handelt es sich um Software im Sinne der Agententechnologie, die auf den Einsatz in dezentralen Energiesystemen spezialisiert ist. Dezentralität und Autonomie der Energie-Agenten ermöglichen es, verteilte Automatisierungslösungen mit hoher Ausfallsicherheit und schneller Reaktionszeit zu entwickeln.
Im Folgenden soll das Konzept des Energie-Agenten erläutert und seine Relevanz für die Entwicklung von Software für die Energiewirtschaft dargelegt werden.

4.1. Definition und Struktur des Energie-Agenten

Energie-Agenten stellen Softwareartefakte dar, welche in unterschiedlichen technischen Systemen Anwendung finden können. Abhängig von der Komplexität des zu steuernden Systems und dem Grad der Vernetztheit werden unterschiedliche Reifegrade des Agenten unterschieden. Eine entsprechende Klassifizierung wird nach einer Einführung in die Grundlagen des Konzeptes dargestellt.
Energie-Agenten, wie sie im Folgenden beschrieben werden, üben nur zu einem gewissen Grad Kontrolle über die ihnen unterlagerten technischen Systeme aus. Der notwendige Grad an Kontrolle ist vom individuellen System abhängig. Es ist eine Grundannahme, dass Energie-Agenten in der Regel mit der eigentlichen Steuerung des Systems kommunizieren und nicht in die niedrigen Ebenen der Systemsteuerung wirken. Somit beeinflussen die Energie-Agenten lediglich Stellgrößen und nicht z.B. die konkrete Stellung eines Ventils oder andere sicherheitskritische Funktionen. Am Beispiel eines Wechselrichters lässt sich dies verdeutlichen. So werden die Funktionen der Frequenzhaltung oder die Überwachung der Gerätetemperatur weiterhin durch die interne Logik des Wechselrichters realisiert. Die Sollgrößen der eingespeisten Leistung oder der Phasenlage können jedoch durch den Agenten vorgegeben werden.
Ausgehend von dieser Ebene, welche mit Supervisory Control and Data Acquisition (SCADA)-Systemen in der Automatisierungspyramide vergleichbar ist, müssen sich Energie-Agenten bewusst sein, wie viel Energie von den technischen Systemen verbraucht, produziert oder gespeichert wird und welche Stellgrößen beeinflusst werden können, um spezifische Verbrauchs- oder Erzeugungsmuster zu generieren. Zwecks Durchsetzung einer entweder wirtschaftlich oder technisch motivierten Betriebsführungsstrategie müssen Energie-Agenten den Betriebszustand und Betriebsmodus der ihnen unterlagerten technischen Systeme erfassen und analysieren können. Im Folgenden sollen diese technischen und ökonomischen Aspekte weiter diskutiert werden.

Die Grundlage zur Beschreibung aller technischen Systeme kann im ersten Hauptsatz der Thermodynamik gefunden werden. Dieses Naturgesetz besagt, dass Energie weder erzeugt noch vernichtet (verbraucht) werden kann. Es ist lediglich möglich Energie von einer Form in eine andere zu wandeln. Beschreibt man diesen Umstand mathematisch, bedeutet dies, dass die Summe der „verbrauchten“ und „erzeugten“ Energie E eines technischen Systems immer gleich 0 ist. Formel 4.1 fasst dies zusammen.

$$\sum E_i = 0 \tag{4.1}$$

Dieses Gesetzt gilt für jedes technische System, welches Energie von Form i in eine andere Form umwandelt. Beispiele hierfür sind elektrische Generatoren, welche mechanische Energie in Elektrizität umwandeln, oder Heizungssysteme, welche chemische Energie in thermische Energie wandeln. Der erste Hauptsatz der Thermodynamik ist sowohl für stationäre Prozesse (konstant über die Zeit) als auch für instationäre Systeme anzuwenden. Es umfasst alle Energieformen wie zum Beispiel Elektrizität, Wärme, chemische Energie (z.B. Erdgas) oder mechanische Energie. Nicht technisch genutzte Energieumwandlungsprozesse - also „Verluste“- werden im Rahmen der Betrachtung ebenfalls mit einbezogen. Hierdurch wird sichergestellt, dass die Summe aller Energieflüsse über die Grenzen eines technischen Systems immer gleich 0 ist. Die damit einhergehenden Effizienzen eines technischen Systems lassen sich so berechnen [AB06].
Die Energieflüsse über Systemgrenzen werden in Watt gemessen, die Menge an Energie wird in Wattsekunden und ihrer SI-Einheit Joule, beziehungsweise der weit verbreiteten Einheit Kilowattstunden, angegeben. Die Beziehung zwischen Energiefluss $\dot{E}$ und der Menge an Energie E eines spezifischen Energieträgers c_i kann mathematisch als Integral über die Zeit beschrieben werden:

$$E_{c_i} = \int_{t_1}^{t_2} \dot{E}_{c_i}(t)dt \tag{4.2}$$

Diese Formel kann genutzt werden, um den Energieaustausch eines technischen Systems an der Schnittstelle zu einem spezifischen Energienetz zu beschreiben. Um individuelle technische Systeme in Hinblick auf ihre Auswirkungen auf ein Netzwerk zu beschreiben und zu analysieren, ist diese Darstellungsform bereits ausreichend. Alle benötigten Parameter zu Energieverbrauch und -erzeugung können auf Basis dieser Daten bewertet werden. Aus der internen Sicht eines technischen Systems heraus mag dies jedoch nicht zutreffen, da gegebenenfalls auch Verlust- und Speicherprozesse betrachtet werden müssen. Diese können unter Zuhilfenahme der folgenden Formel berücksichtigt werden:

$$E_{c_i ein} + E_{c_i aus} + E_{c_i verlust} + E_{c_i gespeichert} = 0 \tag{4.3}$$

Diese Formel ermöglicht die Untersuchung eines technischen Systems in Hinblick auf seine Energieströme unter Berücksichtigung der domänenspezifischen Energieform c_i und eventuell vorhandener Speicherkapazitäten. Hieraus ergibt sich ein breites Spektrum an Anwendungsfeldern, welches über den Bereich elektrischer Systeme hinaus geht und auch technische Systeme anderer Energiedomänen wie zum Beispiel Kompressorstationen, Rohrleitungen, Heizungssysteme und viele mehr umfasst. Darüber hinaus können Energiespeicher aller Art beschrieben und in eine ganzheitliche Betrachtung integriert werden. Hierzu zählen Anlagen wie elektrische Akkumulatoren, Kavernen zur Speicherung von Erdgas oder Wärmespeicher

für Heizungssysteme.
Technische Verluste treten in allen Energieumwandlungsprozessen und während des Transports von Energie auf. Es kann sich dabei zum Beispiel um widerstandsbedingte Verluste in elektrischen Netzen, Wärmeverluste in Verbrennungsmaschinen oder Druckverluste während des Transports von Erdgas handeln.
Auf Basis dieser Verluste lassen sich weitere Faktoren wie die Effizienz und Kosten eines Energiewandlungsprozesses ermitteln. Das Verhältnis der die Systemgrenzen überschreitenden ein- und ausgehenden Energieströme ist dabei von der Effizienz eines technischen Systems abhängig und bestimmt dessen Betriebskosten und damit erzielbare Profite.
Die Berechnung dieser Parameter wird jedoch stark durch die gewählte Berechnungsmethode und Datenbasis geprägt. Die Komplexität des Ansatzes und die Anzahl der betrachteten Parameter muss dabei in Hinblick auf die verfügbare Rechenzeit und erwünschte Realitätsnähe abgewogen werden. Während einfache Abschätzungen der Betriebskosten eventuell nur die Berücksichtigung variierender Marktpreise erfordern, müssen tiefergehende Betrachtungen auch Faktoren wie Abnutzungserscheinungen und Amortisierungsraten einbeziehen [BD95].
In den folgenden Beispielen soll im Sinne der Verständlichkeit die vereinfachende Annahme getroffen werden, dass die Gesamtkosten oder Erlöse CR eines technischen Systems durch die Summenbildung aller zeitvariablen Kostenfunktionen im Verhältnis zu den individuellen Erzeugungs- und Verbrauchsprofilen berechnet werden können. Für eine Schnittstelle einer spezifischen Domäne kann dies mathematisch wie folgt beschrieben werden:

$$CR_{c_i} = \int_{t_1}^{t_2} \dot{E}_{c_i}(t) \cdot CR_{c_i}(t, \dot{E}_{c_i}(t))dt \tag{4.4}$$

Die zeitabhängige Kostenfunktion CR kann dabei als zeitlich veränderlich angenommen werden, da der Bezug, beziehungsweise die Erzeugung von Energie, an einem Anschlusspunkt stets variiert. Beispielhaft hierfür kann ein Hausanschluss mit verbundenem Photovoltaiksystem genannt werden. Während des Tages wird infolge des Erzeugungsüberschusses Energie in das öffentliche Netz eingespeist. Am Abend und in der Nacht kehrt sich aufgrund des erhöhten Energiebedarfs die Flussrichtung jedoch um.
In Hinblick auf Formel 4.3 bleibt jedoch die Frage nach Verlusten und eventuellen Speicherkapazitäten bestehen. Verluste sind dabei stark vom aktuellen Arbeitspunkt und den vorgegebenen Sollwerten abhängig. Am Beispiel eines Verbrennungsmotors, welcher bei verschiedenen Umdrehungszahlen betrieben werden kann und über den gesamten Drehzahlbereich unterschiedliche Energieverluste aufweist, kann dies gut verdeutlicht werden. Die aus den Verlusten resultierenden Kosten ergeben sich dabei aus den Beschaffungskosten der genutzten Energieform.
Die Kosten für den Energiefluss in einen Energiespeicher ergeben sich ebenfalls aus den Beschaffungskosten der eingespeicherten Energie. Abhängig vom Anwendungsfall können die Umsätze jedoch unterschiedlich berechnet werden. In der Domäne der Elektromobilität zum Beispiel wird die Energie vom technischen System selbst in kinetische Energie umgewandelt, wohingegen Großspeicher wie Kavernen zur Speicherung von Erdgas in Abhängigkeit von Marktpreisen zum Arbitragehandel eingesetzt werden.
Unabhängig davon, wie die Kosten in Verhältnis zur Energienutzung im Einzelfall stehen, ist es zielführend eine spezifische Kostenfunktion zu entwickeln. Die Nutzung solcher Kostenfunktionen erlaubt die Ausrichtung des Prozesses auf unterschiedliche Optimierungskriterien. Beispiele hierfür sind die Kostenreduktion bei Demand-Response Ansätzen, die Profitma-

ximierung im Rahmen der Elektrizitätserzeugung, das Senken von Treibhausgasemissionen eines Systems und andere eventuell sogar kombinierte Ziele. Die oben genannten Kriterien und Beispiele sollen verdeutlichen, dass es eine Vielzahl von Aspekten gibt, welche im Rahmen des Engineerings von Energieleitsystemen beachtet werden müssen. Im Zuge der Entwicklung des Energie-Agenten Ansatzes wurde daher dem Umstand der Diversität der betrachteten Problemfelder Rechnung getragen und keine Einschränkung auf spezifische Optimierungsziele vorgenommen. Der Ansatz soll ein *Tailoring* auf die spezifischen Herausforderungen individueller Problemstellungen ermöglichen. Die Definition des Energie-Agenten, welche die obigen Abwägungen berücksichtigt, der allgemeinen Definition von Agenten [WJ95] und der VDI-Richtlinie 2653-1 [VDI10] entspricht, wurde daher wie folgt publiziert und in Facharbeitskreisen wie dem VDI GMA FA 5.15 diskutiert und für vorteilhaft befunden:

Ein Energie-Agent ist ein spezialisiertes autonomes Softwaresystem, welches informations-technologisch ein einzelnes technisches Energiewandlungssystem repräsentiert und dabei die energetischen Freiheitsgrade in Bezug auf Energieerzeugung, -verbrauch und -speicherung bewirtschaftet. Dabei ist der Energie-Agent sowie das technische System gleichzeitig Bestandteil eines oder mehrerer Energieverteilungsnetze und in der Lage sowohl innerhalb dieser Netzwerke als auch mit externen Akteuren zu interagieren und zu kommunizieren.

Diese Definition soll einen abstrakten Prototypen des Energie-Agenten definieren, welcher in unterschiedlichen Energiedomänen und Anwendungsszenarien zum Einsatz kommen kann. Mit dem Konzept der Energiedomänen sollen dabei explizit nicht nur elektrische Netze betrachtet werden, sondern auch Gas-, Wärme-, Druckluft- und alle Energienetze, welche dem ersten Hauptsatz der Thermodynamik unterworfen sind, mit in die Optimierung der immer dynamischeren und anspruchsvolleren Netz- und Systemleittechnik einbezogen werden. Aufbauend auf diesen Zusammenhängen wird ein Energie-Agent zwar durch ein individuelles Stück Software repräsentiert, dieses ist jedoch mit einer Vielzahl anderer Energie-Agenten in den Übertragungs- und Umwandlungsanlagen der unterschiedlichsten Energiedomänen vernetzt.
Im Sinne des ersten Hauptsatzes der Thermodynamik und den charakteristischen Eigenschaften eines Energiesystems ist es die Hauptaufgabe eines Energie-Agenten, die existierenden Energieströme zu beschreiben und diese Informationen mit seinen Nachbarn auszutauschen. Diese grundlegende Anforderung führt zur Notwendigkeit eines generischen Datenmodells, welches den Austausch und eine spätere Optimierung auf Basis dieser Informationen erlaubt. Aufgrund der Zeitabhängigkeit von Energietransport und –wandlungsprozessen ist die Berücksichtigung dieses Faktors von großer Bedeutung. Aktuelle SCADA-Systeme nutzen hierfür diskrete Zeitmodelle mit anwendungsfallspezfischen Zyklenzeiten. Im Falle der Energiesysteme sind dies stark von der betrachteten Energieform abhängig. So bedarf es im Bereich elektrischer Energietechnik zum Teil einer Überwachung und Entscheidungsfindung innerhalb weniger Millisekunden. So können Beispielsweise Lichtbögen in Schaltanlagen in weniger als 10ms erkannt und eliminiert werden. Gasnetze erlauben aufgrund ihrer Trägheit hingegen höhere Latenzen. So können auch Zyklenzeiten in Minutenscheiben als probat erscheinen.
In Zusammenhang damit steht auch die Komplexität der zu steuernden Systeme. Diese wird für die Anwendung des Energie-Agenten formalisiert und in unterschiedliche Reifegrade abgestuft. Diese Reifegrade beschreiben die Fähigkeiten der Energie-Agenten, welche zur Steue-

Reifegrad	Art der Kontrolle	Beschreibung
R0	Zentral	Ursprüngliche Situation aus den 80er Jahren - z.B. Bakelit-Ferrariszähler oder neuere Zähler ohne Kommunikationsanbindung
R1	Zentral	Aktuelle Zählersysteme, welche eine Übermittlung der Verbrauchsdaten für eine zentrale Auswertung erlauben
R2	Zentral	Zukünftige Zählersysteme mit Vorhersage auf Basis lokal aggregierter Daten
R3	Zentral & Lokal	Erweiterte lokale Steuerung, welche lokale Systeme beeinflussen und auf externe Signale (z.B. Preissignale) reagieren kann
R4	Zentral, Verteilt & Lokal	Erweiterte Bereichssteuerung mit eingeschränkten, teilautonomen lokalen Systemen, welche dynamische Koalitionen formen und gemeinsame Optimierung betreiben können - z.B. in Kooperation mit einem Transformator, welcher ein Netzwerksegment überwacht
R5	Verteilt & Lokal	Komplett dezentrale Kontrolle der Energieumwandlung und -Verteilung

Tabelle 4.1.: Reifegrade der Energie-Agenten

rung individueller Anlagen vor Ort oder in der Cloud ausgeführt werden. Hiermit wird es möglich auch heterogene Systeme, welche aus unterschiedlich fortschrittlichen Subsystemen aufgebaut sind, zu modellieren und komplexe Interaktionen zwischen den Subsystemen untereinander und mit ihrer Umgebung zu realisieren. Tabelle 4.1 vermittelt einen Überblick über die definierten Reifegrade. Zur Veranschaulichung wurden Beispiele aus der Anwendungsdomäne der Stromzähler gewählt. Die hier beschriebenen Einschränkungen und Funktionalitäten können auch auf andere Energiedomänen und Anwendungsfälle projiziert werden.
Diese Klassifizierung veranschaulicht, dass das Konzept des Energie-Agenten sein Potential ausspielen kann, wenn zusätzliche Koordinationsprozesse auf lokaler Ebene benötigt werden. Dementsprechend sind Energie-Agenten ab dem Reifegrad R1 in einem realen Anwendungsfall einsetzbar. Für Simulationszwecke können Modelle des Reifegrades R0 auch in Energie-Agenten gekapselt werden. Diese dienen dann jedoch lediglich als Dummy im Rahmen einer Verifikation.
In den höheren Ebenen des Reifegradmodells ist die Anwendung des Energie-Agenten aufgrund des zunehmend dezentralen Charakters des Systems sinnvoll. So stellt der Reifegrad 5 aus heutiger Sicht ein Extrem dar. Um dieses zu erreichen müssen noch einige Hürden überwunden werden. Aufgrund der parallelen Forschungsaktivitäten im Feld der zunehmend softwarezentrierten Energieleittechnik ist daher davon auszugehen, dass parallele Ausprägungen softwarebasierter Lösungsansätze entstehen werden. Diese Diversifizierung bedeutet eine weitere Komplexitätsdimension bei der Betrachtung der Energienetze der Zukunft. Betrachtet man jedoch alleine die hohe Vielfalt an Systemen, welche bereits heute am Markt verfügbar sind und über Schnittstellen zu potentiellen Energie-Agenten verfügen, ergibt sich eine klare Häufung im Reifegrad 1. Ein Großteil der Haushalte in Deutschland ist immer noch nicht

mit einem *Smart-Meter* ausgestattet, welcher die Energieverbräuche digital kommunizieren kann. Große, automatisierte Kraftwerke optimieren eine oder mehrere Stellgrößen mittels einer lokalen Steuerung. Die dabei gewonnenen Informationen werden nur zum Teil an zentrale Einheiten übermittelt. Die Analyse der Daten erfolgt teilautomatisiert und offline - und nicht in Hinblick auf eine energetische Optimierung.
Diese unteren Reifegrade der Ebenen Null und Eins scheinen auf den ersten Blick nicht durch Agenten optimierbar zu sein. Betrachtet man diese Systeme jedoch als integrierte Einheiten eines komplexeren Szenarios, so können sie als eigenständige Modelle Bestandteil der Steuerungslogik sein. Diese Logik kann im Rahmen von Entscheidungsfindungsprozessen mittels Simulation der von ihr nicht beeinflussbaren Einheiten eine automatisierte Folgenabschätzung von Kontrollentscheidungen durchführen.
Auf den höheren Ebenen des Reifegradmodells werden neben der reinen Übermittlung von Messdaten auch Sollwerte, Steuerbefehle und Koordinationsvorgänge kommuniziert. Im Zuge der Entwicklung eines solchen Systems müssen die zugrundeliegenden technischen und wirtschaftlichen Ziele identifiziert, beschrieben und in den Kontext des Gesamtsystems gesetzt werden. Die dabei zu beachtenden Randbedingungen sind, ebenso wie die erlaubten Parameterbereiche, zu spezifizieren. Technische Grenzwerte der Anlagen sind im Kontext dieser Modellierung als solche zu beschreiben, da sie im Zuge der späteren Optimierung nicht überschritten werden dürfen.
Die Quantifizierung der Energieflüsse bedarf einer expliziten Berechnung der einzelnen Ströme, in Abhängigkeit der domänenspezifischen Energieform. Während es zum Beispiel für Elektrizitätsnetzwerke in vielen Fällen genügt Spannung und Strom zu betrachten, sind im Kontext von Gasnetzen weitere Faktoren wie die Gasqualität, der Druck, Temperaturen und Feuchtigkeit von Relevanz. Diese domänenspezifischen Informationen werden somit ebenfalls von Energie-Agenten benötigt, insbesondere wenn eine Optimierung über die Grenzen einer einzelnen Domäne hinweg durchgeführt werden soll.
Fasst man die oben aufgeführten Betrachtungen zusammen, ergibt sich die in Abbildung 4.1 dargestellte Struktur eines Energie-Agenten sowie der Abhängigkeiten der für eine Optimierung notwendigen Parameter.
Der Energie-Agent soll dabei auf einer zusätzlichen Hardware – lokal oder auch in der Cloud – ausgeführt werden und das verbundene technische System teilweise oder vollständig überwachen oder steuern. Unter Zuhilfenahme des domänenspezifischen Wissens über das technische System und sein Zeitverhalten kann der Energie-Agent ein Kosten- & Optionsmodell generieren und dieses für Kostenoptimierung und Einhaltung der Randbedingungen nutzen. Eine Erweiterung der Fähigkeiten des Energie-Agenten, wie in Reifegrad drei beschrieben, erlaubt darüber hinaus die Verarbeitung und Kommunikation externer Preissignale sowie die Koordination mit benachbarten Einheiten. Abbildung 4.1 stellt die hierfür notwendige Struktur und Prozesse dar.
Das oben erwähnte Kosten & Optionsmodell, wird im folgenden Abschnitt erläutert und einzelne Aspekte näher diskutiert.

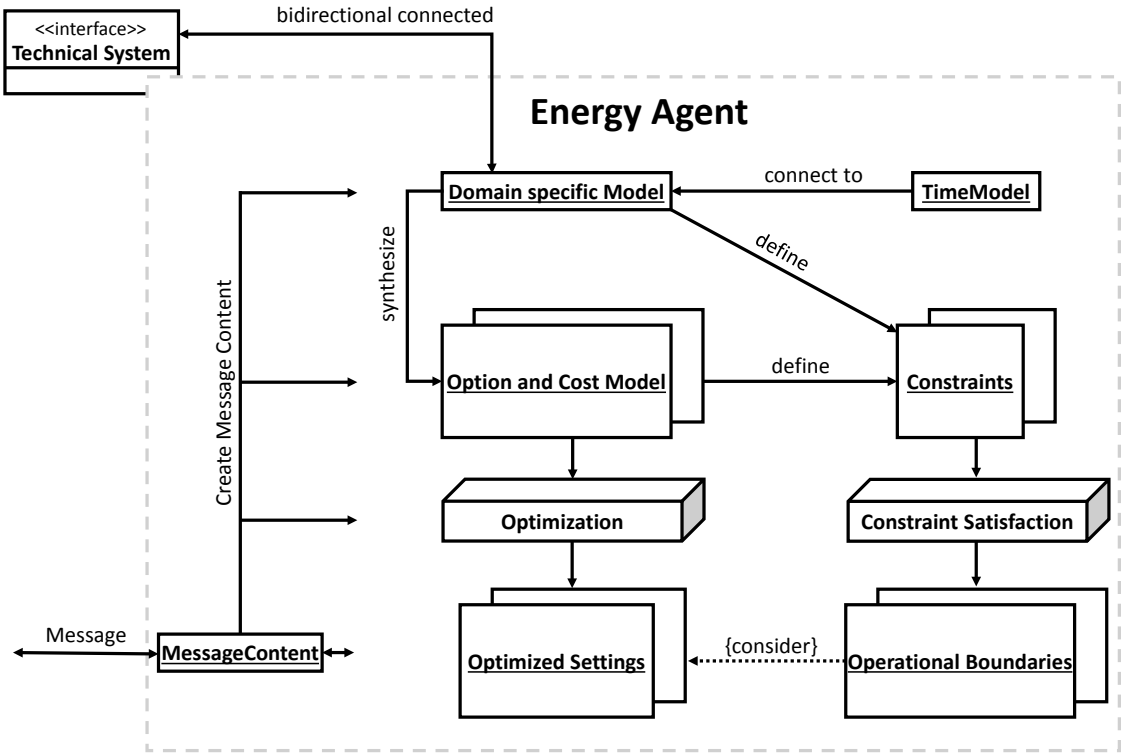

Abbildung 4.1.: Struktur, Schnittstellen und Prozesse eines Energie-Agenten des Reifegrades 3 und höher

4.2. Kosten & Optionsmodell

Im Gegensatz zu Beispielen aus dem Bereich der *Smart-Grid*-Forschung, welche sich primär auf elektrische Energie fokussieren, erlaubt es das generische Energie- & Optionsmodell (EOM) unterschiedliche Energieformen zu modellieren sowie deren Verbrauchs-, Produktions-, und Transportprozesse abzubilden.
Wie in Formel 4.3 dargestellt, variiert die Anzahl und der Umfang ein- und ausgehender Energieflüsse zwischen technischen Systemen erheblich. Während elektrische Verbraucher, wie elektrische Heizungen, lediglich eine Energieform in eine andere wandeln, gibt es Systeme, welche mit mehreren Energiedomänen an ihren Ein- und Ausgangspunkten verbunden sind. Hierzu zählen zum Beispiel Blockheizkraftwerke (BHKWs), welche chemische Energie sowohl in Elektrizität als auch in Wärme wandeln. Es soll daher im Folgenden dargestellt werden, wie individuelle Verbindungen technischer Systeme mit ihrer Umgebung in Hinblick auf Energieflüsse über die Zeit beschrieben werden können. Besitzt ein technisches System unterschiedliche Betriebsmodi wie Notlauf, Wartungs- und Normalbetrieb und Betriebsphasen wie Hochfahren, Betrieb und Herunterfahren, wird sowohl nach den Anschlusspunkten als auch nach den Betriebsphasen differenziert.
Betrachtet man einen individuellen Verbindungspunkt zwischen einem technischen System und einem Netzwerk, kann der beobachtbare Energiestrom idealisiert mittels Formel 4.2 als Integral einer Funktion über die Zeit beschrieben werden. Regelmäßig ist eine derart idealisierte Beschreibung des Verbrauchs- oder Erzeugungsverhalten jedoch nicht vorhanden. Wenn überhaupt, sind empirische Daten verfügbar, welche auf Basis von Messreihen

ermittelt wurden. Weiße Ware, wie zum Beispiel Spülmaschinen oder Waschmaschinen, bei welcher der interne Programmablauf zu einer fluktuierenden Energieaufnahme führt, welcher von der Auswahl des konkreten Programms durch den Nutzer und den sich daraus ergebenden Wasch- und Heizzyklen abhängig ist, gehören in diese Kategorie. Andere technische Systeme, wie klassische dampfbetriebene Kraftwerke, sind hingegen in Abhängigkeit ihrer Betriebsphase mathematisch gut beschrieben. Die für diese Prozesse vorliegenden Formeln können jedoch sehr komplex sein und somit für Betrachtungen umfangreicher Systemverbünde zu zeitintensiven Berechnungen führen. Es werden daher vereinfachte mathematische Modelle zur Beschreibung genutzt. Steuerungssysteme für thermische Großkraftwerke beispielsweise arbeiten mit prozentualen Anteilen der Maximalleistung als Sollwertvorgaben. Dynamische Änderungen werden als Delta der Sollleistung über der Zeit beschrieben. Somit wird der komplexe Prozess in eine lineare Beschreibung mit definierten Minima und Maxima transformiert, welche durch den Anlagenbediener einfach zu überwachen ist. Die Aufgabe der Steuerung ist es dann, diese Werte in Signale für die unterlagerten Steuerungssysteme zu wandeln.
Es wird somit zusammenfassend festgestellt, dass für ein einheitliches Modell von Energieflüssen sowohl empirische als auch mathematische Beschreibungen möglich sein müssen. Darüber hinaus wird es als notwendig erachtet, weit verbreitete, standardisierte Beschreibungsmethoden zu nutzen. Dabei muss ein Kompromiss zwischen Komplexität und Genauigkeit der Beschreibung gefunden werden, der eine Entscheidungsfindung in einem angemessenen Zeitraum ermöglicht. Die Bewertung dessen, was als „angemessen“ betrachtet wird, ist dabei stark von der Energiedomänen-spezifischen zeitlichen Auflösung der Steuerungsprozesse abhängig.
Die Berücksichtigung dieser Zeitintervalle ist auch in Hinblick auf die Beschreibung des Verhaltens eines technischen Systems in seinen spezifischen Betriebsphasen von hoher Relevanz. Darüber hinaus müssen weitere logische- und betriebliche Zusammenhänge berücksichtigt werden, was wiederum zu weiteren Kenngrößen in der Beschreibung der Betriebsphase führt. Diese sollen im Folgenden diskutiert werden.
Eine grundlegende Annahme im Rahmen der weiteren Betrachtung ist die Tatsache, dass eine Betriebsphase nicht zwingenderweise ein definiertes Ende besitzen muss. So können kontinuierliche Prozesse grundsätzlich über den Betrachtungshorizont hinaus, theoretisch sogar unbegrenzt laufen. Batch-Prozesse oder empirische Daten, welche in einem begrenzten Zeitraum aufgenommen wurden, haben jedoch definierte Anfangs und Endzeiten. Um diese beiden Fälle abbilden zu können wird im Rahmen der Modellierung auf ein flexibles Graph-basiertes Beschreibungsmittel zurückgegriffen. Betriebsphasen werden dabei als Knoten und die Übergänge zwischen den Phasen als Kanten abgebildet. Dieses Modell erlaubt es Verweilzeiten t_d in einem Knoten des Graphen zu definieren. Die Übergänge zu den folgenden Knoten erlauben die Auswahl mehrerer Zielzustände, auch des aktuell aktiven Zustands. Es ist somit möglich eine Betriebsphase so häufig wie nötig zu wiederholen. Mittels dieses Ansatzes können Verweilzeiten in einzelnen Zuständen sehr kurz gewählt werden, was zu einer hohen zeitlichen Auflösung des Modells führt und eine flexible und dynamische Systembeschreibung erlaubt.
Zusätzliche temporale Nebenbedingungen können durch technische Grenzwerte des spezifischen technischen Systems oder der aktuellen Betriebsphase bedingt werden. Ein Beispiel hierfür sind minimale Betriebszeiten in einzelnen Betriebsphasen eines Kohlekraftwerks, welche berücksichtigt werden müssen, bevor ein Herunterfahren möglich wird. Durch diese Vorgaben werden im konkreten Fall Verschleißerscheinungen durch thermische Wechsel reduziert. Eine bestimmte Betriebsphase muss somit für eine bestimmte Zeit t_{min} beibehalten werden,

auch wenn die Verweilzeit t_d kleiner gewählt wurde. Im entgegengesetzten Fall können Speichereinheiten als Beispiel dienen, welche nur bis zu einem bestimmten Füllstand geladen werden dürfen. Hier kann eine maximale Zeit t_{max} als Begrenzung der Betrachtungszeit dienen, um zusätzliche Berechnungen der einzelnen Betriebsphasen S zu bestimmten globalen Zeiten t_G anzustoßen. Abbildung 4.2 visualisiert diese Zusammenhänge und die Hauptkomponenten des Energieoptions- sowie des Zustandsmodells in einem erweiterten Zustandsautomaten.

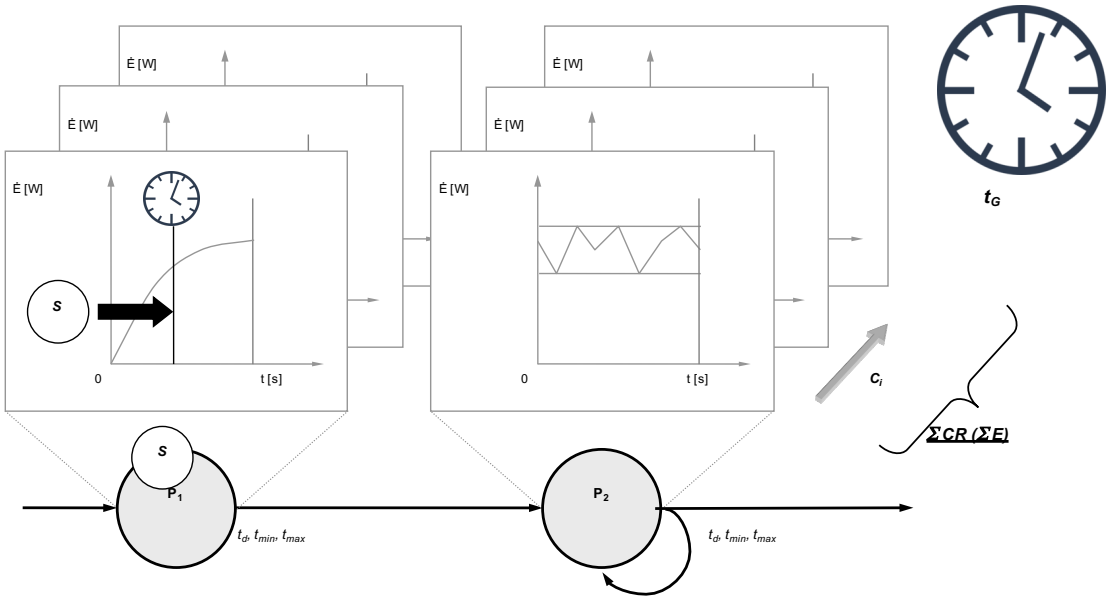

Abbildung 4.2.: Hauptelemente des Energieoptionsmodells und des Zustandsmodells

Aus dem Graphen $G(P, t)$, welcher die möglichen Sequenzen der Betriebsphasen P_i abbildet, und der einem bestimmten Verbindungspunkt C_i zugehörigen betriebsphasenspezifischen Beschreibungen der Energieflüsse E kann der erwartete Energiefluss $\dot{E}$ einer Energiedomäne (z.B. Gas, Wärme oder Elektrizität) an diesem Verbindungspunkt ermittelt werden. Es ist hierfür zweckdienlich die Zeit des Modells in Relation zu der globalen Zeit t_G zu betrachten. So wird es unter Nutzung der globalen Zeit möglich, die Beschreibung des Energieflusses über die Zeit im aktuellen und auch in zukünftigen Betriebszuständen vorherzusagen und zu analysieren. Unter Berücksichtigung der temporalen Randbedingungen t_{min} und t_{max} muss ein Knoten oder der diesem zugeordneten Betriebsphase verlassen werden, wenn die Zeit t_d erreicht wurde. Kosten und Einnahmen eines technischen Systems können über Kostenfunktionen der Energiemengen über die Zeit ermittelt werden.
Im Gegensatz zu dem in Abbildung 4.2 gezeigten Graphen wird davon ausgegangen, dass die Mehrheit der technischen Systeme durch zyklische Zustandsmaschinen abgebildet werden können, da sie permanent mit einem Netzwerk verbunden sind und sich stets wiederholende Betriebszustände einnehmen. Dies trifft jedoch nicht auf jedes technische System zu. So müssen elektrische Fahrzeuge in der Regel während ihrer Mobilitätsphase vom elektrischen Netzwerk getrennt werden. Hierfür müsste ein weiterer Zustandsautomat in Betracht gezogen werden, welcher unter Berücksichtigung weiterer Options- und Kostenmodelle unterschiedliche Zustände der Verbindung zu einem Energienetzwerk abbildet. Dieser Umstand würde jedoch der oben beschriebenen Grundannahme, dass ein Energie-Agent stets in ein Netzwerk eingebunden ist, entgegenstehen. Darüber hinaus würde es die Komplexität des Ansatzes

signifikant erhöhen und somit die lokale Nutzbarkeit und Kommunizierbarkeit der Modelle reduzieren. Es wird daher eine obligatorische, binäre Zustandsvariable eingeführt, welche die Verbindung des Systems mit dem Netzwerk abbildet.
Ähnliche zeit- und graphenbasierte Beschreibungen dienen auch als Grundlage von Zustandsautomaten. Energie-Agenten unterscheiden sich von diesen jedoch wie folgt: Zunächst werden im Kontext der Modellierung von Energie-Agenten Übergänge nicht zur Änderung von Zustandsvariablen genutzt. Sie definieren lediglich die möglichen Folgezustände und Handlungsmöglichkeiten des Modells. Als weiteres Alleinstellungsmerkmal definieren Knoten des Energieoptionsmodells Muster zur Änderungen der Energieflüsse und Kosten. In regulären Zustandsautomaten repräsentiert dies einen festen Systemzustand.
Aufbauend auf der in [SBP+07] beschriebenen Klassifizierung der oben umrissenen Probleme von Agentensystemen und der Betrachtung unterschiedlicher Betriebsphasen technischer Systeme erlaubt das Modell des Energie-Agenten die Differenzierung der folgenden charakteristischen Muster von Energieflüssen:

- **Konstanter Energiefluss**:Auch wenn dies die einfachste Form der Energienutzung ist, so trifft sie auf eine große Zahl an technischen Systemen in unterschiedlichen Betriebsphasen zu. So können Beispiele wie Leuchtkörper, Bügeleisen, Akkumulatoren oder Heizungssysteme in den unterschiedlichen Energiedomänen gefunden werden. In Abhängigkeit des individuellen technischen Systems kann die Länge der Betriebsphase variieren und muss im Rahmen des Engineerings in Hinblick auf mögliche Zeiträume eingeschränkt werden.

- **Gemessener Energiefluss**: Diese Klasse erlaubt die einfachste Form der Berücksichtigung empirischer Daten. Hierfür werden Messungen des Energieflusses als Datenbasis gespeichert und zur Beschreibung der Energieflüsse individueller Betriebsphasen genutzt. Die Länge der Betriebsphase wird dabei in Abhängigkeit von der Länge der Messung definiert.

- **Dynamischer Energiefluss**: Diese Klasse dient der Abbildung der oben beschriebenen Modelle für thermische Großkraftwerke. Mit einer anpassbaren Verweildauer und minimalen und maximalen Werten zur Begrenzung der Energieflüsse kann der Energiefluss im gegebenen Kennfeld variieren. Im Ergänzung zu fixen Änderungswerten über die Zeit ist eine mathematische Beschreibung der An- und Abfahrprozesse möglich, um die vorherrschenden Effekte besser abbilden zu können.

- **Mathematisch und empirisch beschriebener Energiefluss**: Basierend auf mathematischen Zusammenhängen oder einer entsprechenden Wissensbasis können Energieflüsse exakt berechnet oder auf Basis vorhandener Daten ermittelt werden. Um die hierfür notwendige Flexibilität zu erhalten, besitzt die Klasse Schnittstellen für die Berechnung des Energieflusses für den konkreten Zeitpunkt wie auch für die Bestimmung von Zeiträumen in einer Betriebsphase. Dies erlaubt die Einbindung individueller Lösungen für Anlagentypen, welche mit den drei zuvor genannten Klassen nicht abgebildet werden können.

Neben der Beschreibung der Energieflüsse in den unterschiedlichen Betriebsphasen bedarf es einer Abbildung des Wissens über die relevanten und erlaubten Sollwerte der lokalen Steuerungssysteme. Erst dieses Wissen erlaubt die Steuerung der technischen Systeme und wird

daher ebenfalls im Energieoptionsmodell des Energie-Agenten vorgehalten.
Abgeleitet vom hier präsentierten Modell können die möglichen Handlungsabläufe des betrachteten technischen Systems eruiert werden. Sind der Betriebszustand und die zu betrachtende globale Zeit bekannt, dient die Beschreibung der Energieflüsse des Betriebszustandes als Basis für alle weiteren Berechnungen. Im Falle dynamischer oder mathematischer Beschreibungen der Energieflüsse ist es möglich die Sollwerte zu variieren, ohne eine Betriebsphase verlassen zu müssen. In anderen Fällen muss hierfür eine individuelle Betrachtung der Betriebsphase des Modells erfolgen. Um den aus diesen Betrachtungen resultierenden Zustandsraum abschätzen zu können, müssen die ebenfalls im Modell verankerten Kostenfunktionen betrachtet werden.
Derksen et. al. haben dieses Modell in einem XML Schema abgebildet [age], welches in einer hierfür bereitgestellten Software an die Bedürfnisse einzelner Anwendungsfälle angepasst werden kann. Die folgenden Abbildungen sind diesem Schema entnommen und sollen der Visualisierung des Konzeptes dienen.
Abbildung 4.3 zeigt die Wurzelklasse eines technischen Systems, welches die Grundlage eines

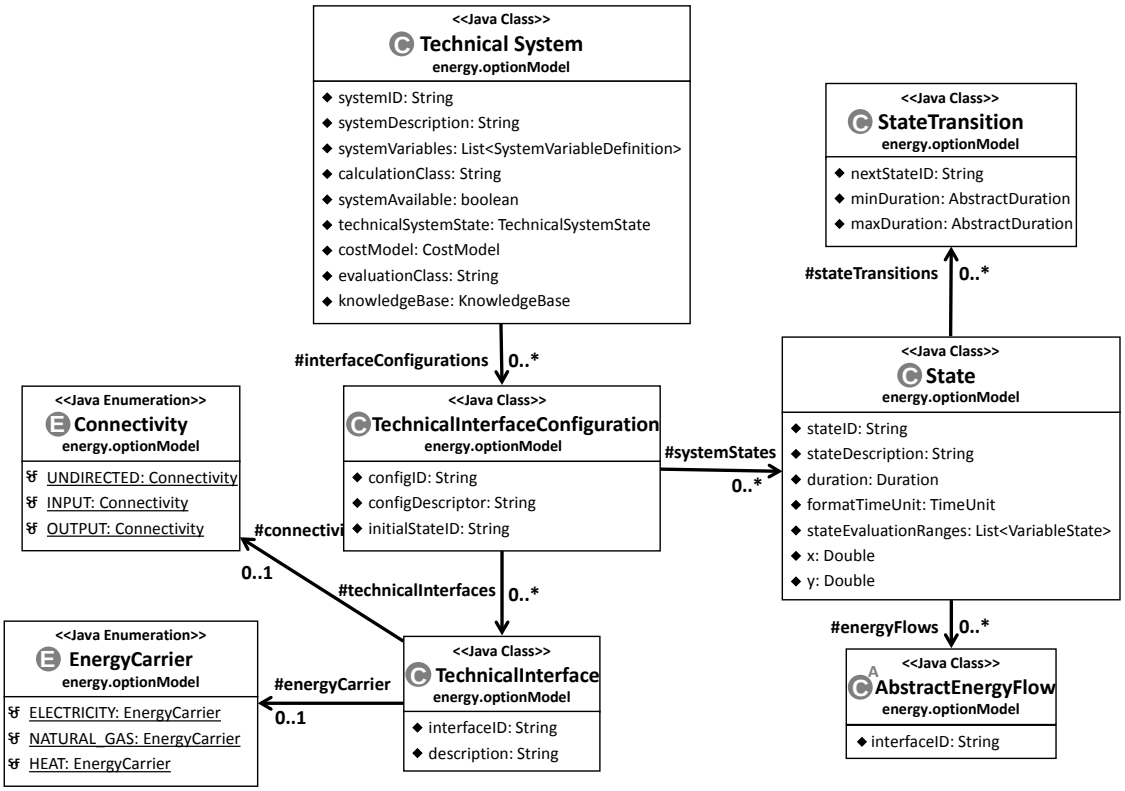

Abbildung 4.3.: Wurzelklasse eines technischen Systems als Grundlage eines Energie-Agenten

Energie-Agenten bildet. Ausgehend vom Typ „Technisches System“ können seine *System ID*, die Liste der zugehörigen *I/O Signale* sowie Mess- und Sollwerte des zu steuernden Systems definiert werden. Bezugnehmend auf die individuelle Berechnungsklasse können darüber hinaus maßgeschneiderte Funktionen zur Berechnung der Energieflüsse der technischen Systeme in unterschiedlichen Betriebszuständen eingesetzt werden. Der Typ *TechnicalInterfaceConfiguration* trägt dem Umstand Rechnung, dass sein technisches System auf unterschiedliche Arten mit einem oder mehreren Energienetzen verbunden sein kann. Dieser Typ wird gefolgt

von der Definition des Technischen Schnittstellen, den unterschiedlichen Verbindungsmöglichkeiten, den möglichen Energieträgern und Betriebszuständen. Die Definition von Speichern wird aus Gründen der Komplexität im UML Diagramm nicht abgebildet. Sie werden als spezieller Typ einer Schnittstelle modelliert und erweitern die Klasse *TechnicalInterface* um Informationen über die Kapazität und Auslastung eines Speichers. Zustände werden als Knoten der Zustandsmaschine abgebildet. Der Type *StateTransition* beschreibt hingegen die möglichen Übergänge und damit die temporalen Zusammenhänge zwischen den Zuständen. Aus der Klasse *AbstractEnergyflow* können alle oben beschriebenen Energiefluss-Typen abgeleitet werden.
Mit den Systemvariablen, wie sie in der Klasse *TechnicalSystem* genutzt werden, ist es möglich Mess- und Sollwerte systematisch zu beschreiben. In Abhängigkeit des Typen einer solchen *SystemVariableDefinition* können die möglichen Wertebereiche definiert und – wenn nötig – diskretisiert werden. Es ist damit auch möglich für individuelle Betriebsphasen die Wertebereiche zu reduzieren, um den Lösungsraum einzuschränken. Der Zustandsraum eines technischen Systems kann somit in generalisierter Form beschrieben und evaluiert werden. Im Gegenzug kann der Systemzustand in Form von Energieflüssen, Speicherständen und Variablenwerten zu einem spezifischen Zeitpunkt ermittelt werden.

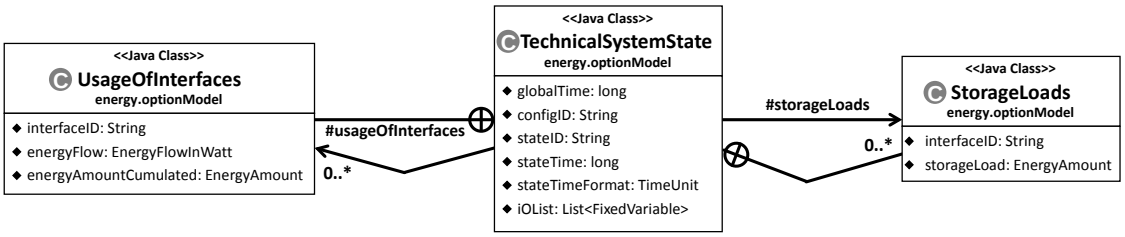

Abbildung 4.4.: Modell der Systemzustände des Energie-Agenten

Zur flexiblen und generalisierten Beschreibung der Kosten und Erlöse von Energieumwandlungs- und Transportprozessen wurde das in Abbildung 4.5 dargestellte Kostenmodell definiert.

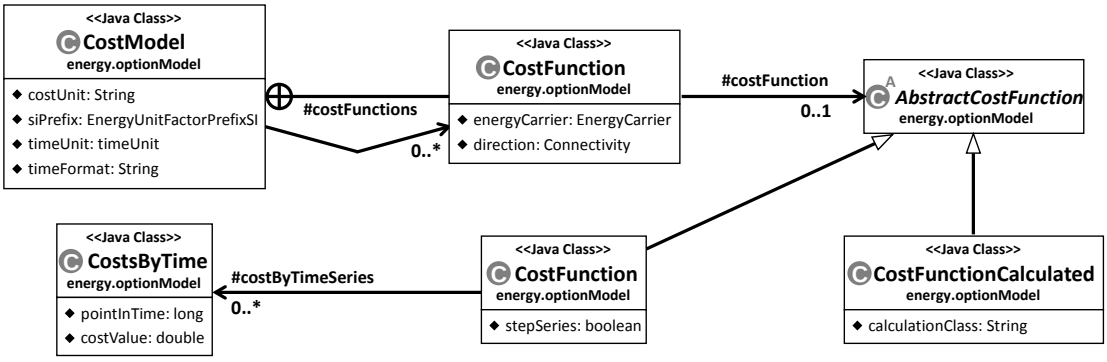

Abbildung 4.5.: Kostenmodell des Energie-Agenten

Basierend auf einer für alle Systeme gemeinsamen Kosteneinheit, welche im Kostenmodell definiert wird, können individuelle Kostenfunktionen für jeden Energieträger und jede Verbindung angelegt werden. Die Kosten können dabei in einer frei definierbaren Einheit (von zum Beispiel Euro pro kWh oder kg (CO2) pro MWh) beschrieben werden. Unabhängig der in

Abbildung 4.5 gezeigten Enumeration *Connectivity* dürfen dabei nur gerichtete Energieflüsse betrachtet werden. Um die hieraus resultierenden Kostenfunktionen zu beschreiben sind jedoch die folgenden Möglichkeiten verfügbar: Zum einen können, unter Annahme gleichbleibender Beschaffungskosten, diskrete Kostenwerte über die Zeit definiert werden. Alternativ kann eine an den Anwendungsfall angepasste *CostFunctionCalculated* implementiert werden, welche unterschiedliche Zustandsübergänge des technischen Systems, aktuelle Marktpreise auf Basis externer Webservices und andere Faktoren berücksichtigen kann.
Der Aufruf und die Mechanismen zur Nutzung der individuell implementierten Energieflüss-Klassen, ihrer Kostenfunktionen und Auswertungsmöglichkeiten (siehe das Attribut *evaluationClass* in der Klasse *TechnicalSystem* in Abbildung 4.3) sind für alle Klassen in gleicher Weise implementiert. Basierend auf den zugrundeliegenden abstrakten Klassen können individuelle Klassen spezifische technische Systeme und Anwendungsfälle abbilden. Ihr praktischer Nutzen, zum Beispiel im Rahmen der Bewertung eines Systems, bedarf lediglich des Aufrufs der *TechnicalSystem*s Klasse, welche die jeweils aktuellen Informationen über die technischen Systeme bereitstellt.

4.3. Durchgängiges Engineering durch Wiederverwendung

Im Folgenden sollen die grundlegenden Anwendungsmöglichkeiten des Energie-Agenten in einem Engineering-Prozess beschrieben werden, welcher alle Entwicklungsphasen von der initialen Idee bis hin zur Installation auf realer Hardware im Feld umfassen soll. Eine Grundanforderung ist dabei, dass Energie-Agenten in realen Systemen aus Sicherheitsgründen nur nach umfangreichen Sicherheits- und Kompatibilitätstests Anwendung finden können. Dieser Umstand ergibt sich aus der autonomen und asynchronen Natur eines Agentensystems und den dafür notwendigen Kommunikations- und Verhandlungsprozessen, welche potentielle Fehlerquellen darstellen und im schlimmsten Falle zu Instabilitäten des Steuerungssystems führen können. Da diese Steuerungssysteme jedoch zur Stabilisierung eines technischen Systems dienen und im Zweifelsfalle für die Sicherheit der Energieversorgung Verantwortung tragen sollen, ist die Auftretenswahrscheinlichkeit solcher Instabilitäten auf ein akzeptables Maß zu reduzieren. Ein wichtiger und quantifizierbarer Faktor ist dafür die Reaktionsgeschwindigkeit der Agenten im Rahmen der Inter-Agenten-Kommunikation. Diese muss somit im Rahmen der Tests berücksichtigt und dokumentiert werden. Zusätzlich hierzu müssen weitere Faktoren, wie die Zuverlässigkeit der Agenten, untersucht werden. Die Zuverlässigkeit ist dabei in zwei unterschiedliche, aber gleichgewichtige Kategorien zu differenzieren: Zum einen die technische Sicherheit als auch die Zuverlässigkeit der vom Agenten eingegangenen Verpflichtungen. Es ist dabei unerheblich, welche Vorgehensweise für die Verifikation gewählt wird. Die der Verifikation zugrundeliegenden Anforderungen müssen lediglich in der Modellierung des Energie-Agenten abbildbar sein.
Um eine durchgängige Berücksichtigung dieser Anforderungen in allen Phasen des Entwicklungsprozesses sicherstellen zu können muss das Konzept des Energie-Agenten ebenfalls in allen Phasen des Entwicklungsprozesses Anwendung finden, unabhängig von den Änderungen der Ausführungsumgebung. Ähnliche Ansätze wurden bereits im Kontext von *Hardware-in-the-loop*-Simulationen oder des *Rapid Control Prototyping* [AB06] vorgestellt. Dabei sind die folgenden Phasen zwingend zu berücksichtigen:

- **Spezifikation und Modellierung**: Basierend auf dem Wissen über das zu steuernde

technische System und seine Schnittstellen müssen die gewünschten Funktionalitäten eines Energiesystems als auch seine internen und externen Interaktionen beschrieben werden.

- **Implementierung**: Aufbauend auf den Ergebnissen des ersten Schritts muss ein System implementiert werden, welches sowohl in der Lage ist, lokal verfügbare Informationen zu nutzen, als auch mit anderen Systemen, bzw. ihren virtuellen Repräsentanten, den Energie-Agenten, zu interagieren. Diese Implementierung sollte idealerweise unabhängig von einer konkreten Programmiersprache möglich sein und sich auf einheitlichen Datenmodellen abstützen.

- **Simulation**: Die Simulationsumgebung muss in der Lage sein, den implementierten Energie-Agenten die gleichen Informationsquellen zur Verfügung zu stellen wie das spezifizierte Produktivsystem. Insbesondere wenn eine dezentrale Anwendung der Software betrachtet wird, ist die Bereitstellung gleichwertiger Schnittstellen unabdingbar, um die entsprechenden Abhängigkeiten und Interdependenzen testen zu können.

- **Testbed-Anwendung**: In hybriden oder komplett aus realer Hardware aufgebauten Testbeds muss die Software mit der gleichen, idealerweise der selben, Hardware getestet werden, wie sie später im Feld zum Einsatz kommt. Hierdurch wird die Software den Einschränkungen der realen Welt, wie zum Beispiel beschränkten Rechen- und Kommunikationsressourcen und harten Zeitanforderungen, ausgesetzt.

- **Installation im Produktivsystem**: Die finale Anwendung der Energie-Agenten im Produktivsystem stellt hohe Anforderungen an die genutzte Hardware sowie die entwickelte Software. So sollte die genutzte Hardware ähnliche Lebenserwartungen aufweisen wie das von ihr kontrollierte technische System und die Software durch Softwareupdates wartbar sein.

Es wird davon ausgegangen, dass sich Energie-Agenten stets in einer Zwischenschicht der Automatisierungspyramide befinden. Da größere Energiesysteme wie Kraftwerke, BHKWs oder Windkraftanlagen bereits über eigene, lokale Kontroll- und Überwachungsprozesse verfügen, müssen Energie-Agenten keinen harten Echtzeitanforderungen genügen, wie sie auf den unteren Automatisierungsebenen notwendig sind.
Diese Eingrenzung ist notwendig, da das Verhalten von Agenten in der Regel nichtdeterministisch ist und somit eine Ausführung in Echtzeit mit festen Antwortzyklen nicht möglich ist. Darüber hinaus ist der primäre Anwendungsfall des Agentenparadigmas die Koordination und Harmonisierung von Planungsprozessen im Kontext komplexer Systeme. Kritische Systeme mit harten Echtzeitanforderungen, wie die oben genannten Kraftwerke und ähnliche Anwendungsfälle, werden sich daher auf absehbare Zeit weiterhin auf klassische Speicherprogrammierbare Steuerungen abstützen.
In Hinblick auf die durchgängige Verwendung des Energie-Agenten-Konzepts in allen oben genannten Phasen eines Entwicklungsprozesses müssen Komponenten des Energie-Agenten identifiziert werden, welche ohne Anpassung in unterschiedlichen Phasen nutzbar sind, und Komponenten, welche explizit für einzelne Phasen erstellt werden müssen. Ein besonderes Augenmerk muss dabei auf die Verbindungspunkte der Komponenten sowie des Energie-Agenten mit seiner sich in den Phasen des Entwicklungsprozesses ändernden Umwelt und anderen Agenten gelegt werden. Die typische Implementierung eines Energie-Agenten besteht dabei

grundsätzlich aus einem internen Datenmodell, einer Auswahl möglicher Verhaltensweisen um Schlussfolgerungen zu ziehen und daraus Aktionen abzuleiten sowie Schnittstellen zur Außenwelt. Diese Architektur erlaubt es die durchgängig benötigten Funktionen des Energie-Agenten von der sich änderenden Umwelt zu kapseln. So ermöglichen es diese Schnittstellen, einen Großteil des Codes eines Energie-Agenten in allen Phasen des Entwicklungsprozesses wiederzuverwenden. Im Mittelpunkt der Analyse muss dabei die Verbindung des Energie-Agenten mit dem zu steuernden technischen System sowie mit anderen Energie-Agenten stehen. Die Inter-Agenten-Kommunikation wird in der Regel durch die Nutzung der Agent Communication Language (ACL) und der Erstellung gemeinsamer Wissensbasen, wie zum Beispiel Ontologien, umgesetzt. Es wird angenommen, dass diese Art der Kommunikation in allen Phasen des Entwicklungsprozesses genutzt werden kann.
Diese Annahme trifft jedoch nicht auf die Verbindung zwischen Energie-Agenten und den zu steuernden technischen Systemen zu. Aus diesem Grund wird ein einfach auszutauschendes *Behaviour* [GSV99] implementiert, welches in Abhängigkeit von der Ausführungsumgebung unterschiedliche Protokolle und Funktionalitäten bereitstellt. So kann dieses *Behaviour* in der Simulationsphase unterschiedliche Messwerte simulieren und in der späteren Testbed- oder Live-Anwendung die Messwerte des Systems verarbeiten. Hierzu müssen unterschiedliche Formate wie Modbus RTU, Profibus, OPC UA, MBus, EEbus, IEC 61850 und andere (siehe [mod18], [opc], [pro], [eeb], [Mac06]) unterstützt werden. Dies ist in der praktischen Anwendung unter Nutzung bereits existierender Protokollbibliotheken mit vertretbarem Aufwand umsetzbar.
Im Gegensatz zu diesem austauschbaren Wissen über Schnittstellen und dort genutzter Protokolle ist der Kern des Energie-Agenten, das Wissen über die möglichen Zustände und ihre Ausprägungen, mit einem hohen Aufwand in der Erstellung verbunden. Das Konzept des Energie-Agenten sieht daher vor, dieses Wissen im gesamten Entwicklungsprozess wieder zu verwenden und weiter zu entwickeln. Es ist dabei sogar möglich, die erarbeiteten Systemmodelle nicht nur für die Prozesse des steuernden Energie-Agenten sondern auch im Rahmen der Simulations- und Testbedphase als Kern der Simulationsmodelle zu nutzen. Abbildung 4.6 zeigt die oben beschriebenen Verbindungen exemplarisch mittels eines hypothetischen Trockner-Agenten, welcher in einer Simulation genutzt werden kann. Wenn der Agent ausgeführt wird, kann er sich selbst in einer simulierten oder realen Umgebung verorten und das adäquate *IO-Behaviour* nutzen. Beide *Behaviours* besitzen die selben Übergabevariablen für ein- und ausgehende Messungen und Sollwerte.
Die Anwendung der hier beschriebenen Energie-Agenten in energietechnischen Systemen führt zu Herausforderungen im Hinblick auf die finanziellen und sicherheitstechnischen Risiken. Spezifische Anforderungen der Domäne müssen im Rahmen der Entwicklung des Energie-Agenten berücksichtigt werden. Verhaltensweisen, welche Fähigkeiten in Bereichen wie marktbasierter Steuerung oder der Stabilisierung von Elektrizitätsnetzen mit einem hohen Anteil dezentraler Erzeugung mit sich bringen, bedürfen enger Grenzen, um unerwünschte Nebeneffekte auszuschließen. Um dies sicherzustellen sind neue Ansätze im Bereich des Engineering notwendig. Diese müssen sowohl das Verhalten der technischen Systeme als auch ihre Grenzen und Schnittstellen beschreiben. Es müssen Modelle zur Beschreibung der Systeme und ihrer Zusammenhänge bereitgestellt und deren Wiederverwendbarkeit sichergestellt werden. Ob diese Modelle dann im Zuge zentraler oder dezentraler Entscheidungsfindung genutzt werden oder im Rahmen von Simulationen das Verhalten technischer Systeme abbilden, ist dabei unerheblich.

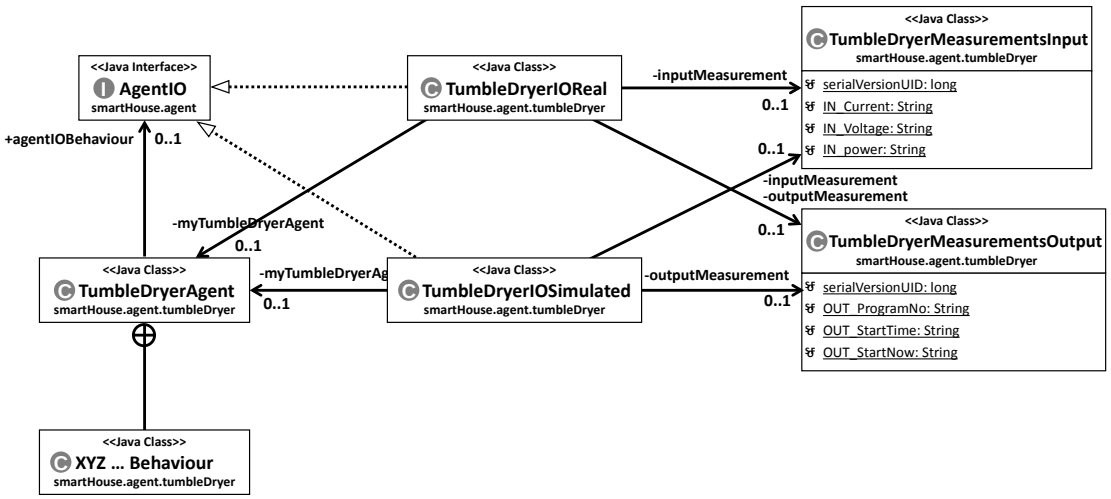

Abbildung 4.6.: Exemplarische Darstellung eines austauschbaren *IO-Behaviours* für die Nutzung des Energie-Agenten in unterschiedlichen Entwicklungsphasen

4.4. Zwischenfazit

In diesem Kapitel wurde das Konzept des Energie-Agenten eingeführt und seine Vorteile im Zuge des Engineering von dezentralen Energiesystemen erarbeitet. Es wurde gezeigt, dass ein Energie-Agent als Repräsentant technischer Systeme auf einer operativen Ebene dienen kann und bereits vorhandene lokale Steuerungssysteme wie Speicherprogrammierbare Steuerungen nicht ersetzen, sondern ergänzen soll. Gemäß den ihm zugeordneten Aufgaben kann die dem Energie-Agenten inhärente Komplexität und damit die Ausgereiftheit variieren. Um diesen Umstand quantifizierbar zu machen wurde ein entsprechendes Reifegradmodell eingeführt.
Es wurde darüber hinaus ein Konzept zur Wiederverwendbarkeit des erstellten Programmcodes in unterschiedlichen Phasen des Entwicklungsprozesses vorgestellt und anhand von Beispielen illustriert. Die Nutzung eines austauschbaren *Behaviours* zur Anpassung der Kommunikation mit Simulationskomponenten oder realen technischen Systemen erlaubt dabei die schnelle Anpassung des Energie-Agenten an unterschiedliche Umgebungen unter Beibehaltung der relevanten Kernkomponenten.
Zuletzt wurde das Konzept eines abstrakten Options- und Kostenmodells erörtert, welches alle Betriebsphasen eines technischen Systems abbilden kann. Es wurden dabei explizit hybride Energiesysteme berücksichtigt, was eine Anwendung in unterschiedlichen Energiedomänen wie Elektrizität, Gas, Wärme, Wasser oder einer Mischung dieser Domänen erlaubt. Die Anwendung dieses Modells wurde anhand von Beispielen erläutert. In [LDFU15] werden ein domänenübergreifender Anwendungsfall beschrieben und die Vorteile in Hinblick auf eine Multi-Domänen-Optimierung eines Haushaltes dargelegt.
Das Konzept des Energie-Agenten soll das Design und die Implementierung von Anwendungen für zukünftige Energienetze vereinfachen. Im Kontext einer steigenden Systemkomplexität und einer immer unübersichtlicheren Menge an proprietären Lösungen kann der Energie-Agent der Vereinheitlichung und Verbindung dieser Lösungen dienen. Es können daher die folgenden Vorteile des Ansatzes erwartet werden:

- Vereinheitlichte Energie-Agenten können ein Werkzeug sein, um die Entwicklungskosten von zukünftigen Energienetzen gering zu halten.

- Ein einheitliches Energie-Options Modell für technische Systeme erlaubt die Verknüpfung unterschiedlicher Energiedomänen. Dies kann zur Kompensation volatiler Energieerzeugung in einer Domäne durch Umwandlung und Speicherung in einer anderen Energieform genutzt werden.
- Die durchgängige Verwendbarkeit des Energie-Agenten im Kontext eines Entwicklungsansatzes erlaubt es, iterative Ansätze im Kontext des Energie-Engineerings einzuführen und Standard- und Normen-konform auszugestalten.

Der Energie-Agent kann somit als Schlüsselelement zukünftiger Energienetze dienen.

5. Vorgehensmodelle und Entwicklungsmethoden für Software-Agenten

Der Entwurf und die Entwicklung von Software wird seit den 1970er Jahren als eigenständige Disziplin wahrgenommen. In Anbetracht der steigenden Komplexität von Softwareprodukten und neuen technischen und daraus resultierenden organisatorischen Möglichkeiten werden die Methoden in der Domäne des Softwareengineerings beständig weiterentwickelt und an aktuelle Erkenntnisse und Notwendigkeiten angepasst. Der folgende Abschnitt soll einen Überblick des Themenbereichs geben und grundlegende Begriffe definieren.

5.1. Grundlagen der Softwareentwicklung - Projekt- und Softwarelebenszyklus

Das *Project Management Institute* definiert in seinem Standardwerk *PMBOK* (Siehe: [Pro13]) ein Projekt als „ein zeitlich begrenztes Vorhaben mit dem Ziel, ein einmaliges Produkt, eine Dienstleistung oder ein Ergebnis zu schaffen“ [Pro13]. Broy und Kuhrmann teilen ein Projekt in die vier Phasen Projektentstehung, Projektdefinition, Projektdurchführung, Projektabschluss [BK13]. Im Rahmen der Produktion von Software dienen die beiden ersten Phasen der Konzeption und Definition des Produktes. In einem ersten Schritt wird eine *Anforderungsfestlegung* durchgeführt, welche die Ziele, Anforderungen und Spezifikationen des Produktes definiert. Im Anschluss folgt dann der *Grob- und Feinentwurf* der Architektur und der Programmstruktur. Im Zuge der Projektdurchführung wird dann das angestrebte Produkt geschaffen und getestet. Zu den hier ausgeführten Tätigkeiten zählen unter anderem die *Implementierung* sowie *Tests*, *Integration*, *Erprobung* und *Inbetriebnahme*. Der Projektabschluss stellt das formale Ende eines Entwicklungsprojektes dar. Die daran anschließende Nutzungsphase ist geprägt von *Wartung* und *Weiterentwicklung* sowie der finalen Außerbetriebnahme des initial geschaffenen Produktes [IEC08].

5.2. Grundlagen der Softwareentwicklung – Vorgehensmodelle

5.2.1. Beispiel eines Vorgehensmodells

Broy et. al definieren Vorgehensmodelle als Organisationsstrukturen, welche „beschreiben, welche Aufgabenfelder zu bearbeiten sind und welche Kompetenzen erforderlich sind, um diese Aufgaben verantwortlich wahrzunehmen“ [BK13]. Auf Basis eines Vorgehensmodells kann

ein gemeinsames Verständnis von Kompetenzen und Obliegenheiten aller Projektbeteiligten geschaffen werden. Es bildet somit die Basis für eine systematische Planung und Steuerung von Projekten. Zwecks Effizienzsteigerung werden Vorgehensmodelle in Firmen- oder konsortialinternen Projektmanagementhandbüchern festgelegt und auf Basis der in den vorangegangenen Projekten gewonnenen Erkenntnisse im Zuge von *Lessons Learned Evaluationen* erweitert. Als Grundlage können allgemeine Standards und verfügbare öffentliche Quellen wie auch eigene *Best Practices* dienen.
Bild 5.1 stellt die wesentlichen Komponenten eines Vorgehensmodells nach [BK13] dar. Struktur- und Prozesswissen sowie die Definition der notwendigen Werkzeuge werden dabei getrennt. Das Aufbaumodell, welches das Strukturwissen enthält, definiert im Rollenmodell die Zusammensetzung von Teams sowie deren Aufgaben im Aktivitätsmodell. Darüber hinaus bietet es im Artefaktmodell eine Sammlung erfolgversprechender Handlungsmuster für die individuellen Aktivitäten. Das Ablaufmodell enthält das Prozesswissen. Es stellt den eigentlichen Entwicklungsverlauf sowie die hierin enthaltenen Prozesszusammenhänge dar. Die Beschreibung der notwendigen Werkzeuge erfolgt in einem eigenen Werkzeugmodell.

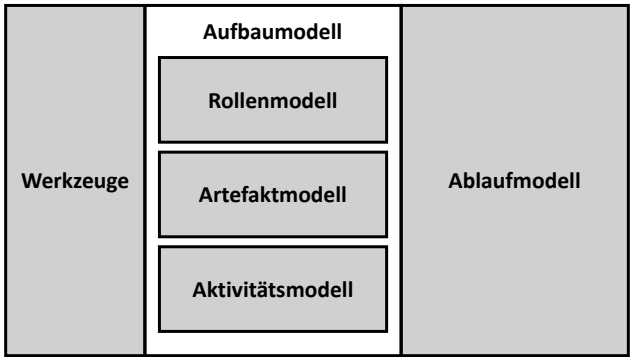

Abbildung 5.1.: Bestandteile eines Vorgehensmodells nach [BK13]

5.2.2. Metamodell und Begriffsdefinition

Im Folgenden sollen auf Basis des oben beschriebenen Vorgehensmodells von Broy et. al die Bestandteile und Zusammenhänge eines Metamodells der Softwarenetwicklung als Grundlage weiterführender Betrachtungen beschrieben werden. Abbildung 5.2 visualisiert den schematischen Aufbau eines Metamodells der Softwareentwicklung in Anlehnung an die Arbeiten von Noack und Schienmann (Siehe: [NS99]). Die einzelnen Elemente lassen sich dabei wie folgt definieren:

- **Phase**: Phasen gliedern Projekte in logische und zeitliche Zusammenhänge. Eine Phase kann in Teilphasen aufgespalten werden, welche wiederum aus individuellen Teilphasen bestehen können. Dies ermöglicht eine Hierarchisierung der Aktivitäten und die Zusammenfassung von Aktivitäten in einer übergeordneten (Teil-)Phase. Wird eine Reihe von Teilphasen mehrfach durchlaufen, so werden diese im Folgenden als zusammenhängende, übergeordnete Phase interpretiert. Als Beispiel kann hier die Lösungsfindung in einem Projekt herangezogen werden. Die übergeordnete Phase *Lösungsfindung* kann in die Teilphasen *Diskussion*, *Zusammenfassung* und *Verteilung der Ergebnisse* aufgeteilt werden. Diese werden gegebenenfalls mehrfach durchlaufen, bis die Phase *Lösungsfindung* abgeschlossen wird.

- **Artefakt**: Die Ergebnisse einer individuellen Aktivität werden als Artefakt bezeichnet. Die Ausprägungsform ist dabei in der Regel im Projektmanagementhandbuch definiert und kann sowohl ein physisches Produkt als auch Software oder Dokumentation umfassen.
- **Meilenstein**: Ein Meilenstein umfasst ein oder mehrere Artefakte, welche durch den Abschluss einer oder mehrerer Aktivitäten erarbeitet wurden. Die hiermit abgesteckte *Projektfortschrittsstufe* umfasst die zu erarbeitenden Artefakte sowie den hierfür vorgesehenen Zeitrahmen und unterstützt damit das Projektcontrolling.
- **Aktivität**: Eine Arbeitsanweisung für eine spezifische Tätigkeit wird unter der Bezeichnung „Aktivität“zusammengefasst und beschreibt alle notwendigen Handlungen um ein Artefakt zu erarbeiten.
- **Rolle**: Nach Noack [NS99] definieren Rollen die „notwendige(n) Fähigkeiten, Kenntnisse und Erfahrungen der am Entwicklungsprozess beteiligten Personen für die Durchführung der vordefinierten Aktivitäten (...). Sie fassen Verantwortlichkeiten oder Zuständigkeiten bei der Ausführung von Aktivitäten zusammen.“
- **Werkzeug**: Werkzeuge unterstützen die am Prozess beteiligten Personen und die von Ihnen verkörperten Rollen darin die ihnen zugeordneten Aktivitäten auszuführen.

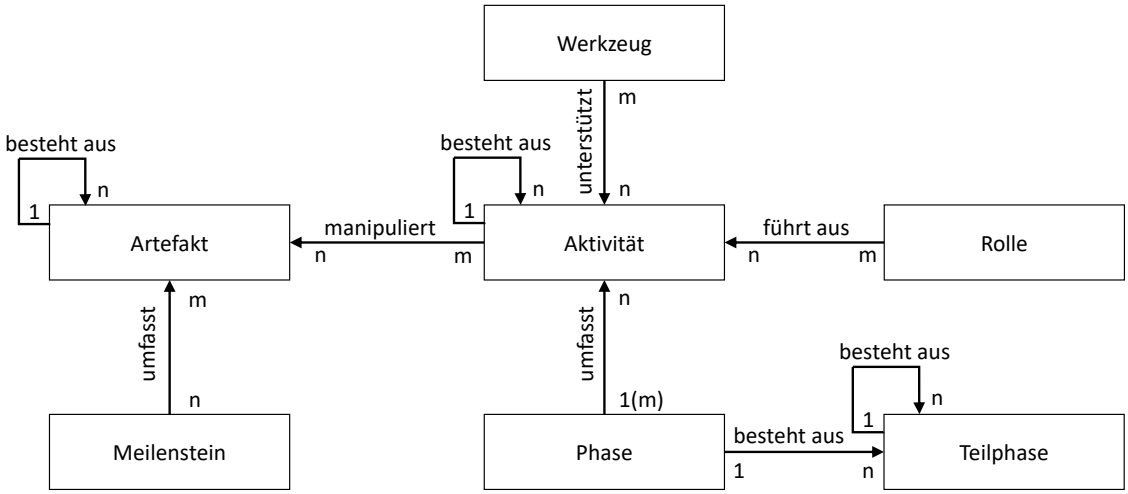

Abbildung 5.2.: Metamodell der Softwareentwicklung, in Anlehnung an [NS99]

5.2.3. Grundlegende Konzepte der Softwareentwicklung

Im Kontext der Softwareentwicklung haben sich drei wesentliche Konzepte etabliert, welche die Adaption und Kombination in anwendungsbezogenen Vorgehensmodellen erlauben [KL14]. Sie werden für die weitere Nutzung in dieser Arbeit wie folgt definiert:

- **Sequentiell**: Die sequentielle Vorgehensweise ist ein klassischer Ansatz im Softwareengineering und bietet auf Basis einer umfassenden Projektplanung ein hohes Maß an

Planbarkeit im Zuge des Projektmanagements. Die individuellen Aktivitäten des Entwicklungsprozesses werden dabei getrennt voneinander betrachtet und lediglich über die ein- und ausgehenden Artefakte miteinander verknüpft. Die folgenden Aktivitäten bauen dabei stets auf den bis dato erarbeiteten Artefakten auf und sind somit einer konsekutiven Ordnung unterworfen. Planabweichungen in einzelnen Projektabschnitten haben somit in der Regel Auswirkungen auf das Gesamtprojekt. [BK13] Die sequentielle Vorgehensweise sieht die Freigabe des Softwaresystems am Ende der Entwicklung vor.

- **Iterativ**: Ein iterativer Entwicklungsprozess wird durch vier rekurrente Projektphasen charakterisiert. Diese umfassen die Schritte *Analyse*, *Evaluierung*, *Realisierung* und *Planung*. Durch die wiederholende Abarbeitung der vier Phasen entsteht in Prototyp, welcher sich dem gewünschten Zielsystem annähert. Dieser Prototyp besitzt dabei nicht den Zustand eines nutzbaren Produktes und dient lediglich dem Feedback von Kunden und späteren Nutzern des Softwareproduktes. [BK13]

- **Inkrementell**: Ziel der inkrementellen Herangehensweise ist nach jeder Phase ein funktionierendes Softwareprodukt vorweisen zu können. [Som10] Im Gegensatz zum iterativen Vorgehen entsteht somit mit jeder Phase ein „fertiges“Produkt. So wird ein Softwarekern, welcher stets eine in sich funktionierende Einheit darstellt, in Inkrementen um zusätzliche Funktionen ergänzt. Die Realisierung der Funktionen kann selbst wiederum inkrementell oder aber auch sequentiell oder iterativ erfolgen. Das Anwachsen des Softwareproduktes ermöglicht analog dem iterativen Entwicklungsprozess Feedbackschleifen mit den Nutzern und Kunden des Softwareproduktes und ein regelmäßiges Hinterfragen des angestrebten Zielzustandes.

5.2.4. Klassische Modelle

In der klassischen Softwareentwicklung wird zunächst in einer aufwändigen Planungsphase das sequentielle Phasenmodell vollumfänglich konzipiert. Als Basis dienen dabei im Vorfeld definierte Rollen-, Artefakt- und Aktivitätsmodelle. Diese Herangehensweise folgt dem Gedanken, dass Änderungen im Produktentstehungsprozess mit hohen Aufwänden und Kosten verbunden sind.
Der Prototyp dieser Vorgehensweise ist das ursprünglich aus der ähnlichen Grundsätzen folgenden Baubranche stammende Wasserfallmodell. Dieses wurde 1970 von Winston W. Royce das erste Mal formell beschrieben [Roy70] und stellte für lange Zeit die häufigste Herangehensweise an Softwareprojekte dar. Abbildung 5.3 zeigt die ursprüngliche Ausprägung des Vorgehensmodells, von welchem mittlerweile unterschiedliche Varianten existieren. Jede Phase des Wasserfallmodells entspricht einer Projektfortschrittsstufe und ist Voraussetzung für die sich anschließenden Aktivitäten. Alle in einer Phase auszuführenden Aktivitäten sowie zu schaffende Artefakte unterliegen strengen Vorschriften. Um eine einfache Projektstruktur zu realisieren dürfen nur zwischen benachbarten Phasen des Entwicklungsverlaufs Informationen ausgetauscht werden. Es ist somit lediglich Möglich eine Phase voranzuschreiten oder in die vorangeganene Phase zurückzufallen. Ein Rückfall in eine noch weiter davor durchgeführte Phase ist nicht möglich.
Aufgrund des klar definierten Ablaufmodells und der darin detailliert beschriebenen Arbeitsschritte liegen die Vorzüge von sequentiellen Vorgehensmodellen wie der Wasserfallmethode in der guten Kommunizierbarkeit des Prozessfortschritts. Auf Basis der Meilensteinplanung

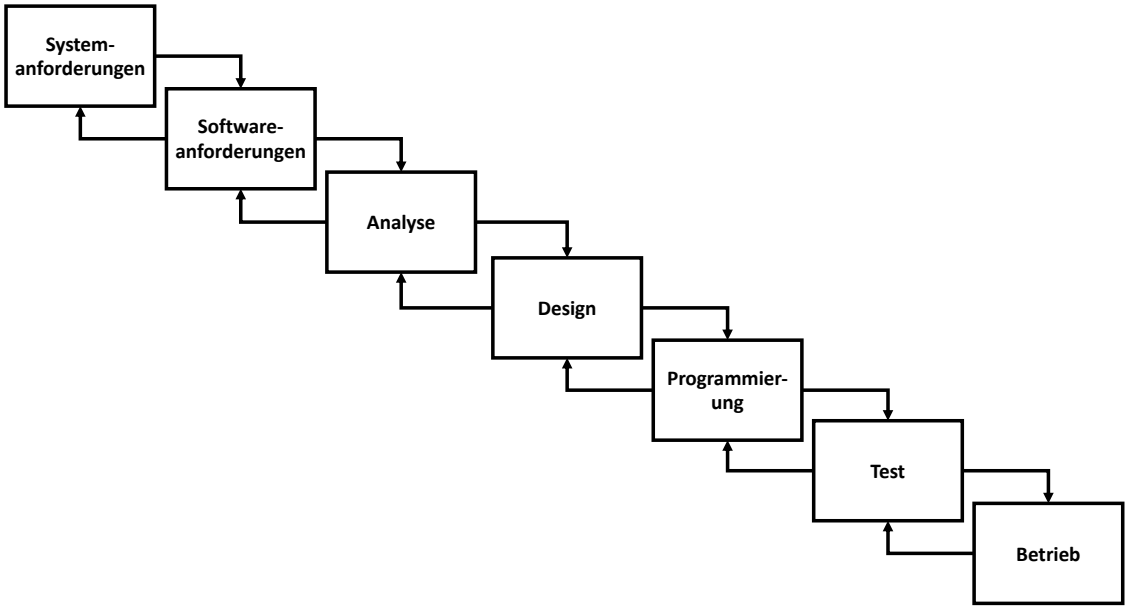

Abbildung 5.3.: Wasserfallmodell nach W. Royce [Roy70]

wird der Arbeitsfortschritt für alle Prozessbeteiligten überprüfbar. Die aktuell auszuführenden Aktivitäten können aus der Projektphase abgeleitet werden. Der Einsatz von Mitarbeitern in parallelen Projekten wird auf Basis des klar definierten Rollenmodells erleichtert. Es ist zudem hervorzuheben, dass der Umfang von Projekten im vergleich zu agilen Vorgehensweisen vermeintlich eingeschränkt werden kann. Da die drei wesentlichen Modelle, namentlich Ablauf-, Aktivitäts- und Artefaktmodell, eine detaillierte Bewertung der Abläufe erlauben sollen unnötige Iterationen vermieden werden. Dies trifft vornehmlich auf komplexe Projekte mit langen Laufzeiten und bereits zu Projektbeginn definierten klaren Zielen und Rahmenbedingungen zu.
Aus diesem Umstand lassen sich auch die offensichtlichen Risiken ableiten. Nachträgliche Änderungen in Artefakten früher Projektphasen resultieren in aufwändigen und kostspieligen Konsequenzen für alle nachgeordneten Aktivitäten und deren Artefakte. Das Risiko solcher Änderungen hatte bereits Royce aufgrund des späten Rückflusses von Informationen aus der Implementierungsphase identifiziert [Roy70]. Die Konsequenz hieraus ist ein Mangel an adäquaten Methoden zum Management von Projektrisiken. Es ist somit festzuhalten, dass klassische, sequentielle Vorgehensweisen in der Praxis nur für risikoarme, idealisierte Projekte Anwendung finden können. Insbesondere Modelle wie das V-Modell XT [Bea16], welches in Abschnitt 5.3.1 näher beschrieben wird, können nur unter ständiger Kontrolle der Konsistenz aller erarbeiteten Artefakte die oben genannten Vorteile ausspielen und die resultierenden Nachteile kompensieren.

5.2.5. Agile Modelle

Agile Vorgehensweisen fußen auf der Annahme, dass ein Produkt nie vollständig definiert werden kann und die Anforderungen im Projektverlauf Anpassungen unterliegen. Sie unterscheiden sich daher grundlegend von klassischen, sequentiellen Ansätzen, indem sie die initiale Analyse- und Planungsphase nicht als exklusiv betrachten sondern diese im Projektverlauf

wiederkehrend durchführen. Darüber hinaus sind sie in der Regel weniger restriktiv und definieren das Ablauf- und Aufbau-Modell in der Regel formloser. Man spricht daher auch von „leichtgewichtigen“Ansätzen. So werden zum Beispiel im Rahmen von Scrum [SS13] (siehe Abschnitt 5.3.2) einzelne Aktivitäten nicht definiert um somit größtmögliche Flexibilität in Hinblick auf die individuellen Projekte zu bieten. Die repetitiven Abläufe des Gesamtprojektes, mit den darin enthaltenen Rollen und notwendigen Abstimmungen zwischen diesen, sind hingegen detailliert vorgegeben. Der Fokus liegt somit weniger auf den einzelnen Entwicklungsschritten als auf der Interaktion im Team, was den Grundgedanken agiler Softwareentwicklung entspricht. Agile Methoden kombinieren dabei iterative und inkrementelle Ansätze und sind somit nicht mit einem planlosen Vorgehen zu verwechseln [HK14].
Abgeleitet aus den inkrementellen Ansätzen findet sich als wiederkehrendes Element vieler agiler Methoden der Grundgedanke des schnellen Produkttests auf Basis von Prototypen und funktionsreduzierten Produktversionen wieder. Die frühzeitige Einbindung der Anwender ermöglicht so frühzeitig Anpassungen in der Produktspezifikation. Basis dessen ist der zeitnahe Start der Umsetzung individueller Fragmente und Teilfunktionen des Softwareproduktes. Die hierfür notwendigen Praktiken und Methoden werden von agilen Ansätzen in der Regel in Form eines Baukastenprinzips bereitgestellt und können so an die Bedingungen des Projektes angepasst werden [BK13]. Aus der Dokumentation des agilen Vorgehensmodells Extreme Programming [War03] werden im Folgenden beispielhaft Praktiken aufgezählt und kurz erläutert:

- **Planungsspiel**: Bei dieser Methodik planen Entwickler und Anwender kontinuierlich während des gesamten Entwicklungsverlaufs. Der Entwickler schätzt dabei den Aufwand und das Risiko eines Lösungsansatzes und der Anwender plant auf Basis der Schätzungen die folgende Entwicklung.

- **Akzeptanztests**: Zu jeder Anforderung beschreibt der Anwender auch die entsprechenden Tests. So wird ein eindeutiger Maßstab für die Fertigstellung der Produktversionen festgelegt.

- **Programmieren in Paaren**: Das Programmieren in Paaren (*Pair Programming*) ist durch die direkte Rückmeldung des Partners besonders fehlertolerant.

- **Einfaches Design**: Die Software wird so einfach gestaltet, wie es die Anforderungen der aktuellen Iteration ermöglichen. Die Anforderungen möglicher kommender Iterationen werden dabei nicht berücksichtigt, da sich diese noch ändern können.

- **Test-getriebene Entwicklung**: Es wird vor jeder Funktion der entsprechende *Unit*-Test durchgeführt.

- **Kontinuierliche Integration**: Alle Funktionen werden kontinuierlich zu einem Gesamtsystem zusammengeführt. Funktionen bleiben nur bestehen, wenn alle Tests auch nach der Integration erfolgreich verlaufen.

- **Ergebnisse-gehören-allen**: Jeder Entwickler darf jeden Quellcode zu jeder Zeit ändern (*Collective Ownership*). Diese Methodik verhindert Wissensmonopole und fördert das Verständnis im Team.

Erfahrungen mit agilen Methoden beschränkten sich in der Vergangenheit überwiegend auf Projekte mit geringem Umfang (Laufzeit unter einem Jahr und drei bis sieben Mitarbeiter) [Car12]. Ein entscheidender Faktor für diese geringe Verbreitung war der Mangel an geschultem Personal und die geringe Akzeptanz auf Kundenseite. Mittlerweile werden agile Methoden auch in Großprojekte eingesetzt [DFP19]. Die größten Risiken liegen dabei weiterhin insbesondere in menschlichen Faktoren. Zwischenmenschliche Beziehungen und eine hohe Gruppendynamik sind im Kontext agiler Vorgehensweisen entscheidend für den Projekterfolg. Aufgrund profunder Rollenbeschreibungen ist der Austausch einzelner Mitarbeiter erleichtert, was ein kompetentes Team fordert, welches flexibel auf sich ändernde Zusammensetzungen des Teams reagieren kann.
Damit geht eine gute Quantifizierbarkeit individueller und kollektiver Leistung einher, welche zum einen motivierend im Team, wie auch vertrauensbildend gegenüber dem Auftraggeber wirkt. Aufgrund der funktionalen Teilung des Produktes können einzelne Inkremente besser erfasst und bewertet werden. Die für ein Inkrement aufgewendeten Ressourcen können diesem direkt gegenüber gestellt werden. Mittels Werkzeugen, welche die Abarbeitung der Artefakte kontrollieren, geschieht dies vollautomatisch und erlaubt so ein Live-Monitoring der Teamleistung. Die Einbindung der Anwender und Auftraggeber, welche nicht zwingend die selben Personen sein müssen, in den Produktentstehungsprozess gibt diesen regelmäßige Erfolgsmeldungen und die Möglichkeit die Projektziele zu validieren. Aufgrund der hohen Eigenverantwortlichkeit der Teammitglieder kann sich jeder einzelne im Rahmen der Lösungsfindung einbringen, was nicht nur das Potential des gesamten Teams, sondern auch die Zufriedenheit eines jeden Einzelnen steigert [HK14].
Broy und Kuhrmann fassen die Herausforderung der Methodenwahl daher wie folgt zusammen: „Agile Vorgehen können in „vagen“ Szenarien mit unklaren Anforderungen ihre Vorteile ausspielen. Diese können jedoch in „stabilen“ Projekten mit klaren Anforderungen nicht zur Geltung kommen. Die Entscheidung für oder gegen ein agiles Vorgehen ist somit stark an den Projektcharakter gekoppelt.“ [BK13].

5.3. Spezifische Vorgehensmodelle

In Abschnitt 5.2 wurden Vorgehensmodelle auf abstrakter Ebene diskutiert. Im Folgenden sollen nun drei relevante Implementierungen dargestellt werden. Die Bewertung der Relevanz beruht auf der Häufigkeit ihrer Anwendung, welche von Kurmann und Linssen 2014 in einer Studie über die Anwendung von Vorgehensmodellen in Deutschland untersucht wurde [KL14]. Abbildung 5.4 zeigt eine Übersicht der populärsten Ansätze in Deutschland, welche von Scrum, dem V-Modell XT und Xtreme Programming sowie Kanban angeführt wird. Da die Kanban Methode eine hohe Ähnlichkeit zu Scrum aufweist, wird sie im Folgenden nicht diskutiert.

5.3.1. V-Modell XT

Das V-Modell XT ist ein hoch standardisiertes Vorgehensmodell, welches den kompletten Projektverlauf planbar und transparent gestalten und dabei Produkte hoher Qualität erschaffen soll [Bea16]. Neben einem umfangreichen Ablauf- und Artefaktmodell besitzt es eine detaillierte Beschreibung der Projektrollen und Verantwortlichkeiten einzelner Akteure. Es ist den klassischen sequentiellen Vorgehensmodellen zuzuordnen und ist eine Weiterentwick-

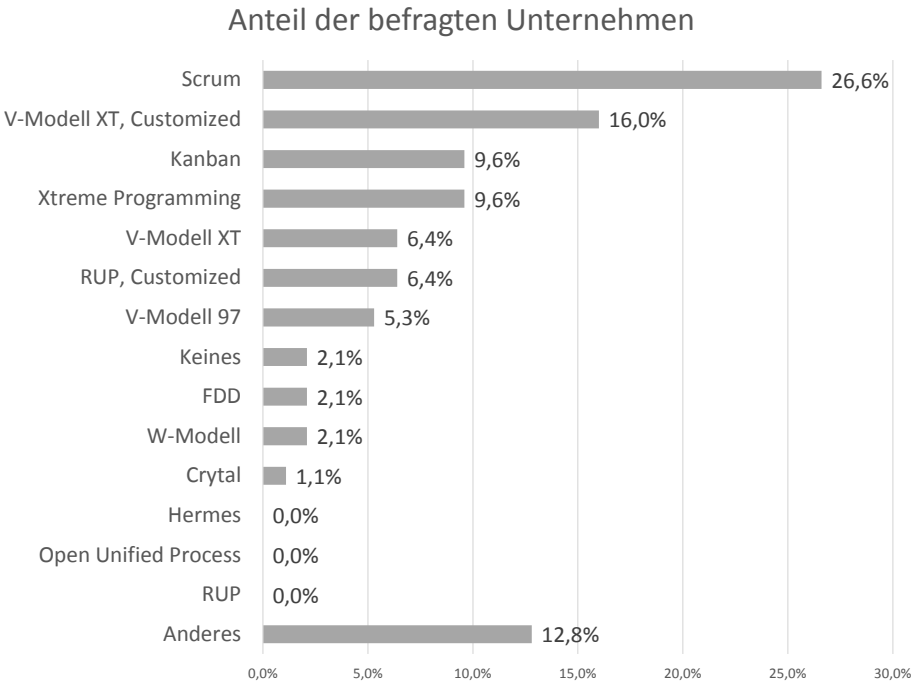

Abbildung 5.4.: Ergebnis der Studie zur Frage: Welche(r) der folgenden Entwicklungsprozesse findet/n bei Ihnen Anwendung? [KL14]

lung des klassischen V-Modell 97. Das Modell ist nach [Bea16] „[...] als Leitfaden zum Planen und Durchführen von Entwicklungsprojekten unter Berücksichtigung des gesamten Systemlebenszyklus konzipiert. Dabei definiert es die in einem Projekt zu erstellenden Ergebnisse und beschreibt die konkreten Vorgehensweisen, mit denen diese Ergebnisse erarbeitet werden“. Im Folgenden soll ein kurzer Überblick über das Modell gegeben werden:

Ablaufmodell Das Ablaufmodell des V-Modell XT bietet eine klare Projektstruktur mit eindeutig definierten Projektphasen, deren Inhalte jedoch an organisatorische Rahmenbedingungen angepasst und deren Reihenfolge individuell kombiniert werden können. Diese Offenheit wird durch einen modularen Ansatz erzielt, welcher auf den folgenden Komponenten aufbaut [Bea16]:

- **Vorgehensbaustein**: *Vorgehensbausteine* definieren die zu absolvierenden Aktivitäten mit den dafür notwendigen Kompetenzen einzelner Rollen und den daraus resultierenden Artefakten einer Aufgabenstellung. Sie sind die kleinsten Elemente des „Projekt-Baukastens“und lassen sich inhaltlich anpassen und frei arrangieren. Es ist dabei zu beachten, dass einzelne Bausteine als zwingend notwendig erachtet werden, um ein Mindestmaß an Projektdurchführungsqualität zu erzielen, wohingegen andere als optional betrachtet werden.

- **Projektdurchführungsstrategie**: Aufbauend auf den inhaltlichen Vorgaben der *Vorgehensbausteine* definiert die *Projektdurchführungsstrategie* den organisatorischen Rahmen und die Sequenz deren Abarbeitung.

- **Entscheidungspunkte**: *Entscheidungspunkte* dienen dem Projektmanagement und der Kontrolle des Projektfortschritts. Sie sind mit Meilensteinen gleichzusetzen und stehen am Ende einer Projektfortschrittstufe.
  Das V-Modell XT basiert auf der Annahme eines planungsgemäßen Projektverlaufs. Somit ist das Verfehlen eines *Entscheidungspunktes* nicht berücksichtigt. Mangelhafte Artefakte müssen korrigiert werden, bis das erwünschte Ziel erreicht wurde. Dies macht eine aufwändige Neuplanung aller folgenden Schritte notwendig.

Um die Auswahl von *Vorgehensbausteinen* und den damit verbundenen Projektdurchführungs- und Entwicklungsstrategien zu erleichtern, bietet das V-Modell XT eine Auswahl prototypischer Projektklassen. Die individuell benötigten *Vorgehensbausteine* sind den in der Dokumentation des Vorgehensmodells [Bea16] detailliert beschriebenen Projekttypen zu entnehmen.

Rollenmodell Das äußerst detailliert beschriebene Rollenmodell des V-Modell XT beschreibt die Kompetenzen aller Projektmitarbeiter für die zu bearbeitenden Aktivitäten und die daraus resultierenden Produkte. Muss eine der insgesamt 35 klar differenzierbaren Rollen neu besetzt werden, ist dies ohne Probleme auf Basis eines spezifischen Fähigkeitsprofils möglich. Es werden dabei zwei Kategorien unterschieden:

- **Organisationsrollen** werden unabhängig von einem individuellen Projekt wahrgenommen. Sie werden dauerhaft besetzt und umfassen Tätigkeiten wie zum Beispiel die Qualitätssicherung oder den Datenschutzbeauftragten.
- **Projektrollen** sind ausschließlich im Rahmen des Projektes existent. Sie sind dem Projektleiter unterstellt und füllen Funktionen im Rahmen von Entwicklungs- und Entscheidungsprozessen aus. Hierbei handelt es sich zum Beispiel um Projektleiter oder Anwender.

Das Rollenmodell des V-Modell XT ist aufgrund der individuellen Konfiguration eines Projektes mit projektspezifischen Artefakten stark kontextabhängig. Die entstehenden Rollenkombinationen sind stets für den Einzelfall zu betrachten und müssen an die projektbearbeitende Organisation angepasst werden. In dieser können einzelne Mitarbeiter mehrere Rollen übernehmen oder diese kollektiv ausfüllen.

Artefaktmodell Die zu erarbeitenden Artefakte oder auch „Produkte“leiten sich aus den gewählten *Vorgehensbausteinen* ab und beschreiben in großer Tiefe die Felder Projektmanagement, -Entwicklung und -Organisation. Besonderes Augenmerk wird dabei auf die Dokumentation des Projektes gelegt. Die hierfür notwendigen Artefakte werden als *initial* bezeichnet und sind zwingend vorgeschrieben. Bei der Auswahl der Artefakte werden die folgenden Leitfragen zugrunde gelegt [Bea16]:

- Sinn und Zweck?
- Wann ist das Produkt entscheidungsrelevant?
- Wer ist beteiligt?
- Wie erstellt man das Produkt?

- Welche Vorlagen sind verfügbar?
- Welche Abhängigkeiten hat das Produkt?

Aktivitätsmodell Parallel zum Artefaktmodell behandeln die im Aktivitätsmodell des V-Modell XT in großer Detailtiefe definierten Aktivitäten und Arbeitsschritte insbesondere Fragen des Projektmanagements sowie der Projektentwicklung und -organisation.

Werkzeugunterstützung Eine Softwareunterstützung ist bei Nutzung des V-Modell XT unabdingbar. Die große Vielfalt und Anpassbarkeit der definierten Rollen und Artefakte ist von Hand nur für kleine Projekte darstellbar und muss hier in Hinblick auf den Überhang des Verwaltungsaufwandes bei Nutzung des V-Modell XT kritisch hinterfragt werden. Die vorhandenen Werkzeuge generieren auf Basis eines auszuwählenden Projekttyps und zu spezifizierenden Projektmerkmalen das Vorgehensmodell und Grundkonzepte der zu erarbeitenden Artefakte [BK13].

5.3.2. Scrum

Scrum wurde das erste Mal von Ken Schwaber und Jeff Sutherland 1995 in einem *Business Object Design and Implementation Workshop* der *Object-Oriented Programming, Systems, Languages & Applications* Konferenz beschrieben. Seitdem findet es im Rahmen umfangreicher Projekte mit vagen Anforderungen Anwendung. Scrum basiert auf den Konzepten des iterativen und inkrementellen Vorgehens (siehe Abschnitt 5.2.5) und bietet, den Konzepten eines leichtgewichtigen Ansatzes folgend, die Möglichkeit projektspezifische Werkzeuge und Methoden für Teilprobleme einzubringen und das Vorgehen nach den lokalen Gegebenheiten auszugestalten. Das übergeordnete Ziel ist ein effektives Management von Projektrisiken und eine damit verbundene Prognosesicherheit der Projektergebnisse unter Berücksichtigung einer unspezifischen Anforderungsbasis. Grundlage des Ansatzes ist „ein Rahmenwerk, innerhalb dessen Menschen komplexe adaptive Aufgabenstellungen angehen können, und durch das sie in die Lage versetzt werden, produktiv und kreativ Produkte mit dem höchstmöglichen Wert auszuliefern.“ [SS13]. Die drei wesentlichen Teilmodelle, das Rollen-, Artefakt- und Ablaufmodell, sollen im folgenden vollständig dargestellt werden.

Ablaufmodell Ein wesentliches Merkmal von Scrum ist die repetitive Erarbeitung von Produktinkrementen in limitierten Zeitintervallen, sogenannten *Sprints*, mit einer maximalen Länge von einem Monat. Bestimmender Faktor der Sprints ist die Länge des Intervalls, an welches die zu erarbeitenden Inhalte angepasst werden müssen. Am Ende eines Sprints steht ein funktionsfähiges Produkt. Die Funktionsmerkmale (*Features*) des Produktes werden in einer initialen Planungsphase für jeden Sprint neu definiert und werden aus der angestrebten Gesamtfunktionalität des Endproduktes abgeleitet. Die strikte Teilung und Terminierung der einzelnen Prozessschritte dient der Transparenz und Kontrolle in allen Phasen des Projektes. Zwecks teaminterner Koordination werden die folgenden Meetings als zwingend erforderlich erachtet [SS13]:

- **Sprint Planning**: Dieses Meeting steht am Anfang eines jeden Sprints. In ihm werden durch das Entwicklerteam und den *Product Owner* auf Basis des *Product Backlogs* und dem aktuellen Produktinkrement die zu implementierenden *Features* des begonnenen

Sprints definiert. Basis des zu vereinbarenden Umfangs sind Erfahrungswerte der vergangenen Sprints sowie des Teams im allgemeinen. Es wird ein gemeinsames Ziel für den Sprint definiert, die sogenannte *Definition of Done*. Die vereinbarten *Features* sowie die *Definition of Done* werden im *Sprint Backlog* festgehalten und im Zuge des Sprints abgearbeitet.

- **Daily Scrum**: Zwecks täglicher Fortschrittskontrolle und gemeinsamer Koordination und Synchronisation wird durch das Entwicklungsteam ein tägliches Meeting von 15 Minuten stets zur selben Zeit am selben Ort abgehalten.
- **Sprint Review**: Am Ende eines jeden Sprints steht die Überprüfung des Produktfortschrittes mit allen Stakeholdern. In der Regel dauert dies etwa vier Stunden und wird durch den *Scrum Master* koordiniert. Im Rahmen dieses Meetings gibt es die Möglichkeit die Ziele des Projektes, also das *Product Backlog*, anzupassen.
- **Sprint Retrospektive**: Anschließend an das *Sprint Review* wird eine dreistündige Selbstreflektion des Entwicklerteams durchgeführt. An dieser nimmt auch der *Scrum Master* teil und unterstützt die Identifikation und Behebung von sozialen, fachlichen oder externen Unzulänglichkeiten. Dieses Meeting zielt auf die kontinuierliche Verbesserung der Teamarbeit ab und unterstreicht den kooperativen Charakter des Scrum-Ansatzes.

Abbildung 5.5 stellt das Ablaufmodell graphisch dar.

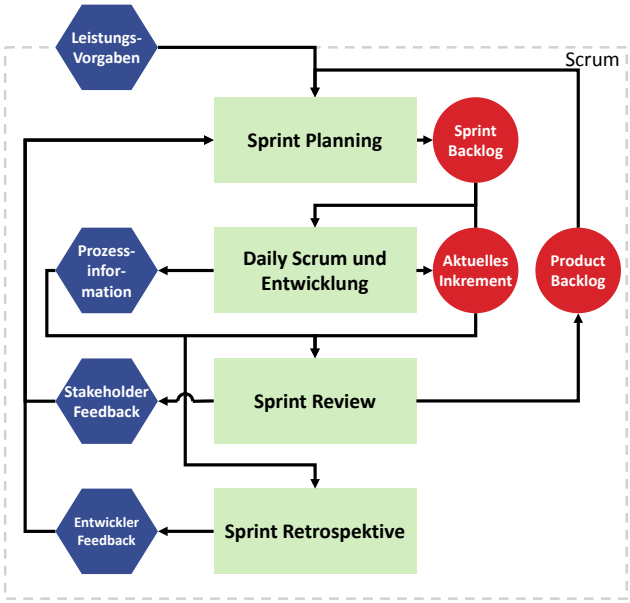

Abbildung 5.5.: Ablaufmodell Scrum nach [Luk15]

Rollenmodell Scrum definiert lediglich drei Rollen, welchen ein hoher Grad an Autonomie und Eigenverantwortung übertragen wird. Sie arbeiten interdisziplinär und besitzen alle Kompetenzen um den ihnen zugeteilten Aufgaben gerecht zu werden [SS13]. Bild 5.6 stellt den Zusammenhang der im Folgenden beschriebenen Rollen dar.

- **Product Owner**: Die Rolle des *Product Owners* wird durch eine Person ausgefüllt. Er ist mit dem Backlog Management beauftragt und hat somit die Gewalt über die in den einzelnen Sprints zu implementierenden *Features*. Im Gegenzug ist er für die Ergebnisse des Entwicklerteams rechenschaftspflichtig.
- **Entwickler**: Ein Entwicklerteam umfasst in der Regel drei bis neun Mitglieder und ist für die eigenen Arbeiten gegenüber dem *Product Owner* rechenschaftspflichtig. Nur die Entwickler dürfen *Features* des *Product Backlogs* implementieren. Die hierfür notwendigen Fähigkeiten bringen die Teammitglieder als Experten selber mit. Ist dies nicht der Fall, werden sie von der Organisation dazu befähigt die Kompetenzen auszubilden. Das Entwicklerteam arbeitet selbstständig und selbstorganisiert.
- **Scrum Master**: Der *Scrum Master* ist die Schnittstelle zwischen dem Team und der übergeordneten Organisation. Er übernimmt eine koordinierende Rolle in Hinblick auf die Zusammenarbeit des Teams und die organisatorischen Rahmenbedingungen des Projektes.

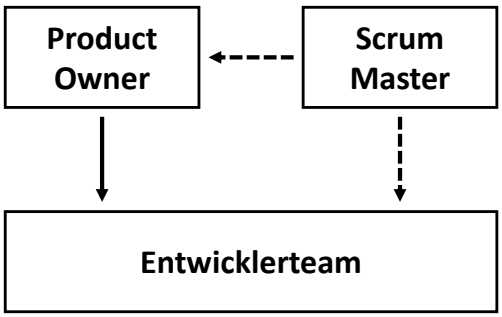

Abbildung 5.6.: Rollenmodell Scrum nach [Luk15]

Artefaktmodell Das Artefaktmodell ist, den Grundsätzen der Methode folgend, auf vier wesentliche Artefakte reduziert. So sollen nur die relevanten Informationen definiert werden, welche eine maximale Transparenz des Prozesses ermöglichen, ohne ihn einzuschränken [SS13].

- **Product Backlog**: Im *Product Backlog* werden die zu erzielenden Produkteigenschaften definiert und priorisiert. Die Produkteigenschaften werden um Schätzungen des Arbeitsumfangs, der benötigten Fähigkeiten etc. ergänzt um eine valide Sprintplanung zu ermöglichen. Die Liste ist äußerst dynamisch und unterliegt konstanten Anpassungen durch den *Product Owner*. Alle Anpassungen werden im *Product Backlog* dokumentiert. Die so gewonnenen Informationen werden für die Evaluation der weiteren Arbeitsschritte herangezogen. Das *Sprint Backlog* mit den darin abgearbeiteten Aufgaben und deren Vorhersagen und tatsächlich benötigten Ressourcen helfen Fortschrittsprognosen und Risikoabschätzungen zu generieren.
- **Sprint Backlog**: Im Rahmen des *Sprint Planning* werden die Ziele des Sprints im *Sprint Backlog* festgehalten. Es stellt somit einen Auszug des *Product Backlog*s dar und

ist ähnlich dem übergeordneten Dokument einer ständigen Anpassung unterworfen. Das *Sprint Backlog* darf nur durch das Entwicklungsteam bearbeitet werden und umfasst neben den zu erarbeitenden *Features* auch eine detaillierte Arbeitsplanung für jede einzelne Funktionalität. Das *Sprint Backlog* unterstützt das Management des einzelnen Sprints, indem es den aktuellen Arbeitsfortschritt abbildet und so eine Abschätzung des Zielerreichungsgrades ermöglicht.

- **Inkrement**: Das *Inkrement* ist ein auslieferbares Teilprodukt und stellt das Ergebnis eines Sprints dar. Es dient der Diskussion mit allen Stakeholdern im *Sprint Review* und wird im Rahmen des Risikomanagements und der Wertoptimierung genutzt. Der *Scrum Master* ist für die Transparenz des Zustandekommens dieses Artefaktes verantwortlich.
- **Definition of Done**: Um ein gemeinsames Verständnis der zu erzielenden Produkteigenschaft zu manifestieren, wird eine eindeutige Definition formuliert, wann eine Eigenschaft als implementiert erachtet wird. Dieses Artefakt ist von besonderer Bedeutung, wenn mehrere Scrum Teams kooperieren.

Aktivitätsmodell Als leichtgewichtiger Ansatz ist Scrum ein Container für unterschiedliche Methodiken. Das Scrum Team ist dafür verantwortlich die für ihre individuelle Situation passenden Werkzeuge zu wählen und die oben beschriebenen Artefakte zu erarbeiten. Auf konkrete Anweisungen zu Aktivitäten des Projektmanagements oder der Entwicklung wird verzichtet.

Werkzeugunterstützung Es werden durch Ken Schwaber und Jeff Sutherland keine konkreten Werkzeuge zur Unterstützung des Ansatzes vorgegeben. Es existieren jedoch eine Vielzahl unterschiedlicher Tools zur Organisation und Überwachung unterschiedlicher Aspekte des Vorgehensmodells. Viele der Werkzeuge sind webbasiert und lassen sich, je nach Bedarf des Teams über Webschnittstellen miteinander verknüpfen.

5.3.3. Extreme Programming

Das Vorgehensmodell des Extreme Programming (XP) wird durch seine flache Rollenstruktur und seinen starken Fokus auf die Programmieraufgabe unter Zurückstellung formaler Methoden charakterisiert. Die autarken Arbeitsgruppen nutzen Techniken der agilen Softwareentwicklung und erarbeiten Artefakte selbstständig. Aufgrund seiner großen Verbreitung soll das XP-Modell nach Kent Beck [BA05] im Folgenden näher beschrieben werden.

Ablaufmodell Das extrem reduzierte Ablaufmodell soll im XP lediglich als Rahmen für die Anwendung agiler Entwicklungsmethoden (siehe Abschnitt 5.2.5) dienen. Es gliedert sich zu diesem Zweck in die drei Phasen: Planung-, Test- und Entwicklungsphase sowie eine anschließende Qualitätskontrolle in Form von *Acceptance Tests*. Über diese Vierteilung und die Auswahl projektspezifischer Vorgehensmodelle für die einzelnen Phasen lässt sich das XP auf alle Lebensphasen eines Produktes anwenden. So beschreibt Beck explizit die Wartung und Weiterentwicklung der Software und hebt den Erkenntnisgewinn im Rahmen der Betreuung der Software in diesen Phasen hervor. Der XP-Prozess ist dann zu beenden, wenn ein System den Anforderungen des Anwenders zur Gänze entspricht oder aber nicht mehr ökonomisch weiter betrieben werden kann [BA05]. Zwecks Erhalt des gewonnenen Wissens steht am Ende

stets eine ausführliche Dokumentation des Produktes und des Projekts. Abbildung 5.7 zeigt das gesamte Ablaufmodell. Die darin dargestellten Phasen werden im Weiteren erläutert.

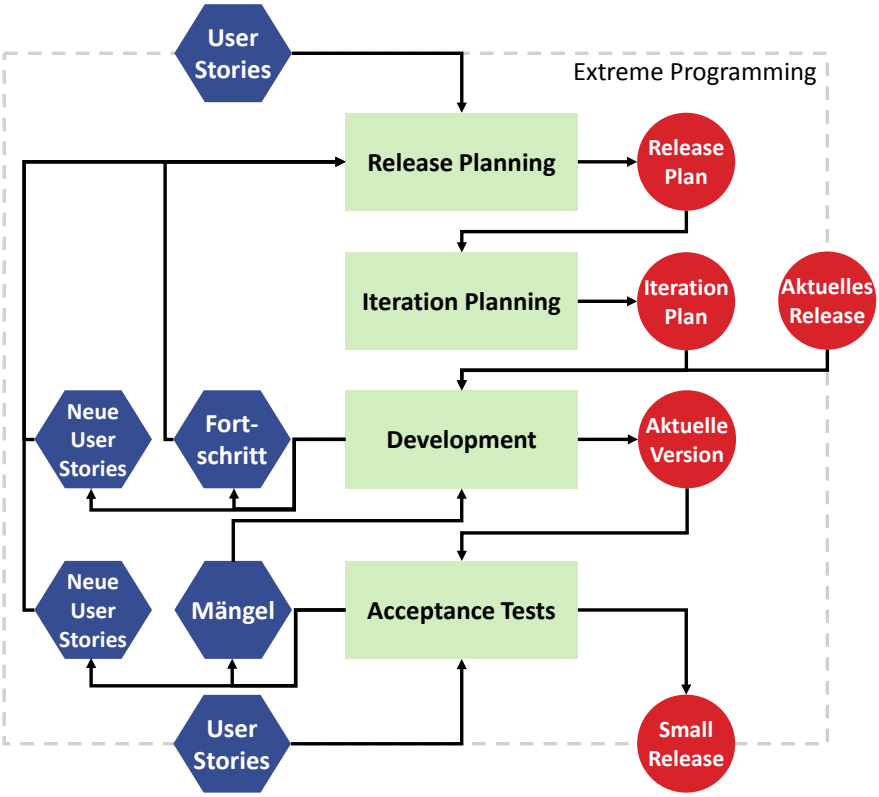

Abbildung 5.7.: Ablaufmodell Extreme Programming nach [Luk15]

- **Release Planning**: Die initiale Projektphase dient der Anforderungserhebung und Zeitplanung. Die Anforderungen des Anwenders werden als *User Stories* formuliert und in Form von *Story Cards* als Eingangsparameter des *Release Plannings* genutzt. Durch eine intensive Zusammenarbeit zwischen Programmierern und Anwendern werden die Grundlagen für eine weitere Zusammenarbeit gelegt. Zwecks Abschätzung des Implementierungsaufwandes werden auf Basis der *User Stories* Prototypen entworfen und zum Teil umgesetzt. Die Qualität der Lösungsfindung wird dabei durch die parallele, von einander getrennte, Ausarbeitung von Lösungen für die selbe Fragestellung erhöht. Die Formulierung der *Story Cards* sowie die Kompetenzen des Teams werden in diesem Schritt evaluiert und auf Basis dessen der Arbeitsumfang für die einzelnen User Stories abgeschätzt. Auch wenn zu diesem Zeitpunkt noch keine lauffähige Software vorliegt, kann die Komplexität der Problemstellung auf Basis der gewonnenen Erkenntnisse bereits gut abgeschätzt werden. Die sich daraus ableitenden Systemgrenzen sowie gangbare Lösungswege lassen sich ebenfalls abstecken. Nach ein bis zwei Wochen sind diese initialen Erhebungen abgeschlossen. Es folgt die Priorisierung der *Story Cards* durch den Anwender und die Erstellung des *Release Plans* durch die Programmierer. Um Abhängigkeiten zwischen den priorisierten *Story Cards* im Rahmen der Planung zu berücksichtigen und die sich hieraus ergebenden zeitlichen Abläufe besser modellieren

zu können, finden im Kreise der *Entwickler* sogenannte *Planning Games* Anwendung. Der hier definierte *Release Plan* ist ein lebendes Dokument und kann im Laufe des Projektes auf Basis des praktischen Projektfortschrittes oder sich ändernder Anforderungen angepasst werden [War03]. Abbildung 5.8 stellt das Release Planning im Zuge der XP schematisch dar.

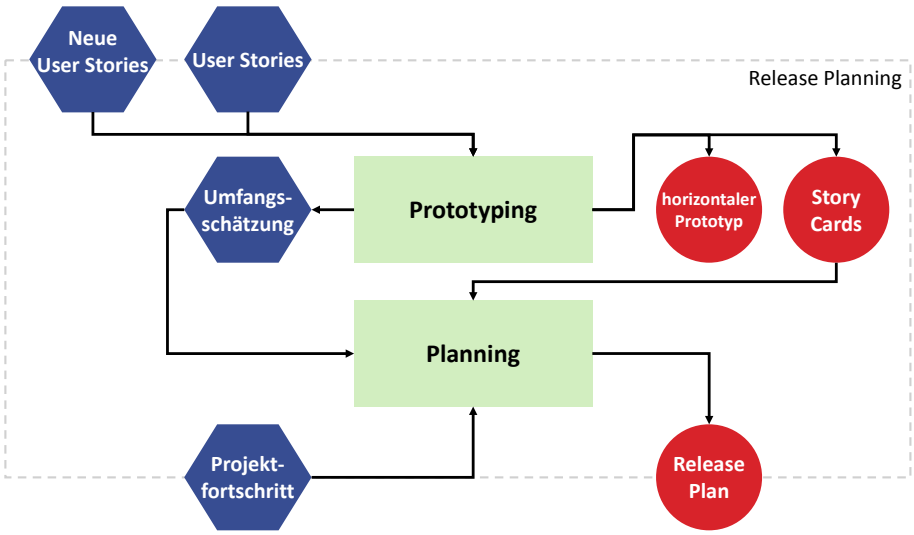

Abbildung 5.8.: Ablaufmodell Release Planning nach [Luk15]

- **Iteration Planning**: Das *Iteration Planning* steht stets am Anfang einer Iteration und umfasst die Auswahl der zu implementierenden *Story Cards*. Die getroffene Auswahl wird im *Iteration Plan* festgehalten. Die erste Iteration - mit einem zeitlichen Umfang von zwei bis sechs Monaten - dient dabei der Erstellung der grundlegenden Software, welche im Zuge der weiteren Entwicklung als verbindendes Rahmenwerk dienen kann. Diese wird als *First Release* bezeichnet. Die folgenden Iterationen, von ein bis vier Wochen Umfang, erweitern diesen Rahmen um spezifische Artefakte. Die Auswahl der zu implementierenden Artefakte basiert auf der im Rahmen des *Release Plannings* getroffenen Priorisierung.

- **Development**: Im Zuge des *Development* wird der *Iteration Plan* abgearbeitet. Einzelne Artefakte werden den Programmierern in Form von *Task Cards* zugeteilt. Diese formulieren für jede von ihnen entwickelte Erweiterung individuelle *Unit*-Tests. Nach einem individuellen Test, wird der Code integriert und als Programmversion einem automatisiertem Systemtest unterzogen. Diese Integration erfolgt idealerweise am Ende eines jeden Arbeitstages. Der aktuelle Projektfortschritt wird dann am nächsten Morgen im Form eines *Stand Up Meetings* ermittelt, und neue *Task Cards* werden definiert.

- **Acceptance Tests**: Die Qualitätskontrolle im Zuge der *Acceptance Tests* wird durch den Anwender durchgeführt. Ist dieser mit der Umsetzung der *User Stories* einverstanden, wird die Software als *Small Release* freigegeben. Werden von ihm Probleme angemerkt, wird die *Story Card* in der kommenden Iteration weiter bearbeitet. Aufgrund der akuten Auseinandersetzung mit dem Produkt entstehen in dieser Phase häufig neue *User Stories*.

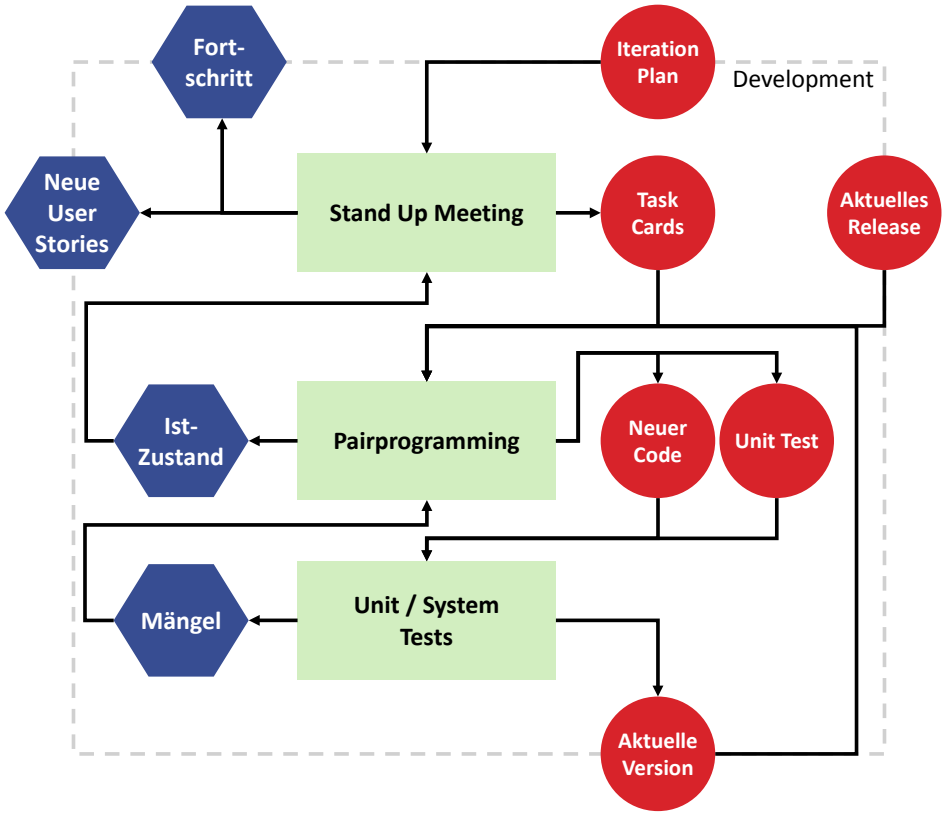

Abbildung 5.9.: Ablaufmodell Development nach [Luk15]

Rollenmodell Das XP Rollenmodell unterscheidet die folgenden sieben Rollen [BA05]:

- **Programmierer**: Die wichtigste Ressource des XP Modells sind Programmierer. Sie implementieren die im *Iteration Planning* definierten Artefakte. Zwecks Steigerung der Produktqualität arbeiten sie in der Regel im Zweierteam. Diese Zusammenarbeit, welche *Pair Programming* genannt wird, erfordert eine beständige Kommunikation zwischen den beiden Partnern, welche sich einen Bildschirm teilen. Die so geformten Teams evaluieren den aktuellen Fortschritt sowie den zeitlichen Umfang ihrer Aufgabe. Darüber hinaus setzten sie die *Unit-* und *System Tests* um.

- **Anwender**: Der Anwender definiert die zu implementierenden Artefakte in Form von *User Stories* und leitet daraus gemeinsam mit den Programmierern die *Story Cards* ab. Er legt, gemeinsam mit dem Tester, mittels funktionaler Tests, sogenannter *Acceptance Tests*, die Qualitätskriterien des Softwareprojektes fest.

- **Tester**: Die *Acceptance Tests* werden maßgeblich durch den Tester konzipiert und umgesetzt. Er konsultiert den Anwender zwecks Freigabe und präsentiert dem Entwicklungsteam die Testergebnisse.

- **Tracker**: Der *Tracker* protokolliert alle Änderungen und Zwischenstände des Projektes. Dazu zählen unter anderem funktionale Anpassungen, Testergebnisse und Umfangsschätzungen. Letztere dienen der Evaluation des Projektfortschrittes und der Selbsteinschätzung der Programmierer.

- **Coach**: Die Rolle des *Coach* dient der Einhaltung der XP Prinzipen. Er hat die Kompetenzen in die Entwicklung einzugreifen, wenn er dafür die Notwendigkeit sieht. Es ist an ihm, die agilen Methoden zu vermitteln und zu optimieren.
- **Externe Experten**: Sollte ein Projekt Kompetenzen erfordern, welche in einem Team nicht vorhanden sind, so kann ein externer Experte hinzugezogen werden. Dieser kann dann die benötigten Kompetenzen vermitteln oder im Rahmen der Lösung eine Führungsrolle übernehmen.
- **Big Boss**: Der *Big Boss* entspricht der Rolle eines Projekt Managers. Er ist die Schnittstelle des Teams nach außen und zeichnet für das Erreichen des Projektziels verantwortlich. Vergleichbar dem *Coach* kann er in den Prozess eingreifen und gemäß seiner Vorstellungen lenken.

Artefaktmodell Das XP beschreibt Artefakte nur sehr formlos [BA05]. Die zentralen Artefakte des leichtgewichtigen Modells sollen im Weiteren dargestellt werden:

- **Story Card**: Die *Story Card* ist die formlose Dokumentation einer *User Story*, also einer Anforderung an die zu erstellende Software. Die Formulierung soll für den Anwender wie auch für die Programmierer verständlich sein und eine Abschätzung des Umfangs durch einen Programmierer ermöglichen. Sie ist das Produkt des *Planning Games.*
- **Release Plan**: Auf Basis der definierten *Story Cards* und deren Umfänge definiert der Anwender den Umfang des nächsten Releases. Das Ergebnis ist der *Release Plan.*
- **Iteration Plan**: Im Rahmen des *Release Plan* wird die Feinplanung einzelner Iterationen durch die Programmierer selbst durchgeführt. Sie Erstellen die *Iteration Planning.* Einzelnen Programmierern werden individuelle Aufgaben in Form von *Task Cards* zugewiesen.
- **Acceptance Test**: Zwecks Sicherung der Produktqualität werden parallel zur Erstellung der Storycards in Kooperation des *Testers* und des *Anwenders* Tests für individuelle Anforderungen entworfen. Sie dienen als Abnahmekriterien.
- **Unit/System Test**: Es handelt sich bei den *Unit-* und *System Tests* um automatisierte Tests, welche im Rahmen der Integration individueller Komponenten durchgeführt werden. Schlagen die Tests fehl, wird die Integration nicht akzeptiert. So wird sichergestellt, dass stets ein funktionsfähiges Programm vorliegt.
- **Small Release**: Ein *Small Release* ist das Ergebnis einer Iteration. Der Funktionsumfang ist in der Regel limitiert, um eine zeitnahe Anwendung in der Praxis zu ermöglichen.

Aktivitätsmodell Das XP Modell besitzt in Ermangelung formaler Vorgaben zur Erstellung von Artefakten kein explizites Aktivitätsmodell. Der Fokus des XP liegt primär auf Modellwerten und Prinzipien.

Werkzeugunterstützung Es ist möglich eine Vielzahl generischer Werkzeuge, welche für die Domäne der agilen Entwicklungsmethoden geschaffen wurden, an die Bedürfnisse des XP anzupassen. Insbesondere im Zuge der kontinuierlichen Integration ist es vorteilhaft eine automatisierte Werkzeugkette zu etablieren.

5.4. Agentenspezifische Ansätze des Software Engineerings

Agentenbasierte Konzepte können mittels generischer Entwicklungsansätze umgesetzt werden. Häufig fehlen jedoch spezifische Methoden und organisatorische Grundlagen um die charakteristischen Eigenschaften von Agenten-Systemen ausreichend abzubilden. Eine Möglichkeit, diese in einen „klassichen“ Entwicklungsprozess einzubringen, sind Referenzarchitekturen, an welchen sich das Design des zu schaffenden Agentensystems orientiert. Farid und Ribeiro präsentieren eine solche für rekonfigurierbare Fertigungssysteme in [FR15]. Der Nachteil dieses Vorgehens ist die damit verbundene Begrenzung des Lösungsraum auf die starre „Kopiervorlage“, an welcher sich der *Entwickler* orientieren muss. Andere Ansätze fokussieren im Gegensatz hierzu auf die elementaren Eigenschaften der zu schaffenden Agenten und erlauben so den Entwurf komplexer Agentensysteme ohne strukturelle Vorgaben. Im Folgenden sollen daher drei relevante Methoden dieser Gattung dargestellt werden. Die Bewertung der Relevanz beruht auf der Anzahl an Zitationen der ursprünglichen Veröffentlichungen der Konzepte. Im Bereich des agentenorientierten Engineerings existiert eine Vielzahl un-

Rang	Titel	Quelle	Anzahl Zitationen
1	Gaia	[WJK00]	2621
2	Tropos	[BPG+02]	1847
3	MaSE	[DWS01]	692
4	Kinny	[KG96]	635
5	Prometheus	[PW04]	570
6	MAS-CommonKADS	[IGGV97]	473
7	MESSAGE	[CCG+02]	415
8	SODA	[Omi01]	338
9	PASSI	[Cos05]	243
10	AML	[TC05]	87

Tabelle 5.1.: Relevanz agentenorientierter Entwicklungsansätze, Stand: 26.07.2017, Quelle: scholar.google.com

terschiedlicher Methoden mit Domänen- und anwendungsspezifischen Herangehensweisen. So gibt es Methoden, welche im Bereich der Sozialwissenschaften Anwendung finden oder auch Probleme im Bereich mobiler Roboter adressieren. Die Gesellschaft Mess- und Automatisierungstechnik (GMA) des Vereins Deutscher Ingenieure (VDI) hat im Rahmen des Fachausschuss 5.15 „Agentensysteme“ die Richtlinienreihe VDI/VDE 2653 erarbeitet, welche sich den Herausforderungen von Agentensystemen in der Automatisierungstechnik widmet. In Blatt 2 [VDI] werden zehn Entwicklungsansätze aufgezählt, welche in der Domäne der Automatisierungstechnik Anwendung finden können. Tabelle 5.1 zeigt die Anzahl an Zitationen der individuellen Ursprungsquellen. Die Erhebung wurde am 26.07.2017 mittels Google

Scholar durchgeführt. Die drei führenden Ansätze Gaia [WJK00], Tropos [BPG+02] und MaSE [DWS01] stammen alle aus den Jahren 2000 bis 2002, also der Zeit des „Agenten-Hypes“. Doch auch heute noch sind sie im Kontext der agentenorientierten Softwareentwicklung führend. Aktuelle Untersuchungen und Entwicklungen im Bereich der Multi-Agentensysteme basieren weiterhin auf diesen etablierten Klassikern (Siehe z.B. [SKS18],[AM18],[MS18] und [HBH18]). Sie sollen daher im Folgenden diskutiert werden.
Alle hier aufgeführten Ansätze fokussieren auf die Inhalte der Softwareentwicklung, Anforderungen an das Projektmanagement werden - wenn überhaupt - nur äußerst knapp und oberflächlich definiert.

5.4.1. Gaia

Gaia („Generic Architecture for Information Availability“) ist eine Entwicklungsmethode, welche im Rahmen des Organisationsentwurfes von Agentensystemen Anwendung findet. Sie wurde im Jahr 2000 von Michael Wooldridge, Nick Jennings und David Kinny erstmals veröffentlicht und soll Softwareentwickler dabei unterstützen die Entwicklung von Agentensysteme als „process of organisational design“[WJK00] zu begreifen. Der Ansatz deckt die Analyse- und Entwurfsphase ab und stellt dem *Entwickler* frei, mit welchen Methoden die Anforderungen ermittelt und die im Rahmen des Prozesses erarbeiteten Modelle implementiert werden. Gaia stellt einen Domänen-neutralen Entwicklungsansatz dar, welcher sowohl die Modellierung der Organisation, die sogenannte Makro-Ebene, also auch die Mikro-Ebene, also die Modellierung individueller Agenten, berücksichtigt. Gaia setzt dabei die folgenden Randbedingungen als gegeben voraus:

- **Komplexität**: Individuelle Agenten sind inhärent komplex und benötigen erhebliche Systemressourcen
- **Kooperation**: Es wird ein kooperatives Agenten-Design angestrebt
- **Statische Struktur**: Inter-Agenten-Beziehungen sind statisch und ändern sich nicht während der Laufzeit
- **Statischer Funktionsumfang**: Die Fähigkeiten der Agenten ändern sich nicht während der Laufzeit
- **Systemgröße**: Das System sollte weniger als 100 Agententypen umfassen
- **Heterogenität**: Agenten können in unterschiedlichen Programmiersprachen und Architekturen implementiert werden

Ablaufmodell Das Ablaufmodell von Gaia umfasst zwei Phasen: Eine Analyse- und eine Entwurfsphase. Im Zuge der Analysephase soll der *Entwickler*, unabhängig von Implementierungsdetails, ein tieferes Verständnis der „Organisation“des Systems, also seiner Struktur, den darin zu erfüllenden Aufgaben und deren Beziehung untereinander, gewinnen. In der darauf folgenden Entwurfsphase sollen die im Vorfeld erarbeiteten abstrakten Modelle soweit spezifiziert werden, dass sie unter Zuhilfenahme traditioneller Entwicklungsmethoden in Form individueller Agenten implementiert werden können.

Rollenmodell Gaia besitzt kein explizites Rollenmodell für das Entwicklerteam.

Artefaktmodell Das Kernstück des Entwicklungsansatzes ist ein auf die Bedürfnisse der agentenorientierten Softwareentwicklung ausgerichtetes Artefaktmodell. Diese umfasst im wesentlichen die fünf in Abbildung 5.10 dargestellten Modelle, für welche die im Vorfeld zu erarbeitenden Systemanforderungen als Eingangsparameter dienen.

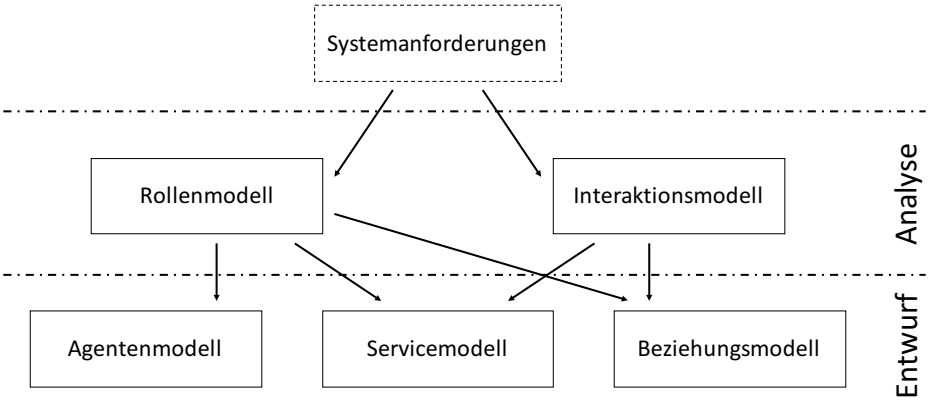

Abbildung 5.10.: Artefaktmodell Gaia, nach [WJ05]

Zunächst werden im Rahmen der Analysephase Rollen identifiziert und basierend darauf zwei abstrakte Modelle erstellt:

- **Interaktionsmodell**: Zunächst wird das Interaktionsmodell definiert. Es beschreibt für jede Interaktion ein zugehöriges Protokoll. Als Interaktion wird dabei der Austausch zwischen Agenten als Träger von spezifischen Rollen definiert. Das so geschaffene Protokoll umfasst nicht die Syntax individueller Nachrichten, sondern Attribute wie den Zweck der Interaktion, die Rollen der Kommunikationspartner, bereitzustellende Informationen und daraus abzuleitende Reaktionen.

- **Rollenmodell**: Im Anschluss werden die Eigenschaften der identifizierten Rollen schriftlich fixiert. Der Fokus liegt dabei auf den folgenden vier *Rollenattributen*, welche für alle Rollen in Tabellenform dargestellt werden:
  - **Protokolle** - umfasst die Namen aller für die Rolle relevanten Interaktionsprotokolle
  - **Aktivitäten** - umfasst die Namen aller internen Aktivitäten einer Rolle
  - **Rechte** - definiert die Rechte einer Rolle zum Zugriff auf Ressourcen
  - **Verantwortlichkeiten** - beschreibt die Funktionalitäten einer Rolle. Es werden dabei „liveness expressions“,welche mit Zielen des Gesamtsystems korrelieren, und „safety conditions“, welche invarianten Randbedingungen des Gesamtsystems berücksichtigen, unterschieden.

Nach der Erstellung des Rollenmodells wird geprüft, ob alle geforderten Eigenschaften durch das erarbeitete Modell abgedeckt werden. Ist dies nicht der Fall, werden die Phasen Rollenidentifikation, Protokolldefinition und Rollendefinition iteriert, bis der gewünschte Zustand erreicht ist.

In der anschließenden Entwurfsphase werden dann aus den abstrakten Modellen der Analysephase die drei folgenden Modelle erarbeitet, welche als Eingangsparameter für eine spätere Implementierung dienen sollen:

- **Agentenmodell**: Das Agentenmodell stellt alle zu entwickelnden Agententypen und deren Instanzen zur Laufzeit dar. Agenten werden dabei als eine Sammlung von Rollen verstanden. Es erfolgt eine Notation in Form eines *Agenten-Typ-Baumes*, in welchem sowohl Rollen als auch Agenten als Knoten dargestellt werden. Beziehungen untereinander werden durch eine an den objektorientierten Ansatz *FUSION* [CAB+94] angelehnte Notation dargestellt. Das Konzept der Vererbung wird nicht berücksichtigt, da die Autoren davon ausgehen, dass es sich in den meisten Fällen um eine eins-zu-eins Beziehung zwischen Agenten und Rollen handelt.

- **Servicemodell**: Seinem Namen folgend, definiert das Servicemodell alle Services, welche zur Erfüllung der im Vorfeld definierten Rollen benötigt werden. Services können mit Methoden objektorientierter Ansätze verglichen werden, mit dem entscheidenden Unterschied, dass sie nicht durch andere Objekte / Agenten aufgerufen werden können. Es muss jede im Rahmen der Analysephase definierte Aktivität und jede Rolle mit einem Service verknüpft werden, jedoch nicht jeder Service mit einer Aktivität.
  Services werden dabei durch ihre Eingänge, Ausgänge, Vor- und Nachbedingungen definiert. Diese können sowohl aus dem Protokollmodell als auch aus den „safety conditions“der Verantwortlichkeiten abgeleitet werden.

- **Beziehungsmodell**: Das einfachste Modell der Gaia Methode ist das Beziehungsmodell. Es beschreibt zwischen welchen Agenten Kommunikationsbeziehungen existieren. Diese werden als gerichtete Graphen dargestellt. Die Inhalte der Nachrichten werden ebenso wenig vermerkt wie zeitliche Zusammenhänge, da das Ziel dieses Modells ist, Engpässe in der Kommunikation zu identifizieren und durch eine Revision der Analysephase zu eliminieren.

Aktivitätsmodell Das in Gaia präsentierte Aktivitätsmodell stellt zwar inhaltliche, zeitliche und organisatorische Zusammenhänge zwischen den oben beschriebenen Artefakten dar, macht jedoch keine Vorgaben, in welcher Form die Inhalte der klar beschriebenen Artefakte erarbeitet werden sollen. Ebenso wenig werden Aussagen über das Projektmanagement oder andere nicht der reinen Erarbeitung von Artefakten zuzuordnenden Tätigkeiten getroffen.

Werkzeugunterstützung Die in Gaia beschriebenen Modelle und deren Interdependenzen lassen sich in ihrer ursprünglichen Ausprägung mittels Stift und Papier umsetzen. Es existiert jedoch ein Gaia-spezifisches Plugin für die integrierte Entwicklungsumgebung (IDE) *Eclipse*[2] [CZ09]. Eine alternative Repräsentation in Form von Agent UML erlaubt darüber hinaus die Nutzung weiterer generischer Werkzeuge [CZ04].

5.4.2. Tropos

Tropos wurde 2002 von Paolo Bresciani et. al. vorgestellt, um die Potentiale der frühen Phasen des Entwicklungsprozesses für die Gestaltung von Agentensystemen besser nutzbar

[2] *Eclipse* ist eine quelloffene Integrierte Entwicklungsumgebung, siehe: https://eclipse.org/ide/

zu machen [BPG+02]. Es wurde mit *Tropos* das erste Mal ein Ansatz vorgestellt, welcher von der Anforderungserhebung bis hin zur Implementierung die Grundideen eines Agenten in den Kern des Prozesses stellt und iterativ weiter entwickelt. Zwar sind das Vorgehen und die zugrunde liegenden Modelle auf Believe, Desire, Intention (BDI) Architekturen ausgerichtet, erlauben aber auch die Entwicklung von deliberativen Agenten.

Ablaufmodell Das Ablaufmodell von Tropos umfasst fünf Phasen:

- **Initiale Anforderungserhebung** (IA): In einem ersten Schritt werden aus der Zielumgebung des zu implementierenden Systems Anforderungen abgeleitet. Diese werden in Form von Ziel orientierten Akteuren modelliert und ihre Abhängigkeiten untereinander in einen organisatorischen Zusammenhang gesetzt.
- **Weitere Anforderungserhebung** (WA): Die Weitere Anforderungserhebung legt den Fokus auf das zu implementierende System und leitet daraus weitere Anforderungen ab.
- **Architektur-Entwurf** (AE): Es folgt die Phase des Architektur-Entwurfs, in welcher die Architektur des zu implementierenden Systems in Form interagierender Subsysteme dargestellt wird.
- **Detail-Entwurf** (DE): Aufbauend auf der im vorangegangenen Schritt entwickelten Systemarchitektur werden im Detail-Entwurf die Funktionalitäten der individuellen Agenten sowie deren Interaktionsprotokolle spezifiziert.
- **Implementierung und Tests**: In dieser Phase erfolgt die eigentliche Implementierung der Software-Agenten. Darüber hinaus werden die Funktionalitäten und Interaktionen der erstellen Software in Hinblick auf ihre Spezifikation verifiziert.

Diese fünf Phasen können iterativ durchlaufen werden und erlauben somit die nachträgliche Ergänzung der initialen Anforderungserhebung oder der Verfeinerung der Entwurfsphasen [CHMS14].

Rollenmodell Tropos besitzt ein klar definiertes Rollenmodell, welches die folgenden sieben Rollen umfasst:

- **Anforderungs-Analyst**: erarbeitet die grundlegenden Modelle der *Initialen Anforder-gunserhebung*. Im Rahmen der *Weiteren Anforderungserhebung* verfeinert er die Systemstruktur.
- **Domänen-Stakeholder**: besitzen das Domänenwissen und stellen dies dem *Anforderungs-Analysten* zur Verfügung.
- **System Architekt**: ist für die Harmonisierung und Strukturierung des Gesamtsystems verantwortlich.
- **Agenten-Architekt**: spezifiziert die Ziele und zugrundeliegenden Fähigkeiten der individuellen Agenten. Im Zuge der *Weiteren Anforderungserhebung* verfeinert er diese bis hin zu den konkreten Funktionalitäten des Agenten.

- **Software-Architekt**: ist für das Detail-Design der einzelnen Agenten und des Gesamtsystems verantwortlich. Darüber hinaus entscheidet er, in welcher Form und auf welcher Plattform die Software-Entwicklung vorangetrieben werden soll.
- **Agenten-Programmierer**: implementiert und verteilt das Agentensystem auf der Test- und Zielhardware.
- **Agenten-Tester**: entwickelt auf Basis der Ziel- und Sequenz-Diagramme Testfälle, welche er auf dem Agentensystem ausführt.

Artefaktmodell Tropos ist ein modellbasierter Engineering-Prozess, welcher sich semantisch an *i** orientiert. *i** verfolgt das Ziel, nicht nur das „Was“und „Wie“in die Modellierung eines Prozesses einzubeziehen, sondern auch das „Warum“abzubilden [Yu195]. Die hieraus abgeleiteten Konzepte der zielorientierten Entwicklungssprache sind *Akteur*, *Ziel*, *Plan* und *Soziale Abhängigkeit*.
Die zu erstellenden Modelle lassen sich aus diesen Konzepten ableiten. In der **Initialen Anforderungserhebung** werden die Grundlagen der weiteren Ausarbeitung geschaffen. Diese umfassen:

- **Domänen-Dokumentation**: Enthält das Wissen über die zu betrachtende Domäne. Dieses wird durch Literaturarbeit und Experteninterviews erarbeitet.
- **IA Akteur Diagramm**: Dies ist das erste Tropos-spezifische Artefakt. Es bildet alle an einem Prozess beteiligten Elemente (Software, Nutzer, externe Komponenten) mit ihren Zielen und Abhängigkeiten ab.
- **IA Ziel Diagramm**: Aufbauend auf dem zuvor geschaffenen *IA Akteur Diagramm* werden hier die funktionalen- und nicht-funktionalen Anforderungen, im Folgenden die „Ziele“, der einzelnen Akteure weiter analysiert. Ihre logischen Grundlagen (im Englischen „rationale“) werden herausgearbeitet und in drei Kategorien unterteilt:

  Verfeinert: Umfasst Sub-Ziele des generischen Ziels der strategischen Ebene des Akteur Modells. Beziehungen können UND- und ODER-Beziehungen umfassen.

  Delegiert: Beschreibt Ziele, welche an andere Akteure delegiert werden können, bzw. von diesen an den zu analysierenden Akteur delegiert wurden.

  Beteiligt: Beschreibt Beziehungen zwischen einzelnen Zielen. Ist die Erlangung eines Ziels von anderen Zielen abhängig, wird dies im *IA Ziel Diagramm* festgehalten.

Die *Weitere Anforderungserhebung* ergänzt die Ergebnisse der *Initialen Anforderungserhebung* um die Ziele des Gesamtsystems. Die Einführung des „Systems“als Akteur kennzeichnet den Übergang zwischen den beiden Phasen.

- **WA Akteur Diagramm**: Ergänzend zu den im Rahmen des *IA Akteur Diagramm* abgebildeten Prozessbeteiligten wird im Rahmen der weiteren Anforderungserhebung das System als zusätzlicher Akteur eingeführt. Das *WA Akteur Diagramm* enthält somit die Verbindungen des Gesamtsystems zu seiner Umgebung.
- **WA Ziel Diagramm**: Unterstützt die Operationalisierung der strategischen Hauptziele des Gesamtsystems. Zudem stellt es die taktischen Subziele und alternative Zielerreichungspfade dar.

- **WA Fähigkeitsmodell**: Unterstützt die Operationalisierung der funktionalen Hauptziele des Gesamtsystems durch Überführung der individuellen Ziele in konkreten Plänen. Zudem werden deren Zusammenhänge mit nichtfunktionalen Anforderungen dargestellt.

Im Zuge des *Architektur-Entwurfs* wird sowohl die Gesamtstruktur des *Multi-Agent-System* (MAS) als auch das detaillierte Design der individuellen Agenten erarbeitet. Die Ergebnisse dieser Phase sind:

- **AE Akteur Diagramm**: Die Akteur Diagramme der vorangegangenen Schritte werden um individuelle Agenten ergänzt. Diese besitzen sowohl Beziehungen zum Gesamtsystem als auch untereinander.
- **AE Ziel Diagramme**: Repräsentieren die Ziele aller identifizierten Agenten.
- **AE Fähigkeitsmodell**: Zählt alle Agenten mit ihren Fähigkeiten und deren Verbindungen zu nichtfunktionalen Anforderungen auf.

Der *Detail-Entwurf* schafft die Voraussetzungen für die spätere Implementierung. Hierzu werden die folgenden Artefakte erstellt:

- **DE Ziel Diagramme**: In dieser finalen Überarbeitung des Ziel Diagramms werden die Ziele und Pläne der einzelnen Agenten in Sub-Pläne heruntergebrochen, welche Funktionalitäten der Agenten entsprechen.
- **UML Aktivitäts Diagramm**: Spezifiziert die Aktivitäten, aus welchen Pläne aufgebaut sind und legt deren zeitliche Reihenfolge fest.
- **UML Sequenz Diagramm**: Spezifiziert die Interaktion zwischen zwei Akteuren und deren Effekte.
- **UML Komponenten und Klassendiagramme**: Ist die Zielsprache objektorientiert, werden alle Komponenten des Systems als Klassen dargestellt.
- **BDI Agenten Struktur**: Basiert das Zielsystem auf BDI Konzepten, werden hier in Anlehnung an JADEX [PBL05] Agent Definition Files erstellt.

Im Zuge der anschließenden Implementierung wird nicht nur der eigentliche Agenten-Code erzeugt, sondern auch Testfälle werden definiert, welche sowohl Funktionalitäten des individuellen Agenten als auch des gesamten Systems betrachten.

Aktivitätsmodell Tropos definiert das zugrundeliegende Aktivitätsmodell in hoher Tiefe. Die einzelnen Arbeitsschritte werden Verantwortlichen zugeordnet und alle notwendigen Ein- und Ausgangsparameter in konsistenter Form festgelegt. Tropos macht keine Vorgaben zum Projektmanagement, sondern fokussiert lediglich auf den Kern der Software-Erstellung.

Werkzeugunterstützung Die Nutzung von Tropos wird durch mehrere Werkzeuge unterstützt. Dazu zählen unter anderem *TAOM4e*, welches die Zielmodellierung im gesamten Lebenszyklus abdeckt [MNP+08], [PS05], *t2x*, welches sich auf die Implementierungsphase fokussiert [MPP09], das *T-Tool*, welches es erlaubt Modelle zu prüfen [FPMT01], sowie das *GR-Tool*, welches formales Schließen auf Ziel-Modell Ebene erlaubt [GMS05].

5.4.3. O-MaSE

Organization-based Multiagent Software Engineering (O-MaSE) ist eine von Scott A. DeLoach und Juan Carlos García-Ojeda vorgestellte Weiterentwicklung der *MASE*-Methode [DGO10]. Das Ziel von O-MaSE ist es, Entwickler mittels vordefinierter Methodenfragmente zu befähigen individuelle Entwicklungsprozesse für agentenbasierte Softwaresysteme zusammenzustellen. Diese Methoden-Fragmente sind Tätigkeiten, welche ausgeführt werden müssen, um agentenspezifische Artefakte in einem systematischen Prozess zu erstellen. O-MaSE basiert dabei auf drei wesentlichen Säulen: 1. einem Metamodell, 2. den Methoden-Fragmenten sowie 3. Richtlinien zur Zusammenstellung der Fragmente [CHMS14].

Ablaufmodell Die Entwicklung mit O-MaSE folgt keinem einheitlichen Schema. So ist es möglich O-MaSE in ein klassisches Wasserfallmodell oder auch in einen agilen Entwicklungsansatz zu integrieren. Die Flexibilität wird durch die Definition fakultativer und obligatorischer Tätigkeiten und Aufgaben erlangt. So ist eine projektspezifische Skalierung der methodischen Prozesstiefe möglich. Folgende Phasen werden durch unterschiedliche Tätigkeiten und zu erarbeitende Artefakte unterstützt:

- **Anforderungserhebung**
- **Lösungsfindung**
- **Architektur-Entwurf**
- **Detail-Entwurf**
- **Implementierung**

Rollenmodell Das in O-MaSE ausführlich definierte Rollenmodell umfasst acht „Produzenten“:

- **Ziel-Modellierer**: Der *Ziel-Modellierer* muss in der Lage sein die System-Beschreibung zu verstehen. Darüber hinaus muss er eine offene und klare Kommunikation mit den Domänenexperten bzw. Kunden pflegen, um das Zielmodell zu erstellen und weiter zu verfeinern.
- **Rollen-Modellierer**: Ziel dieser Person ist es, das Rollen-Modell sowie die Rollen-Beschreibungen zu erstellen. Hierfür muss er sowohl die System-Beschreibung kennen als auch mit dem Ziel- und Agenten-Modellierer eng zusammenarbeiten.
- **Agenten-Modellierer**: Ziel dieser Rolle ist es, das Agenten-Klassen-Modell zu erstellen. Grundlage dafür sind ausreichende UML-Kenntnisse.
- **Plan-Modellierer**: Der *Plan-Modellierer* definiert die Pläne eines Agenten um spezifische Ziele zu erreichen. Hierfür muss er in der Lage sein, Zustandsautomaten zu modellieren.
- **Fähigkeiten-Modellierer**: Die Person, welche diese Rolle ausfüllt, ist für die Erstellung des Fähigkeits-Modells verantwortlich. Grundlage hierfür sind ausreichende UML-Kenntnisse.

- **Protokoll-Modellierer**: Basierend auf grundlegenden Fähigkeiten im Umgang mit AUML ist der *Protokoll-Modellierer* für die Definition der unterschiedlichen Kommunikationsprotokolle zwischen Agenten, Rollen, dem Gesamtsystem und externen Akteuren verantwortlich.
- **Vorschriften-Modellierer**: Aufgabe des *Vorschriften-Modellierers* ist es, das aktuelle Vorschriften-Modell zu erstellen. Hierfür sind Kenntnisse der Prädikatenlogik von Nöten.
- **Tätigkeits-Modellierer**: Der *Tätigkeits-Modellierer* erstellt die Tätigkeits Diagramme der individuellen Agenten. Grundlage hierfür sind UML Kenntnisse sowie ein Grundlegendes Verständnis der Prädikatenlogik.

Artefaktmodell O-MaSE definiert die zu erstellenden Artefakte mit ihren Ein- und Ausgangsparametern sowie deren zugrundeliegenden Abhängigkeiten zu anderen Artefakten. In der ersten „Meta“-Phase *Initiierung* werden die folgenden Modelle und Arbeitsprodukte erstellt:

- **System-Beschreibung**: Beschreibt die technischen Anforderungen an ein Agentenbasiertes Software System
- **Ziel-Modell**: Beschreibt den Zweck der Organisation in Form eines Ziel-Baumes, welcher auch Attribute und Abhängigkeiten der Ziele beinhaltet.
- **Domänen-Modell**: Definiert die Syntax, mit welcher der Betrieb des Systems beschrieben werden kann.
- **Organisations-Modell**: Dokumentiert die Interaktionen zwischen dem Gesamtsystem und externen Akteuren.
- **Rollen-Modell**: Beschreibt System-Rollen sowie deren Ziele und Interaktionen zwischen individuellen Rollen und externen Akteuren

In der zweiten „Meta“-Phase *Design* werden dann die für die eigentliche Implementierung notwendigen Artefakte erarbeitet:

- **Agenten-Klassen-Modell**: Basierend auf dem Rollen-Modell und dem Organisations-Modell, definiert das Agenten-Klassen-Modell die individuellen Agenten Klassen sowie die Sub-Organisationen, aus welchen des Gesamtsystems aufgebaut ist.
- **Protokoll-Modelle**: Ebenfalls basierend auf dem Agenten-Klassen-Modell und dem Rollen-Modell der vorangegangenen Phase werden die dort auf abstrakter Ebene beschriebenen Protokolle nun in individuellen Protokoll-Modellen ausgearbeitet.
- **Vorschriften-Modell**: Beschreibt alle Regeln und Beschränkungen des Systems.
- **Fähigkeits-Modell**: Definiert die innere Struktur der Fähigkeiten einzelner Agenten.
- **Plan-Modell**: Dokumentiert die Pläne, welche das Verhalten der Agenten ausmachen. Dabei werden Aktionen spezifiziert, welche zur Erlangung eines spezifischen Ziels notwendig sind.

- **Tätigkeits-Modell**: Definiert die notwendigen Tätigkeiten zur Durchführung von Plänen oder der Erlangung von Zielen in hohem Detail.

Aktivitätsmodell Ähnlich dem in Gaia präsentierten Aktivitätsmodell werden in O-MaSE zwar inhaltliche, zeitliche und organisatorische Zusammenhänge zwischen den klar definierten Artefakten dargelegt, es werden jedoch auch in O-MaSE keine Vorgaben zum Projektmanagement gemacht. Die Vorgehensweisen, in welcher Form die Inhalte erarbeitet werden sollen, werden im Gegensatz zu Gaia in hohem Detail beschrieben [DGo07].

Werkzeugunterstützung Die Entwickler des O-MaSE Ansatzes stellen ein *Eclipse*-Plugin zur Verfügung, welches es ermöglicht, die oben beschriebenen Artefakte grafisch zu erarbeiten und hieraus die grundlegenden Code-Bausteine für *JADE* basierte Agenten automatisiert erstellen zu lassen. Der *aT3 Prozess Editor* ermöglicht die flexible Gestaltung von O-MaSE konformen Prozessen auf Basis enthaltener und eigen definierter O-MaSE Methoden Fragmente. Eine Konsistenzprüfung sowie ein Tool zur Überwachung des Projektfortschritts sind ebenfalls mit an Bord.

5.5. Evaluation der Vorgehensmodelle

Die Evaluation der Vorgehensmodelle basiert auf einheitlichen Kriterien, welche auf Basis einer Literaturrecherche identifiziert wurden.

5.5.1. Kriterien

Die im Folgenden definierten Modellkriterien dienen dem qualitativen Vergleich der oben beschriebenen Vorgehensmodellen.

Phasenabdeckung Im Rahmen der Softwareentwicklung werden die in 5.1 beschriebenen Phasen des Softwarelebenszyklus durchlaufen. Nach [Kal13] und [NS99] ist es wünschenswert, möglichst viele Phasen dieses Zyklus zu berücksichtigen. Dies ermöglicht eine durchgängige Strukturierung und eine einheitliche Syntax der verwendeten Modellelemente. Die Modellelemente der Vorgehensmodelle sollen daher auf ihre Zuordnung zu den vier Lebenszyklusphasen hin untersucht werden. Eine höhere Phasenabdeckung ist dabei positiv zu bewerten.

Kundenorientierung In Anlehnung an Fittkaus Begriff von Kundenorientierung [Von98] werden die folgenden vier Anforderungen definiert:

- **Kaizen**: Die Kundenzufriedenheit soll durch Maßnahmen der kontinuierlichen Verbesserung erhöht werden.
- **Qualität**: Es muss ein kundenzentrierter Qualitätsbegriff definiert sein.
- **Integration**: Der Prozess muss die Integration des Kunden vorsehen.
- **Qualifizierung**: Der Kunde soll durch den Entwicklungsprozess zusätzlich qualifiziert werden.

Prozesssteuerung Eine Differenzierung der Vorgehensmodelle ist auf Basis von Modell- und Phasenzusammenhängen möglich. Noack und Schienmann [NS99] unterscheiden dafür ergebnis- oder aktivitätsorientierte Ansätze:

- **Ergebnisorientierung**: Das Vorgehensmodell macht Vorgaben zu den Ergebnissen einer Phase, jedoch nicht zu den dafür notwendigen Aktivitäten.
- **Aktivitätsorientierung**: Das Modell definiert die durchzuführenden Aktivitäten, jedoch nicht die zu erzielenden Ergebnisse. Durch eine detaillierte Aktivitätsbeschreibung wird die Erlangung des korrekten Ergebnisses sichergestellt.

Bei hybriden Prozesssteuerungen wird im Zuge der Analyse die maßgebliche Form der Steuerung bestimmt.

Detaillierungsgrad Der Detaillierungsgrad eines Vorgehensmodells lässt sich aus dem Umfang des Aktivitäts-, Rollen- und Artefaktmodells ableiten. In Anlehnung an [LKB+10] und [Luk15] werden hierfür die Anzahl der Elemente sowie der Detaillierungsgrad der Vorgehensmodelle analysiert und wie folgt klassifiziert:

- **(-)**: Untersuchungsobjekt in Vorgehensmodell nicht beschrieben.
- **(0)**: Geringer Detaillierungsgrad des Untersuchungsobjekts.
- **(+)**: Hoher Detaillierungsgrad des Untersuchungsobjekts.

Anpassbarkeit Anpassbarkeit beschreibt die beabsichtigte Möglichkeit des *Tailorings* eines Vorgehensmodells. Kalus [Kal13] nutzt diesen Faktor zur Evaluierung und Chroust [Chr92] liefert mit seiner Theorie der Modellunterscheidung ein Mittel zur Quantifizierung:

- **(-)**: Eine Anpassbarkeit ist aufgrund der Enge der Prozessvorgaben nur in geringem Umfang möglich. Die sogenannten *Dach-Modelle* definieren alle Prozessbausteine sehr detailliert und erfordern für eine Anpassung die Außerachtlassung einzelner Elemente.
- **(0)**: Modelle, welche nur geringe Vorgaben zum Prozess machen, lassen sich unkompliziert an individuelle Projekte anpassen. Hierfür sind umfangreiche Ergänzungen zum *Kern-Modell* möglich, ohne dabei die Vorgaben des Modells zu verletzen.
- **(+)**: Vorgehensmodelle, welche in *Modulare Bausteine* gegliedert sind, erlauben eine projektspezifische Anpassung unter Auswahl der Verfügbaren Submodelle. Da diesen Modellen der Gedanke des *Tailorings* zugrundeliegt, bieten sie die höchste Anpassbarkeit.

Gesetzliche Anforderungen In Hinblick auf eine Nutzung der Vorgehensmodelle im Kontext der Entwicklung kritischer Infrastrukturen ist die Berücksichtigung gesetzlicher Anforderungen und spezifischer Vorschriften von hoher Relevanz. Es wird daher im Folgenden bewertet, ob diese im Zuge des Prozesses Berücksichtigung finden können und ob eine regelmäßige Überprüfung möglich ist. Die folgenden Bewertungsstufen werden dabei differenziert:

- **(-)**: Gesetzliche Anforderungen lassen sich nicht berücksichtigen.
- **(0)**: Eine Berücksichtigung ist möglich, jedoch keine regelmäßige Überprüfung.
- **(+)**: Gesetzliche Anforderungen lassen sich im Prozess berücksichtigen und regelmäßig überprüfen.

5.5.2. Analyse

Im Folgenden werden die eingangs beschriebenen Entwicklungsansätze auf Basis der in Abschnitt 5.5.1 beschriebenen Kriterien bewertet.

Phasenabdeckung

Scrum Scrum definiert die in Tabelle 5.2 dargestellten Bausteine, welche alle Projektphasen von der Anforderungsfestlegung bis hin zur Integration abdecken. Diese Bausteine werden in einem zyklischen Vorgehen stets wiederholt und so das Produkt in Inkrementen verfeinert. Am Ende eines jeden Inkrements steht die *Sprint Retrospektive*, welche zwar keinen Beitrag zum Produktinkrement liefert, jedoch der Prozessoptimierung Qualitätssicherung dient.

V-Modell XT Das V-Modell XT stellt *Vorgehensbausteine* zur Verfügung, welche eine projektspezifische Konfiguration des Entwicklungsprozesses ermöglichen. Das in Tabelle 5.2 dargestellte Portfolio der *Vorgehensbausteine* umfasst Aktivitäten für alle Phasen des Softwarelebenszyklus, außer der In- und Außerbetriebnahme. Die im V-Modell XT ebenfalls beschriebenen *Vorgehensbausteine* zur Qualitätssicherung, dem Projekt-, Konfigurations-, Problem- und Änderungsmanagement ergänzen die dargestellten Komponenten in hohem Detail um weitere Aktivitäten, Rollen und Artefakte.

Extreme Programming Ähnlich des Scrum Ansatzes beschreibt Extreme Programming einen iterativen Entwicklungsprozess, welcher jedoch ein umfangreicheres Repertoire und somit eine höhere Phasenabdeckung bietet. Außer der Phase „Erprobung und Inbetriebnahme“bildet das Aktivitätsmodell, wie in Tabelle 5.2 ersichtlich, alle relevanten Tätigkeiten im Projektverlauf ab.

Gaia Gaia deckt lediglich die beiden ersten Projektphasen „Anforderungsfestlegung“und „Grob- und Feinentwurf“ab. Es werden keine Vorgaben zur Implementierung und allen folgenden Projektphasen gegeben, da hier traditionelle Entwicklungsmethoden zum Einsatz kommen sollen.

Tropos Tropos legt einen starken Fokus auf die frühen Phasen des Entwicklungsprozesses und beschreibt diese bis hin zur Integration der erstellten Software. Ein besonderes Augenmerk wird dabei auf die Abhängigkeiten der einzelnen Prozessschritte und der zu erarbeitenden Artefakte gelegt. Tabelle 5.2 setzt die Tropos spezifische Nomenklatur in Bezug zu den eingangs definierten, generischen Projektphasen.

O-MaSE O-MaSE wurde in Hinblick auf ein möglichst hohes Maß an Flexibilität entwickelt und folgt daher keinem einheitlichen Schema. Die in O-MaSE beschriebenen Modelle können jedoch auf die ersten drei Phasen des Projektlebenszyklusses projiziert werden. Die in O-MaSE diskutierten Projektphasen werden in der folgenden Tabelle 5.2 dargestellt.

Lebenszyklusphase	**Scrum**	**V-Modell XT**	**Extreme Programming**	**Gaia**	**Tropos**	**O-MaSE**
Anforderungsfestlegung	Backlog-Management	Anforderungsfestlegung	Planning Game	Analysephase	Initiale- und Weitere Anforderungserhebung	Anforderungserhebung
Grob- und Feinentwurf	Sprint Planning Daily Scrum	SW-Entwicklung	Release- und Iteration Planning	Entwurfsphase	Architektur- und Detail-Entwurf	Lösungsfindung Architektur- und Detail-Entwurf
Implementierung	Entwicklung	SW-Entwicklung	Development	-	Implementierung und Tests	Implementierung
Test und Integration	Sprint Review	SW-Entwicklung, Systemerstellung	Development Acceptance Tests	-	Implementierung und Tests	-
Erprobung und Inbetriebnahme	-	-	-	-	-	-
Wartung und Weiterentwicklung	-	Weiterentwicklung und Migration	Gesamter Prozess	-	-	-
Außerbetriebnahme	-	-	Abschlussdokumentation	-	-	-

Tabelle 5.2.: Übersicht der Phasenabdeckung

Kundenorientierung

Scrum Der grundlegenden Annahme folgend, dass ein Produkt zu Projektbeginn nicht vollständig definiert werden kann, werden die Anforderungen des Kunden im gesamten Projektverlauf ergänzt und detailliert. Basis hierfür ist das initial mit den wichtigsten Anforderungen befüllten *Product Backlog*. Aus diesem dynamischen Dokument werden für jeden

Sprint durch die Programmierer Aufgaben abgeleitet, im *Sprint Backlog* verschriftlicht und umgesetzt. Diese Einbindung der Fachleute sorgt für eine hohe Termintreue der einzelnen Sprints. Am Ende eines Sprints bewertet der Kunde im Zuge des *Sprint Reviews* die erstellte Software und wird somit regelmäßig in den Prozess integriert.

V-Modell XT Aufgrund seines Ursprungs in der Beschaffung der öffentlichen Hand, geht das V-Modell von einer vollständigen Definition der Kundenanforderungen zu Projektbeginn aus. Mittels *Tailoring* kann der Entwicklungsprozess dann an diese Spezifikation angepasst und das gewünschte Vorgehen modelliert werden. Die sequentiellen Phasen erlauben dem Kunden die Projektevaluation auf Basis klassischer Projektplanung. Da Entwicklungsprojekte jedoch meist den geplanten Pfad verlassen, erlaubt das V-Modell XT die Erweiterung des Vorgehensmodells um auch inkrementelle, komponentenbasierte und prototypische Ansätze abbilden zu können. Der Projektfortschritt wird dem Kunden in diesen Varianten jedoch weiterhin nur über Statusberichte kommuniziert.

Extreme Programming Eines der Ziele von Extreme Programming ist es, langfristige Kundenzufriedenheit zu erzeugen. So können Kunden bereits in der Entwicklungsphase Qualifikationen für den Betrieb des Produktes erwerben und durch eine tiefe Integration in den Prozess eine hohe Identifikation mit dem Produkt gewinnen. Die Prozessgestaltung entspricht Fittkau's Anforderungen an einen kundenorientierten Prozess [Von98].
So lassen sich auch in der Kundenorientierung Parallelen zwischen Extreme Programming und Scrum erkennen: Anforderungen werden durch den tief in den Entwicklungsprozess integrierten Kunden erhoben und das Produkt in kurzen Iterationen entwickelt. Eine kontinuierliche Integration und darauf aufbauende Tests werden durch eine Kooperation von Kunden und Entwicklern sicher gestellt. Die Bewertung der Qualität erfolgt somit stets aus der Kundenperspektive.

Gaia Gaia macht keine Vorgaben zur Interaktion der am Entwicklungsprozess beteiligten Stakeholder.

Scrum	V-Modell XT	Extreme Programming	Gaia	Tropos	O-MaSE
Kundenzufriedenheit im Prozess, kundenorientierter Qualitätsbegriff, regelmäßige Integration des Kunden	Tailoring (kurzfristig kundenorientiert), kundenorientierter Qualitätsbegriff	Kundenzufriedenheit im Prozess, kundenorientierter Qualitätsbegriff, kontinuierliche Integration des Kunden, Zusatzqualifikation des Kunden	-	Modellierung des Nutzers und seiner Ziele	-

Tabelle 5.3.: Übersicht der Kundenorientierung

Tropos Tropos fordert die dedizierte Modellierung der Nutzer und ihrer Ziele. Wenn auch Kunden im Kontext der Rollendefinition nicht explizit Erwähnung finden, so gibt es mehrere Schnittstellen im Rahmen der umfangreichen Anforderungsanalyse. Aufgrund seiner Anlehnung an *i** werden dem Kunden die Hintergründe der getroffenen Designentscheidungen transparent dargelegt.

O-MaSE Zwar werden in O-MaSE Schnittstellen zum Kunden beschrieben, diese dienen aber nicht der Qualitätssicherung, sondern lediglich der Anforderungserhebung und Zielmodellierung. Es ist somit keine explizite Kundenorientierung erkennbar.

Prozesssteuerung

Scrum Zu Beginn eines Scrum Projektes wird eine *Definition of Done* als Ergebnis des Projektes und gewünschten Endzustand der Software festgelegt. Die zu erzielenden Zwischenprodukte sowie der zeitliche Rahmen der hierfür zu absolvierenden Projektphasen sind inhaltlich abstrakt und zeitlich konkret beschrieben. Methoden und Aktivitäten können frei gewählt werden und unterliegen keinen Beschränkungen.

V-Modell XT Die aktivitätsorientierte Prozesssteuerung des V-Modell XT beschreibt den gewünschten Projektablauf in hohem Detail. Ergebnisse und deren zugrundeliegenden Aktivitäten werden gemeinsam mit den relevanten Verantwortlichkeiten und Rollen ausführlich dokumentiert.

Extreme Programming Zwar sollen die in Extreme Programming beschriebenen Methoden nur einen Handlungsrahmen für die Entwicklung der Artefakte geben und Freiraum zur Erstellung von Tests bieten, dennoch fokussiert das Aktivitätsmodell auf spezifische Methoden wie zum Beispiel *Pair Programming* oder *Planning Games*. Daher wird die Prozesssteuerung als aktivitätsorientiert bewertet.

Gaia Gaia erlaubt eine freie Wahl der Methoden zur Erarbeitung der klar definierten Artefakte. So werden zwar inhaltliche, zeitliche und organisatorische Zusammenhänge der Artefakte dargestellt, jedoch nicht in einen Projektrahmen eingebettet. Die Prozesssteuerung wird somit als ergebnisorientiert bewertet.

Tropos Die Prozesssteuerung des Tropos Ansatzes kann, aufgrund seines Charakters als modellbasierter Ansatz als Ergebnisorientiert beschrieben werden. Ein ausführliches Artefaktmodell beschreibt alle relevanten Rahmenbedingungen.

O-MaSE Ähnlich den beiden vorgenannten Ansätzen, liegt der Fokus von O-MaSE auf dem Produkt und den dafür notwendigen Informationen. Diese werden in einem ausführlichen Artefaktmodell dargelegt. Das zugrunde liegende Aktivitätsmodell und Fragen des Projektmanagements geben dem Anwender großen Gestaltungsspielraum. Es handelt sich somit um eine ergebnisorientierte Prozesssteuerung.

Scrum	V-Modell XT	Extreme Programming	Gaia	Tropos	O-MaSE
Ergebnis-orientiert	Aktivitäts-orientiert	Aktivitäts-orientiert	Ergebnis-orientiert	Ergebnis-orientiert	Ergebnis-orientiert

Tabelle 5.4.: Übersicht der Prozesssteuerung

Detaillierungsgrad

Scrum Scrum beschreibt ein Aktivitätsmodell lediglich implizit durch Rollen- und Ablaufbeschreibungen einzelner Projektphasen. Es wird daher mit - bewertet.
Das Rollenmodell von Scrum umfasst lediglich drei Rollen, für welche jedoch klare Zuständigkeiten und Handlungsvorschriften definiert sind. Das Rollenmodell wird daher mit *0* bewertet.
Auch das Artefaktmodell ist mit vier Modellen weniger umfangreich, beschreibt dafür aber die Inhalte und spezifische Eigenschaften der Artefakte. Die Form der Artefakte kann frei gewählt werden. Das Artefaktmodell wird mit *0* bewertet.

V-Modell XT Das äußerst umfangreiche Aktivitätsmodell des V-Modell XT bietet für eine Vielzahl der 102 beschriebenen Aktivitäten individuelle Arbeitsablaufpläne. Die zu erstellenden Artefakte sowie die beteiligten Rollen werden aufgezeigt und vervollständigen die Aktivitätsbeschreibungen. Das Aktivitätsmodell wird daher mit + bewertet.
Jeder der 35 Rollen des V-Modell XT wird ausführlich beschrieben und umfasst neben allen Relevanten Faktoren wie Aufgaben und Befugnissen auch Hinweise zur Rollenbesetzung. Es wird somit mit + bewertet.
Das V-Modell XT bietet für jede der 110 spezifizierten Artefakte eine umfangreiche Beschreibung. Es wird mit + bewertet.

Extreme Programming Das Aktivitätsmodell des Extreme Programming Ansatzes wird anhand der zu erstellenden Artefakte und der beteiligten Rollen definiert. Alle zwölf Aktivitäten werden beispielhaft beschrieben folgen einem Ablaufschema. Aufgrund der begründeten Auswahl und der intensiven Diskussion der beschrieben Aktivitäten wird das Aktivitätsmodell mit + bewertet.
Das ebenfalls ausführlich diskutierte Rollenmodell beschreibt sieben Rollen, welche sowohl die Arbeitsweisen, Fähigkeitsprofile und Leitlinien umfassen. Es wird ebenfalls mit + bewertet.
Extreme Programming bietet kein explizites Artefaktmodell. Artefakte müssen aus den Aktivitätsbeschreibungen abgeleitet werden. Es wird daher mit *0* bewertet.

Gaia Gaia definiert die Aktivitäten zu Erstellung von Artefakten nur oberflächlich. Ebenso werden die Abhängigkeiten der Aktivitäten untereinander nicht beschrieben. Das Aktivitätsmodell wird somit mit - bewertet. In Ermangelung eines Rollenmodells in Gaia, wird es mit - bewertet.
Das äußerst ausführliche Artefaktmodell in Gaia ist an die Bedürfnisse einer Agentenorientierten Entwicklung angepasst. Es bietet die Möglichkeit mittels fünf Modellen alle Aspekte eines Agentensystems abzubilden. Einschränkungen wie eine auf bis zu 100 Agenten limi-

Modell-typ	Scrum	V-Modell XT	Extreme Programming	Gaia	Tropos	O-MaSE
Aktivität	-	+	+	-	+	0
Rollen	0	+	+	-	+	+
Artefakt	0	+	0	0	+	+

Tabelle 5.5.: Übersicht des Detaillierungsgrades

tierte Größe des Systems oder die lediglich statische Natur reduzieren die Bewertung des Artefaktmodells jedoch auf *0*.

Tropos Alle Arbeitsschritte von Tropos werden im Aktivitätsmodell ausführlich beschrieben. Rahmenbedingungen wie Ein- und Ausgangsparameter und Verantwortlichkeiten werden definiert. Das Aktivitätsmodell wird in Folge dessen mit + bewertet. Tropos beschreibt in seinem Rollenmodell sieben Rollen. Diese werden ausführlich erörtert und in Bezug zu den einzelnen Aktivitäten und Artefakten gesetzt. Es erfolgt somit eine Bewertung mit dem Attribut +. Aufgrund seines Charakters als modellbasierter Engineering Ansatz und den Entlehnungen von *i** besitzt Tropos ein sehr ausführliches Artefaktmodell, welches mit + bewertet wird.

O-MaSE Das mit *0* zu bewertende Aktivitätsmodell von O-MaSE beschreibt zwar alle relevanten Artefakt- und Prozess-Zusammenhänge, lässt jedoch Aspekte des Projektmanagements aus.
Das detaillierte Rollenmodell definiert acht Rollen und wird mit + bewertet.
Alle relevanten Parameter und Abhängigkeiten der individuellen Artefakte werden in O-MaSE ausführlich dargelegt. Die elf Modelle beschreiben das zu implementierende System umfassend. Das Artefaktmodell wird daher mit + bewertet.

Anpassbarkeit

Scrum Scrum ist ein Kern-Modell nach Chroust [Chr92]. Aufgrund seiner evolutionären Ablaufstruktur und der freien Definierbarkeit von Aktivitäten ist ein hohes Maß an *Tailoring* möglich. Scrum ist somit sehr gut auf die Bedürfnisse individueller Projekte anpassbar und wird mit einer Adaptierbarkeit von *0* bewertet.

V-Modell XT Das V-Modell XT ist ein Bausteinmodell nach Chroust [Chr92] und bietet daher vielfältige Möglichkeiten des *Tailorings*. Es können unterschiedliche Entwicklungsstrategien abgebildet und die definierten *Vorgehensbausteine* in weitem Umfang frei kombiniert werden.

Extreme Programming Entgegen der Flexibilität anderer evolutionärer Entwicklungsansätze ist Extreme Programming nur beschränkt anpassbar. Die in Extreme Programming beschriebenen Aktivitäten bauen aufeinander auf und sind in beide Richtungen voneinander abhängig. Extreme Programming ist nach der Definition von Chroust [Chr92] ein Dach-Modell und wird in seiner Anpassbarkeit mit - bewertet.

Scrum	V-Modell XT	Extreme Programming	Gaia	Tropos	O-MaSE
0	+	-	0	-	+

Tabelle 5.6.: Übersicht der Anpassbarkeit

Gaia Da Gaia nur wenige Vorgaben zum eigentlichen Entwicklungsprozess macht und seinen Fokus auf die zu erarbeitenden Artefakte legt, ist es nach Chroust [Chr92] als Kern-Modell zu kategorisieren und wird somit mit *0* bewertet.

Tropos Aufgrund der hohen Interdependenzen der Aktivitäten und Artefakte bietet Tropos keine Möglichkeiten der Anpassung. Es ist ein Dach-Modell nach Chroust [Chr92] und wird folglich mit - bewertet.

O-MaSE O-MaSE ist von Grund auf als Baukasten unterschiedlicher Methoden-Fragmente entwickelt worden. Es bietet die Möglichkeit mittels *Tailoring* individuelle Entwicklungsprozesse zu gestalten. O-MaSE ist somit nach Chroust [Chr92] ein Bausteinmodell und wird mit + bewertet.

Gesetzliche Anforderungen

Scrum Aufgrund der guten Integration des Kunden in den Entwicklungsprozess, können gesetzliche Anforderungen aufgenommen und relevante Faktoren an sich ändernde Rahmenbedingungen angepasst werden. Die Berücksichtigung wird somit mit + bewertet.

V-Modell XT Das V-Modell XT verfügt über einen kundenorientierten Qualitätsbegriff und eine extensive Anforderungserhebung, in welcher regulatorische Aspekte Berücksichtigung finden sollen. Die Integration des Faktors wird daher mit + bewertet.

Extreme Programming Alle in Extreme Programming erarbeiteten Produktiterationen werden durch den Kunden verifiziert. Ähnlich des Scrum Ansatzes lassen sich gesetzliche Anforderungen somit kontinuierlich berücksichtigen und anpassen, was zu einer Bewertung von + führt.

Gaia In Ermangelung eines expliziten Qualitätsmanagements und der fehlenden Integration des Kunden in den Entwicklungsprozess, lassen sich gesetzliche Anforderungen nur im Rahmen der initialen Anforderungserhebung berücksichtigen. Eine Überprüfung dieses Faktors ist im weiteren Laufe des Prozesses nur schwer möglich. Die Berücksichtigung wird somit mit - bewertet.

Tropos Auch Tropos erlaubt die Berücksichtigung der Anforderungen nur in einer frühen Phase des Prozesses. Da jedoch eine ausführliche Erläuterung der daraus abgeleiteten Designentscheidungen vorgesehen ist, lässt sich die Berücksichtigung von eingebrachten gesetzlichen Anforderungen verifizieren. Es erfolgt daher eine Bewertung mit *0*.

Scrum	V-Modell XT	Extreme Programming	Gaia	Tropos	O-MaSE
+	+	+	-	0	-

Tabelle 5.7.: Übersicht der Berücksichtigung gesetzlicher Anforderungen

O-MaSE Zwar lassen sich in O-MaSE gesetzliche Anforderungen in der initialen Anforderungserhebung integrieren, jedoch beschreibt der Ansatz keine expliziten Möglichkeiten der Qualitätssicherung. Somit kann nicht sicher gestellt werden, dass diese auch erfüllt werden. Die Berücksichtigung wird in Folge dessen mit - bewertet.

5.6. Zwischenfazit

In diesem Kapitel wurden auf Basis der initial eingeführten abstrakten Ansätze im Bereich des Software Engineerings Kriterien zur Evaluation von Entwicklungsprozessen definiert. Es wurden konkrete Methoden dargestellt und diese anschließend bewertet.

Eine Zusammenfassung der Ergebnisse ist in Tabelle 5.8 dargestellt. Es ist erkennbar, dass

Faktor	Scrum	V-Modell XT	Extreme Programming	Gaia	Tropos	O-MaSE
Phasenabdeckung	4/7	5/7	6/7	2/7	4/7	3/7
Kundenorientierung	++	+	++	-	0	-
Prozesssteuerung	Ergebnis	Aktivität	Aktivität	Ergebnis	Ergebnis	Ergebnis
Detaillierungsgrad	-	++	+	–	++	+
Anpassbarkeit	0	+	-	0	-	+
Gesetzliche Anforderungen	+	+	+	-	0	-

Tabelle 5.8.: Übersicht der Entwicklungsprozesse

individuelle Vorgehensmodelle unterschiedliche Stärken und Schwächen aufweisen und aufgrund ihrer ungleichen Anwendungsszenarien kein Vorgehensmodell alle im Rahmen des Engineerings von Energiesystemen notwendigen Anforderungen abdecken kann. So bietet das Extreme Programming beispielsweise eine Phasenabdeckung von sechs der möglichen sieben Phasen. Gaia hingegen kann lediglich in zwei der sieben Phasen Anwendung finden. Die Kundenorientierung, welche bei einer langfristigen Kundenorientierung mit ++, einer kurzfristigen mit +, einer oberflächlichen mit 0 und einer fehlenden Kundenorientierung mit - bewertet wurde, ist ebenfalls äußerst unterschiedlich ausgeprägt. Ein Blick auf die Prozesssteuerung

zeigt, dass lediglich das V-Modell XP sowie Extreme Programming einen aktivitätsorientierten Ansatz verfolgen, wohingegen alle anderen Methoden eine, grundsätzlich nicht schlechtere, Ergebnisorientierung zugrunde liegt. Der Detaillierungsgrad, welcher im Falle einer ausführlichen Abdeckung aller drei Modelle mit ++ bewertet wurde, im Falle einer ausführlichen Abdeckung von zwei Modellen mit +, der geringfügigen Abdeckung zweier Modelle mit - und der geringfügigen Abdeckung nur eines Modells mit – bewertet wurde, zeigt eine Abdeckung des kompletten Spektrums durch die präsentierten Ansätze. Auch die in Abschnitt 5.5.2 bewertete Anpassbarkeit zeigt diese Streuung. Lediglich die Berücksichtigung der gesetzlichen Anforderungen lässt eine klare Dominanz der generischen Entwicklungsprozesse erkennen.
Trotz der unterschiedlichen Stärken- und Schwächen der untersuchten Ansätze lässt sich feststellen, dass keiner die Anforderungen an ein kundenorientierten Prozess, welcher es erlaubt sowohl die regulatorischen und technischen Rahmenbedingungen der Energietechnik als auch die Spezifika agentenorientierter Softwareentwicklung zu berücksichtigen, erfüllt.

6. Engineering agentenbasierter Automatisierungstechnik für Verteilnetze

Der im Folgenden dargestellte Entwicklungsansatz für verteilte Steuerungssysteme wird seinem englischen Namen *Development approach for DEcentralised Control Systems* folgend als *2DECS* abgekürzt. Es handelt sich bei 2DECS um eine Synthese der in Kapitel 5 dargestellten Entwicklungsansätze, mit dem Ziel, die Entwicklung der in Kapitel 3 beschriebenen Agenten in der Netzautomatisierung im Hinblick auf die in Kapitel 2 erörterten Randbedingungen zu erleichtern. Der in Kapitel 4 eingeführte Energie-Agent soll dabei als gedankliches Konstrukt zur Strukturierung der Software-Fragmente dienen. Darüber hinaus kann das im Energie-Agenten enthaltene *Energie- & Optionsmodell* (siehe Abschnitt 4.2) als generisches Datenmodell zur Abbildung von Energieumwandlungs, -speicher und -transport-Prozessen genutzt werden.
Das in Abschnitt 5.2.2 beschriebene Metamodell eines Entwicklungsprozesses mit der klaren Trennung der Strukturelemente *Artefakt*, *Phase*, *Aktivität*, *Rolle* und *Werkzeug* wird auch im Zuge der systematischen Beschreibung des 2DECS-Prozesses herangezogen. Da es sich bei 2DECS um einen zielorientierten Prozess handelt, stellt das Artefaktmodell die Grundlage des weiteren Prozesses dar. Daher werden zunächst die zu erarbeitenden Artefakte beschrieben.

6.1. Artefakte

Das Artefaktmodell des 2DECS-Prozesses besteht aus zwei Komponenten: einem organisatorischen- und einem technischen Artefaktmodell. Das Erste umfasst alle Artefakte, welche für die Planung und Überwachung des Prozesses benötigt werden. Das Zweite beschreibt alle Artefakte, welche die Grundlage zur Erstellung der eigentlichen Software bilden. Diese Trennung wird etabliert, da hierdurch die Erarbeitung von Inhalten bzw. der Software vom sonstigen Projektgeschäft getrennt werden kann, was den beteiligten Akteuren eine bessere Aufgabentrennung und damit Spezialisierung ermöglicht. Darüber hinaus lässt sich eine Zweiteilung existierender Entwicklungsansätze in der Literatur erkennen. So beschreiben generische Vorgehensmodelle wie XP (siehe: 5.3.3), SCRUM (siehe: 5.3.2) oder das V-Modell XT (siehe: 5.3.1) Lösungsansätze für Aspekte der Projektorganisation, während sich agentenspezifische Methoden wie GAIA (siehe: 5.4.1), TROPOS (siehe: 5.4.2) oder O-MaSE (siehe: 5.4.3) auf die Erarbeitung softwarespezifischer Artefakte fokussieren.

6.1.1. Organisatorische Artefakte

Abbildung 6.1 zeigt die Elemente des organisatorischen Artefaktmodells im Vergleich zu denen der Vorgehensmodelle XP und SCRUM. Das V-Modell XT wurde bewusst nicht in

den Vergleich aufgenommen, da das Artefaktmodell des V-Modell XT so umfassend ist, dass es nur mittels eines eigens dafür entwickelten Konfigurationstools an die Bedürfnisse des individuellen Projektes angepasst und auf ein beherrschbares Maß reduziert werden kann. Ein generischer Vergleich ist somit nicht möglich.

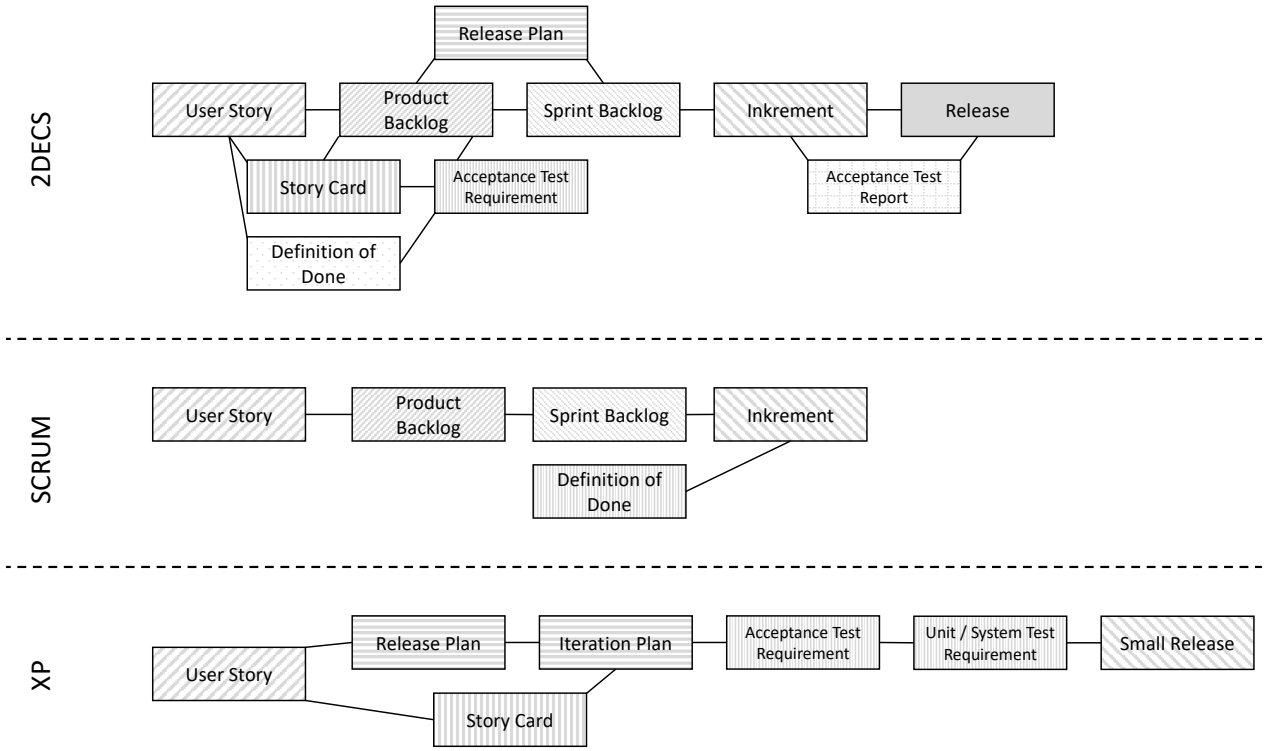

Abbildung 6.1.: Vergleich der organisatorischen Artefaktmodelle von XP, SCRUM und 2DECS

Die Abhängigkeiten der Elemente untereinander werden in der Grafik durch ihre Anordnung auf der Horizontalen abgebildet. Linien zeigen eine Abhängigkeit des rechten (Ziel-)Elements vom linken (Start-)Element. So bildet zum Beispiel die *User Story* die Grundlage für das *Product Backlog*, die *Story Card* und die *Definition of Done*. Ähnlichkeiten der Artefakte zwischen den Ansätzen werden durch eine gleichartige Schraffur angedeutet. So ist beispielsweise ein *Inkrement* der Prozesse 2DECS und und SCRUM mit einem *Small Release* in XP vergleichbar. Die Abhängigkeiten, Ähnlichkeiten und Inhalte der Elemente sollen im Folgenden beschrieben werden:

User Story

Eine *User Story* ist in 2DECS, wie auch in XP, SCRUM und anderen agilen Entwicklungsprozessen, der Ausgangspunkt aller weiteren Aktivitäten. In *User Stories* werden die Anforderungen an das Produkt in informeller Weise gesammelt. Der *Kunde* und idealerweise auch die zukünftigen Nutzer des Produktes beschreiben aus ihrer Perspektive, welche Probleme sie mit dem Produkt lösen beziehungsweise welche Ziele sie erreichen möchten. Dies geschieht in der Regel in der Ich-Form. Die *User Stories* werden in einer Liste gesammelt und ggf. schon für einzelne Funktionsbereiche des Produktes vorsortiert. Diese Sortierung ist abhängig von den Zielen und Umfängen des Produktes und sollte sich an den Hauptkomponenten

Kategorie	Nutzer Typ	User Story
GUI	Anwender	Als Nutzer möchte ich die erzeugte Leistung meines Solarpaneels sehen.
GUI	Administrator	Als Administrator möchte ich eine Übersicht aller registrierten Anlagen sehen.
Interface	Anwender	Als Nutzer möchte ich meinen Solarakku an meinen PC anschließen.
Hardware	Anwender	Als Nutzer möchte ich am Solarakku den Ladestand erkennen können.

Tabelle 6.1.: Auszug der Sammlung an *User Stories* für das OEEX Mobile Battery Projekt

des Produktes, wie dem Graphical-User-Interface (GUI), dem Steuerungsalgorithmus, der Datenverwaltung oder etwaigen Hard- und Software-Schnittstellen, orientieren. Eine alternative Sortierung wäre das Zusammenfassen unterschiedlicher Funktionen wie zum Beispiel der Nutzerverwaltung, der Verwaltung des angeschlossenen Equipments, der Überwachung oder der eigentlichen Steuerung. Tabelle 6.1 zeigt eine beispielhafte Sammlung von *User Stories* für das in Abschnitt 7.1.1 beschriebene *Mobile Power* Projekt der Open Energy EXchange (OEEX) Plattform.

Definition of Done

Parallel zur Sammlung der *User Stories* wird eine *Definition of Done* für das Gesamtprojekt zwischen dem *Kunden* und den Projektverantwortlichen verhandelt. Im Gegensatz zu SCRUM umfasst diese *Definition of Done* das gesamte Softwareprojekt und orientiert sich nicht an den individuellen *User Stories*. Hierbei müssen überprüfbare Projektziele definiert werden. Idealerweise wird in diesem Zuge auch vereinbart, wie die Überprüfung der Ziele stattfindet, so dass beide Seiten am Ende des Projektes diese Überprüfung durchführen können. Die *Definition of Done* ist in 2DECS eine Rückfalloption für die Aufrechterhaltung bzw. Beendigung des Projektes von Kunden- wie auch von Herstellerseite. Ist das gesamte *Product Backlog* (siehe unten) abgearbeitet und weiterhin das in der *Definition of Done* festgehaltene Projektziel nicht erreicht, müssen weitere *User Stories* gesammelt und weitere Iterationen des Prozesses durchgeführt werden. Im entgegengesetzten Fall, dass sich noch *Story Cards* (siehe unten) im *Product Backlog* befinden, die Projektziele jedoch bereits erreicht wurden, kann das Projekt beendet werden.
Die *Definition of Done* wird, ähnlich einem traditionellen Lastenheft, in Textform festgehalten und fokussiert dabei auf die übergeordneten Ziele des Produktes.

Story Card

Sind die *User Stories* initial formuliert, werden diese, in Anlehnung an XP, in sogenannte *Story Cards* transformiert. Im Rahmen der Transformation werden die informellen Anforderungen der *User Stories* in eine technisch eindeutigere Syntax übersetzt und um eine Bewertung des Umsetzungsaufwandes sowie der damit verbundenen technischen und Kostenrisiken ergänzt. Die so gesammelten Informationen werden in eigenen Datensätzen festgehalten. Dies kann entweder in Form individueller Papierkarten oder entsprechender Software geschehen. Wichtig ist dabei, dass diese zu einem späteren Zeitpunkt zueinander in Beziehung gesetzt

und um weitere Informationen ergänzt werden können.
Empfehlenswert ist darüber hinaus die Bewertung des Aufwandes und der Risiken auf einem einfachen Niveau mit einer niedrigen Skala (zum Beispiel eins bis fünf) durchzuführen. Eine detailliertere Abschätzung ist in der Regel nicht zielführend, da sie zum einen sehr aufwändig und zum anderen zu diesem frühen Zeitpunkt im Projekt nicht ausreichend fundiert ist. Abbildung 6.2 zeigt ein Beispiel, wie eine *Story Card* strukturiert sein kann. In dieser wird ein mittlerer Umsetzungsaufwand, ein geringes technisches- sowie ein vernachlässigbares finanzielles Risiko beschrieben. Andere Darstellungsformen, in Abhängigkeit der genutzten Materialien oder Software sowie der Ansprüche des Entwickler-Teams, sind entsprechend umsetzbar.

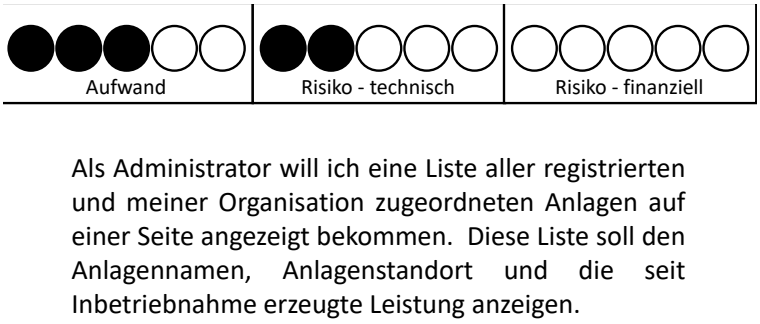

Abbildung 6.2.: Beispiel einer Storycard in 2DECS

Acceptance Test Requirement

Ähnlich der *Definition of Done* in SCRUM oder den *Acceptance Test Requirements* in XP existiert in 2DECS für jede *Story Card* mindestens ein *Acceptance Test Requirement*. Die Definition mehrerer *Acceptance Test Requirements* für eine *Story Card* ist möglich. Ein *Acceptance Test Requirement* kann für mehrere *Story Cards* genutzt werden, wird dann jedoch individuell festgehalten um die Abarbeitung der einzelnen *Story Cards* zu ermöglichen. Die Definition muss eindeutig überprüfbar und mit einem Ja oder Nein zu beantwortbar sein. Im Rahmen automatisierter Softwaretests bilden *Acceptance Test Requirements* so die Basis des Testprocederes.

Nr.	Acceptance Test Requirement
1	User in der Rolle Administrator sehen eine Liste von Anlagen.
2	Diese Liste umfasst nur registrierte Anlagen ihrer Organisation.
3	Die Liste ist vollständig.
4	Es wird der Anlagenname angezeigt.
6	Es wird der Anlagestandort angezeigt.
7	Es wird die seit der Inbetriebnahme erzeugte Leistung angezeigt.

Tabelle 6.2.: Beispiel von Acceptance Test Requirements

Werden mehrere *Acceptance Test Requirements* für eine *Story Card* definiert, werden diese idealerweise in Tabellenform festgehalten. Werden die *Story Cards* in Papierform erstellt, so

kann diese auf die Rückseite der *Story Card* notiert werden. Unterstützt die für das Management der *Story Cards* genutzte Software andere Formen der Verknüpfung oder wird die Sammlung in einer Gesamt-Tabelle präferiert, kann die Darstellung und Verwaltung Kontextabhängig angepasst werden. Tabelle 6.2 zeigt Beispiele von *Acceptance Test Requirements* für die oben gezeigte *Story Card*.

Product Backlog

Das *Product Backlog* ist, wie auch in der SCRUM Methode, eine Sammlung aller *Story Cards*. Es dient der Übersicht darüber, welche Teile des Projektes noch zu erledigen sind beziehungsweise welche bereits abgearbeitet wurden. Das *Product Backlog* wird hierfür in Form einer Tabelle mit mindestens drei Spalten geführt. Die Spalten unterteilen die *Story Cards* in die Kategorien „offen“, „aktueller Sprint“und „erledigt“. Diese Kategorisierung erlaubt eine schnelle Übersicht des Projektfortschritts und eine einfache Planung der kommenden Iterationen. Zielführend ist dabei zusammenhängende *User Stories* zu gruppieren, da eine gemeinsame Abarbeitung korrespondierender Programmbestandteile durch die hierbei erzielte Verringerung des Einarbeitungsaufwandes die höchste Effizienz und Effektivität verspricht. Darüber hinaus kommen im Zuge der *Product Backlog* Erstellung die im Rahmen des *Story Card* Entwurfs erfolgten Aufwands- und Risiko-Bewertungen zum tragen. *Story Cards* mit hohem Aufwand und / oder hohem Risiko sind dabei weiter oben zu platzieren, jene mit geringem Aufwand und / oder geringem Risiko weiter unten. Diese Anordnung unterstützt die Einplanung der offenen Aufgaben nach der Relevanz für den Projektfortschritt, da so die Aufgaben, welche potentiell den größten negativen Effekt auf das Projekt haben, privilegiert und somit Projektrisiken an den Anfang des Projektes verlagert werden. Das *Product Backlog* kann um weitere Spalten zur Iterationsplanung ergänzt werden, dies muss in Abhängigkeit der für das konkrete Projekt vereinbarten Verantwortlichkeiten und Planungshorizonte angepasst werden. So ist es zum Beispiel möglich, den weiter unten beschriebenen *Release Plan* durch entsprechende Spalten in das *Product Backlog* zu integrieren.
Das *Product Backlog* ist das zentrale Planungs- und Informationswerkzeug für die Rolle des *Projektleiters* (siehe: 6.4.2). Wird das *Product Backlog* in einer Software verwaltet, kann darüber hinaus dem *Kunden* des Projektes Zugriff auf diese Übersicht gewährt werden, um damit die Transparenz und Sicherheit zu unterstützen.
Abbildung 6.3 zeigt den Auszug eines *Product Backlogs*, in welchem die Aufgaben der linken Spalte noch nicht für eine Iteration terminiert wurden, die Aufgaben der linken mittleren Spalte im Rahmen des aktuellen Sprints bearbeitet werden sollen, die der rechten mittleren Spalte aktuell bearbeitet werden und die der rechten Spalte bereits implementiert sind.

Release Plan

Der *Release Plan* stellt eine Art höherer Meilenstein-Planung dar. In dieser werden wichtige Deadlines wie Präsentationen, Messen oder andere Verpflichtungen des *Kunden* und des Teams festgehalten. Auf Basis der Kundenwünsche sowie der im *Product Backlog* vorhandenen Übersicht werden hierin wichtige *Story Cards* für die jeweiligen Daten priorisiert. Der *Release Plan* ist dabei, im Gegensatz zur klassischen Projektplanung, kein statisches Dokument, welches als Teil der *Definition of Done* zu Projektbeginn festgeschrieben werden kann. Werden dem Projekt neue *User Stories* hinzugefügt, kann dies zur Anpassung des *Release Plans* führen. Ist die Leistungsfähigkeit eines Teams noch nicht bekannt, so können falsche

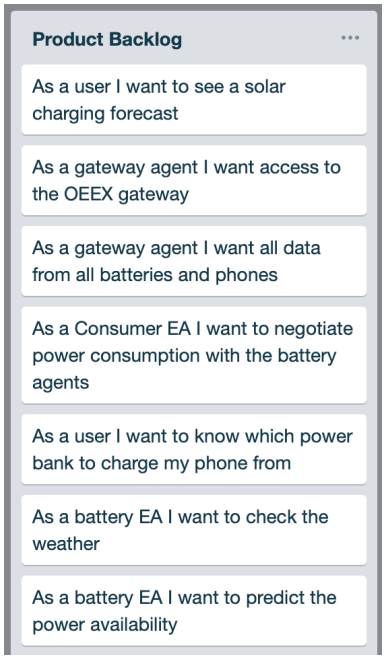

Abbildung 6.3.: Beispiel eines *Product Backlogs*

Annahmen in der Aufwandsabschätzung zu Verzögerungen und somit zu Anpassungen des *Release Plans* führen. Auch wenn der *Release Plan* bei unerfahrenen Teams somit stets nur ein tagesaktuelles Bild der Projektplanung gibt, ist er insbesondere in der Kundenkommunikation ein essentielles Werkzeug. Der *Release Plan* kann in seiner einfachsten Form als Liste geführt werden.

Sprint Backlog

Abgeleitet aus dem *Product Backlog* und unter Einbeziehung der im *Release Plan* festgehaltenen Verpflichtungen wird für jede Iteration der Produkterstellung ein neues *Sprint Backlog* erstellt, dieses korreliert mit dem *Sprint Backlog* in SCRUM und auch dem Iteration Plan des XP Ansatzes. Der Term „Sprint“, welcher in Abschnitt 6.2 näher erläutert wird, beschreibt in 2DECS, wie auch in anderen agilen Ansätzen, die im Rahmen einer Iteration abzuarbeitenden Aktivitäten, weshalb dieser Begriff im Folgenden verwendet werden soll.
Alle in einem Sprint abzuarbeitenden *Story Cards* werden im *Sprint Backlog* festgehalten und einzelnen *Entwicklern* zugewiesen. Abhängigkeiten zwischen *Story Cards* werden durch ihre örtliche Gruppierung im Sprintbacklog visualisiert. Wichtig ist dabei die Länge des Sprints, die Komplexität der zu übernehmenden Arbeiten sowie die bisherige Umsetzungsgeschwindigkeit des Teams zu berücksichtigen. Das Sprintbacklog wird ähnlich dem *Product Backlog* in Tabellenform mit mindestens drei Spalten geführt. Die Spalten unterteilen die dem *Sprint Backlog* hinzugefügten *Story Cards* in die Kategorien „offen“, „in Arbeit“und „erledigt„. Wird der in Abschnitt 6.3 empfohlene Staging-Prozess, also die Validation und Integration des Codes in spezifischen Umgebungen, befolgt, können weitere Spalten für entsprechenden Transformationsschritte ergänzt werden. Abbildung 7.4 zeigt einen Ausschnitt eines *Sprint Backlogs* nebst des aktuellen Sprintfortschritts. Das Sprintbacklog dient primär den *Entwicklern* ihre kurzfristig anstehenden Aufgaben zu strukturieren. Darüber hinaus kann die

Projektleitung den aktuellen Entwicklungsfortschritt überwachen und etwaigen Verzögerungen schnell entgegenwirken.

Acceptance Test Report

Acceptance Test Reports basieren auf den *Acceptance Test Requirements* individueller *Story Cards* und dienen der Verifikation und Dokumentation der Abarbeitung der darin dokumentierten *User Stories*. Mit Hilfe des *Acceptance Test Reports* wird sichergestellt das ein *Inkrement* alle im Rahmen der *Story Card* Erstellung definierten Anforderungen erfüllt. Diese explizite Dokumentation der Ergebnisse wird in anderen Entwicklungsansätzen wie SCRUM oder XP nicht gefordert. Im Hinblick auf die erhöhten Sicherheitsanforderungen, welche an Automatisierungstechnik in Energienetzen gestellt werden, ist dies jedoch ein essentieller Teil der 2DECS Methode. Die Ergebnisse sind, positiv wie auch negativ, in der Liste der *Acceptance Test Requirements* bzw. in der zur Verwaltung dieser Artefakte genutzten Software zu vermerken. Es ist dabei kenntlich zu machen, wann und wer die Erfüllung konstatiert hat. Wird in einem *Acceptance Test Report* festgestellt, dass alle Anforderungen an eine *Story Card* erfüllt wurden, kann diese im *Product Backlog* in die Kategorie „erledigt“ verschoben werden.

Inkrement

Nach der Abarbeitung einer *Story Card* und der erfolgreichen Prüfung aller *Acceptance Test Requirements* der spezifischen *Story Card* werden die implementierten Features als *Inkrement* betrachtet. Dieses wird in 2DECS, ähnlich dem *Inkrement* in SCRUM oder auch dem *Small Release* in XP, als spezifischer Mehrwert für das Produkt interpretiert. Im Gegensatz zu den beiden genannten Prozessen setzt 2DECS jedoch nicht voraus, dass jedes *Inkrement* eine in sich lauffähige Software darstellen muss, welche in das Zielsystem integriert werden könnte. Im Hinblick auf die spezifischen Sicherheitsanforderungen der Energienetztechnik und anderer industrieller Anwendungen wird von diesem Anspruch Abstand genommen, um die für eine Integration notwendigen Test und Zertifizierungen nach der Erarbeitung mehrerer *Inkremente* bündeln zu können und damit den Prozess so effizient wie möglich zu gestalten.

Zusammenfassung der organisatorischen Artefakte

Wie in Abbildung 6.1 dargestellt und auf den vorangegangenen Seiten beschrieben, ist das organisatorische Artefaktmodell des 2DECS-Ansatzes an die Artefaktmodelle von SCRUM und XP angelehnt. Trotz einer gemeinsamen Basis unterscheiden sich die Modelle in den folgenden Aspekten:

- **Ausgestaltung der Artefakte** - Die Ausgestaltung der individuellen Artefakte unterscheidet sich zum Teil wesentlich zwischen den drei Modellen.
- **Anzahl der Artefakte** - Die Anzahl an Artefakten variiert wie im Falle von SCRUM von fünf bis hin zu zehn bei 2DECS. Dies begründet sich durch die unterschiedlichen Inhalte und Umfänge der individuellen Artefakte und Ansätze.
- **Reihenfolge der Artefakte** - Ebenfalls durch die Inhalte der Artefakte, sowie die philosophien der Entwicklungsansätze bedingt, variiert die Reihenfolge der individuellen Artefakte.

- **Fokus auf Qualitätssicherung** - Die Möglichkeiten der Qualitätssicherung sind SCRUM, XP und 2DECS unterschiedlich ausgeprägt. Während in SCRUM eine solche zunächst nur in abstrakter Form vorgesehen ist, wird in 2DECS hierauf besonderer Augenmerkt gelegt. XP bewegt sich mit den darin definierten Acceptance- und Unit-/ System-Test-Requirements im Mittelfeld.

6.1.2. Technische Artefakte

Abbildung 6.4 zeigt die Elemente des technischen Artefaktmodells im Vergleich zu denen der Vorgehensmodelle GAIA, TROPOS und O-MaSE.

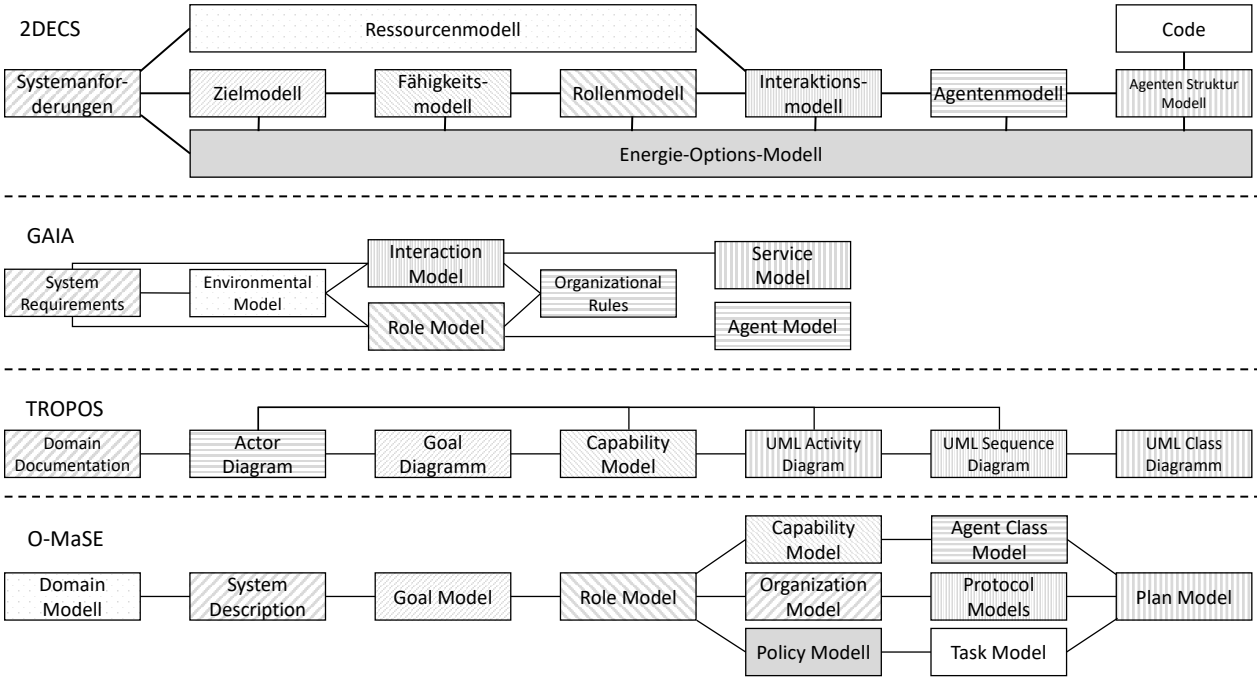

Abbildung 6.4.: Vergleich der technischen Artefaktmodelle von GAIA, TROPOS, O-MaSE und 2DECS

Wie bereits in Abbildung 6.1 werden die Abhängigkeiten der Elemente voneinander in Bild 6.4 in durch ihre Anordnung auf der Horizontalen abgebildet. Linien, welche von links nach rechts führen, zeigen ebenfalls eine Abhängigkeit des rechten (Ziel-)Elements vom linken (Start-)Element. So bilden zum Beispiel die *Systemanforderungen* die Grundlage für das *Ressourcenmodell*, das *Zielmodell* und das *Energie-Options-Modell*. Ähnlichkeiten der Artefakte zwischen den Ansätzen werden auch in dieser Abbildung durch eine gleichartige Schraffur angedeutet. Die Abhängigkeiten, Ähnlichkeiten und Inhalte der Elemente sollen im Folgenden beschrieben werden:

Systemanforderungen

Systemanforderungen bilden in 2DECS, ebenso wie in den als Referenz dienenden Ansätzen GAIA, TROPOS und O-MaSE, die Grundlage der weiteren Entwicklung. In ihnen werden die Anforderungen an die zu entwickelnde Software lösungsneutral definiert. Diese Beschreibung

ist zunächst noch technisch abstrakt und umfasst lediglich die Rand- und Rahmenbedingungen der Lösung. Die *Systemanforderungen* unterscheiden sich von den *User Stories* und *Story Cards* der organisatorischen Artefakte durch ihre inhaltliche Ausrichtung. Die organisatorischen Artefakte werden aus Sicht des Nutzers definiert und enthalten die für ihn relevanten Fragestellungen. *Systemanforderungen* fokussieren auf technische Faktoren und Rahmenbedingungen. So können beispielsweise zu berücksichtigende Standards und Normen ebenso aufgeführt werden wie die übergeordneten Ziele oder der Umfang und die Ausprägung der zu steuernden Hardware. Diese Anforderungen werden bereits in der ersten Phase, der *Initialen Anforderungsermittlung* (siehe Abschnitt 6.2.1), definiert und im Anschluss im Rahmen der *Vorplanung* (siehe Abschnitt 6.2.2) weiter ausgearbeitet. Sie stellen somit Ausgangsartefakte der beiden Phasen dar. Die *Systemanforderungen* können in unterschiedlicher Form dargestellt werden, wobei die einfachste eine Liste ist. Alternative Darstellungsformen wie zum Beispiel SysML-Anforderungsdiagramme, Mindmaps oder schriftliche Ausarbeitungen sind von den jeweiligen Projektanforderungen und Präferenzen des Management- und Entwicklungsteams abhängig.

Ressourcenmodell

Das *Ressourcenmodell* umfasst technische Beschreibungen aller in der Zielumgebung befindlichen technischen Ressourcen, welche durch die zu entwickelnde Leitlösung beeinflusst werden können beziehungsweise Einfluss auf die Regelstrecke haben. Die Beschreibung wird idealerweise in Form von Simulationsmodellen bereitgestellt, welche im Rahmen der *Simulation* (Siehe Abschnitt 6.2.7) und *Testbedanwendung* (Siehe Abschnitt 6.2.8) zur Validierung und Test der implementierten Software von der System- bis hin zur *Unit*-Ebene wie beispielsweise dem agenteninternen *Reasoning* genutzt werden können. Die Charakteristika und Grundlagen der Modelle werden durch 2DECS nicht eingeschränkt oder vorgegeben und müssen entsprechend den Bedürfnissen des konkreten Projektes gewählt werden. So kann die Simulation und das Testbed in unterschiedlichen Sprachen und Werkzeugen wie *MATLAB*[3], *GridLAB-D*[4] oder eigenen (agentenbasierten) Simulationen umgesetzt werden. Neben der korrekten Abbildung des Anlagenverhaltens, ist es dabei wichtig, die Schnittstellen der Komponenten denen der realen Anlagen nachzuempfinden, um so bereits frühzeitig Protokolle und Kompatibilitäten berücksichtigen zu können.
Das *Ressourcenmodell* beschreibt somit die Umgebung des Agentensystems. Diese Beschreibung findet sich in GAIA als Environmental Model und in O-MaSE als Domain Model wieder.

Energie- & Optionsmodell

Mittels des *Energie- & Optionsmodell* (EOM) werden alle zur Beschreibung der Umwelt notwendigen Skalen und Faktoren sowie deren Abhängigkeiten eindeutig definiert. Hierzu zählen sowohl die zu verwendenden Einheiten und Verknüpfungen zwischen den einzelnen Energieträgern als auch grundlegende Definitionen, wie die der Zeit und der darin zu betrachtenden Schrittweiten. Es ersetzt die in klassischen Reglern mathematisch definierten individuellen Übertragungsfunktionen. Das zur Regelung eines Systems benötigte Modell wird im *Energie- & Optionsmodell* vorgehalten und später im *Ressourcenmodell* gekapselt. Das *Energie- & Optionsmodell* basiert auf dem in Abschnitt 4.2 beschriebenen Modell des Energie-Agenten

[3]Siehe: https://de.mathworks.com
[4]Siehe: https://www.gridlabd.org

und bildet die Grundlage zur Erstellung des *Ressourcenmodells*. Das *Energie- & Optionsmodell* muss in der Anfangsphase des Projektes erarbeitet werden, da es, gemeinsam mit dem *Ressourcenmodell*, die Grundlage der weiteren Entwicklungsarbeit darstellt. Das *Energie- & Optionsmodell* wird in keinem der betrachteten Ansätze beschrieben. Der 2DECS zugrundeliegenden hybride Entwicklungs- und Erprobungsansatz macht die Nutzung eines solchen Modells jedoch notwendig. Nur über eine klare und über den gesamten Entwicklungsprozess konsistente Definition der zu steuernden und beeinflussbaren physikalischen Randbedingungen kann die Umsetzung der Anforderungen überprüfbar gemacht werden.
Als Beschreibungsmittel eignen sich Ontologien, welche wiederum in den zur Simulation des *Ressourcenmodells* genutzten Werkzeugem eingebunden werden können. Für kleine Projekte kann aber auch eine Tabellenform ausreichend sein.

Zielmodell

Angelehnt an die in TROPOS und O-MaSE beschriebenen *Zielmodelle* sieht der 2DECS Ansatz eine Hierarchisierung der Systemziele vor. An der Spitze steht das bzw. stehen die Ziele des Gesamtsystems. Diese werden dann in Sub-Ziele heruntergebrochen, welche die Menge der zur Erreichung der höherwertigen Ziele notwendigen Elemente repräsentieren. Die Anzahl der Hierarchieebenen oder die darin enthaltene Zahl an Zielen ist nicht vorgegeben und variiert in Abhängigkeit des Projektumfangs und der Komplexität der zu lösenden Aufgabenstellung. Abbildung 7.5 zeigt beispielhaft ein *Zielmodell* mit vier Hierarchieebenen. Neben der Nutzung von Stift und Papier oder auch eines Whiteboards mit entsprechenden Moderationskarten kann das *Zielmodell* auch mit anderen Grafik-Tools wie zum Beispiel PowerPoint oder MindMapping-Programmen abgebildet werden. Es ist dabei zu beachten, dass sich das *Zielmodell* über den Projektverlauf weiterentwickelt. Dabei können Ziele hinzugefügt, verschoben oder sogar entfernt werden. Auch ist es möglich die Hierarchieebenen von Zielen an die sich entwickelnden Bedürfnisse im Projekt anzupassen. Es ist daher vorgesehen das *Zielmodell* sowohl im Rahmen der Anforderungsermittlung (siehe Abschnitt 6.2.3) als auch während der *Basisplanung* (siehe Abschnitt 6.2.4) zu überarbeiten.

Fähigkeitsmodell

Das *Fähigkeitsmodell* wird aus dem *Zielmodell* abgeleitet und stellt die zur Erlangung der Ziele notwendigen Fähigkeiten dar. Dieses Modell orientiert sich ebenfalls an den in TROPOS und O-MaSE beschriebenen Fähigkeitsmodellen. Eine im *Fähigkeitsmodell* beschriebene Fähigkeit muss zur Umsetzung mindestens eines Zieles notwendig sein. Die Fähigkeit kann dabei auch der Erreichung mehrerer Ziele dienen. Alternativ können auch mehrere Fähigkeiten notwendig sein, um ein Ziel zu implementieren. Das *Fähigkeitsmodell* wird in der Phase der Anforderungsermittlung (Siehe Abschnitt 6.2.3) erarbeitet. Sollte in der späteren *Basisplanung* das *Zielmodell* derart überarbeitet werden, dass auch eine Anpassung des *Fähigkeitsmodells* notwendig wird, ist ein Rückfall in die Anforderungsermittlung notwendig. Das *Fähigkeitsmodell* wird idealerweise in einer Liste oder Matrix dargestellt, in welcher die Verbindung der Fähigkeiten zu den von ihnen unterstützten Zielen visualisiert wird.

Rollenmodell

Ähnlich dem *Fähigkeitsmodell* beschreibt das *Rollenmodell* Entitäten, welche in Beziehung zu denen eines anderen Modells stehen. In diesem Fall dienen die im *Rollenmodell* definierten Rollen als Container für die zuvor beschriebenen Fähigkeiten. Eine Rolle kann dabei eine oder mehrere Fähigkeiten umfassen. Auch können mehrere Rollen die selbe Fähigkeit aufweisen. Rollen sind voneinander unabhängig und dienen der Aggregation ähnlicher oder zur Erreichung bestimmter Ziele korrelierender Fähigkeiten, um diese später konkreten Agenten zuzuweisen. Das *Rollenmodell* ist aus GAIA und O-MaSE bekannt, wo es ebenfalls zur Abstraktion der Funktionalitäten von der tatsächlichen späteren *Implementierung* genutzt wird. Auch zur Repräsentation des *Rollenmodells* eignet sich eine Liste oder Matrix, in welcher die Verbindung der Rollen zu den von Ihnen umfassten Fähigkeiten dargestellt wird. Diese sollte mit einem möglichst flexiblen Werkzeug wie beispielsweise Excel oder mittels Moderationskarten an einem Whiteboard geführt werden, da sich diese Zuordnung über die Projektlaufzeit ändern kann. Um diesen Umstand abzubilden, wird das *Rollenmodell* sowohl in der Phase der Anforderungsermittlung (Siehe Abschnitt 6.2.3) als auch im Rahmen der *Basisplanung* (Siehe Abschnitt 6.2.4) betrachtet.

Interaktionsmodell

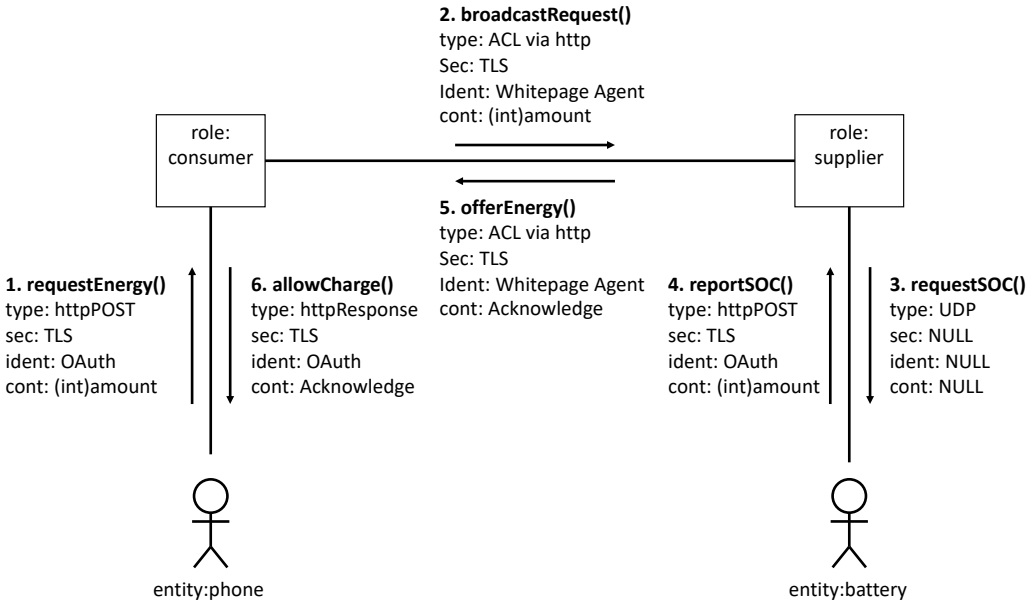

Abbildung 6.5.: Beispiel eines 2DECS-Interaktionsmodells

Das *Interaktionsmodell* reflektiert die in einem Agentensystem zentralen Eigenschaften der Inter-Agenten- sowie der externen-Kommunikation. Das *Interaktionsmodell* bildet dabei sowohl alle beteiligten Knoten der Kommunikation sowie die darin transportierten Inhalte und dafür vorgesehenen Protokolle ab. Mittels dieser Übersicht ist es möglich, die für eine Kommunikationsverbindung notwendige Bandbreite und Sicherheit zu bemessen. Das *Interaktionsmodell* umfasst dabei eine Sammlung aller notwendigen Kommunikationsverbindung in Form von erweiterten UML-Kommunikationsdiagrammen (ehemals Kollaborations-

diagramm). Die Erweiterungen betreffen dabei die Informationen zu Kommunikationsinhalten und -Protokollen. Diese umfangreiche Darstellung wird zur besseren Diskussion und Analyse der für Agenten äußerst wichtigen Eigenschaft der Kommunikation genutzt. Hierbei sollen neben technischen Fragen insbesondere Aspekte der Normen- und Standardkonformität im Hinblick auf Themen wie Sicherheit und Datenschutz betrachtet werden. Andere Ansätze wie GAIA, TROPOS oder O-MaSE nutzen ebenfalls *Interaktionsmodelle*, verzichten dabei aber auf die inhaltliche Tiefe und beschränken sich auf die reine Verknüpfung der Kommunikationsknoten. Abbildung 6.5 zeigt beispielhaft ein *Interaktionsmodell* mit vier Kommunikationspartnern.
Das Kommunikationsmodell wird im Zuge der *Basisplanung* erarbeitet (siehe Abschnitt 6.2.4) und bildet neben dem *Rollenmodell* die Grundlage zur Erarbeitung des *Agentenmodells*. Das *Interaktionsmodell* kann sowohl mit Stift und Papier als auch mit entsprechender IT-Unterstützung in anpassbaren UML-Editoren erstellt werden. Änderungen am *Interaktionsmodell* werden nur dann nötig, wenn sich einzelne Kommunikationsinhalte, Rollen oder deren Fähigkeiten ändern.

Agentenmodell

Die letzte Stufe der Aggregation von Rollen und den ihnen untergeordneten Fähigkeiten wird durch das *Agentenmodell* repräsentiert. In diesem werden auf Basis der notwendigen Interaktionen Agenten definiert. In Abhängigkeit von den gegebenen Randbedingungen der Kommunikation und des Datenschutzes werden den Agenten spezifische Rollen zugewiesen um so einen angemessenen Grad an dezentraler Autonomie und Kapselung von Daten zu erreichen. Das *Agentenmodell* ist aus den drei betrachteten Ansätzen bekannt. In diesem werden lediglich die Agenten und in GAIA und O-MaSE die ihnen zugeteilten Rollen bzw. in TROPOS deren Fähigkeiten abgebildet. Dies lässt sich wiederum sehr einfach in einer Tabelle oder Matrix, aber auch in einem Klassendiagramm oder einer grafischen Darstellung, in welcher die Agenten mit den ihnen zugeteilten Rollen und deren Kommunikationsverbindungen untereinander dargestellt werden, realisieren. Abbildung 6.6 zeigt beispielhaft eine solches grafisches *Agentenmodell*. Das *Agentenmodell* wird im Rahmen der *Ausführungsplanung* erstellt und repräsentiert den letzten Schritt vor der eigentlichen Implementierung und dafür notwendigen *Feinplanung* der einzelnen Agenten.

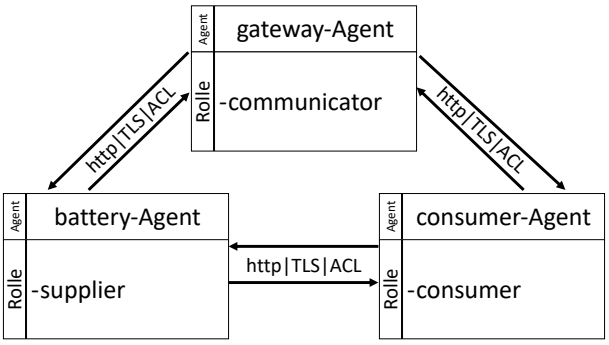

Abbildung 6.6.: Beispiel eines Agentenmodells

Agenten-Struktur-Modell

Das *Agenten-Struktur-Modell* beschreibt das Innenleben der individuellen Agenten. Es wird in Form eines UML-Klassendiagramms dargestellt und durch den mit der Implementierung beauftragten *Entwickler* erstellt. Es beinhaltet alle für die Implementierung und spätere Dokumentation notwendigen Informationen über die Fähigkeiten des Agenten, welche in Funktionen abgebildet werden, sowie dessen mögliche interne Zustände, Variablen, Schnittstellen und Abhängigkeiten.

Code

Die Schaffung des Software *Codes* ist das Ziel des Umsetzungsstadiums. Er beinhaltet alle in den vorangegangenen Modellen erarbeiteten Informationen und Abhängigkeiten und setzt diese in funktionale Einheiten um. Andere Entwicklungsansätze wie GAIA, TROPOS oder O-MaSE betrachten den Quellcode nicht als eigenes Artefakt, da sie eine weitere Überprüfung des geschaffenen Produktes nicht vorsehen. Im Hinblick auf die Verschränkung organisatorischer- und technischer Aspekte in 2DECS wird die Berücksichtigung dieses Artefakts jedoch als unabdingbar betrachtet.

Zusammenfassung der technischen Artefakte

Wie in Abbildung 6.4 dargestellt und auf den vorangegangenen Seiten beschrieben, ist das technische Artefaktmodell des 2DECS-Ansatzes an die Artefaktmodelle von GAIA, TROPOS und O-MaSE angelehnt. Trotz einer gemeinsamen Basis unterscheiden sich die Modelle, ähnlich dem organisatorischen Arefaktmodell, in den folgenden Aspekten:

- **Ausgestaltung der Artefakte** - Die Ausgestaltung der individuellen Artefakte unterscheidet sich zum Teil wesentlich zwischen den drei Modellen.
- **Anzahl der Artefakte** - Die Anzahl an Artefakten variiert wie in den Fällen von GAIA und TROPOS von sieben bis hin zu elf bei 2DECS. Dies begründet sich durch die unterschiedlichen Inhalte und Umfänge der individuellen Artefakte und Ansätze.
- **Reihenfolge der Artefakte** - Ebenfalls durch die Inhalte der Artefakte, sowie die philosophien der Entwicklungsansätze bedingt, variiert die Reihenfolge der individuellen Artefakte.
- **Vollständigkeit der Modelle** - Die vier Entwicklungsansätze fokussieren auf unterschiedliche Phasen des Entwicklungsprozesses, weshalb sie zum Teil Artefakte wie z.B. den eigentlichen Code nicht betrachten.

6.2. Phasen

Das Phasenmodell des 2DECS Ansatzes gliedert die Erstellung der oben beschriebenen Artefakte in vier wesentliche Projektstadien. Dies ist zunächst das initiale Stadium der Projektarchitektur und grundlegenden Produktdefinition, welches die Phasen der *Initialen Anforderungsermittlung* und der *Vorplanung* umfasst. Daran schließt sich ein Entwicklungsstadium an, welches einen iterativen Prozess der weiteren Produktbeschreibung umfasst. Es folgt das

ebenfalls iterative Umsetzungsstadium, welche die Produkterstellung und -analyse umfasst. Das letzte Stadium beinhaltet lediglich die Phase des *Lifecycle Managements*, aus welcher, basierend auf entsprechend identifiziertem Bedarf, ein neuer Projektzyklus angestoßen und somit in die *Initiale Anforderungsermittlung* zurückgegangen werden kann. Die einzelnen Phasen, welche den Stadien untergeordnet sind, werden über Ein- und Ausgangsartefakte definiert. Die beiden mittleren Stadien der Entwicklung und Umsetzung, welche durch eine repetitive Abarbeitung mehrerer Projektphasen gekennzeichnet sind, werden in sogenannten Sprints abgearbeitet. Sprints beschreiben in agilen Entwicklungsmodellen einen Zyklus dieser repetitiven Abarbeitung. Ein Sprint ist dabei stets von einer initial definierten Länge. *Entwicklungsteam*, *Projektleitung* und *Kunde* müssen sich unter Beachtung der Projektbesonderheiten wie der Komplexität des Produktes, den vorhandenen Ressourcen und deren Erfahrungsschatz sowie den Anforderungen des *Kunden* an regelmäßige Interaktion am Anfang des Projektes auf die anzustrebende Sprintlänge einigen. In der Regel liegt diese bei einfachen Projekten zwischen ein und zwei Wochen und kann bei komplexen und mehrjährigen Projekten auf mehrere Monate ansteigen. In 2DECS werden die Phasen der *Anforderungsermittlung*, *Basisplanung*, *Ausführungsplanung*, *Implementierung*, *Simulation*, *Testbedanwendung*, des *Debriefings* sowie des *Deployments* einem solchen Sprint zugeordnet. Wird ein Sprint komplett durchlaufen, also von der *Anforderungsermittlung* bis hin zum *Deployment*, kann, in Abhängigkeit der Entwicklungsaufgaben, ein Rückfall in die Phase der *Anforderungsermittlung*, der *Basisplanung* oder der *Ausführungsplanung* erfolgen. Wird in den Phasen der Basis- oder *Ausführungsplanung* ein Defizit in den zuvor definierten *Story Cards*, *Acceptance Test Requirements*, des *Product Backlogs* oder auch des *Release Plans* festgestellt, ist ein Rückfall in die Phase möglich, in welcher diese Artefakte erarbeitet wurden. Sind alle Ausgangsartefakte adäquat erstellt, schließen sich die Phasen der *Implementierung*, *Simulation* und *Testbedanwendung* an, welche im Umsetzungsstadium subsumiert werden. Werden im Zuge der *Simulationsphase* Unzulänglichkeiten des in der vorherigen Phase erarbeiteten *Codes* festgestellt, so wird die Phase der *Implementierung* erneut aktiviert und auf eine Erfüllung der *Acceptance Test Requirements* hingearbeitet. Auch aus der an die Simulation anschließende Phase der *Testbedanwendung* ist ein Rückfall in die *Implementierungsphase* möglich. Die Abbildungen 6.7 und 6.8 zeigen die Projektphasen und deren Verknüpfungen durch Ein- und Ausgangsartefakte sowie die beiden zentralen Projektstadien. Parallelen zu bekannten Entwicklungsprozessen werden durch farbige Hervorhebungen der Artefakte dargestellt.
Im Folgenden werden nun die einzelnen Phasen sowie deren Ein- und Ausgangsartefakte und die hierdurch definierten vorhergehenden bzw. nachfolgenden Phasen beschrieben. Die Ein- und Ausgangsartefakte der Phasen werden in separaten Abbildungen illustriert.

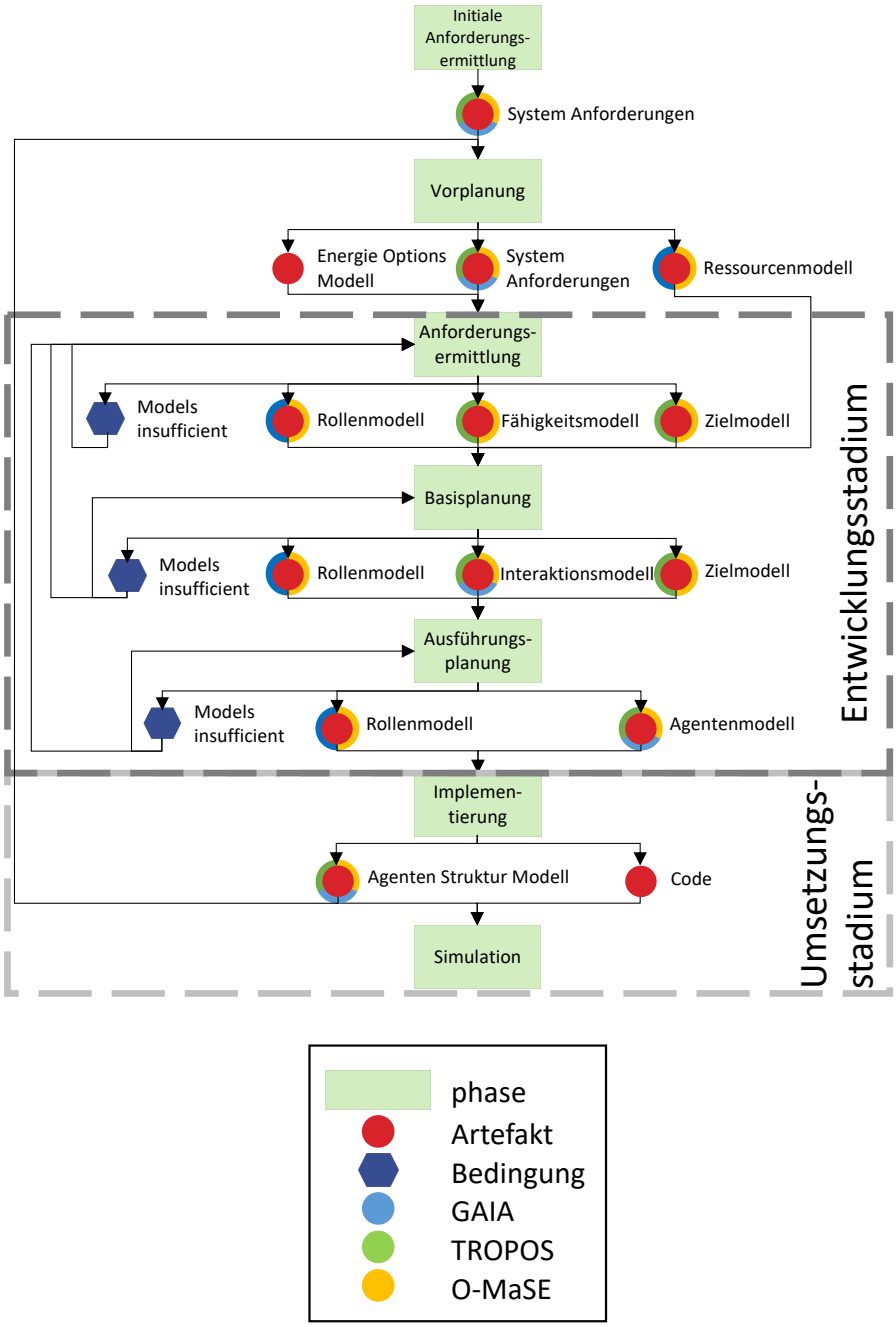

Abbildung 6.7.: Technisches Phasenmodell

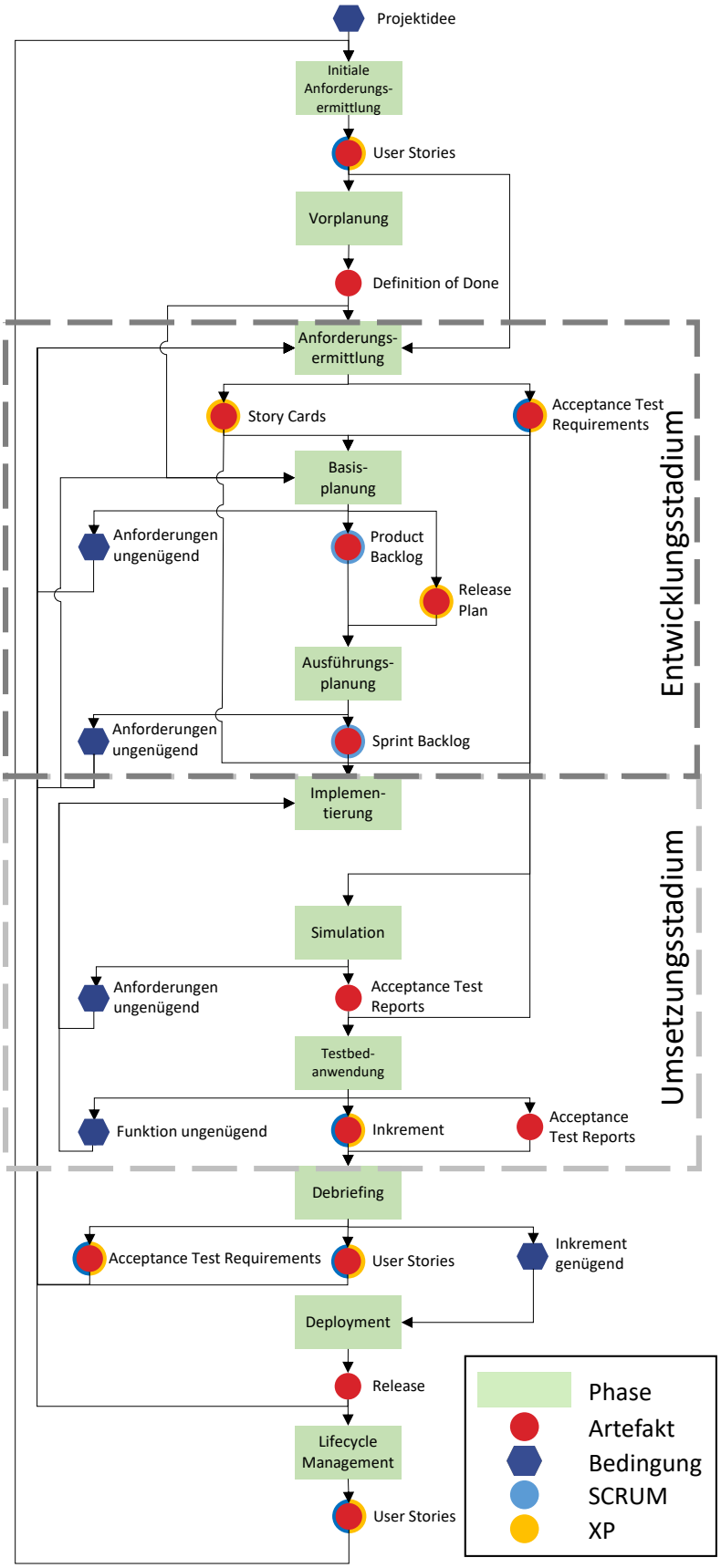

Abbildung 6.8.: Organisatorisches Phasenmodell

6.2.1. Initiale Anforderungsermittlung

Die *Initiale Anforderungsermittlung* repräsentiert die erste Phase des Entwicklungsansatzes. Als Eingangsparameter der Phase dient eine Projektidee, deren Struktur und Repräsentationsform nicht weiter definiert wird. Ziel der *Initialen Anforderungsermittlung* ist es zwischen *Kunde* und *Projektleitung* ein gemeinsames Verständnis über die übergeordneten Ziele des Projektes zu entwickeln und über deren grundsätzliche Realisierbarkeit und die dafür benötigte Zeit und Ressourcen Einigung zu erzielen. Im Zuge der *Initialen Anforderungsermittlung* werden daher bereits die ersten *User Stories* und *System Anforderungen* erfasst. Diese sind sehr weit gefasst und bilden den Ausgangspunkt der weiteren Entwicklung. Die Phase der *Initialen Anforderungsermittlung* kann, je nach Projektumfang, zwischen mehreren Stunden, bis hin zu mehreren Tagen dauern. Gegebenenfalls kann es auch als notwendig erscheinen den Personenkreis bereits um Fachleute zu ergänzen um die getroffenen Bewertungen auf ein solides Fundament stellen zu können oder auch die Definition der *User Stories* in strukturierter Form zu moderieren.

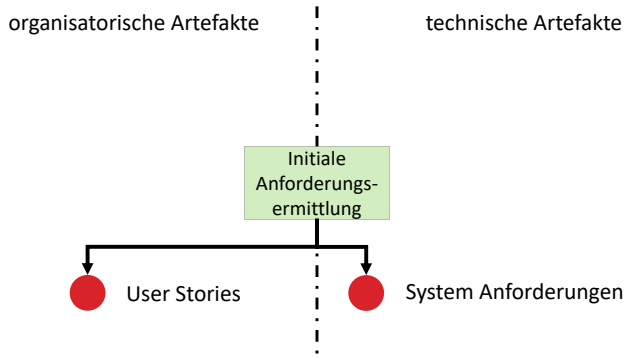

Abbildung 6.9.: Ein- und Ausgangsartefakte der Initialen Anforderungsermittlung

6.2.2. Vorplanung

Eingangsparameter der *Vorplanung* sind die zuvor definierten *User Stories* und *System Anforderungen*. Diese werden in dieser Phase unter Einbeziehung von Domänen-Experten analysiert und hieraus eine *Definition of Done* für das Gesamtprojekt abgeleitet. Die hierbei zu Rate gezogenen *Experten* sollten das Vertrauen des *Kunden* sowie der *Projektleitung* genießen, um so möglichst objektive Abbruchkriterien für den weiteren Prozess festlegen zu können. Neben der *Definition of Done* werden das *Energie-Options-Modell* und das *Ressourcenmodell* umfänglich beschrieben. Die abschließende Spezifizierung der *Definition of Done* sowie des *Ressourcenmodells* wird gefordert, da es sich hierbei und die Parameter mit den größten Auswirkungen auf den Projektumfang handelt. Dies umfasst die im Rahmen der Steuerung zu berücksichtigenden Anlagentypen und ihre grundlegenden Charakteristiken sowie die Ziele der Steuerung. Die nachfolgend nicht erweiterbaren *Energie Options-* und *Ressourcenmodell*e sollten in diesem Schritt bereits die Grenzen im Hinblick auf die Betriebs- und Informationssicherheit, umfassen. Darüber hinaus werden die in der *Initialen Anforderungsermittlung* definierten *System Anforderungen* weiter verfeinert und um Elemente ergänzt, welche sich aus den Diskussionen um die Inhalte des *Energie Options-* und *Ressourcenmodells* ergeben.

Die Phase der *Vorplanung* kann, je nach Projektumfang, zwischen mehreren Stunden bis hin zu mehreren Tagen in Anspruch nehmen.

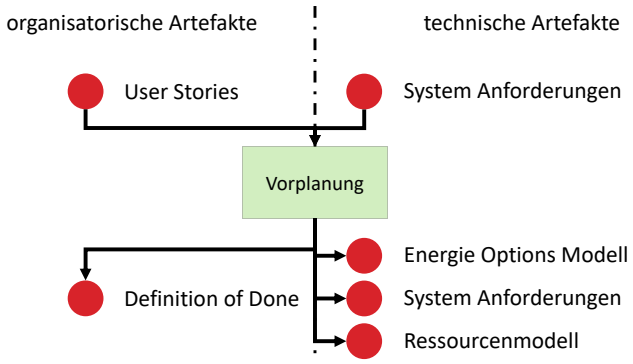

Abbildung 6.10.: Ein- und Ausgangsartefakte der Vorplanung

6.2.3. Anforderungsermittlung

Die *Anforderungsermittlung* ist grundsätzlich die erste Phase eines jeden Sprints und der Einstieg in das Entwicklungsstadiums. Diese Phase kann, sollte sich keines der Eingangsmodelle geändert haben und auch kein Update der Folgemodelle notwendig sein, übersprungen werden. Das primäre Ziel dieser Phase ist die Grundlagen eines Sprints zu definieren bzw. diese zu verfeinern. Dies gilt sowohl für die technischen wie auch die organisatorischen Aspekte eines Sprints. Als Eingangsparameter der *Anforderungsermittlung* dienen aus organisatorischer Sicht die *User Stories* sowie die *Definition of Done*. Die *User Stories* werden, unter Berücksichtigung der *Definition of Done*, vom Entwicklungsteam analysiert und in *Story Cards* sowie zugehörige *Acceptance Test Requirements* transformiert. Aus technischer Sicht bedeutet dies, dass das *Rollen-* und *Fähigkeitsmodell* auf Basis der so gewonnenen Erkenntnisse geschaffen, bzw. erweitert werden kann sowie die Ziele des Agentensystems definiert bzw. angepasst werden können. Als Eingang hierfür dienen die final definierten *System Anforderungen* sowie das *Energie Options-* und *Ressourcenmodell*.
Auch diese Phase kann, in Abhängigkeit der Projektkomplexität und des in der spezifischen Iteration benötigten Änderungsbedarfs, zwischen mehreren Stunden bis hin zu mehreren Tagen in Anspruch nehmen.

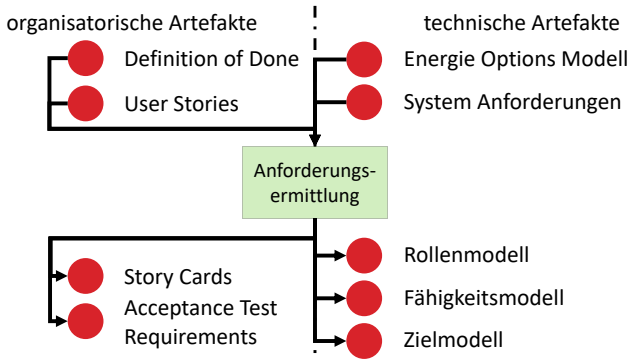

Abbildung 6.11.: Ein- und Ausgangsartefakte der Anforderungsermittlung

6.2.4. Basisplanung

Die *Basisplanung* dient der groben Projektplanung sowie der Detaillierung der technischen Umsetzungsplanung. In dieser Phase werden alle Grundlagen für die weitere Implementierung gelegt. Grundsätzlich folgt diese Phase auf die *Anforderungsermittlung*, kann jedoch auch zum Beispiel der *Ausführungsplanung* folgen, sollten hier ungenügende Anforderungen identifiziert werden. Darüber hinaus ist es möglich, aus dem *Debriefing* oder sogar einem *Deployment* in diese Phase zu wechseln, sollte eine weitere Abarbeitung der *Anforderungsermittlung* als nicht opportun betrachtet werden.
Als Eingangsparameter der organisatorischen Projektanteile dienen dabei in jedem Fall die *Story Cards* und *Acceptance Test Requirements*. Diese werden in der *Basisplanung* zu einem *Product Backlog* und dem übergeordneten *Release Plan* zusammengefasst. Existieren diese bereits, werden sie mit den neu hinzugefügten Karten aktualisiert. Aus technischer Sicht wird das Rollenmodell aktualisiert und den als Eingangsparameter dienenden *Fähigkeits- und Zielmodell* ein Interaktions- und *Rollenmodell* zur Seite gestellt. Die Phase ist ein Schlüsselelement im Hinblick auf die Einhaltung der Sicherheitsanforderungen sowie der Auswahl von adäquaten Kommunikationsstandards und -Modellen. Diese können sowohl aus der *Definition of Done* als auch den *Acceptance Test Requirements* abgeleitet werden und müssen zwingenderweise im *Interaktions- und Rollenmodell* Berücksichtigung finden. Sollten die *Acceptance Test Requirements* diese Anforderungen nicht in eindeutiger Weise wiedergeben, ist ein Rückfall aus der *Basisplanung* in die *Anforderungsermittlung* möglich. Ein weiterer Grund hierfür können nicht eindeutig definierte *Story Cards* sein.
Wie bereits in den vorangegangenen Phasen kann die *Basisplanung* zwischen mehreren Stunden bis hin zu mehreren Tagen dauern.

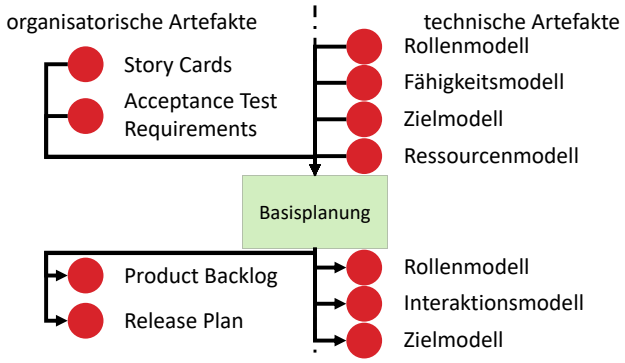

Abbildung 6.12.: Ein- und Ausgangsartefakte der Basisplanung

6.2.5. Ausführungsplanung

Im Zuge der *Ausführungsplanung* werden die Inhalte und Prioritäten der späteren *Implementierung* feingeplant und final definiert. In dieser Phase müssen Faktoren wie Zusammenhänge zwischen zu schaffenden Artefakten und somit die Zweckmäßigkeit der parallelen Entwicklung als auch die Bedürfnisse des *Kunden* über die Bereitstellung bestimmter Features zu einem bestimmten Zeitpunkt miteinander abgewogen werden. Darüber hinaus wird das *Agentenmodell* geschaffen, welches erstmals die zuvor abstrakt in Zielen, Fähigkeiten und Rollen definierten Eigenschaften der Agenten konkreten Elementen zuordnet. Die Phase der *Ausführungspla-*

nung folgt stets der *Basisplanung*. Einzig im Falle ungenügend definierter Modelle, ist eine Rekursion möglich. Da die *Ausführungsplanung* die letzte Phase des Entwicklungsstadiums ist, besteht hier nochmals die Möglichkeit zurück in die *Basisplanung* oder sogar die *Anforderungsermittlung* zurück zu kehren. Dieser Schritt kann notwendig werden, sollten die bisher erarbeiteten Artefakte nicht ausreichend definiert sein. Bei erfolgreicher Abarbeitung der Phase folgt jedoch im Regelfall die *Implementierung*.
Die organisatorischen Anteile der Phase benötigen sowohl das aktuelle *Product Backlog* als auch den *Release Plan* als Eingangsvariablen um hieraus die in diesem Sprint zu bearbeitenden *Story Cards* zu identifizieren und hieraus das *Sprint Backlog* zu erstellen. Für die Schaffung des *Agentenmodells* sowie die Verfeinerung des Rollenmodells, welche beide am Ende der Phase stehen, sind als Eingangsvariablen der Phase sowohl der aktuelle Stand des *Rollen-* und *Zielmodells* als auch das neu geschaffenen *Interaktionsmodell* notwendig. Diese Zusammenführung von Rollen in individuellen Agenten als auch die Attribution von bestimmten Kommunikationseigenschaften und Protokollen bedarf einer intensiven Evaluation im Hinblick auf sicherheits- und organisationstechnische Fragestellungen. Insbesondere Fragen der IT-Sicherheit, des Datenschutzes und der Marktrollen einzelner Akteure sollten dabei in den Fokus rücken.
Die sachgerechte Abarbeitung der *Ausführungsplanung* bedarf mindestens eines Tages. In der Regel sind, insbesondere bei komplexeren Projekten mit Sprintzyklen von einem Monat oder mehr, hierfür jedoch mehrere Tage zu veranschlagen.

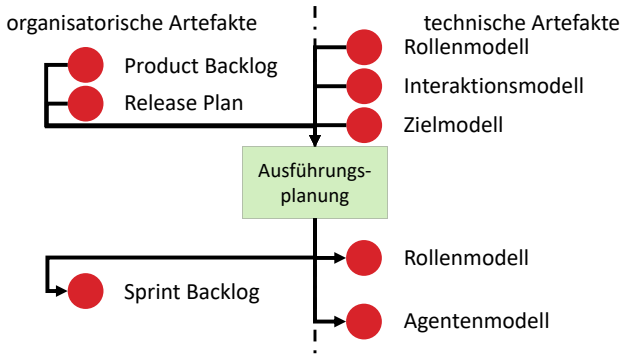

Abbildung 6.13.: Ein- und Ausgangsartefakte der Ausführungsplanung

6.2.6. Implementierung

Die Implementierungsphase dient der Erschaffung technischer Artefakte - namentlich dem *Agenten Struktur Modell* sowie dem eigentlichen Programmcode, welcher sich direkt aus dem *Agenten Struktur Modell* ergibt. Als Eingangsparameter dienen aus dem Bereich der technischen Artefakte das *Rollen-* sowie das *Agentenmodell*, von den organisatorischen Artefakten wird lediglich das *Sprint Backlog* vorausgesetzt. Auf Basis dieser Vorgaben werden im Rahmen der *Implementierung* die individuellen Agenten geschaffen, beziehungsweise erweitert. Die *Implementierung* ist der Einstieg in das Umsetzungsstadium, welches neben der Implementierungsphase auch die Phasen *Simulation* und *Testbedanwendung* umfasst.
Vorgänger der *Implementierung* ist in der Regel die *Ausführungsplanung*. Sollten jedoch in der folgenden *Simulation* oder Testbed Anwendung Nachbesserungsbedarf am hier geschaffenen Code festgestellt werden, ist ein Rückfall aus diesen Phasen in die *Implementierung*

möglich. Auf die *Implementierung* folgt stets die *Simulation* des geschaffenen Codes. Ein direkter Rückfall in eine Phase des Entwicklungsstadium ist nicht möglich. Hierdurch soll die Selbstständigkeit des Entwicklungsteams gestärkt und ein Stocken des Projektes durch Rückfälle in bereits abgeschlossene Diskussionen verhindert werden.
Die *Implementierung* sollte den Hauptteil eines jeden Sprints einnehmen - da dies die Phase ist, in welcher das Produkt vorangebracht und somit konkrete Wertschöpfung betrieben wird. Die Dauer wird somit maßgeblich von der Länge eines Sprints und der zuvor absolvierten Phasen bestimmt.

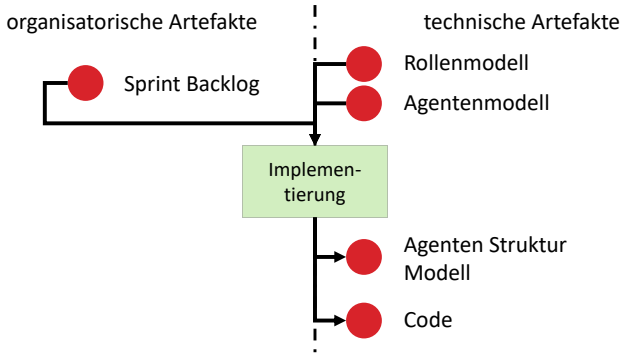

Abbildung 6.14.: Ein- und Ausgangsartefakte der Implementierung

6.2.7. Simulation

Im Rahmen der Simulationsphase wird der zuvor geschaffene Code grundlegenden Funktionstests unterzogen. Diese Basieren auf den der implementierten *Story Card* zugeordneten *Acceptance Test Requirements* und dienen der Überprüfung des einzelnen Moduls. Die hierfür genutzte Simulationsumgebung baut idealerweise auf den in der *Vorplanung* geschaffenen Ressourcenmodellen auf und erlaubt das automatisierte Testen der individuellen Anforderungen. Die Simulationsphase ist für die Qualitätssicherung des Produktes essentiell, da in dieser sichergestellt wird, dass die zuvor definierten Anforderungen erfüllt werden. Im Kontext eines Staging Prozesses[5] repräsentiert diese Phase bereits den zweiten Schritt, also den Übergang zwischen Entwicklungs- und Testumgebung. Zeigen die Tests Schwächen des Codes auf, sieht 2DECS einen Rückfall in die Implementierungsphase vor, um so die aufgetretenen Mängel zu beseitigen. Verlaufen die Tests hingegen erfolgreich, wird die Erfüllung der einzelnen Anforderungen in sogenannten *Acceptance Test Reports* dokumentiert, welche der weiteren Qualitätssicherung dienen und das in dieser Phase zu erschaffende Artefakt darstellen. Dieses dient dann der folgenden Phase der *Testbedanwendung* als Eingangsparameter.
Da das *Deployment* des Codes in der der Simulation zugeordneten Ausführungsumgebung idealerweise ebenso automatisiert durchgeführt wird, wie das eigentliche Testen, sollte diese Phase nur einen geringen zeitlichen Aufwand bedeuten, welcher wesentlich durch die Komplexität der Berichtserstellung geprägt ist. Sollten die benötigten Simulationen einen längeren Zeitraum umfassen, kann sich der verantwortliche *Entwickler* in dieser Zeit bereits neuen *Story Cards* zuwenden.

[5]Ein Staging Prozess sieht die Verifikation der entwickelten Software in mehreren Phasen und Ausführungsumgebungen vor. In der Regel handelt es sich hierbei um eine Entwicklungs-, Test-, Integrations- und Produktivumgebung, welcher ggf. einer Bereitstellungsumgebung vorgelagert ist

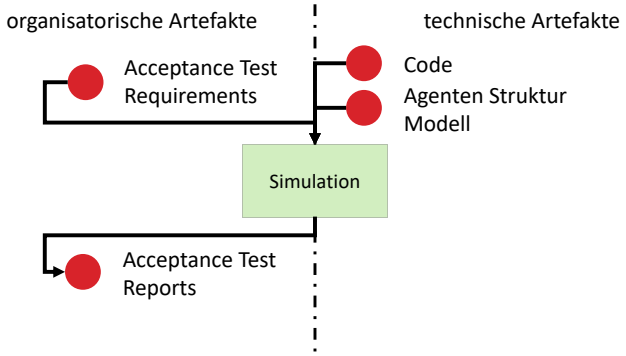

Abbildung 6.15.: Ein- und Ausgangsartefakte der Simulation

6.2.8. Testbedanwendung

Aufbauend auf den Ergebnissen der Simulationsphase sowie den *Acceptance Test Requirements* wird der implementierte *Code* in der Phase der *Testbedanwendung* weitergehenden Integrations- und *Unittests* unterzogen. Diese umfassen zusätzlich zu den bereits erfolgten Prüfung in einer idealisierten Simulation die Exposition des Codes gegenüber realer Hardware, welche, gekoppelt mit einer Simulation, ein hybrides Testbed bildet. So kann die Resilienz der Einheit, aber auch des durch die Einheit beeinflussten Gesamtsystems gegenüber Latenzen oder nicht dokumentiertem, bzw. schwer simulierbarem Verhalten der Zielumgebung getestet werden. Wichtig ist es hierbei die Ein- und Ausgangsparameter der Einheiten auf Anomalien zu überwachen, um so etwaige (falsche) Annahmen über Konstanten und Wirkzusammenhänge im physikalischen System aufzudecken. Parallel der in der Simulationsphase etablierten Prozesse sollte auch im Zuge der *Testbedanwendung* idealerweise ein automatisierter Installations- und Überprüfungs-Prozess genutzt werden, um so zum einen die Anwendung zu beschleunigen und das Testen einer Vielzahl von Konstellationen und Parametern zu ermöglichen, aber auch um die hierbei erzielten Ergebnisse von etwaigen Fehlern durch manuelle Anpassungen zu bereinigen. Im Kontext eines Staging Prozesses steht die *Testbedanwendung* auf der dritten Stufe, also der Ebene der Integrationstest-Umgebung, welche der Installation in das Produktivsystem vorgelagert ist. Ebenso wie in der Phase der *Simulation* bedeutet die Nichterfüllung der *Acceptance Test Requirements* einen Rückfall in die Implementierungsphase. Werden alle Tests positiv absolviert, werden die Ergebnisse dokumentiert und die Konformität mit den zu beachtenden Normen und Standards bestätigt. Die implementierte *Story Card* wird somit als *Inkrement* freigegeben, welches neben den obligatorischen *Acceptance Test Reports* das Ausgangsartefakt der Phase darstellt. Die *Testbedanwendung* stellt die letzte Phase des Umsetzungsstadium dar, ihr folgt stets das *Debriefing*.
Ebenso wie die Simulationsphase wird der zeitliche Aufwand der *Testbedanwendung* stark durch den Dokumentationsumfang sowie den Automatisierungsgrad der Installation und Überprüfung geprägt. Auch hier kann der *Entwickler* sich im Kontext zeitaufwändiger Tests der Implementierung neuer *Story Cards* zuwenden.

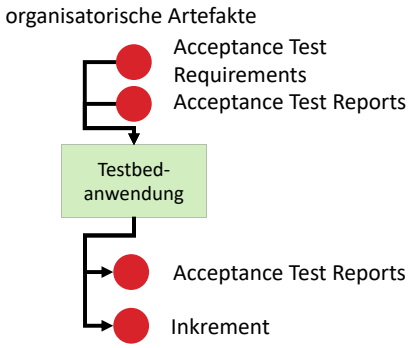

Abbildung 6.16.: Ein- und Ausgangsartefakte der Testbedanwendung

6.2.9. Debriefing

In der Debriefingphase wird das zuvor erarbeitete *Inkrement* dem *Kunden* präsentiert und durch ihn final freigegeben. Darüber hinaus werden weitere *User Stories* gesammelt und zugehörige *Acceptance Test Requirements* definiert. Diese stellen die zu erstellenden Artefakte der Phase dar. Ziel der Phase ist es die Integration des *Kunden* in den Prozess sicher zu stellen. So soll ihm zum einen der Fortschritt dargestellt, aber auch die Weiterentwicklung seiner Anforderungen erfasst werden. Diese Phase stellt somit ein Kernelement der Kundenorientierung dar. In Abhängigkeit des erzielten Feedbacks des *Kunden*, aber auch des Aufwands und Risikos des *Deployments* in der Produktivumgebung kann auf das *Debriefing* sowohl die *Deploymentphase* als auch ein Rückfall in das Entwicklungsstadium folgen. Ist das *Deployment* mit Risiken oder einem erheblichen Aufwand verbunden, so muss nicht jedes *Inkrement* individuell im Feld ausgebracht werden. Ist es jedoch möglich Over the Air Updates durchzuführen und ein zuverlässiges Rollback Verfahren etabliert, so kann man auch jedes *Inkremente* einzeln installieren.
Die Debriefingphase bedarf, in Abhängigkeit vom Projektumfang und der Anzahl der zu diskutierenden *Inkremente*, zwischen wenigen Stunden bis hin zu wenigen Tagen.

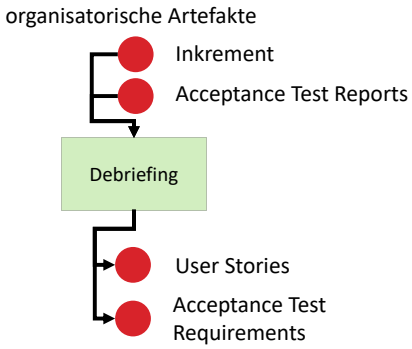

Abbildung 6.17.: Ein- und Ausgangsartefakte des Debriefings

6.2.10. Deployment

Im Zuge der *Deployment* Phase werden ein oder mehrere *Inkremente* im Produktivsystem installiert. Dieses *Inkrement*, beziehungsweise die Aggregation an *Inkrementen*, wird dann als

Release bezeichnet und stellt das Ausgangsartefakt der Phase dar. Ist der *Deploymentprozess* aufwändig oder durch hohe Risiken geprägt, so kann das Projekt durch ein finales *Deployment* aller geschaffenen *Inkremente* gekennzeichnet sein. Alternativ sind mehrere und kontinuierliche Releases während des gesamten Projektverlaufes möglich. So kann auf die Phase des *Deployments* sowohl ein Rückfall in das Entwicklungsstadium als auch ein Übergang in das *Lifecycle Management* folgen. Ersteres ist der Fall, wenn das *Product Backlog* noch nicht abgearbeitet, beziehungsweise die *Definition of Done* noch nicht erreicht ist. Letzteres kennzeichnet den Abschluss des Projektes und bedarf somit keiner weiteren Iterationen.
Die Länge der *Deploymentphase* ist vom Automatisierungsgrad des *Deploymentprozesses* sowie der Verteilung der Zielsysteme abhängig. Sie kann somit zwischen wenigen Minuten bis hin zu mehreren Wochen dauern.

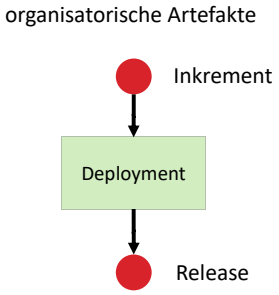

Abbildung 6.18.: Ein- und Ausgangsartefakte des Deployments

6.2.11. Lifecycle Management

Die Phase des *Lifecycle Managements* wird durch die Unterstützung des *Kunden* durch den Auftragnehmer geprägt. Das Maß der Unterstützung, welches von der Bereitstellung von Sicherheitsupdates und Patches, bis hin zum aktiven Management der Software und ihrer Ausführungsumgebung reichen kann, ist stark Projektabhängig. Auch die damit einhergehende Laufzeit der vereinbarten Services kann somit nur im Kontext des individuellen Projektes bestimmt werden. Abhängig vom initial getroffenen Projektumfang ist auch, ob beispielsweise umfangreiche Updates oder Patches als neue Entwicklungsprojekte betrachtet und in einem neuen Projektzyklus bereitgestellt werden. So können als Ergebnis des *Lifecycle Managements* neue *User Stories* generiert und neue Entwicklungsprojekte angestoßen werden.

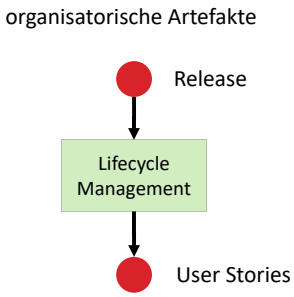

Abbildung 6.19.: Ein- und Ausgangsartefakte des Lifecycle Managements

6.3. Aktivitäten

Da die zu erschaffenden Inhalte im Zentrum des 2DECS Ansatzes stehen, orientiert sich das Aktivitätsmodell an der Struktur und den Inhalten des Artefaktmodells. Es erfolgt somit wiederum eine Zweiteilung in organisatorische und technische Aktivitäten, welche in den entsprechenden Phasen parallel oder auch aufeinander aufbauend abgearbeitet werden. Die unten beschriebenen Aktivitäten sollen dabei primär als Leitlinien verstanden werden und bedürfen alle einer projektspezifischen Ausgestaltung durch die *Projektleitung*. Ziel der Aktivitäten ist dabei stets die Erarbeitung eines oder mehrerer konkreter Artefakte. In einer Phase können eine oder mehrere Aktivitäten parallel oder sequentiell ausgeführt werden. Aktivitäten werden in einer Phase abgeschlossen, können jedoch auch in einer späteren Phase nochmals ausgeführt werden.

6.3.1. Aktivitäten zur Erarbeitung organisatorischer Artefakte

Die Aktivitäten zur Erarbeitung der organisatorischen Artefakte dienen originär dem Projektmanagement und der Einbindung des *Kunden*. Sie sind darauf ausgerichtet alle relevanten Informationen aller Stakeholder in einer effizienten Form zu erfassen und in den dafür vorgesehenen Artefakten zu aggregieren. Ein besonderer Fokus liegt dabei auf der Berücksichtigung aller relevanter Standards und Normen. Tabelle 6.3 zeigt, in welcher Projektphase welche Aktivität durchgeführt wird, welche Stakeholder an dieser beteiligt sind und welche Artefakte dabei geschaffen werden.

Aktivität	**Artefakte**	**Projektphase**	**Beteiligte**
User Story Mapping	User Stories	Initiale Anforderungs-ermittlung	Kunde Projektleiter
Definition der gemeinsamen Projektziele	Definition of Done	Vorplanung	Kunde Projektleiter Experten
Übersetzten der User Stories	Story Cards Acceptance Test Requirements	Anforderungs-ermittlung	Entwickler Experten
Grobplanung	Product Backlog	Basisplanung	Koordinator Entwickler
Feinplanung	Sprint Backlog	Ausführungsplanung	Koordinator Entwickler
Testen	Acceptance Test Reports	Simulation Testbedanwendung	Entwickler Experte
Freigeben	Inkrement	Testbedanwendung	Entwickler
Veröffentlichen	Release	Deployment	Entwickler

Tabelle 6.3.: Organisatorische Aktivitäten des 2DECS Ansatzes und ihre Zusammenhänge

User Story Mapping

Im Zuge des *User Story Mappings* werden die durch den *Kunden* oder Nutzer gewünschten Features des Softwareproduktes in Form von *User Stories* (Abschnitt 6.1.1) gesammelt. Diese Tätigkeit wird zu Beginn des Entwicklungsprozesses im Rahmen der *Initialen Anforderungsermittlung* (Abschnitt 6.2.1), nach einem Sprint im Zuge des *Debriefings* (Abschnitt 6.2.9) sowie im Kontext des nach dem Projektabschluss praktizierten *Lifecycle Managements* (Abschnitt 6.2.11) absolviert. Die *Projektleitung* sollte im Zuge dieses Brainstormings eine moderierende Funktion einnehmen, oder diese auf den *Koordinator* delegieren. Im Rahmen der Sammlung sollten die *User Stories* bereits grob kategorisiert und einzelnen Bereichen und Stakeholdern des Produktes zugeordnet werden. Wichtig ist es dabei auch sicherheitsrelevante Aspekte zu adressieren und bereits ein erste Übersicht über die anzuwendenen Standards und Normen zu erstellen. Der Ablauf einer solchen Sitzung kann grob in vier Phasen unterteilt werden. Diese beginnen mit einem allgemeinen Brainstorming. Nachdem dieses abgeschlossen ist, schließt sich eine Kategorisierungs- und daran eine Sortierphase an. Abgeschlossen wird die Zusammenkunft mit einer gemeinsamen Bestätigung und Publikation der neuen *User Stories*. Durch diesen letzten Schritt soll ein kollektives Committment geschaffen und die Qualität der Beiträge hoch gehalten werden. Das *User Story Mapping* ist eine Kernkomponente im Hinblick auf die Nutzerintegration und sollte daher einen entsprechenden Stellenwert in der Projektorganisation genießen.

Definition der gemeinsamen Projektziele

Nach einer initialen Sammlung relevanter *User Stories* ist es im Zuge der *Vorplanung* (Abschnitt 6.2.2) erforderlich, dass sich sowohl die *Kunden* als auch die Anbieterseite auf einen gemeinsame Projektumfang einigen und in einer *Definition of Done* (Abschnitt 6.1.1) niederschreiben. Diese Einigung erfolgt idealerweise zwischen der *Projektleitung* und dem *Kunden* unter Zuhilfenahme rechtlicher und gegebenenfalls fachlicher Vertreter und soll einem gemeinsamen Verständnis über die Ziele des Projektes dienen. Es kommen hier die klassischen Methoden der Lasten- und Pflichtenheft Erstellung zu tragen, wobei die Tiefe der Definition zum einen vom Projektumfang und zum anderen von der Kunden- Anbieterbeziehung abhängig ist.

Übersetzen der User Stories

In der Phase der *Anforderungsermittlung* (Abschnitt 6.2.3) analysieren *Entwickler* (Abschnitt 6.4.4) und *Experten* (Abschnitt 6.4.5) gemeinsam die im Rahmen des *User Story Mappings* gesammelten *User Stories* (Abschnitt 6.1.1) und übersetzen diese in *Story Cards* (Abschnitt 6.1.1) und zugehörige *Acceptance Test Requirements* (Abschnitt 6.1.1). Hierfür werden alle *User Stories* analysiert und in eine für die *Entwickler* verständliche Terminologie transferiert. Im Rahmen dieses Prozesses werden auch die zu berücksichtigenden Gesetzte und Regelungen eingehend studiert und deren Auswirkungen auf die individuellen *User Stories* als *Acceptance Test Requirements* in den Prozess eingebracht. Dieser Vorgang kann in der gesamten Gruppe von *Entwicklern* und *Experten*, oder bei großen Projekten in Untergruppen, geschehen. Wichtig ist, dass dieser Vorgang stets im Kollektiv erfolgt um so mittels 4-Augen Prinzip alle relevanten Faktoren zu berücksichtigen und die Möglichkeit zur Diskussion der Hintergründe einzelner *User Stories* zu geben.

Grobplanung

Die *Grobplanung* erfolgt in der Phase der *Basisplanung* (Abschnitt 6.2.4). Bei dieser Tätigkeit werden die zuvor erstellen *Story Cards* mit den zugehörigen Acceptance Test Requirements durch den *Koordinator* und das Entwicklerteam in das *Product Backlog* (Abschnitt 6.1.1)eingefügt. Sollten Zusammenhänge zwischen *Story Cards* unklar sein, so können bei Bedarf entsprechende Domänenexperten hinzugezogen werden. In Abstimmung mit der *Projektleitung* werden einzelne *Story Cards* für Präsentationen, Kundentermine oder Messen priorisiert und mit Fertigstellungsdaten versehen. Diese Werden dann im *Release Plan* (Abschnitt 6.1.1) festgehalten. Die *Grobplanung* wird neben diesen ordnenden und priorisierenden Tätigkeiten durch die eventuelle Bewertung der für die Implementierung und Überprüfung der *Story Cards* notwendigen Ressourcen geprägt. Diese Bewertung und deren Inhalten sind stark Projektspezifisch und können sich auf Aspekte wie die für die Abarbeitung benötige Zeit, der zum Test benötigten Hardware oder Kapazitäten in Clouddiensten richten. Eine Abstraktion dieser Faktoren ist durch die Einführung eines Punktesystems möglich. Anhand dessen kann dann nach Abarbeitung mehrerer *Story Cards* die Burn Down Rate des Teams gemessen und somit eine bessere *Feinplanung* vorgenommen werden.

Feinplanung

Im Zuge der *Ausführungsplanung* (Abschnitt 6.2.5) wird mittels *Feinplanung* das *Sprint Backlog* (Abschnitt 6.1.1) erstellt. Diese Tätigkeit hat zum Ziel eine angemessene Zahl an entsprechend priorisierten und idealerweise zusammenhängenden Aufgaben (*Story Cards*) für den bevorstehenden Sprint zu identifizieren. Diese Aktivität wird hauptsächlich durch den *Koordinator* und das Entwicklungsteam wahrgenommen. Die primäre Herausforderung liegt dabei bei der Skalierung des Aufgabenumfanges. Diese wird durch die Nutzung einer Metrik zur Repräsentation der Komplexität der einzelnen *Story Cards* im Zuge der *Grobplanung* erleichtert. Nichtsdestotrotz bedarf diese Aktivität eines erfahrenen Entwicklerteams und *Koordinators*, da die Ableitung von Zusammenhängen in der Regel keinem formalen Konzept folgt.

Testen

Das *Testen* des implementierten Codes ist eine der wichtigsten und potentiell aufwändigsten Tätigkeiten im Rahmen der Softwareentwicklung. In 2DECS wird diese Tätigkeit durch das Entwicklungs- und Expertenteam ausgeübt und erstreckt sich über zwei Phasen. Im Rahmen der *Simulation* (Abschnitt 6.2.7) werden zunächst *Unit-* und grundlegende Integration-Tests vollzogen. In der späteren *Testbedanwendung* (Abschnitt 6.2.8) liegt dann der Fokus auf Integration Tests, welche mit realer Hardware vollzogen werden. In beiden Phasen werden *Acceptance Test Reports* (Abschnitt 6.1.1) erstellt. Sowohl die Durchführung der eigentlichen Tests als auch die Dokumentation des Vorgehens und der Ergebnisse muss durch projekt- oder organisationsspezifische Regelungen festgelegt werden um so eine Einheitlichkeit und Transparenz zu gewährleisten. Diese dient der Einhaltung und Überprüfbarkeit der anzuwendenden Standards und Normen und unterstützt so eine etwaige Zertifizierung. Aus diesem Grund wird diese Aktivität im Kontext der organisatorischen Artefakte betrachtet. Auch wenn es vor allem Entwickler sind, die diese Aktivität ausführen um den geschaffenen Code und die zur Entstehung dessen definierten Modelle und Artefakte zu überprüfen, ist das Resultat ein für das Projektmanagement relevantes. Somit handelt es sich bei dieser Aktivität

um einen Brückenschlag zwischen der internen (technischen) Welt des Auftragnehmers und der öffentlichen Welt der Standards, Normen und Anforderungen des Kunden. Idealerweise erfolgen die Tests weitestgehend automatisiert um so ökonomisch ein breites Spektrum möglicher Fehlerfälle abbilden zu können und ein einheitliches und nachvollziehbares Vorgehen im Rahmen der Tests sicherzustellen.

Freigeben

Die Freigabe erfolgt durch das Entwickler- und Expertenteam in der Phase der *Testbedanwendung* (Abschnitt 6.2.8). Sind alle Tests erfolgreich verlaufen, werden die hier geschaffenen Berichte und dokumentierten Resultate durch das Team validiert und freigegeben. Erst nach einer Freigabe durch alle Prozessbeteiligten wird der geschaffene Code als *Inkrement* (Abschnitt 6.1.1) betrachtet und durch den *Entwickler* welcher mit der Implementierung betraut war in die für das nächste *Release* vorbereitete Softwareversion eingepflegt.

Veröffentlichen

Der Abschluss eines Projektes oder einer Projektphase ist durch die Veröffentlichung der Software und somit die Erstellung eines *Releases* (Abschnitt 6.1.1) geprägt. Diese erfolgt im Zuge des *Deployments* (Abschnitt 6.2.10), welches durch das Aufspielen der Software auf die zu steuernden Systeme geprägt ist. Die hierbei zu vollziehenden Tätigkeiten richten sich nach der vorhandenen Infrastruktur und den technischen Besonderheiten der Leitsysteme. Ist es möglich die Firmware Over the Air upzudaten, so kann der Prozess vollautomatisiert geschehen. Handelt es sich um ein geschlossenes System, welches vor Ort aktualisiert werden muss, kann die Veröffentlichung einen erheblichen manuellen Aufwand beinhalten.

6.3.2. Aktivitäten zur Erarbeitung technischer Artefakte

Analog der Aktivitäten zur Erarbeitung der organisatorischen Artefakte dienen die Aktivitäten zur Erarbeitung der technischen Artefakte originär dem Design und der Erstellung des Agentensystems sowie der Einhaltung der regulatorischen Rahmenbedingungen. Sie sind darauf ausgerichtet alle technischen Details des gesamten Agentensystems in einer effizienten Form zu erarbeiten und in den dafür vorgesehenen Artefakten zu aggregieren. Tabelle 6.4 zeigt in welcher Projektphase welche Aktivität durchgeführt wird, welche Stakeholder an dieser beteiligt sind und welche Artefakte dabei geschaffen werden.

Anforderungsanalyse

Die *Anforderungsanalyse* wird im Rahmen der *Initialen Anforderungsermittlung* (Abschnitt 6.2.1) durchgeführt und hat zum Ziel die *Systemanforderungen* (Abschnitt 6.1.2), welche an das Gesamtsystem gestellt werden, zu erarbeiten. In diese Aktivität sind noch keine *Entwickler* oder *Experten* eingebunden, da es sich lediglich um eine sehr abstrakte Betrachtung der zu entwickelnden Lösung handelt. Die *Anforderungsanalyse* orientiert sich stark am *User Story Mapping*, konzentriert sich dabei jedoch auf die technischen Rahmenbedingungen und gegebenenfalls bereits formulierten technischen Details.

Aktivität	Artefakte	Projektphase	Beteiligte
Anforderungs-analyse	Systemanforderungen	Initiale Anforderungs-ermittlung	Kunde, Projektleiter
Umgebungs-modellierung	Energie-Options-Modell Ressourcenmodell	Vorplanung	Kunde, Koordinator Experte, Entwickler
Zieldefinition	Zielmodell	Anforderungs-ermittlung	Koordinator Experte, Entwickler
Ableiten der Fähigkeiten	Fähigkeitsmodell	Anforderungs-ermittlung	Koordinator Experte, Entwickler
Rollen-verteilung	Rollenmodell	Anforderungs-ermittlung	Koordinator Experte, Entwickler
Interaktions-analyse	Interaktionsmodell	Basisplanung	Koordinator Entwickler
Grobentwurf	Agentenmodell	Ausführungsplanung	Koordinator Experte, Entwickler
Feinentwurf	Agenten Struktur Modell	Implementierung	Entwickler
Umsetzung	Code	Implementierung	Entwickler

Tabelle 6.4.: Technische Aktivitäten des 2DECS Ansatzes und ihre Zusammenhänge

Umgebungsmodellierung

Im Zuge der *Vorplanung* (Abschnitt 6.2.2) definieren das Entwicklungs- und Expertenteam gemeinsam mit dem *Kunden* unter etwaiger Moderation des *Koordinators* zunächst das Energie-Options-Modell (Abschnitt 6.1.2) und anschließend das *Ressourcenmodell* (Abschnitt 6.1.2). Das Vorgehen ist hierbei mehrstufig. Zunächst werden gemeinsam mit dem *Kunden* die groben Rahmenbedingungen definiert und anschließend in Detailworkshops einzelne Aspekte wie Kommunikationsstrecken und Protokolle, Anlagentypen oder Physikalische Grundlagen diskutiert. Sind diese Details erarbeitet, sollte das gesamte Team gemeinsam mit dem *Kunden* das so geschaffene Modell validieren. Dieser letzte Schritt ist aufgrund der hohen Auswirkungen des Umgebungsmodells auf den weiteren Entwicklungsprozess von großer Relevanz. Änderungen an den hier erarbeiten Artefakten sind aufgrund der Einflüsse auf das Gesamtprojekt und Rückwirkungen auf bereits erschaffene Artefakte zu einem späteren Zeitpunkt im Entwicklungsprozess nicht vorgesehen.

Zieldefinition

Auf Basis der *Systemanforderungen* und des zuvor erstellen Umgebungsmodells wird im Rahmen der *Zieldefinition* unter Leitung des *Koordinators* von *Experten* und *Entwicklern* das *Zielmodell* (Abschnitt 6.1.2) des Agentensystems erstellt. In der Phase der *Anforderungsermittlung* (Abschnitt 6.2.3) werden so parallel zur organisatorischen Übersetzung der *User Stories* (Abschnitt 6.3.1) die Ziele aus den Anforderungen des *Kunden* abgeleitet. Dies geschieht ebenso im Rahmen von Workshops, wodurch Synergien genutzt und gegebenenfalls sogar beide Aktivitäten in der selben Veranstaltung durchgeführt werden. Es ist dabei jedoch zu beachten, dass im Gegensatz zur vorgenannten Übersetzung der *User Stories* die *Zieldefinition* den Fokus vermehrt auf technische Ziele legt.

Ableiten der Fähigkeiten

Im weiteren Verlauf der *Anforderungsermittlung* (Abschnitt 6.2.3) werden auf Basis des zuvor erstellen *Zielmodells* die zur Erreichung der Ziele notwendigen Fähigkeiten abgeleitet. Diese Aktivität wird primär durch die *Entwickler* des Systems durchgeführt. Es können jedoch weitere *Experten* hinzugezogen werden, sollte in spezifischen Fachfragen die Notwendig bestehen. Auch ist eine Moderierung durch den *Koordinator* bei entsprechendem Projektumfang denkbar. Ziel ist es das *Fähigkeitsmodell* (Abschnitt 6.1.2) zu erstellen. Die Analyse erfolgt in Form eines Brainstormings mit anschließender Strukturierung und gemeinsamer Überprüfung der Ergebnisse. Wichtig ist es mit den definierten Fähigkeiten alle Ziele erreichen zu können.

Rollenverteilung

Die *Rollenverteilung* erfolgt ebenenfalls in der Phase der *Anforderungsermittlung* (Abschnitt 6.2.3). Im Zuge dieser Tätigkeit werden durch das Entwicklerteam die Fähigkeiten in Rollen gekapselt. Diese Zuordnung erfolgt auf Basis der Zweckdienlichkeit im Hinblick auf die zu erreichenden Ziele. Bei Fragen zu Themen des Datenschutzes können *Experten* hinzugezogen werden. Auch hier kann eine Moderation durch den *Koordinator* angedacht werden. Die Erstellung des *Rollenmodells* 6.1.2 ist somit, ebenso wie das *Ableiten der Fähigkeiten* eine primär strukturierende Tätigkeit.

Interaktionsanalyse

Nach der grundlegenden Gliederung der Ziele und den Möglichkeiten diese zu erreichen werden im Rahmen der *Interaktionsanalyse* die Kommunikationspfade des Agentensystems gestaltet. Das Entwicklungsteam erstellt das *Interaktionsmodell* (Siehe 6.1.2), welches alle relevanten Interaktionen, sowohl innerhalb des Agentensystems als auch mit externen Kommunikationspartner, abbildet und spezifiziert. Sollte das Projekt einen großen Umfang besitzen, kann diese Tätigkeit auf mehrere Teams aufgeteilt und durch den *Koordinator* gesteuert werden. Die *Interaktionsanalyse* ist die Haupttätigkeit in der Phase der *Basisplanung* (Siehe 6.2.4). Da die Kommunikation eine entscheidende Rolle in Agentensystemen einnimmt, ist diese Aktivität von besonders erfahrenem Personal durchzuführen.

Grobentwurf

Der *Grobentwurf* hat zum Ziel das *Agentenmodell* (Abschnitt 6.1.2) zu erstellen. Es handelt sich dabei um die primäre Aktivität der *Ausführungsplanung* (Abschnitt 6.2.5), welche aus einem gemischten Team aus *Entwicklern* und *Experten* unter Leitung des *Koordinators* durchgeführt wird. Die Hinzunahme der *Experten* ist aufgrund der zu beachtenden rechtlichen und strukturellen Rahmenbedingungen notwendig. So soll verhindert werden, dass Agenten und somit steuernde Elemente des zu schaffenden Systems Aufgaben übertragen bekommen, welche sie entweder nicht ausführen dürfen oder können. Als mögliche Hinderungsgründe seien zum Beispiel Fragen des Datenschutzes, der gesetzlich zugeteilten Rollen im Energiesystem oder der Übertragungswege und deren Verschlüsselungsmöglichkeiten genannt.

Feinentwurf

Als vorbereitende Tätigkeit für die anschließende Umsetzung werden im Rahmen des *Feinentwurfs* durch die *Entwickler* die individuellen *Agenten Struktur Modell*e (Abschnitt 6.1.2) erstellt. In der Phase der *Implementierung* (Abschnitt 6.2.6) bereiten sie so die eigentliche Umsetzung des Agenten in *Code* vor. Im Zuge des *Feinentwurfs* werden die zuvor definierten Fähigkeiten und den individuellen Agenten in aggregierter Form zugeordneten Rollen in konkret zu implementierende Funktionen übertragen. Dies geschieht zweckmäßigerweise in UML-Diagrammen. Wichtig ist ein Austausch zwischen den *Entwicklern* um Fähigkeiten oder sogar Rollen, welche von mehreren Agenten geteilt werden nicht redundant zu implementieren. Hierbei ist ein adäquater Abstraktionsgrad zu finden, welcher eine Allokation der Funktionen zu den unterschiedlichen Agententypen erlaubt.

Umsetzung

Als letzte Tätigkeit des technischen Prozesses wird im Rahmen der *Umsetzung* durch die *Entwickler* der eigentliche Programmcode erstellt (Siehe Abschnitt 6.1.2). Die Programmierung kann dabei individuell oder in Paaren erfolgen. Hierbei ist in Abhängigkeit der Sicherheitsrelevanz der zu implementierenden Elemente eine regelmäßige Verifikation des implementierten *Codes* einzuplanen.

6.4. Rollen

Das *Rollenmodell* des 2DECS Ansatzes ist, verglichen mit klassischen Ansätzen, bewusst leichtgewichtig gehalten um so ein der Projektorganisation angemessenes *Tailoring* zu ermöglichen. Es werden lediglich fünf Rollen vorgesehen, welche in einer flachen Hierarchie stehen. An der Spitze steht dabei der *Projektleiter*, welchem ein *Koordinator* untergeordnet ist. In kleinen Projekten und Teams kann diese Aufgabe von der selben Person ausgefüllt werden. Es ist in diesem Kontext festzuhalten, dass es sich bei den Rollen allgemein nicht um Personen handelt. Eine Rolle kann von einer Vielzahl an Personen wahrgenommen werden, ebenso wie eine Person mehrere Rollen ausfüllen kann. Die (arbeitende) Basis des Projektes bilden die *Entwickler* und *Experten*. In diese Struktur, jedoch nicht in die Hierarchie, eingegliedert, ist die Rolle des *Kunden*.
Abbildung 6.20 zeigt die Rollen im Vergleich zum noch leichtgewichtigeren SCRUM, dem etwas umfangreicheren XP und einem Auszug der im V-Modell definierten Rollen. Mittels dieser Grafik soll auch der mit 2DECS eingeschlagene Mittelweg verdeutlicht werden. Sollte im Zuge eines Projektes die Notwendigkeit weiterer Rollen, wie beispielsweise die expliziter Tester oder von Qualitätsmanagern, notwendig werden, so lassen sich diese in den Prozess integrieren.

6.4.1. Kunde

Die Rolle des *Kunden* ist in 2DECS, ebenso wie in anderen agilen Prozessen, eine besondere. Entgegen klassischen Projekt- und Entwicklungsansätzen wird der *Kunde* im laufenden Prozess regelmäßig gefordert. So ist es ihm möglich regelmäßig Kurskorrekturen vorzunehmen und Feedback zum entstehenden Produkt zu geben. Dies bedeutet aber auch, dass der *Kunde* entsprechenden Sachverstand und soziale Fähigkeiten besitzen muss. Der *Kunde* kann aus

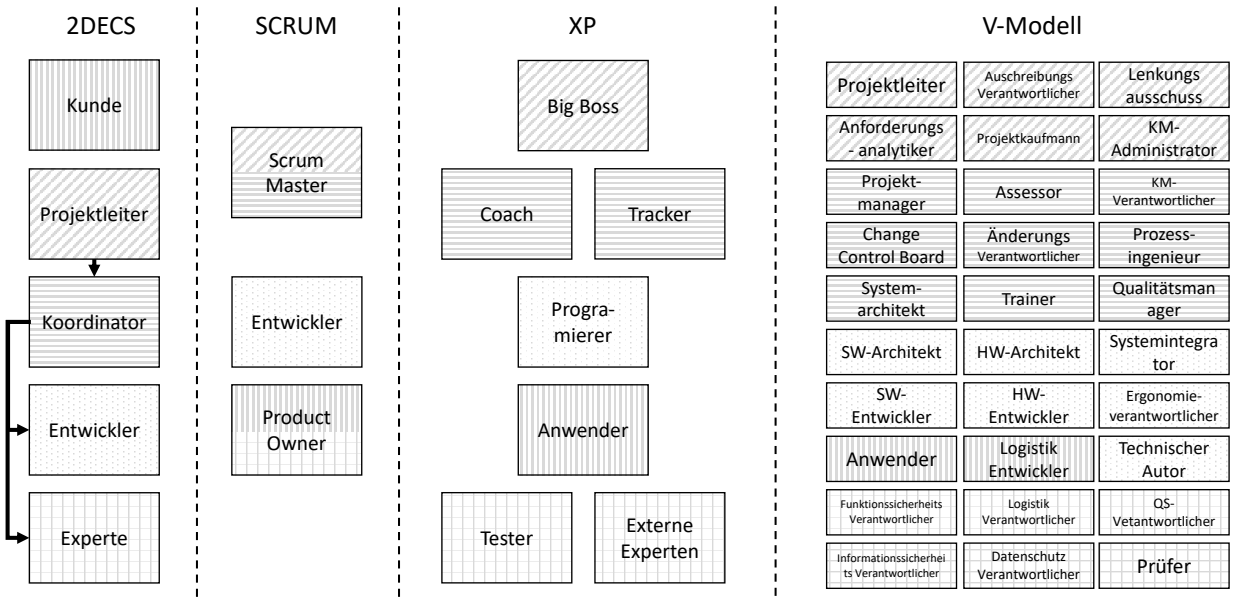

Abbildung 6.20.: Vergleich der Rollenmodelle von SCRUM, XP, V-Modell und 2DECS

einer Gruppe beauftragter Personen bestehen, welche nicht ausschließlich durch den späteren Anwender gestellt werden müssen. So ist es denkbar externe Berater einzubinden um die Qualität des Prozess unabhängig zu monitoren oder auch als Ideengeber zu fungieren. Im Rahmen eines mittels 2DECS umgesetzten Entwicklungsprojektes sollte der *Kunde* idealerweise an Expertise gewinnen und das von ihm später genutzte Produkt besser verstehen als es mit einem klassischen Entwicklungsansatz der Fall wäre.

6.4.2. Projektleiter

Die Rolle des *Projektleiters* bedarf wirtschaftlicher, rechtlicher und sozialer Expertise, da es sich bei diesem um den ersten und direkten Ansprechpartner des *Kunden* handelt. Es ist die *Projektleitung*, die den Projektumfang, den Preis und die organisatorische Ausgestaltung verhandelt und dafür gegenüber allen Stakeholdern verantwortlich zeichnet. Darüber hinaus ist die *Projektleitung* den vornehmlich fachlichen Rollen des *Koordinators*, *Entwicklers* und *Experten* vorgesetzt. Technischer Sachverstand ist im diesem Falle wünschenswert, jedoch nicht zwingend erforderlich. Die in dieser Rolle beschriebenen Kompetenzen können problemlos auf eine Gruppe aufgeteilt werden.

6.4.3. Koordinator

Der *Koordinator* stellt ein vermittelndes Element sowohl zwischen der *Projektleitung* und dem Entwicklungsteam als auch, bei Bedarf und nach Rücksprache mit dem Projektleiter, zwischen dem *Kunden* und den *Entwicklern* dar. Darüber hinaus ist er für die Einhaltung der vereinbarten Prozesse und Regeln durch alle Stakeholder verantwortlich. Seine Arbeit besteht darin alle Interessenten in diesen zu schulen und auf deren Einhaltung zu drängen. Seine primäre Fokusgruppe sind dabei die *Entwickler* und *Experten*, welche einen Großteil der Aufgaben wahrnehmen müssen und am stärksten von etwaigen Prozessfehlern betroffen sind. Der *Koordinator* braucht somit zum einen gute soziale Fähigkeiten, zum anderen aber

auch ein großes Prozess- und Strukturwissen. Idealerweise sind diese Fähigkeiten mit einer hohen technischen Expertise und Seniorität gepaart, um so die Herausforderungen der ihm unterstellten *Experten* nachvollziehen zu können.

6.4.4. Entwickler

Die primäre Aufgabe der *Entwickler* ist das Design und die Umsetzung des Steuerungssystems. Es muss sich daher um Experten ihres Fachs handeln, welche idealerweise, neben dem Software-Wissen, auch Domänenwissen in Bezug auf das zu steuernde System besitzen. *Entwickler* sind in das Gros der Aktivitäten und Phasen eingebunden und bilden faktisch das Fundament des Projektes. Trotz dieser starken Verankerung sollte durch den *Koordinator* bzw. die *Projektleitung* darauf geachtet werden, dass einzelne Personen nicht unabdingbar werden. Die durch die *Entwickler* geschaffenen Artefakte und das erarbeitete Wissen sollte stets allen *Entwicklern* zugänglich und verständlich sein um so etwaige Ausfälle unkompliziert kompensieren zu können.

6.4.5. Experte

Die Rolle des *Experten* wird durch Fachleute mit expliziten Domänenwissen ausgefüllt. Diese werden bei spezifischen technischen, rechtlichen, organisatorischen oder sozialen Fragestellungen oder generellen Fragen, welche einen weiteren Überblick über diese Themenbereiche benötigen ,herangezogen. *Experten* müssen nicht zwingenderweise konstante Mitglieder des Teams sein und können auch von außerhalb der mit dem Projekt betrauten Organisation stammen.

6.5. Werkzeuge

Da das Angebot an Werkzeugen und die daran gestellten Ansprüche einem konstanten Wandel unterliegen, sind die folgenden Ausführungen abstrakter Natur. 2DECS kann durchgängig mit einem klassischen „Stift und Papier“-Ansatz praktiziert werden. Genauso können aktuell verfügbare Werkzeuge wie die von ATLASSIAN bereitgestellte *JIRA* [6] Suite, oder auch eine Kombination unterschiedlicher Angebote wie die Code Verwaltung *GitHub* [7], das KanBan-Werkzeug *Trello* [8] und Werkzeuge wie *Postman*[9] zum Testen und Dokumentieren der *Application Programming Interfaces* (API) in Kombination mit Continuos Integration Tools wie *Jenkins*[10] genutzt werden.
Unabhängig davon, für welches Werkzeug oder welche Werkzeugkette man sich entscheidet, das ausschlaggebende Merkmal sollte stets die Akzeptanz im Entwicklungsteam sein. Wird das Werkzeug nicht umfassend genutzt oder entstehen gar Informationslücken, so verkehrt sich dessen Nutzen in das Gegenteil und wird zum Vehikel für Fehler und Nachlässigkeiten, welche die Fragen der Sicherheit, Standard-Compliance und damit die Produktqualität

[6] Siehe: https://de.atlassian.com/software/jira
[7] Siehe: https://github.com
[8] Siehe: https://trello.com
[9] Siehe: https://www.getpostman.com
[10] Siehe: https://jenkins.io

insgesamt negativ beeinflussen. Positiv auf die Akzeptanz wirken sich in der Regel eine durchgängige Nutzung der erarbeiteten Informationen und ein geringer Pflegeaufwand aus. Eine Vielzahl der Werkzeuge, welche aktuell auf den Markt gebracht werden, bieten die Möglichkeit der wechselseitigen Verknüpfung, was eben diesem Gedanken der kontinuierlichen Nutzung von Informationen unterstreicht. Nachteile entstehen immer dann aus diesen Annehmlichkeiten, wenn Fragen des Datenschutzes oder Verortung der Daten aufgeworfen werden. Müssen die Werkzeuge beispielsweise in einem privaten Netz auf einem deutschen Server gehostet werden, so reduziert sich die Anzahl und Qualität der verfügbaren Werkzeuge bei gleichzeitiger Kosten- und Aufwandssteigerung, erheblich.
Es ist daher wichtig diese Fragen bereits im Rahmen der *Initialen Anforderungsermittlung* zu thematisieren und die damit verbundenen Faktoren Kosten, Zeit und Qualität zu adressieren.

7. Anwendung und Evaluation

Nach Hevner et al. [HMPR04] ist neben dem Design eines Entwicklungsansatzes, der Relevanz des damit gelösten Problems und dem Beitrag zur wissenschaftlichen Diskussion insbesondere die Evaluation von besonderer Relevanz für den Erfolg eines solchen Konzeptes. Daher wurde die Evaluation des 2DECS Ansatzes zweigeteilt um so zum Einen die praktische Anwendbarkeit und logische Konsistenz zu erproben und zum Anderen die Relevanz und Vorteilhaftigkeit für die Energiedomäne abzuschätzen. Durch dieses Vorgehen soll nach Hevner et al. zunächst eine „beobachtende“ Feldstudie und anschließend eine „beschreibende“ Untersuchung im Sinne der „kundigen Akteure“ in Form von Experteninterviews durchgeführt werden. Hierfür wurde 2DECS zunächst in drei Fallstudien genutzt um Software für Energiesysteme zu entwickeln. Nach Abschluss der Projekte wurden Interviews mit Experten aus den Domänen Energietechnik und Softwareentwicklung geführt. Die Folgenden Abschnitte stellen die Verfahrensweisen, Ergebnisse und deren Interpretationen dar.

7.1. Fallstudien

Aufgrund der Vielfalt möglicher Anwendungsfälle wurde 2DECS bewusst in Projekten mit unterschiedlichen Charakteristika erprobt. Dabei dienten die folgenden Leitfragen als organisatorischer Rahmen der Studie:

- *Sind die Artefakte logisch konsistent?* - Ein Ziel der praktischen Erprobung war es die durchgängige Anwendung und den konsistenten Aufbau der Artefakte zu untersuchen.
- *Sind die Phasen sachdienlich dimensioniert?* - Neben der inhaltlichen Erarbeitung der Artefakte sollte die Gliederung und der Umfang der Projektphasen geprüft werden.
- *Werden die Rollen wahrgenommen?* - Aufgrund der unterschiedlichen Teams der Fallstudien sollte geprüft werden, wie die definierten Rollen in den Projekten ausgefüllt werden.
- *Ist eine projektspezifische Anpassung des Ansatzes möglich? - Tailoring* erlaubt die flexible Nutzung eines Prozesses über die Grenzen der ursprügnlichen Konzeption hinaus. Aus diesem Grund soll es in einer praktischen Erprobung, welche nicht prognostizierte Anforderungen hervorbringen kann, untersucht werden.

Die Projektstadien, -umfänge sowie die unterschiedlichen Bedürfnisse der Projektteams ermöglichten es den Ansatz in unterschiedliche Projektphasen einzubringen. Tabelle 7.1 gibt eine Übersicht über die in den Projekten evaluierten Phasen des 2DECS Konzeptes. Ein „X“ symbolisiert dabei, dass diese Phase in der Fallstudie bearbeitet wurde.

Phase	Agent.HyGrid 2016	Agent.HyGrid 2018	OEEX
Initiale Anforderungsermittlung		X	X
Vorplanung		X	X
Anforderungsermittlung		X	
Basisplanung		X	X
Ausführungsplanung		X	X
Implementierung	X		X
Simulation	X		X
Testbedanwendung	X		
Debriefing			X
Deployment			X
Lifecycle Management			X

Tabelle 7.1.: Übersicht der in den Fallstudien evaluierten Phasen

7.1.1. Vorstellung der Anwendungsfälle

Agent.HyGrid - 2016

Das *Agent.HyGrid - 2016* Projekt ist ein Teilprojekt des durch das Bundesministerium für Wirtschaft und Energie geförderten Forschungsprojektes Agent.HyGrid. An diesem Projekt arbeiteten zeitweise bis zu sieben Mitarbeiter. Ziel dieses Projektes war es die Skalierbarkeit der im Rahmen des Forschungsprojektes genutzten Simulationsplattform zu evaluieren und deren Zusammenspiel mit realen Datenquellen zu erproben. Hierfür wurde das Niederspannungsnetz einer ländlich geprägten Ortschaft, welche von Landwirtschaftsbetrieben mit alternierenden Lastspitzen und fluktuierender Solarstromerzeugung geprägt ist, mittels Energie-Agenten abgebildet. In der als Prototyp herangezogenen Ortschaft wurde bereits ein System zur Netzzustandsüberwachung installiert. Von diesem konnten reale Messwerte und Daten bezogen werden um sie als externe Datenquellen der Simulation zu nutzen. Die Ergebnisse der Simulation konnten aufgrund der guten Datenbasis dann mit den realen Netzzuständen verglichen werden. Abbildung 7.1 zeigt den Aufbau des Systems. Die durchgezogenen Linien stehen dabei für das Energienetz, Punkte für individuelle Knoten darin, wobei farbige Knoten überwacht werden und graue Knoten keine Hardware zur Überwachung besitzen. Gestrichelte Linien symbolisieren eine Datenverbindung, die damit verbundenen Kästchen repräsentieren die individuellen Agenten.
Das Agentensystem wurde in JADE implementiert und umfasst die vier folgenden Agententypen:

- *Sensor Agent* - Dieser Agent dient als Schnittstelle zu Messgeräten, welche an individuellen Strängen netzspezifische Daten wie Spannung, Stromstärke, Frequenz, Phasenlage, etc. erfassen. Die so erfassten Daten kommuniziert er an den *Grid State Agenten*.

- *PV Agent* - Der *PV Agent* dient der Erfassung von Parametern der mit ihm verbundenen Photovoltaikanlagen sowie des lokalen Netzzustandes. Darüber hinaus ist er in der Lage die verbundene PV Anlage herunterzuregeln, beziehungsweise vom Netz zu trennen. Diese Entscheidung trifft der Agent auf Basis des lokalen Netzzustandes, oder auch

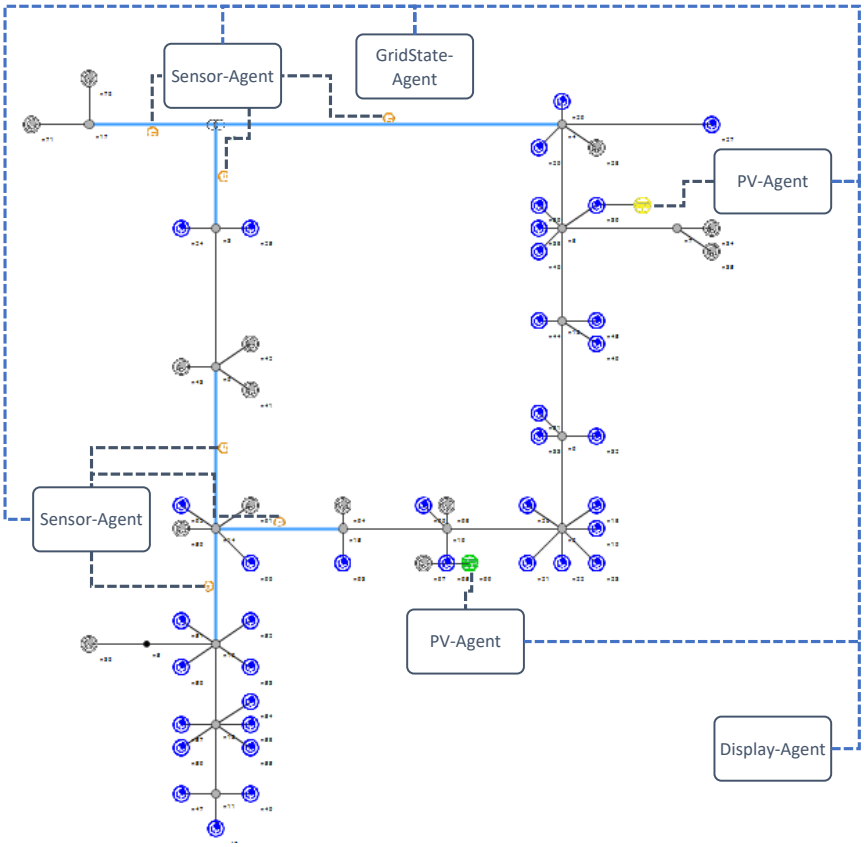

Abbildung 7.1.: Schematische Struktur des Agent.HyGrid - 2016 Projektes

des durch den *Grid State Agenten* errechneten globalen Netzzustandes. Der *PV Agent* informiert den *Grid State Agenten* über die von ihm erfassten Werte sowie die von ihm getroffenen Entscheidungen. Darüber hinaus befolgt er alle durch den *Grid State Agenten* erteilten Regelbefehle.

- *Grid State Agent* - Aufgabe dieses Agenten ist es auf Basis der ihm zugespielten Daten den globalen Netzzustand zu berechnen. Sollten hierbei Grenzwerte verletzt werden, erarbeitet der *Grid State Agent* eine Lösungsstrategie und verteilt daraus abgeleitete Regelungsbefehle an die angeschlossenen *PV Agenten*.

- *Display Agent* - Dieser Agent dient der Visualisierung der Simulationsergebnisse. Er ist Bestandteil des Simulationsframeworks.

Die an dieses System gestellten Sicherheitsanforderungen waren niedrig, da es sich lediglich um eine Simulation handelte, welche keinen Einfluss auf reale Netzkomponenten nehmen konnte. Es wurde auf die Implementierung standardisierter Kommunikationsprotokolle verzichtet, da ein projektspezifischer Protokoll-Stack Anwendung finden sollte. Dieser baut auf dem Energie-Agenten Konzept auf und nutzt die darin definierten Energie-Optionsmodelle. Diese wurden dann mittels der in JADE implementierte Agent Communication Language (ACL) transportiert.
Da das Projekt zum Testzeitpunkt bereits vorangeschritten war, war es möglich direkt in das Umsetzungsstadium einzusteigen und die Phasen Implementierung, Simulation und Testbedanwendung zu evaluieren. Hierdurch mussten die Eingangsartefakte wie das Agenten- oder

Rollenmodell auf Basis der vorhandenen Planungsdaten nachträglich erstellt werden. Die dadurch induzierten Mängel konnten jedoch nicht systematisch erfasst oder evaluiert werden.

Agent.HyGrid - 2018

Das *Agent.HyGrid - 2018* Projekt ist ebenfalls ein Teilprojekt des durch das Bundesministerium für Wirtschaft und Energie geförderten Forschungsprojektes Agent.HyGrid mit sechs Mitarbeitern. In diesem Projekt soll das Niederspannungsnetz eines Universitätscampus mittels Energie-Agenten überwacht und geregelt werden. Der Fokus liegt dabei auf der Integration realer Hardware. Hierfür werden vor Ort Industrie-PCs betrieben, auf welchen die Agenten ausgeführt werden. Diese koordinieren dann sowohl die Erzeugung als auch den Verbrauch und Transport der Energie. Als Energiequellen stehen eine große Photovoltaikanlage sowie ein Anschluss an das Mittelspannungsnetz zur Verfügung. Als Energiesenken werden Elektrofahrzeuge sowie eine regelbare Wärmepumpe dienen. Abbildung 7.2 zeigt den Aufbau des Netzes. Die Verbindungslinien repräsentieren das vor Ort befindliche Elektrizitätsnetz, welches zweier Sterne entspricht. Einzelne Gebäude, beziehungsweise größere Verbrauchseinheiten werden in Distrikten zusammengefasst, welche durch gestrichelte Linien kenntlich gemacht werden. In diesen Distrikten existiert eine hierarchische Struktur, der jeweils ein *District Agent* vorsteht. Agenten werden durch geschlossene Kästen dargestellt. Aus Komplexitätsgründen wurden die Kommunikationsverbindungen nicht abgebildet, da grundsätzlich alle Agenten miteinander kommunizieren können.

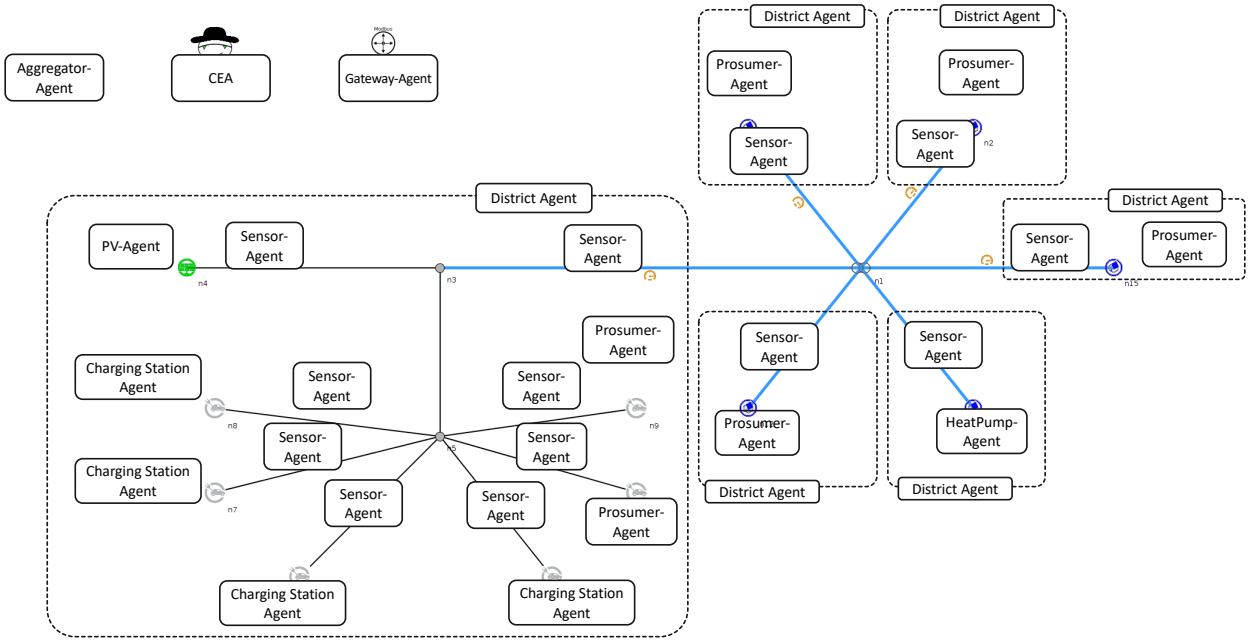

Abbildung 7.2.: Schematische Struktur des Agent.HyGrid - 2018 Projektes

Das Agentensystem wurde in JADE implementiert und umfasst die neun folgenden Agententypen:

- *Aggregator Agent* - Als Grundlage seiner Arbeit erhält der *Aggregator Agent* von den ihm untergeordneten *District Agenten* die lokalen Netzzustände. Aus diesen errechnet

er den globalen Netzzustand und veranlasst bei Grenzwertverletzungen korrigierende Steuerbefehle.

- *District Agent* - Die *District Agenten* sind primär Aggregatoren der ihnen unterlagerten Informationsknoten. Die so gesammelten Informationen reichen Sie an den *Aggregator Agenten* weiter. Befindet sich das Netz in einem unkritischen Zustand, können sie den ihnen unterlagerten Bereich autonom steuern.

- *CEA Agent* - Aufgabe dieses Agenten ist es die Prozesse und Zustände des Systems zu visualisieren.

- *Gateway Agent* - Der *Gateway Agent* dient der Anbindung des Systems an externe Softwarelösungen.

- *Charging Station Agent* - Dieser Agent überwacht und steuert den Ladevorgang des an ihn angeschlossenen Elektrofahrzeuges. Die von ihm erhobenen Daten leitet er an den *District Agent* weiter. Von diesem erhält er auch die von ihm umzusetzenden Steuerentscheidungen. Im Falle eines Ausfalls dieser vorgesetzten Instanz kann der *Charging Station Agent* auf Basis des lokalen Netzzustandes autonom den Ladevorgang regeln.

- *HeatPump Agent* - Ähnlich dem *Charging Station Agenten* ist dieser Agent für die Überwachung und Steuerung der ihm unterstellten Wärmeheizpumpe verantwortlich und besitzt somit die selben Funktionalitäten, jedoch mit einer auf die Bereitstellung von Wärme ausgerichteten Logik.

- *PV Agent* - Der *PV Agent* überwacht die ihm unterstellte Photovoltaikanlage. Die hierbei erfassten Werte übermittelt er an den *District Agent*. Es ist dem *PV Agenten* möglich den Inverter der Solaranlage zu regeln. Die Sollvorgaben hierfür erhält er vom *District Agent*. Im Zuge eines Ausfalls des *District Agent* ist er in der Lage auf Basis der lokal verfügbaren Informationen die Solaranlage autonom zu regeln.

- *Prosumer Agent* - Ähnlich den vorgenannten Agenten überwacht und steuert der *Prosumer Agent* die mit ihm verbundenen Verbraucher auf Basis der Vorgaben des *District Agenten*, beziehungsweise im Fehlerfall autonom.

- *Sensor Agent* - Die *Sensor Agenten* dienen als erweiterte Datenlieferanten für die dezentrale, wie auch die zentrale Regelung. Ihre Messwerte können von den unterschiedlichen Agenten abgerufen werden.

Die an dieses System gestellten Sicherheitsanforderungen sind relativ hoch, da es mit realer Hardware interagiert. Die Kommunikation mit den existierenden Vor-Ort System findet mittels standardisierter Protokolle statt. Die Inter-Agenten Kommunikation wird auf Basis des bereits im Projekt *Agent.HyGrid - 2016* genutzten Protokoll-Stacks realisiert.
Im Zuge des Projektes wurden alle Phasen von der Initialen Anforderungsermittlung bis hin zur Ausführungsplanung evaluiert.

OEEX Mobile Power

OEEX Mobile Power ist ein Nebenprojekt der OEEX Plattform [11], an welchem sich drei Personen beteiligten. Ziel dieses Projektes war es im Rahmen eines praktischen Anwendungsfalls die Verknüpfung einer Service-Orientierten Plattform mit einem Agentensystem zu demonstrieren. Die Agenten wurden in diesem Projekt in der Cloud ausgeführt und erhielten Informationen von dezentralen Knoten. Hierfür wurden mobile Powerbanks, welche durch ebenfalls mobile Solarpaneele geladen wurden, mit Mobiltelefonen verknüpft. Bei der für dieses Projekt genutzten Hardware handelte es sich um die marktverfügbaren Produkte *Kalhuohfummi* und *Maroshi* aus dem Hause Changers [12]. Die Ausführungsumgebung der Agenten war eine *Amazon AWS EC2* Instanz [13]. Abbildung 7.3 zeigt den grundlegenden Aufbau des Systems.

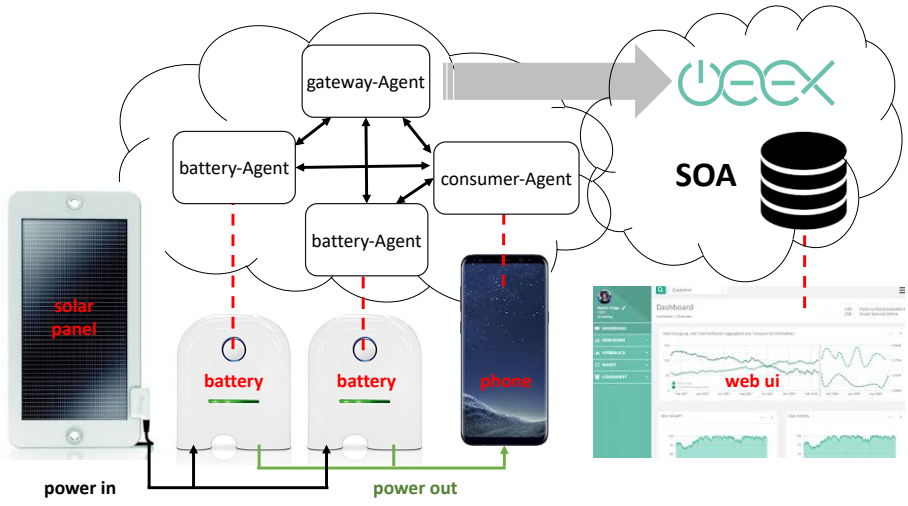

Abbildung 7.3.: Schematische Struktur des OEEX Mobile Power Projektes

Das Agentensystem wurde in JADE implementiert und umfasst die drei folgenden Agententypen:

- *Battery Agent* - Dieser Agent erhält in regelmäßigen Abständen Informationen über den Füllstand der mit ihm verknüpften Powerbank. Darüber hinaus bezieht er eine Wetterprognose. Ist der Füllstand hoch, die Wetterprognose positiv, oder beides der Fall, so ist er bereit Energie abzugeben. Sollte der Füllstand niedrig und die Prognose negativ sein, so stellt er keine Energie bereit. Darüber hinaus kommuniziert der *Battery Agent* regelmäßig den Zustand der von ihm überwachten Powerbank an den *Gateway Agenten*.

- *Consumer Agent* - Der *Consumer Agent* erhält in regelmäßigen Abständen Informationen über den Ladezustand eines Mobiltelefons. Ist dieser niedrig, so kontaktiert er die *Battery Agenten* und erfragt eine mögliche Ladung. Erhält er eine positive Antwort, teilt er dem Nutzer mit, welche Powerbank zum Laden bereit steht. Darüber hinaus kommuniziert der *Consumer Agent* regelmäßig den Zustand des von ihm überwachten Mobiltelefons an den *Gateway Agenten*.

[11] Siehe: http://www.oeex.org
[12] Siehe: https://changers.com/de
[13] Siehe: https://aws.amazon.com/de/ec2/

- *Gateway Agent* - Der *Gateway Agent* stellt die Schnittstelle zur bestehenden OEEX Infrastruktur dar. Er fasst die an ihn übermittelten Messwerte zusammen und kommuniziert sie an den *OEEX Gateway Service*.

Die an dieses System gestellten Sicherheitsanforderungen waren niedrig, da es ausschließlich im geschlossenen OEEX-Virtual Private Network (VPN) ausgeführt wurde. Alle Kommunikationsverbindungen waren somit vor externem Einfluss geschützt. Aufgrund des Prototypencharacters des Projektes wurde auf die Implementierung standardisierter Kommunikationsprotokolle verzichtet und die bereitstehenden Lösungen wie Beispielsweise der in JADE implementierte ACL Stack genutzt. Lediglich die Kommunikation mit dem OEEX Gateway erfolgte mittels des proprietären OEEX-Ready Protokolls, für welches keine öffentliche Dokumentation existiert.
Im Rahmen des Projektes wurden die Phasen der 2DECS Zyklus von der Initialen Anforderungsermittlung bis zum Lifecycle Management evaluiert. Dabei wurden die Phasen Anforderungsermittlung und Testbedanwendung nur rudimentär oder gar nicht bearbeitet.

7.1.2. Ergebnisse der Fallstudien

Im Folgenden sollen Auszüge der in den Fallstudien erarbeiteten Ergebnisse vorgestellt und die im Zuge der Bearbeitung gemachten Erfahrungen dargestellt werden. Der Fokus liegt dabei, wie bereits bei der Definition von 2DECS, auf den zu schaffenden Artefakten und den sie gliedernden Phasen des Prozesses. Darüber hinaus wird die Anpassbarkeit an unterschiedliche Teamgrößen diskutiert.

Auszug der Artefakte

Im Zuge derAusführungsplanung des *OEEX Mobile Power* Projektes wurde in *Trello* [14] eine regelmäßige Sprintplanung durchgeführt. Abbildung 7.4 zeigt das Sprint Backlog und den Fortschritt des ersten Sprints. Die für diesen Sprint gewählten User Stories umfassen sowohl Stories aus Sicht des menschlichen Nutzers als auch Stories, welche aus Sicht individueller Agenten definiert wurden. Es wurde darauf geachtet aufeinander aufbauende Stories auszuwählen, welche eine durchgängige Implementierung ermöglichen. Ziel dieses Entwicklungsschrittes ist es somit den Ladestand der Power Bank, wie auch des Telefons, an den zugehörigen Agenten zu übermitteln und dem Nutzer zu visualisieren. Die ursprünglich erhofften Synergien zwischen der Erhebung und Kommunikation des Ladestandes der beiden Geräte konnten jedoch nur in Hinblick auf die Kommunikation der Sensoren mit den zugehörigen Agenten genutzt werden.
Abbildung 7.5 zeigt einen Ausschnitt des Zielmodells des Agent.HyGrid 2018 Projektes. Erkennbar ist hier die Hierarchisierung der Ziele, welche von der Abstrakten Forderung das „Netz stabil [zu] halten“ über die darunter gelagerten Sub-Ziele zu konkreten Anwendungen wie Beispielsweise des „Auto laden“s oder der Datenübertragung führt. Aus diesem Zielmodell können in den folgenden Schritten problemlos die notwendigen Fähigkeiten und Rollen abgeleitet werden.
Aufbauend auf dem zuvor gezeigten Zielmodell sowie eines dazwischen liegenden Fähigkeitsmodells zeigt Abbildung 7.6 einen Auszug des Rollenmodells des *Agent.HyGrid - 2018* Projektes. Hier werden einzelnen Zielen und den zur Erreichung dieser Ziele notwendigen Fähigkeiten

[14] Siehe: https://trello.com

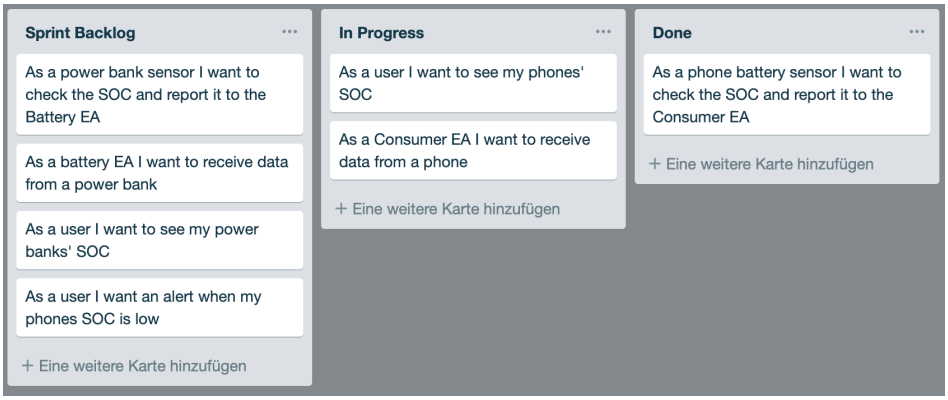

Abbildung 7.4.: Auszug aus der OEEX Mobile Power Sprint Planung in Form eines Sprint Backlogs

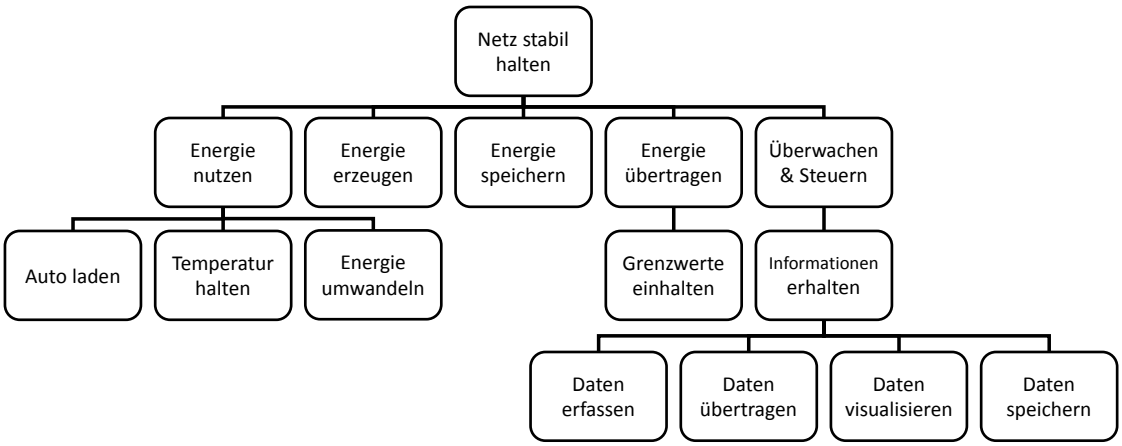

Abbildung 7.5.: Auszug des Agent.HyGrid - 2018 Zielmodells

Rollen zugeteilt. Rollen können dabei mehrere Ziele verfolgen. Siehe hierzu die Rollen „Heizer“, „EV-Nutzer“, „Speicher Controller“, „Verbraucher“ und „Erzeuger“. Auch können Ziele von mehreren Rollen verfolgt werden. Beispiel hierfür sind die Ziele „Auto laden“ und „Energie speichern“, welche mittels der Fähigkeiten „Füllstand überwachen“ und „Einspeichern“ durch die Agenten „EV-Nutzer“ und „Speicher Controller“ verfolgt werden.

Einhaltung der Phasen

Wie in Tabelle 7.1 dargestellt, fand 2DECS in unterschiedlichen Projektphasen der Fallstudien Anwendung. Im Zuge des *Agent.HyGrid - 2016* Projektes wurden die Phasen Implementierung, Simulation und Testbedanwendung mehrfach iteriert. Ein konkretes *Debriefing* oder *Deployment* fand nicht statt, da die Simulation und Testbedanwendung das Ziel des Projektes waren. Die Implementierung erfolgte auf Basis eines Vorschriften und Agenten-Modells. Auf ein Sprint Backlog wurde zu diesem Zeitpunkt verzichtet. Ein Agenten Struktur Modell wurde zu Demonstrationszwecken geschaffen um den zuvor implementierten Code besser visualisieren und diskutieren zu können. Die Simulation und Testbedanwendung erfolgte sequentiell und in Übereinstimmung der Vorgaben von 2DECS.
Für das *Agent.HyGrid - 2018* Projekt wurde 2DECS in den ersten Phasen bis hin zur Im-

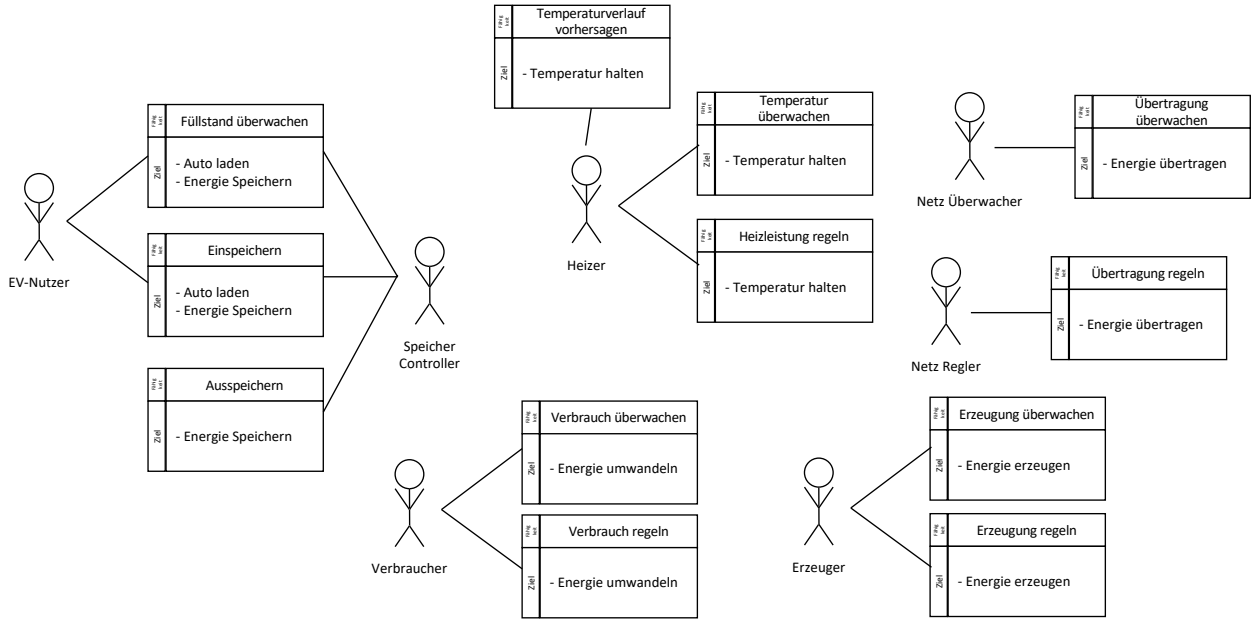

Abbildung 7.6.: Auszug des Agent.HyGrid - 2018 Rollenmodells

plementierung angewendet. Da die Projektpartner disloziert sind und Projekttreffen somit in größeren zeitlichen Abständen stattfinden, wurden dabei Phasen zusammengefasst um aufeinander aufbauende Artefakte stringent bearbeiten zu können. So wurden die Initiale Anforderungsermittlung und Vorplanung genauso wie die Anforderungsermittlung und Basisplanung jeweils in eigenen Projekttreffen miteinander verbunden. Die Ausführungsplanung wurde im Gegensatz dazu über mehrere Sitzungen gestreckt. Es wurden dabei, neben den oben in Ausschnitten gezeigten Ziel- und Rollenmodellen sämtliche technische Artefakte erarbeitet. Im Kontext der organisatorischen Artefakte wurden die Definition of Done, Acceptance Test Requirements und Story Cards oberflächlich behandelt. Die weiteren organisatorischen Artefakte (User Stories, Product- und Sprint Backlog, Release Plan) wurden gemäß den Vorgaben des 2DECS Ansatzes erstellt.
Das *OEEX Mobile Power* Projekt war durch ein ebenfalls disloziertes Team geprägt, welches für dieses Projekt keine persönlichen Treffen arrangierte. Die Planungen wurden in Videokonferenzen und mittels Online-Kollaborations Werkzeugen realisiert. Es wurde dabei der gesamte 2DECS Zyklus durchlaufen, lediglich die Phasen Anforderungsermittlung und Testbedanwendung wurden in rudimentärer Form umgesetzt. Die Phasen Initiale Anforderungsermittlung und Vorplanung sowie *Debriefing* und *Deployment* wurden jeweils miteinander verbunden. Alle anderen Phasen wurden individuell und in mehreren Iterationen bearbeitet. Artefakte wie das Product und Sprint Backlog wurden in den selben Werkzeugen und Übersichten geführt. Aufgrund der niedrigen Priorität des Projektes wurde auf einen Release Plan verzichtet. Auch wurden die User Stories nicht in explizite Story Cards übersetzt, da sie bereits durch die später an der Implementierung beteiligten Entwickler formuliert wurden.

Verteilung der Rollen

Da sich der Personalkörper der Agent.HyGrid Projekte nur unwesentlich voneinander unterschied, variierte auch die Rollenverteilung zwischen diesen Projekten nur unwesentlich. Das

Agent.HyGrid - 2016 Projekt wurde durch einen Kunden aus dem industriellen Umfeld geprägt, welcher ebenfalls die Rolle des Experten einnahm. Das *Agent.HyGrid - 2018* Projekt wurde im Gegensatz dazu auf eine universitäre Einrichtung ausgelegt, für welche dann der zugehörige Lehrstuhl die Kunden- und Expertenrolle einnahm. Die Projektleitung verblieb in beiden Projekten bei einem zweiten akademischen Partner. Ebenso wurde die Rolle des Koordinators in beiden Projekten durch einen dritten akademischen Partner wahrgenommen. Dieser brachte ebenso wie alle anderen akademischen Projektbeteiligten seine Leistung als Entwickler ein. Die Wahrnehmung mehrerer Rollen - Beispielsweise der des Projektleiters und des Entwicklers, oder der des Kunden und des Experten führten im Projektverlauf zu keinen wahrnehmbaren Problemen.
Das *OEEX Mobile Power* Projekt wurde lediglich von drei Personen begleitet. Von diesen nahm eine Person die Rolle des Kunden ein, eine zweite die des Projektleiters, Koordinators und Entwicklers sowie die dritte Person die Rollen des Entwicklers und des Experten. Auch diese Aufteilung führte zu keinen wahrnehmbaren Problemen im Projektverlauf.

7.1.3. Bewertung

Der Entwicklungsansatz wurde mittels der oben beschriebenen Fallstudien auf Basis der initial definierten Leitfragen evaluiert. Zum Zwecke einer besseren Vergleichbarkeit wurden die Leitfragen in Faktoren überführt und wie folgt quantifiziert:

- *Logische Konsistenz* - Wurden alle Artefakte in der vorgesehenen Reihenfolge mittels der hierfür definierten Aktivitäten bearbeitet, so wird dies mit ++ bewertet. Wurden hingegen alle Artefakte vertauscht oder ignoriert und auch die vorgesehenen Aktivitäten nicht durchgeführt, wird dies mit – bewertet.

- *Dimensionierung der Phasen* - Wurden alle Phasen in der vorgesehenen Reihenfolge durchlaufen und die hierin vorgesehenen Aktivitäten bearbeitet, so wird dies mit ++ bewertet. Wurden hingegen alle Phasen vertauscht oder ignoriert und auch die darin vorgesehenen Aktivitäten in anderen Phasen oder nicht durchgeführt, wird dies mit - - bewertet.

- *Zuschnitt der Rollen* - Wurden die Rollen wie definiert von einzelnen Personen ausgeführt, so wird dies mit ++ bewertet. Wurden als unteilbar definierte Rollen von mehreren Personen ausgeführt, oder inkompatible Rollen zusammen von einer Person, so wird dies mit - - bewertet.

- *Möglichkeit des Tailorings* - War es möglich die Spezifika der individuellen Projekte mit dem Ansatz abzubilden, bzw. diesen daran anzupassen, so wird dies mit ++ bewertet. Musste das Projekt an den Ansatz angepasst werden oder war eine Projektdurchführung mittels des Ansatzes nicht möglich, wird dies mit - - bewertet.

Eine Neutrale Bewertung wurde mit 0 gekennzeichnet. Im Rahmen der Auswertung ergaben sich die in Tabelle 7.2 dargestellten Werte.
Die Ergebnisse der Fallstudien führen zu einer positive Bewertung des Ansatzes. Die durch den Prozess vorgegebene Arbeitsfolge zeigte sich als logisch konsistent. Im Rahmen des *Agent.HyGrid - 2016* Projektes wurden alle beschriebenen Artefakte der bearbeiteten Phasen erarbeitet und genutzt. Das Anschlussprojekt *Agent.HyGrid - 2018* erforderte nicht alle

Phase	Agent.HyGrid 2016	Agent.HyGrid 2018	OEEX
Logische Konsistenz	++	+	+
Dimensionierung der Phasen	++	0	+
Zuschnitt der Rollen	++	++	0
Möglichkeit des Tailorings	+	+	++

Tabelle 7.2.: Ergebnisse der Fallstudien

Artefakte in voller Tiefe und im Kontext des *OEEX Mobile Power* Projektes wurden Artefakte ausgelassen, weshalb es zu einer Abwertung in der Kategorie „Logische Konsistenz“kam. Es wurden jedoch in keinem der Projekte zusätzliche Artefakte oder Aktivitäten eingeführt, weshalb von einer Vollständigkeit des Artefakt- und Aktivitätenmodells ausgegangen werden kann.
Das alle Phasen logisch aufeinander aufbauen und deren Dimensionierung sinnvoll erscheint, zeigen die Ergebnisse der Kategorie „Dimensionierung der Phasen“. Im Rahmen des *Agent.HyGrid - 2016* Projektes wurden die hierfür bestimmten Phasen mehrfach iteriert. Das *Agent.HyGrid - 2018* Projekt erforderte jedoch aufgrund der Dislozierung des Teams die Zusammenfassung, bzw. Streckung einzelner Phasen, weshalb es zu einer Abwertung in dieser Kategorie kam. Eine leichte Abwertung zeigt sich auch im Kontext des *OEEX Mobile Power* Projektes, in welchem zwei Phasen nur rudimentär bearbeitet, und wiederum zwei Phasen zusammengefasst wurden. Im Zuge der Projekte wurden jedoch keine weiteren Phasen geschaffen. Das definierte Phasen Modell wird somit ebenfalls als vollständig betrachtet. In Hinblick auf die Zusammenlegung der *Initialen Anforderungsermittlung* und *Vorplanung*, welche sowohl im *Agent.HyGrid - 2018* Projekt als auch im *OEEX Mobile Power* Projekt gemeinsam abgearbeitet wurden, ist eine weitere Untersuchung der Sinnhaftigkeit der Trennung angebracht.
Der Zuschnitt der Rollen zeigte sich in allen Projekten als zweckdienlich. Lediglich im *OEEX Mobile Power* Projekt kam es aufgrund der geringen Teamgröße zu einer Vermischung inkompatibler Rollen, weshalb hier Abzüge in der Kategorie „Zuschnitt der Rollen“vermerkt werden. Die *Agent.HyGrid* Projekte führten zwar auch zu Überschneidungen von Rollen, welche jedoch durch 2DECS zugelassen sind und kein einheitliches Bild ergaben. Es ist somit zu vermuten, dass eine weitere Bündelung von Rollen nicht notwendig ist. Darüber hinaus wurden keine weiteren Rollen definiert, womit auch hier die Hypothese der Vollständigkeit des Rollenmodell erlaubt ist.
Die Kategorie „Möglichkeit des Tailorings“zeigt durchgängige Höchstwerte. Diese ergeben sich aus der guten Anpassbarkeit des Ansatzes auf die individuellen Projekte. So war es zum einen möglich zu einem fortgeschrittenen Zeitpunkt des Projektes in den Ansatz einzusteigen, bzw. diesen nach Abarbeitung mehrerer Phasen wieder zu verlassen als auch den Ansatz für den kompletten Projektlebenszyklus zu nutzen und an die Bedürfnisse eines kleinen und verteilten Teams im *OEEX Mobile Power* Projekt anzupassen. *Tailoring* war in Hinblick auf Artefakte und Phasen somit ohne Probleme möglich.
Die guten Bewertungen bestätigen die praktische Anwendbarkeit des 2DECS Ansatzes und dessen Vollständigkeit. In Hinblick auf die hypothetische Nutzung alternativer Ansätze, wie beispielsweise die in Kapitel 5 beschriebenen, im Kontext der Fallstudien kann 2DECS seine Vorteile insbesondere in Hinblick auf sein *Artefakt-* und *Phasenmodell* verdeutlichen. Wären

bei den als Basis des 2DECS Ansatzes genutzten Vorgehensmodellen die Schaffung weiterer Artefakte, wie etwa zur Berücksichtigung der agenten- oder energiespezifischen Faktoren, oder die Aufnahme weiterer Phasen, wie der Testbedanwendung, notwendig gewesen, zeigte sich 2DECS als umfassend definiert. Diese alternativen Vorgehensmodelle sind zwar in sich selbst stets logisch konsisten mit aufeinander aufbauenden *Artefakt-*, *Aktivitäten-*, *Rollen-* und *Phasenmodellen* definiert. Will man jedoch alle Phasen mit den in 2DECS berücksichtigten Faktoren abdecken, so stoßen sie, trotzt der zum Teil guten Eigenschaften in Hinblick auf *Tailoring*, an ihre Grenzen.

7.2. Experteninterviews

Da die vorangegangenen Fallstudien zwar den Schluss zulassen, dass 2DECS im Bereich der Softwareentwicklung agentenbasierter Netzautomatisierung angewendet werden kann, jedoch nicht, ob dies auch sinnvoll oder vorteilhaft ist, wurden über die Fallstudien hinaus Experteninterviews mit Fachleuten aus den Bereichen Energietechnik und Softwareentwicklung geführt. Das hierbei genutzte Vorgehen, die erzielten Ergebnisse und eine Bewertung dieser sollen im Folgenden dargelegt werden.

7.2.1. Methodik

Als Methode wurde ein Hybrid aus qualitativer und quantitativer Befragung gewählt. So sollten zum einen mittels einer standardisierten Skala eine Indikation über den Erfüllungsgrad der zu Beginn des Kapitel 6 definierten Anforderungen erhoben und zum anderen über offene Fragen eventuell im Rahmen der Konzeption übergangene Faktoren erfasst werden. Die Interviews wurden unter Zuhilfenahme eines Leitfadens standardisiert, wobei qualitative Fragen die quantitativen ergänzten. War es einem Experten nicht möglich, auf eine Frage zu antworten, wurde dies vermerkt. Der Leitfaden ist in Anhang A, Tabelle A.1 abgebildet. Die Interviews dauerten inklusive einer Einführung in 2DECS zwischen 60 Minuten und 180 Minuten, je nach Umfang der Antworten auf die offenen Fragestellungen.
Im Vorfeld der Interviews wurden zehn Experten ausgewählt, wovon fünf die Bereiche Energietechnik und Softwareentwicklung abdeckten, drei einzig dem Bereich Softwareentwicklung und zwei ausschließlich dem Bereich der Energietechnik zuzuordnen waren. Durch diese Mischung sollte zum einen die Perspektive derer erfasst werden, für welche 2DECS vornehmlich entwickelt wurde - Experten im Bereich der Softwareentwicklung für Energiesysteme. Es sollte aber ebenso sichergestellt werden, dass die Spezifika der Domänen Energietechnik und Softwareentwicklung in ihrer Gänze erfasst werden - weshalb zusätzlich Experten mit einem starken Fokus auf diese Felder ausgewählt wurden. Die statistische Repräsentativität ist bei einem Stichprobenumfang von zehn Experten nicht gegeben. Dafür ist jedoch die Relevanz der untersuchten Subjekte hervorzuheben. Hiervon stammen sechs aus einem relevanten akademischen Umfeld und vier aus der industriellen Praxis. Alle vereint der tägliche Umgang mit den untersuchten Themenfeldern. Für die mit 2DECS nicht vertrauten Interviewteilnehmer wurde im Vorfeld des Interviews der Ansatz ausführlich präsentiert und diskutiert.

7.2.2. Ergebnisse

Die Ergebnisse werden zweigeteilt dargestellt. Tabelle 7.4 zeigt die individuellen Antworten der Experten auf die quantitativen Fragen. Die erste Spalte enthält dabei die Referenz zum Fragebogen, die letzte Spalte den Mittelwert der Antworten. Die zehn Spalten dazwischen stehen für je einen Experten. Die in der ersten Zeile genutzten Symbole markieren die Domänen und Provenienz der Experten und werden in Tabelle 7.3 erläutert. War es einem Experten nicht möglich, auf eine Frage zu antworten, findet sich in der Tabelle ein Strich. Leere Antworten wurden in der Mittelwertbildung nicht berücksichtigt. Der zweite Teil der Ergebnisse beinhaltet die Antworten der Experten auf die qualitativen Fragen im originalen Wortlaut. Es ist dabei fest zu halten, dass nicht jeder Experte auf jede Frage eine Antwort formulierte.

Ergebnisse - quantitativ

Im Tabelle 7.4 werden die quantitativen Ergebnisse der Experteninterviews dargestellt. Tabelle 7.3 erläutert die darin genutzte Nomenklatur zur Klassifizierung der Interviewpartner.

Symbol	Erläuterung
A	Akademia - Provenienz
I	Industrie - Provenienz
E	Energietechnik - Fachgebiet
S	Softwareentwicklung - Fachgebiet
SuE	Softwareentwicklung für die Energietechnik - Fachgebiet

Tabelle 7.3.: Symbole zur Einordnung der befragten Experten

Frage Nr.	Stichwort	Einordnung und Antworten der Experten										Mittelwert
		SuE|A	SuE|A	SuE|I	SuE|A	S|I	SuE|A	S|I	E|A	E|I	S|A	
11	Sicherheit	10	9	9	8	10	8	7	8	10	10	8,78
12	ISO27000	10	9	-	8	-	-	-	-	10	-	9,25
13	ISO62351	10	9	-	9	-	-	-	-	10	-	9,50
14	Qualitätsmerk.	10	9	3	7	8	8	6	8	10	10	7,67
21	Interop.	4	5	9	8	10	9	8	9	10	5	8,00
22	IEC61850	10	10	-	8	-	9	-	9	10	-	9,33
23	CIM	10	10	-	8	-	9	-	9	10	-	9,33
31	Agilität	8	9	6	7	6	9	5	6	5	10	6,78
41	Agentensysteme	7	8	7	10	10	8	9	10	10	10	8,78
51	Energiesysteme	8	9	9	8	7	8	8	9	10	-	8,44
61	Phasenabd.	6	6	6	5	5	7	5	6	7	6	5,89
62	Kundenori.	9	8	2	9	8	7	5	8	10	10	7,33
63	Detaills	5	7	6	5	7	5	5	7	7	5	6,00
65	Anpassbarkeit	5	3	8	8	8	3	9	8	6	10	6,44
66	Gesetze	10	10	7	9	9	9	9	7	8	10	9,00
71	Aufwand	8	9	5	4	7	5	-	7	5	-	6,25
72	Efizienz	8	-	9	9	8	5	8	7	10	10	8,00
73	Effektivität	-	-	7	9	9	9	9	9	10	-	8,86
74	Qualitätsori.	7	7	10	8	9	8	9	9	10	10	8,56
75	Ziele	8	7	8	9	8	9	8	8	10	10	8,33
76	Nutzen	1	1	1	1	1	1	1	1	1	1	1

Tabelle 7.4.: Quantitative Ergebnisse der Experteninterviews

Ergebnisse - qualitativ

Die Antworten der Experten auf die offenen Fragen des Interviews finden sich in Anhang B.

7.2.3. Bewertung

Auch die Auswertung der im Rahmen der Experteninterviews erhaltenen Antworten führen zu einer positiven Bewertung des 2DECS Konzeptes. Das durchgängig überdurchschnittlich positive Feedback der Experten mit mehr als 7,5 von 10 Punkten in den Bereichen Produktsicherheit, Interoperabilität, Agentenorientierung und Anwendbarkeit auf die Entwicklung von Software für Energieleitsysteme zeigen, dass der Ansatz in Hinblick auf die technischen Inhalte die richtigen Akzente setzt. Die Agilität des Prozesses wird in Frage 31 „Bewerten Sie die Agilität [..].“mit 6,78 von 10 Punkten zwar immer noch überdurchschnittlich bewertet, jedoch im Vergleich zu den vorgenannten Aspekten in abgeschwächter Form. Diese Antwort kommt nicht unerwartet, besitzt der Prozess doch eine starre Anfangs- und Endphase sowie die Notwendigkeit den Lösungsraum bereits in der ersten Phase einzuschränken. Beides sind erwünschte und den Spezifika der Energiewirtschaft geschuldete Faktoren. Mit diesen Einschränkungen hängt ebenfalls die Bewertung der Kundenorientierung in Frage 62 „Bewerten Sie die Kundenorientierung [..].“mit 7,33 von 10 Punkten zusammen. Hier sehen die Experten die Einflussmöglichkeiten der Kunden im Projektverlauf eingeschränkt. Diese Einschränkung ist grundsätzlich unerwünscht. Sie ist jedoch nur durch eine freiere Gestaltung des Anfangsstadiums zu begegnen, was wiederum den Spezifika des Energiesektors entgegenwirken würde. Die Einordnung der Phasenabdeckung in Frage 61 „Bewerten Sie die Phasenabdeckung [..].“bei 5,89 von 7 möglichen Phasen entspricht ebenfalls den Erwartungen, da bewusst auf die Außerbetriebnahme des Systems verzichtet wurde. Bemerkenswert ist die Bewertung des Detaillierungsgrades in Frage 63 „Bewerten Sie den Detaillierungsgrad [..].“welcher mit 6 Punkten über dem vorgegebenen Optimum von 5 Punkten liegt und somit als etwas zu hoch bewertet wird. Die grundlegenden Möglichkeiten des Tailorings, welches in Frage 65 „Bewerten Sie die Anpassbarkeit des Prozesses [..].“, bedingt durch den wahrgenommen hohen Detaillierungsgrad, mit 6,44 von 10 möglichen Punkten bewertet werden, hängen mit diesem Faktor direkt zusammen, korrelieren jedoch nicht mit den im Rahmen in der praktischen Erprobung des Ansatzes gemachten Erfahrungen. Die Integrierbarkeit gesetzlicher Anforderungen wird in Frage 662 „Bewerten Sie die Integrierbarkeit von gesetzlichen Anforderungen [..].“mit erfreulichen 9 von 10 Punkten bewertet, was sich mit den inhaltlichen Fragen des ersten Teils der Umfrage deckt. Die Zusammenfassung der Werte Aufwand (Frage 71 „Bewerten Sie den Aufwand des Prozesses [..].“mit 6,25 von 10 Punkten), Effizienz (Frage 72 „Bewerten Sie die Effizienz des Prozesses [..].“mit 8 von 10 Punkten) und Effektivität (Frage 73 „Bewerten Sie die Effektivität des Prozesses [..].“mit 8,86 von 10 Punkten) sind ebenso positiv wie die Einordnung der Qualitätsorientierung und der Wahl der Anforderungen in den Fragen 74 „Bewerten Sie die Qualitätsorientierung [..].“und 75 „Bewerten Sie die Wahl der Anforderungen [..].“, welche beide ebenfalls über acht Punkten liegen. Die Gesamtheit der Daten weist eine geringe Varianz auf. So sind nur einzelne Fragen durch Ausreißer geprägt, wodurch eine Extrapolation der Ergebnisse möglich erscheint.
Die Bewertung der qualitativ erhobenen Antworten erfolgt in zusammenfassender Form, auf Basis von Sinnzusammenhängen der getätigten Äußerungen. Wird ein Sachverhalt von mehr als einem Experten thematisiert, fließt es in die anschließende Diskussion ein. Die folgenden Themenbereiche wurden im Rahmen einer Kategorisierung identifiziert:

- **Fehlen von Ontologien** - Das Fehlen von Ontologien sowie einer stärkeren Fokussierung des Kommunikationsmodells auf eine semantische oder syntaktische Ebene, gegebenenfalls sogar bis hin zum OSI Schichtmodell, wurde von zwei Experten mehr-

fach in Beantwortung der Fragen 141 „Welche Aspekte des Prozesses erschweren die Integration von Qualitätsmerkmalen?“ und 411 „Welche Aspekte des Prozesses behindern dessen Anwendbarkeit auf die Entwicklung von Agentensystemen?.“ erwähnt. Der dahinter liegende Wunsch sind konkretere Vorgaben über die Form und Inhalte der Kommunikation. Da die Ausgestaltung der Kommunikation aktuell jedoch noch den Spezifika der individuellen Projekte folgt, müsste diese zunächst klassifiziert werden um so adäquate Modelle bereitstellen zu können. Die Alternative wäre die Festlegung auf ein Kommunikationsprotokoll und die starre Definition aller damit zusammenhängenden Parameter, was der allgemeinen Anwendbarkeit des Ansatzes nicht zuträglich wäre.

- **Einschränkungen der Agilität durch initiale Definition des Umgebungsmodells** - Mehrere Experten sehen bei der Beantwortung von Frage 321 „Welche Aspekte des Prozesses behindern die Agilität?“ das Problem eines zu starren Umgebungsmodells. Die Grundlage hierfür findet sich in den starren Anfangs- und Endphasen sowie der Notwendigkeit den Lösungsraum bereits in der ersten Phase des Prozesses einzuschränken. Beides sind jedoch erwünschte und den Spezifika der Energiewirtschaft geschuldete Faktoren.

- **Mangelnde Akzeptanz von Agentensystemen bei Energieunternehmen** - Auf die Frage nach der Anwendbarkeit des Prozesses in der Entwicklung von Software für Energiesysteme (Frage 511 „Welche Aspekte des Prozesses behindern dessen Anwendbarkeit auf die Entwicklung von Software für Energiesysteme?“) antworteten drei Experten, dass Agentensysteme noch immer nicht in der Energiewirtschaft akzeptiert seien. Einer schlug daher vor, das Wording zu ändern, ein anderer sah 2DECS als eine Möglichkeit die Akzeptanz zu erhöhen - was ein Ziel dieser Arbeit ist.

- **Agilität wird als der Regelkonformität hinderlich wahrgenommen** - Ebenfalls in Bezug auf Frage 511 führten zwei Interview Teilnehmer die Agilität als nicht vereinbar mit der Berücksichtigung von Regeln und Vorschriften ins Feld. Diese Auffassung wird von anderen Teilnehmern der Interviews nicht geteilt, da sie in der Agilität und der damit verbundenen Integration des Kunden eine Stärkung der Anforderungen, zu welchen auch die Regelkonformität gehört, sehen.

- **Außerbetriebnahme wird nicht berücksichtigt** - Wie bereits in der Bewertung der quantitativen Antworten diskutiert, wird auch Frage 611 „Werden für die Entwicklung von Agentensystemen oder Software in der Energietechnik relevante Phasen unterschlagen? Wenn ja, welche?“ mit dem Fehlen der Außerbetriebnahme beantwortet.

- **Die Komplexität des Prozesses erschwert die Kundenintegration** - Drei Interviewteilnehmer wünschen sich bei Frage 621 „Welche weiteren Aspekte der Kundenorientierung würden Sie als sinnvoll erachten?“ eine Reduktion der Komplexität des Prozesses um so den Einarbeitungsaufwand des Kunden zu reduzieren. Als Lösung werden hierfür eine vereinfachte Prozessansicht oder entsprechende Schulungsmaterialien für den Kunden empfohlen.

- **Rückführungen klarer Kennzeichnen** - Die im Zuge der Einführung in den 2DECS Ansatz genutzten Folien waren in Teilen unklar in Bezug auf die existierenden Rückführungen aus einer fortgeschrittenen Phase in eine vorangegangene. Frage 651 „Welche

weiteren Möglichkeiten der Anpassbarkeit würden Sie als sinnvoll erachten?“ wurde daher mit der Bitte um Anpassung beantwortet. Aus diesem Grund wurden die Folien im Zuge des Interviewprozeses entsprechend überarbeitet.

- **Der Prozess sollte eine stärke Marktorientierung besitzen** - Frage 751 „Welche weiteren Anforderungen sollten darüber hinaus aufgenommen werden?“ führte zu dem Wunsch nach einer stärkeren Marktorientierung. Die Interviewteilnehmer verbanden mit dieser jedoch unterschiedliche Schwerpunkte. So fokussierten die einen auf die Berücksichtigung von Marktteilnehmern, wohingegen andere die Vermarktbarkeit des entwickelten Produktes in den Vordergrund stellten. Eine Marktorientierung ist in Hinblick auf beide Aspekte von Relevanz für die industrielle Anwendbarkeit des Ansatzes und sollte daher in weiteren Projekten untersucht werden.

7.3. Zwischenfazit

Die Evaluation des 2DECS Konzeptes wurde anhand dreier Fallstudien und einer Serie von Experteninterviews realisiert. Über einen Zeitraum mehrerer Monate wurde 2DECS hierfür von zwei Teams in drei unterschiedlichen Projekten angewendet. Im Anschluss wurden zehn Experten aus den Bereichen Softwareentwicklung und Energietechnik aus Industrie und Akademia anhand eines Leitfadens interviewt.
Die durchgängig positiven Erfahrungen der Fallstudien sowie die bestätigenden Äußerungen der interviewten Experten vermitteln den Eindruck, dass die im Zuge der Entwicklung von 2DECS gesetzten Akzente in die richtige Richtung weisen und eine praktische Anwendbarkeit gegeben ist. Es konnte in der praktischen Anwendung gezeigt werden, dass die Phasen- und Artefaktmodelle von 2DECS vollständig definiert sind, dabei aber nicht über das notwendige Maß hinaus gehen. Lediglich die Phasen der initialen Anforderungsermittlung und der Vorplanung bedürfen in Hinblick auf ihre Trennung weiterer Untersuchungen. Auch die Rollenmodelle wurden durch die Teams der Fallstudien umgesetzt und bedurften keiner weiteren Anpassung.
Die im Zuge der Experteninterviews erhobenen quantitativen und qualitativen Bewertungen untermauern die Ergebnisse der Fallstudien. So werden die als relevant erachteten Faktoren wie die Berücksichtigung von Sicherheits- und Kommunikationsregeln, die Integrierbarkeit gesetzlicher Anforderungen in den Entwicklungsprozess wie auch die Agentenorientierung und die Anwendbarkeit im Bereich der Energietechnik äußerst positiv bewertet. Einzig die Frage des *Tailorings* wurde von den Experten, abweichend von den guten Erfahrungen aus der praktischen Anwendung, aufgrund des als leicht überhöht wahrgenommenen Detaillierungsgrades, weniger gut bewertet. Im Gegensatz dazu wurden jedoch Faktoren wie der Aufwand, die Effizienz wie auch die Effektivität und die Qualitätsorientierung von den Experten wiederum mit hohen Punktwerten versehen. Generelle Anmerkungen der Experten ließen sich in die Kategorien Markt und Prozess trennen. So wurden Wünsche in Hinblick auf eine stärkere Integration der Marktteilnehmer, die Erhöhung der Akzeptanz von Agentensystemen oder die vereinfachte Integration des Kunden geäußert. Darüber hinaus wurden auch konkrete Forderungen wie die Integration von Ontologien im Kommunikationsmodell oder der Berücksichtigung der Außerbetriebnahme aufgestellt.
Zusammenfassend lässt sich festhalten, dass die Evaluation des 2DECS Ansatzes positive Ergebnisse hervorbrachte und potentielle Weiterentwicklungsmöglichkeiten aufzeigt.

8. Zusammenfassung, kritische Bewertung und Ausblick

8.1. Zusammenfassung

Im Zuge der Anstrengungen, dem fortschreitenden Klimawandel sowie politischer und ökonomischer Abhängigkeiten zu begegnen, wird weltweit der Wandel der Energieversorgung vorangetrieben. Ziel ist es den Anteil erneuerbarer Energien zu erhöhen um so den Ausstoß von Klimagasen und anderen Schadstoffen zu minimieren und die lokale Autarkie zu fördern. Dieser Wandel von ehemals zentralen Großkraftwerken hin zu dezentralen Kleinkraftwerken, welche zum Teil von fluktuierenden Energiequellen wie Wind oder Sonne abhängig sind, führt zu einem erhöhten technischen Regelungsbedarf in den Energienetzen. Diese Regelung kann aufgrund der steigenden Komplexität zukünftig nicht mehr rein manuell bewerkstelligt werden. Netzbetreiber, Energieversorger und Energiehändler arbeiten daher gemeinsam mit akademischen Partnern an automatisierten Leitlösungen, welche die Energie-Erzeugung, -Übertragung und den -Handel zunehmend automatisieren. Die grundlegende Machbarkeit dieses Vorhabens wurde in diversen Feldversuchen bereits gezeigt. Als nächster Schritt steht nun die Industrialisierung dieser Ansätze an. Entgegen den bisherigen Implementierungen wird dabei die Qualität der Software eine zentrale Rolle spielen, da die regulatorischen Hürden sowie die Bedenken der potentiellen Kunden eine der kritischen Infrastruktur angemessene Relevanz besitzen. Die Anwendung existierender Entwicklungsansätze für derartige Problemstellungen ist in diesem Kontext jedoch schwerlich möglich, da sich die gesetzlichen und technischen Anforderungen in einem konstanten Wandel befinden. Konzepte, welche eine tiefe Integration des Kunden in den Entwicklungsprozess vorsehen, sogenannte Agile Prozesse, würden die Möglichkeit bieten, diese sich ändernden Anforderungen im Laufe des Projektes zu berücksichtigen. Diese Ansätze sind jedoch in der Regel auf die Entwicklung kleiner, nicht sicherheitskritischer Anwendungen ausgelegt und legen daher keinen Fokus auf Fragen der Legislative und Sicherheit.

In dieser Arbeit wird daher ein Ansatz beschrieben, welcher die kundenorientierte Entwicklung dezentraler Automatisierungssysteme in der Energietechnik ermöglicht. Hierfür werden zunächst bestehende Standards und Normen analysiert, welche im Rahmen der Entwicklung solcher Systeme Berücksichtigung finden müssen. Gemeinsam mit externen Experten werden diese auf ihre Relevanz in aktuellen und zukünftigen Systemen hin bewertet und ihre Effekte auf die Softwareentwicklung abgeschätzt. Es konnte dabei gezeigt werden, dass im Rahmen der Softwareentwicklung für Energiesysteme insbesondere Fragen der Sicherheit einen hohen Stellenwert besitzen. Die Normen der ISO 27000 Reihe sowie die Sicherheitsmaßnahmen nach IEC 62351 wurden daher als besonders relevant eingestuft.

Im Anschluss an diese grundlegende Untersuchung wurden aktuelle Entwicklungsansätze, sowohl für das generelle Software-Projektmanagement als auch für die konkrete Implementierung agentenbasierter Systeme, im Hinblick auf Faktoren wie die zuvor betrachteten gesetz-

lichen Anforderungen und entwicklungsprozessspezifische Eigenschaften wie die Phasenabdeckung, Kundenorientierung, Prozesssteuerung, den Detaillierungsgrad oder auch die Anpassbarkeit an individuelle Softwareprojekte untersucht. Es zeigte sich dabei eine klare Trennlinie zwischen generischen- und agentenspezifischen Entwicklungsprozessen, welche zum einen die Faktoren Kundenorientierung und Berücksichtigung der gesetzlichen Anforderungen als auch die Art der Prozesssteuerung betrafen. So sind Erstere wesentlich kundenorientierter und bieten mehr Möglichkeiten gesetzliche Anforderungen zu berücksichtigen als Zweitere, welche primär ergebnisorientiert in einem starren Korsett arbeiten. Es wurde dabei festgestellt, dass keiner der analysierten Ansätze die Anforderungen in Gänze erfüllt und daher die Definition eines angepassten Prozesses wertvoll erscheint.
Zwecks besserer Verständlichkeit wurde ein abstraktes Energie-Agenten-Konzept eingeführt, welches als Grundlage des Prozesses dient. Dieses Konzept eröffnet die Möglichkeit die entwickelten Artefakte in einem iterativen Prozess durchgängig zu nutzen und weiter zu entwickeln. Darüber hinaus wurde im Rahmen der Konzeption Wert darauf gelegt, möglichst alle Energieformen beziehungsweise -Träger mit dieser Beschreibungsform abbilden zu können.
Der auf dieser Basis definierte Entwicklungsansatz *Development approach for DEcentralised Control Systems* (2DECS) umfasst die Möglichkeit im Rahmen eines iterativen Prozesses die sich ändernden Anforderungen an ein dezentrales Automatisierungssystem in der Energietechnik zu erfassen und strukturiert in Software zu wandeln, welche darüber hinaus den regulatorischen und technischen Anforderungen Rechnung trägt. Hierfür wurden ein umfangreiches Phasen- und Artefaktmodell entwickelt, welches den Softwarelebenszyklus von der initialen Anforderungsermittlung bis hin zum Lifecycle Management abdeckt. Der Ansatz und die in ihm zu erarbeitenden Artefakte sind zweigeteilt. Sie berücksichtigen sowohl eine organisatorische als auch eine technische Ebene um so die Anforderungen an ein strukturiertes Projektmanagement als auch an die konkrete technische Umsetzung zu erfüllen. Die hierfür notwendigen Aktivitäten und die diese ausführenden Rollen wurden ebenfalls in separaten Modellen definiert. Eine generische Beschreibung der im Rahmen von 2DECS zu nutzenden Werkzeuge rundet den Ansatz ab.
Die Evaluation des Methode wurde zweigeteilt und umfasst zum einen die praktische Anwendung in drei konkreten Projekten wie auch eine Reihe von Experteninterviews, um so nicht nur die reine Funktionalität, sondern auch die Vor- und Nachteile des Ansatzes beleuchten zu können. Im Zuge der praktischen Erprobung wurde 2DECS aufgrund langer Projektlaufzeiten in verschiedenen Projektstadien eingeführt. Somit wurden unterschiedliche Aspekte des Ansatzes auf ihre Anwendbarkeit hin geprüft. Die Ergebnisse dieser Untersuchungen waren durchweg positiv und zeigten, dass die strukturierte Vorgehensweise des Ansatzes sowie die beschriebenen Artefakte, Aktivitäten und Rollen in der definierten Form angewendet werden können. Die Experteninterviews machten darüber hinaus deutlich, dass der Bedarf für einen solche Methode besteht, dass 2DECS die definierten Anforderungen erfüllt und 2DECS in der dargestellten Form von den Experten für zukünftige Projekte als Entwicklungsmethode in Betracht gezogen wird.
Der vorgestellte Ansatz kann somit die oben beschriebenen Herausforderungen im Rahmen der Softwareentwicklung erfüllen und bietet die Möglichkeit der strukturierten Entwicklung agentenbasierter Automatisierungssysteme in der Energietechnik unter Berücksichtigung sich ändernder Anforderungen und einer dauerhaften Integration des Kunden in den Entwicklungsprozess.

8.2. Kritische Bewertung

Um den oben beschriebenen Entwicklungsprozess definieren und evaluieren zu können, mussten diverse Annahmen und Vereinfachungen getroffen werden. Diese sollen im Folgenden beschrieben werden.
Im Zentrum des Ansatzes steht die Annahme, dass die Automatisierungssysteme in den Energienetzen der Zukunft dezentral strukturiert sein werden. Dieser Annahme steht entgegen, dass auch zentrale Steuerungen, welche mit BigData Ansätzen zur Analyse der Netzsituation kombiniert werden, eine potentielle Lösung für das Problem der hohen Datenmenge und Problemkomplexität bieten könnten [ZHB18]. Zum aktuellen Zeitpunkt wurden keine richtungsweisenden Entscheidungen getroffen, welche den einen oder den anderen Weg manifestieren würden, weshalb diese Annahme als notwendig erachtet wird. Sollte sich in Zukunft eine zentrale Netzführung etablieren, welche nicht auf Agenten basiert, so müssten die technischen Artefakte und als Konsequenz daraus auch das technische Phasenmodell von 2DECS an diesen Umstand angepasst werden. Modelle wie die System Anforderungen, das Energie-Options-Modell oder auch das Ressourcenmodell könnten dabei mit einer hohen Wahrscheinlichkeit beibehalten werden. Das Ziel-, Fähigkeits-, Rollen-, Interaktions- und Agentenmodell wären in der aktuellen Form jedoch wahrscheinlich nicht zweckdienlich.
Eine weitere Festlegung, welche aufgrund fehlender Datenbasis getroffen werden musste, ist die des notwendigen Abstraktionsgrades, beziehungsweise des Detaillierungsgrades. So ermöglicht ein hoher Detaillierungsgrad das einfache Abarbeiten der im Prozess definierten Vorgaben, erschwert jedoch auch die Anpassung an individuelle Entwicklungsprojekte, das sogenannte *Tailoring*. Es wurde daher ein mittlerer Detaillierungsgrad gewählt und im Zweifel die Komplexität reduziert um so eine hohe Prozesssicherheit, bei gleichzeitiger Flexibilität, zu ermöglichen.
In diesem Kontext wurde darüber hinaus darauf verzichtet 2DECS an der Struktur des *Smart-Grid-Architecture-Models* (SGAM) [Rep12] auszurichten. Der Grund hierfür findet sich in der Tatsache, dass dieses sowohl einzelne Domänen wie Erzeugung, Übertragung, Verteilung, etc., die unterschiedlichen Ebenen der Automatisierungspyramide sowie die Interoperabilitätsebenen wie Business-, Function-, Information-, Communication- und Component-Layer unterscheidet. Diese Differenzierung würde entweder die Komplexität des 2DECS Ansatz derart erhöhen, dass ein *Tailoring* nur noch schwer möglich wäre, oder aber den Ansatz auf wenige spezifische Elemente der SGAM-Matrix reduzieren.
Damit einher geht die Festlegung der abzubildenden Projektphasen, -Aktivitäten und der damit verbundenen Artefakte. So wurde bewusst auf die Definition eines Rollouts, der Außerbetriebnahme oder einer expliziten Wartungs-Phase verzichtet, da die Notwendigkeit dieser Elemente stark projektspezifisch ist. Sollte im Zuge eines Projektes die Notwendigkeit erwachsen, dass diese Phasen und Aktivitäten im Prozess berücksichtigt werden, so können diese durch das Projektteam in die 2DECS-Struktur integriert werden.
Abschließend ist hervorzuheben, dass die Evaluation des Ansatzes lediglich qualitativ und ohne marktorientierte Testfälle erfolgte. So konnte gezeigt werden, dass mit 2DECS die Softwareentwicklung möglich ist und das Experten aus der Energiedomäne sowie der Softwareentwicklung den Ansatz positiv bewerten. Die Evaluation in drei unterschiedlichen technisch getriebenen Projekten ermöglichte die Betrachtung unterschiedlicher Projekt- und Produktgruppen, führte aber auch dazu, dass die Eingangsartefakte außerhalb des Prozesses entwickelt wurden, beziehungsweise die Ausgangsartefakte in einem anderen Kontext als dem

in 2DECS vorgesehenen weiter genutzt wurden. Darüber hinaus konnte die Einbeziehung wirtschaftlicher Gesichtspunkte nicht in die Evaluation mit einbezogen werden. Ein quantitativer Vergleich des kompletten Ansatzes von der initialen Anforderungsermittlung bis hin zum Lifecyclemanagement mit anderen Ansätzen bedürfte der mehrfachen Durchführung einzelner Projekte durch die selben Teams unter Anwendung der zu vergleichenden Ansätze. Hierfür müssten entweder die Projekte in ihrem Umfang derart reduziert werden, dass daraus schwerlich Rückschlüsse auf „echte“ Entwicklungsprojekte möglich wären, oder aber die notwendigen Ressourcen zur doppelten Abarbeitung der Projekte erbracht werden.

8.3. Ausblick

Das Engineering von Software für Automatisierungssysteme in kritischer Infrastruktur wird insbesondere an der Einhaltung von Normen und Standards sowie dem unabdingbaren Fokus auf Sicherheitsaspekte gemessen. Auch in anderen Industrien finden sich diese Maxime wieder. Im Bereich der Luft- und Raumfahrttechnik, ebenso wie im Automobilbau, aber auch in automatisierten Produktions- und Logistikprozessen ist die Einhaltung von Standards ein Schlüsselement. Es ist daher denkbar 2DECS auch in diesen Industrien anzuwenden. Im Zuge der Experteninterviews wurde von unterschiedlichen Interviewpartnern auf diese Möglichkeit hingewiesen. Aufbauend auf diesem positiven Feedback werden weitere Anwendungsfälle für 2DECS sowohl im Bereich der Energieautomatisierung als auch in anderen Industrien gesucht und entsprechende Workshops angeboten werden.
Im Zuge dieser Erprobungen sollen weitere Anpassungen erfolgen, welche insbesondere auf eine Wiederverwendbarkeit der Artefakte auch über die Grenzen eines einzelnen Projektes hinaus abzielen. So sollen Wissens- wie auch Software-Artefakte in Lessons-Learned-Datenbanken, Vorlagen bzw. Schablonen für spezifische Projekttypen oder auch Modelldatenbanken für Simulationsmodelle fließen. Die strukturierte Wiederverwendung kann so Entwicklungszeit sparen und das Vertrauen, welches aus erfolgreich abgeschlossenen Projekten geschöpft wird, auf das neue Produkt übertragen.
Damit verbunden ist ein stärkerer Fokus auf einen automatisierten Staging- und Testprozess. Dieser ist in der aktuellen Form von 2DECS nur abstrakt definiert und bedarf zur besseren Anwendung weiterer Vorgaben. Hierfür müssen jedoch unterschiedliche Simulations- und Deployment-Szenarien analysiert und daraus Muster abgeleitet werden, um so an die individuellen Anwendungsfälle anpassbare Vorgehensbausteine definieren zu können.
Darüber hinaus sollte der Testprozess klarer zwischen *Unit*- und Systemtests unterscheiden. Zur Zeit wird dies über die lokale Simulation der angepassten Einheit und der späteren Integration dieser in eine umfangreichere Simulation sowie die Testbedumgebung abgebildet. Dieser Ansatz sollte weiter ausgebaut und klarer gegliedert werden, um so die Qualität des Produktes gegebenenfalls weiter zu erhöhen. Hierfür sind jedoch zunächst Untersuchungen über die entsprechenden Anforderungen der Kontrollorgane sowie der Kunden notwendig.
Es sollte darüber hinaus eruiert werden, wie im Zuge der Anforderungsermittlung „feste Werte“, beziehungsweise individuelle Grundannahmen systematisch hinterfragt werden können. So könnten Ansätze des Design Thinkings in das Artefakt- und Aktivitätsmodell aufgenommen werden um so neue Lösungen für neue Probleme zu entwickeln.
Die hier aufgeführten Entwicklungsmöglichkeiten bedürfen weiterer Forschung und Umsetzung. In seiner aktuellen Form ist 2DECS ein Konzept, welches weiter mit Leben gefüllt und an die Bedürfnisse unterschiedlicher Anwendungen und Projekte angepasst werden kann. Die

mangelnde Quantifizierbarkeit der Prozesse wie 2DECS macht es dabei schwer über Erfolg oder Misserfolg zu entscheiden. Das positive Feedback der Experteninterviews nährt jedoch die Hoffnung auf eine weitere Verbreitung des Ansatzes.

A. Interviewleitfaden zur Evaluation des Ansatzes

Leitfaden für 2DECS Experteninterview	
1.	**Produkt - Sicherheit**
11.	Bewerten Sie die Integrierbarkeit von sicherheitsrelevanten Fragestellungen in den Prozess auf einer Skala von 1 bis 10, wobei 1 für nicht integrierbar steht.
12.	Bewerten Sie die Integrierbarkeit der ISO 27000 in den Prozess auf einer Skala von 1 bis 10, wobei 1 für nicht integrierbar steht.
13.	Bewerten Sie die Integrierbarkeit der ISO 62351 in den Prozess auf einer Skala von 1 bis 10, wobei 1 für nicht integrierbar steht.
14.	Bewerten Sie die Berücksichtigung von Qualitätsmerkmalen der Stabilität und Verfügbarkeit in den Prozess auf einer Skala von 1 bis 10, wobei 1 für nicht integrierbar steht.
141.	Welche Aspekte des Prozesses erschweren die Integration von Qualitätsmerkmalen?
2.	**Produkt - Interoperabilität**
21.	Bewerten Sie die Integrierbarkeit von Fragestellungen der Interoperabilität in den Prozess auf einer Skala von 1 bis 10, wobei 1 für nicht integrierbar steht.
22.	Bewerten Sie die Integrierbarkeit der IEC 61850 in den Prozess auf einer Skala von 1 bis 10, wobei 1 für nicht integrierbar steht.
23.	Bewerten Sie die Integrierbarkeit des CIM (Common Interface Model) in den Prozess auf einer Skala von 1 bis 10, wobei 1 für nicht integrierbar steht.
231.	Welche Aspekte des Prozesses behindern die Integration von Fragestellungen der Interoperabilität?
3.	**Agilität**
31.	Bewerten Sie die Agilität des Prozesses auf einer Skala von 1 bis 10, wobei 1 für nicht agil steht.
321.	Welche Aspekte des Prozesses behindern die Agilität?
4.	**Agentenorientierung**
41.	Bewerten Sie die Anwendbarkeit des Prozesses auf die Entwicklung von Agentensystemen auf einer Skala von 1 bis 10, wobei 1 für nicht anwendbar steht.
411.	Welche Aspekte des Prozesses behindern dessen Anwendbarkeit auf die Entwicklung von Agentensystemen?
5.	**Energiesysteme**
51.	Bewerten Sie die Anwendbarkeit des Prozesses auf die Entwicklung von Software für Energieleitsysteme auf einer Skala von 1 bis 10, wobei 1 für nicht anwendbar steht.

511.	Welche Aspekte des Prozesses behindern dessen Anwendbarkeit auf die Entwicklung von Software für Energiesysteme?
6.	**Entwicklungsprozesse**
61.	Bewerten Sie die Phasenabdeckung [Anforderungsfestlegung / Grob- und Feinentwurf / Implementierung / Test und Integration / Erprobung und Inbetriebnahme / Wartung und Weiterentwicklung / Außerbetriebnahme] des Prozesses auf einer Skala von 1 bis 7, wobei 1 für keine Phasenabdeckung steht.
611.	Werden für die Entwicklung von Agentensystemen oder Software in der Energietechnik relevante Phasen unterschlagen? Wenn ja, welche?
62.	Bewerten Sie die Kundenorientierung des Prozesses auf einer Skala von 1 bis 10, wobei 1 für keine Kundenorientierung steht.
621.	Welche weiteren Aspekte der Kundenorientierung würden Sie als Sinnvoll erachten?
63.	Bewerten Sie den Detaillierungsgrad [Aktivitäts-, Rollen- und Artefaktmodell und Detaillierungsgrad des Vorgehensmodells] des Prozesses auf einer Skala von 0 bis 10, wobei 0 für zu gering, 10 für zu hoch und 5 für sehr gut steht.
64.	Haben Sie weitere Anmerkungen bezüglich des Detaillierungsgrades?
65.	Bewerten Sie die Anpassbarkeit des Prozesses auf einer Skala von 1 bis 10, wobei 1 für nicht anpassbar steht.
651.	Welche weiteren Möglichkeiten der Anpassbarkeit würden Sie als Sinnvoll erachten?
66.	Bewerten Sie die Integrierbarkeit von gesetzlichen Anforderungen in den Prozesses auf einer Skala von 1 bis 10, wobei 1 für nicht Integrierbar steht.
7.	**Abschließende Fragen**
71.	Bewerten Sie den Aufwand des Prozesses auf einer Skala von 1 bis 10, wobei 1 für geringer Aufwand steht.
711.	Wie ließe sich der Aufwand verringern?
72.	Bewerten Sie die Effizienz des Prozesses auf einer Skala von 1 bis 10, wobei 1 für Ineffizient steht.
721.	Wie ließe sich die Effizienz steigern?
73.	Bewerten Sie die Effektivität des Prozesses auf einer Skala von 1 bis 10, wobei 1 für Ineffektiv steht.
731.	Wie ließe sich die Effektivität steigern?
74.	Bewerten Sie die Qualitätsorientierung des Prozesses auf einer Skala von 1 bis 10, wobei 1 für keine Qualitätsorientierung steht.
75.	Bewerten Sie die Wahl der Anforderungen (Sicherheit, Interoperabilität, Agilität, Agenten- und Energieorientierung) des Prozesses auf einer Skala von 1 bis 10, wobei 1 für eine schlechte Wahl steht.
751.	Welche weiteren Anforderungen sollten darüber hinaus aufgenommen werden?
752.	Welche Anforderungen sind überflüssig?
76.	Würden Sie den Prozess in zukünftigen Projekten anwenden? Nein = 0; Ja = 1
77.	Welche Optimierungen würden Sie empfehlen?
78.	Haben Sie weitere Anmerkungen?

Tabelle A.1.: Leitfaden für Experteninterviews zur Evaluation des 2DECS Konzeptes

B. Qualitative Ergebnisse der Experteninterviews

Frage 141. Welche Aspekte des Prozesses erschweren die Integration von Qualitätsmerkmalen?

- Die Traceability von Lösungen zurück zu individuellen Requirements scheint unklar zu sein.
- Festgelegte Anforderungen machen die Berücksichtigungen neuer Sicherheitsaspekte unmöglich. Ggf. ist dies jedoch durch die Agilität ausgleichbar.
- Nichtfunktionale Anforderungen können nur schwer in Test Requirements übersetzt werden. Dies ist jedoch ein generelles Problem, welches nicht auf den Ansatz gemünzt ist.
- Im Energiemarkt habe ich durch die starke Regulierung und die vielen Akteure Bauchschmerzen, dass wenn etwas nur im eigenen Testbed getestet wird. Dann kann man es nicht einfach auf die vorhandene Infrastruktur loslassen. Die Verfügbarkeit ist kein Problem. Die Stabilität ist ein allgemeines Problem in der Energiedomäne.
- Unterscheidung zwischen Unit- und Systemtests einbauen. Dies ist aktuell nicht erkennbar. Ansonsten erlaubt es das Abstraktionslevel alle Anforderungen zu integrieren.

Frage 231. Welche Aspekte des Prozesses behindern die Integration von Fragestellungen der Interoperabilität?

- Es fehlen Ontologien und eine explizite Betrachtung der semantischen und syntaktischen Ebene fehlt. Außerdem sollte das OSI Schichtenmodell beachtet werden.
- Es fehlen Ontologie und es sollte mehr Standardisiert werden. Vorgaben müssen konkreter gemacht werden.
- Hängt nur an Requirements und technischen Modellen die man Nutzt. Das ist ein Punkt der in den Requirements berücksichtigt werden muss.
- Nein
- Es werden keine übergeordneten Regeln definiert. Das ist aber Abhängig von der Umsetzung.

Frage 321. Welche Aspekte des Prozesses behindern die Agilität?

- Die Kundenanforderungen sind statisch.

- Es fehlt ein Feedbackloop.
- Der Projektanfang ist starr definiert.
- Es gibt eine feste Struktur. Bei extrem agilen Ansätzen gibt es die nicht. Die sind dann auch viel schlanker und Kürzer und betrachten nur einen geringeren Zeithorizont. Allerdings sind diese nicht für große Projekte geeignet.
- Die Anforderungen des Energiemarktes erlauben keine 100protzentige Agilität.
- Der User muss selbst herausfinden, ob das was er konzipiert hat den Requirements entspricht. Das sollte automatisiert passieren.
- Integrierbarkeit neuer Anforderungen ist schwer - das Festlegen der Außengrenzen behindert die Flexibilität. Lösung wäre ggf. Die Erstellung von Story Cards und Acceptance Test Requirements in Bezug auf DOD, dabei müssen die Schnittstellen der einzelnen Module festgelegt werden um so die Bausteine miteinander besser verknüpfen zu können und so sauberer Grenzen zu erhalten.
- Abhängig von der Komplexität der Artefakte - also dem Reifegrad der Modelle.

Frage 411. Welche Aspekte des Prozesses behindern dessen Anwendbarkeit auf die Entwicklung von Agentensystemen?

- Es werden keine Ontologien berücksichtigt.
- Zentrale Datenmodelle (Ontologien) fehlen.
- 4 x Keine

Frage 511. Welche Aspekte des Prozesses behindern dessen Anwendbarkeit auf die Entwicklung von Software für Energiesysteme?

- Die Festlegung auf Agenten bedarf der Überzeugung des Kunden. Gegebenenfalls nicht von Agenten sprechen, sondern anderes Wording nutzen.
- Die Erklärungsbedürftigkeit von Energieagenten und deren Konkretisierung fehlt - Der Kunde muss verstehen was er kauft. Die Akzeptanz muss mehr gefördert werden.
- Es ist ein sehr starrer Markt. Der klar definierte Anfang ist dafür super.
- Inhärent durch die gegebene Agilität ist ggf. die Berücksichtigung regulatorischer Rahmenbedingungen nicht 100prozentig möglich.
- Anforderungen des Energiemarktes erlauben keine 100prozentig perfekte Software Entwicklung.
- Es fehlt die GARANTIE, dass das System so gehärtet ist, dass ich es mit gutem Gewissen einsetzen kann. Das kann man mit einer einfachen Simulation / Testbed Anwendung nicht machen. Man muss nach dem Testbed nochmals die Sonderbedingungen des Herstellers testen. Vorschlag: Testbed als „Proof of Technology, Proof of Concept (unter Realbedingungen)“ benennen und ein „Mehrstufiges Testbed bei Entwicklung und für Kundeneinsatz„einführen.

Frage 611. Werden für die Entwicklung von Agentensystemen oder Software in der Energietechnik relevante Phasen unterschlagen? Wenn ja, welche?

- Die Außerbetriebnahme.
- Die Außerbetriebnahme / Wartung und Weiterentwicklung, was durch die Agilität inhärent sein sollte.
- Wenn die Außerbetriebnahme Teil des Lifecyclemanagements ist, keine.

Frage 621. Welche weiteren Aspekte der Kundenorientierung würden Sie als Sinnvoll erachten?

- 2DECS ist ein sehr komplexer Prozess, welche die Entwickler unterstützt. Ggf. vereinfachte Ansicht für den Kunden erstellen.
- Der Prozess muss erlernt werden - es fehlen edukative Elemente um den Kunden zu erziehen.
- Da der Prozess kein Design Thinking enthält, halte ich ihn für nicht agil. Es werden immer Systeme auftauchen, die die Planung sprengen.
- Das Modell ist auf Stakeholder ausgelegt, aber vielleicht nicht auf User Interaktion und Integration des Users.
- Es könnten Unsicherheiten bzgl. des Rollenmodells auftauchen, da dieses sehr abstrakt ist.
- Wichtig ist es den Kunden dann auch wirklich abzuholen - das wird häufig vergessen.

Frage 64. Haben Sie weitere Anmerkungen bezüglich des Detaillierungsgrades?

- Die organisatorische Granularitätsebenen der Agentensysteme müssen besser spezifiziert werden. Deren Hierarchisierung ist aktuell nicht erkennbar. Darüber hinaus sollte dem Nutzer die Entscheidung über Hierarchieebenen überlassen werden.
- Das Kommunikationsmodell muss verbessert werden. Es ist ein höherer Detaillierungsgrad notwendig, welcher auf Beispielen basiert. Ein Organisationsmodell muss eingeführt werden - dieses muss die Zugehörigkeit von Agenten zu Organisationen abbilden.
- Ggf. noch besser klären, was Pre-Planning und Base Planning trennt.
- Besser auf Teamgröße skalierbarer machen, aktuell scheint 2DECS nicht gut für kleine Projekte zu sein.
- Darstellung der Integration der beiden Ebenen zu komplex, da es zu detailliert ist.

Frage 651. Welche weiteren Möglichkeiten der Anpassbarkeit würden Sie als Sinnvoll erachten?

- Die Rekursivität müsste frei wählbar sein.
- Rückführungen müssen klarer werden.

Frage 711. Wie ließe sich der Aufwand verringern?

- Das ist nicht möglich - eher eine Erhöhung, durch spezifischere Definition.
- Der Prozess sollte noch verschlankt werden und gewisse Aspekte sollten heraus genommen werden um mehr Agilität und weniger Struktur zu erhalten. Das wäre jedoch wahrscheinlich für Agenten und Regulation ungünstig.
- Vorlagen für einzelne Artefakte aufnehmen.
- Das ist sehr relativ zum Projektumfang.

Frage 721. Wie ließe sich die Effizienz steigern?

- Weniger Energiewirtschafts-Standards berücksichtigen und mehr auf die Umsetzung fokussieren.

Frage 731. Wie ließe sich die Effektivität steigern?

- Wenn sich die Gesetzeslage ändert ist man mit diesem Prozess aufgeschmissen.
- Effektivität ist ein prinzipielles Problem der agilen Prozesse und somit schwer zu erhöhen.

Frage 751. Welche weiteren Anforderungen sollten darüber hinaus aufgenommen werden?

- Die Skalierbarkeit der Größe des Agentensystems sollte mehr Beachtung finden.
- Der Prozess sollte mehr Marktorientierung und konkretere Ziele bieten.
- Der Rollout und die Einfachheit das an den Markt zu bringen wird nicht berücksichtigt. Die Vermarktbarkeit sollte mehr in den Fokus rücken.
- Anpassbarkeit, Maintenance
- 6 x Keine

Frage 752. Welche Anforderungen sind überflüssig?

- Die Agilität ist ggf. überflüssig. Eigentlich sind nur die Sicherheit und Interoperabilität von Relevanz.
- Agenten sind kein muss. Ggf. einige Stufen aggregieren, so dass es beispielsweise nur drei Phasen gibt.

Frage 77. Welche Optimierungen würden Sie empfehlen?

- Es sollten Ontologien aufgenommen werden.
- Das Organisations- und Kommunikationsmodell sollten verbessert werden. Die Nachhaltigkeit bzw. Wiederverwendbarkeit muss mehr in den Fokus gestellt werden.
- Mehr Design Thinking und noch mehr Agilität. Meine Welt ist nicht so starr. Für einen Anwender eines Stadtwerkes ist das sicherlich was anderes, der würde das nutzen.

- Artefakte im technischen Bereich sollten anders benannt werden. Beispielsweise das EnergieOptionsModell sollte generischer benannt werden.
- Begriffsschärfung bei der Herleitung (Anforderungen an den Prozess), nicht im Prozess selbst. Im Prozess selbst sind die Rückpfeile nicht immer logisch. Nicht alle Requirements sind immer Insuffizient. Manchmal auch andere Faktoren.
- Manche Stufen sollten aggregiert werden.
- Auch „Feste Werte“sollten hinterfragt werden - um Integrationstests zu ermöglichen. Auch Patches sollten betrachtet werden, da die mit höhere Frequenz ausgeliefert werden müssen. Ggf. Anderes Wording anstelle Lifecycle Management finden - Lifecycle Management macht idR jemand der keinen Plan von der Software hat. Ggf. anderes Wording nutzen (zB Quality Management anstelle Lifecycle Management). Ggf. keine User Stories daraus ableiten.
- Schwierig - ich bin nicht tief genug in der Softwareentwicklung drin, macht aber in Summe Sinn.

Frage 78. Haben Sie weitere Anmerkungen?

- Alle Experten antworteten mit Nein

Literaturverzeichnis

Das Literaturverzeichnis enthält alle referenzierten Quellen.
Sie werden mit [<Kurzbeleg>] referenziert.

[AB06] Abel, Dirk ; Bollig, Alexander: *Rapid Control Prototyping*. 1. Heidelberg : Springer-Verlag Berlin Heidelberg, 2006. – 400 S. https://www.springer.com/de/book/9783540295242. – ISBN 978–3–540–29524–2

[ABBU12] Appelrath, Hans-Jürgen ; Beenken, Petra ; Bischofs, Ludger ; Uslar, Mathias: *IT-Architekturentwicklung im Smart-Grid*. Berlin, Heidelberg : Springer Gabler, 2012. – 279 S. – ISBN 9783642292071

[AD13] Aichele, Christian ; Doleski, Oliver D.: *Smart Meter Rollout*. 1. Wiesbaden : Springer Fachmedien Wiesbaden, 2013. – 589 S. – ISBN 978–3–8348–2439–4

[age] *Agent.Gui Homepage*. https://www.agentgui.org/. – Besucht: 13.11.2018

[AM18] Abdalla, Reem ; Mishra, Alok: Comparing the artifacts of agent methodologies. In: *TEM Journal* 7 (2018), Nr. 2, S. 433–438. – ISSN 22178333

[BA05] Beck, Kent ; Andres, Cynthia: *Extreme Programming Explained: Embrace Change*. 2. Boston, MA : Addison-Wesley, 2005 (XP Series). – 189 S. https://www.safaribooksonline.com/library/view/extreme-programming-explained/0321278658/. – ISBN 9780321278654

[BD95] Bröhl, Adolf-P. ; Dröschel, Wolfgang ; Bröhl, Adolf-P. (Hrsg.) ; Dröschel, Wolfgang (Hrsg.): *Das V-Modell. Der Standard für die Softwareentwicklung mit Praxisleitfaden*. 2. Aufl. München : Oldenbourg Wissenschaftsverlag, 1995. – 656 S. – ISBN 3486234706

[BDE15] BDEW Bundesverband der Energie- und Wasserwirtschaft e.V.: *Whitepaper Anforderungen an sichere Steuerungs- und Telekommunikationssysteme*. Berlin, 2015

[BDL+13] Beck, Andreas ; Derksen, Christian ; Lehnhoff, Sebastian ; Linnenberg, Tobias ; Niesse, Astrid ; Rohbogner, Gregor: Energiesysteme und das Paradigma des Agenten. Version: 1, 2013. http://link.springer.com/content/pdf/10.1007/978-3-642-31768-2.pdf. In: *Agentensysteme in der Automatisierungstechnik*. 1. Berlin : Göhner, Peter, 2013. – ISBN 9783642317675, Kapitel 2, 21–42

[Bea16] Beauftrager der Bundesregierung für die Informationstechnik: *V-Modell XT*. http://download.gsb.bund.de/BundesCIO/V-Modell{_}XT{_}Bund/V-Modell-XT-Bund-2.0-Gesamt.pdf. Version: 2016

[BK13] Broy, Manfred ; Kuhrmann, Marco: *Projektorganisation und Management im Soft ware Engineering.* Berlin, Heidelberg : Springer Vieweg, 2013. – 424 S. – ISBN 9783642292897

[BPG+02] Bresciani, Paolo ; Perini, Anna ; Giorgini, Paolo ; Giunchiglia, Fausto ; Mylopoulos, John: TROPOS: AN AGENT-ORIENTED SOFTWARE DEVELOPMENT METHODOLOGY / University of Trento. Trento Italy, 2002. – Forschungsbericht

[BS14] Buchholz, Bernd M. ; Styczynski, Zbigniew: *Smart Grids – Fundamentals and Technologies in Electricity Networks.* Berlin Heidelberg : Springer-Verlag, 2014. – 396 S. – ISBN 978–3–642–45119–5

[BSI08a] BSI: *BSI-Standard 100-1 Managementsysteme für Informationssicherheit (ISMS).* Bonn, 2008

[BSI08b] BSI: *BSI-Standard 100-2 IT-Grundschutz-Vorgehensweise.* Bonn, 2008

[BSI13a] BSI: *ICS-Security-Kompendium.* `https://www.bsi.bund.de/DE/Themen/weitereThemen/ICS-Security/Empfehlungen/empfehlungen.html`. Version: 2013

[BSI13b] BSI: *Technische Richtlinie BSI TR-03109-1 Anforderungen an die Interoperabilität der Kommunikationseinheit eines.* Bonn, 2013

[CAB+94] Coleman, Derek ; Arnold, Patrick ; Bodoff, Stephanie ; Dollin, Chris ; Gilchrist, Helena ; Hayes, Fiona ; Jeremaes, Paul: *Object-Oriented Development: The fusion Method.* Hemel Hempstead, England : Prentice Hall International, 1994. – 350 S.

[Car12] Carjell, Andreas M.: Der Beitrag von Scrum zur Motivation in klassisch geführten Projekten. In: *Tagungsunterlagen zum PM-Summit 2012.* München : PMI Project Management Institute Munich Chapter e.V, 2012, 4

[CCG+02] Caire, Giovanni ; Coulier, Wim ; Garijo, Francisco ; Gomez, Jorge ; Pavon, Juan ; Leal, Francisco ; Chainho, Paulo ; Kearney, Paul ; Stark, Jamie ; Evans, Richard ; Massonet, Philippe: Agent Oriented Analysis Using Message/UML. In: Wooldridge, Michael J. (Hrsg.) ; Weiss, Gerhard (Hrsg.) ; Ciancarini, Paolo (Hrsg.): *Agent-Oriented Software Engineering II.* Berlin, Heidelberg : Springer Berlin Heidelberg, 2002. – ISBN 978–3–540–70657–1, S. 119–135

[CHMS14] Cossentino, Massimo ; Hilaire, Vincent ; Molesini, Ambra ; Seidita, Valeria: *Handbook on Agent-Oriented Design Processes.* Berlin : Springer Berlin Heidelberg, 2014 `http://link.springer.com/10.1007/978-3-642-39975-6`. – ISBN 978–3–642–39974–9

[Chr92] Chroust, Gerhard: *Modelle der Entwicklung.* Oldenbourg Wissenschaftsverlag, 1992. – 316 S. – ISBN 3486218786

[Cos05] Cossentino, Massimo: From Requirements to Code with the PASSI Methodology. In: *Agent-oriented methodologies*. Michigan : Idea Group Pub., 2005. – ISBN 9781591405818, Kapitel 4, S. 79–106

[CZ04] Cernuzzi, Luca ; Zambonelli, Franco: Experiencing AUML in the GAIA methodology. In: *Proceedings of the 6th International Conference on Enterprise Information Systems (ICEIS)* (2004), 283–288. `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.63.1200{&}rep=rep1{&}type=pdf`. ISBN 9728865007

[CZ09] Cernuzzi, Luca ; Zambonelli, Franco: Gaia4E: A Tool Supporting the Design of MAS using Gaia. In: *Proceedings of the 11th International Conference on Enterprise Information Systems (ICEIS 2009)* (2009), S. 82–88. ISBN 9789898111845

[DFP19] Dingsoeyr, T ; Falessi, D ; Power, K: Agile Development at Scale: The Next Frontier. In: *IEEE Software* 36 (2019), mar, Nr. 2, S. 30–38. – ISSN 0740–7459

[DGo07] DeLoach, Scott A. ; Garcia-ojeda, Juan C.: *Framework Description - Organization-based Multiagent System Engineering (O-MaSE)*. `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.128.4908{&}rep=rep1{&}type=pdf`. Version: 2007

[DGO10] DeLoach, Scott A. ; Garcia-Ojeda, Juan C.: O-MaSE: a customisable approach to designing and building complex , adaptive multi-agent systems. In: *International Journal of Agent-Oriented Software Engineering* 4 (2010), sep, Nr. 3, S. 244–280

[DIN05] DIN: *DIN EN 61508 Beiblatt 1 - Funktionale Sicherheit sicherheitsbezogener elektrischer/elektronischer/programmierbarer elektronischer Systeme Teil 0: Funktionale Sicherheit und die IEC 61508*. Berlin, 2005

[DIN08] DIN: *DIN ISO/IEC 27002 Informationstechnik – IT-Sicherheitsverfahren – Leitfaden für das Informationssicherheits-Management*. Berlin, 2008

[DIN11a] DIN: *DIN EN 50160 Merkmale der Spannung in öffentlichen Elektrizitätsversorgungsnetzen*. Berlin, 2011

[DIN11b] DIN: *DIN EN 61508-1 Funktionale Sicherheit sicherheitsbezogener elektrischer/elektronischer/programmierbarer elektronischer Systeme*. Berlin, 2011

[DIN11c] DIN: *DIN ISO/IEC 27000 Informationstechnik – IT-Sicherheitsverfahren – Informationssicherheits-Managementsysteme – Uberblick und Terminologie*. Berlin, 2011

[DIN11d] DIN: *DIN ISO/IEC 27005 Informationstechnik - IT-Sicherheitsverfahren - Informationssicherheits-Risikomanagement*. Berlin, 2011

[DIN14a] DIN: *Deutsche Normungs-Roadmap IT-Sicherheit*. Berlin, 2014

[DIN14b] DIN: *DIN EN 61850-7-4 Kommunikationsnetze und -systeme für die Automatisierung in der elektrischen Energieversorgung – Teil 7-4: Grundlegende Kommunikationsstruktur – Kompatible Logikknoten- und Datenklassen*

[DIN14c] DIN: *DIN EN 61968-1 ICS 35.240.99 DEUTSCHE NORM Februar 2014 DIN EN 61968-1 D Ersatz für DIN EN 61968-1:2004-09 Siehe Anwendungsbeginn Integration von Anwendungen in Anlagen der Elektrizitätsversorgung – Systemschnittstellen für Netzführung – Teil 1: Schnittst.* Berlin, 2014

[DIN14d] DIN: *DIN ISO/IEC 27001 Informationstechnik – IT-Sicherheitsverfahren – Informationssicherheits-Managementsysteme – Anforderungen*

[DIN15a] DIN: *DIN EN 60870-6-503 Fernwirkeinrichtungen und -systeme – Teil 6-503: Fernwirkprotokolle, die mit ISO-Normen und ITU-T-Empfehlungen kompatibel sind – TASE.2-Dienste und -Protokoll.* Berlin, 2015

[DIN15b] DIN: *DIN EN 60870-6-802.* Berlin, 2015

[DIN15c] DIN: *DIN ISO IEC 62443-3-3 Industrielle Kommunikationsnetze - IT-Sicherheit für Netze und Systeme - Teil 3-3: Systemanforderungen zur IT-Sicherheit und Security-Level.* Berlin, 2015

[DIN15d] DIN: *DIN ISO/IEC TR 27019 Informationstechnik – Sicherheitsverfahren – Leitfaden für das Informationssicherheitsmanagement von Steuerungssystemen der Energieversorgung auf Grundlage der ISO/IEC 27002*

[DIN17] DIN: *DIN ISO/IEC 27003 Informationstechnik - Sicherheitsverfahren - Informationssicherheitsmanagementsysteme - Anleitung.* Berlin, 2017

[DL13] Derksen, Christian ; Linnenberg, Tobias: Unified Energy Agents as a Base for the Systematic Development of Future Energy Grids Initial Situation Agent . GUI & DEMAPOS. In: *Multiagent System.* 2013. – ISBN 978–3–642–40775–8, S. 236–249

[DLNLS15] Derksen, Christian ; Linnenberg, Tobias ; Neusel-Lange, Nils ; Stiegler, Martin: Agent.HyGrid : A seamless Development Process for agent- based Control Solutions in hybrid Energy Infrastructures. Essen : E-world energy & water, 2015, S. 1–8

[DU11] Derksen, Christian ; Unland, Rainer: Agent.GUI: A Multi-agent Based Simulation Framework. In: *Proceedings of the Federated Conference on Computer Science and Information Systems.* Mragowo : Piscataway, NJ IEEE, 2011. – ISBN 9788360810224, S. 623–630

[DVM03] Drogoul, Alexis ; Vanbergue, Diane ; Meurisse, Thomas: Multi-Agent Based Simulation: Where are the Agents? In: *Lecture Notes in Computer Science* 2581 (2003), S. 1–15. ISBN 978–3–540–00607–7

[DWS01] DeLoach, Scott A. ; Wood, Mark F. ; Sparkman, Clint H.: Multiagent Systems Engineering. In: *International Journal of Software Engineering and Knowledge Engineering* 11 (2001), Nr. 3, S. 231–258

[eeb] *eebus Homepage*. http://www.eebus.org. – Besucht: 13.11.2018

[FPMT01] Fuxman, Ariel ; Pistore, Marco ; Mylopoulos, John ; Traverso, Paolo: Model checking early requirements specifications in Tropos. In: *Proceedings of the IEEE International Conference on Requirements Engineering* (2001), S. 174–181. – ISBN 0–7695–1125–2

[FR15] Farid, Amro M. ; Ribeiro, Luis: An Axiomatic Design of a Multi-Agent Reconfigurable Manufacturing System Architecture. In: *IEEE Transactions on Industrial Informatics* 11 (2015), Nr. 5, S. 1142 – 1155

[Fra14] Fraunhofer IWES, Siemens AG, Institut für Energieversorgung und Hochspannungstechnik Fachgebiet Elektrische Energieversorgung der Universität Hannover, CUBE Engineering G.: Kombikraftwerk 2 Abschlussbericht. 2014 (August). – Forschungsbericht. – 24 S.

[GGFJ11] Giordano, Vincenzo (JRC-IE) ; Gangale, Flavia (JRC-IE) ; Fulli, Gianluca (JRC-IE) ; Jiménez, Manuel Sánchez (DG ENER): *Smart grid projects in Europe*. http://research.microsoft.com/apps/pubs/default.aspx?id=157701. Version: 2011

[GMS05] Giorgini, Paolo ; Mylopoulos, John ; Sebastiani, Roberto: Goal-oriented requirements analysis and reasoning in the Tropos methodology. In: *Engineering Applications of Artificial Intelligence* 18 (2005), Nr. 2, S. 159–171. – ISSN 09521976

[GSV99] Gerber, Christian ; Siekmann, Jörg ; Vierke, Gero: Flexible Autonomy in Holonic Agent Systems. In: *Proceedings of the 1999 AAAI Spring Symposium*. Palo Alto, 1999, 52–58

[GSV+15] Gulzar, Kashif ; Sierla, Seppo ; Vyatkin, Valeriy ; Papakonstantinou, Nikolaos ; Flikkema, Paul G. ; Yang, Chen W.: An auction-based smart district heating grid. In: *Proceedings of IEEE International Conference on Emerging Technologies and Factory Automation, ETFA* 2015-Octob (2015). – ISBN 9781467379298

[HBH18] Hosni, A ; Bedor, H ; Hassanin, W: Applying Tropos for Web-Based Multi-Agent eBookstore. In: *Proceedings of 2018 International Conference on Computing Sciences and Engineering (ICCSE)*. Kuwait City : IEEE, mar 2018. – ISBN 978–1–5386–4681–6, S. 1–6

[Hel16] Heller, Jacob: *Technische und regulatorische Anforderungen an das Software Engineering von Smart-Grid-Projekten*, Helmut-Schmidt-Universität Universität der Bundeswehr Hamburg, Masterthesis, 2016. – 146 S.

[HK14] Hilmer, Stefan ; Krieg, Alexander: *Standardisierung vs. Kultur: Klassisches und agiles Projektmanagement im Vergleich*. Hamburg, 2014

[HLR11] Häger, Ulf ; Lehnhoff, Sebastian ; Rehtanz, Christian: Verteilte koordinierte Lastflusssteuerung in elektrischen Energieübertragungsnetzen. In: *At-Automatisierungstechnik* 59 (2011), Nr. 3, S. 153–160. – ISSN 01782312

[HMPR04] Hevner, Alan ; March, Salvatore ; Park, Jonsoo ; Ram, Sudha: DESIGN SCIENCE IN INFORMATION SYSTEMS RESEARCH. In: *MIS Quarterly* 28 (2004), Nr. 1, S. 75–105

[IEC07a] *IEC TS 62351-6 - Power systems management and associated information exchange – Data and communications security – Part 6: Security for IEC 61850.* Genf, 2007

[IEC07b] IEC: *IEC 62351-1 Power systems management and associated information exchange – Data and communications security - Part 1: Communication network and system security – Introduction to security issues.* Genf, 2007

[IEC08] IEC: *ISO/IEC 12207 System and software Engineering - Software life cycle processes.* Genf, 2008

[IEC10] IEC SMB Smart Grid Strategic Group (SG3): *IEC Smart Grid Standardization Roadmap.* Genf, 2010

[IEC11] *IEC TS 62351-8 - Power systems management and associated information exchange – Data and communications security – Part 8: Role-based access control.* Genf, 2011

[IEC14] IEC: *IEC TR 62351-10 Power systems management and associated information exchange – Data and communications security – Part 3: Communication network and system security – Profiles including TCP/IP, 33.200.* Genf, 2014

[IGGV97] Iglesias, Carlos A. ; Garijo, Mercedes ; Gonzales, José C. ; Velasco, Juan R.: Analysis and Design of Multiagent Systems Using MAS-CommonKADS *. In: *Proceedings of the International Workshop on Agent Theories, Architectures, and Languages.* Berlin Heidelberg : Springer, 1997. – ISBN 978–3–540–64162–9, S. 313–327

[Ise] IsecT Ltd.: *ISO / IEC TR 27019.* `http://www.iso27001security.com/html/27019.html`. – Besucht: 27.06.2017

[JSBG17] Jones, Dave ; Sakhel, Dr. A. ; Buck, Matthias ; Graichen, Dr. P.: *Agora Energiewende and Sandbag (2018): The European Power Sector in 2017. State of Affairs and Review of Current Developments.* `https://www.agora-energiewende.de/fileadmin/Projekte/2018/EU{_}Jahresrueckblick{_}2017/Agora{_}EU-report-2017{_}WEB.pdf`. Version: 2017

[Kal13] Kalus, Georg: *Projektspezifische Anpassung von Vorgehensmodellen: Featurebasiertes Tailoring*, Technische Universität München, Dissertation, 2013. `http://d-nb.info/1038526981`. – 263 S.

[KG96] Kinny, David ; Georgeff, Michael: A methodology and modelling technique for systems of BDI agents. In: Van de Velde, W. (Hrsg.) ; Perram, J.W (Hrsg.): *Proceedings of Agents Breaking Away. MAAMAW.* Berlin, Heidelberg : Springer, 1996. – ISBN 978–3–540–60852–3, S. 56–71

[KL14] Kuhrmann, Marco ; Linssen, Oliver: Welche Vorgehensmodelle nutzt Deutschland? / PMV-Bericht. 2014. – Forschungsbericht

[Klü08] Klügl, Franziska: A validation methodology for agent-based simulations. In: *Proceedings of the 2008 ACM symposium on Applied computing - SAC '08*. New York, New York, USA : ACM Press, 2008. – ISBN 9781595937537, 39

[Koc15] Koch, Tanja; Hermsmeier, Jörg: Die IEC 61850 als Standard für die Digitalisierung des Energiesystems. In: *etz elektrotechnik & automation* (2015), Nr. 11. `http://www.etz.de/5725-0-Die+IEC+61850+als+Standard+fuer+die+Digitalisierung+des+Energiesystems.html`

[KRB+11] Kamphuis, I G. ; Roossien, B ; Bliek, F.W. ; Noort, A. van d. ; Velden, J. van d. ; Wit, J. de: *Architectural and first results evaluation of the PowerMatchingCity fieldtest*. Petten, 2011

[KWK+05] Kok, Koen ; Warmer, Cor ; Kamphuis, Rene ; Mellstrand, Per ; Gustavsson, Rune: Distributed control in the electricity infrastructure. In: *Proceedings of 2005 International Conference on Future Power Systems*, 2005. – ISBN 90–78205–02–4, 7 pp.–7

[LDFU15] Linnenberg, Tobias ; Derksen, Christian ; Fay, Alexander ; Unland, Rainer: Cross-Domain Energy Savings by Means of Unified Energy Agents. In: *Engineering Applications of Artificial Intelligence* 41 (2015), may, Nr. C, S. 310–324. ISBN 9780128004111

[Leh10] Lehnhoff, Sebastian: *Dezentrales vernetzes Energiemanagement*. 1. Wiesbaden : Vieweg + Teubner, 2010. – 264 S. – ISBN 9783834812704

[LKB+10] Langer, Philipp ; Köbler, Felix ; Berkovich, Marina ; Weyde, Felix ; Leimeister, Jan M. ; Krcmar, Helmut: Vorgehensmodelle für die Entwicklung hybrider Produkte-eine Vergleichsanalyse. In: Schumann, M. (Hrsg.) ; Kolbe, L. M. (Hrsg.) ; Breitner, M. H. (Hrsg.) ; Frerichs, A. (Hrsg.): *Proceedings of Multikonferenz Wirtschaftsinformatik 2010: Integration von Produkt & Dienstleistung - Hybride Wertschöpfung*. Göttingen, 2010. – ISSN 3839144418, S. 2043–2056

[LKRW11] Lehnhoff, Sebastian ; Krause, Olav ; Rehtanz, Christian ; Wedde, Horst F.: Dezentrales autonomes Energiemanagement: Für einen zulässigen Betrieb innerhalb verfügbarer Kapazitätsgrenzen. In: *At-Automatisierungstechnik* 59 (2011), Nr. 3, S. 167–179. – ISSN 01782312

[Luk15] Lukas Koppe: *Analyse von Software-Entwicklungs-Methoden*, Helmut Schmidt-Universität Hamburg, Master Thesis, 2015. – 100 S.

[LWF13] Linnenberg, Tobias ; Wior, Ireneus ; Fay, Alexander: Analysis of potential instabilities in agent-based smart grid control systems. In: Palensky, Peter (Hrsg.) ; Gomes, Luis (Hrsg.) ; Chow, Mo-Yuen (Hrsg.): *Proceedings of IECON (Industrial Electronics Conference)*. Wien : IEEE, 2013. – ISBN 9781479902248, S. 7422–7427

[LWSF11] Linnenberg, Tobias ; Wior, Ireneus ; Schreiber, Sebastian ; Fay, Alexander: A market-based multi-agent-system for decentralized power and grid control. In: Mammeri, Zoubir (Hrsg.) ; Sabatier, Paul (Hrsg.): *Proceedings of IEEE International Conference on Emerging Technologies and Factory Automation, ETFA*. Toulouse : IEEE, 2011. – ISBN 9781457700187, S. 1–8

[Mac06] MacKiewicz, R. E.: Overview of IEC 61850 and benefits. In: *Proceedings of 2006 IEEE PES Power Systems Conference and Exposition, (PSCE)*. Atlanta : IEEE, 2006. – ISBN 142440178X, S. 1–8

[MNP+08] Morandini, Mirko ; Nguyen, Duy C. ; Perini, Anna ; Siena, Alberto ; Susi, Angelo: Tool-supported development with tropos: The conference management system case study. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 4951 LNCS (2008), S. 182–196. – ISBN 3540794875

[mod18] *modbus Homepage*. `http://www.modbus.org`. Version: 2018. – Besucht: 13.11.2018

[MPP09] Morandini, Mirko ; Penserini, Loris ; Perini, Anna: Operational Semantics of Goal Models in Adaptive Agents. In: Sierra, Charles (Hrsg.) ; Castelfranchi, Cristiano (Hrsg.) ; Decker, Keith S. (Hrsg.) ; Sichman, Jaime S. (Hrsg.): *Proceedings of the 8th International Conference on Autonomous Agents and Multiagent Systems*. Budapest : International Foundation for Autonomous Agents and Multiagent Systems, 2009. – ISBN 978–0–9817381–6–1, 129–136

[MS18] Mishra, S K. ; Singh, V K.: Building Semantic Information Retrieval System for Legal Cases From Heterogeneous Adapted and Diverse Data Sources Using Extended GAIA Methodology for Multi Agent System. In: *Proceedings of 3rd International Conference On Internet of Things: Smart Innovation and Usages (IoT-SIU)*. Bhimtal : IEEE, 2018. – ISBN 978–1–5386–3833–0, S. 1–4

[Nes07] Nestle, David: *Energiemanagement in der Niederspannungsversorgung mittels dezentraler Entscheidung*, Universität Kassel, PhD Thesis, 2007

[NS99] Noack, Jörg ; Schienmann, Bruno: Objektorientierte Vorgehensmodelle im Vergleich. In: *Informatik-Spektrum* 22 (1999), S. 166–180

[Omi01] Omicini, Andrea: SODA : Societies and Infrastructures in the Analysis and Design of Agent-based Systems. In: Ciancarini, P. (Hrsg.) ; Wooldridge, M.J. (Hrsg.): *Proceedings of Agent-oriented software engineering*. Berlin, Heidelberg : Springer, 2001. – ISBN 978–3–540–41594–7, S. 311–326

[opc] *OPC Foundation Homepage*. `https://opcfoundation.org`. – Besucht: 13.11.2018

[PBL05] Pokahr, Alexander ; Braubach, Lars ; Lamersdorf, Winfried: Jadex: A BDI reasoning engine. In: Bordini, R.H. (Hrsg.) ; Dastani, M. (Hrsg.) ; Dix, J. (Hrsg.) ; El Fallah Seghrouchni, A. (Hrsg.): *Multi-Agent Programming*. Boston, MA : Springer, 2005. – ISBN 978–0–387–24568–3, Kapitel 1, S. 149–174

[pro] *Profibus Webpage*. `http://www.profibus.com`. – Besucht: 13.11.2018

[Pro13] Project Management Institute: *PMBOK Guide*. 5. Newtown Square PA, US : Projekt Management Institute, Inc, 2013. – 589 S. – ISBN 8925598620

[PS05] Perini, Anna ; Susi, Angelo: Agent-Oriented visual modeling and model validation for engineering distributed systems . In: *International Journal of Computer Systems Science and Engineering* 20 (2005), Nr. 4

[PW04] Padgham, Lin ; Winikoff, Michael: The Prometheus Methodology. In: Bergenti, Federico (Hrsg.) ; Gleizes, MP. (Hrsg.) ; Zambonelli, Franco (Hrsg.): *Methodologies and Software Engineering for Agent Systems. Multiagent Systems, Artificial Societies, and Simulated Organizations*. Springer : Springer, 2004. – ISBN 978–1–4020–8057–9, Kapitel 11, S. 217–234

[RBRH09] Richardot, Olivier ; Besanger, Yvon ; Radu, Daniel ; Hadjsaid, Nouredine: Optimal location of pilot buses by a Genetic Algorithm approach for a coordinated voltage control in distribution systems. In: Lucian, Toma (Hrsg.) ; Otomega, Bogdan (Hrsg.): *Proceedings of IEEE Bucharest PowerTech*. Bukarest : IEEE, 2009. – ISBN 9781424422357, S. 1–7

[Rep12] Report, Smart Grids Coordination Group T.: Smart Grid Reference Architecture. 2012 (November). – Forschungsbericht

[RFB+14] Rohbogner, Gregor ; Fey, Simon ; Benoit, Pascal ; Wittwer, Christof ; Christ, Andreas: Design of a Multiagent-Based Voltage Control System in Peer-to-Peer Networks for Smart Grids. In: *Energy Technology* 2 (2014), Nr. 1, S. 107–120. – ISBN 2194–4296

[RFH+12] Rohbogner, Gregor ; Fey, Simon ; Hahnel, U.J.J. ; Benoit, P. ; Wille-Haussmann, B.: What the term agent stands for in the Smart Grid definition of agents and multi-agent systems from an engineer's perspective. In: Ganzha, Maria (Hrsg.) ; Maciaszek, Leszek A. (Hrsg.) ; Paprzycki, Marcin (Hrsg.): *Proceedings of Computer Science and Information Systems (FedCSIS), 2012 Federated Conference*. Breslau, Polen : IEEE, 2012. – ISBN 9788360810484, S. 1301–1305

[RGRG95] Rao ; Georgeff ; Rao, Anand S. ; Georgeff, Michael P.: BDI agents: From theory to practice. In: Lesser, Victor (Hrsg.) ; Gasser, Les (Hrsg.): *Proceedings of the FIRST INTERNATIONAL CONFERENCE ON MULTI-AGENT SYSTEMS (ICMAS-95)* Bd. 95. San Francisco : MIT Press, 1995. – ISBN 0262621029, 312–319

[Ric06] Richardot, Olivier: *Réglage Coordonné de Tension dans les Réseaux de Distribution à l'aide de la Production Décentralisée*, Grenoble, Doctoral Thesis, 2006. `http://tel.archives-ouvertes.fr/tel-00160006/`

[Roy70] Royce, Winston.W.: Managing the development of large software systems. In: *Proceedings of IEEE WESCON*. Los Angeles : IEEE, 1970, S. 328–338

[SBP+07] Sudeikat, Jan ; Braubach, Lars ; Pokahr, Alexander ; Lamersdorf, Winfried ; Renz, Wolfgang: Validation of BDI Agents. In: Bordini, Rafael H. (Hrsg.) ; Dastani, Mehdi (Hrsg.) ; Dix, Jürgen (Hrsg.) ; Seghrouchni, Amal El F. (Hrsg.): *Proceedings of the 4th international conference on Programming multi-agent systems (ProMAS'06)*. Hakodate, Japan : Springer, 2007. – ISBN 978–3–540–71955–7, S. 185–200

[SKD+15] Sato, Takuro ; Kammen, Daniel M. ; Duan, Bin ; Macuha, Martin ; Zhou, Zhenyu ; Wu, Jun ; Tariq, Muhammad ; Asfaw, Solomon A.: *Smart Grid Standards*. Bd. 1. Singapore : John Wiley & Sons Singapore Pte. Ltd, 2015 http://doi.wiley.com/10.1002/9781118653722. – ISBN 9781118653722

[SKS18] Siswahyudi, Puad ; Kurniawan, Tri A. ; Sugiarto, Very: Agent-Oriented Methodologies Comparison: A Literature Review. In: *Advanced Science Letters* 24 (2018), nov, S. 8710–8716

[Som10] Sommerville, Ian: *Software Engineering*. 9. Boston, MA, USA : Pearson Addison Wesley, 2010 https://www.bibsonomy.org/bibtex/2094b37a4bb8b242d7694cdc9142e0d80/flint63. – ISBN 9780132165433

[SS13] Schwaber, Ken ; Sutherland, Jeff: *The Definitive Guide to Scrum: The Rules of the Game*. http://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-US.pdf. Version: 2013

[TC05] Trencansky, Ivan ; Cervenka, Radovan: Agent Modeling Language (AML): A Comprehensive Approach to Modeling MAS. In: *Informatica* 29 (2005), nov, 391–400. http://wen.ijs.si/ojs-2.4.3/index.php/informatica/article/viewFile/55/48

[Trö10] Tröschel, Martin: *Aktive Einsatzplanung in holonischen Virtuellen Kraftwerken*. Oldenburg : Oldenburger Verlag für Wirtschaft, Informatik und Rech, 2010. – 213 S. – ISBN 978–3–939704–55–3

[UBA18] *AGEE Statistiken*. https://www.umweltbundesamt.de/sites/default/files/medien/372/dokumente/agee-stat{_}monatsbericht{_}10-2018.pdf. Version: 2018. – Besucht: 26.11.2018

[UG07] Uslar, Mathias ; Griming, Fabian: Zur Semantischen Interoperabilität in der Energiebranche: CIM IEC 61970. In: *Wirtschaftsinformatik* 49 (2007), Nr. 4, S. 295–303. – ISSN 09376429

[USD+13] Uslar, Mathias ; Specht, Michael ; Dänekas, Christian ; Trefke, Jörn ; Rohjans, Sebastian ; González, José M. ; Rosinger, Christine ; Bleiker, Robert ; Uslar, Mathias (Hrsg.) ; Specht, Michael (Hrsg.) ; Dänekas, Christian (Hrsg.) ; Trefke, Jörn (Hrsg.) ; Rohjans, Sebastian (Hrsg.) ; González, José M. (Hrsg.) ; Rosinger, Christine (Hrsg.) ; Bleiker, Robert (Hrsg.): *Standardization in Smart Grids*. Berlin Heidelberg : Springer, 2013. – 15–37 S. – ISBN 978–3–642–34915–7

[Usl09] Uslar, Mathias; Rohjans, Sebastian; Schmedes, Tanja; González, José M.; Beenken, Petra; WEidelt, Tobias; Specht, Michael; Mayer, Christoph; Niesse, Astrid; Kamenik, Jens; Busemann, Claas: Untersuchung des Normungsumfeldes zum BMWi-Förderschwerpunkt e-Energy-IKT-basiertes Energiesystem der Zukunft / OFFIS. Oldenburg, Germany, 2009. – Forschungsbericht. – 230 S.

[USR+12] Uslar, Mathias ; Specht, Michael ; Rohjans, Sebastian ; Trefke, Jörn ; Vasquez Gonzalez, Jose M.: *Power Systems*. Bd. 66: *The Common Information Model CIM*. Berlin Heidelberg : Springer, 2012. – 186 S. http://link.springer.com/10.1007/978-3-642-25215-0. – ISBN 978–3–642–25214–3

[VDE13] VDE/DKE: *Normungsroadmap E-Energy / Smart Grids 2.0: Status, Trends und Perspektiven der Smart Grid Normung*. Berlin, 2013

[VDI] VDI GMA FA 5.15: *VDI/VDE Richtlinie 2653 Agentensysteme in der Automatisierungstechnik Grundlagen Blatt 2*

[VDI10] VDI GMA FA 5.15: *VDI/VDE Richtlinie 2653 Agentensysteme in der Automatisierungstechnik Grundlagen Blatt 1*. 2010

[VDI15] VDI Fachausschuss „Regenerative Energien“: *Statusreport Regenerative Energien in Deutschland 2015*. Düsseldorf, 2015

[VDM13] VDMA: Status Quo der Security in Produktion und Automation / VDMA. Frankfurt am Main, 2013. – Forschungsbericht. – 32 S.

[Von98] Von Fittkau, Boris: *Die Bedeutung von Kundenorientierung für die Softwareentwicklung*, Univeristät Hamburg, Diplomarbeit, 1998. https://swa.informatik.uni-hamburg.de/files/abschlussarbeiten/DiplomarbeitFittkau.pdf

[War03] Warden, Shane: *Extreme Programming Pocket Guide*. 1. Sebastopol, Kalifornien : O'Reilly Media, Inc., 2003. – 106 S. – ISBN 0596004850

[WJ95] Wooldridge, Michael ; Jennings, Nicholas R.: Intelligent agents: theory and practice. In: *The Knowledge Engineering Review* 10 (1995), Nr. 2, S. 115–152. – ISSN 10958630

[WJ05] Weiss, Gerhard ; Jakob, Ralf: *Agentenorientierte Softwareentwicklung : Methoden und Tools*. Berlin Heidelberg : Springer, 2005. – 316 S. http://primo.bibliothek.kit.edu/primo{_}library/libweb/action/display.do?tabs=detailsTab{&}ct=display{&}fn=search{&}doc=KITSRC107670097{&}indx=1{&}recIds=KITSRC107670097{&}recIdxs=0{&}elementId=0{&}renderMode=poppedOut{&}displayMode=full{&}frbrVersion={&}dscnt=2{&}fromLogin=true{&}t. – ISBN 3540000623

[WJK00] Wooldridge, M ; Jennings, N R. ; Kinny, D C N W-2000-01: The Gaia Methodology for Agent-Oriented Analysis and Design. In: *Journal of Autonomous Agents and Multi-Agent Systems* 3 (2000), Nr. 3, S. 285–312. – ISSN 1387–2532

[WLHK06] Wedde, Horst F. ; Lehnhoff, Sebastian ; Handschin, Edmund ; Krause, Olav: A Distributed Real-Time Multi-Agent System for Decentralized Management of Electric Power. In: *Proceedings of the 18th Euromicro Conference on Real-Time Systems*. Dresden : IEEE, 2006. – ISBN 0–7695–2619–5, S. 1–10

[Woo02] Wooldridge, Michael: *An Introduction to MultiAgent Systems*. 1. Hoboken, New Jersey : Wiley Publishing, 2002. – 365 S. – ISBN 0470519460

[Yu195] *Modelling strategic relationships for process reengineering*, University of Toronto, PhD Thesis, 1995. `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.415.2739{&}rep=rep1{&}type=pdf`. – 131 S.

[ZHB18] Zhang, Yang ; Huang, Tao ; Bompard, Ettore F.: Big data analytics in smart grids : a review. In: *Energy Informatics* 1 (2018), Nr. 1, S. 1–24. `http://dx.doi.org/10.1186/s42162-018-0007-5`. – DOI 10.1186/s42162–018–0007–5