

Reihe 20

Rechnerunter-
stützte Verfahren

Nr. 476

M.Sc. Christian von Trotha,
Aachen

Entwicklung eines Konzepts zur Kontextualisierung von Prozessdaten

ACPLT
AACHENER
PROZESSLEITTECHNIK

Lehrstuhl für
Prozessleittechnik
der RWTH Aachen

Entwicklung eines Konzepts zur Kontextualisierung von Prozessdaten

Von der Fakultät für Georessourcen und Materialtechnik der
Rheinisch-Westfälischen Technischen Hochschule Aachen

zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften

genehmigte Dissertation

vorgelegt von

M.Sc. Christian von Trotha

aus Offenbach am Main

Berichter: Univ.-Prof. Dr.-Ing. Ulrich Epple
Univ.-Prof. Dr.-Ing. Christian Diedrich
Univ.-Prof. Dr.-Ing. Tobias Kleinert

Tag der mündlichen Prüfung: 23.04.2021

Fortschritt-Berichte VDI

Reihe 20

Rechnerunterstützte
Verfahren

M.Sc. Christian von Trotha,
Aachen

Nr. 476

Entwicklung eines
Konzepts zur
Kontextualisierung
von Prozessdaten

von Trotha, Christian

Entwicklung eines Konzepts zur Kontextualisierung von Prozessdaten

Fortschr.-Ber. VDI Reihe 20 Nr. 476. Düsseldorf: VDI Verlag 2021.

188 Seiten, 82 Bilder, 12 Tabellen.

ISBN 978-3-18-347620-6, ISSN 0178-9473,

€ 67,00/VDI-Mitgliederpreis € 60,30.

Für die Dokumentation: Kontextualisierung – Datenbasierte Entscheidung – Wissensgraph – Vernetzung von Informationen – Prozessdaten – Assistenzsystem – Graphdatenbank

Keywords: Contextualization – data-based decision – knowledge graph – interlinked information – process data – operator support system – graph database

Die vorliegende Arbeit wendet sich an Ingenieur:innen und Wissenschaftler:innen aus der Prozessindustrie. Sie befasst sich mit der Vernetzung und der Verknüpfung von verschiedenen Prozessdaten. Grundlage dafür sind die in diversen technischen Dokumenten, wie zum Beispiel dem R&I-Diagramm, verfügbaren Strukturinformationen. Dank der Vernetzung bzw. der Verknüpfung können die verfügbaren Prozessdaten deutlich effizienter gefiltert und durchsucht werden. Dadurch kann der situationsbezogene Kontext bereitgestellt werden und als Entscheidungsgrundlage dienen. Kern der Arbeit ist eine Modellhierarchie, die die Modellierung der Abhängigkeiten in einem Wissensgraphen ermöglicht. Demonstriert wird das Konzept anhand der Laboranlage des Lehrstuhls. Grundlage des Demonstrators ist die Graphdatenbankplattform Grakn. Verschiedene Verwaltungsschalen, BaSys40-Komponenten und PandIX dienen dabei als Informationsquellen.

Bibliographische Information der Deutschen Bibliothek

Die Deutsche Bibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliographie; detaillierte bibliographische Daten sind im Internet unter www.dnb.de abrufbar.

Bibliographic information published by the Deutsche Bibliothek

(German National Library)

The Deutsche Bibliothek lists this publication in the Deutsche Nationalbibliographie (German National Bibliography); detailed bibliographic data is available via Internet at www.dnb.de.

D82 (Diss. RWTH Aachen University, 2021)

© VDI Verlag GmbH · Düsseldorf 2021

Alle Rechte, auch das des auszugsweisen Nachdruckes, der auszugsweisen oder vollständigen Wiedergabe (Fotokopie, Mikrokopie), der Speicherung in Datenverarbeitungsanlagen, im Internet und das der Übersetzung, vorbehalten.

Als Manuskript gedruckt. Printed in Germany.

ISSN 0178-9473

ISBN 978-3-18-347620-6

Vorwort

Die vorliegende Dissertation entstand während meiner Tätigkeit am Lehrstuhl für Prozesstechnik der RWTH Aachen. An dieser Stelle möchte ich mich bei allen bedanken, die mir geholfen haben, sie erfolgreich abzuschließen.

Besonderer Dank gilt Herrn Prof. Dr.-Ing. Ulrich Epple für die ermöglichten Freiräume, die konstruktiven Diskussionen und die förderliche Arbeitsatmosphäre am Lehrstuhl. Sowohl der intensive fachliche Austausch als auch das kreative Arbeitsumfeld stellen die Basis für den erfolgreichen Abschluss dieser Arbeit dar.

Herrn Prof. Dr.-Ing. Christian Diedrich danke ich für die Übernahme des Koreferats und Herrn Prof. Dr.-Ing. Pfeifer für die Übernahme des Prüfungsvorsitzes.

Herrn Prof. Dr.-Ing. Tobias Kleinert danke ich für die konstruktiven Hinweise zu meiner Arbeit und wünsche viel Erfolg bei der Fortführung des Lehrstuhls.

Meinen ehemaligen Kolleg:innen danke ich für die anregenden Diskussionen, den intensiven Austausch und die angenehme sowie produktive Arbeitsatmosphäre. Ganz besonderer Dank gilt Frau Margarete Milescu für die Unterstützung bei der Navigation durch die vielen administrativen Hürden des Unialltags.

Bei meiner Frau Almut möchte ich mich von Herzen für die unendliche Geduld und Unterstützung bedanken. Außerdem möchte ich mich bei meinem Freund Markus bedanken, der mir immer mit Rat und Tat zur Seite stand.

Abschließend möchte ich mich ganz besonders bei meinen Eltern Sabine und Eckhard und meinen Schwiegereltern Grete und Peter bedanken. Ohne die liebevolle und großzügige Unterstützung meiner Eltern wäre diese Arbeit nicht möglich und ohne die herzliche und kulinarische Unterstützung meiner Schwiegereltern nur halb so angenehm gewesen.

Vielen Dank!

Aachen, im Mai 2021

Christian v. Trotha

Inhaltsverzeichnis

Abkürzungen und Symbole	VIII
Kurzfassung	X
Abstract	XI
1 Einleitung	1
1.1 Motivation	1
1.2 Vorgehensweise	4
1.2.1 Methoden zur Ontologieentwicklung aus der Literatur	4
1.2.2 Vorgehensweise und Struktur der Arbeit	6
2 Definition von Zielszenarien	7
2.1 Verknüpfung von Informationen zu Wissen	7
2.2 Aufdeckung von bisher nicht modellierten Zusammenhängen	7
2.3 Erstellung von Datensubsets zur Datenanalyse	8
2.4 Validierung von Ergebnissen einer Datenanalyse	9
2.5 Analyse der Zielszenarien	9
3 Grundlagen	10
3.1 Eigene Vorarbeiten	10
3.1.1 Assistenzsysteme	10
3.1.2 Strukturmodellierung	11
3.1.3 Steuerungsarchitektur	11
3.2 Datenqualität	13
3.3 Modelle und Modellbildung	14
3.4 Wissensrepräsentation	18
3.4.1 Semantische Netze	22
3.4.2 Graphdatenbanken	23
3.5 Definition des Begriffs Kontext für die Prozessindustrie	25
3.6 Ursache-Wirkungsanalyse	28
3.7 Verwandte Ansätze aus der Literatur	29
4 Bestehende Modellierungskonzepte und Informationsquellen	33
4.1 Systeme	33
4.1.1 Systemmodellierung	34
4.1.2 Rollenmodell	35
4.1.3 Konzeptuelles Datenmodell der ISO 15926	35
4.1.4 Grakn-Metamodell	37

4.2	Datenquellen	37
4.2.1	Strukturmetamodelle	37
4.2.2	Verwaltungsschale	44
4.2.3	Prozesshistorian	45
4.3	Ereignisse	46
4.3.1	Abnormale Situationen	46
4.3.2	Ereignistypen	47
4.4	Diskussion der Informationsquellen	48
5	Anforderungsanalyse und -spezifikation	49
5.1	Analyse der Randbedingungen aus dem Anwendungsgebiet	49
5.1.1	Klassische Randbedingungen der Prozessindustrie	50
5.1.2	Neue Randbedingungen durch Industrie 4.0	51
5.2	Anforderungen an das Konzept	51
5.3	Anforderungen an die Modellhierarchie	54
5.4	Anforderungen des Konzepts an die Umwelt	55
6	Entwicklung der Systemstruktur	57
6.1	Definition der Systemarchitektur	57
6.2	Entwicklung der Modellhierarchie	59
6.3	Beziehungsmodell	59
6.3.1	Konzept	59
6.3.2	Anwendungsbereich	60
6.3.3	Allgemeine Spezifikation	60
6.3.4	Formale Spezifikation	60
6.3.5	Diskussion des Beziehungsmodells	62
6.4	Spezifikation des Metametamodells	62
6.4.1	Objekt (object)	63
6.4.2	Subjekt (subject)	63
6.4.3	Eigenschaften (property)	63
6.4.4	Lebenszyklus (lifecycle)	64
6.4.5	Ereignis (event)	65
6.4.6	Kontextbeziehung (context relation)	65
6.4.7	Entität (entity)	66
6.4.8	Rollen (role)	67
6.4.9	Anwendung des Metametamodells	67
6.5	Aspektmetamodelle	68
6.5.1	Aspektmetamodell Topologie	69
6.5.2	Aspektmetamodell Messen	73
6.5.3	Aspektmetamodell Deployment	75
6.5.4	Aspektmetamodell Steuerungsstruktur	77
6.5.5	Aspektmetamodell Hardware-Software-Kopplung	80
6.6	Mögliche Erweiterungen zur Vereinfachung des Datenzugriffs	81
6.6.1	Definition von Perspektiven	81
6.6.2	Analyse der Wirkrichtung der Beziehungen	81
6.7	Definition eines Prozesses zur modularen Erweiterung	83

6.8	Evaluation der Modellhierarchie	84
6.8.1	Evaluation anhand der Anforderungen an die Modellhierarchie . . .	84
6.8.2	Evaluation anhand der Anforderungen an das Konzept	85
6.8.3	Klassifikation als Assistenzsystem	86
7	Realisierung und prototypische Implementierung	88
7.1	Demonstration am Pumpwerk	88
7.1.1	Datenquelle: openAAS-Server	89
7.1.2	Datenquelle: Prozesshistorian	91
7.1.3	Datenquelle: Einzelsteuereinheiten	91
7.1.4	Datenquelle: Engineering	92
7.2	Verwendete Werkzeuge	92
7.2.1	GRAKN.AI	92
7.2.2	ACPLT/RTE	95
7.3	Modellierung	97
7.3.1	Abbildung des Metametamodells auf das Grakn-Metamodell	97
7.3.2	Abbildung der Aspektmetamodelle auf das Grakn-Metamodell . . .	102
7.3.3	Abbildung der Datenquellen auf die Aspektmetamodelle	102
8	Evaluation der Ergebnisse	103
8.1	Evaluation anhand der Anforderungen	103
8.2	Evaluation anhand der Anwendungsszenarien	105
8.2.1	Verknüpfung von Informationen zu Wissen	105
8.2.2	Aufdeckung von bisher nicht modellierten Zusammenhängen	108
8.2.3	Erstellung von Datensubsets zur Datenanalyse	110
8.2.4	Validierung von Ergebnissen der Datenanalyse	110
8.2.5	Weitere Beziehungstypen	112
9	Diskussion und Ausblick	116
A	Anhang	119
A.1	Pandix-Datei des Pumpwerks	119
A.2	Abbildung der Aspektmetamodelle in Graql	143
A.2.1	Aspektmetamodell Topologie	143
A.2.2	Aspektmetamodell Messen	145
A.2.3	Aspektmetamodell Deployment	146
A.2.4	Aspektmetamodell Steuerungsstruktur	146
A.2.5	Aspektmetamodell Hardware-Software-Kopplung	147
A.2.6	Aspektmetamodell Wirkung	148
A.3	GraphML-Dateien der Nachbarschaft von L26	148
	Literaturverzeichnis	166

Abkürzungen und Symbole

ACPLT/RTE	Aachener Prozessleittechnik/ Run Time Environment
ASM	Abnormal Situation Management
CAE	Computer Aided Engineering
CAEX	Computer Aided Engineering eXchange
CPS	Cyber-Physisches System
DDL	Data Definition Language
DEXPI	Data EXchange in the Process Industry
DML	Data Manipulation Language
DOM	Document Object Model
DYMASOS	DYNAMIC Management of physically coupled System of Systems
ER	Entity-Relationship
ESKAPE	Evolving Sematic Knowledge Aggregation and Processing Engine
JSON	JavaScript Object Notation
LCA	Lebenszykluseintragsarchiv (LiveCycle Entry Archive)
LC.SM	Lebenszyklus-Teilmodell (LifeCycle SubModel)
M2M	Maschine-zu-Maschine-Kommunikation
OMG	Object Management Group
OPC UA	Open Platform Communication Unified Architecture
OWL	Ontology Web Language
PandIX	Pipe and Instrumentation eXchange
PLT	Prozessleittechnik
PPE	ProcessPlantElement
RDF	Resource Description Framework
R&I	Rohrleitungs- und Instrumentenfließschema
RIO	Remote Input/Output
RIVA	Rohrleitungs- und Instrumentenfließschema verfahrenstechnisch aufbereitet
SAX	Simple API for XML
SIC	Systemelement Interface Connection
SoSX	System of System eXchange
SPS	speicherprogrammierbare Steuerung

UML Unified Modeling Language
URI Uniform Resource Identifier
W3C World Wide Web Consortium
XML Extensible Markup Language
XSD XML Schema Definition

Kurzfassung

In der Prozessindustrie werden im Rahmen des Betriebs prozesstechnischer Anlagen mit zunehmendem Umfang Daten erzeugt und abgespeichert. Dabei handelt es sich beispielsweise um Daten aus Planungsprozessen (z. B. das R&I-Fließbild) oder um Prozessdaten (z. B. Messreihen). Für den Austausch und Zugriff auf diese Daten setzen sich standardisierte Schnittstellen und Austauschformate, wie zum Beispiel die Verwaltungsschale, OPC UA, DEXPI oder PandIX immer stärker durch und ermöglichen so eine eindeutige Identifikation und semantische Referenzierung der abgelegten Daten. Darauf aufbauend ist der nächste Schritt auf dem Weg zu datenbasierten Entscheidungen die Aufbereitung und Kontextualisierung dieser Daten. Diese Kontextualisierung und Aufbereitung muss aufwandsarm und mithilfe von standardisierten Werkzeugen erfolgen, damit die Ausnutzung des bestehenden Datenschatzes schnell und effizient erfolgen kann.

Die vorliegende Arbeit adressiert die Herausforderung der Kontextualisierung. Sie beschreibt ein Konzept, das die konsistente Modellierung des Kontextes von Prozessdaten in der Prozessindustrie ermöglicht. Kern der Arbeit ist eine Modellhierarchie, mithilfe derer die Zusammenhänge zwischen einzelnen Daten dargestellt werden können. Auf diesem Weg können die verfügbaren Informationen zu Wissen vernetzt werden. Dabei wird ausgenutzt, dass in der Prozessindustrie einerseits vielfältige Informationen über strukturelle Zusammenhänge, beispielsweise in Form von Schalt-, Bau- und Aufstellungsplänen, Fließbildern oder Steuerungsarchitekturen, beschrieben sind und dass sich andererseits lokale Abhängigkeiten zwischen einzelnen Daten gerade aus diesen Strukturen ergeben. So ermöglicht das Kontextualisierungssystem eine effiziente Suche und Filterung der verfügbaren Daten und erleichtert so die Nutzung dieser.

Technologische Grundlage der Arbeit ist eine Wissensrepräsentation mithilfe eines semantischen Netzes. Semantische Netze repräsentieren Informationen mithilfe von Knoten und Kanten und eignen sich insbesondere zur Beschreibung von Zusammenhängen und Abhängigkeiten zwischen Informationen. Damit bilden sie die ideale Grundlage für die Kontextualisierung. Eine wesentliche Herausforderung beim Einsatz von semantischen Netzen ist die oft fehlende formale Spezifikation der Begriffe und Beziehungen. Diese Herausforderung wird im Rahmen dieser Arbeit und für den Anwendungsfall der Kontextualisierung von Prozessdaten mithilfe der oben genannten Modellhierarchie adressiert.

Die praktische Anwendbarkeit des Konzepts wird mithilfe einer prototypischen Implementierung für das institutseigene Pumpwerk demonstriert. Sie basiert auf der Graphdatenbankplattform Grakn und baut auf verschiedenen Verwaltungsschalen, BaSys40-Komponenten sowie PandIX als Informationsquellen auf.

Abstract

Data is being generated and stored during the operation of process plants. This is, for example, data from planning processes (e.g. the P&ID flow chart) or process data (e.g. measurement series). Standardized interfaces, information models and exchange formats, such as the asset administration shell, OPC UA, DEXPI or PandIX are becoming increasingly popular for the exchange and access to this data. This enables clear identification and semantic referencing of the stored data. The next step to accomplish data-based decisions is the preparation and contextualization of this data. This contextualization and preparation must be carried out with little overhead and using standardized tools so that the existing data can be used easily and efficiently.

This work addresses the challenge of contextualization. It describes a concept that enables consistent modeling of the data context in the process industry. The core of the work is a model hierarchy, which can be used to model the relationships between individual data points. In this way, the available information can be interconnected to form knowledge. The concept takes advantage of the wide range of available information about structural relationships, for example in the form of circuit, construction and installation plans, flow diagrams or control architectures. These structures determine the local dependencies between individual data points. Thereby the contextualization system enables the efficient search and filtering of the available data, making it easier to use them.

The technological basis of the work is a knowledge representation using a semantic network. Semantic networks represent information using nodes and edges. They are particularly suitable for describing relationships and dependencies between data points and form the ideal basis for contextualization. A major challenge of using semantic networks is the lack of formal specification of terms and relationships. For the use case of contextualization of process data this challenge is addressed in this work by employing the model hierarchy mentioned above.

The applicability of the concept is demonstrated by means of a prototype implementation for the lab system “pumping station”. It is based on the Grakn graph database platform and uses various asset administration shells, BaSys40-components and PandIX as information sources.

1 Einleitung

1.1 Motivation

Der Vergleich, Daten seien das Öl des 21. Jahrhunderts, wird oft herangezogen, um die neu gewonnene Bedeutung von Daten für die heutige Gesellschaft hervorzuheben. Von diesem Wandel ist auch die Prozessindustrie betroffen. Gleichzeitig nimmt die Komplexität eingesetzter Automatisierungslösungen kontinuierlich zu [1] und die vierte industrielle Revolution steht unweigerlich bevor, sodass die Industrie vor vielfältigen Herausforderungen steht. Durch die zunehmende Vernetzung, die Höchstautomatisierung sowie die Entwicklung autonomer Überwachungs- und Entscheidungsprozesse entstehen neue Möglichkeiten für die Steuerung und die Optimierung ganzer Wertschöpfungsnetzwerke [2]. Basis dieser Entwicklung sind Cyber-Physische Systeme (CPS), die dank Maschine-zu-Maschine-Kommunikation (M2M) über das Internet komplexe Netzwerke bilden, eigenständig Informationen austauschen und in Kooperation Produktionsprozesse durchführen [2]. Die daraus resultierende horizontale und vertikale Integration der verschiedenen IT-Systeme führt zu einer Auflösung der strengen Ebenenstruktur der Automatisierungspyramide hin zu einer über die Ebenen vernetzten Systemarchitektur [3]. Darüber hinaus führt die angestrebte Flexibilisierung zu dynamischen Änderungen dieser Systemarchitektur, sodass sich die Überwachung dieser - sich dynamisch ändernden und komplex strukturierten - Systemnetzwerke immer herausfordernder gestaltet. Diese Veränderungen führen zu neuen Fragestellungen bezüglich der Handhabung der in den unterschiedlichen Systemen anfallenden Daten, denn „nur wer aktuelle Daten zeitnah zu aussagekräftigen Informationen verarbeiten kann, [...] ist in der Lage, fundierte Entscheidungen zu treffen und zum richtigen Zeitpunkt Maßnahmen einzuleiten“ [4, S. 271].

Die Umsetzungsstrategie Industrie I4.0 der Plattform Industrie 4.0 sieht in der Datenanalyse eine wesentliche „Querschnittstechnologie“ zur Erreichung der gesetzten Ziele [5]. Mithilfe der Datenanalyse können unter anderem Zusammenhänge aufgedeckt, (autonome) Entscheidungen unterstützt sowie Fehler frühzeitig erkannt werden [5]. Der hierfür notwendige Prozess, der ausgehend von verfügbaren Daten zu einer erkenntnisbasierten Handlung führt, ist in Abbildung 1.1 dargestellt.

Ein Großteil der Daten einer typischen Anlage der Prozessindustrie wird über eine große Anzahl Feldgeräte erzeugt, deren Messwerte in aller Regel über große Zeiträume archiviert und gespeichert werden [3]. Außerdem werden in einer Vielzahl unterschiedlicher

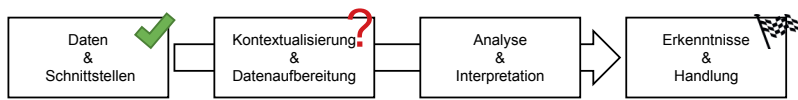


Abbildung 1.1: Von Daten zur erkenntnisbasierten Handlung

IT-Systeme Ereignisse, wie zum Beispiel Meldungen, Warnungen, Alarme, Bedieneingriffe, aber auch Prozessparameter oder Produktionsaufträge erfasst [3]. Allerdings werden dabei die Beziehungen zwischen diesen einzelnen Informationen in der Regel nicht systematisch mit abgelegt [6].

Für eine erfolgreiche Datenanalyse (z. B. Value Based Service [7]) müssen die drei oben genannten Informationsklassen Systemarchitektur, Messwert und Ereignis kombiniert, also die Beziehungen zwischen den einzelnen Informationen offengelegt werden. So würde beispielsweise „die gemeinsame Auswertung und Aggregation dieser bisher häufig unabhängig voneinander betrachteten Daten und den daraus potentiell zu ermittelnden zusätzlichen Informationen (Datenmustern) [...] einen optimierten Anlagenbetrieb und ein verbessertes Anlagen- und Gerätedesign ermöglichen“ [3, S. 162]. Dabei ist eine manuelle Auswertung der Daten aufgrund der zuvor beschriebenen Komplexität (dynamische Änderungen, heterogene IT-Systeme etc.) nur sehr eingeschränkt möglich. Dies zeigt sich beispielsweise auch in den Erfahrungen, die im Bereich des Data Mining gemacht werden. So beschreibt [8], dass bei einem durchschnittlichen Data Mining-Projekt 20-30% des Aufwands für das Verstehen der verfügbaren Daten und 50-70% für die Datenvorverarbeitung aufgewendet werden müssen. Damit der verfügbare Datenschatz in Zukunft effizient gehoben werden kann, ist eine standardisierte und aufwandsarme Kontextualisierung der Prozessdaten notwendig [9]. Als Grundlage für eine solche Kontextualisierung bedarf es geeigneter Zugriffspunkte bzw. Beschreibungen für die maschinelle Auswertung der unterschiedlichen Informationsklassen.

Ereignisinformationen werden in der Prozessindustrie „in mehreren hundert IT-Systemen (Prozessleit-, Asset Management-, Engineering-, Laborsystemen und MES)“ [3, S. 162] verwaltet. Diese IT-Systeme sind hoch spezialisiert und für konkrete Anwendungsszenarien entwickelt worden, sodass die Integration aller Ereignisse in ein einheitliches „Ereignissystem“ keine valide Lösung darstellt. Vielmehr scheinen standardisierte Zugriffspunkte, die in einer einheitlichen Syntax alle relevanten Informationen zur Verfügung stellen, das Mittel der Wahl zu sein. Einen solchen einheitlichen Zugriffspunkt stellt die Verwaltungsschale dar [10]. Sie „erlaubt den [...] Zugriff auf alle Informationen des [verwalteten] Gegenstands“ [11, F. 8].

In der Prozessindustrie wird auch heutzutage schon eine große Anzahl an Messwerten aufgezeichnet und archiviert. Dies geschieht z. B. auch zur Erfüllung regulatorischer Anforderungen [3] und mithilfe dedizierter Archivsysteme. Beispiele für kommerziell verfügbare Produkte sind das OSIsoft PI System¹, die Plant Historian PDA² oder der SIMATIC Process Historian³. All diese Produkte haben gemein, dass sie Messwerte der Sensoren als sogenannte Zeitreihen archivieren. Darüber hinaus können einige dieser Systeme (z. B. Plant Historian PDA oder SIMATIC Process Historian) Meldungen aus dem Prozessleitsystem in das Archiv integrieren.

Zur Beschreibung der Anlagenstruktur bzw. der Systemarchitektur bestehen in der Prozessindustrie eine große Anzahl unterschiedlicher Modelle. Das gängigste Modell zur Anlagenbeschreibung ist sicherlich das Rohrleitungs- und Instrumentenfließschema (R&I), das alle für den Anlagenbetrieb erforderlichen Geräte (z. B. Ventile, Pumpen oder Behälter), die dazugehörigen Rohrleitungen und die benötigten PLT-Stellen abbildet. Grundsätzlich ist das R&I eine graphische Darstellung der Anlage, für die allerdings in Form von

¹<https://www.osisoft.de/>

²<https://www.imes-solutions.com/de/root/industrie-4-0-mes-loesungen.html#plant-historian-pda>

³<https://w3.siemens.com/mcms/automation-software/de/scada-software/scada-optionen/simatic-process-historian/seiten/default.aspx>

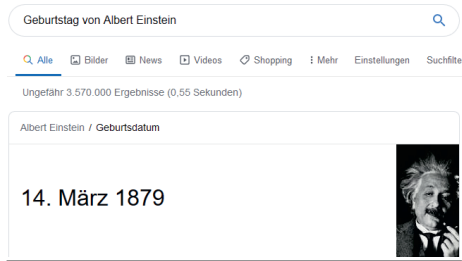


Abbildung 1.2: Ergebnis der Google-Suche nach dem Geburtstag von Albert Einstein

Pipe and Instrumentation eXchange (PandIX) oder Data EXchange in the Process Industry (DEXPI) offene und maschinenlesbare Austauschformate bzw. Metamodelle vorliegen. Neben PandIX und DEXPI bestehen weitere Metamodelle zur Beschreibung der Anlagenstruktur, wie zum Beispiel System of System eXchange (SoSX) [12]. Weitere Strukturen können sich aus den Elektroschaltplänen, den Signalverbindungen, der Energieversorgung, den Bau- und Aufstellungsplänen oder der Architektur der Automatisierungslösung ergeben.

Da einheitliche Zugriffspunkte für Ereignisinformationen, Archivsysteme für Zeitreihen und Metamodelle zur Strukturbeschreibung bestehen (siehe [6]), ist die zentrale Fragestellung dieser Arbeit, ob es möglich ist, diese drei Datenquellen zu kombinieren und so eine effiziente Datenanalyse zu ermöglichen. Dabei können die Strukturinformationen als Bindeglied zwischen den Zeitreihen und den Ereignisinformationen dienen und so den Kontext einzelner Daten bereitstellen.

In der Praxis haben sich Graphdatenbanken als Technologie für die Verknüpfung von unterschiedlichen Daten bzw. zur Verwaltung von verknüpften Daten durchgesetzt [13]. So nutzt beispielsweise Google eine als Knowledge Graph bezeichnete Wissensdatenbank⁴ zur direkten Beantwortung von gestellten Suchanfragen [14]. Wird zum Beispiel, wie in Abbildung 1.2 gezeigt, nach dem Geburtstag von Albert Einstein gesucht, wird dank des Knowledge Graph unmittelbar die Antwort „14. März 1879“ angezeigt, ohne dass eine weitere Webseite aus den Suchergebnissen aufgerufen werden muss [14]. Der zur Beantwortung dieser Fragestellung notwendige Vorgang kann in zwei Schritte unterteilt werden. Zunächst muss die natürlichsprachlich gestellte Frage vom System verstanden und anschließend die zuvor aufbereitete und in der Wissensdatenbank hinterlegte Antwort herausgesucht werden. Dementsprechend ist eine Schlüsselaufgabe, um Wissen in dieser Art und Weise bereitstellen zu können, das entsprechende Aufbereiten, Kontextualisieren und Vernetzen der verfügbaren Daten.

Zentrale Herausforderung dabei ist der Umstand, dass während der Erzeugung und der Kontextualisierung der Daten die spätere Verwendung oder die untersuchten Fragestellungen nicht bekannt sind. Deshalb muss zur Kontextualisierung ein metamodellbasierter Ansatz gewählt werden, der flexibel erweitert werden kann und eine kohärente Beschreibung der Daten erlaubt. Aktuell existiert kein „Weltmodell“ zur Beschreibung aller aktuellen und zukünftigen Daten, sodass es eines Konzepts bedarf, das aktuelle Teilmodelle integrie-

⁴https://en.wikipedia.org/wiki/Knowledge_Graph

ren und in Zukunft einfach um weitere Aspekte und Datenquellen erweitert werden kann. Dementsprechend muss ein Informationsmodell definiert werden, das eine einfache Abbildung und Integration der Daten aus den einzelnen Informationsquellen ermöglicht. Das Ziel ist nämlich nicht die Eins-zu-Eins-Abbildung der bestehenden Datenmodelle, sondern das geschickte Verweben der einzelnen Informationsstränge.

1.2 Vorgehensweise

Wie zuvor beschrieben, ist die Kernaufgabe um zu einer strukturierten Datenablage mithilfe einer Graphdatenbank zu gelangen, die Entwicklung eines Informationsmodells, das eine einfache Integration aller relevanten Daten ermöglicht. Nach [15] handelt es sich bei einem Informationsmodell um eine Ontologie (siehe Abschnitt 3.3 bzw. Abbildung 3.3). Außerdem können Ontologien nach [16] die konzeptuelle und technologische Basis für Informationsmodelle darstellen. Daher eignen sich Methoden zur Entwicklung von Ontologien auch für die Entwicklung von Informationsmodellen. Aus diesem Grund werden in diesem Abschnitt zunächst Methoden zur Entwicklung von Ontologien betrachtet und im Anschluss die daraus resultierende Vorgehensweise für diese Arbeit abgeleitet. Abschließend werden die einzelnen Kapitel der Arbeit den einzelnen Schritten der Vorgehensweise zugeordnet.

1.2.1 Methoden zur Ontologieentwicklung aus der Literatur

In [16] und [17] wird jeweils eine Methodik zur Ontologieentwicklung vorgestellt. Hildebrand et. al. fokussieren in [16] die Anwendungsdomäne der CPS, während [17] einen allgemeingültigen Ansatz beschreibt. Die grundlegenden Methodiken von [17] und [16] stimmen in großen Teilen überein. Beide Methodiken beginnen mit der Definition von Anforderungen und Zielen. Darauf folgt die Definition der Ontologie durch Analyse der innerhalb der betrachteten Domänen relevanten Begriffe und Beziehungen unter Berücksichtigung bestehender Konzepte und Standards. Anschließend erfolgt eine formale Beschreibung der Ontologie. Uschold et. al. benennen zusätzlich die Evaluation anhand der Anforderungen und die Dokumentation der Ontologie als explizite Schritte der Methodik.

Die von [16] beschriebene Entwurfsmethode ist in Abbildung 1.3 dargestellt. Kern der Vorgehensweise sind die drei Prozessschritte Ontology Requirements Specification, Lightweight Ontology Building und Heavyweight Ontology Building. Zusätzlich werden die beteiligten Personengruppen beschrieben. Außerdem gilt, dass die Vorgehensweise in der Regel nicht linear abläuft, sondern Rückschritte und Iterationsschleifen existieren. Im Folgenden werden die drei Prozessschritte sowie die dazugehörigen wesentlichen Aufgabenstellungen kurz erläutert.

Ontology Requirements Specification [16]

In diesem Prozessschritt muss das Ziel bzw. der Zweck und der Umfang der Ontologie definiert, außerdem die geplante Nutzung beschrieben werden. Anschließend können aus diesen Überlegungen Anforderungen abgeleitet und bei Bedarf verifiziert, gruppiert und priorisiert werden. Hildebrand et. al. empfehlen, dass die Definition der Anforderungen in Form von Kompetenzfragen und den gewünschten Antworten erfolgt. Dabei muss die Ontologie

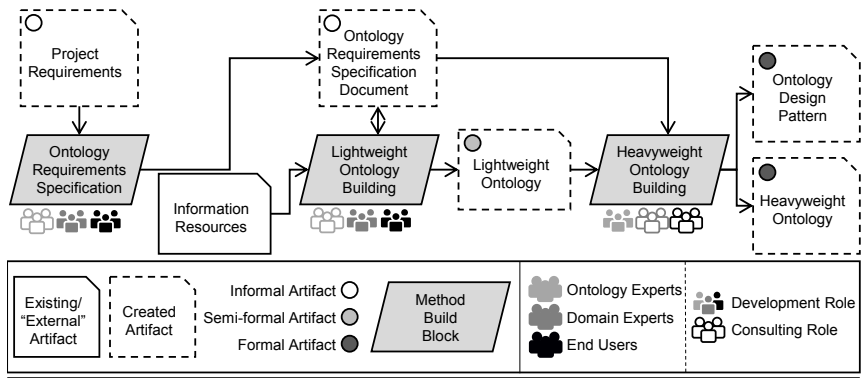


Abbildung 1.3: Methode für den Entwurf von Ontologien für die Domänen CPS aus [16]

nach Abschluss der Entwicklung in der Lage sein, diese Fragen korrekt zu beantworten. Damit definieren die Kompetenzfragen die späteren Kompetenzen des zu entwickelnden Systems.

Lightweight Ontology Building [16]

In diesem Prozessschritt wird eine leichtgewichtige Ontologie konstruiert. Dazu müssen zunächst mögliche Informationsquellen gefunden werden, die die Beantwortung der im vorherigen Schritt definierten Kompetenzfragestellung ermöglichen. Anschließend müssen diese hinsichtlich ihrer Eignung für die Konstruktion der Ontologie untersucht werden. Bei der Kombination von Informationsmodellen unterschiedlicher Informationsquellen gibt es grundsätzlich vier mögliche Operationen:

- **Äquivalent zu:** Die Begriffe zweier Informationsmodelle entsprechen einander.
- **Ist eine Spezialisierung von:** Ein Begriff aus einem Informationsmodell ist eine Spezialisierung eines Begriffs aus einem anderen Informationsmodell.
- **Steht in Beziehung zu:** Zwei Begriffe aus unterschiedlichen Informationsmodellen stehen in einer beliebigen Beziehung zueinander.
- **Ist eine Eigenschaft von:** Ein Begriff eines Informationsmodells ist eine Eigenschaft eines Begriffs aus einem zweiten Informationsmodell.

Mithilfe dieser vier Operationen können einzelne Informationsmodelle zu sogenannten Clustern zusammengesetzt werden. Anschließend können die ausgewählten Cluster zu einer leichtgewichtigen Ontologie zusammengesetzt und beispielsweise in Form eines Unified Modeling Language (UML)-Klassendiagramms dokumentiert werden.

Heavyweight Ontology Building [16]

In diesem Prozessschritt wird die leichtgewichtige in eine schwergewichtige Ontologie überführt. Dazu wird eine formalisierte Beschreibung erstellt. Hildebrandt et. al. schlagen

hierfür die Verwendung der Ontology Web Language (OWL) vor. Anschließend können die zuvor formulierten Kompetenzfragen in ein entsprechendes Query übersetzt und die entwickelte Ontologie validiert werden.

1.2.2 Vorgehensweise und Struktur der Arbeit

Die vorliegende Arbeit folgt grundsätzlich den aus der Literatur vorgestellten Methoden. Einen Überblick über die Vorgehensweise gibt Abbildung 1.4. In der Abbildung sind die einzelnen Kapitel den Prozessschritten aus der Vorgehensweise nach [16] zugeordnet. Außerdem wird gemäß der Methodik nach [17] die Evaluation berücksichtigt. Es werden also in Kapitel 2 zunächst Zielszenarien definiert und benötigte Informationsklassen identifiziert. Darauf aufbauend erfolgt in Kapitel 4 eine Analyse möglicher Informationsquellen und Modellierungskonzepte. Anschließend erfolgt in Kapitel 5 eine erneute Betrachtung und Verfeinerung der Anforderungen. Darauf aufbauend wird in Kapitel 6 die leichtgewichtige Ontologie definiert. Diese wird in Kapitel 7 formalisiert und so in eine schwergewichtige Ontologie überführt. Abschließend erfolgt in Kapitel 8 die Evaluation anhand der in Kapitel 5 definierten Anforderungen und den in Kapitel 2 gestellten Kompetenzfragen. Zusätzlich liefert Kapitel 3 einen Überblick über die zugehörigen Grundlagen. In Kapitel 9 erfolgt eine Diskussion der Ergebnisse. Außerdem wird ein Ausblick auf weitere Forschungsfragen gegeben.

Der Autor dieser Arbeit übernimmt entsprechend der vorgestellten Vorgehensweise vornehmlich die Rolle des Ontologieexperten, da ein grundsätzliches Modellierungskonzept entwickelt wird, das bei Bedarf in Zukunft von Domäneexperten um weitere Aspekte ergänzt werden kann.

Das Vorgehen unterscheidet sich in zwei Punkten von der in [16] vorgestellten Methodik. Einerseits wird im Rahmen dieser Arbeit eine Modellhierarchie konstruiert. Das heißt, dass zunächst der grundlegende Kern (hier: dynamische Beziehungen) der Informationsmodellierung beschrieben wird. Im Anschluss werden alle weiteren Aspekte auf diesen Kern abgebildet. Es wird also zunächst ein Metabegriffscluster gebildet, der im Anschluss zur Konstruktion weiterer Cluster verwendet wird. Andererseits ist das Ziel dieser Arbeit die Konstruktion eines Wissensgraphen. Aus diesem Grund werden für die Implementierung Werkzeuge verwendet, die eine native Unterstützung von Graphen bieten. Dementsprechend erfolgt die formalisierte Beschreibung der Modellhierarchie mithilfe eines solchen Werkzeugs (hier: GRAKN.AI bzw. Graql).

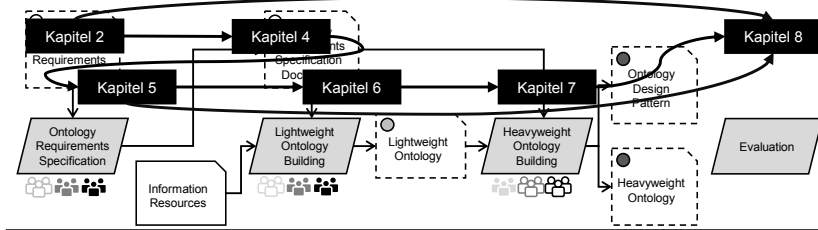


Abbildung 1.4: Überblick über die Vorgehensweise und Struktur dieser Arbeit

2 Definition von Zielszenarien

Entsprechend der vorgestellten Vorgehensweise zur Ontologieentwicklung (siehe Abschnitt 1.2) und um eine spätere Evaluation dieser Arbeit zu ermöglichen, werden in diesem Abschnitt verschiedene Zielszenarien und entsprechende Kompetenzfragen definiert, welche mit dem vorgestellten Konzept erreicht bzw. beantwortet werden sollen. Dabei wird versucht, diese Szenarien relativ allgemein zu definieren, sodass später ein weites Feld möglicher Anwendungsfälle abgedeckt werden kann. Außerdem wird grob benannt, welche Informationen zur Beantwortung der Kompetenzfragen benötigt werden. Ein Teil der Szenarien hat eine große Übereinstimmung mit den in [6] beschriebenen möglichen Anwendungsgebiete für die Kontextualisierung.

Die definierten Zielszenarien lassen sich grob zwei unterschiedlichen Gruppen zuordnen. So dienen erstere eher als Grundlage für Assistenzsysteme im operativen Betrieb einer Anlage, letztere beschreiben vornehmlich die Nutzung des Kontextualisierungssystems als Grundlage zur Datenanalyse.

2.1 Verknüpfung von Informationen zu Wissen

Das einfachste und grundlegendste Zielszenario des Kontextualisierungssystems ist die Verknüpfung von Informationen zu Wissen durch die Bereitstellung des Kontextes. Dabei soll das System Fragen beantworten, wie beispielsweise „Wie hoch ist die Temperatur in Behälter Y?“, „Welcher Druck herrscht vor und nach Pumpe N18?“, „Wie kann von Behälter X in den Reaktor Y gepumpt werden und welche Kommandos bieten die dazugehörigen Einzelsteuerungseinheiten an?“ oder „Welche Aktuatoren wurden von einer konkreten Rezeptinstanz belegt?“. Eine beispielhafte Verwendung für diese Fähigkeit ist ein Assistenzsystem für das Wartungspersonal einer Anlage. Dieses könnte mithilfe eines solchen Werkzeugs über die Verknüpfungen intuitiv Informationen aus dem Kontext ihres Betrachtungsgegenstands abfragen. Alternativ könnte zum Beispiel über eine Rezeptinstanz die Integration von Prozessdaten mit der Produktqualität erfolgen (vgl. [6]).

Zur Beantwortung der Kompetenzfragen dieses Zielszenarios werden also Informationen über die Anlagenstruktur, die aktuellen und historischen Messwerte der Sensoren sowie Informationen zum Automatisierungskonzept (z. B. Rezeptinstanzen oder Einzelsteuereinheiten) der Anlage benötigt. Außerdem ist die Semantik der einzelnen Begriffe von großer Bedeutung. Sie müssen Teil einer Ontologie sein.

2.2 Aufdeckung von bisher nicht modellierten Zusammenhängen

Durch die Verknüpfung der unterschiedlichen Informationen aus den verschiedenen Datenquellen soll das Konzept das Aufdecken von bisher nicht modellierten Zusammenhängen

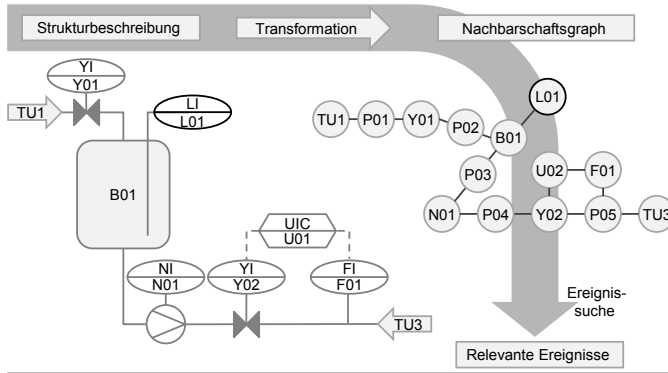


Abbildung 2.1: Zielszenario: Aufdeckung von bisher nicht modellierten Zusammenhängen

ermöglichen. Konkretes Ziel dieser Arbeit ist das Verknüpfen von diskreten Ereignissen, wie beispielsweise das Öffnen oder Schließen eines Ventils oder die Kalibrierung eines Reglers mit den Zeitreihen der Sensoren. Unter der Annahme, dass zwischen den einzelnen Zeitreihen und den zugehörigen Ereignissen ein örtlicher bzw. logischer Zusammenhang besteht, soll eine automatische Suche nach den relevanten Ereignissen ermöglicht werden. Eine vereinfachte Darstellung des Ansatzes ist in Abbildung 2.1 gegeben. Die Abbildung zeigt die Strukturbeschreibung einer einfachen Anlage, die mithilfe einer Transformation in einen Nachbarschaftsgraphen überführt wurde. Dieser ermöglicht die Suche nach relevanten Ereignissen. Sollte sich zum Beispiel der Füllstand L01 auffällig verhalten, können mithilfe des Nachbarschaftsgraphen die Verwaltungsschalen der Elemente in der Umgebung des Füllstandsensors nach Ereignissen durchsucht werden. Auf diese Art und Weise können dem Nutzenden Informationen automatisiert zur Verfügung gestellt und der Kontext der Zeitreihen ersichtlich gemacht werden. Ein möglicher konkreter Anwendungsfall für dieses Szenario könnte beispielsweise das in [6] konzeptionell beschriebene Alarmmanagementsystem sein, das basierend auf dem örtlichen bzw. logischen Zusammenhang Alarme miteinander verknüpft und so hilft Alarmfluten zu reduzieren.

In diesem Szenario steht die Frage im Mittelpunkt, welche Ereignisse im Kontext der beobachteten Auffälligkeit aufgetreten sind. Die Grundlage zur Beantwortung dieser Frage stellen die aktuellen und historischen Messwerte der Sensoren, die Struktur der Anlage, die verschiedenen Ereignisse sowie die Semantik der einzelnen Begriffe dar.

2.3 Erstellung von Datensubsets zur Datenanalyse

Wie bereits zuvor beschrieben, ist eine wesentliche Herausforderung der Datenanalyse das Verstehen der verfügbaren Daten bzw. Informationen und die Datenvorverarbeitung. Das Konzept dieser Arbeit soll hier Erleichterung verschaffen, indem es die automatische Erstellung von Datensubsets erlaubt. Sollen beispielsweise für einen Predictive-Maintenance-Ansatz Pumpenmodelle trainiert werden, wird dazu im Wesentlichen der Druck vor und hinter den Pumpen benötigt. Ziel des Konzepts ist die effiziente Bereitstellung der

benötigten Informationen. In [6] wird dieses Szenario als Apperate- und Gerätediagnose beschrieben.

Es muss also die Frage nach bestimmten Anlagenkonstellationen beantwortet werden. Für das genannte Beispiel stellen sich die Fragen: Vor und hinter welcher Pumpe sind Drucksensoren installiert? Wann war die Pumpe aktiv? Welchen Druck haben die Sensoren in diesem Zeitraum gemessen? Damit diese Fragen beantwortet werden können, müssen die historischen Messwerte der Sensoren, die Struktur der Anlage und Ereignisinformationen zu den Akteuren vorliegen und alle Begriffe Teil einer Ontologie sein.

2.4 Validierung von Ergebnissen einer Datenanalyse

Ein wesentliches Ziel von Datenanalysen ist die Aufdeckung bzw. das Finden von Abhängigkeiten. Mithilfe des Konzepts sollen gefundene Abhängigkeiten durch Analyse der modellierten Abhängigkeiten und Nachbarschaften validiert und erklärt werden können.

Wird zwischen A und B eine Abhängigkeit vermutet, kann der Graph nach entsprechenden Beziehungen zwischen A und B durchsucht werden. Diese Beziehungen können einen Ansatz zur Erklärung der Abhängigkeiten liefern. Dazu muss die Frage, welche Beziehungen zwischen A und B bestehen, beantwortet werden. Damit werden zur Beantwortung dieser Frage Informationen zu den unterschiedlichen Beziehungen innerhalb der Anlage benötigt. Diese ergeben sich aus der Anlagenstruktur, aus verschiedenen Ereignissen und aus der Typisierung der Beziehungen bzw. der Begriffe.

2.5 Analyse der Zielszenarien

Tabelle 2.1 gibt einen Überblick über die zur Beantwortung der gestellten Kompetenzfragen benötigten Informationsklassen. Außerdem werden zu jeder Informationsklasse einige Beispiele genannt. Dabei ist es wichtig, dass die bereitgestellten Informationen in einem zeitlichen Bezug zu der konkreten Kompetenzfrage stehen. Diese Analyse dient als Grundlage für die Auswahl der Informationsquellen.

Tabelle 2.1: Für die Beantwortung der Kompetenzfragen benötigte Informationen

Informationsklasse	Zielszenario				Beispiele
	1	2	3	4	
Anlagenstruktur	✓	✓	✓	✓	Anlagentopologie, Struktur der Automatisierungslösung
Messwerte	✓	✓	✓		historische und aktuelle Sensorwerte, Ventilposition, Stellgröße
Ereignisse		✓	✓	✓	Öffnen oder Schließen eines Ventils, Belegungsvorgang, Sensorkalibrierung, Aktivierung eines Reglers
Semantik	✓	✓	✓	✓	Zuordnung der einzelnen Begriffe zu Kategorien bzw. Typmodellen
Zeitlicher Bezug	✓	✓	✓	✓	Zeitpunkt, Zeitspanne

3 Grundlagen

Kern dieser Arbeit ist die Kontextualisierung von Zeitreihen und Ereignisinformationen durch Ausnutzung bestehender Strukturbeschreibungen bzw. bestehender Topologiemodelle prozesstechnischer Anlagen. Wesentliches Ziel der Kontextualisierung ist das Verweben und Vernetzen von Informationen aus unterschiedlichen Quellen. Wie zuvor beschrieben, ist die wesentlich Grundlage dafür die Entwicklung eines Informationsmodells, das die Abbildung aller relevanten Informationen innerhalb der Graphdatenbank ermöglicht. Bevor diese Entwicklung erfolgen kann, werden in diesem Grundlagenkapitel zunächst die wesentlichen Begriffe untersucht und beschrieben. Außerdem werden kurz die eigenen Vorarbeiten erläutert und verwandte Konzepte aus der Literatur vorgestellt.

3.1 Eigene Vorarbeiten

Nachfolgend werden kurz die Vorarbeiten des Autors erläutert. Diese sind im Rahmen der Arbeit am Lehrstuhl für Prozessleittechnik der RWTH Aachen University entstanden und können in die drei Bereiche Assistenzsysteme, Strukturmodellierung und Steuerungsarchitektur unterteilt werden.

3.1.1 Assistenzsysteme

Eine Möglichkeit zur Beherrschung von komplexen Automatisierungslösungen sind Assistenzsysteme [18]. Eine wesentliche Herausforderung bei dem Einsatz von Assistenzsystemen ist die gebrauchstaugliche Gestaltung dieser (vgl. [19]), denn schlecht gestaltete Assistenzsysteme können unter anderem zu einem erhöhten Stresslevel [20], reduziertem Situationsbewusstsein oder einer ungünstigen Arbeitsbelastung führen [21–23].

Diese Arbeit folgt der Begriffsdefinition „Assistenzsystem“ aus [22, 23]. Demnach ist eine Assistenzsystem „eine interaktive Schnittstelle, die Komplexität kapselt und so die Handlungsfähigkeit des Nutzers steigert und dementsprechend eine verbesserte Nutzung der Anlage ermöglicht“.

Um eine fundierte Diskussion der unterschiedlichen Gestaltungsalternativen von Assistenzsystemen zu ermöglichen, wurde in [22, 23] ein Rahmenwerk vorgestellt, dass eine einheitliche Beschreibung und Klassifizierung dieser Systeme ermöglicht. Dabei beschreibt das Rahmenwerk sechs bzw. sieben unterschiedliche Stufen der Assistenz. In jeder Stufe wird ein unterschiedlicher Anteil der betrachteten Aufgabenstellung vom Menschen bzw. vom Assistenzsystem übernommen. Dies bedeutet, dass auf der untersten Stufe die Aufgabe vollständig vom Menschen und auf der höchsten Stufe vollständig vom Assistenzsystem übernommen wird. Diese sechs bzw. sieben Stufen werden in dem Rahmenwerk für fünf unterschiedliche Handlungsphasen beschrieben. Dabei kann ein Assistenzsystem in jeder Phase einen anderen Grad der Unterstützung aufweisen. Die Handlungsphasen beschreiben

die unterschiedlichen Schritte, die zur Lösung einer Aufgabenstellung notwendig sind. Ausgangspunkt ist die Aktivierung. In dieser Phase muss der Handlungsbedarf erkannt werden. Darauf folgt die Phase des Beobachtens. Hier müssen verfügbare Daten gesammelt, gefiltert und priorisiert werden. Anschließend folgt eine Orientierung. In dieser Phase müssen die gesammelten Daten analysiert, priorisiert, interpretiert, prädiert und integriert werden. Im der nächsten Phase müssen verschiedene Alternativen bestimmt, priorisiert und ausgewählt werden. Abschließend muss die Entscheidung umgesetzt werden.

Das in [22, 23] vorgestellte Rahmenwerk soll im Rahmen dieser Arbeit als eine Grundlage für die Evaluation des entwickelten Konzepts dienen (siehe Kapitel 6.8.3).

3.1.2 Strukturmodellierung

In [12] und [24] wird sowohl die im Rahmen des DYMASOS¹-Projekts entwickelte Informationsplattform als auch das dazugehörige Metamodell zur Informationsmodellierung beschrieben. Die beschriebene Informationsplattform ist Teil einer Werkzeugkette zur Beschreibung, Modellierung und Optimierung von Verbundstandorten. Dabei bietet die Informationsplattform sowohl die Möglichkeit der Strukturbeschreibung des Verbundstandorts als auch die Möglichkeit zur Integration von Prozessdaten. Die Informationen können anschließend an ein Simulations- und Validierungsframework übergeben werden und bilden dort die Grundlage zur Erprobung von Optimierungs- und Steuerungsstrategien.

Ein semantisch eindeutiger Informationsaustausch wird mithilfe eines Metamodells sichergestellt. Dieses Metamodell, insbesondere das Konzept zur Integration von dynamischen Informationen, stellt eine wesentliche Grundlage für das Konzept dieser Arbeit dar und wird im Folgenden kurz vorgestellt.

Eine erste Übersicht über das Metamodell gibt Abbildung 3.1, in der das Modell als Klassendiagramm dargestellt ist. In der Abbildung sind vier verschiedene Bereiche (I-IV) markiert. Die Begriffe im Bereich I dienen zur Strukturbeschreibung. Zusätzlich bestehen Begriffe (II) zur Integration von Eigenschaften mit entsprechenden Aussagen. Außerdem wird in der Abbildung das Konzept (III) zur Integration von dynamischen Informationen sowie zur Darstellung von zukünftigen Alternativen (IV) gezeigt.

Wie in der Abbildung gezeigt, werden dynamische Informationen mithilfe der beiden Attribute `validFrom` und `validUntil` modelliert. Dabei beschreibt `validFrom` den Beginn und `validUntil` das Ende des Gültigkeitszeitraums des jeweiligen Elements. Innerhalb dieses Zeitraums ist das Element ein Teil des betrachteten und modellierten Verbundsystems. Auf diese Art und Weise ist es möglich Lebenszyklusinformationen in das Modell zu integrieren und diskrete Veränderungen des Modells zu beschreiben.

3.1.3 Steuerungsarchitektur

In [25] werden, basierend auf Softwarequalitätsattributen aus der Literatur, verschiedene Anforderungen an eine Automatisierungslösung von Kaltwalzwerken definiert. In [26] wird darauf aufbauend eine Softwarearchitektur und ein zugehöriger Entwicklungsprozess vorgestellt, die die Erfüllung dieser Anforderungen für Automatisierungslösung ermöglichen. Die vorgestellte Softwarearchitektur baut auf dem in [27] eingeführten Maßnahmen- und Betriebsmittelmodell auf. Die Verwendung einer solchen Softwarearchitektur ermöglicht

¹Dynamic Management of physically coupled System of Systems (DYMASOS)

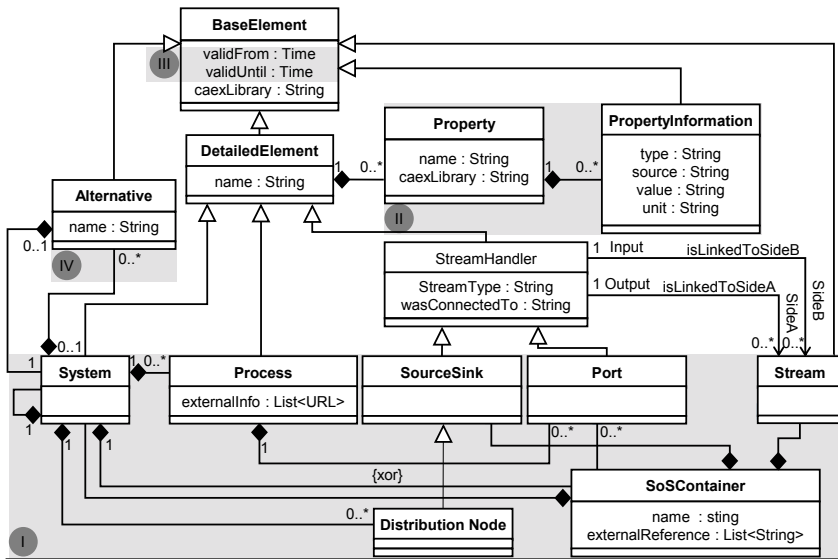


Abbildung 3.1: Klassendiagramm des Metamodells zur Modellierung von Verbundstandorten als physikalisch gekoppeltes System of Systems (I: Strukturbeschreibung, II: Eigenschaften, III: Dynamische Informationen, IV: Zukünftige Alternativen)

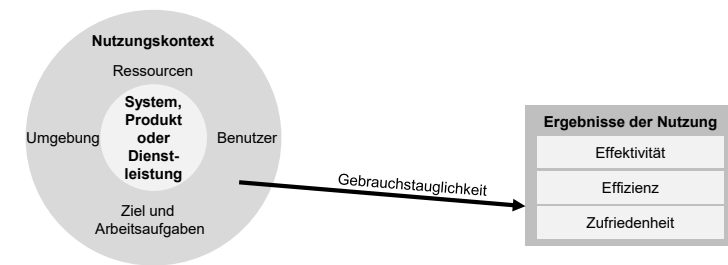


Abbildung 3.2: Gebrauchstauglichkeit ergibt sich aus der relativen Bewertung des Nutzungsergebnisses bezogen auf den Nutzungskontext [19, S. 15]

eine effiziente Erkundung der Abhängigkeiten innerhalb der Automatisierungslösung und stellt damit eine Grundvoraussetzung für die Anwendung des in dieser Arbeit entwickelten Konzepts dar. Weitere Details zu diesem Thema sind in Abschnitt 4.2.1 beschrieben.

3.2 Datenqualität

Die wichtigste Anforderung, die an jedes zu entwickelnde Produkt gestellt wird, ist die Eignung, die gesetzte Aufgabestellung zufriedenstellend lösen zu können. In der DIN EN ISO 9241-11 wird dies mit dem Begriff der Gebrauchstauglichkeit umschrieben. Dabei ist die Gebrauchstauglichkeit als „das Ausmaß, in dem ein System, ein Produkt oder eine Dienstleistung durch bestimmte Benutzer in einem bestimmten Nutzungskontext genutzt werden kann, um bestimmte Ziele effektiv, effizient und zufriedenstellend zu erreichen“ [19, S. 9], definiert. Dargestellt ist dieses Konzept in Abbildung 3.2. Zentrales Element des Konzepts ist der Nutzungskontext. Dieser besteht, wie in der Abbildung dargestellt, aus den vier Elementen Benutzer, Ziel und Aufgabe, Ressourcen sowie Umgebung. Nur in diesem Kontext kann bewertet werden, ob das Produkt für eine effektive, effiziente und zufriedenstellende Nutzung geeignet ist und damit die gestellte Aufgabenstellung lösen kann. Entsprechend der Norm wird dabei das Ziel effektiv erreicht, wenn eine hohe Genauigkeit (Übereinstimmung zwischen angestrebtem und erreichtem Ergebnis) und eine hohe Vollständigkeit (Übereinstimmung der Menge der angestrebten und der erreichbaren Ergebnisse) gegeben ist. Die Effizienz ergibt sich aus dem Verhältnis zwischen den eingesetzten Ressourcen und dem erreichten Ergebnis [19]. Die Zufriedenheit ergibt sich aus physischen, kognitiven und emotionalen Reaktionen des Nutzers auf das Produkt [19].

Eine weitere Perspektive zur Gebrauchstauglichkeit speziell für Softwareprodukte liefert die Norm ISO/IEC 25010:2011 mit der Definition eines Qualitätsmodells für Software [28]. Dabei entspricht die Definition von Qualität in etwa der Definition von Gebrauchstauglichkeit und baut auf denselben Charakteristiken Effektivität, Effizienz und Zufriedenheit auf. Darüber hinaus werden in der ISO/IEC 25010:2011 zusätzlich die Aspekte Risikofreiheit und Kontextabdeckung definiert [28]. Dabei beschreibt die Kontextabdeckung, wie weit das Produkt sowohl in zuvor spezifiziertem als auch in nicht spezifiziertem Kontext eingesetzt werden kann. Außerdem werden in der ISO/IEC 25010:2011 explizit verschiedene Nutzergruppen beschrieben. Hier wird unter anderem zwischen Primärnutzenden und Sekundärnutzenden unterschieden. Letztere sind zum Beispiel für die Wartung des Systems zuständig.

Entscheidend für die Erreichung einer hohen Gebrauchstauglichkeit des in dieser Arbeit angestrebten Konzepts ist die Qualität der modellierten Daten. Zur Bewertung dieser liefert die ISO/IEC 25012:2008 ein Datenqualitätsmodell, dessen Charakteristika in Tabelle 3.1 inklusive einer kurzen Erläuterung aufgeführt sind. Dieses Modell betrachtet die Datenqualität aus zwei unterschiedlichen Perspektiven. Einerseits ist die Qualität eine intrinsische Eigenschaft der Daten. In diesem Fall spricht das Datenmodell von inhärenten Qualitätsmerkmalen [29]. Andererseits leiten sich einige Qualitätsmerkmale aus den Eigenschaften des verwendeten Datenhandhabungssystems ab [29]. Dementsprechend sind die einzelnen Charakteristika in der Tabelle 3.1 den jeweiligen Perspektiven „inhärent“ und „systemabhängig“ zugeordnet. Außerdem macht die Norm verschiedene beispielhafte Vorschläge zur Messung der Datenqualität. So lässt sich beispielsweise die Exaktheit anhand des Verhältnisses korrekter Aufzeichnungen zu allen Aufzeichnungen beschreiben.

Ganz wesentlich wird die Datenqualität durch das für die Repräsentation gewählte Metamodell beeinflusst. Dazu ist in Tabelle 3.2 aufgeführt, welche Qualitätscharakteristika durch das Metamodell beeinflusst werden. Außerdem wird erläutert, warum das jeweilige Charakteristikum durch das Metamodell beeinflusst wird.

Das Konzept der Gebrauchstauglichkeit, das Software-Qualitätsmodell und das Datenqualitätsmodell haben gemein, dass die Qualität bzw. die Gebrauchstauglichkeit nur aus der Perspektive der Anwendung bewertet werden kann.

3.3 Modelle und Modellbildung

Ein Modell ist gemäß der DIN SPEC 40912 ein „Gegenstand, der es erlaubt, Aussagen über einen anderen modellierten Gegenstand zu treffen“ [30, S. 8]. Darüber hinaus geht die viel zitierte (z. B. in [31], [32] oder im Wikipediaartikel Modell²) Definition des Modellbegriffs nach Stachowiak. Stachowiaks Definition erfolgt anhand des Abbildungs-, Verkürzungs- und Pragmatischen Merkmals. Auch die Definitionen nach [33] oder der Object Management Group (OMG)³ beschreiben diese drei Merkmale. Diese werden nachfolgend kurz erläutert.

Das Abbildungsmerkmal beschreibt, dass „Modelle [...] stets Modelle *von* etwas, nämlich Abbildungen, Repräsentationen natürlicher oder künstlicher Originale, die selbst wieder Modelle sein können“, sind [34].

Das Verkürzungsmerkmal hebt hervor, dass „Modelle [...] im allgemeinen *nicht* alle Attribute des durch sie repräsentierten Originals, sondern nur solche, die den jeweiligen Modellerschaffern und/oder Modellnutzern relevant scheinen“, erfassen [34].

Außerdem werden Modelle immer zielgerichtet bzw. für eine konkrete Aufgabenstellung erstellt. Diese Eigenschaft wird von Stachowiak als Pragmatisches Merkmal beschrieben: „Modelle sind ihren Originalen nicht per se eindeutig zugeordnet. Sie erfüllen ihre Ersetzungsfunktion a) für *bestimmte* - erkennende und/oder handelnde, modellbenutzende - *Subjekte*, b) innerhalb *bestimmter Zeitintervalle* und c) unter Einschränkung *auf bestimmte gedankliche oder tatsächliche Operationen*“ [34].

Eine große Herausforderung im Umfeld der Entwicklung und Normierung von technischen Systemen ist „die Verwendung einer Vielzahl von abstrakten Begriffen, technologischen Konzepten und grundlegenden Modellierungskonzepten“, da dadurch die Zugänglichkeit eingeschränkt wird bzw. der Umfang beispielsweise von Normen stark zunimmt [30, S. 6]. Eine Lösung zur Beherrschung dieser Herausforderung ist die Verwendung standardisierter Begriffe aus anerkannten Wörterbüchern (z. B. aus dem Internationales Elektronisches Wörterbuch⁴) oder die Verwendung einheitlicher Kernmodelle als Grundlage zur Modellierung [30]. Dabei sind Kernmodelle „einfache modellmäßige Beschreibungen von grundlegenden Konzepten und Zusammenhängen, die einen allgemeinen Aspekt von Systemen betreffen“ [30, S. 8]. Eine Vielzahl unterschiedlicher Kernmodelle werden in der DIN SPEC 40912 [30] definiert.

²<https://de.wikipedia.org/wiki/Modell>

³https://www.omg.org/gettingstarted/terms_and_acronyms.htm

⁴<http://www.electropedia.org/>

Tabelle 3.1: Charakteristika zur Bewertung der Datenqualität aus [29, S. 5]

Charakteristika	Datenqualität		Erläuterung
	inhärent	system- abhängig	
Exaktheit	✓		Ist der wahre Wert des beabsichtigten Attributs in Bezug auf den Nutzungskontext in den Daten enthalten?
Vollständigkeit	✓		Sind Werte für alle im Nutzungskontext relevanten Attribute und Entitäten in den Daten enthalten?
Konsistenz	✓		Sind die Daten widerspruchsfrei und zusammenhängend?
Glaubwürdigkeit	✓		Werden die Daten im Nutzungskontext als wahr angesehen?
Aktualität	✓		Haben die Daten das richtige Alter?
Zugänglichkeit	✓	✓	Kann auf die Daten in einem bestimmten Nutzungskontext zugegriffen werden?
Compliance	✓	✓	Entsprechen die Attribute der Daten den relevanten Regularien, Standards und Konventionen?
Vertraulichkeit	✓	✓	Können nur berechtigte Personen auf die Daten zugreifen und diese interpretieren?
Effizienz	✓	✓	Wie hoch ist der Aufwand, um im Nutzungskontext auf die Daten zuzugreifen und diese zu verarbeiten?
Präzision	✓	✓	Sind die Daten für den Nutzungskontext genau genug?
Nachverfolgbarkeit	✓	✓	Können Änderungen und Zugriffe auf die Daten nachvollzogen werden?
Verständlichkeit	✓	✓	Können die Daten im Nutzungskontext gelesen und interpretiert werden?
Verfügbarkeit		✓	Kann im Nutzungskontext auf die Daten zugegriffen werden?
Portabilität		✓	Können die Daten von einem auf ein anderes System übertragen werden, ohne dass die Qualität negativ beeinflusst wird?
Wiederherstellbarkeit		✓	Kann auch im Fehlerfall ein gewisses Betriebslevel aufrechterhalten werden?

Tabelle 3.2: Charakteristika der Datenqualität, die von dem Metamodell beeinflusst werden

Charakteristika	Meta- modell	Erläuterung
Exaktheit		Diese ergibt sich aus den Datenquellen.
Vollständigkeit	✓	Das Metamodell muss alle für den Nutzungskontext notwendigen Begriffe und Beziehungen abdecken, andernfalls kann eine Vollständigkeit der Daten nicht sichergestellt werden.
Konsistenz	✓	Die Konsistenz der Daten wird durch Einhaltung eines Metamodells sichergestellt.
Glaubwürdigkeit		Diese ergibt sich aus den Datenquellen.
Aktualität		Ergibt sich einerseits aus der Aktualität der verfügbaren Datenquellen und ist andererseits stark von dem verwendeten Update-Mechanismus des Systems abhängig.
Zugänglichkeit	✓	Das Metamodell und das System definieren, wie auf bestimmte Daten zugegriffen werden kann und haben damit wesentlichen Einfluss auf die Zugänglichkeit.
Compliance	✓	Das Metamodell stellt einen Standard dar, der von den Daten eingehalten werden muss. Gleichzeitig kann das Metamodell auf weiteren Standards aufbauen. Je nach Umsetzung kann das System die Einhaltung des Metamodells mehr oder weniger stark erzwingen.
Vertraulichkeit	(✓)	Das Metamodell ermöglicht eine Klassifikation der Daten und erlaubt so dem System, den Zugriff einzuschränken.
Effizienz	✓	Das Metamodell legt die Begriffe und Beziehungen der Daten fest, damit hat es einen wesentlichen Einfluss auf die Effizienz des Datenzugriffs.
Präzision		Diese ergibt sich aus den Datenquellen.
Nachverfolgbarkeit	✓	Das Metamodell kann entsprechende Konzepte zur Erhöhung der Nachverfolgbarkeit beinhalten.
Verständlichkeit	✓	Durch das Metamodell kann die Semantik der Daten klar beschrieben werden.
Verfügbarkeit		Es ist Aufgabe des Systems, bei Bedarf entsprechende Redundanzen etc. vorzuhalten.
Portabilität	✓	Portabilität ist eine inhärente Eigenschaft von Daten, die mithilfe eines Metamodells verwaltet werden, denn durch ein Metamodell werden Modelltransformationen erleichtert. Dadurch ist die Übertragbarkeit zwischen Systemen sichergestellt.
Wiederherstellbarkeit		Es ist Aufgabe des Systems, angemessene Sicherungsmechanismen bereitzustellen.

Kernmodelle weisen entsprechend der Norm folgende Eigenschaften auf:

- Ein Kernmodell erfasst den Kern vieler unterschiedlicher Systemmodelle.
- Es beschreibt selbst kein konkretes System.
- Es ist grundsätzlich alternativlos und kann nur genau in einer Art und Weise umgesetzt werden.
- Gilt ein Kernmodell für ein konkretes System, ist es vollständig für dieses System gültig.

Diese Kernmodelle stellen die Grundlage zur Definition von Metamodellen dar und bilden damit das Fundament einer Modellhierarchie. Durch die Verwendung von Metamodellen wird der Datenaustausch zwischen verschiedenen Anwendungen sowie die Applikationsintegration stark vereinfacht [35].

Gemäß der DIN SPEC 40912 ist ein Metamodell die „Definition des Aufbaus und der Bedeutung von Modellen“ [30, S. 8]. Ein Metamodell ist also ein Modell für Modelle und beschreibt demnach Begriffe, also Elemente und deren Beziehungen, die für die Konstruktion eines Modells verwendet werden können. Außerdem kann ein Metamodell Regeln zur Nutzung dieser Begriffe festlegen. Metamodelle stellen einen vielversprechenden Ansatz zur Vereinheitlichung von Modellwelten dar, weil sie ohne Einschränkungen für konkrete Anwendungen standardisiert werden können [36]. Auf diesem Wege können sie die Modell-zu-Modell-Kommunikation stark vereinfachen [36].

Durch die Verwendung einer Modellhierarchie können konzeptuelle und begriffliche Uneindeutigkeiten ausgeräumt, ein besseres Verständnis für alle beteiligten Personen erreicht und eine eindeutigere Spezifikation erstellt werden [17]. Auch kann so die Wiederverwendbarkeit und die Zuverlässigkeit einer entwickelten Lösung erhöht werden [17].

Eine Herausforderung bei der Strukturierung von Daten und Informationen ist der Umstand, dass die unterschiedlichen Modellbegriffe wie semantisches Modell, Informationsmodell, Ontologie oder Datenmodell je nach Kontext unterschiedlich verwendet werden [15]. Um Ordnung in diese Begriffe zu bringen, schlägt [15] basierend auf [37, 38] das in Abbildung 3.3 dargestellte System vor. Die Abbildung wurde um den Begriff des Kernmodells erweitert. Außerdem wurde nach [36] die reale Welt ergänzt. Nachfolgend werden die Begriffe kurz erläutert:

- **Informationsmodelle** beschreiben Begriffe und Beziehungen auf einer konzeptionellen Ebene [37, 38]. Sie sind unabhängig von einer konkreten Implementierung [37, 38]. Sie dienen dem Systemdesigner zur Beschreibung des Systems, dem Nutzer zum Verstehen der Begriffe und dem Entwickler als Anforderungen [37]. Informationsmodelle können sowohl natürlichsprachlich als auch formell (z. B. mithilfe von UML) beschrieben werden [37]. Aus den Informationsmodellen werden die Datenmodelle abgeleitet [37, 38]. Damit sind die Informationsmodelle Metamodelle der Datenmodelle. Nach [15] handelt es sich bei einem Informationsmodell um eine Ontologie.
- **Semantische Modelle** ergänzen sowohl Informations- als auch Datenmodelle um eine explizite Beschreibung der Begriffe und Beziehungen [38]. Auch bei den semantischen Modellen handelt es sich um Ontologien [15].

- **Kernmodelle** stellen Templates für die Entwicklung von Informationsmodellen dar. Wie oben erläutert, beschreiben Sie grundlegende Konzepte und Zusammenhänge, die im Informationsmodell auf einen konkreten Anwendungsfall abgebildet werden.
- **Datenmodelle** werden im Rahmen einer konkreten Implementierung aus den Informationsmodellen abgeleitet und enthalten aus diesem Grund technologie- und implementierungsspezifische Informationen (z. B. Anleitungen zur Abbildung auf eine bestimmte Syntax) [37]. Beispiele für Datenmodelltechnologien sind XML Schema Definition (XSD) oder Resource Description Framework (RDF).
- Eine **Syntax**⁵ ist ein Regelsystem, mithilfe dessen elementare Zeichen zu zusammengesetzten Zeichen (z. B. Wörter mithilfe einer Grammatik zu Sätzen) kombiniert werden können. Extensible Markup Language (XML) oder JavaScript Object Notation (JSON) sind Beispiele für Syntaxen zur Beschreibung von modellierten Zusammenhängen [15].
- **Objektmodelle** bzw. Instanzmodelle beinhalten konkrete Instanzen, die bestimmte Eigenschaften konkreter Elemente der realen Welt repräsentieren. Auf diese kann je nach Implementierung über bestimmte Schnittstellen zugegriffen werden [37, 38]. Beispiele für diese Schnittstellen sind die Simple API for XML (SAX) oder das Document Object Model (DOM) [15].
- Eine **Ontologie** ist ein gemeinsames Verständnis eines Ausschnittes der Welt (Domäne), das mithilfe von Begriffen (Entitäten, Attributen und Prozessen) sowie ihren Beziehungen und Definitionen beschrieben wird [17].

3.4 Wissensrepräsentation

Damit ein Computersystem in der Lage ist, sich intelligent zu verhalten, werden zwei wesentliche Komponenten benötigt. Zum einen bedarf es einer expliziten Wissensrepräsentation und zum anderen wird eine Logik benötigt, mit deren Hilfe das explizit verfügbare Wissen auf neue Erkenntnisse bzw. implizites Wissen untersucht werden kann [39].

Die erkenntnistheoretische Debatte ist an einer klaren Definition⁶ für Wissen gescheitert. Deswegen wird im Rahmen dieser Arbeit zunächst die weniger allgemeine und eher technische Definition für Wissen aus dem Bereich der wissensorientierten Unternehmensführung verwendet [40]. Diese ist in Abbildung 3.4 dargestellt. Demnach handelt es sich bei Wissen um vernetzte Informationen. Informationen wiederum sind Daten⁷ mit Bedeutung, während Wissen die Grundlage zum Handeln ist. Diese Definitionen stimmen mit den Festlegungen aus der VDI Richtlinie 5610 überein [41]. Entsprechend der Richtlinie sind Daten „objektive Fakten“ [41, S. 4], welche ohne Zusammenhänge und weitere Daten nicht

⁵<https://de.wikipedia.org/wiki/Syntax>

⁶https://de.wikipedia.org/wiki/Wissen#Ist_Wissen_definierbar?

⁷Der Begriff „Daten“ wird oft als Synonym für den Begriff „Information“ verwendet. So wird beispielsweise oft von Prozessdaten gesprochen, obwohl die Bedeutung der Daten (z. B. Einheit, Messbereich und Zeitstempel) bekannt ist.

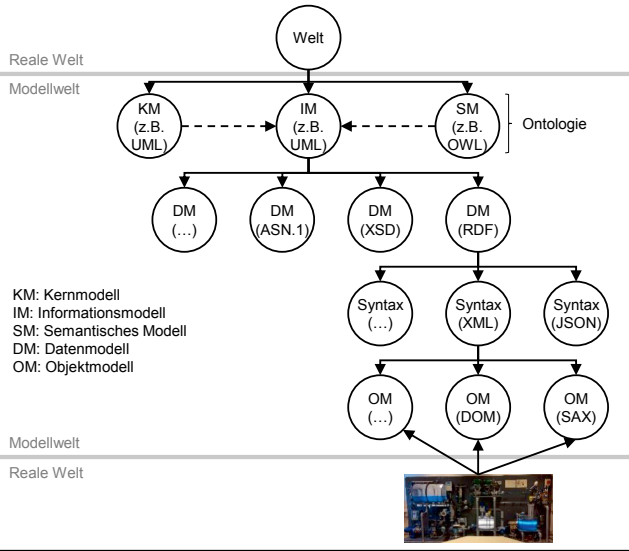


Abbildung 3.3: Beziehungen zwischen verschiedenen Modellarten nach [15, 37, 38], erweitert um den Begriff des Kernmodells und die reale Welt

interpretiert werden können. Informationen werden von der Richtlinie als strukturierte Daten beschrieben, die Relevanz und einen Zweck besitzen sowie kontextualisiert wurden [41]. Gemäß des Standards ist Wissen vernetzte Information. Dank der Vernetzung können beispielsweise Schlüsse gezogen oder Vergleiche angestellt werden [41].

Kern von Wissen sind immer verschiedene Denkeinheiten oder Beschreibungsgegenstände. Diese Denkeinheiten werden in der Literatur als Begriff⁸ [42] oder als Konzept⁹ [43] bezeichnet. Im Rahmen dieser Arbeit wird zur Benennung dieser Denkeinheiten der Begriff „Begriff“ verwendet. Entsprechend der Definitionen werden Begriffe durch Eigenschaften beschrieben. Außerdem können Begriffe in Beziehung zueinander stehen. So kann beispielsweise ein Fahrzeug die Eigenschaft „grün zu sein“ haben und in der Beziehung „gehört zu“ zur besitzenden Person stehen. Dementsprechend setzt sich Wissen also aus Begriffen, Eigenschaften und Beziehungen zusammen. Dabei gilt, dass Eigenschaften und Beziehungen selbst Begriffe darstellen.

Insgesamt unterscheidet man zwischen Allgemein- und Individualbegriffen [42]. Mithilfe dieses Konzepts können Begriffshierarchien konstruiert werden [43]. Dazu repräsentiert ein

⁸ „Die gedankliche Zusammenfassung derjenigen Merkmale, die bei Gegenständen als gemeinsame Eigenschaften erkannt werden, führt zu Denkeinheiten, die man als „Allgemeinbegriffe“ bezeichnet. Ein Allgemeinbegriff muss nicht an bestimmte Sprachen gebunden sein; er ist jedoch von dem jeweiligen gesellschaftlichen und kulturellen Hintergrund beeinflusst“ [42, S. 7].

⁹ „Ein Konzept ist ein 3-Tupel (*Konzeptname*, *Extension*, *Intension*). Die *Extension* ist die Menge aller Objekte, die zu dem Konzept gehört, während die *Intension* die Merkmale angibt, die ein Objekt aufweisen muß, um zu dem Konzept zu gehören. Diese Merkmale nennen wir auch Konzeptmerkmale“ [43, S. 17]

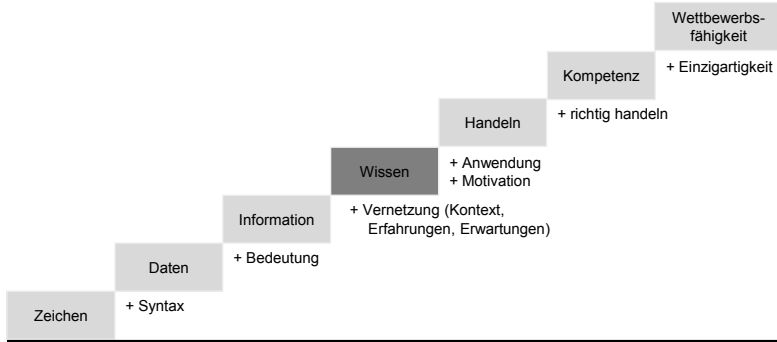
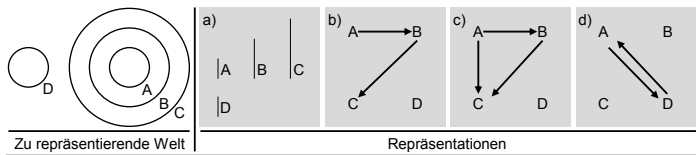


Abbildung 3.4: Einordnung des Begriffs „Wissen“ in die Wissenstreppe zur wissensorientierten Unternehmensführung [40]

Allgemeinbegriff (z. B. „technische Hochschule“) eine Klasse von Denkeinheiten und ein Individualbegriff (z. B. „die RWTH Aachen“) genau eine Denkeinheit [42]. Dabei steht ein Individualbegriff in einer „ist Instanz von“-Beziehung zu einem Allgemeinbegriff (d. h. „die RWTH Aachen“ ist eine Instanz von „technische Hochschule“) [43]. Darüber hinaus können verschiedene Allgemeinbegriffe in einer Vererbungsbeziehung zueinander stehen [43]. Dabei werden alle Eigenschaften eines allgemeineren Begriffs auf einen spezielleren Begriff übertragen [43]. In der Literatur wird dies als „is a“-Beziehung bezeichnet (d. h. „technische Hochschule“ ist eine besondere Hochschule) [43]. In der DIN 2330 wird eine solche Begriffshierarchie als Begriffssystem bezeichnet [42]. Begriffssysteme dienen zur Ordnung, Vereinheitlichung und Standardisierung von Wissen [42]. Sie stellen damit die Grundlage einer Wissensrepräsentation dar.

Der Begriff Wissensrepräsentation lässt sich als das „Aufschreiben von Symbolen - *Repräsentationsstrukturen* genannt - bezeichnen, die in einer erkennbaren Weise einem Ausschnitt einer zu repräsentierenden Welt entsprechen (Brachman/Levesque 85)“ [43, S. 9]. Dabei gilt, dass die Wissensrepräsentation aus der repräsentierten Welt ausgewählte Merkmale enthält [43]. Das heißt, dass in der Regel nicht alle Merkmale der repräsentierten Welt dargestellt werden. Darüber hinaus können aber auch Merkmale repräsentiert werden, die nicht Teil der repräsentierten Welt sind [43]. Durch diese Abweichungen zwischen beiden Welten ist eine eindeutige Interpretation der Wissensrepräsentation ohne zugehörige Interpretationsvorschrift nicht möglich. Deshalb ist die Interpretationsvorschrift ein wesentlicher Bestandteil der Wissensrepräsentation [43]. Dies lässt sich mithilfe des in Abbildung 3.5 gezeigten Beispiels verdeutlichen. Die Abbildung zeigt eine zu repräsentierende Welt und vier unterschiedliche Repräsentationen dazu. Zusätzlich sind die zugehörigen Interpretationsvorschriften in der Tabelle angegeben. Deutlich zu erkennen ist, dass nicht alle Merkmale repräsentiert werden. Zusätzlich enthält die Repräsentation neue Merkmale (z. B. Länge der Pfeile).

Damit eine Wissensrepräsentation praktisch einsetzbar ist, muss sie vier Bedingungen bzw. Kriterien erfüllen, die nachfolgend näher beschrieben werden [44]:



Interpretationsvorschriften			
a)	b)	c)	d)
Strichlänge entspricht Objektgröße, Objekte von links nach rechts befinden sich ineinander	$x \rightarrow y$: x befindet sich in y	$x \rightarrow y$: x befindet sich in y, keine Berücksichtigung der Kantenfolgen	$x \rightarrow y$: x hat die gleiche Fläche wie y

Abbildung 3.5: Beispiel für eine Wissensrepräsentation und die dazugehörigen Interpretationsvorschriften aus [43]

1. Zunächst muss die Wissensrepräsentation eine **hinreichende Ausdrucksstärke** besitzen. Dabei hängt diese im Wesentlichen von der für die Wissensrepräsentation verwendeten Sprache ab [44]. Die Ausdrucksstärke ist dann hinreichend, wenn alle für den Anwendungsfall benötigten Informationen beschrieben werden können.
2. Bei der Repräsentation von Wissen muss das **Uniformitätskriterium** beachtet werden [44]. Dies fordert, dass gleiche oder analoge Informationen auf die gleiche Art und Weise dargestellt (repräsentiert) werden [44]. Damit soll eine uniforme Lesbarkeit sichergestellt werden, sodass unabhängig von der gegenwärtigen Situation auf das repräsentierte Wissen zurückgegriffen werden kann [44]. Damit kann ein System (in gewissen Grenzen) befähigt werden, dynamisch auf neue Situationen zu reagieren [44].
3. Wissen besteht aus verschiedenen Wissensinhalten und deren Beziehungen zueinander. Dabei stellen die Beziehungen einen wesentlichen Wissensinhalt dar und können einen Problemlöseprozess wesentlich unterstützen [44]. Deshalb ist die **Erhaltung von Strukturen** ein weiteres Kriterium für die Angemessenheit einer Wissensrepräsentation [44]. Dabei sind insbesondere die horizontalen Beziehungen, also die Beziehungen zwischen unterschiedlichen Elementen, und die vertikalen Beziehungen, sprich Typhierarchien oder Taxonomien, von Interesse [44].
4. Zuletzt gilt es bei der Wissensrepräsentation das **Effizienzkriterium** zu beachten [44]. Dieses Kriterium fordert, dass die Wissensrepräsentation in solch einer Form erfolgt, dass die geplante Wissensnutzung effizient ermöglicht wird [44]. Damit steht dieses Kriterium teilweise im Widerspruch zu den zuvor genannten Aspekten.

Weitere Anforderungen lassen sich aus [17] und [16] ableiten. In [17] werden Klarheit, Kohärenz (inhaltlicher Zusammenhang) und Erweiterbarkeit als Anforderungen genannt.

Hildebrand et. al. beschreiben Modularität, die Verwendung von bestehenden Konzepten und die Verfügbarkeit einer Prozedur für Erweiterungen als weitere Anforderungen [16].

In der Literatur werden unterschiedlichste Formen der Wissensrepräsentation diskutiert. So beschreibt beispielsweise [43] Logik, Produktionsregeln, analoge Repräsentationen, Frames und semantische Netze als mögliche Formen der Wissensrepräsentation. In [45] werden die Methoden Folksonomie, Schlagwortmethode, Klassifikation, Thesaurus und Ontologien vorgestellt.

Im Rahmen dieser Arbeit wird auf Grund der hohen Eignung für den Anwendungsfall ausschließlich die Methode der semantischen Netze betrachtet. Für alle weiteren Methoden der Wissensrepräsentation sei auf die entsprechende Literatur (z. B. [43, 45]) verwiesen.

3.4.1 Semantische Netze

Mit semantischen Netzen¹⁰ werden netzartige Repräsentationsformate beschrieben, die u. a. in der Informatik zur Wissensrepräsentation eingesetzt werden und eine automatisierte Erkundung und Kombination (engl. Reasoning) des repräsentierten Wissens ermöglichen [43, 46]. In einem solchen semantischen Netz werden Begriffe durch Knoten und Beziehungen durch Kanten dargestellt [43, 46, 47]. In Abhängigkeit von dem gewählten Detaillierungsgrad der Beschreibung der Assoziationsstärke der Kanten wird zwischen assoziativen und semantischen Netzwerken unterschieden [43]. In assoziativen Netzwerken werden Beziehungen mit nur einem Kantentyp beschrieben [43]. Ggf. wird dabei die Assoziationsstärke mithilfe eines numerischen Werts angegeben [43]. Semantische Netzwerke werden dagegen mit unterschiedlichen Kantentypen aufgebaut [43, 47]. Auf diesem Weg wird die Ausdrucksstärke des Netzes erhöht und Inferenzen werden erleichtert [43, 47]. Bei der Konstruktion eines semantischen Netzes können grundsätzlich zwei unterschiedliche Ansätze gewählt werden. Einerseits kann der Open-World-Ansatz und andererseits der Closed-World-Ansatz verfolgt werden [47]. Wird der Closed-World-Ansatz verfolgt, wird angenommen, dass alle verfügbaren Informationen in dem semantischen Netz enthalten sind. Wird dagegen der der Open-World-Ansatz gewählt, dann bedeutet die Tatsache, dass ein bestimmte Aussage nicht enthalten ist, nicht, dass diese nicht wahr sein kann [47].

Wird das in Abbildung 3.3 dargestellte Ordnungskonzept der unterschiedlichen Modellbegriffe betrachtet, kann das Konzept „semantisches Netz“ auf unterschiedlichen Ebenen eingeordnet werden. Auf der Ebene der Ontologien stellt das semantische Netz ein Kernmodell dar. Durch die Definition von wesentlichen Wissensrepräsentationselementen, wie beispielsweise Knoten und Kanten, liefern semantische Netze ein grundlegendes Template für die Entwicklung von Informationsmodellen. In aller Regel stellt konkretes Werkzeug zur Handhabung eines semantischen Netzes (d. h. Graphdatenbank, siehe Abschnitt 3.4.2) eine eigene Syntax bereit, auf deren Basis Datenmodelle und Objektmodelle erstellt werden können. Das eigentliche semantische Netz, also die Repräsentation von konkreten Begriffen mithilfe von Knoten und Kanten, stellt ein Objektmodell dar.

¹⁰https://de.wikipedia.org/wiki/Semantisches_Netz

Tabelle 3.5: Vor- und Nachteile von semantischen Netzen

Vorteile nach [43]	Nachteile nach [43]
<ul style="list-style-type: none">• semantische Nähe wird dargestellt• Struktur geht nicht verloren• Objektzentrierung (alle dem Begriff zugeordneten Eigenschaften befinden sich in unmittelbarer Nähe)• lokale Beschränkung bei der Suche• parallele Suche ist möglich	<ul style="list-style-type: none">• Regelhafte Zusammenhänge, einschränkende Bedingungen, unvollständiges Wissen lassen sich nur schwer abbilden.• geringere Darstellungsmächtigkeit als andere Repräsentationsformen, z. B. Logik• selten formal spezifizierte Semantik, dadurch entsteht Interpretationsspielraum

Damit Wissen mithilfe eines semantischen Netzes modelliert werden kann, müssen folgende Bedingungen erfüllt sein [43]:

1. **Namenseindeutigkeit:** Innerhalb eines semantischen Netzes müssen alle Namen eindeutig sein. Das heißt, dass Knoten mit unterschiedlichen Namen auch unterschiedliche Begriffe darstellen müssen. Im Umkehrschluss bedeutet dies, dass Knotennamen eindeutig sein müssen. [43]
2. **Implizite Konjunktion:** Es gilt, dass alles Wissen innerhalb eines semantischen Netzwerks implizit verknüpft sein muss [43].

Je nach Implementierung und Anwendung wird zusätzlich zu diesen Bedingungen noch eine Vielzahl weiterer Festlegungen getroffen. Diese definieren unter anderem die konkrete graphische Darstellung des Netzwerks. So legt beispielsweise [43] unterschiedliche Darstellungsformen für unterschiedliche Knotentypen (z. B. Begriffe, Eigenschaften, Begriffsklassen, usw.) sowie diverse Kanten typen (z. B. semantische Beziehungen, Vererbung, usw.) fest. Auf Grund der hohen Abhängigkeit von der Umsetzung werden diese Festlegungen in diesem Kapitel nicht weiter untersucht.

In Tabelle 3.5 sind verschiedene Vor- und Nachteile von semantischen Netzen aufgeführt.

3.4.2 Graphdatenbanken

Als Graphdatenbanken werden Datenbanksysteme bezeichnet, die für das Verwalten und effiziente Traversieren¹¹ von Graphen geeignet sind [48]. Dies bedeutet, dass sie Knoten und Kanten abbilden können [49]. Damit eignen sie sich insbesondere für die Speicherung von semantischen Netzen. Graphdatenbanken können grundsätzlich anhand des unterstützten Graphmodells unterschieden werden [13]. Die drei gängigsten Modelle sind Property-Graph, RDF-Triple und Hypergraph [13].

¹¹Allgemein: Etwas überschreiten, durchqueren; konkret: alle Knoten eines Graphen durchlaufen

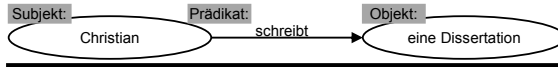


Abbildung 3.6: Beispiel für ein RDF-Triple

Property-Graphen erlauben zusätzlich zur Abbildung von Knoten und Kanten die Annotation dieser mithilfe von Eigenschaften in Form von Schlüssel-Wert-Paaren [48]. Zusätzlich zeichnen sie sich durch folgende Eigenschaften aus:

- Knoten besitzen mindestens einen Bezeichner (Label) [13].
- Kanten haben einen Namen, sind gerichtet und besitzen jeweils genau einen Start- und einen Endknoten [13].

Eine Erweiterung bzw. Generalisierung des Property Graphen stellt der Hypergraph dar [13]. In einem Hypergraph können Kanten beliebig viele Start- und Endknoten besitzen [13]. Diese Kanten werden dann als Hyperkanten bezeichnet [13].

Das RDF¹² ist eine vom World Wide Web Consortium (W3C) standardisierte Methode bzw. eine Sprache zur Formulierung von Fakten. Die Formulierung eines Fakts besteht immer aus den drei Elementen Subjekt, Prädikat und Objekt [13]. Aus diesem Grund werden sie auch als RDF-Triple bezeichnet. Abbildung 3.6 zeigt ein Beispiel eines solchen Triples. Wird eine große Anzahl solcher Fakten formuliert, können sie die Grundlage zur Wissens- oder Beziehungsentdeckung bilden [13]. Aus diesem Grund werden Datenbanken auf Basis dieser Triple als Graphdatenbanken klassifiziert [13]. Allerdings unterstützen sie die native Verwaltung von Graphen nicht [13]. Daraus ergeben sich Performance-Einschränkungen beim Traversieren der Beziehungen [13].

Eine weitere Methode zur Klassifikation von Graphdatenbanksystemen stellt die verwendete Technologie sowohl für die unterliegenden Datenspeicher als auch für den Datenverarbeitungs-Engine dar [13]. Die aktuell populärste Implementierung¹³ einer Graphdatenbank ist Neo4j¹⁴. Neo4j verwendet sowohl für die Datenverarbeitung als auch für die Datenspeicherung graph-native Technologien [13].

Im Rahmen dieser Arbeit wird das Graphdatenbanksystem Grakn verwendet. Kern von Grakn ist die Möglichkeit, domänenspezifisches Wissen mithilfe eines Metamodells auf Basis des Entity-Relationship (ER)-Systemmodells (vgl. Kapitel 4.1.4) modellieren und als Hypergraph verwalten zu können. Durch die Verwendung eines Metamodells, das bei Grakn als Schema bezeichnet wird, unterscheidet sich Grakn von anderen Graphdatenbank-Plattformen wie beispielsweise Neo4j. Neo4j wird als schema-optional bezeichnet¹⁵. Daten, also Knoten, Beziehungen und Eigenschaften, können ohne Schema in einer Neo4j-Datenbank angelegt werden. Deshalb muss die Konsistenz der Daten durch die Klienten sichergestellt werden. Zusätzlich können sogenannte Constraints zur Sicherstellung der Datenintegrität verwendet werden. Im Gegensatz dazu muss für eine Grakn-Datenbank jeder Beziehungs-, Entitäts- und Attributstyp erst mithilfe des Schemas definiert werden, bevor entsprechende Instanzen angelegt werden können. Bei Bedarf kann das Metamodell nachträglich erweitert werden.

¹²https://de.wikipedia.org/wiki/Resource_Description_Framework

¹³<https://db-engines.com/de/ranking> Abgerufen am: 31.03.2020

¹⁴<https://neo4j.com/>

¹⁵<https://neo4j.com/docs/getting-started/current/graphdb-concepts/#graphdb-schema>

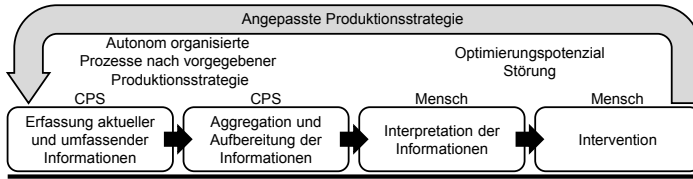


Abbildung 3.7: Aufgabenteilung zwischen Mensch und CPS [50]

Wesentliche Vorteile des schemabasierten Ansatzes von Grakn sind:

- Dank des klar definierten Metamodells wird der Interpretationsspielraum des Graphen eingeschränkt (vgl. Nachteile semantische Netze in Tabelle 3.5). Damit wird einerseits die Integrität der Daten erhöht und andererseits die Zugänglichkeit der Daten sichergestellt.
- Indem das Metamodell explizit beschrieben werden muss, werden klare Verantwortlichkeiten erzwungen. Ohne das Metamodell müsste die Datenstruktur an verschiedenen Stellen der Anwendung definiert werden. Dies würde die Wartbarkeit der Datenbank sowie die Zugänglichkeit und Integrität der Daten negativ beeinflussen.

Damit ein entsprechendes Schema entworfen werden kann, besitzt Grakn ein eigenes Metametamodell. Dies wird in Abschnitt 4.1.4 vorgestellt. Alle Details, die für die prototypische Implementierung relevant sind, werden in Kapitel 7.2.1 beschrieben.

3.5 Definition des Begriffs Kontext für die Prozessindustrie

Mit der Wandlung der industriellen Produktion infolge der vierten industriellen Revolution wird sich auch das Aufgabenfeld des Menschen ändern. Gorecky et al. nehmen auf Grund des *Gesetzes der erforderlichen Varietät*¹⁶ an, dass der Mensch in Zukunft wegen seiner Handlungsvarietät zunehmend einerseits planerisch-schöpferische Tätigkeiten und andererseits intervenierende Aktivitäten übernehmen wird (vgl. Abbildung 3.7) [50]. Dementsprechend soll das CPS bzw. das Automatisierungssystem entsprechend der vorgegebenen Produktionsstrategien den Produktionsprozess möglichst autonom organisieren. Außerdem soll das System aktuelle Informationen umfassend erfassen und dem Menschen aufbereitet sowie aggregiert zur Interpretation vorlegen, damit dieser anschließend Optimierungspotenzial oder Störungen erkennen und beheben kann.

Eine der größten Herausforderungen beim Erfassen, Aufbereiten und Aggregieren der Informationen ist die Bereitstellung des dazugehörigen Kontextes. Dabei ist Kontext¹⁷ als „die Menge an Information definiert, die für die Charakterisierung der Situation von Personen oder Objekten und der Interaktion zwischen Benutzer und einer informationstechnischen Applikation relevant ist [51]“ [50]. Neben dieser Definition werden nach [51] Standort

¹⁶https://de.wikipedia.org/wiki/Ashbysches_Gesetz (Je größer die Handlungsvarietät, desto mehr Störungen kann ein steuerndes System ausgleichen.)

¹⁷[https://de.wikipedia.org/wiki/Kontext_\(IT-Umfeld\)](https://de.wikipedia.org/wiki/Kontext_(IT-Umfeld))

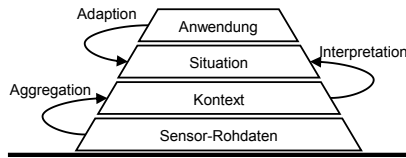


Abbildung 3.8: Zusammenhang zwischen Situation, Kontext, Rohdaten und der Adaption von Anwendungen (Zaslavsky et al. 2010) [52]

(engl. Location), Identität, Aktivität und Zeit als die wesentlichen Informationskategorien des Kontextes beschrieben. Dabei gilt es zu beachten, dass es immer situationsabhängig ist, ob eine bestimmte Information zum Kontext einer Interaktion gehört oder nicht. Eine Übersicht über den Zusammenhang von Rohdaten, Kontext, Situation und einer Anwendung ist in Abbildung 3.8 gegeben. Dabei bilden die Rohdaten das Fundament, auf das aufgebaut werden kann. Die Rohdaten können durch Aggregation zu einem bestimmten Kontext zusammengefasst werden. Wird dieser Kontext interpretiert, ergibt sich eine bestimmte Situation, an die sich eine Anwendung idealerweise anpassen kann.

Aufbauend auf dieser allgemeinen Kontextdefinition kann die Bedeutung der Begriffe „Kontext“ und „Kontextualisierung“ für die Prozessindustrie analog zu [6] festgelegt werden. Bei Betrachtung der genannten Informationskategorien gilt, dass dem Standort die größte Bedeutung zukommt, denn lokale Beziehungen, Abhängigkeiten und Interaktionen spielen beim Betrieb einer Anlage eine ganz wesentliche Rolle. Deshalb ist ihre Aufdeckung zur Verknüpfung von Informationen von wesentlicher Bedeutung. Anders als im Konsumentenbereich (z. B. Smartphone) spielt der geografische Standort (z. B. GPS-Koordinaten) hier eine untergeordnete Rolle. Dagegen ist die relative Position innerhalb der Anlage zur logischen Verknüpfung einzelner Informationen essentiell für die Interpretation von Informationen und Ereignissen. Mögliche logische Verknüpfungen sind beispielsweise, dass A mit B über eine Rohrleitung verbunden ist oder dass C von D unter Zuhilfenahme von E geregelt wird. Dabei ergibt sich die relative Position aus der Anlagenstruktur, die sich beispielsweise aus den Rohrleitungen, Elektroschaltplänen, Signalverbindungen, Bau- und Aufstellungsplänen sowie der Steuerungsstruktur (z. B. Dienstaufrufe) ergibt. In [6] werden diese Verknüpfungen als Umgebungszusammenhänge oder Strukturzusammenhänge bezeichnet. Insbesondere weil sich diese logischen Verknüpfungen über die Zeit ändern können, darf der zeitliche Kontext (zeitlicher Zusammenhang in [6]) bei dieser Betrachtung nicht außer Acht gelassen werden.

Innerhalb einer prozesstechnischen Anlage werden von unterschiedlichsten Akteuren in unterschiedlichen Situationen verschiedenste Aktivitäten durchgeführt. Hier ist einerseits das Personal zu nennen, das beispielsweise Aufträge ausführt, Proben nimmt, Steuerungseingriffe durchführt oder Wartungsvorgänge vornimmt. Andererseits sind in einer Anlage eine große Anzahl technischer Komponenten aktiv, die die unterschiedlichsten Aktivitäten durchführen. Beispiele für diese Komponenten sind Aktoren wie Pumpen und Ventile und die zugehörigen Einzel- oder Gruppensteuereinheiten. Zur Interpretation einer bestimmten Information ist es ganz wesentlich zu wissen, ob diese mit einer Aktivität eines solchen Aktuators in Beziehung steht. Dies ist Teil des in [6] beschriebenen Situationszu-

sammenhangs¹⁸. Neben den genannten Aspekten kann sich der Kontext auch aus einem semantischen Zusammenhang ergeben, also zum Beispiel aus einer Ontologie oder einem Typmodell [6].

Aus der Perspektive der Prozessindustrie besteht der Kontext (vgl. Tabelle 2.1) also aus den Wissensattributklassen zeitlicher Zusammenhang, Struktur- bzw. Umgebungszusammenhang, semantischer Zusammenhang und Ereigniszusammenhang [6]. Der Bereitstellungsvorgang des Kontextes wird als Kontextualisierung bezeichnet. Dabei kann die Kontextualisierung nach [53] in verschiedenen Abstufungen geschehen. Dies ist in Abbildung 3.9 dargestellt. Wie in der Abbildung gezeigt, ergeben sich je nach betriebenem Aufwand und verfügbaren Informationen unterschiedliche Möglichkeiten zur Beschreibung der lokalen Beziehungen. Dabei nimmt die Ausdrucksstärke der Kontextualisierung mit dem betriebenen Aufwand zu. Dies bedeutet: Je höher die Ausdrucksstärke der Kontextualisierung ist, desto feingliedriger und zielgerichteter kann eine Suche in einem konkreten Anwendungsfall erfolgen. Auf diesem Weg kann die Komplexität des Verknüpfungsproblems deutlich eingeschränkt werden. Sollte die Suche auf einer höheren Stufe der Kontextualisierung ohne Ergebnis bleiben, besteht immer die Möglichkeit die Suche auf einer niedrigeren Stufe fortzusetzen, denn jeder modellierte Zusammenhang auf einer ausdrucksstärkeren Stufe ist unspezifischer auch in den darunterliegenden Stufen enthalten. Dementsprechend hat der Wechsel von einer höheren auf eine niedrigere Stufe der Kontextualisierung eine Zunahme des Aufwands bei der Auswertung der Suchergebnisse zufolge.

Die einfachste Variante stellt Ebene I der Abbildung dar. Hier werden alle möglichen Beziehungen bei der Kontextualisierung berücksichtigt. Dieser Kontext besitzt die geringste Ausdrucksstärke. Diese kann erhöht werden, indem für die Kontextualisierung nur die zu einem bestimmten Zeitpunkt relevanten Beziehungen, wie sie sich zum Beispiel aus den Flusswegen ergeben, betrachtet werden. Eine weitere Erhöhung der Ausdrucksstärke lässt sich durch eine Klassifizierung der Beziehungen erreichen. Hier ist beispielsweise eine Unterscheidung zwischen Rohrleitungen, Signalverbindungen oder der Energieversorgung denkbar. Außerdem können die Beziehungen anhand von Produkttypen oder Druckstufen differenziert werden. Die Kontextmodellierung kann durch die Betrachtung der Wirkrichtung, die sich unter anderem aus den Fließrichtungen oder der Steuerungsstruktur ergibt, weiter verfeinert werden. Beispielsweise zur zusätzlichen Berücksichtigung von Totzeiten, die sich auf Grund von Rohrlängen ergeben, können die Beziehungen zusätzlich gewichtet werden. Damit wird die größtmögliche Ausdrucksstärke erreicht. Allerdings werden für eine Kontextualisierung mit dieser Ausdrucksstärke viele zusätzliche Informationen benötigt, die in der Regel nicht standardisiert zu Verfügung stehen. Das optimale Verhältnis zwischen Ausdrucksstärke und dem notwendigen Aufwand zur Modellierung bzw. den dazu benötigten Informationen liegt bei heutigen Systemen typischerweise auf Ebene 4 der Abbildung 3.9.

¹⁸Die Verwendung des Begriffs „Situation“ ist an dieser Stelle widersprüchlich. Gemäß [50] ist der Begriff Kontext als die Information definiert, die zur Charakterisierung einer Situation relevant ist. Damit ergibt sich der Kontext aus der Situation und nicht die Situation aus dem Kontext. Aus diesem Grund wird in dieser Arbeit der Begriff „Ereignis“ verwendet.

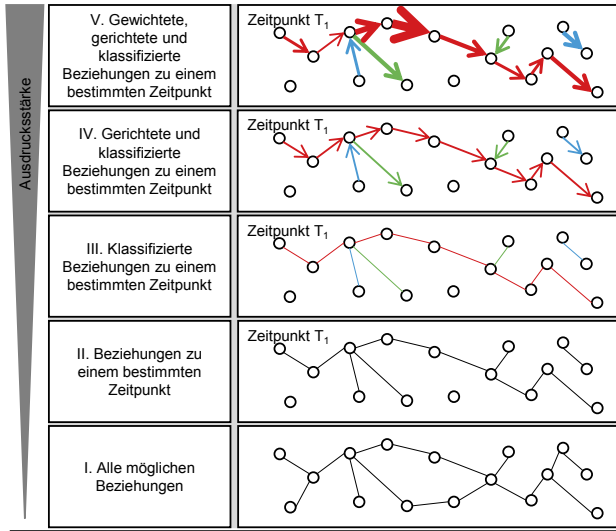


Abbildung 3.9: Verschiedenen Abstufungen der Kontextualisierung [53]

3.6 Ursache-Wirkungsanalyse

Das in Kapitel 2 beschriebene Zielszenario „Aufdeckung von bisher nicht modellierten Zusammenhängen“ beschreibt das Ziel der Verknüpfung von diskreten Ereignissen mit kontinuierlichen Messreihen. Ein Anwendungsfall für diese Verknüpfung kann die Ursache-Wirkungsanalyse sein. Diese ist im Bezug auf die Prozessindustrie ein viel beachtetes Thema in der Literatur (z. B. in [54–59]). In [54–56] wird eine umfassende Übersicht und Klassifikation der unterschiedlichen Ansätze und Methodiken gegeben. Für diese Übersicht werden die Methoden in die drei Klassen „quantitative modellbasiert“, „qualitative modellbasiert“ und „datenbasiert“ eingeteilt.

Modellbasierte Methoden nutzen a priori Wissen (in Form von Modellen) über den Prozess aus, um eine Wirkung mit einer Ursache zu verknüpfen. Dabei kann zwischen quantitativem und qualitativem Wissen unterschieden werden. Quantitative Modelle stellen Beziehungen zwischen Ein- und Ausgangsgrößen in Form von funktionalen mathematischen Modellen dar [54]. Qualitative Modelle stellen Beziehungen zwischen einzelnen Prozesseinheiten qualitativ dar [56].

Im Gegensatz zu den modellbasierten Methoden nutzen die datenbasierten Methoden a posteriori Wissen zur Erkenntnisgewinnung. Dazu müssen große Bestände an historischen Prozessdaten vorliegen [54]. Im Rahmen dieser Arbeit wird der Ansatz der qualitativen modellbasierten Methoden verfolgt. Mögliche Beziehungen zwischen unterschiedlichen Daten sollen auf die einfachste Art und Weise modelliert und so als Grundlage zur Ursache-Wirkungsanalyse verwendet werden.

Venkatasubramanian et al. definieren zehn wünschenswerte Eigenschaften für ein Fehleranalysewerkzeug [54]. Einige dieser Eigenschaften werden im Rahmen dieser Arbeit als

Grundlage für die Anforderungsentwicklung verwendet. Daher werden diese im Folgenden kurz erläutert:

1. **Schnelle Erkennung und Diagnose**

Ein Analysewerkzeug sollte eine möglichst schnelle und einfache Analyse und Diagnose ermöglichen [54]. Zu beachten ist das Spannungsfeld zwischen Geschwindigkeit und Genauigkeit [54]. Je schneller das Analysewerkzeug Fehler erkennt, desto größer ist auch die Sensitivität zu Rauschen und daher auch die Wahrscheinlichkeit von falschen Schlüssen [54].

2. **Isolierbarkeit**

Ein Analysewerkzeug sollte in der Lage sein zwischen verschiedenen Ereignissen zu unterscheiden und entsprechende Ursachen eindeutig zu zuweisen [54].

3. **Anpassbarkeit**

Das Analysewerkzeug muss an Veränderungen beispielsweise der Anlagenstruktur angepasst werden können [54].

4. **Erklärungsmöglichkeit**

Neben dem Erkennen einer abnormalen Situation ist auch das Finden und Erklären der Ursache eine wünschenswerte Eigenschaft des Analysewerkzeugs [54].

5. **Fähigkeit zur Identifikation multipler Ursachen bzw. Fehler**

Die Fähigkeit, unterschiedliche Fehler unabhängig voneinander identifizieren zu können, ist wünschenswert [54].

Weitere wünschenswerte Eigenschaften sind nach [54] Robustheit, Identifizierbarkeit von Neuartigkeiten sowie die Bereitstellung einer Fehlerschätzung. Außerdem müssen die Anforderungen des Werkzeugs zur Ursache-Wirkungsanalyse an die Modelle und an den Speicher und Prozessor so gestaltet sein, dass sie praktisch zu erfüllen sind [54].

3.7 Verwandte Ansätze aus der Literatur

In diesem Abschnitt werden verschiedene Ansätze aus der Literatur vorgestellt, die eine gewisse Nähe zu dem im Rahmen dieser Arbeit entwickelten Konzept besitzen.

In [60] wird ein interessanter Ansatz für die Nutzung von Prozess- und Anlageninformationen für eine automatisierte Diagnose vorgestellt. Dazu werden Informationen sowohl über den Prozess als auch über die Anlagenstruktur in ein einheitliches Modell integriert. Dabei dient die VDI/VDE 3682 als Grundlage für die formale Prozessbeschreibung [60]. Anlageninformationen werden dem R&I-Diagramm entnommen [60]. Als zentrale Herausforderung wird der hohe manuelle Modellierungsaufwand, welcher proportional mit der Anlagengröße zunimmt, genannt [60]. Der Ansatz unterscheidet sich einerseits durch die Berücksichtigung von Prozessinformationen von dem Konzept dieser Arbeit. Andererseits bietet das Konzept dieser Arbeit die Möglichkeit, dynamische Beziehungen zu beschreiben und baut auf einer automatisierten Modellerstellung auf.

In [61] wird ein Konzept zur automatischen Interpretation von Diagnoseergebnissen beschrieben. Dazu wird ein semantisches Modell vorgestellt, das gemeinsame Begriffe verschiedener Diagnosealgorithmen definiert und so die effiziente Nutzung dieser zur gleichen Zeit erlaubt. Zur Ursachenanalyse wird die Struktur des Fertigungssystems ausgewertet [61]. Dank des semantischen Modells wird eine natürlichsprachliche Interaktion mit dem Diagnosesystem ermöglicht [61]. Wird vom System eine bekannte Abnormalität erkannt, werden dem Nutzer bekannte Ursachen vorgestellt. Tritt die Abnormalität zum ersten Mal auf, werden anhand von kausalen Zusammenhängen mögliche Ursachen vorgeschlagen [61].

Petersen et. al. stellen in [62] ein RDF-basiertes Informationsmodell für ein Unternehmen der Fertigungsindustrie vor. Ziel des vorgestellten Ansatzes ist es, die Zugänglichkeit von bestehenden Daten zu erhöhen, eine effiziente Nutzung zu ermöglichen sowie möglichst viel Expertenwissen zu formalisieren [62]. Dabei wurden zwei unterschiedliche Anwendungsfälle betrachtet [62]: Erstes konkretes Ziel ist die Überwachung der Position, Abnutzung und Konfiguration von Werkzeugen. Zweites Ziel ist die Überwachung des Energieverbrauchs zur Laufzeit und pro Werkzeugmaschine. Dazu wird in [62] eine Informationsmodellhierarchie definiert, die auf der oberen Ebene aus einem Kerninformationsmodell besteht, auf das auf der unteren Ebene die Informationsmodelle der verschiedenen Datenquellen abgebildet werden. Mit dem vorgestellten Konzept können Fragen, wie beispielsweise „Wo befindet sich das nächste Werkzeug vom Typ X?“ oder „Wie hoch war der Energieverbrauch während der Produktion von Y?“ beantwortet werden.

In [63] und [64] wird ein Konzepts vorgestellt, das semantische Informationen ausnutzt, um die Zeit zwischen der Datenerfassung und der Datenanalyse in der Fertigungsindustrie reduzieren zu können. Im Zusammenhang mit der Datenanalyse werden zwei Mitarbeiterrollen definiert. Einerseits existiert ein Dateneigentümer, der eine tiefes Verständnis für die Semantik der Daten besitzt. Andererseits gibt es einen Datenanalyst, welcher das Verständnis des Dateneigentümers zur Datenanalyse benötigt. Dementsprechend ist es Aufgabe des Dateneigentümers, die Daten mithilfe des Konzepts so mit Metainformationen anzureichern, dass alle notwendigen Informationen für den Datenanalyst verfügbar sind.

Grundlage des Konzepts sind die beiden Bausteine Ingestion Agents und Evolving Semantic Knowledge Aggregation and Processing Engine (ESKAPE). Die Agenten greifen Daten aus verschiedenen Informationsquellen auf den unteren Ebenen der Automatisierungspyramide ab und reichern diese mit Metainformationen an [63]. ESKAPE sammelt alle Informationen der Agenten und fordert zu Beginn einmalig den Dateneigentümer automatisch dazu auf, sein semantisches Wissen in Bezug auf die Daten zu modellieren¹⁹, damit es gemeinsam mit den Daten abgelegt werden kann [63]. Hierin liegt die Besonderheit des Ansatzes: Die Ontologien zur Wissensrepräsentation müssen nicht im Vorhinein definiert werden, sondern können während des Betriebs schrittweise angelegt, erweitert und verknüpft werden. Für eine Verknüpfung der gespeicherten Daten zu Wissen ist eine konsistente und einheitliche Modellierung unabdingbar, was diese zur wesentlichen Herausforderung macht. Deswegen ist ein Forschungsschwerpunkt von ESKAPE auch die automatische Unterstützung des Modellierungsvorgangs [64]. Abbildung 3.10 zeigt das aus drei Ebenen bestehende ESKAPE Informationsverwaltungskonzept. Es zeigt, wie die verschiedenen semantischen Modelle einerseits zu einem Wissensgraphen verknüpft werden und andererseits auf die Dateninstanzen im Data Lake verweisen.

¹⁹<https://www.youtube.com/watch?v=tmeFWiFZgUs>

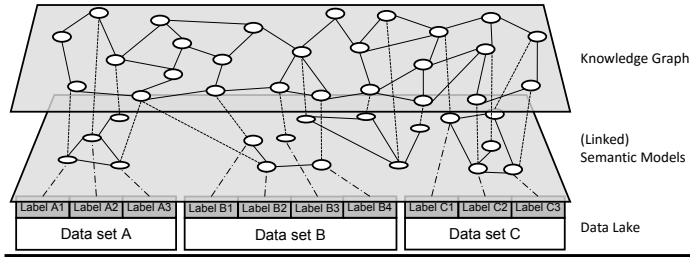


Abbildung 3.10: Überblick über das ESKAPE Informationsverwaltungsconzept aus [63, S. 13]

Damit stellt ESKAPE ein Prozess sowie eine Plattform zur Datenintegration zur Verfügung. Konzepte für die eigentliche Datenmodellierung liefert ESKAPE nicht. Hier liegt die Verantwortung bei den Dateneigentümern, die Daten (unterstützt durch das Tool) so zu modellieren, dass am Ende ein hoch vernetzter Graph entsteht. Das im Rahmen dieser Arbeit entwickelte Beziehungsmodell kann hier einen Beitrag leisten, indem es als gemeinsames Metamodell für alle Datenmodellierungen der Dateneigentümer verwendet wird. Auf diesem Weg kann es zu einer einheitlichen Modellierung und hohen Vernetzung der Daten beitragen.

In [65, 66] wird die Nutzung von Linked Data als Integrationstechnologie für industrielle Daten vorgeschlagen. Ziel des Ansatzes ist es, den agilen und einfachen Austausch von Informationen innerhalb von (virtuellen) Unternehmen zu ermöglichen [66]. Der Ansatz baut auf RDF-Triples auf, die als gerichteter Graph interpretiert werden können [66]. Für das Erzeugen der RDF-Triples wird die Implementierung jeweils eines Adapters für unterschiedliche Datenquellen vorgeschlagen. Als größte Herausforderung wird die formale Beschreibung der Linked Data beschrieben, da ein definiertes und konsistentes Datenmetamodell fehlt [66].

In [67] wird die Intermediate Engineering Ontology als Ebene zwischen bestehenden domänenspezifischen Ontologien und Top-Level-Ontologien mit dem Ziel eingeführt, die Verknüpfung von Informationen von unterschiedlichen Nutzern und aus verschiedenen Domänen zu formalisieren und damit den Nutzen der verfügbaren Informationen zu erhöhen. Für die Definition der Ontologie erfolgt zunächst eine umfassende Analyse der genannten Ontologien mithilfe von sieben Leitfragen. Diese Fragen betreffen den gesamten Produktlebenszyklus, fokussieren allerdings die Design- und Produktionsplanungsphase und greifen nicht die in Abschnitt 2 beschriebenen Zielszenarien dieser Arbeit auf. Auch zeigen die Ergebnisse der Analyse der domänenspezifischen Ontologien, dass mit diesen vornehmlich die Planungs- und Engineeringphase abgebildet werden können. Sie beschreiben Begriffe wie Produkt, Produktionsprozess, Fähigkeit, Spezifikation sowie Qualität.

Eine große Herausforderung beim Bau und Betrieb einer prozesstechnischen Anlage ergibt sich aus dem Zusammenwirken der vielen unterschiedlichen Gewerke. Sei es für die Verfahrensentwicklung, den Anlagenbau oder die Entwicklung der Automatisierungslösung: Jedes Gewerk verwendet sein eigenes Werkzeug mit einem eigenen Datenmodell und eigenen Prozessen. Dabei sind diese Werkzeuge in der Regel nicht auf Kooperation und Informationsaustausch zwischen den Gewerken ausgelegt. Für einen erfolgreichen und effizienten Projektabschluss ist genau dieser Austausch essentiell. Zur Lösung dieses Pro-

blems schlägt [68] eine „engineering-knowledge-base“ als Rahmenwerk vor, in dem werkzeübergreifend Projektinformationen abgelegt werden. Kern dieser Lösung ist ein virtuelles gemeinsames Datenmodell. Daten werden aus den jeweiligen Werkzeugen abgegriffen und in einer zentralen Datenbank abgelegt. Dazu werden die Werkzeugontologien auf eine gemeinsame und allgemeinere Domäneontologie bzw. das virtuelle Datenmodell abgebildet. Aus diesen Abbildungen lassen sich dann Transformationsregeln für den Informationsaustausch zwischen den einzelnen Werkzeugen ableiten. Eine große Herausforderung beim Verknüpfen von Engineering Daten ist die Handhabung von unterschiedlichen Versionsständen [69].

Das Verknüpfen von Engineering Daten ist ein Anwendungsfall von OntoCape²⁰. OntoCape ist eine der domänenspezifischen Ontologien, die in [67] untersucht wurden. Diese Ontologie wurde zur generischen Erfassung von Wissen der Verfahrenstechnik entwickelt [70]. Wichtigste Anwendungsfälle von OntoCape sind unter anderem die mathematische Prozessmodellierung [71] sowie die Integration von Designinformationen (z. B. in [72]) aus verteilten Datenquellen.

In [73] wird OntoCape zu OntoSafe erweitert. Gemäß [73] kann damit das Alarmverhalten eines Prozessleitsystems automatisch an Änderungen in der Anlagenstruktur sowie an Änderungen im Betriebszustand (z. B. Stationär- oder Transientbetrieb) angepasst werden.

In [74] wird ein Produktflusswegemodell vorgestellt. Damit wird die Betrachtung und Verwaltung verschiedener Flusswege als eigenständige Assets ermöglicht. Dadurch können die Flusswege als einzelne Softwareobjekte im Leitsystem verwaltet werden. Dabei ergeben sich die Flusswege aus der Anlagenstruktur. Im Gegensatz zur vorliegenden Arbeit ist das Ziel des Produktflusswegemodells vornehmlich eine Unterstützung bei der Steuerung und Überwachung der Flusswege während der operativen Phase. Beispielsweise kann der Operator dank des Modells auf ungewollte Vermischungen aufmerksam gemacht werden.

²⁰<https://www.avt.rwth-aachen.de/cms/AVT/Forschung/Software/ipts/OntoCape/>

4 Bestehende Modellierungskonzepte und Informationsquellen

Die in Kapitel 3 vorgestellten Konzepte aus der Literatur zeigen, dass ein semantisches Netz bzw. eine Graphdatenbank die ideale Grundlage zur Modellierung und Ablage von verknüpften Informationen ist. Damit wird es möglich, für eine bestimmte Situation den Kontext, also die Menge an Informationen, die zur Charakterisierung einer Situation benötigt wird, bereitzustellen. Die untersuchten verwandten Ansätze aus der Literatur zeigen, dass eine Herausforderung die geschickte Modellierung der Informationsverknüpfung ist. Ohne eine solche können die Qualitätskriterien einer Wissensrepräsentation nicht eingehalten werden. Wie in Tabelle 3.2 beschrieben, hat dies direkten Einfluss auf die Datenqualität und damit auch auf die spätere Gebrauchstauglichkeit eines Werkzeugs, das auf der Wissensrepräsentation aufbaut. Mithilfe der vorgestellten Informationsmodellierungskonzepte und durch die Berücksichtigung der bestehenden Kern- sowie Informationsmodelle kann eine solche Modellierung erreicht werden. Dies steht im Einklang mit der gewählten Vorgehensweise. Diese schreibt vor, dass nach der Anforderungsspezifikation in Form der Kompetenzfragen eine leichtgewichtige Ontologie konstruiert wird. Der erste Schritt dorthin ist die Untersuchung bestehender Informationsquellen und ihrer Datenmodelle. Dies erfolgt in diesem Kapitel in zwei Schritten. Zunächst werden verschiedene grundlegende Konzepte zur Modellierung von Systemen betrachtet. Anschließend wird eine Auswahl der im Umfeld einer prozesstechnischen Anlage bestehenden Informationsquellen und die zugehörigen Informations- bzw. Datenmodelle vorgestellt. Zusätzlich werden verschiedene Ereignistypen betrachtet. Dabei werden ausschließlich die Datenquellen betrachtet, die für die Beantwortung der in Kapitel 2 definierten Kompetenzfragen benötigt werden.

4.1 Systeme

Ziel dieser Arbeit ist die Kontextualisierung von Zeitreihen-Informationen durch Ausnutzung von Strukturbeschreibungen und der Bestimmung eines räumlichen Abstands. Dabei wird der Umstand ausgenutzt, dass sich eine Anlage als System beschreiben lässt. Entsprechend der DIN SPEC 40912 ist ein System eine „Menge miteinander in Beziehung stehender Elemente, die in einem bestimmten Zusammenhang als Ganzes gesehen und als von ihrer Umgebung abgegrenzt betrachtet werden. (DIN IEC 60050-351)“ [30, S. 9]. Grafisch dargestellt wird diese Definition in Abbildung 4.1. Demnach lässt sich ein System entweder aus einer Innen- oder einer Außensicht beschreiben. Die Außensicht beschreibt die Schnittstelle des Systems zur Umgebung. Aus der Innensicht werden die einzelnen Teilsysteme oder Systemelemente und deren Beziehungen sichtbar. Die Systemhülle grenzt beide Sichten voneinander ab. Da ein System immer „in einem bestimmten Zusammenhang als Ganzes gesehen“ [30, S. 9] wird und selbst wieder aus Systemen bestehen kann, handelt es sich bei der Festlegung der einzelnen Systemgrenzen um Designentscheidungen.

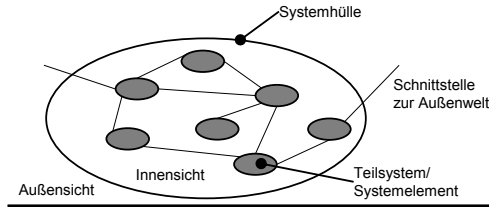


Abbildung 4.1: Definition des Systembegriffs nach [30]

Eine gängige Konvention für die Systemgrenzen und Abstraktionsgrade zur Beschreibung von Anlagen in der Prozessindustrie liefert die ISA 88 in Form des physischen Modells [75, 76]. Auch wenn die ISA 88 eine Norm für Anlagen mit chargenorientierter Fahrweise ist, ist das physische Modell ohne Einschränkungen auch auf kontinuierlich produzierende Anlagen anwendbar (vgl. Anlagenmodell aus [77] mit dem physischen Modell). Das Modell ist in Abbildung 4.2 dargestellt.

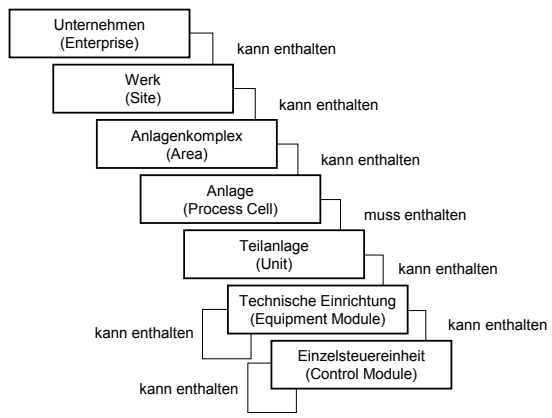


Abbildung 4.2: Physisches Modell aus [75, 76]

4.1.1 Systemmodellierung

Eine gängige Variante zur Modellierung eines Systems ist das ER-Kernmodell, wie es in Abbildung 4.3 dargestellt ist und in [30] beschrieben wird. Gemäß [30] besteht ein System aus Systemelementen und Beziehungen. Diese Beziehungen stellen über Quell- und Zielkanten Verknüpfungen zwischen einzelnen Systemelementen dar. Zusätzlich besitzen sowohl die Beziehungen als auch die Systemelemente Eigenschaften oder Attribute. Als Basisbeziehungstypen sind in der Abbildung Abstraktion, Aggregation und Komposition dargestellt. Mithilfe dieser können hierarchische Beziehungsstrukturen aufgebaut werden.

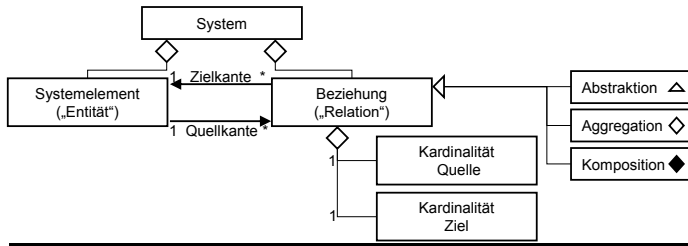


Abbildung 4.3: Grundstruktur des ER-Metamodells nach [30, S. 22]

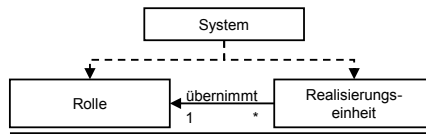


Abbildung 4.4: Rollenmodell aus [30]

4.1.2 Rollenmodell

Ein weiteres Konzept zur Systemmodellierung liefert [30] mit dem Rollenmodell. Dabei stellt die Rolle ein Element dar, das innerhalb einer Systembeschreibung eine Realisierungseinheit repräsentiert [30]. Dazu beschreibt die Rolle Anforderungen, die durch die Realisierungseinheit erfüllt werden müssen, falls diese die Rolle einnehmen soll [30]. Eine Übersicht über die Begriffe des Modells und deren Beziehungen gibt Abbildung 4.4.

4.1.3 Konzeptuelles Datenmodell der ISO 15926

Ein weiteres Metamodell (in der Norm als konzeptuelles Datenmodell bezeichnet) zur Systemmodellierung liefert die ISO 15926-2 [78]. Dieses Modell stellt den Beziehungsaspekt in den Vordergrund und weist aus diesem Grund eine hohe Relevanz für das Konzept dieser Arbeit auf. Ziel der Norm ist es, die Integration von Daten aus unterschiedlichen Quellen über den gesamten Lebenszyklus einer Anlage zu ermöglichen [79, 80]. Dazu werden Anlagen und ihre atomaren Elemente mithilfe eines vierdimensionalen (drei räumliche Dimensionen und Zeit) Ansatzes beschrieben [80]. Das Metamodell baut auf den in Abbildung 4.5 dargestellten Kernbegriffen auf. Gemäß des Metamodells ist jeder modellierte Gegenstand ein Ding (thing). Dabei kann in einem entsprechenden Modell ein Ding entweder ein possible_individual oder ein abstract_object sein. Ein possible_individual repräsentiert einen Gegenstand, der in Raum und Zeit existiert und wird deshalb auch als konkretes Objekt bezeichnet [78]. Im Gegensatz dazu existieren abstract_objects in der realen Welt nicht. Klassen (class) beschreiben die Natur von Dingen [78]. Dinge sind aufgrund bestimmter Kriterien entweder teil einer Klasse oder nicht [78]. Beziehungen (relationship) beschreiben Verknüpfungen von Dingen [78]. In der ISO 15926-2 wird eine Beziehung als geordnetes Paar verstanden [78]. Dinge übernehmen innerhalb von Beziehungen Rollen, die beschreiben in welchem Kontext sie sich in der Beziehung befinden.

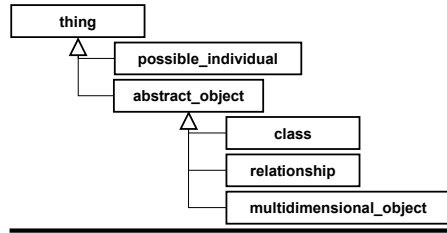


Abbildung 4.5: Kernbegriffe des Metamodells der ISO 15926-2

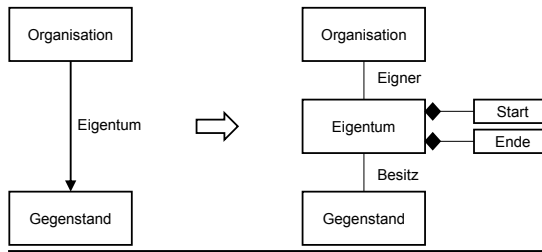


Abbildung 4.6: Gegenüberstellung einer Beziehungsmodellierung mithilfe einer einfachen Beziehung (links) oder einem Beziehungsobjekt (rechts) aus [78]

Die Norm definiert eine Reihe von Standardbeziehungen zur Beschreibung von Zusammenhängen (z. B. „ist ein Teil von“, „ist verbunden mit“ oder „ist eine Lebenszyklusphase von“). Sie bietet aber auch die Möglichkeit eigene Beziehungsklassen (other_relationship) zu definieren [78]. Bei multidimensional_objects handelt es sich um Listen von Dingen [78].

Besonders relevant für diese Arbeit ist der informative Anhang E der ISO 15926-2. Hier wird der mögliche Vorteil durch die Verwendung von expliziten Beziehungsobjekten im Gegensatz zu einfachen Beziehungen diskutiert. Beispielhaft ist dies in Abbildung 4.6 dargestellt. Auf der linken Seite wird die Eigentumsbeziehung mithilfe einer einfachen Beziehung modelliert. Die Darstellung einer Veränderung dieser Verknüpfung ist nicht möglich. Sie kann nur durch Überschreiben der ursprünglichen Eigentumssituation an eine neue Situation angepasst werden [78]. Allerdings würde durch diesen Vorgang die ursprüngliche Information verloren gehen [78]. Aus diesem Grund schlägt die ISO 15926-2 eine Modellierung entsprechend der rechten Seite der Abbildung vor. Sollte sich hier die Eigentumssituation verändern, kann dies mithilfe eines zweiten Beziehungsobjektes modelliert werden.

Der beschriebene Ansatz zeigt, wie mithilfe des konzeptuellen Datenmodells der ISO 15926-2 Lebenszyklusinformationen modelliert werden können. Der Lebenszyklus wird nicht wie sonst oft üblich mithilfe von Attributen wie „geplant“, „gefordert“ oder „erforderlich“ modelliert, sondern mithilfe von Beziehungen zwischen zwei possible_individuals [78]. Unterstützt wird die Modellierung von Lebenszyklusinformationen durch den Umstand, dass possible_individuals auch Ereignisse (event) sein können [78]. Ereignisse sind Dinge ohne zeitliche Ausdehnung, also Zeitpunkte, die zeitliche Grenzen von possible_individuals beschreiben.

4.1.4 Grakn-Metamodell

Das Grakn-Metamodell baut auf dem zuvor beschriebenen ER-Systemmodell auf und erlaubt die Definition von Typhierarchien und die Erstellung von Hypergraphen. Außerdem können nicht modellierte Zusammenhänge mithilfe von Regeln beschrieben werden¹.

Kern des Metamodells sind die Begriffe Entität, Beziehung und Attribut. Zusätzlich zum ER-Systemmodell führt Grakn den Begriff der Rolle ein. Dabei verknüpft eine Beziehung verschiedene Rollen miteinander. Die verschiedenen Rollen werden von den Entitäten eingenommen. Nehmen verschiedene Entitäten Rollen in einer Beziehung ein, knüpfen sie eine entsprechende Beziehung.

Für die Definition von Typhierarchien und zur Verknüpfung von Attributen mit Entitäten und Beziehungen besitzt das Grakn-Metamodell die zwei Standardbeziehungstypen *sub* und *has*. Mithilfe des *sub*-Beziehungstypen werden die Typhierarchien definiert. Befindet sich B in einer *Sub*-Beziehung zu A, dann handelt es sich bei B um eine Spezialisierung von A, die alle Eigenschaften von A erbt. Mithilfe dieses Beziehungstypen können sowohl für Entitäten und Beziehungen als auch für Attribute Typhierarchien definiert werden. Mithilfe des *has*-Beziehungstypen können den Entitäts-, Beziehungs- oder Attributstypen weitere Attributstypen zugewiesen werden.

4.2 Datenquellen

Während des Betriebs einer prozesstechnischen Anlage wird eine Vielzahl an unterschiedlichen Daten erzeugt. Für Ablage, Austausch und Zugriff wurden verschiedene Modellierungskonzepte entwickelt. Diese werden nachfolgend vorgestellt.

4.2.1 Strukturmetamodelle

Wie in Abbildung 4.7 dargestellt, besteht eine prozesstechnische Anlage grundsätzlich aus einem Prozesssystem (1), einem Instrumentierungs- und Prozessführungssystem (3) und einem Energieversorgungssystem (2) [81]. Obwohl in der Norm ISO 15519-1 explizit ein elektrisches Energieversorgungssystem beschrieben wird, werden in prozesstechnischen Anlagen auch andere Arten an Hilfsenergien verwendet (z. B. Druckluft für Ventile). Alle drei zuvor genannten Systeme sind in einer prozesstechnischen Anlage eng miteinander verwoben und stehen in einer starken Abhängigkeit zueinander. Zur Darstellung dieser drei Aspekte definieren Teile 1 und 2 der Norm ISO 15519 unterschiedliche Diagrammtypen. Diese dienen vornehmlich zur Darstellung des Prozesssystems und des Prozessführungssystems auf unterschiedlichen Abstraktionsniveaus (Ebenen des physischen Modells) und in unterschiedlichen Phasen im Lebenszyklus einer Anlage [81, 82]. Dabei fokussieren die Normen vor allem eine standardisierte grafische Darstellung, beispielsweise von Merkmalen oder Objektreferenzen, und weniger das zugrundeliegende Metamodell [83, 84]. Eine vielversprechende Grundlage für das Konzept dieser Arbeit scheint das R&I-Fließschema zu sein, das im nächsten Abschnitt betrachtet wird.

¹<https://dev.grakn.ai/docs/schema>

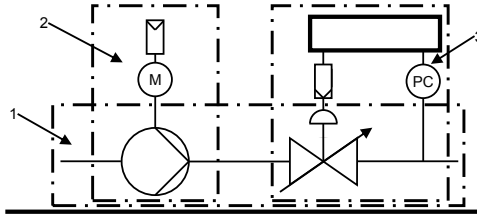


Abbildung 4.7: Beziehungen zwischen Prozesssystem (1), Energieversorgungssystem (2) und Instrumentierungs- und Prozessführungssystem (3) aus [81]

R&I-Fließschema

Das R&I-Fließschema beschreibt das in Abbildung 4.7 gezeigte Prozesssystem und vereint es teilweise mit dem dazugehörigen Instrumentierungs- und Prozessführungssystem. Bezogen auf das physische Anlagenmodell (Abbildung 4.2) wird ein R&I-Fließschema in aller Regel zur Darstellung der Ebenen Anlage oder Teilanlage verwendet. Dazu wird mithilfe des Schemas eine (Teil-)Anlage als ein Netzwerk aus Produkträumen und Produktförderwegen inklusive der dazugehörigen Aktoren (z. B. Pumpen oder Ventile) beschrieben [83, 84]. Zusätzlich beinhaltet es aus prozesstechnischer Perspektive Informationen über die verwendeten Apparate [83, 84]. Darüber hinaus sind in ihm alle Prozessleittechnik (PLT)-Stellen verzeichnet, PLT-Stellen bezeichnen Positionen, an denen Informationen aus dem Prozess abgegriffen (Sensoren) oder in den Prozess eingeleitet (Aktuatoren) werden können [83, 84]. Zusätzlich können PLT-Stellen Elemente des Prozessführungssystems (z. B. Regler) darstellen.

Das R&I-Fließschema gilt als das wesentliche Planungsdokument im verfahrenstechnischen Anlagenbau [83–85]. Es dient zum Informationsaustausch zwischen allen beteiligten Gewerken während der Planung, dem Bau sowie dem Betrieb einer Anlage.

Außerdem wird es im wesentlichen in den Teilen 1 und 2 der Norm DIN EN ISO 10628 definiert. Beide Normteile beschreiben das Fließschema vornehmlich als technische Zeichnung und legen vor allem gestalterische Aspekte, wie zum Beispiel die Blattgröße, die Linienbreite oder grafische Symbole fest. In der Norm wird zwischen Grundinformationen und Zusatzinformationen unterschieden. Grundinformationen müssen, Zusatzinformationen dürfen im Fließschema enthalten sein. [86, 87] Darüber hinaus wird die Bezeichnung der einzelnen PLT-Stellen in der IEC 62424 festgelegt [88].

- Grundinformationen:
 - Für alle Apparate und Maschinen muss das R&I-Fließschema enthalten:
 - * Funktion und Art
 - * Kennzeichnung
 - * kennzeichnende Größen (ggf. in einem separaten Dokument)
 - Für alle Rohrleitungen muss das R&I-Fließschema enthalten:
 - * Nennweite, Druckstufe, Werkstoff, Ausführung
 - * Symbole für Prozessleittechnik inkl. Regel- und Steuerungsfunktionen

Tabelle 4.1: Hauptmerkmale des Metamodells R&I-Fließschema

Merkmal	Beschreibung
Abbildung	Ein Fließschema gibt Teile der Funktionalität sowie den Aufbau einer verfahrenstechnischen Anlage wieder.
Verkürzung	Durch den Umstand, dass es sich beim Fließschema um eine zweidimensionale nicht maßstabsgerechte technische Zeichnung handelt, bei der standardisierte Symbole zur Beschreibung einzelner Komponenten verwendet werden, werden Einbaupositionen verkürzt bzw. nur logisch dargestellt. Das heißt, dass zum Beispiel Rohrleitungslängen oder die genaue Position eines Sensors einem R&I-Fließschema nicht entnommen werden kann.
Pragmatisch	Ein Fließschema wird vornehmlich für den Informationsaustausch zwischen Akteuren, die an der Entwicklung, dem Bau, der Montage oder dem Betrieb einer verfahrenstechnischen Anlage beteiligt sind, verwendet. Damit ist der Informationsaustausch aufgrund der grafischen Natur des Fließschemas vor allem auf menschliche Nutzende (im Unterschied zu Maschinen) ausgelegt.

- Zusatzinformationen:
 - Benennung der Durchflussmengen
 - Fließwege und Richtungen
 - Art wichtiger Mess-, Steuer und Regelungsgeräte
 - relevante Werkstoffe der Maschinen und Apparate
 - relative Höhenunterschiede zwischen Anlagenteilen
 - Kennzeichnung der Armaturen
 - Namen der Anlagenteile

In Tabelle 4.1 werden entsprechend der Definition des Modellbegriffs von [34] die Hauptmerkmale des R&I-Fließschemas aufgeführt. Aufgrund seiner graphischen Natur eignet sich das R&I-Diagramm nur sehr begrenzt für einen automatisierten Informationsaustausch zwischen verschiedenen IT-Werkzeugen (z. B. einem R&I- und einem PLT-Planungswerkzeug) oder für eine automatische Auswertung der beschriebenen Anlagenstruktur und Funktionalität. Die Lücke wollen sowohl PandIX als auch DEXPI schließen. Da die automatische Auswertung von Strukturinformationen eine wesentliche Herausforderung dieser Arbeit ist, werden im Folgenden beide Ansätze beschrieben.

PandIX ist ein auf Computer Aided Engineering eXchange (CAEX) [88] aufbauendes Metamodell zur modellbasierten Beschreibung eines R&I-Fließschemas [80]. CAEX wird im Anhang der IEC 62424 als Metamodell für den automatisierten Austausch von Modellinformationen definiert [88]. Dazu definiert das CAEX-Metamodell aufbauend auf dem Systemelement Interface Connection (SIC)-Systemmodell [30] eine Reihe von Objektklassen und unterstützt Konzepte wie Kapselung, Klassen, Klassenbibliotheken, Instanzen, Instanzhierarchien, Schnittstellen, Attribute und Beziehungen. Auch wenn die IEC 62424

für den Austausch von PLT-Planungsdaten entwickelt worden ist, lässt sie viele unterschiedliche Abbildungsvarianten zu. Damit ist sie für einen automatisiert auswertbaren Informationsaustausch zu unspezifisch [83, 84]. Diese Lücke schließt PandIX durch eine semantische Ausprägung des Metamodells [83]. Diese Ausprägung erfolgt mithilfe einer Modellbeschreibung, einer ProcessPlantElement (PPE)-Musterbibliothek und einem Form- und Positionssystem [83, 84, 89].

Mithilfe von PandIX können prozesstechnische Anlagen als flaches System, das aus miteinander verknüpften Anlagenteilen besteht, modelliert werden [89]. Für die Verknüpfung der Anlagenteile werden deren Konnektoren (Interfaces) explizit beschrieben und mithilfe von Links verknüpft. Diese drei Elemente sind auch im PandIX-Metamodell wiederzufinden, das in Abbildung 4.8 dargestellt ist.

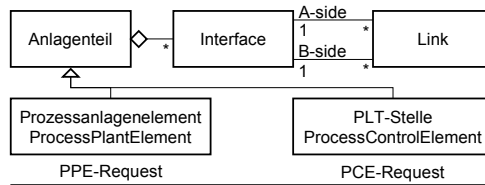


Abbildung 4.8: Klassen des PandIX-Metamodells [83, 84]

Zur Beschreibung von Anlagenhierarchien baut PandIX auf eine Rollen- und eine Systembeschreibung auf, die beide strikt voneinander getrennt sind [89]. Im Gegensatz zur Systembeschreibung ist bei der Rollenbeschreibung die interne Struktur der beschriebenen Anlagenteile nicht bekannt. Die Trennung zwischen Rollen- und Systembeschreibung bietet den Vorteil, dass für die beschriebenen Rollen nachträglich Informationen hinzugefügt werden können, indem ihnen eine ausführende Einheit zugeordnet wird [89].

Wie in Abbildung 4.8 dargestellt, sind Anlagenteile immer entweder vom Typ Prozessanlagenelement oder vom Typ PLT-Stelle. Prozessanlagenelemente sind produktführende Anlagenteile wie beispielsweise Rohrleitungen, Pumpen, Ventile, Behälter usw. [89]. Für typische produktführende Anlagenteile wird von PandIX eine standardisierte Rollenbibliothek (PandIX-PPE-BasicRoleType-Lib) definiert [89]. PLT-Stellen repräsentieren die Schnittstellen zwischen der physikalischen und der Informationswelt [89]. Sie wandeln entweder als Sensoren physikalische Größen in Messwerte oder als Aktuatoren Stellinformationen in physikalische Größen um [89]. Tabelle 4.2 fasst entsprechend der Modellbegriffsdefinition von [34] die Hauptmerkmale von PandIX zusammen.

DEXPI² ist eine ProcessNet Initiative, die von verschiedenen Unternehmen aus der Prozessindustrie (z. B. BASF, Bayer, Covestro und Evonik) gemeinsam mit Herstellern von Computer Aided Engineering (CAE)-Anwendungen getragen wird [90]. Gemeinsam wollen diese Unternehmen auf Basis der ISO 15926 (vgl. Abschnitt 4.1.3) das Kompatibilitätsproblem zwischen den einzelnen CAE-Anwendungen beheben und so einen durchgängigen und intelligenten R&I-Austausch ermöglichen [90]. Dabei baut DEXPI wo möglich auf bestehende Standards und Normen auf (vgl. Tabelle 4.3).

²www.dexpi.org

Tabelle 4.2: Hauptmerkmale des Metamodells PandIX

Merkmal	Beschreibung
Abbildung	PandIX modelliert aus Perspektive der PLT-Planung oder PLT-Betriebssteuerung die funktionale Struktur einer Anlage [83, 84].
Verkürzung	Ein PandIX-Modell beinhaltet ausschließlich für die PLT-Planung relevante Informationen über die Anlage. Dementsprechend sind keine Informationen über die Produkte, Prozesse, Anlagenelemente oder organisatorische Aspekte enthalten [83, 84]. Auch ist weder eine detaillierte Beschreibung der physikalischen oder chemischen Zusammenhänge noch eine genau Darstellung der Automatisierungsfunktionen durch das PandIX-Metamodell vorgesehen [89].
Pragmatisch	PandIX wurde für den einfachen und systematischen Austausch von Anlagenstrukturdaten zwischen Anlagenplanungssystemen und PLT-Systemen entwickelt [83]. Aus diesem Grund besitzt PandIX die zuvor beschriebenen Verkürzungen und berücksichtigt entsprechende Informationen nicht.

Tabelle 4.3: Normen und Standards auf denen DEXPI aufbaut

Anlagen-hierarchie	Apparate u. Maschinen	Rohrleitungs-komponenten	Instrumen-tierung	Kommuni-kation
ISO 10209	ISO 10628	ISO 10628	IEC 62424 IEC 61987 (ANSI 5.1) (DIN 19227)	ISO 15926 Proteus 4.0.1 (vornals XMPlant)

Aktueller Schwerpunkt des DEXPI-Projekts ist der vollständige inhaltliche Austausch von R&I-Diagrammen [90, 91]. Dazu stellt DEXPI mehrere Taxonomien bereit, die die zur Anlagenbeschreibung notwendigen Begriffe definieren und ordnen [90, 91]. Eine solche Taxonomie geht beispielsweise von dem Begriff Rohrleitungskomponente aus und spezifiziert Untertypen wie Absperr- und Sicherheitsarmaturen, welche wiederum zum Beispiel vom Typ Kugelhahn sein können [90, 91]. Dabei werden die Begriffe auf jeder Ebene der Taxonomie mithilfe von Attributen näher beschrieben [90, 91]. Ein solches Attribut kann zum Beispiel für eine Rohrleitungskomponente der Identifizierungscode des durchströmenden Mediums oder für eine Sicherheitsarmatur der maximale Druck sein [90, 91].

Insgesamt definiert DEXPI mithilfe dieser Taxonomien 164³ unterschiedliche Klassen zur Beschreibung von prozesstechnischen Anlagen. Da der aktuelle Schwerpunkt von DEXPI der Austausch von R&I-Informationen ist, ist ein sehr großer Teil des zugehörigen Metamodells der graphischen Darstellung gewidmet [83]. Dadurch wird die zugehörige Modellierung sehr komplex [83]. Trotzdem kann die Struktur einer Anlage einer DEXPI-Modellierung entnommen werden.

Bei der Beschreibung von Anlagenstrukturen unterscheidet DEXPI zwischen drei Aspekten:

1. Beispielsweise Behälter, Pumpen oder Wärmetauscher werden in DEXPI als Equipment modelliert. Equipment besitzt Nozzles als Anschlusspunkte für Sensoren, Aktuatoren oder Rohrleitungen.
2. Physische Verbindungen, z. B. über Rohrleitungen und Ventile, werden in DEXPI mithilfe von PipingNetworkSystems dargestellt. Dabei besteht ein solches System aus mehreren PipingNetworkSegments, die über Connections entweder miteinander oder mit dem Equipment verbunden sind.
3. ProcessInstrumentationFunctions stellen die PLT-Stellen in einem DEXPI-Anlagemodell dar. Diese können sowohl eine ProcessSignalGeneratingFunction als auch eine ActuatingFunction besitzen und in InstrumentationLoopFunctions verknüpft werden.

Weitere Metamodelle für Anlagenstrukturinformationen

Weitere Metamodelle zur Modellierung einer Anlagenstruktur stellen das in Abschnitt 3.1.2 vorgestellte Metamodell SoSX oder das von [92] beschriebene Rohrleitungs- und Instrumentenfließschema verfahrenstechnisch aufbereitet (RIVA) dar. SoSX eignet sich vornehmlich zur Anlagenbeschreibung aus einer übergeordneten Perspektive, beispielsweise zur Modellierung eines Verbundstandortes. Bei RIVA handelt es sich um ein Konzept, dass ein R&I-Modell um verfahrenstechnische Aspekte erweitert.

Prozessführungssysteme

Wie in Abbildung 4.7 dargestellt besteht eine Anlage neben dem Prozesssystem und der Instrumentierung auch noch aus einem Prozessführungssystem, welches sich wiederum sowohl aus verschiedenen Hardwarekomponenten als auch aus einem wachsenden Anteil an Softwarekomponenten zusammensetzt [93]. Das zuvor betrachtete R&I-Fließschema ist in

³http://dexpi-information-model.aixcape.org/queries/classes_result.html Abgerufen am: 07.04.2020

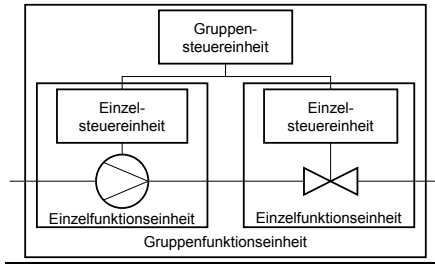


Abbildung 4.9: Aufbau von verschiedenen Einzel- und Gruppenfunktionseinheiten nach [27]

der Lage einen kleinen Teil der Softwarestruktur abzubilden. So werden in der Regel bestimmte (besonders wichtige) Regelkreise oder Steuerfunktionen über PLT-Stellen in dem Fließschema dargestellt. Allerdings beschränkt sich diese Darstellung meistens auf ausgewählte Teile der Automatisierungslösung. Aus diesem Grund kann eine zusätzliche Analyse der Automatisierungssoftware ggf. weitere Abhängigkeiten aufdecken.

In der Prozessindustrie wird die Software zur Automatisierung mithilfe der IEC 61131-3 [94] bzw. IEC 61499 [95] Sprachen erstellt. Dabei bietet sich zum Beispiel die Funktionsbaustein-Sprache zur Analyse der Abhängigkeiten an. Allerdings ist die Struktur der Automatisierungssoftware auf der Ebene der Funktionsbausteine sehr feingliedrig und damit auch sehr umfangreich. Aus diesem Grund empfiehlt sich eine höhere Abstraktionsebene für die Abhängigkeitsanalyse.

Eine höhere Abstraktionsebene kann die einer Automatisierungslösung zugrunde liegende Softwarearchitektur darstellen. Ein in der Automatisierungstechnik gängiger Ansatz dafür ist die hierarchische Prozessführung in Form des in [27] vorgestellten Maßnahmen- und Betriebsmittelmodells. Grundlage dessen ist die Annahme, dass jede Betriebsführungsaufgabe in einzelne Maßnahmen heruntergebrochen werden kann, welche anschließend von unterschiedlichen Betriebsmitteln ausgeführt werden können [27]. Betriebsmittel sind in sich geschlossene Dienstbringer, die selbstständig an sie übertragene Aufgaben lösen können. Dazu besitzen sie eine definierte Führungsschnittstelle. Außerdem können sie zu einem bestimmten Zeitpunkt maximal einen Auftrag übernehmen. Aus der Perspektive der Prozesssteuerung werden Betriebsmittel auch als Funktionseinheiten bezeichnet. Dabei werden, wie in Abbildung 4.9 dargestellt, verschiedene Anlagen- und Steuereinheiten sowie mehrere Steuereinheiten (z. B. eine Pumpe und ein Ventil) zu Funktionseinheiten zusammengefasst, sodass handhabbare Module zur Lösung der Automatisierungsaufgabe entstehen. [27]

Diese Steuereinheiten werden von den Maßnahmen mit der Erfüllung unterschiedlicher Aufgaben beauftragt. Zur Verhinderung von Mehrfachbeauftragungen besitzt jede Funktionseinheit einen Belegungsmechanismus. Das bedeutet, dass wenn eine übergeordnete Maßnahme oder Steuereinheit eine untergeordnete Einheit beauftragen möchte, muss diese zunächst für die Dauer des Auftrags belegt werden [27]. Gerade diese Beziehungen zwischen den einzelnen Steuereinheiten und den Maßnahmen sind sehr interessant für eine Abhängigkeitsanalyse. Insbesondere gilt dies, weil sich diese Beziehungen zur Laufzeit dynamisch verändern können. Abbildung 4.10 zeigt den Lebenszyklus einer Maßnahme von

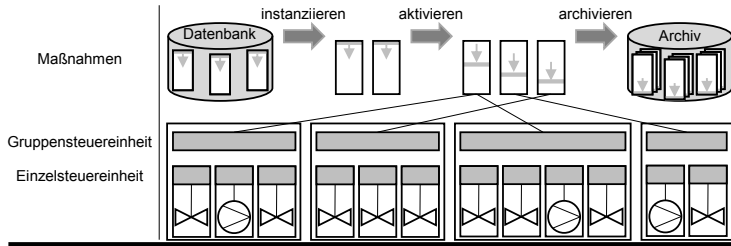


Abbildung 4.10: Lebenszyklus und Beziehungen der Maßnahmen nach [26, 27]

der Instanziierung über die Aktivierung bis hin zur Archivierung. Außerdem sind Beziehungen in Form von Belegungen zwischen den Maßnahmen und den Gruppensteuereinheiten zu einem bestimmten Zeitpunkt dargestellt.

Damit die hierarchische Prozessführung auf Basis des Maßnahmen- und Betriebsmittelmmodells ihr volles Potenzial entfalten kann, müssen die einzelnen Hierarchieebenen eine lose Koppelung miteinander aufweisen. Dies gilt insbesondere für die Ebenen Maßnahmen und Gruppensteuereinheiten. Erreicht werden kann diese lose Kopplung durch den Einsatz einer auftragsorientierten Prozessführung [96]. Die auftragsorientierte Prozessführung basiert auf einem Dienssystem, bei dem eine vollständige Trennung zwischen Übertragungskanal und Kommando vorliegt [96]. Möchte eine Maßnahme einen Auftrag an eine Gruppensteuereinheit vergeben, sendet sie diesen über das Dienssystem in Form eines Kommandos. Da die Steuereinheit, wie zuvor beschrieben, einen Belegungsmechanismus besitzt, muss jeder Auftrag mit einem Belegungswunsch beginnen und mit einer Freigabe enden. Dieser Umstand bietet großes Potenzial zur Erkennung und Analyse der Prozessführungsstruktur. Es müssen ausschließlich die versendeten Kommandos überwacht, die Belegungs- und Freigabebefehle analysiert und Absender, Empfänger und der dazugehörige Zeitpunkt extrahiert werden.

Eine konkrete Umsetzung des Maßnahmen- und Betriebsmittelmmodells wird mit der im BaSys4.0-Projekt⁴ entwickelten BaSys40-Komponenten (vgl. [97]) ermöglicht. BaSys40-Komponenten sind Kommunikationsteilnehmer in einem BaSys40-Dienssystem und besitzen standardisierte Verhaltensweisen [97]. Für die Kommunikation stellen sie eine Dienstschnittstelle zur Verfügung [97]. Dabei kann auf alle Fähigkeiten der Komponente über diese Dienste zugegriffen werden [97]. Außerdem besitzen alle BaSys40-Komponenten der Prozessführungsebene für ihre operativen Fähigkeiten einen Belegungsmechanismus, der sicherstellt, dass Zugriffe ordnungsgemäß ablaufen [97].

4.2.2 Verwaltungsschale

Zwei wesentliche Anforderungen im Umfeld der Industrie 4.0 sind erstens, dass Daten und Definitionen eines Gegenstandes über den gesamten Lebenszyklus des Gegenstandes genutzt, gepflegt und erweitert werden können und zweitens, dass Verknüpfungen zwischen Produkttyp und Produktinstanz ebenfalls über den gesamten Lebenszyklus bestehen sollten [98]. Da die Daten eines Gegenstandes in unterschiedlichen Systemen anfallen sowie

⁴<https://www.basys40.de/>

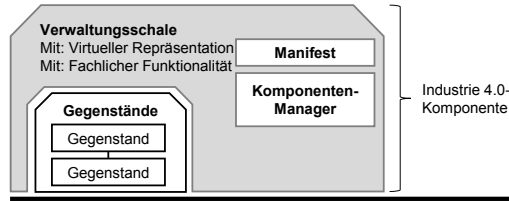


Abbildung 4.11: Industrie 4.0-Komponente bestehend aus einer Kombination aus einer Verwaltungsschale und einem oder mehreren Gegenständen [98, 99]

verwaltet und verarbeitet werden, ist die Erfüllung dieser Anforderungen eine Herausforderung. Zur Lösung dieser Herausforderungen wurde das Konzept der Industrie 4.0-Komponente entwickelt [5, 98, 99]. Eine solche Komponente ist in Abbildung 4.11 dargestellt. Sie besteht aus einer Kombination von zusammengesetzten Gegenständen und einer dazugehörigen Verwaltungsschale [10]. Dabei stellen die Gegenstände sogenannte Assets dar. Assets sind Gegenstände mit einem Wert für mindestens eine Organisation, beispielsweise können Ideen, Software, Services, Archive oder beliebige physische Dinge Assets sein [99]. In der virtuellen Welt werden diese Assets durch eine Verwaltungsschale repräsentiert [99]. Die Verwaltungsschale ist damit die digitale, virtuelle und aktive Repräsentanz des Assets in der Informationswelt [99] und ermöglicht so eine effiziente Identifikation, Adressierung und semantische Referenzierung der dem Asset zugehörigen Daten [10].

Eine Verwaltungsschale besitzt immer einen Header und einen Body [99]. Im Header werden Informationen zur Identifikation, Verwaltung und Verwendung des Assets hinterlegt [99]. Die eigentlichen Informationen über das Asset werden in sogenannten Teilmodellen im Body der Verwaltungsschale archiviert [99]. Diese Teilmodelle können unabhängig voneinander gehandhabt werden und beinhalten Informationen zu unterschiedlichen Aspekten des Assets [99]. Der Zugriff auf die in der Verwaltungsschale hinterlegten Informationen wird mithilfe des Komponenten-Managers organisiert [99]. Das Manifest enthält verpflichtende Angaben zum Asset [99]. Ob ein Asset als eigene Entität in der Informationswelt verwaltet wird, „ist in jedem Einzelfall eine Designentscheidung“ [99, S. 14] und muss in Abhängigkeit des Anwendungsfalls getroffen werden. Eine erste Implementierung der Verwaltungsschale wurde im Rahmen des openAAS-Projekts⁵ geschaffen.

4.2.3 Prozesshistorian

Mit dem Begriff Prozesshistorian⁶ werden Datenbanksysteme bezeichnet, die zur Aufzeichnung und Archivierung von Zeitreihen in der Prozessindustrie verwendet werden. Dabei werden in der Regel die Messergebnisse der verschiedenen Messstellen zur Erfüllung von internationalen Normen und Standards aufgezeichnet [3].

⁵<https://github.com/acplt/openAAS>

⁶https://en.wikipedia.org/wiki/Operational_historian

4.3 Ereignisse

Ein wesentliches Ziel dieser Arbeit ist die Bereitstellung des Kontexts kontinuierlich aufgenommenen Messwerte, also deren Verknüpfung mit diskreten Ereignissen. Hierbei sind insbesondere die Ereignisse von Interesse, die eine Abweichung der Anlage von ihrem normalen Betriebszustand zur Folge haben. Aus diesem Grund werden nachfolgend zunächst die Ursachen für eine solche Abweichung betrachtet. Anschließend werden verschiedene Ereignistypen vorgestellt. Aus diesen kann später abgeleitet werden, welche Aspekte während der leichtgewichtigen Ontologieentwicklung (vgl. Abschnitt 1.2) berücksichtigt werden müssen.

4.3.1 Abnormale Situationen

Eine zentrale Aufgabe beim Überwachen und Steuern eines Prozesses ist der Umgang und die Handhabung von abnormalen Ereignissen bzw. Situationen [54]. Dabei definiert das Abnormal Situation Management (ASM)-Konsortium eine abnormale Situation anhand der folgenden fünf Kriterien [100]:

- Weicht der Anlagenbetrieb von seinem normalen Betriebszustand aufgrund einer Störung bzw. einer Serie von Störungen ab, so liegt eine abnormale Situation vor.
- Die Folgen einer abnormalen Situation können im Bereich von minimal bis katastrophal liegen. Im besten Fall führen sie ausschließlich zu einer Reduktion der Anlageneffizienz. Im schlimmsten Fall besteht Gefahr für Mensch oder Umwelt.
- Das Bedienpersonal hat die Aufgabe, eine solche Situation zu erkennen, die Ursache zu finden und die Wirkung zu beheben.
- Abnormale Situationen können sich in einem dynamischen Prozess über die Zeit entwickeln, verändern oder wachsen. Deshalb werden an einen korrigierenden Eingriff besondere Anforderungen gestellt.

Entsprechend dieser Definition ist, wie in Abbildung 4.12 dargestellt, das Erkennen, Verstehen und Beheben eine wesentliche Aufgabe der Anlagebedienenden. Dabei sind in vielen Betreiberunternehmen strukturelle Veränderungen zu beobachten, die eine Reduktion des Personals im Umfeld der Produktion zur Folge haben [101]. Somit muss immer weniger Personal immer komplexere Produktionsprozesse überblicken und verstehen können. Hinzu kommt auf der einen Seite, dass Entscheidungen zur Behebung der Ereignisse anhand von unvollständigen, unplausiblen, unzuverlässigen oder fehlerhaften Informationen getroffen werden müssen [54]. Auf der anderen Seite bestehen eine Vielzahl an unterschiedlichen möglichen Ursachen für eine abnormale Situation. So kann beispielsweise ein Gerät ausfallen, schleichend seine Funktionalität einbüßen oder ein Parameter wegdriften [54]. In Zukunft ist damit zu rechnen, dass durch die für die Industrie 4.0-Szenarien eingeführte Flexibilisierung und Wandelbarkeit und die damit einhergehenden automatischen und dynamischen Änderungen in der Systemarchitektur weitere mögliche Ursachen für abnormale Situationen entstehen können.

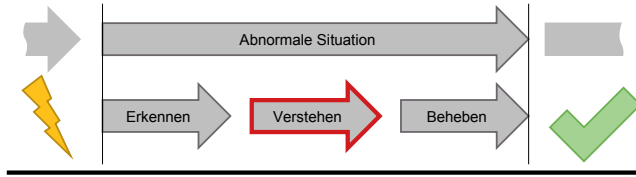


Abbildung 4.12: Schritte zur Behebung einer abnormalen Situation (angelehnt an [102])

Ursachen für abnormale Situationen

Das ASM-Konsortium beschreibt drei unterschiedliche Ursachen für abnormale Situationen. Diese können zum einen durch die technische Ausrüstung einer Anlage und zum anderen durch den Prozess selbst verursacht werden. Außerdem können menschliche sowie betriebsbedingte Faktoren zu abnormalen Situationen führen. Die relative Häufigkeit der unterschiedlichen Ursachen ist in Abbildung 4.13 gegeben. Entsprechend des Konsortiums [103] sind Beispiele für mögliche Ursachen:

- Technische Ausrüstung:
 - schlagartiges Ausfallen eines Geräts (z. B. Kompressor oder Pumpe)
 - schleichender Leistungsverlust (z. B. Fouling von Wärmetauschern)
 - Ausfall eines Sensors oder Aktors
 - Versagen der Steuerung
- Prozess:
 - Betrieb außerhalb der Spezifikationsgrenzen des Prozesses
 - ungünstige Auslegung des Prozesses
- menschliche oder betriebsbedingte Faktoren:
 - mangelnde Erfahrung oder Ausbildung
 - Stress
 - fehlende oder unklare Entscheidungshierarchien
 - unzureichende Kommunikation

4.3.2 Ereignistypen

Die zuvor beschriebenen Ursachen für eine abnormale Situation zeigen, dass eine große Anzahl unterschiedlicher Ereignistypen eine solche Situation auslösen kann. Eine nicht abgeschlossene Übersicht über mögliche Ereignistypen gibt Abbildung 4.14. Beispielsweise kann die Topologie einer Anlage durch ein Umbau-, ein Aktuatorereignis oder auch einen Ausfall beeinflusst werden. Die Funktionalität einzelner Geräte kann durch einen Austausch verändert werden. Lastverteilungen bzw. Redeployment (vgl. [104]), Ressourcenadaption (vgl. [105]) oder das Aktualisieren von Automatisierungslösungen (vgl. [106, 107])

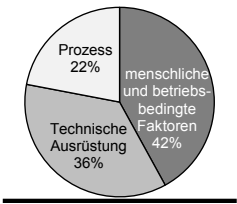


Abbildung 4.13: Häufigkeit der Ursachen für abnormale Situationen nach [103]

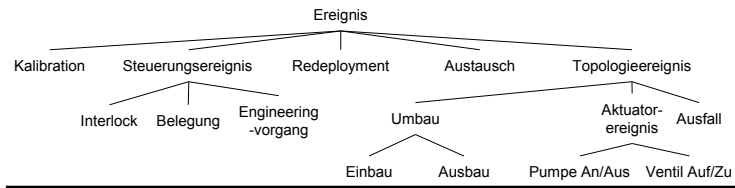


Abbildung 4.14: Hierarchische Darstellung unterschiedlicher Ereignistypen im Umfeld einer prozesstechnischen Anlage

stellen das Leitsystem betreffende Ereignisse dar. Innerhalb einer Automatisierungslösung können zum Beispiel Belegungsvorgänge, das Auslösen einer Verriegelung oder Engineeringvorgänge als Ereignis betrachtet werden. Auch die Kalibration eines Sensors oder die Aktualisierung einer Sensorfirmware stellt ein Ereignis dar.

4.4 Diskussion der Informationsquellen

In Tabelle 2.1 werden die zur Beantwortung der Kompetenzfragen benötigten Informationsquellen benannt. Zum Abgleich erfolgt in Tabelle 4.4 eine Zuordnung der benötigten Informationsklassen zu den betrachteten Informationsquellen bzw. Informationsmodellen. Dabei wird ersichtlich, dass zu jeder Informationsklasse ausreichend Quellen verfügbar sind. Neben den im Rahmen dieses Kapitels betrachteten Quellen existieren im Umfeld einer prozesstechnischen Anlage eine Vielzahl weiterer Systeme zur Informationsverarbeitung. Diese konnten im Rahmen dieser Arbeit nicht alle betrachtet werden. Vielmehr wurde der Fokus auf offene und standardisierte Austauschformate gelegt. Dadurch konnte ein einfacher und effizienter Zugriff auf die benötigten Informationen von vornherein sichergestellt werden.

Tabelle 4.4: Mögliche Informationsquellen zur Beantwortung der Kompetenzfragen

Informationsklasse	Informationsquelle bzw. Informationsmodell
Anlagenstruktur	PandIX, DEXPI, SoSX, Maßnahmen- und Betriebsmittelmodell
Messwerte	Prozesshistorian
Ereignisse	Teilmodelle der Verwaltungsschale
Semantik	ergibt sich aus den Informationsmodellen
Zeitlicher Bezug	Teilmodell der Verwaltungsschale, Prozesshistorian

5 Anforderungsanalyse und -spezifikation

Die Untersuchungen in Kapitel 4 zeigen einerseits, dass im Umfeld der Prozessindustrie eine große Menge unterschiedlicher Daten anfallen. Dabei ist festzuhalten, dass der wesentliche Teil dieser Daten in unterschiedlichsten Systemen anfällt und keinerlei Verknüpfung stattfindet. Andererseits haben die Untersuchungen gezeigt, dass in der Prozessindustrie eine Vielzahl unterschiedlicher Informationsmodelle existiert und immer mehr Verwendung findet. Diese Informationsmodelle erlauben einen einfachen und automatisierten Zugriff auf diese Daten. Damit fehlt im nächsten Schritt ein Konzept, das diese Informationsmodelle ausnutzt und die modellierten Daten über die Grenzen der einzelnen Datenquellen hinweg verknüpft und auf diesem Weg die verfügbaren Informationen einfacher zugänglich und erkundbar macht, also den Kontext darstellen kann. Im Rahmen dieses Kapitels werden die Randbedingungen für ein solches Konzept erläutert. Dazu wird zunächst der domänentypische Rahmen beschrieben. Anschließend werden aus den zuvor beschriebenen Zielszenarien bzw. Kompetenzfragen Anforderungen an das Konzept abgeleitet. Außerdem finden auf diesem Weg die in Kapitel 3 beschriebenen Anforderungen an eine Wissensrepräsentation sowie an eine Ursache-Wirkungsanalyse strukturierten Eingang in die Konzeptentwicklung. Im Sinne der in Kapitel 1.2 vorgestellten Vorgehensweise werden also die Anforderungen an die Ontologie weiter ausspezifiziert.

5.1 Analyse der Randbedingungen aus dem Anwendungsgebiet

Wie jede Anwendungsdomäne ist auch die Prozessindustrie geprägt von ganz eigenen Randbedingungen, die einen erheblichen Einfluss auf die eingesetzten Lösungen haben. Diese Randbedingungen können in zwei unterschiedliche Bereiche unterteilt werden. Einerseits bestehen in der Prozessindustrie klassische Anforderungen, beispielsweise bezüglich der Dauer eines Einsatzes und der Laufzeit einer Komponente. In der Literatur (z. B. in [105, 106]) werden diese Randbedingungen detailliert beschrieben. Zusätzlich ergeben sich aber durch die Vorbereitungen auf eine vierte industrielle Revolution neue Aspekte bzw. Randbedingungen. In den beiden nachfolgenden Abschnitten sollen diese noch einmal kurz zusammengefasst werden und dienen dann als Grundlage für die späteren Überlegungen in Bezug auf die Anforderungen an das Konzept dieser Arbeit.

5.1.1 Klassische Randbedingungen der Prozessindustrie

Lebenszyklus einer prozesstechnischen Anlage

Die Anlagen der Prozessindustrie sind geprägt durch mehrere Jahrzehnte andauernde Lebenszyklen [85, 108]. Innerhalb eines Lebenszyklus wird eine Anlage über lange Phasen ohne Unterbrechung betrieben und nur zu definierten Zeitpunkten für geplante Wartungsphasen abgeschaltet. Daraus resultieren besondere Anforderungen an die Lebensdauer der eingesetzten Komponenten [109]. Über die Gesamtdauer ihres Lebenszyklus unterliegt eine Anlage kontinuierlichen Veränderungen: Es finden u. a. Kapazitätserweiterungen, Modernisierungen der Automatisierungstechnik, Reparaturen oder Änderungen an der Software statt [85]. Dies hat in den meisten Fällen zur Folge, dass eine Anlage mit einem sehr heterogen System bestehend aus unterschiedlichsten Technologien betrieben wird.

Anlagendiversität

Aufgrund der langen Lebenszyklen und der hohen Investitionskosten herrscht eine hohe Anlagendiversität. Das heißt, dass es sich bei den meisten Anlagen um Unikate handelt. Aus diesem Grund sind neue Lösungen für bestehende Anlagen immer Spezialanfertigungen, die „auf Maß“ für die jeweilige Anlage erstellt werden müssen. Damit ist die Übertragbarkeit von neuen Ansätzen von einer zur nächsten Anlage deutlich eingeschränkt und kann bei schlechtem Design mit einem hohen Konfigurationsaufwand verbunden sein.

Konsequenzen einer Störung

Prozesstechnische Anlagen besitzen in der Regel ein relativ hohes Schadenspotenzial. Damit können die Konsequenzen einer Störung des Betriebs katastrophal sein [110]. Die Spannweite der möglichen Konsequenzen reicht dabei von einem Produktivitätsverlust bis hin zu Gefährdung oder Verlust von Menschenleben [110]. Aus diesem Grund besitzt die Sicherheit einen extrem hohen Stellenwert in der Prozessindustrie.

Rolle des Menschen

Im gesamten Lebenszyklus einer Anlage übernimmt eine heterogene Gruppe von Mitarbeitern unterschiedliche Rollen in Bezug auf die Anlage. Dabei unterscheiden sich die Personen dieser Gruppen ganz wesentlich in ihrer Aufgabestellung, ihrem Bildungsstand, ihrem fachlichen Hintergrund oder in ihrem Erfahrungsschatz. Daraus ergeben sich unterschiedliche Anforderungen an die verschiedenen Systeme und Lösungen im Umfeld einer Anlage.

Bewährte Ansätze und Sprachen

Aufgrund der hohen Sicherheitsanforderungen und der hohen Komplexität, die mit der Anlagengröße einhergeht, haben sich in der Prozessleittechnik bzw. Automatisierungstechnik besondere Lösungsansätze bewährt und durchgesetzt. So ist zum Beispiel die Automatisierungspyramide¹ die grundlegende Architektur einer jeden Automatisierungslösung [111]. Außerdem haben sich eigene Programmiersprachen (z. B. in Deutschland in der Regel auf Basis von Funktionsbausteinen) zur Konfiguration der Leitsysteme durchgesetzt [111]. Diese Programmiersprachen sind in der IEC 61131-3 standardisiert [94].

¹<https://de.wikipedia.org/wiki/Automatisierungspyramide>

5.1.2 Neue Randbedingungen durch Industrie 4.0

Dynamische Änderungen

Ein wesentliches Ziel der Initiative Industrie 4.0 ist Wandelbarkeit [5]. Dabei geht diese über Flexibilität hinaus und soll (im Idealfall automatische) Adaptionen an ungeplante Änderungen ermöglichen [5]. Mögliche automatische Reaktionen auf eine ungeplante Änderung sind zum Beispiel die in [104] beschriebene Lastverteilung, die Nutzung des in [106] beschriebenen Konzepts zur Aktualisierung von Steuerungskomponenten oder die Ressourcenadaption [105].

Neue Rollen für den Menschen

Durch die starke Vernetzung und Digitalisierung sowie die deutlich stärkere Automatisierung im Rahmen der Initiative Industrie 4.0 wird die Rolle des Menschen im Bezug auf eine Anlage einem starken Wandel unterliegen, denn sowohl Kenntnisse und Wissensbestände als auch die Steuerung der Anlage werden vermehrt in das Aufgabenspektrum der Technik verschoben [112]. Deshalb sind neue Mensch-Technik-Interaktionskonzepte notwendig, damit der Mensch den neuen Aufgaben gewachsen ist und diese lösen kann [112].

Hochautomatisierung

In der NAMUR Empfehlung 161 werden die Grundlagen für einen autonomen oder fernbedienten Betrieb einer prozesstechnischen Anlage beschrieben [113]. Dabei wird ersichtlich, dass die Hochautomatisierung eine wesentliche Grundlage für jeden weiteren Schritt darstellt. Entscheidend dabei ist, dass der Verantwortungsbereich eines einzelnen Mitarbeiters in Zukunft wachsen wird. Ob Bedien- oder Wartungspersonal, in Zukunft wird der Verantwortungsbereich größer und damit das persönliche Wissen über die einzelne Anlage vermindert sein. Damit braucht es unterstützende Systeme, die die richtigen Informationen zum rechten Zeitpunkt bereitstellen können.

5.2 Anforderungen an das Konzept

Im Folgenden werden die funktionalen Anforderungen² an das Konzept dieser Arbeit kurz beschrieben. Auf diesem Weg wird festgelegt, welche Funktionen das Konzept besitzen muss. Die Anforderungen orientieren sich teilweise an den in Absatz 3.4 und Absatz 3.6 vorgestellten Anforderungen an ein Fehleranalysewerkzeug bzw. an eine Wissensrepräsentation.

A0: Erfüllung der Kriterien einer praktisch einsetzbaren Wissensrepräsentation

Die zu entwickelnde Wissensrepräsentation muss eine hinreichende Ausdrucksstärke besitzen und in der Lage sein, alle relevanten Begriffe abbilden zu können. Dabei ist die Erfüllung des Uniformitätskriteriums (vgl. Abschnitt 3.4) eine weitere Anforderung. Das heißt, dass gleiche Informationen auf die gleiche Art und Weise repräsentiert werden müssen. Gleichzeitig dürfen bestehende Informationen oder Wissen bei einer Integration in die Wissensrepräsentation nicht verloren gehen.

²[https://de.wikipedia.org/wiki/Anforderung_\(Informatik\)](https://de.wikipedia.org/wiki/Anforderung_(Informatik))

A1: Modellmäßige Kombination unterschiedlicher Anlagenbeschreibungen

Das Konzept muss verschiedenstes Wissen über eine Anlage für einen definierten Zeitpunkt vernetzen und integrieren können. Mögliches Wissen über eine Anlage beinhaltet die Anlagentopologie, Informationen aus dem Prozesshistorian oder Ereignisse aus Verwaltungsschalen der Anlagenkomponenten. Darüber hinaus sind beispielsweise auch Informationen aus einem elektronischen Schichtbuch denkbar. Dabei muss das Konzept...

- A1.0 auf bestehenden Modellen, Ontologien und Konzepten aufbauen. Nur so kann eine Kompatibilität mit anderen Lösungen aus der Domäne sichergestellt werden.
- A1.1 Änderungen innerhalb des Betrachtungszeitraums berücksichtigen und darstellen können. Dazu muss es in der Lage sein, Zeitinformationen zu integrieren (vgl. Anpassbarkeit in Abschnitt 3.6).
- A1.2 in der Lage sein, Nachbarschaftsbeziehungen aufzuzeigen. Wesentliche Aufgabe des Konzepts ist das Aufdecken von Abhängigkeiten zwischen einzelnen Anlagenkomponenten. Deshalb ist es von großer Bedeutung, dass Strukturinformationen und Beziehungen adäquat abgebildet werden (vgl. Erklärungsmöglichkeit 3.6).
- A1.3 Nachbarschaftsbeziehungen näher beschreiben können. Zur näheren Untersuchung der Abhängigkeiten und zur vereinfachten Identifikation von Ursache-Wirkungsbeziehungen muss das Konzept in der Lage sein, Nachbarschaftsbeziehungen näher zu spezifizieren (vgl. schnelle Erkennung und Diagnose, Fähigkeit zur Identifikation multipler Ursachen bzw. Fehler sowie Isolierbarkeit in Abschnitt 3.6).
- A1.4 Verschiedene Produktnetze unterscheiden können. Die Unterscheidung kann je nach verfügbaren Informationen anhand von Druckstufen oder unterschiedlichen Produkten erfolgen (vgl. schnelle Erkennung und Diagnose, Fähigkeit zur Identifikation multipler Ursachen bzw. Fehler sowie Isolierbarkeit in Abschnitt 3.6).
- A1.5 die Möglichkeit bieten, neue Aspekte leicht zu integrieren. Aufgrund der fortschreitenden Digitalisierung, zum Beispiel im Rahmen der Industrie 4.0 Initiative, werden zukünftig immer mehr Informationen maschinenlesbar zu Verfügung stehen. Das Konzept und die Implementierung müssen für diesen Fall in der Lage sein, neue Aspekte leicht zu integrieren (vgl. Anpassbarkeit in Abschnitt 3.6). Eine modulare Erweiterbarkeit ist also essentiell.

A2: Rückwirkungsfreiheit auf den Anlagenbetrieb

Wie in Abschnitt 5.1.1 beschrieben, besteht in der Prozessindustrie ein hohes Gefährdungspotenzial. Damit einher gehen hohe Sicherheitsanforderungen an die eingesetzten Automatisierungslösungen. Aus diesem Grund dürfen zusätzlich eingesetzte Analysewerkzeuge, wofür diese Arbeit ein Konzept anstrebt, auf keinen Fall Einfluss auf den Anlagenbetrieb haben. Deshalb muss die Rückwirkungsfreiheit auf den Anlagenbetrieb sichergestellt werden.

A3: Automatische Sicherstellung der Konsistenz der Wissensrepräsentation

Damit die Wissensrepräsentation für spätere Untersuchungen geeignet ist, muss die Konsistenz³ der abgespeicherten Daten automatisch sichergestellt werden.

A4: Minimaler Konfigurations- und Wartungsaufwand

Durch die hohe Anlagendiversität und den großen Bestand bestehender Anlagen einerseits und der sich andererseits verändernden Rolle des Menschen (siehe Abschnitt 5.1.1) besteht das Risiko eines Zielkonflikts zwischen zusätzlichen Kosten für Wartung sowie Inbetriebnahme und der benötigten Informationstransparenz. Deshalb muss sichergestellt sein, dass das Konzept mit einem möglichst geringen Wartungs- und Konfigurationsaufwand auskommt und im Idealfall neue Aspekte möglichst automatisch berücksichtigt.

A5: Trennung von Wissen und Anwendung

Zur Sicherstellung der Wartbarkeit und damit die Wissensrepräsentation in Zukunft auch anderen Analysewerkzeugen zur Verfügung steht, ist eine klare Trennung zwischen der Wissensrepräsentation und den dazugehörigen Anwendungen von wesentlicher Bedeutung. Durch die Trennung wird sichergestellt, dass die Wissensbasis und die Anwendungen getrennt von einander gewartet werden können. Außerdem verhindert diese Trennung, dass Domäneexperten sich bei der Erstellung einer Wissensbasis mit Implementierungsdetails der Anwendung auseinandersetzen müssen [70]. Durch die klare Fokussierung können hochwertigere Wissensrepräsentationen aufgebaut werden. Außerdem lassen sich klare Verantwortungsbereiche festlegen.

A6: Nachverfolgbarkeit

Entscheidend für den Einsatz des Konzepts ist das Vertrauen in die darin verwalteten Daten. Deshalb ist eine klare Nachverfolgbarkeit der Daten und ihres Ursprungs essentiell. Aus diesem Grund muss das Konzept die Möglichkeit bieten, den Ursprung eines Datums zu erkennen.

A7: Sicherstellung der Vertraulichkeit

Das Konzept dieser Arbeit integriert Daten aus den unterschiedlichsten Quellen. Dabei können diese unterschiedliche Vertraulichkeitsanforderungen besitzen. Deshalb muss das Konzept die Vertraulichkeit der Daten sicherstellen können. Dafür muss es sowohl aus Nutzer- als auch aus Datenperspektive möglich sein, die Vertraulichkeit zu definieren.

A8: Definierter Prozess zur Erweiterung um neue Aspekte

Die Untersuchung der unterschiedlichen Ereignistypen in Abschnitt 4.3, zeigt, dass eine abgeschlossen Beschreibung aller relevanten Ereignisse nicht möglich ist. Aus diesem Grund kann es je nach Anwendungsfall notwendig sein, dass das Konzept um weitere Aspekte erweitert werden muss. Deshalb muss ein Prozess zur Erweiterung um weitere Aspekte definiert werden.

³[https://de.wikipedia.org/wiki/Konsistenz_\(Datenspeicherung\)](https://de.wikipedia.org/wiki/Konsistenz_(Datenspeicherung))

5.3 Anforderungen an die Modellhierarchie

Die Datenqualität hat einen entscheidenden Einfluss auf die spätere Nutzung. Dabei wird die Qualität der Daten unter anderem wesentlich von den verwendeten Metamodellen bzw. der Modellhierarchie beeinflusst. Nur eine elegant gestaltete Modellhierarchie erlaubt eine effiziente Abfrage und Erkundung des vom System verwalteten Wissens. Ein schlecht entworfenes Metamodell hat zum Beispiel unnötig komplexe Abfragen und damit einhergehend auch verlängerte Antwortzeiten zur Folge. Damit die Designentscheidungen beim Entwurf des Metamodells nachvollziehbar getroffen werden können, werden in diesem Abschnitt Anforderungen an die Modellhierarchie bzw. Qualitätskriterien für die Modellierung definiert. Diese basieren vor allem auf dem Konzept der Gebrauchstauglichkeit bzw. dem Datenqualitätsmodell, das in Abschnitt 3.2 bzw. Tabelle 3.1 und 3.2 vorgestellt wurde. Von den im Modell vorgestellten Datenqualitätsattributen werden vor allem die Attribute Vollständigkeit, Konsistenz, Zugänglichkeit, Effizienz, Nachverfolgbarkeit, Verständlichkeit sowie Portabilität durch eine Modellhierarchie bzw. Metamodelle beeinflusst. Zusätzlich soll hier die Anforderung der Flexibilität bzw. Adaptierbarkeit eingeführt werden. Es ist davon auszugehen, dass eine vollständige Definition des Metamodells a priori nicht möglich ist, sondern, dass im Laufe des Lebenszyklus des Systems weitere Datenklassen hinzukommen werden. Deswegen muss die Modellhierarchie eine gewisse Flexibilität zur Adaption und Erweiterung aufweisen.

Q1: Vollständigkeit

Im Rahmen der Definition des Begriffs Kontext für die Prozessindustrie wurden die benötigten Informationsklassen herausgearbeitet. Diese sind Zeitpunkt, relative Position, Akteur, semantischer Zusammenhang sowie Aktivität. Die Modellhierarchie muss also alle notwendigen Begriffe dieser Gesichtspunkte abbilden können, damit sie eine ausreichende Vollständigkeit erreicht. Damit ist die Vollständigkeit ein absolutes Kriterium. Entweder das Metamodell erlaubt eine vollständige Abbildung aller Begriffe und ist vollständig oder es erlaubt die Abbildung nicht und ist damit auch nicht vollständig. Es ist fraglich, ob eine absolute Vollständigkeit erreicht werden kann. Aus diesem Grund kommen den Kriterien Flexibilität und Adaptierbarkeit eine besondere Bedeutung zu.

Q2: Flexibilität bzw. Adaptierbarkeit

Da davon ausgegangen wird, dass eine vollständige Definition des Metamodells a priori nicht möglich ist, sondern dass Erweiterungen im Rahmen des Einsatzes notwendig werden können, ist Flexibilität ein wesentliches Qualitätskriterium für das zu definierende Metamodell. Die Flexibilität kann durch die Anzahl und die Komplexität von notwendigen Anpassungen an dem Metamodell für im Anwendungskontext vorauszusehende Adaptionen evaluiert werden.

Q3: Konsistenz und Portabilität

Ein voll umfassend definiertes Metamodell verhindert die Abbildung von inkonsistenten Daten und ermöglicht so die Übertragbarkeit in andere Systeme. Beispielsweise kann ein Metamodell zur Modellierung der Betriebszugehörigkeit eines Angestellten festlegen, dass das Einstellungsdatum eines Mitarbeiters immer nach dessen Geburtsdatum liegen muss

und kann so die Modellierung inkonsistenter Daten verhindern. Etwaige Definitionslücken resultieren in Inkonsistenzen oder Unvollständigkeiten in der späteren Anwendung. Deren Aufdeckung dient auch zur Evaluation des Metamodells.

Q4: Zugänglichkeit, Verständlichkeit und Effizienz

Grundsätzlich gilt, dass die Abbildung von Informationen und deren Aufbereitung für diverse Nutzergruppen unabhängig voneinander gehandhabt werden können. Allerdings definiert das Metamodell Semantik und Syntax zur Beschreibung der verfügbaren Daten. Damit legt es fest, auf welchem Weg die Daten und deren Beziehungen erkundet werden können und hat damit erheblichen Einfluss auf die Zugänglichkeit sowie Verständlichkeit der Daten und die Effizienz des Datenzugriffs. Deshalb sind Zugänglichkeit, Verständlichkeit und Effizienz wesentliche Qualitätskriterien für ein Metamodell. Eine Methode zur Evaluierung dieser Kriterien ist die Betrachtung der Komplexität von für den definierten Nutzungskontext häufig zu erwarteten Datenabfragen (vgl. Kompetenzfragen in Abschnitt 1.2). Eine weitere Möglichkeit stellt die Messung des Ressourcenverbrauchs (z. B. Zeit) für eine bestimmte Abfrage dar. Dieser Ansatz ist jedoch sehr technologieabhängig.

5.4 Anforderungen des Konzepts an die Umwelt

Damit das in dieser Arbeit vorgestellte Konzept erfolgreich umgesetzt werden kann, muss die Umwelt einige Voraussetzungen erfüllen. Diese werden nachfolgend kurz beschrieben.

V1: Durchgängige Digitalisierung

Wesentliche Idee des Konzepts ist die Verknüpfung verschiedener Informationsquellen zu einer Wissensbasis, mithilfe derer anschließend neue Schlüsse gezogen werden können. Voraussetzung dafür ist eine durchgängige Digitalisierung der Informationsquellen. Darüber hinaus ist es notwendig, dass die digitalisierten Informationen basierend auf expliziten Datenmodellen bereitgestellt werden. Ohne explizite Datenmodelle ist das Übertragen der Informationen in eine Wissensrepräsentation nur unter erheblichem sowie manuellem Aufwand möglich. Damit das entwickelte Konzept effizient angewendet werden kann, müssen alle relevanten Komponenten mindestens die Anforderungen der CP-Klassifikation CP33 (individuell bekannte, aktiv kommunikationsfähige Komponente) nach [99] erfüllen.

V2: Eindeutiges Namenskonzept

Damit die unterschiedlichen Informationsquellen miteinander verknüpft werden können, ist eine eindeutige Identifikation der verschiedenen Begriffe über die unterschiedlichen Informationsquellen hinweg notwendig. Nur wenn sicher gestellt ist, dass Denkeinheiten aus den diversen Informationsquellen einander richtig zugeordnet werden können, ist eine sinnvolle Informationsverknüpfung möglich.

V3: Aktualität der Informationsquellen

Damit im Konzept aktuelle Informationen berücksichtigt werden können, müssen diese in den verschiedenen Informationsquellen zur Verfügung stehen. Sollten diese nicht aktuell

sein, haben auch alle darauf aufbauenden Systeme keine Chance, aktuell zu sein. Dies kann insbesondere bei der Anlagenbeschreibung eine Herausforderung darstellen, insbesondere dann, wenn Änderungen an der Anlage nicht in die Dokumentation eingepflegt werden.

V4: Zeitsynchronität

Da im Rahmen des Konzepts verschiedene Daten aus unterschiedlichen Datenquellen miteinander verknüpft werden, ist es entscheidend, dass zwischen den verschiedenen Datenquellen Zeitsynchronität herrscht. Ansonsten ist die Abbildung der unterschiedlichen Daten auf einer gemeinsamen Zeitachse nicht möglich.

6 Entwicklung der Systemstruktur

Nachdem in den vorherigen Kapiteln unterschiedliche Informationsquellen betrachtet, die Anforderung an das Kontextualisierungssystem diskutiert und verschiedene Modellierungskonzepte vorgestellt wurden, wird in diesem Kapitel der gewählte Lösungsansatz beschrieben. Dazu wird entsprechend der gewählten und in Abschnitt 1.2 vorgestellten Vorgehensweise der Schritt einer leichtgewichtigen Ontologieentwicklung fortgesetzt und eine Informationsmodellhierarchie definiert.

Abbildung 6.1 stellt das Konzept als Skizze dar. Kern des Konzepts ist die Konstruktion eines semantischen Netzes, das die verschiedenen Daten aus den unterschiedlichen Quellen miteinander verknüpft und so die Beziehungen zwischen den einzelnen Aspekten aufzeigt und dadurch den Kontext der individuellen Daten ersichtlich macht. Dabei wird der Closed-World-Ansatz verfolgt. Im Rahmen dieser Arbeit wird das konstruierte semantische Netz als Nachbarschaftsgraph bezeichnet, da es die Erkundung der Daten auf Grundlage kontextueller Nähe bzw. Nachbarschaften zwischen den einzelnen Daten der unterschiedlichen Quellen ermöglicht.

Um eine Diskussion der Ergebnisse dieser Arbeit innerhalb der Prozessindustrie zu ermöglichen, wurden Teile dieser bereits in [53] veröffentlicht.

6.1 Definition der Systemarchitektur

In Abbildung 6.2 ist die Systemarchitektur des im Rahmen der vorliegenden Arbeit definierten Kontextualisierungssystems dargestellt. Sie zeigt aus einer sehr abstrakten Perspektive die wesentlichen Komponenten und ordnet diese in die typischerweise bestehende Systemlandschaft ein.

Wie zuvor beschrieben und in der Abbildung dargestellt, bestehen in der Prozessindustrie eine Vielzahl unterschiedlicher Datenquellen, die die benötigten Informationen für den oben definierten Kontext bereithalten. Damit dieser Kontext für andere Systeme nutz- und erkundbar wird, müssen die Daten aus diesen Quellen kombiniert und verknüpft werden. Dazu wird im Rahmen dieser Arbeit ein Kontextualisierungssystem entwickelt. Grundlage des Systems ist eine Anlagenmodellierung in Form eines semantischen Netzes (vgl. Kapitel 3.4.1). Dadurch wird die semantische Nähe einzelner Daten und deren Struktur ersichtlich sowie die Suche nach Abhängigkeiten ermöglicht. Ein wesentlicher Nachteil von semantischen Netzen ist die selten formal spezifizierte Semantik und der daraus entstehende Interpretationsspielraum. Zur Überwindung dieses Nachteils wird im Rahmen der vorliegenden Arbeit auf Basis einer Modellhierarchie, ein metamodellbasierter Ansatz gewählt. Nur so kann ein einheitlicher Zugriff auf alle modellierten Aspekte gewährleistet und eine hohe Gebrauchstauglichkeit sichergestellt werden. Fundament dieser Modellhierarchie ist ein Metametamodell (1). Darauf aufbauend werden für einzelne Aspekte einer Anlage Aspektmetamodelle (2) definiert. Diese dienen schlussendlich zur Erstellung eines konkreten Nachbarschaftsgraphen (3) für eine spezifische Anlage innerhalb einer Datenbank. Der

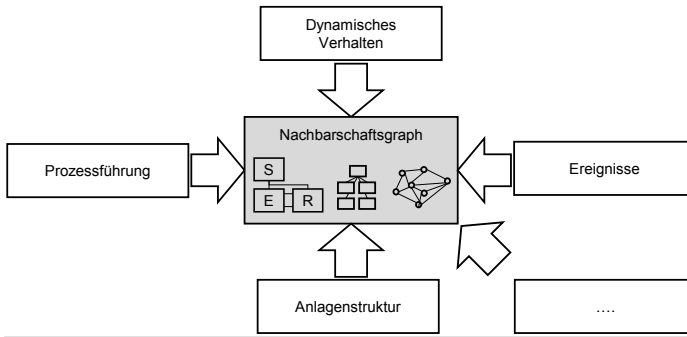


Abbildung 6.1: Der Nachbarschaftsgraph als Grundlage zur Anlagenmodellierung in Form eines semantischen Netzes

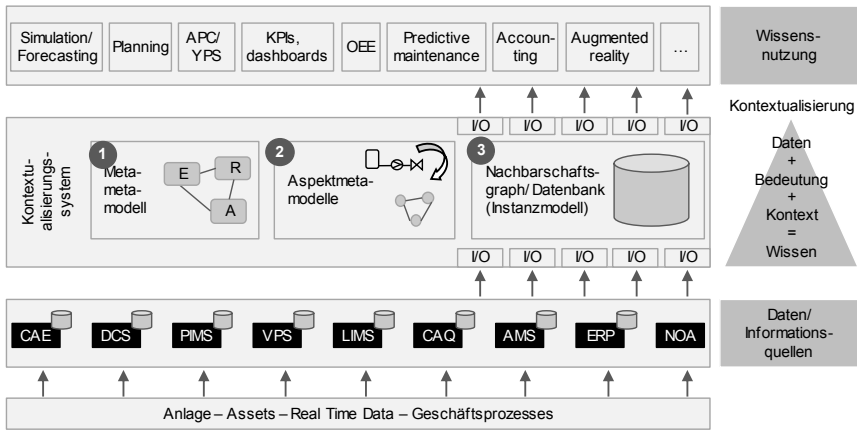


Abbildung 6.2: Systemarchitektur und Eingliederung in die bestehende Systemlandschaft angelehnt an [9]

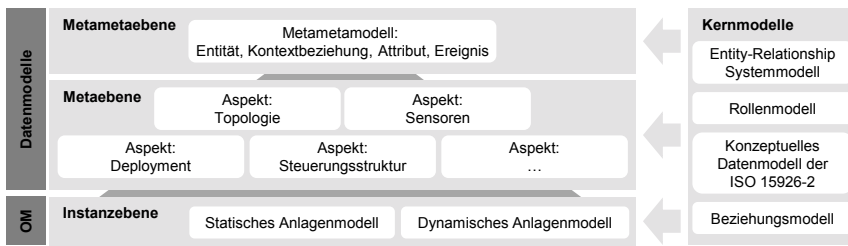


Abbildung 6.3: Modellhierarchie (OM: Objektmodell)

Nachbarschaftsgraph stellt also das eigentliche semantische Netz der Anlage dar. Sowohl das Metametamodell als auch die Aspektmetamodelle sind in diesem implizit enthalten.

6.2 Entwicklung der Modellhierarchie

Abbildung 6.3 zeigt die dieser Arbeit zugrundeliegende Modellhierarchie. Sie ist in drei Ebenen gegliedert, wobei der Abstraktionsgrad von oben nach unten über die einzelnen Ebenen hinweg abnimmt. Zusätzlich ist eine Zuordnung zu dem in Abschnitt 3.3 bzw. in Abbildung 3.3 vorgestellten Ordnungssystem dargestellt. Grundlage der Modellierung sind die Kernmodelle aus der Literatur. Sie beeinflussen die Modellierung auf allen Ebenen und sind deshalb orthogonal zu der Hierarchie dargestellt. Konkret liefern die Kernmodelle Templates für die Erstellung der Metameta- und Metamodelle. Auf diesem Weg beeinflussen sie auch die Instanzebene bzw. das spätere Objektmodell. Neben den Kernmodellen aus der Literatur wird hier aufgrund der großen Bedeutung für diese Arbeit, zusätzlich der Begriff der Beziehung im Detail beleuchtet und die Definition aus dem ER-Systemmodell erweitert, detaillierter ausspezifiziert und mit dem Konzept aus der ISO 15926-2 kombiniert. Dazu wird der Begriff Beziehung im folgenden Abschnitt in Form eines Kernmodells beschrieben. Basierend auf diesen Grundlagen wird auf der Metametaebene ein Metameta-modell für den Anwendungsfall dieser Arbeit beschrieben. Es definiert die Begriffe Entität, Beziehung, Attribut und Ereignis, sodass sie sich für die Modellierung der Betriebsphase einer prozesstechnischen Anlage eignen und die vier Informationsklassen des Kontext einer Anlage (Zeitpunkt, relative Position, Akteur sowie Aktivität) modelliert werden können.

Das Metametamodell dient dann auf der Metaebene als Grundlage zur Beschreibung unterschiedlicher Perspektiven oder Aspekte der Anlage. Dazu werden die vom Metameta-modell definierten Begriffe weiter ausspezifiziert und detailliert beschrieben.

Auf Basis der Metamodelle kann anschließend eine konkrete Anlage modelliert werden. Dazu werden Instanzen aus den einzelnen Aspektmetamodellen angelegt und über die definierten Beziehungen miteinander verknüpft.

6.3 Beziehungsmodell

In diesem Abschnitt wird der Begriff Beziehung näher beleuchtet und in Form eines Kernmodells beschrieben. Deshalb orientiert sich der Aufbau der Beschreibung an der DIN SPEC 40912 [30], welche unterschiedliche Kernmodelle normiert. Das Modell baut auf dem ER-Systemmodell auf und berücksichtigt wesentliche Aspekte des konzeptuellen Datenmodells der ISO 15926-2 (vgl. Abb. 4.6) [78]. Das Beziehungsmodell beschreibt den Begriff Beziehung im Detail.

6.3.1 Konzept

Eine Beziehung beschreibt ein wechselseitiges Verhältnis, den inneren Zusammenhang¹ oder die Verbindungen zwischen (mindestens) zwei Entitäten mit einer gewissen zeitlichen Dauer. Dabei nehmen die Entitäten innerhalb der Beziehung bestimmte Rollen ein.

¹<https://www.duden.de/rechtschreibung/Beziehung>

6.3.2 Anwendungsbereich

Das Beziehungsmodell legt die grundlegenden Begriffe zur Beschreibung von Verbindungen zwischen Entitäten fest und dient einer klaren und eindeutigen Modellierung von (komplexen) Strukturen, Zusammenhängen und Abhängigkeiten (z. B. in Systemen) und ist somit in vielen anderen Kernmodellen implizit enthalten (z. B. ER-Systemmodell). Dabei erweitert das Modell die bekannten Konzepte durch Berücksichtigung der Beziehungsdauer um eine dynamische Komponente und ermöglicht so die Beschreibung von zeitlichen Abläufen. Außerdem werden die Rollen, die eine Entität im Rahmen einer Beziehung einnehmen kann, explizit benannt.

6.3.3 Allgemeine Spezifikation

Verbindungen zwischen Entitäten werden durch Beziehungen beschrieben. Dabei nehmen die Entitäten im Rahmen der Beziehung bestimmte Rollen oder Funktionen ein.

Das Einnehmen einer Rolle ist durch ein Ereignis gekennzeichnet. Dieses Ereignis markiert den Beginn einer Beziehung. Auf die gleiche Art und Weise wird eine Beziehung auch durch ein Ereignis beendet. Dies bedeutet, dass ein Ereignis mehrere Beziehungen miteinander verknüpfen kann.

Ergibt sich eine Beziehung aus nur einem Rollentyp, handelt es sich um eine nicht gerichtete Beziehung. Ergibt sich eine Beziehung aus mehr als einem Rollentyp, handelt es sich um eine gerichtete Beziehung.

Beispielsweise übernehmen alle Kolleg:innen in einer Beziehung des Typs Kollegium die Rolle Kolleg:in. Im Gegensatz dazu übernimmt in einer Beziehung Leitung der/die Chef:in die Rolle Leiter:in und alle anderen die Rolle Mitarbeiter:in. Dementsprechend handelt es sich bei der ersten Beziehung um eine ungerichtete und bei der zweiten Beziehung um eine gerichtete Beziehung. Dabei ist die Wirkungsrichtung der Beziehung nicht immer klar und hängt von der Perspektive ab. So könnte beispielsweise der Erfolg einer Person in der Rolle Leiter:in anhand der Erfolge der Personen in der Rolle Mitarbeiter:in gemessen werden. Aus dieser Perspektive besteht also eine Wirkung der Rolle Mitarbeiter:in auf die Rolle Leiter:in. Eine umgekehrte Wirkrichtung kann ggf. aus der Perspektive der Zufriedenheit beobachtet werden.

Sind verschiedene konkrete Entitäten eine konkrete Beziehung eingegangen, so kann diese Beziehung nur insgesamt aufgelöst werden. Veränderungen, wie beispielsweise der Austausch einer Entität, sind nicht erlaubt. Soll eine Entität innerhalb einer Beziehung ausgetauscht werden, ist dies nur durch das Ende der ursprünglichen und den Beginn einer neuen Beziehung möglich.

6.3.4 Formale Spezifikation

In Abbildung 6.4 werden die verschiedenen Elemente einer Beziehung und deren Zusammenhänge dargestellt. Diese werden im Folgenden näher erläutert.

Beziehung

Die Beziehung beschreibt den inneren Zusammenhang und die Verbindung bzw. das wechselseitige Verhältnis zwischen Entitäten. Beziehungen sind klassifiziert und besitzen zur

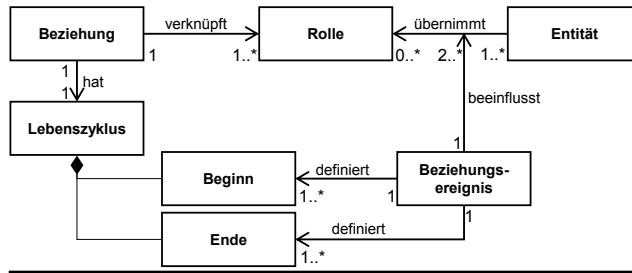


Abbildung 6.4: Grundstruktur einer Beziehung

näheren Beschreibung Eigenschaften. Dementsprechend existieren unendlich viele Beziehungsklassen. Der Lebenszyklus einer Beziehung ist linear und beginnt und endet immer mit einem Ereignis. Eine Beziehung ergibt sich, wenn Entitäten Rollen innerhalb einer Beziehung übernehmen. Dabei setzt sich eine Beziehung aus mindestens einer Rolle zusammen, die zur Knüpfung einer Beziehung von mindestens zwei Entitäten eingenommen werden muss.

Rolle

Die Rolle definiert einerseits die Funktionalität, Aufgabe oder Verantwortung, die eine Entität innerhalb einer Beziehung übernimmt. Sie legt andererseits die Anforderung an die Entität zur Ausfüllung der Rolle fest. Rollen sind klassifiziert und besitzen Eigenschaften. Eine Rolle ist immer genau einer Beziehung zugeordnet. Innerhalb einer Beziehungsinstantz wird eine Rolle von mindestens einer Entität eingenommen. Außerdem ist innerhalb einer einmal instanziierten Beziehung die Rollen-Entität-Zuordnung festgelegt. Dementsprechend können in einer bestehenden Beziehung keine Entitäten ausgetauscht oder hinzugefügt werden. Änderungen der Zuordnung haben immer das Vergehen der aktuellen und das Entstehen einer neuen Beziehung zur Folge.

Entität

Entitäten sind physische oder gedankliche Einheit, die eine Rolle einer Beziehung ausfüllen und so eine Verbindung mit einer anderen Entität eingehen kann. Eine Entität kann beliebig viele aber auch keine Rollen in unterschiedlichen Beziehungen einnehmen, wenn dies die Anforderungen der Beziehungsrollen zulassen. Entitäten sind klassifiziert und besitzen Eigenschaften. Direkte Verbindungen zwischen Entitäten sind nicht möglich.

Beziehungsereignis

Hierbei handelt es sich um ein Geschehnis bzw. einen Vorgang, der zur Folge hat, dass mindestens zwei Entitäten jeweils eine Rolle einer Beziehungsklasse übernehmen bzw. nicht mehr übernehmen und so eine Instanz des Beziehungstyps bilden bzw. auflösen. Damit führt ein Beziehungsereignis zu Veränderungen des Lebenszyklus mindestens einer Beziehung bzw. der beteiligten Entitäten. Ein Beziehungsereignis kann mehrere Beziehungen

miteinander verknüpfen. In diesem Fall legt das Beziehungsereignis jeweils Ende oder Beginn der verknüpften Beziehungen fest.

Im Unterschied zum allgemeinen Ereignisbegriff umfasst das Beziehungsereignis nicht die Ereignisse, die ausschließlich eine diskrete Veränderung eines kontinuierlichen Werts zur Folge haben.

Lebenszyklus

Eine Beziehung besitzt einen linearen Lebenszyklus. Dieser beginnt und endet mit einem Ereignis. Der **Beginn** beschreibt das Entstehensereignis und markiert den zeitlichen Start einer Beziehung. Der genaue Zeitpunkt und das zugehörige Ereignis sind nicht immer bekannt. Der Zeitpunkt kann dem Ende-Zeitpunkt anderer Beziehungen entsprechen. In diesem Fall können diese Beziehungen über das Ereignis verknüpft sein. Das **Ende** beschreibt das Vergehensereignis und markiert das zeitliche Ende einer Beziehung. Der genaue Zeitpunkt ist nicht immer bekannt, insbesondere dann, wenn er in der Zukunft liegt. Der Zeitpunkt kann dem Beginn-Zeitpunkt anderer Beziehungen entsprechen. In diesem Fall können diese Beziehungen über das zugehörige Ereignis verknüpft sein.

Regeln

- R1** Eine Beziehung besitzt mindestens eine Rolle.
- R2** In einer Beziehung übernehmen mindestens zwei unterschiedliche Entitäten jeweils genau eine Rolle.
- R3** In einer Beziehung kann eine Entität nur genau eine Rolle einnehmen.
- R4** Eine Entität kann beliebig viele Rollen in unterschiedlichen Beziehungen einnehmen.
- R5** In einer bestehenden Beziehung kann die Rollen-Entität-Zuordnung nicht verändert werden, das heißt, dass weder Entitäten hinzugefügt noch entnommen werden können.

6.3.5 Diskussion des Beziehungsmodells

Die in diesem Kapitel beschriebene Definition des Beziehungsmodells ist sehr allgemein. Sie schließt sowohl $1\text{-zu-}1$, $1\text{-zu-}n$ als auch $n\text{-zu-}m$ -Beziehungen mit ein. Im Rahmen einer Anwendung des Beziehungsmodells kann es zur Reduktion der Komplexität sinnvoll sein, diesen Freiheitsgrad einzuschränken und nur $1\text{-zu-}1$ -Beziehungen zuzulassen. $1\text{-zu-}n$ -Beziehungen können dann durch n $1\text{-zu-}1$ -Beziehungen und $n\text{-zu-}m$ -Beziehungen durch $n*m$ $1\text{-zu-}1$ -Beziehungen dargestellt werden.

6.4 Spezifikation des Metametamodells

In diesem Abschnitt werden die grundlegenden Eigenschaften des dem Kontextualisierungssystem zugrundeliegenden Metametamodells beschrieben. Es muss in der Lage sein, die Beziehungen zwischen den Daten und die zeitlichen Veränderungen in diesen Beziehungen explizit zu modellieren.

Die Elemente des definierten Metametamodells sowie deren Beziehungen sind in Abbildung 6.5 dargestellt. Als Grundlage für das Metametamodell dienen das ER Systemmo-

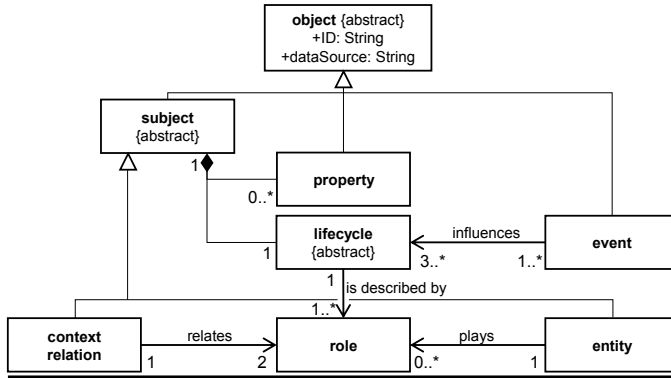


Abbildung 6.5: Metamodell zur Anlagenbeschreibung mithilfe von Beziehungen

dell [30], das eine Systembeschreibung auf Basis der beiden Begriffe Entität (entity) und Relation (relation) ermöglicht, sowie das konzeptionelle Datenmodell der ISO 15926-2 bzw. das aus beiden resultierende Beziehungsmodell aus Kapitel 6.3. Aus diesem wird zur detaillierten Beschreibung der Beziehungen das Rollen- und Ereigniskonzept aufgegriffen. Damit wird das Modellieren von zeitlichen Veränderungen möglich.

Zur Reduktion der Komplexität werden für das Metamodell die zuvor diskutierten Vereinfachungen angewendet und nur 1-zu-1-Beziehungen zugelassen.

6.4.1 Objekt (object)

Der Begriff Objekt dient als gemeinsame Oberklasse für Beziehungen (relation), Entitäten (entity), Eigenschaften (property) sowie Ereignisse (event). Die Klasse legt fest, dass alle Objekte eine eindeutige ID zur Identifikation besitzen. Dementsprechend gilt:

- R1** Jedes Objekt besitzt eine im Namensraum eindeutige ID.
- R2** Zu jedem Objekt wird die Datenquelle angegeben.

6.4.2 Subjekt (subject)

Die abstrakte Oberklasse Subjekt (subject) dient als Grundlage zur Modellierung von Entitäten (entity) und Beziehungen (relation). Diesen beiden Begriffen gilt das Hauptinteresse dieser Arbeit. Ein jedes Subjekt zeichnet sich durch einen Lebenszyklus (lifecycle) aus. Außerdem kann jedes Subjekt mithilfe von beliebig vielen Eigenschaften (property) näher beschrieben werden. Es gilt:

- R1** Ein Subjekt besitzt genau einen Lebenszyklus.
- R2** Ein Subjekt wird durch beliebig viele Eigenschaften beschrieben.

6.4.3 Eigenschaften (property)

Mithilfe von Aussagen zu Eigenschaften können Subjekte detaillierter beschrieben werden. Dabei existieren unterschiedliche Möglichkeiten zur Umsetzung dieser Beschreibung. Die

einfachste Möglichkeit ist die Angabe eines Schlüssel-Wert-Paars, wobei der Schlüssel die Eigenschaft benennt und der Wert eine Aussage über die Ausprägung trifft (z. B. Schlüssel: Rohrleitungslänge, Wert: 3.73 m; Schlüssel: Seriennummer, Wert: XYZ1234). Zur Sicherstellung der Eindeutigkeit der mithilfe des Schlüssel-Wert-Paars getroffenen Aussagen empfiehlt sich zusätzlich mindestens die Angabe einer Referenz auf ein entsprechendes Wörterbuch, wie zum Beispiel eCI@ss, sodass eine eindeutige Interpretation der bereitgestellten Eigenschaften sichergestellt werden kann.

Handelt es sich bei der Eigenschaft um einen sich über die Zeit verändernden Wert, wie beispielsweise die Temperatur in einem Raum, ist die Aussage um einen Zeitstempel zu ergänzen (z. B. Schlüssel: Raumtemperatur, Wert: 21.7 °C, Zeitstempel: UTC 2019-05-12T10:15:00).

Für Eigenschaftsaussagen, die höheren Anforderungen genügen sollen, kann das in der DIN SPEC 92000 [114] definierte Konzept der Eigenschaftswertaussagen aufgegriffen und in das Metametamodell integriert werden. Hier empfiehlt sich die Verwendung des Eigenschaftswertaussagencontainers insbesondere für kontinuierliche Aussagen zu einer Eigenschaft. Folgende Regeln gelten für Eigenschaften:

- R1** Ein Attribut kann nur im Zusammenhang mit einer Relation oder einer Entität bestehen.
- R2** Eine einmal getroffene Aussage zu einer Eigenschaft eines Attributs ändert sich innerhalb des Betrachtungszeitraums nicht.

6.4.4 Lebenszyklus (lifecycle)

Jedes Subjekt besitzt einen eindeutigen Lebenszyklus. Dieser ist linear, wird von Ereignissen (event) beeinflusst und kann mithilfe von Rollen (role) beschrieben werden. Hierbei muss zwischen Beziehungen (relation) und Entitäten (entity) unterschieden werden.

Wie in Abbildung 6.5 dargestellt setzt eine Beziehung genau zwei Rollen in Relation. Dementsprechend ist der Lebenszyklus einer Beziehung dadurch ausgezeichnet, dass er mit einem Ereignis beginnt, das dazu führt, dass die beiden Rollen durch eine Entität eingenommen werden und dass er mit dem Ereignis endet, das zur Folge hat, dass beide Entitäten ihre jeweilige Rolle nicht mehr ausfüllen.

Der Lebenszyklus einer Entität dagegen lässt sich durch die verschiedenen Rollen beschreiben, die eine Entität über die Dauer ihres Lebenszyklus einnimmt. Sowohl für den Lebenszyklus einer Entität als auch einer Beziehung gilt:

- R1** Ein Lebenszyklus ist immer linear.
- R2** Anfang und Ende eines Lebenszyklus werden durch Ereignisse markiert.

Im Gegensatz zu dem Lebenszyklusmodell aus der DIN SPEC 40912 [30] stehen bei der hier vorgestellten Betrachtungsweise die Beziehungen im Vordergrund. Der Lebenszyklus einer Entität ergibt sich nur implizit aus den Rollen, die die Entität im Laufe ihrer Existenz einnimmt. Damit wird der Lebenszyklus einer Entität immer in Relation zu anderen Entitäten beschrieben. Aus dieser Perspektive ist eine klare Unterscheidung zwischen Entstehungs-, Übergangs-, Lebenszyklus- und Vergehensprozessen, wie sie in der DIN SPEC 40912 beschrieben werden, nicht eindeutig möglich.

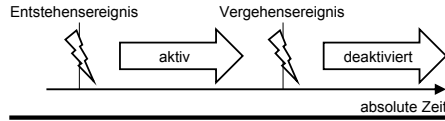


Abbildung 6.6: Abstrakter Lebenszyklus einer Beziehung

6.4.5 Ereignis (event)

Ein Ereignis (event) ist ein Geschehnis oder ein Vorgang, der zur Folge hat, dass mindestens eine neue Beziehung entsteht oder mindestens eine bestehende Beziehung vergeht. Damit beeinflusst ein Ereignis immer den Lebenszyklus von mindestens drei Subjekten. Sowohl die beiden Entitäten, die die Beziehung knüpfen, als auch die Beziehung selbst werden durch das Ereignis beeinflusst. Ereignisse sind zeitlos und besitzen keinen eigenen Lebenszyklus. Es gilt daher:

- R1** Ein Ereignis hat das Entstehen und/oder Vergehen mindestens einer Beziehung zur Folge.
- R2** Ein Ereignis beeinflusst mindestens drei Subjekte.
- R3** Ein Ereignis ist zeitlos.
- R4** Ein Ereignis besitzt keinen Lebenszyklus.

6.4.6 Kontextbeziehung (context relation)

Mithilfe von Kontextbeziehungen (context relation) können für den Kontext relevante Zusammenhänge, Verknüpfungen oder wechselseitige Verhältnisse zwischen Entitäten modelliert werden. Beispielsweise Spezialisierungs- oder Generalisierungsbeziehungen haben in der Regel keine Relevanz für den Kontext. Für die Modellierung werden die verschiedenen Rollen, die eine Entität innerhalb einer Beziehung übernehmen kann, beschrieben. Dabei verknüpft eine Beziehung genau zwei unterschiedliche Rollen, die bei einer Instanziierung von jeweils genau einer Entität eingenommen werden.

Abbildung 6.6 zeigt den abstrakten Lebenszyklus einer Beziehung. Zu erkennen ist, dass die Beziehung durch ein Entstehensereignis geknüpft und durch ein Vergehensereignis aufgelöst wird. Dementsprechend definieren diese Ereignisse den Lebenszyklus der Verbindung. Zwischen beiden Ereignissen liegt die Phase, in der die Beziehung aktiv, gültig oder intakt ist. Im Anschluss an das Vergehensereignis gilt die Beziehung als deaktiviert, ungültig oder nicht existent. Dementsprechend ist der Lebenszyklus einer Beziehung durch zwei Phasen gekennzeichnet. Ein erneutes Aktivieren einer einmal deaktivierten Beziehung ist nicht erlaubt. Ein solcher Vorgang kann durch eine neue Beziehung modelliert werden.

Verschiedene Beziehungen können durch ein Ereignis verknüpft sein. Eine solche Verknüpfung wird beispielsweise bei Betrachtung der beiden Eigentumsbeziehungen in Abbildung 6.7 ersichtlich. Die Abbildung zeigt das Eigentumsverhältnis zwischen Person 1, Person 2 und Gegenstand zu den zwei unterschiedlichen Zeitpunkten t_1 und t_2 . Zum Zeitpunkt t_1 besteht eine Eigentumsbeziehung zwischen Person 1 und dem Gegenstand. Durch das Eigentumsübergangereignis vergeht (z. B. durch Verkauf) diese Beziehung und es entsteht gleichzeitig eine neue Eigentumsbeziehung (z. B. durch Kauf) zwischen Person 2 und dem Gegenstand. Zum Zeitpunkt t_2 übernimmt also Person 2 die Rolle Eigner, die Rolle Besitz übernimmt der Gegenstand. Die ursprüngliche Eigentumsbeziehung zwischen

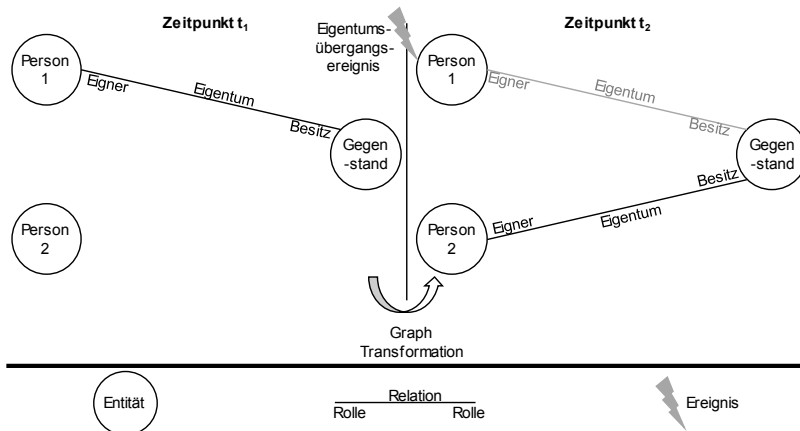


Abbildung 6.7: Beispiel für eine durch ein Ereignis hervorgerufene Graph Transformation

Person 1 und Gegenstand besitzt zum Zeitpunkt t_2 keine Gültigkeit mehr.

Sollte in dem Modell eine etwaige Vorbereitungsphase modelliert werden, kann dies mithilfe eines zweiten Beziehungstyps geschehen. Beispielsweise kann mithilfe einer Interessenbeziehung die dem Eigentumsübergangsereignis vorausgehende Phase modelliert werden. In diesem Fall sind sowohl die beiden Eigentumsbeziehungen als auch die Interessenbeziehung über das Ereignis miteinander verknüpft. Für Beziehungen gilt:

- R1** Eine Beziehung verknüpft genau zwei Rollen.
- R2** Der Lebenszyklus einer Beziehung ist durch genau ein Entstehensereignis und ein Vergehensereignis gekennzeichnet.
- R3** Der Lebenszyklus einer Beziehung besteht aus genau einer aktiven und einer inaktiven Phase.
- R4** Eine einmal deaktivierte Beziehung darf nicht wieder aktiviert werden.

6.4.7 Entität (entity)

Entitäten dienen zur Modellierung der physischen oder gedanklichen Einheiten, deren Zusammenhänge durch die Beziehungen dargestellt werden. Die Abbildung 6.8 zeigt beispielhaft den Lebenszyklus der Entität Gegenstand aus Abbildung 6.7. In der Abbildung sind die unterschiedlichen Rollen, die der Gegenstand während des betrachteten Ausschnitts des Lebenszyklus einnimmt, aufgeführt. Wie auch in Abbildung 6.8 dargestellt, übernimmt der Gegenstand zunächst die Rolle Besitz von Person 1 und wechselt durch das Ereignis Eigentumsübergang in die Rolle Besitz von Person 2. Im Rahmen dieser Rolle wird der Gegenstand zusätzlich zunächst Ersatzteil und anschließend Teil von Anlage X. Hier ist zu sehen, dass der Gegenstand gleichzeitig die Rolle Besitz und Ersatzteil bzw. Teil-von übernehmen kann. Dagegen schließen sich zum Beispiel die Rollen Ersatzteil sowie Teil-von gegenseitig aus. Anschließend übernimmt der Gegenstand durch ein erneutes Eigentumsübergangsereignis die Rolle Besitz von Person 3. Für Entitäten gelten daher folgende Regeln:

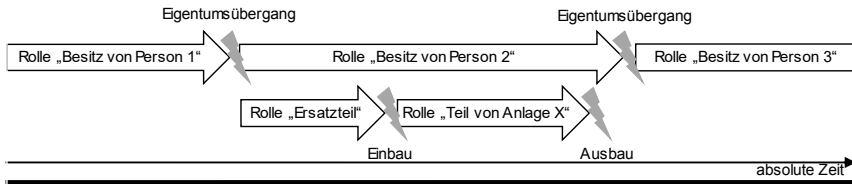


Abbildung 6.8: Lebenszyklus der Entität Gegenstand auf Abbildung 6.7

- R1** Eine Entität besitzt einen definierten Lebenszyklus, der durch das Einnehmen von Rollen in Beziehungen beschrieben wird.
- R2** Eine Entität kann, wenn die Anforderungen der Rolle dies zulassen, in beliebig vielen Beziehungen gleichzeitig Rollen übernehmen.

6.4.8 Rollen (role)

Entitäten übernehmen in einer Beziehung bestimmte Rollen. Diese stellen Anforderungen, die von Entitäten erfüllt werden müssen, wenn diese die entsprechenden Rollen einnehmen sollen. Diese Anforderungen legen einerseits den Entitätstypen fest und können andererseits die weiteren Rollen der Entität einschränken. Beispielsweise kann eine Entität des Typs Person eine Entität des Typs Gegenstand besitzen. Ein Gegenstand kann aber keine Person besitzen. Wird ein Gegenstand von einer Person besessen, kann er nicht Teil des Besitzes einer weiteren Person sein. Dagegen kann eine Person mehrere Gegenstände besitzen.

- R1** Rollen beschreiben Anforderungen, die von den Entitäten erfüllt werden müssen, damit diese die Rollen einnehmen und so Beziehungen eingehen können.
- R2** Rollen werden im Zusammenhang mit Beziehungen definiert.

6.4.9 Anwendung des Metametamodells

Bei der Anwendung des vorgestellten Metametamodells ist es hilfreich, zwischen einer Änderung aufgrund eines Ereignisses und einer Änderung aufgrund einer weiteren kontinuierlich getroffenen Eigenschaftsaussage zu unterscheiden. Grafisch ist dies in Abbildung 6.9 dargestellt. Auf der linken Seite der Abbildung ist eine Änderung durch ein Ereignis abgebildet. Hier wechselt der Nachbarschaftsgraph aufgrund eines Ereignisses durch eine Graphtransformation vom Zustand T_1 in den Zustand T_2 . Im Rahmen eines Zustandes können zusätzlich kontinuierlich Eigenschaftsaussagen zu den Eigenschaften der einzelnen Subjekte des Graphen getroffen werden. Dies ist in der Abbildung auf der rechten Seite dargestellt. So werden im Zustand T_1 für die Beziehungen zwischen A und B sowie B und D als auch für die Entität C Eigenschaftsaussagen archiviert. Dementsprechend ist die Unterscheidung zwischen kontinuierlichen Eigenschaftsaussagen und Ereignisse von wesentlicher Bedeutung. Der gewählte Modellierungsansatz ist eine Designentscheidung, die individuell getroffen werden muss.

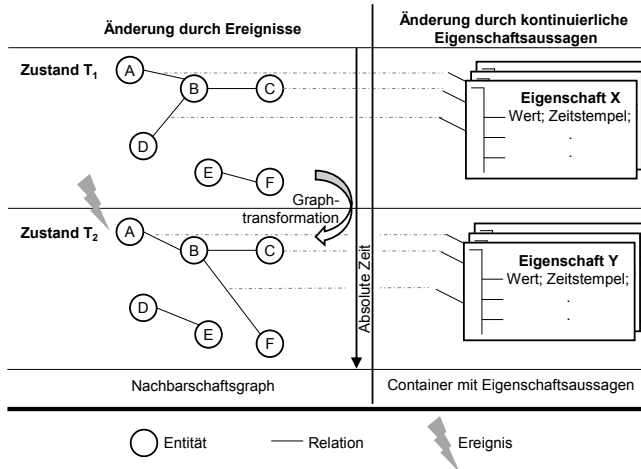


Abbildung 6.9: Unterscheidung zwischen Änderungen einerseits durch Ereignisse und andererseits durch kontinuierliche Eigenschaftsaussagen

6.5 Aspektmetamodelle

Wie in der Modellhierarchie in Abbildung 6.3 dargestellt ist, werden auf Basis des definierten Metametamodells verschiedene Aspektmetamodelle entwickelt. Diese dienen der konkreten Abbildung unterschiedlicher Anlagenaspekte im eigentlichen Instanzmodell. Um eine Erweiterbarkeit sicherzustellen, wird hier ein modularer Ansatz gewählt, der es erlaubt, einzelne Aspekte teilweise unabhängig voneinander zu betrachten. Zur Sicherstellung einer ausreichenden Ausdrucksstärke dienen die in Kapitel 4 vorgestellten Informationsmodelle als Grundlage für die Definition der Aspektmetamodelle. Gemäß dem in Abschnitt 1.2 vorgestellten Vorgehen handelt es sich bei jedem Aspektmetamodell um einen Begriffscluster.

Jedes einzelne Aspektmetamodell besteht aus einer bestimmten Kombination von Ereignis-, Beziehungs- und Entitätsklassen. Damit die einzelnen Aspekte später zu einem gemeinsamen Instanzmodell zusammengesetzt werden können, gibt es zwischen den einzelnen Aspektmetamodellen Überschneidungen in den Entitätsklassen.

Zur Auswahl der unterschiedlichen Aspekte ist es hilfreich, unterschiedliche Ereignisse im Umfeld einer prozesstechnischen Anlage zu betrachten. Im Kapitel 4.3.2 und in Abbildung 4.14 wird eine Übersicht über die unterschiedlichen Ereignisse gegeben.

Nachfolgend werden die einzelnen Aspektmetamodelle vorgestellt. Der Aufbau der Beschreibung und der Abbildungen orientiert sich an der zuvor vorgestellten Modellhierarchie. Dies bedeutet, dass in den Abbildungen von oben nach unten der Abstraktionsgrad abnimmt. Ganz oben befinden sich die Klassen des Metametamodells während sich darunter die gemeinsame Oberklassen des Metamodells befinden.

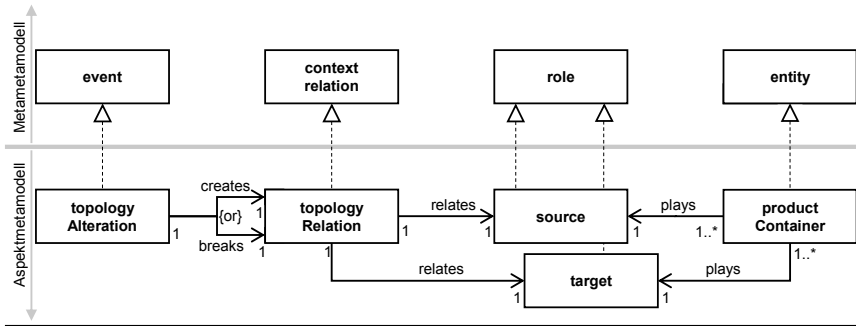


Abbildung 6.10: Klassendiagramm des Aspektmetamodells Topologie

6.5.1 Aspektmetamodell Topologie

Mithilfe des Metamodells Topologie kann die physikalische Struktur einer Anlage in Form eines semantischen Netzes beschrieben werden. Abbildung 6.10 gibt einen Überblick über die wichtigsten Begriffe des Metamodells. Diese sind Produktcontainer (productContainer), Produktverbindungen (productConnection) und Topologiemodifikation (topologyModification), die im Folgenden näher erläutert werden.

Zum einfacheren Verständnis soll zunächst das Beispiel in Abbildung 6.11 betrachtet werden. Hier ist die beispielhafte Modellierung einer Anlage auf Basis des vorgestellten Metamodells dargestellt. Die Anlage besteht aus dem Behälter B1, der Pumpe N18, dem Ventil Y16 sowie drei Rohrleitungen, die diese verbinden. Modelliert wird diese Anlage mithilfe der entsprechenden Produktcontainer vessel, pump, valve sowie pipe. Die verbauten Sensoren finden in diesem Aspektmetamodell keine Berücksichtigung. Für sie wird in Abschnitt 6.5.2 ein eigenes Aspektmetamodell beschrieben. Verknüpft sind die Produktcontainer durch Beziehungen vom Typ Primärproduktverbindung. Außerdem ist die Energieversorgung der Pumpe und des Ventils modelliert. Die Pumpe wird über ein Stromversorgungsnetz und das Ventil über eine Druckluftversorgung gespeist. Die Verknüpfung erfolgt hier über eine Sekundärproduktverbindung. Außerdem ist angegeben welche Rollen die Entitäten in der jeweiligen Beziehung einnehmen.

Entität Produktcontainer

Mit Instanzen des Entitätstypen Produktcontainer (productContainer) wird die Modellierung von Gegenständen, die der Handhabung von Produkten dienen, ermöglicht. Hier wird zwischen aktiven und passiven Produktcontainern unterschieden. Mögliche passive Produktcontainer sind beispielsweise Rohrleitungen, Behälter, Reaktoren sowie Schnittstellen zur Umwelt. Aktive Produktcontainer können Ereignisse auslösen und stellen Akteure, wie beispielsweise Ventile oder Pumpen, dar. Eine Übersicht über die wichtigsten Produktcontainertypen ist in Abbildung 6.12 gegeben. Die dort gezeigte Auflistung basiert auf der PandIX (vgl. Abschnitt 4.2.1) Standard Prozessanlagenelemente-Bibliothek (PandIX-PPE-BasicRoleType-Lib), ist aber nicht abgeschlossen und kann bei Bedarf zum Beispiel um die von DEXPI definierten Typen erweitert werden.

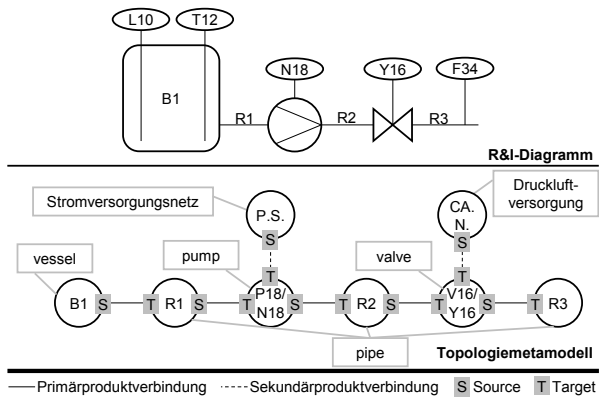


Abbildung 6.11: Beispielhafte Modellierung einer (Teil)Anlage auf Basis des Topologiemeta-modells

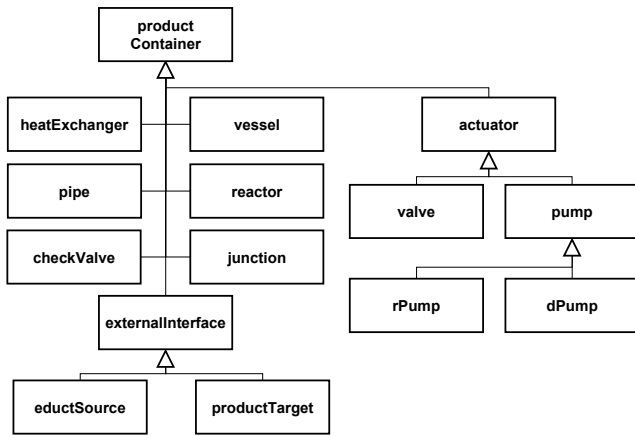


Abbildung 6.12: Unterschiedliche Produktcontainertypen

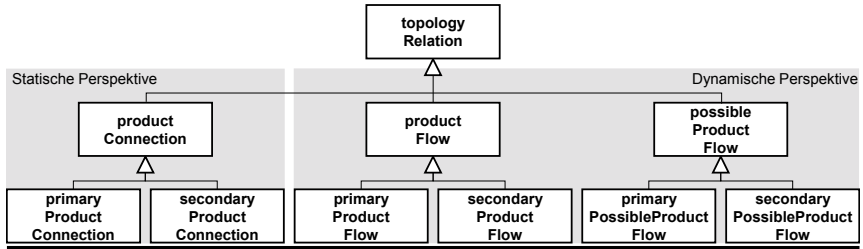


Abbildung 6.13: Unterschiedliche Topologiebeziehungstypen

Topologiebeziehung

Die Zusammenhänge zwischen den einzelnen Produktcontainerentitäten werden mithilfe der Topologiebeziehung modelliert. Dabei handelt es sich um eine gerichtete Beziehung, in der jeweils ein Produktcontainer die Rolle Quelle (source) und die Rolle Senke (target) einnimmt. Dabei beschreiben Quelle und Senke die Richtung des intendierten (d. h. geplanten bzw. typischen) Produktflusses. Ein Produktcontainer kann gleichzeitig in verschiedenen Topologiebeziehungen sowohl die Rollen Quelle als auch Senke einnehmen. Zur Unterscheidung zwischen unterschiedlichen Topologiebeziehungen werden die in Abbildung 6.13 dargestellten Beziehungstypen eingeführt.

Mithilfe der Beziehungstypen wird eine Unterscheidung zwischen einer statischen und einer dynamischen Perspektive ermöglicht. Die statische Perspektive beschreibt alle existierenden Beziehungen zwischen den Produktcontainern. Sind zum Beispiel, wie in Abbildung 6.11 dargestellt, Behälter, Pumpe und Ventil über Rohrleitungen miteinander verbunden, so werden diese Beziehungen mithilfe des Beziehungstyps `productConnection` modelliert. Auf diese Art und Weise kann der mögliche Produktfluss beschrieben werden. Soll dagegen der tatsächliche Produktfluss modelliert werden, geschieht dies mithilfe des Typs `productFlow`. Dementsprechend wird mithilfe der `productFlow`-Beziehung der wahre Produktfluss, z. B. in Abhängigkeit der aktuellen Ventilstellung, beschrieben. Für Fälle, in denen unklar ist, ob ein wahrer Produktfluss existiert oder nicht, wird der Beziehungstyp `possibleProductFlow` eingeführt. Damit kann ein möglicher Produktfluss beschrieben werden. Ein Beispiel für einen möglichen Produktfluss ist der Produktstrom durch eine deaktivierte Strömungspumpe, da diese im Gegensatz zu einer Verdrängerpumpe den Flussweg des Produkts nicht sperrt. Eine beispielhafte Modellierung des Produktflusses unter Berücksichtigung des Pumpentyps ist in Abbildung 6.14 gegeben. Außerdem ist in der Abbildung zu sehen, dass es sich bei den (possible)`productFlow`-Beziehungen immer um ein Subset der `productConnection`-Beziehungen handelt.

Für alle drei Topologiebeziehungstypen wird außerdem eine Unterscheidung zwischen einer Primär- und einer Sekundärproduktverbindung eingeführt. Primärprodukte sind solche, für die die entsprechende Anlage entwickelt und gebaut worden ist. In einer Raffinerie zum Beispiel sind Rohöl, Benzin usw. Primärprodukte, während es sich bei (Hilfs-)Energien wie Druckluft, Dampf oder Strom um Sekundärprodukte handelt. Dementsprechend können Produktcontainer sowohl durch Primär- als auch durch Sekundärproduktverbindungen verknüpft sein. Je nach Anlagentyp kann eine weitere Klassi-

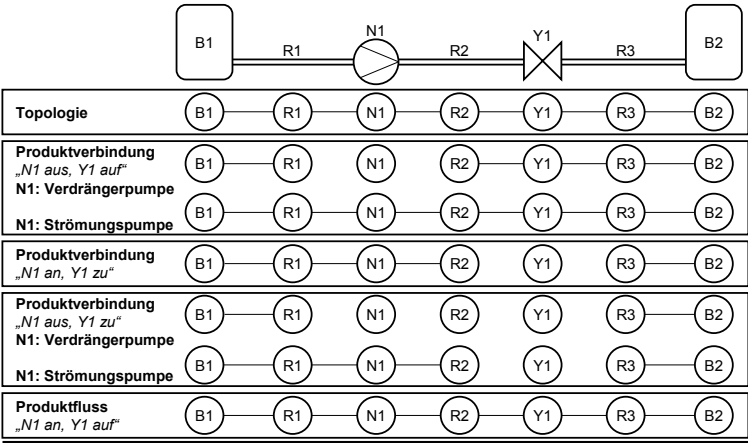


Abbildung 6.14: Beispielhafte Modellierung des Produktflusses zwischen zwei Behältern

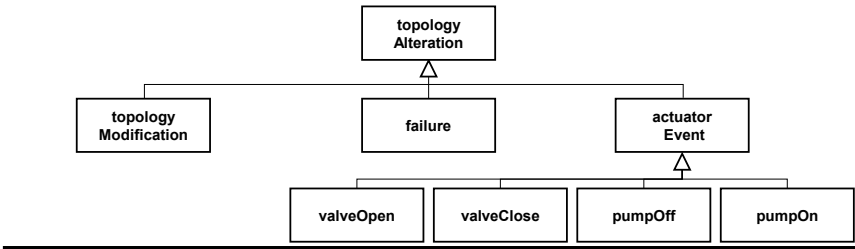


Abbildung 6.15: Verschiedene Ereignistypen

fikation, beispielsweise entsprechend des Produkttyps oder der Druckstufe, an dieser Stelle angebracht sein.

Ereignis Topologieveränderungen

In Abbildung 6.15 sind unterschiedliche Subtypen des Ereignisses Topologieveränderung (topologyAlteration) dargestellt. Allen dargestellten Ereignistypen ist gemein, dass sie die Topologie der Anlage beeinflussen. Dabei muss zwischen Änderungen an der statischen und der dynamischen Anlagentopologie unterschieden werden. Wird die modellierte Anlage umgebaut, hat dies in der Regel Einfluss sowohl auf die statische als auch die dynamische Topologie der Anlage. Das bedeutet, dass Verknüpfungen zwischen Anlagenelementen aufgelöst und/oder hinzugefügt werden. Diese Geschehnisse können in dem Metamodell mithilfe des Ereignisses Topologiemodifikation (topologyModification) abgebildet werden. Ähnliche Überlegungen gelten auch für das Ereignis Ausfall (failure), das den Ausfall einer Komponente beschreibt.

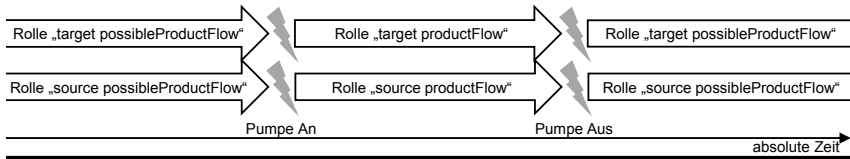


Abbildung 6.16: Ausschnitt aus dem Lebenszyklus einer Pumpe

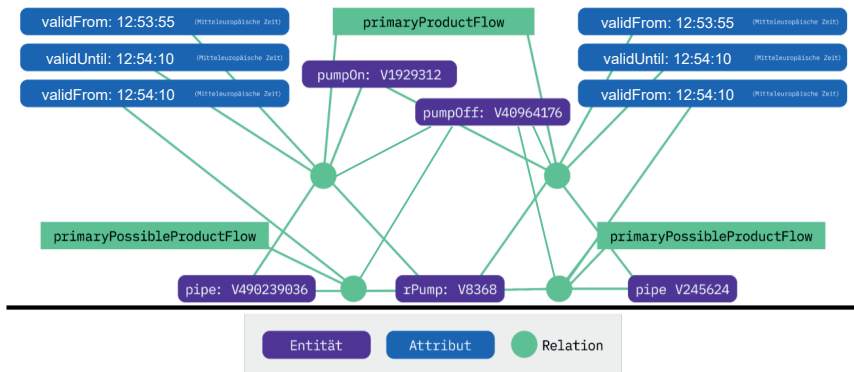


Abbildung 6.17: Ausschnitt aus der Modellierung des Produktflusses mithilfe von Grakn - Screenshot aus der Grakn Workbase

Dagegen betreffen actuatorEvents ausschließlich die dynamische Struktur der Anlage. Diese Ereignistypen beschreiben das Starten bzw. Stoppen einer Pumpe sowie das Öffnen bzw. Schließen eines Ventils. Diese Ereignisse haben jeweils das Entstehen oder Vergehen der zugehörigen Beziehungen zur Folge. Exemplarisch ist dies für eine Strömungspumpe in Abbildung 6.16 dargestellt. Zu sehen ist ein reduzierter Ausschnitt aus dem Lebenszyklus, in dem die Pumpe einmal ein- und einmal ausgeschaltet wird und daraufhin in unterschiedlichen Beziehungen Rollen einnimmt. Gegenstück in den Beziehungen ist die Quell- bzw. Zielrohrleitung der Pumpe.

Abbildung 6.17 zeigt einen Screenshot aus der Grakn Workbase, der einen Ausschnitt der Modellierung eines Produktflusses zeigt. Einerseits sind die am Produktfluss beteiligten Rohrleitungen sowie die zugehörige Pumpe zu sehen. Andererseits sind die zum Produktfluss gehörenden Ereignisse pumpOn und pumpOff dargestellt. Die Abhängigkeiten zwischen den einzelnen Entitäten sind mithilfe von Beziehungen modelliert. Anhand der Attribute validFrom und validUntil kann der zeitliche Ablauf des Produktflusses rekonstruiert werden. Die Pumpe wurde um 12:53:55 Uhr an- und um 12:54:10 Uhr abgeschaltet.

6.5.2 Aspektmetamodell Messen

Zur Modellierung von Messvorgängen wird das Aspektmetamodell Messen eingeführt. Es erlaubt eine Erweiterung des Topologiemetamodells um Sensoren. Kern des Aspektmeta-

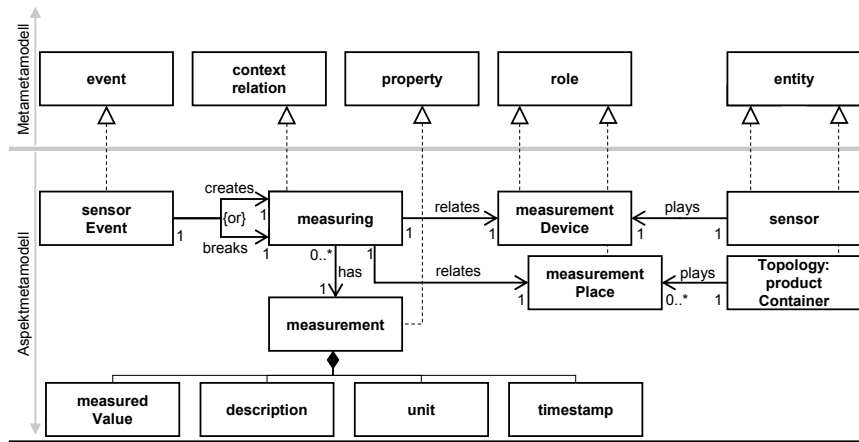


Abbildung 6.18: Klassendiagramm des Aspektmetamodells Messen

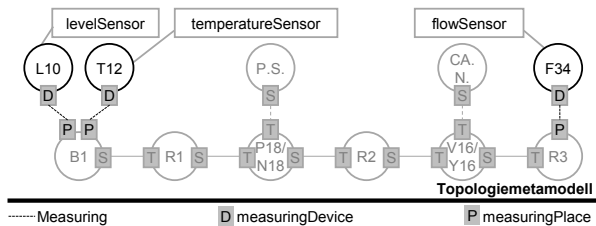


Abbildung 6.19: Modellierung der Sensoren der in Abbildung 6.11 dargestellten Anlage

modells ist die Beziehung Messen (*measuring*). Sie ist mit dem dazugehörigen Eventtyp und den Rollenklassen in Abbildung 6.18 dargestellt.

Außerdem ist in Abbildung 6.19 beispielhaft die Modellierung der Sensoren der in Abbildung 6.11 gezeigten (Teil)Anlage dargestellt.

Beziehung Messen

Die Beziehungsklasse Messen beschreibt den Zusammenhang zwischen einer Messstelle (*measurementPlace*) und einem Messgerät (*measurementDevice*). Dazu stellt die Beziehungsklasse eben diese Rollen zur Verfügung. Die Rolle Messstelle wird von einem im Aspektmetamodell Topologie beschriebenen Produktcontainer eingenommen. Damit kann eine Messung zum Beispiel in einer Rohrleitung, in einem Behälter aber auch in einer Pumpe oder einem Reaktor erfolgen. Die Rolle Messgerät wird von einer Entität des Typs *Sensor* eingenommen.

Sollen die Messwerte des Sensors in das Modell integriert werden, erfolgt dies entsprechend der Abbildung 6.9. Dabei werden die gemessenen Werte als Eigenschaftsaussagen

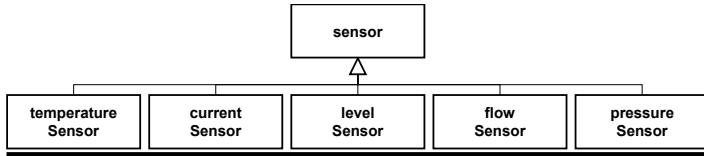


Abbildung 6.20: Verschiedenen Sensortypen

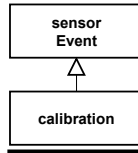


Abbildung 6.21: Kalibrierung als möglicher sensorEventtyp

der Beziehung zwischen Produktcontainer und Sensor zugeordnet.

Eigenschaft Messwert

Zur Integration von Messwerten werden Eigenschaftswertaussagen integriert. Diese werden der Beziehungsinstanz measuring zugeordnet und beschreiben so die in der Beziehung zwischen Messgerät und Messstelle aufgenommenen Messwerte. Sollte sich zum Beispiel durch ein Sensorereignis, wie beispielsweise eine Sensorkalibrierung, die Beziehung zwischen Messgerät und Messstelle ändern, wird dies auf diese Art und Weise in dem Modell ersichtlich.

Entität Sensor

Diese Entität beschreibt die Funktionalität des Sensors. Für das Aspektmetamodell werden, wie in Abbildung 6.20 dargestellt, zunächst die Standardsensoren für die Messung der Temperatur, des Füllstands, des Durchflusses, des Drucks sowie des Stroms berücksichtigt. Bei Bedarf können an dieser Stelle weitere Sensortypen ergänzt werden.

Sensorereignis

Das Sensorereignis (sensorEvent) beschreibt Geschehnisse in der Verknüpfung zwischen Sensor und Messstelle. Wie in Abbildung 6.21 dargestellt, kann ein möglicher sensorEvent-Typ die Kalibrierung (calibration) eines Sensors sein.

6.5.3 Aspektmetamodell Deployment

Mithilfe des Aspektmetamodells Deployment können die Abhängigkeiten zwischen Softwarekomponenten und den ausführenden Hardwarekomponenten modelliert werden und so von der Steuerungsstruktur unabhängige Beziehungen zwischen einzelnen Softwarekomponenten aufgedeckt werden. Das Aspektmetamodell ist in Abbildung 6.22 dargestellt und

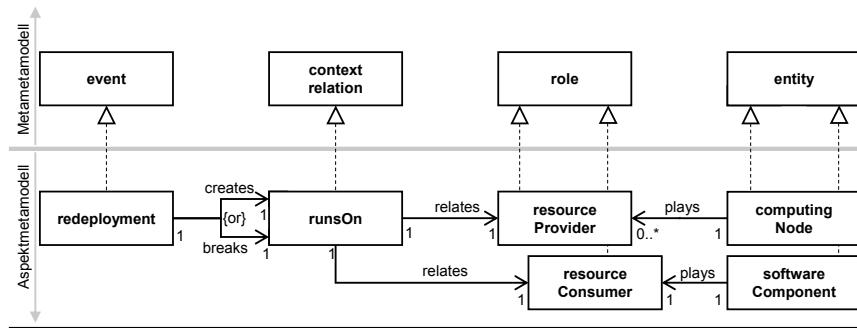


Abbildung 6.22: Klassendiagramm des Aspektmetamodells Deployment

besteht aus den beiden Entitätsklassen `computingNode` sowie `softwareComponent`, die über Beziehungen des Typs `runsOn` miteinander verknüpft sind.

Entität `computingNode`

Eine Entität der Klasse `computingNode` repräsentiert in dem Nachbarschaftsgraphen eine Stelle, die Rechenkapazitäten bereitstellt. Typischerweise sind dies speicherprogrammierbare Steuerungen (SPS) oder Industrie-PCs. Diese werden in aller Regel zur Ausführung der Steuerungslogik verwendet und können auf Grund der begrenzten Kapazitäten (z. B. CPU-Zeit, Netzwerkbandbreite oder Speicher) zu Querabhängigkeiten zwischen den einzelnen Komponenten der Steuerungslogik auf einem Rechenknoten führen.

Entität `softwareComponent`

Die Entitätsklasse `softwareComponent` dient der Repräsentation der einzelnen Komponenten der Steuerungslogik, die im Aspektmetamodell Steuerungsstruktur näher spezifiziert werden. Sie verbrauchen die von den `computingNodes` bereitgestellten Ressourcen.

Beziehung `runsOn`

Durch Beziehungen der Klasse `runsOn` wird der Zusammenhang zwischen den Entitäten der Klassen `softwareComponent` und `computingNode` modelliert. Dabei nimmt im Rahmen einer solchen Beziehung die `computingNode` die Rolle `resourceProvider` und die `softwareComponent` die Rolle `resourceConsumer` ein.

Ereignis `redeployment`

Wird eine `softwareComponent` auf einer `computingNode` instanziiert oder auf eine andere `computingNode` verschoben, so wird dies als (Re)deployment bezeichnet. Dieser Vorgang wird im Rahmen des Nachbarschaftsgraphen als Ereignis vom Typ `redeployment` modelliert.

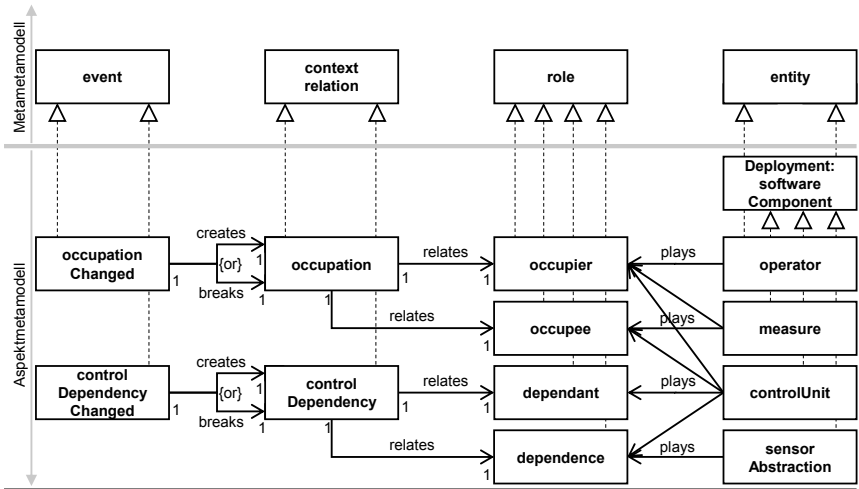


Abbildung 6.23: Klassendiagramm des Aspektmetamodells Steuerungsstruktur

6.5.4 Aspektmetamodell Steuerungsstruktur

Neben der physischen Anlagentopologie haben auch die Abhängigkeiten zwischen den einzelnen Steuerungskomponenten einen wesentlichen Einfluss auf die Prozessdaten einer Anlage. Zur Modellierung dieser Abhängigkeiten wird hier das Aspektmetamodell Steuerungsstruktur definiert. Entscheidend zur Modellierung dieser Struktur ist die Wahl des richtigen Abstraktionsgrads bei der Modellierung der Steuerungsstruktur. Beispielsweise ist eine Abbildung jedes einzelnen (IEC61131-)Funktionsbausteins nicht zielführend. Dieser Ansatz würde die Datenbank unnötig aufblähen und die Aufdeckung relevanter Abhängigkeiten erschweren. Ein geeigneter Ansatz stellt das im Kapitel 3.1.3 vorgestellte Maßnahmen- und Betriebsmittelmodell dar. Es besteht einerseits aus Abstraktionskomponenten für bestimmte Hard- oder Softwarefunktionalitäten (Betriebsmittel) und andererseits aus Ablaufbeschreibungen (Maßnahmen), wobei die Ablaufbeschreibungen eine Orchestrierung der Betriebsmittel vornehmen. Abbildung 6.23 zeigt die verschiedenen Klassen zur Modellierung der Steuerungsstruktur als semantisches Netz, welche in den nachfolgenden Abschnitten näher erläutert werden. In der Abbildung ist außerdem zu erkennen, dass das Aspektmetamodell Steuerungsstruktur auf der Klasse softwareComponent des Aspektmetamodells Deployment aufbaut.

Da dieses Aspektmetamodell auf Basis einer Steuerungsarchitektur gemäß dem Maßnahmen- und Betriebsmittelmodell definiert wird, besteht eine große Abhängigkeit zwischen der Steuerungsarchitektur und dem Metamodell. Sollte die Steuerungsstruktur auf Basis einer anderen Architektur entworfen worden sein, kann dies Veränderungen am definierten Metamodell zur Folge haben.

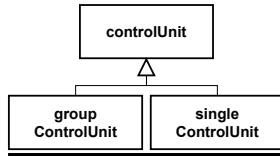


Abbildung 6.24: Unterschiedliche ControlUnit-Typen

Betriebsmittel

Die Betriebsmittel werden in dem Aspektmetamodell mithilfe der Entitätsklassen Steuereinheit (controlUnit), Sensorabstraktion (sensorAbstraction) und der Beziehungsklasse Steuerungsabhängigkeit (controlDependency) modelliert.

Entität Steuereinheit

Eine Entität der Klasse Steuerungseinheit (controlUnit) repräsentiert eine Steuerungseinheit aus dem Prozessleitsystem innerhalb des Nachbarschaftsgraphen. Hier wird zwischen Einzel- und Gruppensteuereinheiten unterschieden (vgl. Abbildung 6.24). Eine Steuereinheit kann über eine Beziehung der Klasse Steuerungsabhängigkeit (controlDependency) mit einem Sensor (sensorAbstraction) oder einer weiteren Steuereinheit verknüpft sein. In dieser Beziehung übernimmt die Steuereinheit die Rolle dependant und der Sensor bzw. die zweite Steuereinheit die Rolle dependence. Eine Abhängigkeit zwischen zwei Steuereinheiten existiert beispielsweise, wenn eine Pumpe nur bei geöffnetem Ventil aktiviert werden darf. Außerdem kann eine Einzelsteuereinheit von einer Gruppensteuereinheit belegt werden. Dieser Vorgang wird mit einer Beziehung der Klasse Belegung (occupation) modelliert.

Entität Sensorabstraktion

Die Entitätsklasse Sensorabstraktion (sensorAbstraction) repräsentiert einen Sensor innerhalb der Steuerungslogik. Dort ist sie für die Umrechnung des Messsignals in einen physikalischen Wert und ggf. für die Generierung von Verriegelungssignalen beim Erreichen bestimmter Grenzwerte zuständig. Über beide Signaltypen kann eine Entität der Klasse Sensorabstraktion mit Steuereinheiten verknüpft sein. Diese Abhängigkeiten werden in dem Nachbarschaftsgraphen über die Beziehungsklasse Steuerungsabhängigkeit (controlDependency) modelliert. Im Rahmen dieser Beziehung übernimmt die Entität der Klasse Sensorabstraktion immer die Rolle des measurementInstrument.

Beziehung Steuerungsabhängigkeit

Beziehungen der Klasse Steuerungsabhängigkeit dienen zur Modellierung der Verknüpfungen zwischen den Entitäten der Klasse Sensorabstraktion und der Klasse Steuereinheit. Diese Beziehungen können aufgrund von zwei unterschiedlichen Signaltypen auftreten. Einerseits stellt die Sensorabstraktion den physikalischen Messwert des Sensors zur Verfügung. Dieser kann von den Steuereinheiten ausgewertet werden. Andererseits kann die Sensorabstraktion auf Basis von Grenzwerten unterschiedliche Verriegelungssignale gene-

rieren. Auch diese können von den Steuereinheiten ausgewertet werden. Außerdem können zwischen verschiedenen Steuereinheiten Abhängigkeiten bestehen.

Ereignis `controlDependencyChanged`

Ereignisse der Klasse `controlDependencyChanged` dienen zur Modellierung von Veränderungen in der Steuerungsstruktur. Deshalb haben die Ereignisse von dem Typ `controlDependencyChanged` immer mindestens entweder das Vergehen oder das Entstehen einer Beziehung der Klasse Steuerungsabhängigkeit zur Folge. Ereignisse dieser Klasse können beispielsweise durch Engineeringvorgänge oder Rekonfigurationen von modularen Anlagen auftreten.

Entität `Maßnahme`

Eine Entität der Klasse `Maßnahme` (`measure`) dient zur Repräsentation von Maßnahmen im Nachbarschaftsgraph. Maßnahmen können untergeordnete Maßnahmen und Steuereinheiten belegen. Dieser Zusammenhang kann mithilfe der Beziehung `Belegung` (`occupation`) modelliert werden. In dieser Beziehung übernimmt die übergeordnete `Maßnahme` die Rolle `occupier` und die untergeordnete `Maßnahme` oder `Steuereinheit` die Rolle `occupee`.

Entität `Operator`

Entitäten der Klasse `Operator` repräsentieren die menschlichen Bediener, welche unterschiedliche Aktionen innerhalb der Anlage ausführen können. Grundvoraussetzung dafür ist in dem `Maßnahmen-Betriebsmittelmodell` die `Belegung` der entsprechenden Steuereinheiten. Deshalb können die Eingriffe eines Operators mithilfe der Beziehung `Belegung` modelliert werden. Innerhalb dieser Beziehung übernimmt eine `Operatorentität` immer die Rolle `occupier`.

Beziehung `Belegung`

Mithilfe der Beziehungsklasse `Belegung` werden die Belegungsvorgänge innerhalb des Nachbarschaftsgraphen modelliert. Dazu muss eine Entität des Typs `Operator`, `Maßnahme` oder `Gruppensteuereinheit` die Rolle `occupier` und eine Entität der Klasse `Maßnahme`, `Gruppen-` oder `Einzelsteuereinheit` die Rolle `occupee` übernehmen. Dabei kann eine Entität die Rolle `occupee` nur genau einmal übernehmen. Eine Entität kann also zu einem bestimmten Zeitpunkt nur einmal belegt werden.

Ereignis `Belegungsänderung`

Typischerweise ändert sich der Belegungszustand zum Beispiel in Abhängigkeit der Betriebsphase regelmäßig. Solche Änderungen werden in dem Nachbarschaftsgraphen mithilfe der Ereignisklasse `Belegungsänderung` (`occupationChanged`) modelliert. Ein Ereignis des Typs `Belegungsänderung` hat das Vergehen oder Entstehen einer Beziehung der Klasse `Belegung` zur Folge. Wie in Abbildung 6.25 dargestellt, ist eine `Belegungsänderung` entweder vom Typ `Belegung` (`occupied`) oder vom Typ `Freigabe` (`freed`). Je nach Komplexität des Belegungsmechanismus sind auch noch weitere Ereignisse (z. B. manueller Eingriff) denkbar.

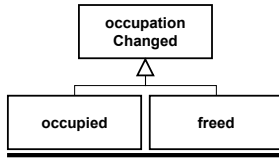


Abbildung 6.25: Unterschiedliche Typen der Belegungsänderung

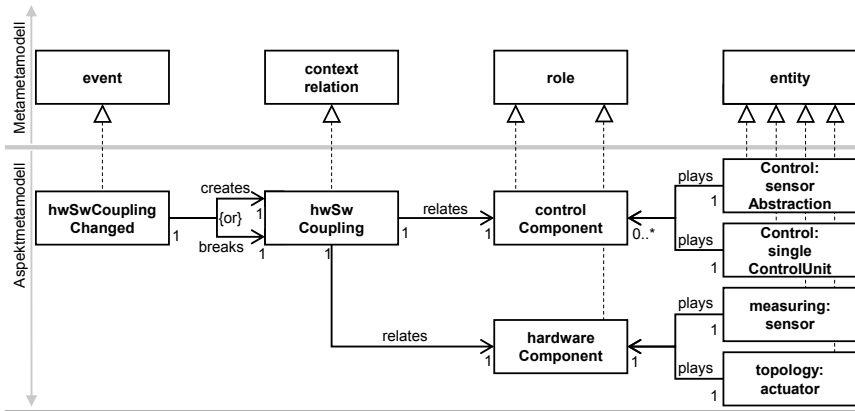


Abbildung 6.26: Klassendiagramm des Aspektmetamodells Hardware-Software-Kopplung

6.5.5 Aspektmetamodell Hardware-Software-Kopplung

Zur Beschreibung der Verknüpfung zwischen den Entitäten des Topologie-Aspektmetamodells und den zugehörigen Steuerungskomponenten in der Automatisierungssoftware wird das Aspektmetamodell Hardware-Software-Kopplung definiert. Das Modell ist in Abbildung 6.26 in Form eines Klassendiagramms dargestellt.

Kern des Modells ist die Beziehung `hwSwCoupling`, die die beiden Rollen `controlComponent` und `hardwareComponent` in Beziehung setzt. Dabei wird die Rolle `controlComponent` entweder von einer Entität des Typs `sensorAbstraction` oder des Typs `singleControlUnit` ausgefüllt. Beide Entitätstypen stammen aus dem Aspektmetamodell Steuerungsstruktur. Die Rolle `hardwareComponent` dementsprechend entweder von einer Entität des Typs `Sensor` aus dem Aspektmetamodell Messen oder von einer Entität des Typs `Aktuator` aus dem Topologie-Aspektmetamodell übernommen.

Der Lebenszyklus der Beziehung wird durch Ereignisse des Typs `hwSwCouplingChanged` beeinflusst. Typischerweise werden solche Ereignisse durch Engineeringvorgänge ausgelöst.

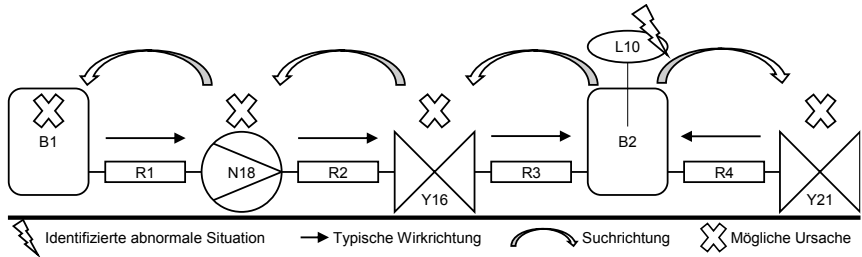


Abbildung 6.27: Beispielhafte Darstellung des Vorgehens zur Auswertung der typischen Wirkrichtung zum Auffinden einer Ursache für eine identifizierte abnormale Situation

6.6 Mögliche Erweiterungen zur Vereinfachung des Datenzugriffs

Zur Vereinfachung der Suche bzw. des Datenzugriffs innerhalb des Nachbarschaftsgraphen bestehen verschiedene Alternativen. Nachfolgend werden zwei mögliche Erweiterungen diskutiert.

6.6.1 Definition von Perspektiven

Eine Möglichkeit zur Erweiterung stellt die Definition von unterschiedlichen Sichten dar. Da sowohl alle Beziehungen als auch alle Entitäten und alle Ereignisse klassifiziert sind, können verschiedene Subsets dieser Klassen zu Sichten oder Perspektiven zusammengefasst werden.

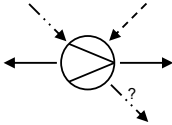
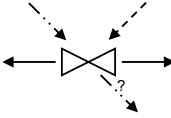
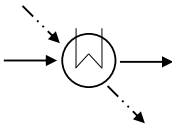
Zwei mögliche Perspektiven wurden zusammen mit dem Aspektmetamodell Topologie vorgestellt. Das Topologiemetamodell erlaubt einerseits die Betrachtung der statischen und andererseits die Betrachtung der dynamischen Struktur der Anlage. Darauf aufbauend können eine dynamische und eine statische Anlagenstrukturperspektive definiert werden.

6.6.2 Analyse der Wirkrichtung der Beziehungen

Eine weitere Möglichkeit zur Erweiterung stellt die Analyse der Wirkrichtungen der in den Aspektmetamodellen definierten Beziehungen dar. Wenn diese eindeutig definiert werden, können sie durch Inferenzen innerhalb des Instanzmodells entdeckt und zur Verfeinerung der Suche ausgenutzt werden. Bei der Definition der Wirkrichtungen ist auf eine möglichst eindeutige Einschränkung dieser zu achten, sodass sie typische oder mögliche Wirkzusammenhänge aufzeigen. Werden diese Wirkrichtungen anschließend im Instanzmodell durch Schlussfolgerungen aufgedeckt, ist es Aufgabe des Nutzers, mithilfe seines Prozessverständnisses die möglichen von den wahren Wirkzusammenhängen zu unterscheiden.

Wird das Aspektmetamodell Topologie betrachtet, so ist grundsätzlich eine Wirkung entlang des Produktflusses zu vermuten. In Tabelle 6.1 ist dies exemplarisch für die Entitätsklassen Pumpe, Ventil und Wärmetauscher aufgeführt.

Tabelle 6.1: Mögliche Wirkrichtungen im Aspektmetamodell Topologie

Entitätsklassen	Erläuterung	
Pumpe	Eine Pumpe wirkt in beiden Richtungen auf den Primärproduktfluss. Das Steuerungssignal und die Energieversorgung bzw. der Sekundärproduktfluss wirken auf die Pumpe. Je nach Energietyp kann die Pumpe selbst auch Rückwirkungen auf die Energieversorgung haben.	
Ventil	Ein Ventil besitzt dieselben Wirkrichtungen wie eine Pumpe. Es wirkt in beiden Richtungen auf den Primärproduktfluss. Das Steuerungssignal und die Energieversorgung bzw. der Sekundärproduktfluss wirken auf das Ventil. Außerdem kann auch das Ventil selbst Rückwirkungen auf den Sekundärproduktfluss haben.	
Wärmetauscher	Sowohl der zufließende Primärproduktfluss als auch der zufließende Sekundärproduktfluss wirken auf den Wärmetauscher. Dieser wirkt auf den ausfließenden Primär- und Sekundärproduktfluss.	

—→ Primärproduktfluss - - - → Softwaresignal - · - · → Sekundärproduktfluss

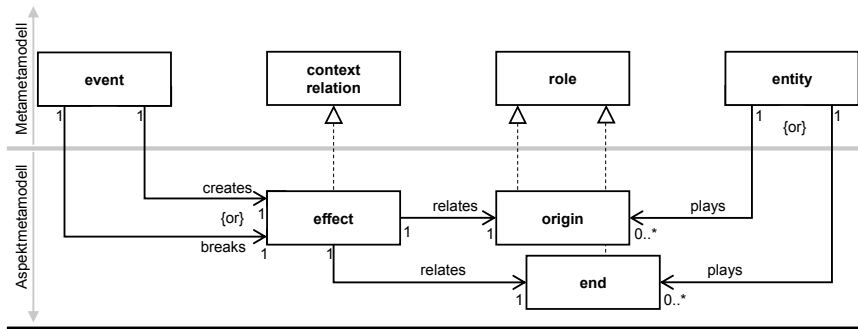


Abbildung 6.28: Klassendiagramm des Aspektmetamodells Wirkrichtung

Abbildung 6.27 zeigt, wie die zuvor definierten typischen Wirkrichtungen zur Suche nach möglichen Ursachen ausgenutzt werden können. In dem Beispiel wird eine Abnormität des Füllstands in Behälter B3, der mit dem Sensor L10 gemessen wird, identifiziert. Ein mögliches Beispiel für eine solche Abnormität könnte das Erreichen des oberen oder unteren Grenzwerts sein. Für die Suche nach der Ursache muss untersucht werden, ob B2 das Ende einer Wirkrichtung ist. In dem gezeigten Beispiel wirken die Ventile Y16 und Y21 auf B2. Dementsprechend stellen sowohl Y16 als auch Y21 mögliche Ursachen für die Abnormität des Füllstands in B1 dar. Anschließend kann ausgehend von den Ventilen die Suche entgegen der typischen Wirkrichtung fortgesetzt werden, bis entweder kein weiterer Pfad verfolgt werden kann oder ein vorher definierter Suchradius erreicht wird.

Aspektmetamodell Wirkrichtung

Damit durch Inferenzen gefundene Wirkzusammenhänge effizient innerhalb des Instanzmodells dargestellt werden können, empfiehlt sich die Definition eines eigenen Aspektmetamodells für die Wirkrichtung. Dies ist in Abbildung 6.28 in Form eines Klassendiagramms dargestellt. Kern des Modells ist die Beziehung Wirkung (effect) mit den beiden Rollen Ursprung (origin) und Ende (end). So existieren zwei Entitäten, die die Rollen Ursprung und Ende einnehmen. Dabei übt die Entität mit der Rolle Ursprung eine Wirkung auf die Entität mit der Rolle Ende aus.

6.7 Definition eines Prozesses zur modularen Erweiterung

Soll die beschriebene Systemstruktur um neue Aspekte erweitert werden, ist die Definition eines neuen Aspektmetamodells notwendig. Dabei ist grundsätzlich der in Abschnitt 1.2 beschriebenen Vorgehensweise zu folgen. Es muss also ausgehend von einer Anforderungsanalyse ein semiformales Aspektmetamodell erstellt werden. Dazu müssen mithilfe der vier Operationen Äquivalent-zu, Spezialisierung-von, in-Beziehung-zu oder Eigenschaft-von die Begriffe des neuen Aspekts auf das Metamodell bzw. die Aspektmetamodelle abgebildet werden. Anschließend kann das semiformale Aspektmetamodell auf eine konkrete Technologie abgebildet und in eine Implementierung überführt werden.

6.8 Evaluation der Modellhierarchie

In diesem Abschnitt wird die Modellhierarchie evaluiert. Dies geschieht auf zwei Wegen. Zunächst erfolgt die Evaluation anhand der in Kapitel 5 definierten Anforderungen. Anschließend erfolgt die Einordnung in das Klassifikationsschema für Assistenzsysteme.

6.8.1 Evaluation anhand der Anforderungen an die Modellhierarchie

Im Folgenden werden die Modellhierarchie bzw. die definierten Metamodelle mithilfe der in Abschnitt 5.3 definierten Anforderungen bzw. Qualitätskriterien Vollständigkeit (Q1), Flexibilität bzw. Adaptierbarkeit (Q2), Konsistenz und Portabilität (Q3), Zugänglichkeit sowie Verständlichkeit und Effizienz (Q4) bewertet.

Wie bereits in Abschnitt 5.3 beschrieben, ist eine a priori Definition eines grundsätzlich vollständigen Metamodells fraglich. Deshalb wurde bei der Entwicklung der Modellhierarchie auf Flexibilität, Adaptier- und Erweiterbarkeit der Systemstruktur Wert gelegt. Durch die Verwendung eines Metametamodells wird sichergestellt, dass Erweiterungen möglich sind. So ist es nur dank des gemeinsamen Metametamodells möglich, unterschiedliche Aspekte einer Anlage getrennt voneinander zu betrachten und einheitliche, aber separate Aspektmetamodelle zu definieren. Sollten in Zukunft weitere Aspekte einer Anlage betrachtet werden, können für diese auf Basis des Metametamodells neue Aspektmetamodelle entwickelt werden. Außerdem besteht die Möglichkeit, die bestehenden Aspektmetamodelle durch die Definition von weiteren Entitäts- oder Ereignisklassen zu erweitern. Auf diesem Weg wird eine hohe Flexibilität und Adaptierbarkeit des Konzepts sichergestellt. Außerdem wurde eine Prozedur zur modularen Erweiterung beschrieben (vgl. Abschnitt 6.7).

Neben der grundsätzlichen Vollständigkeit (d. h. auch für alle zukünftigen Anwendungsfälle) kann auch die aktuelle Vollständigkeit betrachtet werden. Dazu erfolgt in Tabelle 6.8.1 eine Zuordnung der einzelnen zuvor definierten Aspektmetamodelle zu den in Abschnitt 2.5 als benötigt identifizierten Informationsklassen. Es ist zu erkennen, dass alle benötigten Informationsklassen mithilfe der Aspektmetamodelle abgebildet werden können. Damit lässt sich mithilfe der definierten Modellhierarchie in Bezug auf die gestellten Kompetenzfragen ein vollständiges Objektmodell erstellen.

Tabelle 6.2: Zuordnung der benötigten Informationen zu den Aspektmetamodellen

Informationsklasse	Aspektmetamodelle				
	Topologie	Messen	Deployment	Steuerungs-Hardwarestruktur	Software-Kopplung
Anlagenstruktur	✓	✓	✓	✓	✓
Messwerte		✓			
Ereignisse	✓	✓	✓	✓	✓
Semantik	✓	✓	✓	✓	✓
Zeitlicher Bezug	✓	✓	✓	✓	✓

Flexibilität und Adaptierbarkeit stehen im Widerspruch zu Qualitätskriterien wie Konsistenz und Portabilität. Eine hohe Flexibilität oder Adaptierbarkeit führt durch die vielen möglichen Veränderungen automatisch zu Einschränkungen hinsichtlich der Konsistenz und der Portabilität. Dieser Widerspruch wird in der definierten Modellhierarchie durch den Einsatz des Metametamodells aufgehoben. Dieses stellt die Konsistenz der einzelnen Aspektmetamodelle sicher und bildet damit die Grundlage für Portabilität. Dank des Metametamodells ist eine Abbildung der gesamten Modellhierarchie auf die Syntax einer bestimmten Technologie einfach möglich.

Durch den Umstand, dass die unterschiedlichen Modelle auf allen Ebenen der Hierarchie bestehende Konzepte aus der Literatur aufgreifen, ist eine hohe Zugänglichkeit, Vollständigkeit und Verständlichkeit sichergestellt. Eine tiefgründigere Evaluation dieser Qualitätskriterien muss im Kontext einer konkreten Implementierung erfolgen.

Betrachtet man die verschiedenen Abstufungen der Kontextualisierung aus Abbildung 3.9, erreicht die vorgestellte Modellhierarchie die Stufe IV. Dank der unterschiedlichen

Aspektmetamodelle wird zwischen verschiedenen Beziehungstypen unterschieden. Das Rollenkonzept der Beziehungen ermöglicht die Evaluation der Wirkrichtung der Beziehungen. Außerdem wird durch das Lebenszykluskonzept der Beziehungen, das Entstehens- und dem Vergehensereignisse abbildet, eine Betrachtung der Beziehungen zu konkreten Zeitpunkten ermöglicht.

6.8.2 Evaluation anhand der Anforderungen an das Konzept

In Kapitel 5 wurden eine Reihe von Anforderungen an das spätere Kontextualisierungssystem definiert. An dieser Stelle wird untersucht, inwieweit diese bereits durch die Modellhierarchie erfüllt werden können und welche der Anforderungen erst im Zuge einer konkreten Implementierung evaluiert werden können. Die Evaluation ist in Tabelle 6.4 aufgeführt.

Tabelle 6.4: Evaluation anhand der in Kapitel 5 definierten Anforderungen

Anforderung	Erfüllt	Erläuterung
A1.0	✓	Die Modellhierarchie baut auf bestehenden Kernmodellen auf und berücksichtigt existierende Informationsmodelle.
A1.1	✓	Dank der Übernahme des in Abschnitt 3.1.2 vorgestellten Konzepts zur Integration von dynamischen Informationen und des Konzepts des Beziehungsobjekts aus der ISO 15926-2 ist es möglich, mithilfe der entwickelten Modellhierarchie Änderungen innerhalb des Betrachtungszeitraums darzustellen.
A1.2	✓	Basis der entwickelten Modellhierarchie ist das Beziehungskernmodell. Dies stellt sicher, dass Strukturinformationen und Beziehungen adäquat abgebildet werden können.
A1.3	✓	Die unterschiedlichen Aspektmetamodelle definieren unterschiedliche Beziehungsklassen. Durch die Zuordnung einer Verknüpfung zu einer Beziehungsklasse kann diese näher beschrieben werden. Dank der Klassifikation kann diese Beschreibung im Gegensatz zu einer näheren Beschreibung mit beliebigen Attributen auch maschinell ausgewertet werden.
A1.4	✓	Siehe A1.3.
A1.5	✓	Neue Aspekte können durch die Definition eines neuen Aspektmetamodells unter Berücksichtigung des Metamodells integriert werden.
A2	-	Die Rückwirkungsfreiheit auf den Anlagenbetrieb ist abhängig von der Implementierung.
A3	-	Die Implementierung muss die Konsistenz der Daten sicherstellen.

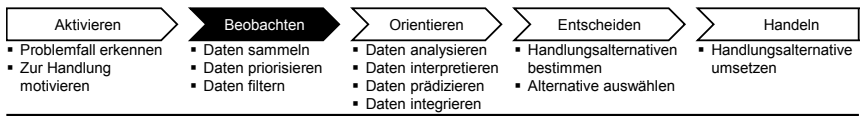


Abbildung 6.29: Unterschiedliche Phasen einer Handlung gemäß dem Klassifikationsschema für Assistenzsysteme aus [22, 23]

Tabelle 6.4: Evaluation anhand der in Kapitel 5 definierten Anforderungen

Anforderung	Erfüllt	Erläuterung
A4	✓	Durch die Modellhierarchie bzw. die Verwendung von Metamodellen auf Basis von bestehenden Informationsmodellen können einfache Modelltransformationen zwischen den jeweiligen Informationsmodellen für die Integration von Informationen definiert werden. Es müssen also ausschließlich die Modelltransformationen konfiguriert und gewartet werden.
A5	✓	Die Trennung von Wissen und Anwendung muss durch die Implementierung sichergestellt werden.
A6	✓	Mithilfe des Attributs dataSource aus dem Metamodell kann jedes Informationsobjekt mit einem Informationsursprung versehen werden. Auf diesem Weg wird die Nachverfolgbarkeit sichergestellt.
A7	-	Die Vertraulichkeit wird aktuell nicht betrachtet.
A8	✓	Es wurde ein Prozess zur Erweiterung um neue Aspekte definiert.

6.8.3 Klassifikation als Assistenzsystem

Im Abschnitt 3.1.1 wird ein Klassifikationsschema für Assistenzsysteme vorgestellt. Da im Rahmen dieser Arbeit ein Konzept als Grundlage für ein Kontextualisierungssystem entwickelt wird, das den Nutzenden bei der Verwendung und dem Verständnis von bestehenden Daten unterstützen soll, gibt eine Einordnung des Konzepts in das Klassifikationsschema einen guten Überblick über dessen Einsatzzweck und Grenzen.

In Abbildung 6.29 sind die fünf Handlungsphasen des Klassifikationsschemas mit jeweils einer kurzen Erläuterung dargestellt. Hervorgehoben ist die Phase Beobachtung, in der, nachdem die Notwendigkeit einer Handlung erkannt wurde, Daten gesammelt, priorisiert und gefiltert werden. Hier liegt das Hauptaugenmerk des Konzepts dieser Arbeit. Dank der expliziten Modellierung des Kontexts können relevante Daten einfacher und schneller gesammelt werden. Aufgrund des modellbasierten Ansatzes wird außerdem eine Priorisierung und Filterung anhand der unterschiedlichen Klassen erleichtert. Damit unterstützt das Konzept in der Phase der Orientierung die Analyse, Interpretation und Integration der Daten.

Innerhalb der Phase Beobachten kann ein auf dem Konzept aufbauendes Kontextualisierungssystem die Stufen 2 bis 4 der Assistenzstufen des Klassifikationsschemas erreichen.

Das System sammelt die Daten eigenständig (Vollständigkeit vorausgesetzt) und klassifiziert diese. Dank der Klassifikation werden die Nutzenden beim Filtern und Priorisieren der Informationen unterstützt. Auch ist eine teilweise automatische Priorisierung und Filterung durch das System denkbar.

In den Phasen Aktivieren, Orientieren, Entscheiden und Handeln findet für die Nutzenden grundsätzlich erstmal keine Unterstützung durch das System statt. Denkbar sind aber Erweiterungen, die auf Basis der durch das Konzept bereitgestellten Daten, in diesen Phasen Assistenz bereitstellen. So könnten beispielsweise zur Unterstützung in der Phase Entscheiden unterschiedliche Handlungsalternativen auf Basis vergangener Entscheidungen im Modell gefunden werden. Dazu könnte zum Beispiel passend zu einem aktuellen Problemfall nach ähnlichen Kombinationen aus Ereignissen, Beziehungen und Entitäten aus der Vergangenheit gesucht und als Handlungsalternativen präsentiert werden.

7 Realisierung und prototypische Implementierung

In diesem Kapitel wird die Realisierung und prototypische Implementierung des vorgestellten Konzepts erläutert. Dabei wird zunächst auf die genutzte Anlage eingegangen. Anschließend werden die eingesetzten Werkzeuge und die eigentliche Umsetzung beschrieben. Im Rahmen dieser Umsetzung wird entsprechend der Vorgehensweise (vgl. Abschnitt 1.2) die in Kapitel 4 und 6 entwickelte leichtgewichtige Ontologie in eine schwergewichtige Ontologie überführt. Die Modellhierarchie wird also formalisiert.

7.1 Demonstration am Pumpwerk

Das Konzept dieser Arbeit wurde für das Pumpwerk des Lehrstuhls für Prozessleittechnik prototypisch umgesetzt. Bei dem Pumpwerk handelt es sich um eine Conti-Anlage. Eine Fotografie der Anlage ist in Abbildung 7.1 zu sehen. Wie hier zu erkennen ist, besteht die Anlage im Wesentlichen aus drei Behältern, zwischen denen mithilfe von fünf Pumpen Wasser hin und her gepumpt werden kann. Außerdem sind verschiedene Ventile installiert, mit denen einerseits die Durchflüsse geregelt und andererseits die Flusswege bestimmt werden können. Alle verbauten Sensoren und Aktoren stammen von namhaften Herstellern aus der Prozessindustrie und werden so auch in prozesstechnischen Anlagen eingesetzt. Das in Abbildung 7.2 dargestellte R&I-Diagramm gibt eine Übersicht über alle verbauten Sensoren.

Alle verbauten Sensoren und Aktoren sind für die Kommunikation mit dem Prozessleitsystem an ein Remote Input/Output (RIO) Modul über die Stromschnittstelle¹ bzw. 4...20 mA angeschlossen. Im Rahmen dieser Arbeit wurde eine ET200M der Firma Siemens

¹<https://de.wikipedia.org/wiki/Stromschnittstelle>



Abbildung 7.1: Die Praktikumsanlage Pumpwerk des Lehrstuhls für Prozessleittechnik

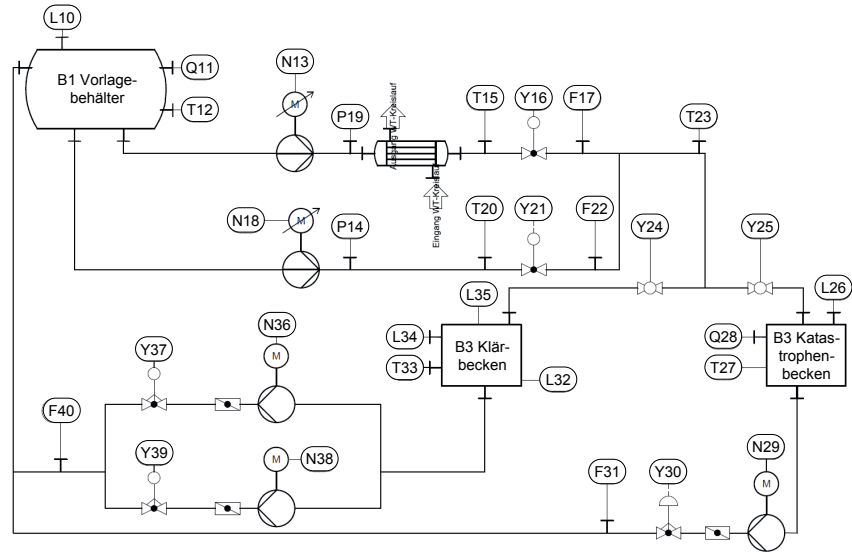


Abbildung 7.2: vereinfachtes R&I-Diagramm des Pumpwerks

als RIO eingesetzt. Zur Automatisierung des Pumpwerks wurde ein Siemens SIMATIC Microbox PC 427 B verwendet. Dieser Industrie-PC besitzt eine Profibus-Schnittstelle und kann so mit der RIO kommunizieren. Da für die eigentliche Prozessführung eine ACPLT/RTE²-basierte Lösung (vgl. Abschnitt 7.2.2) eingesetzt wurde, wurde die WinLC RTX SoftSPS auf dem Industrie-PC so projiziert, dass das Prozessabbild in einen geteilten Speicherbereich geschrieben wird und so durch eine ACPLT/RTE-Instanz verarbeitet werden kann. Das genaue Vorgehen dazu ist in [115] und [116] beschrieben.

Da der Autor dieser Arbeit aufgrund der Corona-Pandemie im Jahre 2020 gezwungen war im Homeoffice zu arbeiten, wurde die SIMATIC Microbox alternativ zur echten Anlage auch an eine Anlagensimulation angeschlossen. Dazu wurde eine zweite SIMATIC Microbox mit einer emulierten RIO und einer Prozesssimulation verwendet. Die Kommunikation mit der Anlagensimulation erfolgt ebenso wie mit der realen Anlage über Profibus. Dieser Aufbau ist in Abbildung 7.3 dargestellt. In der Abbildung ist auch die weitere Infrastruktur zu erkennen. So wurde je ein Raspberry Pi als openAAS-Server, Prozesshistorian und für die Einzelsteuereinheiten eingesetzt. Im Folgenden werden die unterschiedlichen Datenquellen zum Pumpwerk kurz erläutert.

7.1.1 Datenquelle: openAAS-Server

Zur Verwaltung von Assetinformationen wurde für das Pumpwerk der Verwaltungsschalenserver aus dem openAAS-Projekt (vgl. Kapitel 4.2.2) auf einem Raspberry Pi installiert.

²Aachener Prozessleittechnik/ Run Time Environment (ACPLT/RTE)

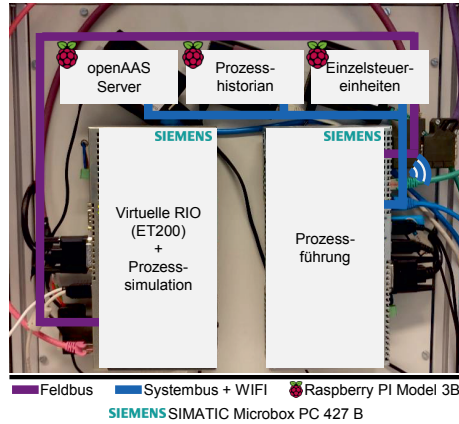


Abbildung 7.3: Aufbau der Automatisierungslösung^a

^a Quelle der Logos: https://de.wikipedia.org/wiki/Datei:Raspberry_Pi_Logo.svg,
<https://de.wikipedia.org/wiki/Datei:Siemens-logo.svg>

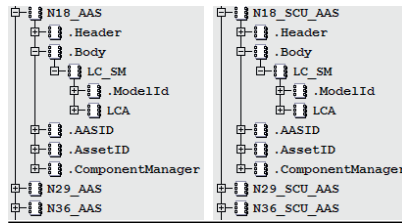


Abbildung 7.4: Struktur der Verwaltungsschale der Stelle N18 (links) sowie der dazugehörigen Einzelsteuereinheit (rechts) - Screenshot aus iFBSpro

Innerhalb des Servers wurden sowohl für jede PLT-Stelle als auch für zugehörige Einzelsteuereinheiten eine Verwaltungsschale angelegt. In Abbildung 7.4 ist die Struktur jeweils einer Beispielverwaltungsschale zu sehen: Auf der linken Seite für die Stelle N18 und auf der rechten Seite für die dazugehörige Einzelsteuereinheit.

Kern aller für das Pumpwerk angelegten Verwaltungsschalen ist das Lebenszyklus-Teilmodell (LifeCycle SubModel) (LC_SM), das im Body der Verwaltungsschale verwaltet wird. Es besteht aus dem Lebenszykluseintragsarchiv (LiveCycle Entry Archive) (LCA), in dem beliebig viele Lebenszykluseinträge archiviert werden können. Im Rahmen dieser Arbeit wurde hier für alle Stellbefehle (Ventil auf/zu bzw. Pumpe an/aus) der Aktuatoren in der jeweiligen Stellenverwaltungsschale ein Lebenszykluseintrag angelegt. Außerdem wurden in den Verwaltungsschalen der Einzelsteuereinheiten die Belegungsvorgänge in Form von Lebenszykluseinträgen archiviert.

ts	F17_PV	F22_PV	F31_PV	F40_PV	L10_PV	L26_PV	L34_PV	L35_PV	P14_PV	P19_PV	T12_PV
2020-01-22 16:16:17	0.209413	0.809874	0	0	17.0382	14.5753	14.6491	14.6491	1069.7	1269.93	21.9974
2020-01-22 16:18:19	0.209413	0.809874	0.670851	0	33.0849	11.8161	3.63597	3.63597	1069.7	1269.93	22.0372
2020-01-22 16:18:49	0.209413	0.809874	0.635957	0	31.1966	11.1889	5.79954	5.79954	1069.7	1269.93	22.1203

Abbildung 7.5: Datentabelle des Prozesshistorian - Screenshot aus pypMyAdmin

7.1.2 Datenquelle: Prozesshistorian

Als Prozesshistorian wurde auf einem Raspberry Pi das freie Datenbankmanagementsystem MariaDB³ installiert. Bei MariaDB handelt es sich um eine Abspaltung von MySQL. Es eignet sich dementsprechend zur Verwaltung von relationalen Datenbanken. In MariaDB wurde eine Tabelle zur Archivierung aller Messwerte angelegt. Ein Ausschnitt davon ist in Abbildung 7.5 zu sehen. Die Daten werden über die httpREST-Schnittstelle der ACPLT/RTE (vgl. Abschnitt 7.2.2) aus der Prozessführungs-Microbox ausgelesen, in die Tabelle geschrieben und dort archiviert. Zur Archivierung der Prozessdaten wurde der Raspberry Pi mit einer 1TB externen Festplatte ausgestattet.

7.1.3 Datenquelle: Einzelsteuereinheiten

Auf den SIMATIC Microbox Industrie PCs ist das Betriebssystem Windows XP installiert. Aufgrund der Inkompatibilität zwischen Windows XP und der offenen OPC UA Implementierung open62541 auf der die ACPLT/RTE OPC UA-Bibliothek aufbaut, wurde entschieden, für die Demonstration des Konzepts die Einzelsteuerebene auf einem weiteren Raspberry Pi zu installieren. Die Kommunikation zwischen der ACPLT/RTE-Instanzen auf dem Raspberry Pi und auf der Microbox erfolgt mittels des ACPLT/KS-Kommunikationsprotokolls.

Der Aufbau der Einzelsteuereinheiten erfolgt gemäß dem Metamodell für Komponenten [97], das im Rahmen des BaSys 4.0-Projekts⁴ definiert wurde. Dementsprechend besitzen die Einzelsteuereinheiten einen Belegungsautomaten, eine Fahrweise und eine Fähigkeit. Die Einzelsteuereinheiten der Pumpen besitzen die Fahrweise Pump, die die Fähigkeit Produktflusserzeugen aktiviert, während die Einzelsteuereinheiten der Ventile die Fahrweise Valve besitzen, die die Fähigkeit Produktflusszulassen aktiviert. Abbildung 7.6 zeigt exemplarisch die Einzelsteuereinheit der Pumpe N29 aus der ACPLT/RTE Engineering Oberfläche. Zu erkennen ist der Kommandoeingang, die Signaleingänge und der Zustand der Komponente.

Abbildung 7.7 zeigt exemplarisch die Repräsentanz des Füllstandsensors L26 in der Prozessführung. Diese ist insbesondere für die Erzeugung der Alarm- und Warnsignale verantwortlich. In dem gezeigten Fall werden sowohl für das Erreichen einer oberen Füllstandgrenze als auch für das Unterschreiten einer unteren Füllstandgrenze Alarm- und Warnsignale generiert. Über diese Signale ist die Sensorabstraktionskomponente zum Beispiel mit der Einzelsteuereinheit der Pumpe N29 verknüpft. Diese Verknüpfungen können vollständig per OPC UA erkundet werden.

³<https://de.wikipedia.org/wiki/MariaDB>

⁴<https://www.basys40.de/>

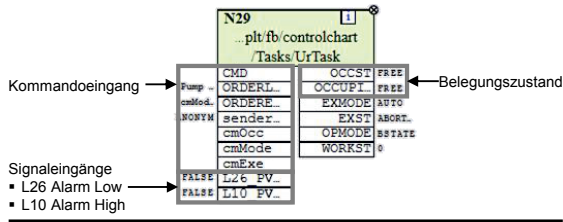


Abbildung 7.6: Einzelsteuereinheit der Pumpe N29 - Screenshot aus der ACPLT/RTE Engineering Oberfläche

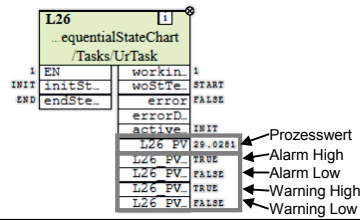


Abbildung 7.7: Sensorabstraktion des Füllstandsensors L26 - Screenshot aus der ACPLT/RTE Engineering Oberfläche

7.1.4 Datenquelle: Engineering

Zur Erstellung und Verwaltung von Engineering Daten wurde die ACPLT/RTE PandIX-Engineering Oberfläche (vgl. Abschnitt 4.2.1) verwendet. Das so erstellte R&I-Diagramm wurde als CAEX-XML-Datei exportiert. Diese ist im Anhang in Abschnitt A.1 beigefügt. In der Datei ist zum Beispiel von Zeile 36 bis 98 der Vorlagebehälter B1 als CAEX-InternalElement mit verschiedenen Attributen wie Nominalvolumen oder Gesamtvolumen beschrieben. Außerdem sind die Schnittstellen des Behälters als ExternalInterface aufgeführt. So ist zum Beispiel in Zeile 59 bis 65 die Produktschnittstelle beschrieben, über die der Behälter mit einer Rohrleitung verbunden ist. Diese ist in Zeile 154 bis 162 als ein weiteres CAEX-InternalElement beschrieben. Die Verknüpfung zwischen beiden Elementen ist in Zeile 495 mithilfe eines CAEX-InternalLinks definiert.

7.2 Verwendete Werkzeuge

7.2.1 GRAKN.AI

Als Grundlage für die Erstellung und Speicherung des Wissensgraphen wird GRAKN.AI⁵ verwendet. Bei GRAKN.AI handelt es sich um eine unter AGPLv3⁶ Lizenz veröffentlichte Implementierung einer Graphdatenbank. Die Implementierung besteht aus der Datenbank-

⁵<https://grakn.ai/>, <https://en.wikipedia.org/wiki/GRAKN.AI>

⁶GNU Affero General Public License - Version 3

Listing 7.1: Schemadefinition in Graql

```

define
# Definition des Entitätstypen aEntity
aEntity sub entity,
    has aName,
    plays aRole;

# Definition des Beziehungstypen aRelation
aRelation sub relation,
    relates aRole;

# Definition des Attributstypen aName
aName sub attribute,
    datatype string;

```



Abbildung 7.8: Das in Listing 7.1 definierte Schema - Screenshot aus der Grakn Workbase

Plattform Grakn, aus der Abfragesprache Graql und aus dem Visualisierungswerkzeug GRAKN.AI-Workbase. Außerdem können die Daten in einer Grakn-Datenbank mithilfe der Grakn-Console oder mithilfe von Java, Node.js sowie Python-Klienten manipuliert und abgefragt werden.

Die Datenbank-Plattform basiert auf dem in Abschnitt 4.1.4 beschriebenen Metameta-modell. Mit GRAKN.AI-Workbase bietet Grakn ein Werkzeug zur grafischen Erstellung des Schemas sowie zur Darstellung der Instanzebene.

Graql

Graql ist die Abfragesprache für einen in Grakn abgelegten Wissensgraphen [117]. Es handelt sich dabei um eine deklarative Sprache. Dementsprechend wird mit ihr die gesuchte Information beschrieben, die dann von Grakn bereitgestellt wird. Graql stellt sowohl die Data Manipulation Language (DML) (dt.: Datenbearbeitungssprache) als auch die Data Definition Language (DDL) (dt.: Datendefinitionssprache) dar. Das heißt, dass mithilfe von Graql einerseits der eigentliche Wissensgraph abgefragt und manipuliert sowie andererseits das Schema editiert und ausgelesen werden kann.

Exemplarisch ist in Listing 7.1 dargestellt, wie mithilfe von Graql ein Schema für eine Grakn-Datenbank definiert werden kann. Das definierte Schema ist in Abbildung 7.8 dargestellt. In Listing 7.2 wird gezeigt, wie auf Basis des Schemas Instanzen mit Graql angelegt werden können. Das dazugehörige Instanzmodell ist in Abbildung 7.9 zu sehen.

Listing 7.2: Instanziierung einer Entität und einer Beziehung

```
#Instanziierung einer Entität vom Typ aEntity mit dem Name
  EntityA
insert
  $a isa aEntity, has aName "EntityA";

#Instanziierung einer Beziehung vom Typ aRealtion zwischen
  EntityA
# und EntityB
match
  $a isa aEntity, has aName "EntityA";
  $b isa aEntity, has aName "EntityB";
insert
  $r (aRole: $a, aRole: $b) isa aRelation;
```

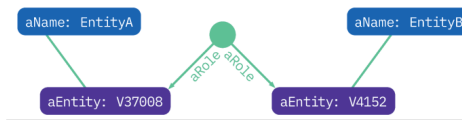


Abbildung 7.9: Die in Listing 7.2 definierte Beziehung zwischen EntityA und EntityB - Screenshot aus der Grakn Workbase

Aktuelle Einschränkungen von Grakn und Graql

Grakn befindet sich aktuell in der Entwicklung. Aus diesem Grund sind noch nicht alle Funktionalitäten vollumfänglich verfügbar. Bei der prototypischen Umsetzung für diese Arbeit sind folgende Einschränkungen aufgefallen:

- **Kardinalitäten:** Aktuell ist es nicht möglich die Kardinalitäten einer Beziehung im Schema festzulegen. Damit muss auf Klient-Seite sicher gestellt werden, dass das definierte Metamodell eingehalten wird.
- **Abstrakte Attribute:** Aktuell ist die Definition von abstrakten Attributstypen nicht möglich. Das bedeutet, dass eine explizite Definition einer Attributstyphierarchie in Graql nicht möglich ist.
- **Vererbung von Beziehungsrollen:** Aktuell werden Rollen in einer Beziehungstypenhierarchie nicht vererbt. Damit müssen diese für jede Beziehung explizit definiert werden. Dies erhöht die Wahrscheinlichkeit von Fehlern in der Schemadefinition und führt zu redundanten Aussagen in dieser.
- **Ineffizienter Umgang mit dem Datentyp Date:** Der Umgang mit dem Datentyp Date (z. B. 2020-02-02 10:10:10) ist sehr ineffizient. So sind Sortierungen entsprechend der zeitlichen Reihenfolge (beispielsweise zur Abfrage der aktuellsten Eigenschaftswertaussage) aktuell nicht möglich. Abhilfe schafft hier die Verwendung eines Zeitstempels.
- **Keine Berücksichtigung der Rollen bei kürzester Pfadfindung:** Aktuell werden Rollen bei der Suche nach dem kürzesten Pfad nicht berücksichtigt. Momentan ist ausschließlich eine Einschränkung der Entitäts- und Beziehungstypen möglich. Das bedeutet, dass Randbedingungen von gerichteten Beziehungen nicht berücksichtigt werden können. Wird zum Beispiel ein Pfad zwischen zwei Behältern gesucht, kann die Flussrichtung nicht als Einschränkung in einem entsprechenden Query vorgegeben werden. Dadurch kann es vorkommen, dass der kürzeste Pfad physikalisch nicht sinnvoll ist. Dies ist insbesondere bei dem Pumpwerk der Fall, da es einen geschlossenen Kreis darstellt.
- **Umkreissuche:** Eine Abfrage mit einem bestimmten Suchradius (z. B. drei Knoten) ist aktuell nicht möglich. Abhilfe schafft hier der iterative Aufruf des Queries mithilfe eines Klienten.

7.2.2 ACPLT/RTE

Wie in Abschnitt 7.1 beschrieben, wird die Laufzeitumgebung ACPLT/RTE⁷ für die Prozessführung eingesetzt. Außerdem dient sie als Grundlage für den Verwaltungsschalen-Server.

ACPLT/RTE ist eine am Lehrstuhl für Prozessleittechnik entwickelte offene Laufzeitumgebung für die Prozessindustrie. Sie wurde entwickelt, um eine effiziente und schnelle Validierung von neuen Konzepten der Automatisierungstechnik zu ermöglichen. Da die

⁷<https://github.com/acplt/rte>

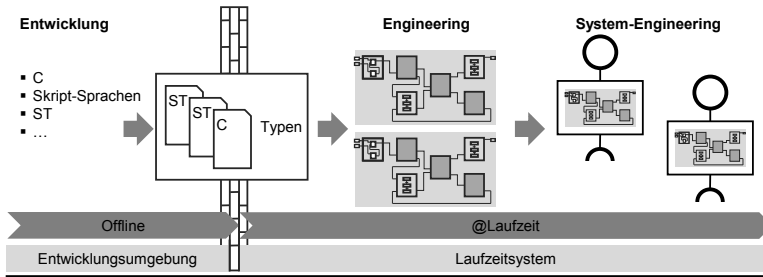


Abbildung 7.10: Entwicklungsprozess für eine Automatisierungslösung in ACPLT/RTE

Laufzeitumgebung in ANSI C umgesetzt ist, ist sie hoch portabel. Kern ist ein modellbasiertes Objektverwaltungssystem, in dem die Objekte zur Laufzeit erkundet werden können. Eine Anpassung der Laufzeitumgebung an neue Einsatzzwecke ist schnell und einfach durch die Definition von neuen Objektbibliotheken möglich. Diese Objektbibliotheken können dynamisch zur Laufzeit geladen werden.

Abbildung 7.10 zeigt den Standardentwicklungsprozess für ACPLT/RTE. Dieser wird zur Entwicklung einer neuen Automatisierungslösung von links nach rechts durchlaufen. Zunächst wird in der Offline-Phase mithilfe einer Entwicklungsumgebung zum Beispiel in C, ST oder einer Skriptsprache eine neue Typenbibliothek entwickelt. Wurde diese erfolgreich getestet und zertifiziert, können von den verschiedenen Typen Instanzen angelegt werden und zu einer Steuerungslogik zusammengeschaltet werden. Dieser Schritt wird als Engineering bezeichnet und findet zur Laufzeit und in der Laufzeitumgebung statt. Abschließend folgt das System-Engineering. Hier werden einzelne Teillösungen in eine Gesamtlösung integriert.

ACPLT/RTE bietet eine Reihe unterschiedlicher Kommunikationsschnittstellen an. Relevant für diese Arbeit ist einerseits die httpREST⁸-Schnittstelle und andererseits Open Platform Communication Unified Architecture (OPC UA).

Über die httpREST-Schnittstelle ist es möglich, sowohl einzelne Daten aus dem ACPLT/RTE-Server als auch die interne Struktur des Servers abzufragen. Außerdem können sowohl die Struktur als auch einzelne Werte über das Protokoll manipuliert werden. Wie in Abschnitt 7.1.2 beschrieben, wird diese Schnittstelle im Rahmen dieser Arbeit zur Abfrage der aktuellen Prozesswerte verwendet.

OPC UA ist ein plattformunabhängiger Standard für den Austausch von Daten. OPC UA zeichnet sich dadurch aus, dass es nicht nur Daten transportieren kann, sondern auch in der Lage ist, diese in hierarchischen Strukturen zu beschreiben. Dieser Umstand und der Vorteil von ACPLT/RTE, dass die Instanzstruktur zur Laufzeit erkundet werden kann, wird im Rahmen dieser Arbeit zur Erkundung der Steuerungsstruktur mithilfe von OPC UA ausgenutzt.

⁸https://de.wikipedia.org/wiki/Representational_State_Transfer

Aktuelle Einschränkungen von ACPLT/RTE

ACPLT/RTE befindet sich aktuell in der Entwicklung. Aus diesem Grund sind nicht alle Funktionalitäten voll umfänglich verfügbar. Folgende Einschränkungen sind im Rahmen der prototypischen Implementierung aufgefallen:

- **OPC UA Schnittstelle:** Die OPC UA Bibliothek für ACPLT/RTE transformiert für jede Abfrage die Objekte aus dem ACPLT/RTE-Metamodell in das OPC UA Metamodell. Dieser Vorgang ist sehr ressourcenintensiv. Aus diesem Grund muss sichergestellt werden, dass die Erkundung der Steuerungsstruktur keine Rückwirkungen auf andere Prozesse hat, die auf dem gleichen Rechenknoten laufen.
- **Datetime-Datentyp bei httpREST-Schnittstelle:** Beim Setzen eines Wertes vom Typ `Datetime` über die `httpRest`-Schnittstelle erfolgt eine automatische Anpassung der Zeitzone. Dies muss bei der Sicherstellung der Synchronizität berücksichtigt werden.

7.3 Modellierung

Zur Umsetzung des Konzepts dieser Arbeit mithilfe von Grakn muss das in Kapitel 6.4 definierte Metametamodell auf das Grakn-Metamodell, das in Abschnitt 4.1.4 erläutert wurde, abgebildet werden. Diese Abbildung wird nachfolgend beschrieben. Zuvor werden noch einige allgemeine Festlegungen getroffen.

Grundsätzlich ist das Ziel dieser Arbeit der Aufbau einer Datenbank, die die Betriebsphase einer Anlage beschreibt. Deshalb ist das Überschreiben, Löschen oder Verändern von einmal instanziierten Daten nicht zielführend. Viel mehr müssen neue Informationen mithilfe von neuen Entitäten, Beziehungen oder Eigenschaftsausagen in dem Kontextualisierungssystem abgelegt werden. Aus dieser Festlegung ergibt sich auch, dass das Modell zu jedem Zeitpunkt genau einen Anlagenzustand aus der Retroperspektive beschreibt. Es werden also zum Beispiel keine zukünftigen Alternativen modelliert. Daraus ergeben sich folgende allgemeine Festlegungen:

R1 Einmal instanziierte Daten dürfen nicht verändert werden.

R2 Es existiert zu jedem Zeitpunkt nur genau ein gültiges Modell.

Die einzig erlaubten Ausnahmen zu diesen Regeln ist einerseits die Korrektur von Fehlern und andererseits das Aktualisieren des Attributs `validUntil` und des Vergehensereignisses.

Außerdem wird zu jedem Objekt der Datenursprung angegeben, um so eine Nachverfolgbarkeit der Daten zu ermöglichen. Daher gilt folgende Festlegung für alle Objekte:

R3 Zu jedem modellierten Objekt muss der Datenursprung angegeben werden.

7.3.1 Abbildung des Metametamodells auf das Grakn-Metamodell

Für die Umsetzung des Konzepts müssen die Begriffe `entity`, `relation`, `event`, `property` und `role` aus dem Metametamodell auf die Begriffe des Grakn-Metamodells abgebildet werden. Diese Abbildung wird nachfolgend erläutert. Zur besseren Unterscheidung zwischen den Begriffen beider Metametamodelle wird der Prefix `base` für die doppelten Begriffsbezeichnungen im Metametamodell verwendet.

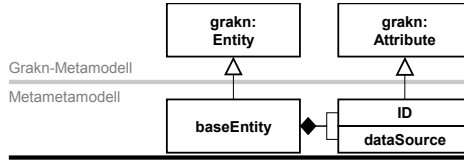


Abbildung 7.11: Abbildung einer Entität (baseEntity) aus dem Metametamodell auf das Grakn-Metamodell

Listing 7.3: Definition der baseEntity in Graql

```
define
  baseEntity sub entity , abstract ,
    has ID ,
    has dataSource ;

  ID sub attribute ,
    datatype string ;

  dataSource sub attribute ,
    datatype string ,
    has IDType ;

  IDType sub attribute ,
    datatype double ;
```

Entität

Abbildung 7.11 und Listing 7.3 zeigen die Abbildung einer Entität als Klassendiagramm und als Schemadefinition in Graql. In dem Listing werden außerdem die Attribute ID und dataSource definiert. Mithilfe von dataSource kann zur besseren Nachverfolgbarkeit der Datenursprung angegeben werden. Dabei setzt sich das Attribut aus der eigentlichen Referenz und einer Referenztypangabe zusammen. Eine Möglichkeit der Referenzangabe ist die Verwendung von Uniform Resource Identifier (URI)⁹.

Beziehung

In Listing 7.4 und in Abbildung 7.12 ist die Abbildung einer Beziehung aufgeführt. Zur Beschreibung des Lebenszyklus der Beziehung werden hier die beiden Attribute validFrom und validUntil eingeführt. validFrom beschreibt den Zeitpunkt des Entstehensereignisses der Beziehung, während validUntil den Zeitpunkt des Vergehensereignisses beschreibt. Mithilfe dieser beiden Attribute lässt sich der in Abbildung 6.6 beschriebene Lebenszyklus einer Beziehung vollständig darstellen. Für den Fall, dass $validFrom < t < validUntil$ gilt, ist die Beziehung zu dem Zeitpunkt t gültig/aktiv. In allen anderen Fällen gilt die Beziehung als ungültig oder deaktiviert.

⁹z. B.: <http://de.acplt.org/pumpwerk/F17>

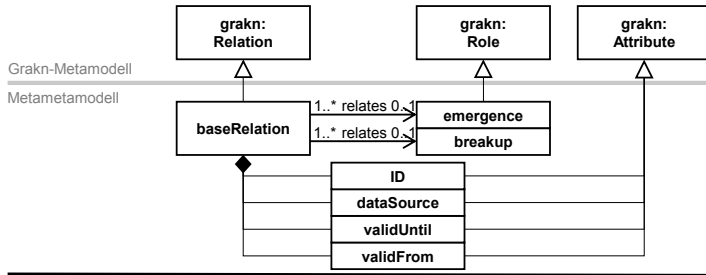


Abbildung 7.12: Abbildung einer Beziehung (baseRelation) aus dem Metametamodell auf das Grakn-Metamodell

Listing 7.4: Definition der baseRelation in Graql

```

define
  baseRelation sub relation , abstract ,
    has ID ,
    has dataSource ,
    has validFrom ,
    has validUntil ,
    relates emergence ,
    relates breakup ;

  validFrom sub attribute ,
    datatype date ;

  validUntil sub attribute ,
    datatype date ;

```

Zur Verknüpfung der Beziehung mit dem entsprechenden Vergehens- oder Entstehensereignis werden die beiden Graknrollen emergence und breakup eingeführt. Diese Rollen können jeweils von einem Ereignis ausgefüllt werden. Dabei wird die Rolle emergence (dt. Entstehung) vom Entstehensereignis und die Rolle breakup vom Vergehensereignis eingenommen.

Folgende Festlegungen gelten für eine Beziehung:

- R1** Der Lebenszyklus der Beziehung wird durch die Attribute validFrom und validUntil beschrieben.
- R2** Ist validUntil unbekannt, wird ein Defaultwert, welcher weit in der Zukunft liegt, gesetzt.
- R3** Eine Beziehung hat genau kein oder ein Entstehensereignis.
- R4** Eine Beziehung hat genau kein oder ein Vergehensereignis.

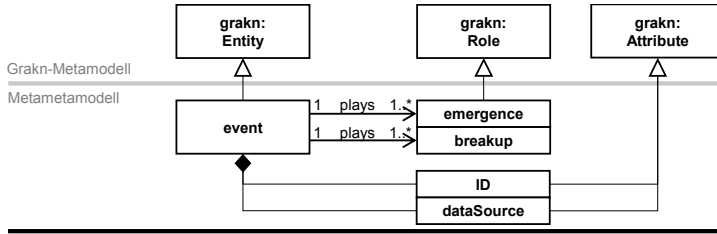


Abbildung 7.13: Abbildung eines Ereignisses (event) aus dem Metametamodell auf das Grakn-Metamodell

Listing 7.5: Definition des events in Graql

```
define
  event sub entity , abstract ,
    has ID ,
    has dataSource ,
    plays emergence ,
    plays breakup ;
```

Ereignis

Wie in Abbildung 7.13 und in Listing 7.5 gezeigt, werden Ereignisse (event) als Grakn-Entitäten modelliert. Damit können sie die Rollen emergence und breakup der baseRelation einnehmen und so das Entstehen oder Vergehen der Beziehung beschreiben.

Folgende Festlegungen gelten für ein Ereignis:

- R1** Ein Ereignis übernimmt im Bezug auf eine Beziehung entweder die Rolle emergence oder breakup oder keine Rolle.
- R2** Ein Ereignis kann in mehreren Beziehungen jeweils eine der beiden Rollen übernehmen.

Eigenschaft

In Abbildung 7.14 ist die Abbildung einer generischen Eigenschaft als abstrakte Grakn-Entität dargestellt. Da Grakn aktuell die Definition von abstrakten Attributstypen nicht erlaubt, werden in Listing 7.6 ausschließlich die Teilattribute timestamp, description und unit definiert. Für einen konkreten Eigenschaftstypen, wie z. B. Messwerte, muss ein eigenes Grakn-Attribut für die eigentliche Eigenschaftswertaussage bestimmt werden.

Rolle

Wie in Abbildung 7.15 dargestellt, können die Rollenklassen beider Modelle direkt aufeinander abgebildet werden. Die Definition von Rollen in Graql erfolgt implizit über die Definition von Beziehungen und Entitäten. So werden zum Beispiel in Listing 7.4 und 7.5 implizit die beiden Rollen emergence und breakup definiert.

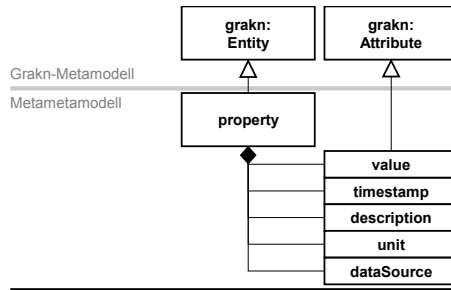


Abbildung 7.14: Abbildung einer Eigenschaft (property) aus dem Metametamodell auf das Grakn-Metamodell

Listing 7.6: Definition der Attribute in Graql

```
define
  property sub entity, abstract,
    has description,
    has timestamp,
    has unit,
    has dataSource;

  description sub attribute,
    datatype string;

  timestamp sub attribute,
    datatype date;

  unit sub attribute,
    datatype string;
```



Abbildung 7.15: Abbildung einer Rolle (role) aus dem Metametamodell auf das Grakn-Metamodell

Tabelle 7.1: Übersicht über die definierten Aspektmetamodelle

Aspektmetamodell	Definition in Kapitel	Listing
Topologie	6.5.1	A.2
Messen	6.5.2	A.3
Deployment	6.5.3	A.4
Steuerungstruktur	6.5.4	A.5
Hardware-Software-Kopplung	6.5.5	A.6
Wirkung	6.6.2	A.7

7.3.2 Abbildung der Aspektmetamodelle auf das Grakn-Metamodell

Dank der Abbildung des Metametamodells auf das Grakn-Metamodell können die einzelnen Aspektmetamodelle sehr einfach in Graql umgesetzt werden. Die Abbildungen sind im Anhang in Abschnitt A.2 aufgeführt. In Tabelle 7.1 ist eine Übersicht über die verschiedenen Definitionen sowie die jeweiligen Listings der Graql-Abbildungen der Aspektmetamodelle gegeben.

7.3.3 Abbildung der Datenquellen auf die Aspektmetamodelle

Im letzten Schritt müssen die Metamodelle der verfügbaren Datenquellen auf die Aspektmetamodelle abgebildet werden, damit anschließend das Instanzmodell generiert werden kann. Die für den Demonstrator dieser Arbeit verfügbaren Datenquellen sind im Abschnitt 7.1 aufgeführt. Für die Abbildung dieser Datenquellen wurde auf Basis des Grakn-Python-Klienten Transformationsvorschrift implementiert, die die Daten auslesen und in die Graphdatenbank überführen.

8 Evaluation der Ergebnisse

8.1 Evaluation anhand der Anforderungen

In Kapitel 5.2 wurde eine Reihe von Anforderungen an das zu entwickelnde Konzept bzw. die Implementierung beschrieben. In Tabelle 6.4 wurde bereits betrachtet, inwieweit die gestellten Anforderungen durch das Konzept erfüllt werden und welche Anforderungen durch die Implementierung erfüllt werden müssen. In Tabelle 8.1 wird diskutiert, in welchem Umfang die einzelnen Anforderungen durch die prototypische Implementierung erfüllt werden.

Tabelle 8.1: Evaluation anhand der in Kapitel 5 definierten Anforderungen

Anforderung	Erfüllt	Erläuterung
A1.0	✓	Das entwickelte Metamodell berücksichtigt die Begriffe der in Kapitel 4 vorgestellten Metamodelle und baut auf den dort vorgestellten Modellierungskonzepten auf. Dadurch wird sichergestellt, dass die relevanten Begriffe berücksichtigt werden. Auch zeigt die prototypische Implementierung, dass alle zuvor gestellten Kompetenzfragen beantwortet werden können. Aufgrund des metamodellbasierten Ansatzes des entwickelten Konzepts wird auch das Uniformitätskriterium erfüllt. Gleiche Informationen werden auf die selbe Art und Weise repräsentiert. Dank der in Abschnitt 1.2 beschriebenen Vorgehensweise ist sichergestellt, dass bestehende Informationsmodelle die konzeptionelle Basis für die Modellhierarchie dieser Arbeit liefern.
A1.1	✓	Das Konzept bietet sowohl eine Lösung zur Beschreibung des Lebenszyklus von Entitäten (mithilfe von Beziehungen) als auch von Beziehungen (mithilfe von validFrom und validUntil) und ist damit in der Lage, Änderungen im Betrachtungszeitraum darzustellen und Zeitinformationen zu integrieren.
A1.2	✓	Mithilfe der Beziehungen werden zeitlich begrenzte Nachbarschaftsabhängigkeiten modelliert.
A1.3	✓	Dank der unterschiedlichen Beziehungsklassen können die Nachbarschaftsbeziehungen näher beschrieben bzw. klassifiziert werden.
A1.4	✓	Dank der klassifizierten Beziehungen kann zwischen verschiedenen Produktnetzen (z. B. primaryProduct bzw. secondaryProduct) unterschieden werden. Liefert die Informationsquelle weitere Unterscheidungsmerkmale zur Klassifikation der Produktnetze, können weitere Beziehungsklassen eingeführt werden.

Tabelle 8.1: Evaluation anhand der in Kapitel 5 definierten Anforderungen

Anforderung	Erfüllt	Erläuterung
A1.5	✓	Die Aspektorientierung bei der Entwicklung des Konzepts zeigt, dass eine modulare Erweiterung einfach möglich ist.
A2	(✓)	Da das Konzept zur Datenabfrage auf Standardschnittstellen der jeweiligen Datenquellen zugreift, liegt die Sicherstellung der Rückwirkungsfreiheit im Verantwortungsbereich des Schnittstellenherstellers. Sind diese so gestaltet, dass sie keinen negativen Einfluss auf die Datenquelle haben, ist auch die Rückwirkungsfreiheit des Konzepts dieser Arbeit gewährleistet.
A3	✓	Dank des Metamodells und des Datenbanksystems mit explizitem Schema, können der Datenbank keine Daten hinzugefügt werden, die nicht dem Schema bzw. dem Metamodell entsprechen. Dadurch wird die Konsistenz der Daten automatisch sichergestellt.
A4	✓	Dank des metamodellbasierten Ansatzes ist, unter der Voraussetzung dass auch die Datenquellen auf einem Metamodell aufbauen, die Konfiguration auf die Definition einer Transformationsvorschrift beschränkt. Ist diese einmal definiert, können alle entsprechenden Daten eingelesen werden. Dank des modularen Ansatzes kann dies schrittweise und nach Bedarf erfolgen. Zusätzlich wurde eine Prozedur zur Erweiterung beschrieben (siehe A8).
A5	✓	Durch die Verwendung eines Datenbanksystems mit explizitem Schema wird die Trennung zwischen dem in der Datenbank verwalteten Wissen und der Anwendung sichergestellt. In der Datenbank ist das Wissen in Form des Metamodells und der Instanzdaten hinterlegt und kann dort von einer beliebigen Anwendung abgefragt und erkundet werden. Das Metamodell ist explizit beschrieben und wird nicht verteilt in verschiedenen Anwendungskomponenten verwaltet.
A6	✓	Durch die Verankerung der Datenquelle im Metamodell (vgl. Abbildung 7.11, 7.13, 7.12 und 7.14) ist die Nachverfolgung des Datenursprungs einfach möglich.
A7	-	Die Sicherstellung der Vertraulichkeit wurde bei der aktuellen Entwicklung des Konzepts weder auf der Ebene des Metamodells noch innerhalb der Implementierung berücksichtigt. Allerdings könnten entsprechende Aspekte (z. B. Nutzerrollen- und Zugriffsrechtssystem) auf der Ebene des Metamodells integriert und die Implementierung entsprechend erweitert werden.
A8	✓	Die vorliegende Arbeit beschreibt das Vorgehen zur Modellentwicklung. Der konkrete Prozess ist in Abschnitt 1.2 und in Abschnitt 6.7 beschrieben.

Listing 8.1: Query zur Beantwortung der Frage: Wie hoch ist die Temperatur in Behälter B2?

```

match
# Welcher Tempertatursensor in dem Behälter mit der ID
  Vessel2 verbaut?
  $v isa vessel, has ID "Vessel2";
  $s isa temperatureSensor;
  $r(measurementPlace:$v, measurementDevice:$s) isa measuring;
# Was ist der Messwert mit dem aktuellsten Zeitstempel?
  $r has measuredValue $mV; $mV has timestamp $ts;
  $ts has isLatest true;
get
  $mV; offset 0; limit 1;

```

8.2 Evaluation anhand der Anwendungsszenarien

In Kapitel 2 wurden vier verschiedene Zielszenarien für das Konzept dieser Arbeit definiert. Nachfolgend wird anhand der prototypischen Implementierung gezeigt, wie diese Zielszenarien mithilfe des Konzepts erreicht werden.

8.2.1 Verknüpfung von Informationen zu Wissen

Im Rahmen dieses grundlegenden Zielszenarios soll gezeigt werden, wie mithilfe des Meta-modells Informationen aus unterschiedlichen Quellen verknüpft werden können und so der Kontext zu einer bekannten Information hergestellt werden kann.

Dazu müssen sich z. B. folgende Fragen beantworten lassen:

- Wie hoch ist die Temperatur in Behälter B2?
- Wie hoch ist der Druck nach Pumpe N18?
- Wie kann von Behälter 1 in den Behälter 3 gepumpt werden und welche Kommandos bieten die dazugehörigen Einzelsteuerungseinheiten an?

Zur Beantwortung dieser Fragen müssen Informationen aus mindesten zwei Datenquellen kombiniert werden. Für die ersten beiden Fragen werden einerseits Informationen über die Anlagenstruktur (Welcher Temperatursensor ist in Behälter B2 verbaut? bzw. Welcher Drucksensor ist nach Pumpe N18 verbaut?) und andererseits Informationen aus dem Prozesshistorian (Was ist der aktuelle Messwert des jeweiligen Sensors?) benötigt. Dank der definierten Aspektmetamodellen können diese Fragen mit einem einzelnen Query beantwortet werden. Die den Beispielfragen zugehörigen Queries sind in Listing 8.1 und 8.2 aufgeführt.

Die Beantwortung der dritten Beispielfrage gestaltet sich auf Grund der in Abschnitt 7.2.1 aufgeführten Einschränkungen von Grakn etwas komplizierter und muss in drei Schritten erfolgen:

1. **Finden der beiden Behälter:** Zunächst müssen die beiden Behälter in der Datenbank gefunden werden. Anschließen müssen die primaryProductConnection-Beziehungen, in denen der Ausgangsbehälter die Rolle source und der Endbehälter

Listing 8.2: Query zur Beantwortung der Frage: Wie hoch ist der Druck nach Pumpe N18?

```

match
# Welcher Drucksensor nach der Pumpe mit der ID N18 verbaut?
$a isa actuator, has ID "N18";
$p isa pipe;
$r1(source:$a, target:$p);
$s isa pressureSensor;
$r2(measurementPlace:$p, measurementDevice:$s) isa measuring;
# Was ist der Messwert mit dem aktuellsten Zeitstempel?
$r2 has measuredValue $mV; $mV has timestamp $ts;
$ts has isLatest true;
get
$mV; offset 0; limit 1;

```

Listing 8.3: Pfadbestimmung zwischen Behälter 1 und 3

```

compute
path from V41144, to V32832,
in [pipe, actuator, junction, externalInterface, heatExchanger,
    checkValve, primaryProductConnection];

```

die Rolle target einnimmt, gefunden werden. Auf diese Art und Weise wird sichergestellt, dass nur physikalisch sinnvolle Pfade gefunden werden.

2. **Pfadbestimmung:** Im nächsten Schritt kann der Pfad zwischen den zuvor gefundenen Elementen bestimmt werden. Dies geschieht mit dem Query aus Listing 8.3. Dabei ist V41144 die Grakn-ID der Rohrleitung aus Behälter 1 und V32832 die der Rohrleitung in Behälter 2. Mithilfe der Klausel „in“ wird eine White-List für Beziehungen und Entitäten angegeben.
3. **Finden der Kommandos:** Anschließend müssen die IDs für alle Entitäten entlang des Pfads aus der Datenbank ausgelesen und die Kommandos der den Aktuatoren zugehörigen Einzelsteuereinheiten herausgesucht werden.

Werden alle drei Schritte nacheinander durchgeführt, ergeben sich als Antwort für die Beispielfrage und als Pfad von Behälter 1 zu Behälter 2 zwei Möglichkeiten:

1. pipe111, **Pump18**, pipe112, **Valve21**, pipe113, jun130, pipe131, jun140, pipe161, **Valve25**, pipe162
2. pipe121, **Pump13**, pipe122, Heater43, pipe123, **Y16**, pipe124, jun130, pipe131, jun140, pipe161, **Valve25**, pipe162

Außerdem nehmen die Einzelsteuereinheiten aller Aktuatoren die Kommandos *ABORT*, *STOP*, *CLEAR*, *HOLD*, *UNHOLD*, *START*, *OCCUPY*, *FREE*, *RESET*, *AUTO*, *PRIO*



Abbildung 8.1: Screenshot einer Unterhaltung mit einem Chatbot als Assistenzsystem

und *MANUAL* entgegen. Zusätzlich können die Einzelsteuereinheiten der Ventile das Kommando *Valve* und die der Pumpen das Kommando *Pump* verarbeiten.

Assistenzsystem

Zur Demonstration der Interaktionsmöglichkeiten mit einer prozesstechnischen Anlage, die sich durch das Konzept ergeben, wurde mithilfe des KI-basierten Frameworks für virtuelle Assistenten Rasa¹ ein Chatbot als Prototyp für ein mögliches Assistenzsystem aufgebaut. Abbildung 8.1 zeigt beispielhaft eine Unterhaltung mit dem Assistenzsystem.

Es ist zum Beispiel vorstellbar, dass Feldingenieure oder Techniker für ihre Wartungsarbeiten und Vorortfehleranalysen mit einem solchen Assistenzsystem ausgestattet werden. Sie können sich dann, analog zu den von zuhause bekannten Assistenten Ok Google, Siri oder Alexa, mit der Anlage unterhalten und intuitiv den Kontext zu ihrer eigentlichen Problemstellung erfragen.

Dank des metamodellbasierten Ansatzes des Konzepts dieser Arbeit ist die Entwicklung eines solchen Assistenzsystems sehr effizient möglich. Zunächst können basierend auf den Aspektmetamodellen bestimmte Frageklassen definiert werden. Für diese müssen anschließend entsprechende Aktionen implementiert werden. Für den Chatbot in Abbildung 8.1 wurde jeweils eine Aktion implementiert, die ein Query, entweder ähnlich zu dem in Listing 8.1 oder zu dem in Listing 8.2, auslöst.

Da Rasa KI-basiert ist, werden sowohl für die Dialogführung als auch für die Absichtserkennung der einzelnen Aussagen Trainingsdaten benötigt. Dank des modellbasierten Ansatzes eignet sich das Konzept dieser Arbeit zur automatischen Generierung von Trainingsdaten zur Absichtserkennung.

¹<https://rasa.com/>

8.2.2 Aufdeckung von bisher nicht modellierten Zusammenhängen

Ziel dieses Szenarios ist es, zu zeigen, dass mithilfe des Konzepts das Aufdecken von bisher nicht modellierten Zusammenhängen möglich ist. Dabei sollen Zeitreihen mit diskreten Ereignissen in Verbindung gebracht werden. Da die prototypische Implementierung nur die automatische Abbildung von Aktor- und Belegungsereignissen erlaubt, werden ausschließlich diese in diesem Szenario betrachtet. Exemplarisch wird hier die in Abbildung 8.2 dargestellte Zeitreihe des Füllstands in Behälter 2 untersucht. Dieser wird mithilfe des Sensors L26 gemessen. Die Zeitreihe zeigt einen Befüllungsvorgang des Behälters. Auffällig sind fünf Änderungen der Zuflussgeschwindigkeit während des Vorgangs. Zur Identifikation der Ursache für diese Änderungen wird der Nachbarschaftsgraph auf Ereignisse durchsucht. Dazu wird der Graph für jede der sechs Phasen in zwei Schritten analysiert. Zunächst wird mithilfe des in Listing 8.4 dargestellten Queries die Nachbarschaft durchsucht. Anschließend werden die einzelnen Beziehungen in der Nachbarschaft auf Ereignisse untersucht. Dies geschieht mithilfe des Queries aus Listing 8.5. Die Ergebnisse dieser Suche sind für die sechs Phasen in Abbildung 8.3 dargestellt. Dabei ist L26 als Ausgangspunkt der Suche grün hervorgehoben. Der Suchradius wurde auf 15 Knoten beschränkt. Rot markiert sind Beziehungen, für die ein Ereignis gefunden wurde. Damit das Ergebnis der Suche übersichtlich und handhabbar bleibt, wurden diese auf Beziehungen des Typs `primaryProductFlow` beschränkt (vgl. Listing 8.4). Alternativ hätte die Suche beispielsweise auch auf bestimmte Ereignisse begrenzt werden können.

Konkret wurden folgende Phasen untersucht:

- **bis 17:14:16.874790:** Alle Ventile vor und nach Behälter sind geschlossen. Der Füllstand im Behälter ändert sich nicht.
- **ab 17:14:16.874790 bis 17:15:39.235964:** Die Ventile Y25 und Y16 sind geöffnet und die Pumpe N13 ist aktiv. Aus diesem Grund steigt der Füllstand und der Kontext erweitert sich um 15 Knoten in Richtung Behälter 1.
- **ab 17:15:39.235964 bis 17:16:41.343148:** Ventil Y21 und Pumpe N18 sind zusätzlich geöffnet bzw. aktiv. Der Füllstand steigt damit schneller. Da sich in dem Pfad über N18 und Y21 kein Wärmetauscher befindet, liegt Behälter 1 jetzt innerhalb des Suchradius.
- **ab 17:16:41.343148 bis 17:18:05.807487:** Das Ventil Y16 und die Pumpe N13 sind nicht mehr geöffnet bzw. aktiv. Damit reduziert sich die Zuflussgeschwindigkeit.
- **ab 17:18:05.807487 bis 17:19:07.909849:** Das Ventil Y24 ist geöffnet und Behälter 3 wird befüllt. Damit befindet sich dieser auch im Kontext von L26.
- **ab 17:19:07.909849:** Ventil Y25 ist geschlossen. Damit reduziert sich der Kontext von L26 auf den Behälter 2. Über den Zustand von Y21, Y24 und N18 kann aus dieser Perspektive keine Aussage getroffen werden.

Die in Abbildung 8.3 dargestellten Visualisierungen der Suchergebnisse wurde mithilfe von Gephi², einem offenen Graphvisualisierungswerkzeug, erzeugt. Dazu wurden die Suchergebnisse als GraphML-Dateien gespeichert und so in Gephi importiert. Die GraphML-Dateien sind im Anhang in Abschnitt A.3 beigelegt.

²<https://gephi.org/>

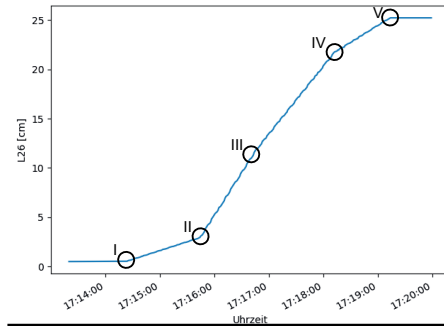


Abbildung 8.2: Füllstand in Behälter 2

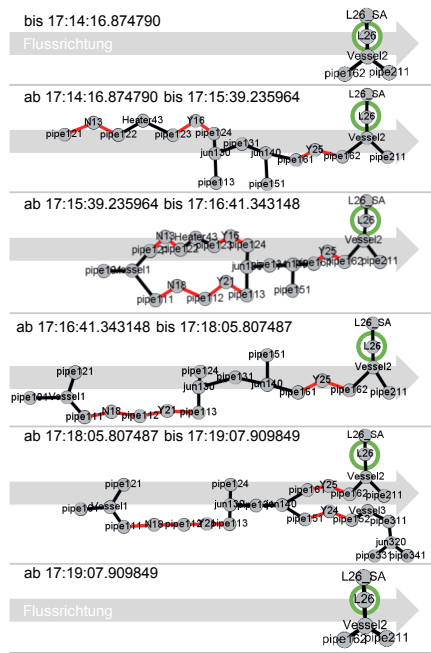


Abbildung 8.3: Nachbarschaft von L26 während der Phasen I-V aus Abbildung 8.2. Beziehungen mit einem Ereignis sind rot dargestellt.

Listing 8.4: Query zur iterativen Suche im Graph

```
# CURRENTID: ID des Elements dessen Nachbarn gesucht werden.
# PARENTID: ID des Elements dessen Nachbarn zuvor gesucht wurden.
# TIMESTAMP: Zeitpunkt
match
  $x has ID CURRENTID;
  $r($x, $y) isa primaryProductFlow;
  $r has validFrom $vF; $vF < TIMESTAMP ;
  $r has validUntil $vU; $vU > TIMESAMP ;
  $y isa productContainer; $y has ID $i; not{$y has ID PARENTID
  };
get;
```

Listing 8.5: Query zur Ereignisüberprüfung

```
# RELATIONID: ID der Beziehung
match
  $x id RELATIONID; $x($e); $e isa event;
get;
```

8.2.3 Erstellung von Datensubsets zur Datenanalyse

In diesem Anwendungsszenario soll gezeigt werden, dass es mithilfe des entwickelten Konzepts möglich ist, automatisch bestimmte Datensubsets zu erstellen. Dies kann zum Beispiel bei bestimmten datenbasierten Prediktive-Maintenance-Ansätzen hilfreich sein. Hier müssen zunächst bestimmte Anlagenkonstellationen gefunden und anschließend die entsprechenden Daten in den relevanten Betriebszuständen exportiert werden. In dem Pumpwerk kann beispielsweise nach einer Kombination aus einer Pumpe und einem Drucksensor gesucht werden (vgl. Abbildung 7.2). Die hierfür notwendigen Queries sind in Listing 8.6 aufgeführt. Mit dem ersten Query wird zunächst nach Kombinationen aus einer Pumpe, einer Rohrleitung und einem Drucksensor gesucht. Außerdem wird der Zeitraum, innerhalb dessen die Pumpe aktiv war, abgefragt. Das Ergebnis ist als Grakn Workbase-Screenshot in Abbildung 8.4 dargestellt. Mithilfe des zweiten Queries werden die eigentlichen Messwerte abgefragt. Das Ergebnis ist in Abbildung 8.5 dargestellt. Der gezeigte Druckverlauf ergibt sich, da zur Evaluation die Simulation des Pumpwerks verwendet wurde.

8.2.4 Validierung von Ergebnissen der Datenanalyse

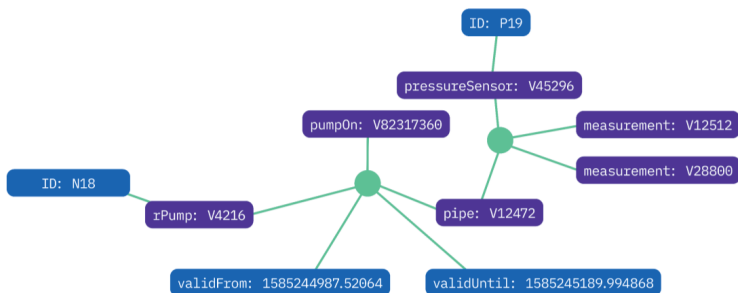
Werden mithilfe einer Datenanalyse Abhängigkeiten zwischen einzelnen Aspekten der Anlage gefunden, bietet der Nachbarschaftsgraph eine Möglichkeit zur weiteren Analyse bzw. zur Validierung dieser Ergebnisse. Dazu können die Pfade, ähnlich wie in den zuvor beschriebenen Szenarien (z. B. Abbildung 8.3), zwischen den Aspekten in dem Graphen bestimmt und untersucht werden.

Listing 8.6: Query zur Suche nach einer Pumpe mit nachgeschaltetem Drucksensor

```

match
  $p isa rPump, has ID $i1;
  $y isa pipe;
  $s isa pressureSensor, has ID $i2;
  $r1($p,$y) isa primaryProductFlow; $r1 has validFrom $vF, has
    validUntil $vU;
  $r2($y,$s);
get;
# RELATIONID: ID der Beziehung ($r2) zwischen Sensor und
  Messstelle
# VALIDFROM, VALIDUNTIL: Gültigkeitsbereich der Beziehung $r1 (
  relevanter Betriebszustand)
match
  $r2 id RELATIONID;
  $r2(measurementResult: $mR);
  $ts > VALIDFROM ; $ts < VALIDUNTIL;
  $mR has measuredValue $mV, has timestamp $ts, has unit $u;
get;

```

**Abbildung 8.4:** Teil des Ergebnisses des ersten Querys aus Listing 8.6 - Screenshot aus Grakn Workbase

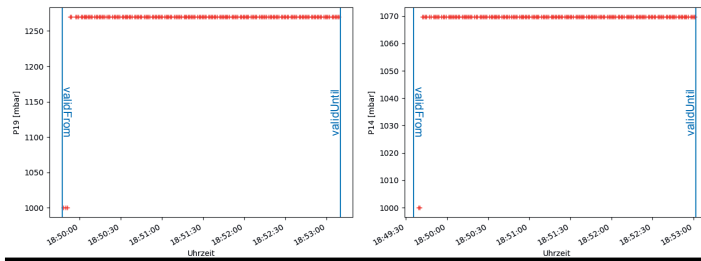


Abbildung 8.5: Ergebnis des zweiten Querys aus Listing 8.6 für die Sensoren P13 und P19

8.2.5 Weitere Beziehungstypen

Die Abbildungen 8.6 bis 8.9 zeigen Screenshots von verschiedenen Ausschnitten aus dem Nachbarschaftsgraphen. Jeder Ausschnitt zeigt die Instanzen eines bestimmten Beziehungstypen, die im Rahmen der prototypischen Umsetzung des Konzepts modelliert wurden. Diese könnten zusätzlich zu den Beziehungen der zuvor genannten Kontextbeispielen erkundet werden.

Der Ausschnitt in Abbildung 8.6 zeigt die zu dem in Abbildung 8.3 dargestellten Beispiel zugehörigen Belegungsvorgänge. Wie in der Abbildung dargestellt, hat die Entität vom Typ `Operator` die Entitäten vom Typ `singleControlUnit` erst belegt und anschließend wieder frei gegeben. Die Belegungsvorgänge werden durch die Ereignisse `occupied` und die Freigabevorgänge durch die Ereignisse `freed` repräsentiert.

In Abbildung 8.7 sind die modellierten Abhängigkeiten zwischen den verschiedenen Steuerungskomponenten (`singleControlUnit`, `sensorAbstraction`) und dem Rechenknoten (`computingNode`), auf dem diese ausgeführt werden, dargestellt. In den entsprechenden Beziehungen übernimmt der Rechenknoten die Rolle des `resourceProviders` und die Steuerungskomponenten die Rolle `resourceConsumer`.

Die sich aus der Druckluftversorgung der Ventile ergebenden Beziehungen sind in Abbildung 8.8 dargestellt.

Abbildung 8.9 stellt die Beziehungen dar, die sich aus der Verriegelungslogik ergeben. Dabei bestehen verschiedene Abhängigkeiten zwischen den Einzelsteuereinheiten und den Sensoren.

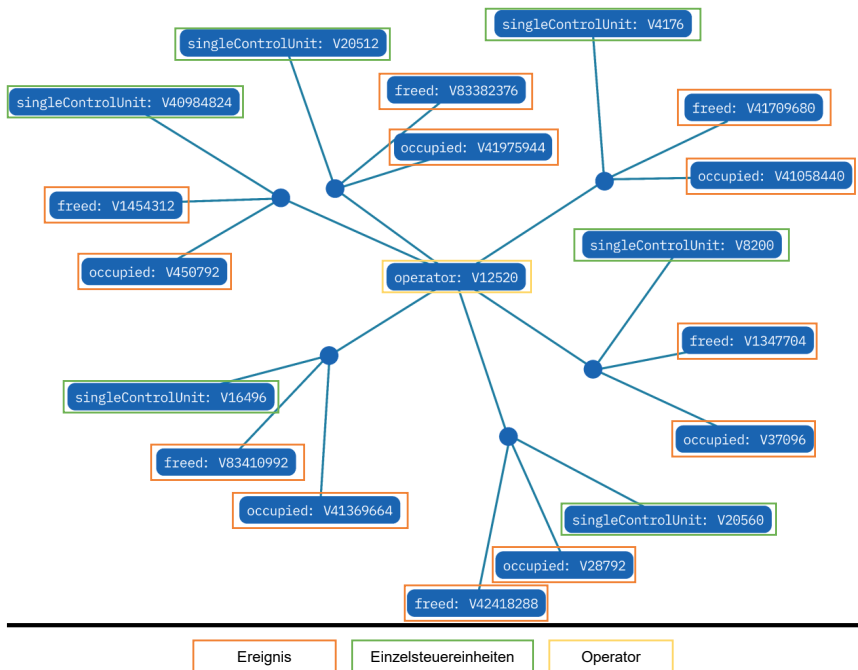


Abbildung 8.6: Belegungsmechanismus - Ausschnitt aus dem Nachbarschaftsgraphen

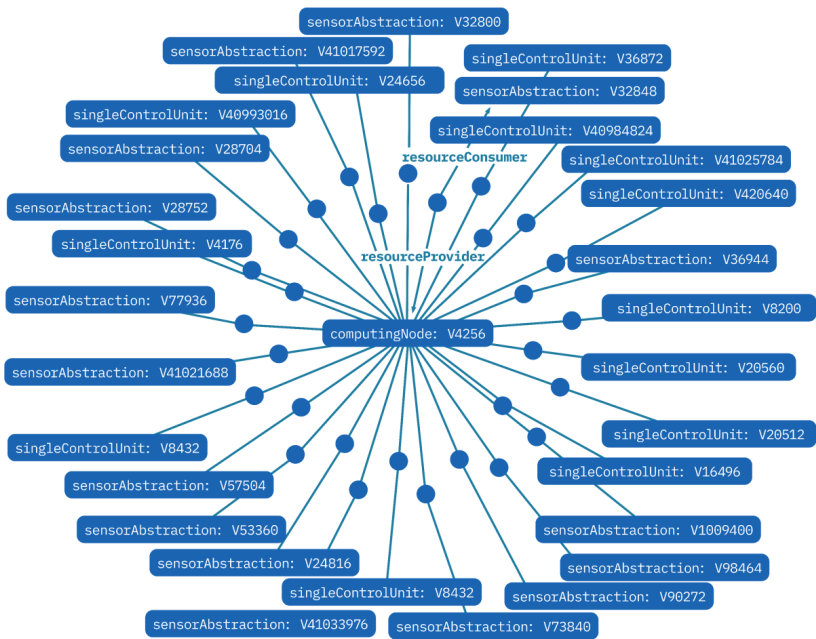


Abbildung 8.7: Deployment - Ausschnitt aus dem Nachbarschaftsgraphen

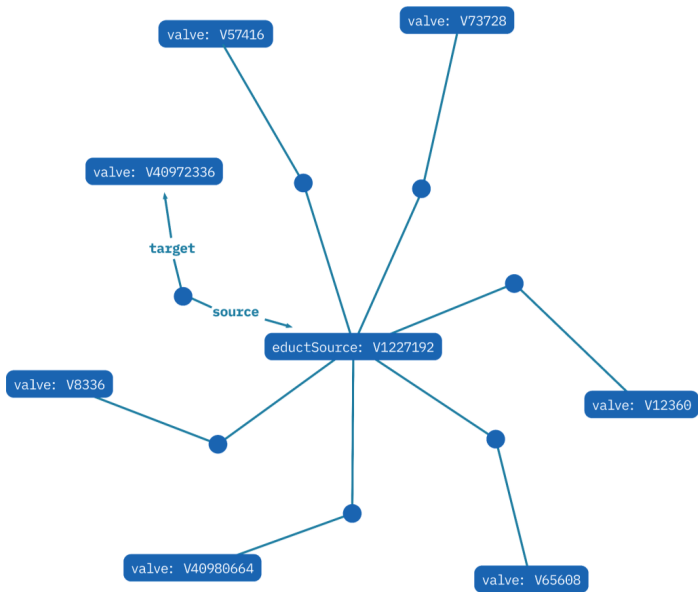


Abbildung 8.8: Energieversorgung - Ausschnitt aus dem Nachbarschaftsgraphen

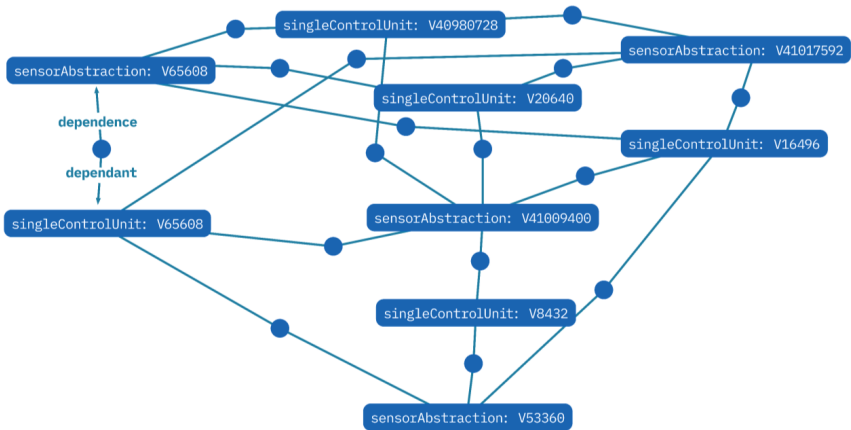


Abbildung 8.9: Steuerungsabhängigkeiten - Ausschnitt aus dem Nachbarschaftsgraphen

9 Diskussion und Ausblick

Während des Betriebs einer prozesstechnischen Anlage fallen Unmengen an unterschiedlichen Daten in den verschiedensten Systemen an. Moderne Methoden der Datenanalyse bieten ganz neue Nutzungsmöglichkeiten dieser Daten. Grundvoraussetzung für diese Möglichkeiten ist eine Verknüpfung der bestehenden Daten aus den unterschiedlichsten Informationsquellen zu einem nutzbaren Wissensnetz. Diese Verknüpfung ist mit großem Aufwand verbunden. An dieser Stelle setzt die vorliegende Arbeit an. Sie dokumentiert ein Konzept zur Kontextualisierung von Prozessdaten in der Prozessindustrie. Kern des Konzepts ist der sogenannte Nachbarschaftsgraph, der die Beziehungen der Prozessdaten aufzeigt und damit eine Möglichkeit zur Erkundung des Kontexts bietet.

Als Voraussetzung für die erfolgreiche Umsetzung des Konzepts wurden eine durchgängige Digitalisierung, ein eindeutiges Namenskonzept, die Aktualität der Informationsquellen und Zeitsynchronität identifiziert (siehe Abschnitt 5.4). Im Rahmen dieser Arbeit wurden diese Voraussetzungen für die in der prototypischen Implementierung betrachteten Anlage geschaffen. Allerdings ist davon auszugehen, dass die wenigsten prozesstechnischen Anlagen diese Voraussetzungen aktuell vollumfänglich erfüllen. Gleichwohl zeigt die Dynamik der Entwicklung von Konzepten wie der Verwaltungsschale und die Diskussion um die Umsetzung von Industrie 4.0, dass mit einer Erfüllung dieser Voraussetzungen in naher Zukunft zu rechnen ist. Aus diesem Grund scheint die Annahme dieser Voraussetzungen realistisch.

Grundlage des Nachbarschaftsgraphen ist das in Kapitel 6.3 entwickelte Beziehungsmodell, das auf dem ER-Systemmodell und dem konzeptuellen Datenmodell der ISO 15926-2 aufbaut. Dieses Modell ermöglicht die Integration von Lebenszyklusinformationen in ein ER-Systemmodell und erlaubt so die Modellierung von zeitlich begrenzten Beziehungen zwischen einzelnen Entitäten. Dabei ergibt sich der Lebenszyklus einer Entität aus den unterschiedlichen Rollen, die die Entität in den entsprechenden Beziehungen einnimmt. Das Modell besticht durch seine Einfachheit und ist dadurch leicht zu verstehen und einfach anzuwenden. Dank des Beziehungsmodells ist das Konzept in der Lage, klassifizierte und gerichtete Beziehungen zu einem bestimmten Zeitpunkt und damit die Stufe IV der in Abbildung 3.9 vorgestellten Abstufung der Kontextualisierung abzubilden.

Aufbauend auf dem Beziehungsmodell wurde im Rahmen dieser Arbeit eine Modellhierarchie entwickelt, die aus einem Metametamodell und einer Reihe von Aspektmetamodellen besteht. Das Metametamodell schränkt das Beziehungsmodell weiter ein und dient so als Grundlage für eine konkrete Implementierung. Die Aspektmetamodelle sind Abbildungen bestimmter Begriffscluster aus der Domäne der Prozessindustrie auf das Metametamodell. Hier dienen dem Konzept verschiedene bestehende Informationsquellen bzw. Informationsmodelle als Grundlage. Auf diesem Wege wurde sichergestellt, dass die entwickelte Modellhierarchie eine ausreichende Ausdrucksstärke besitzt und wesentliche Begriffe der Prozessindustrie abbilden kann. Auch ist klar, dass eine vollständige Berücksichtigung aller möglicherweise notwendigen Begriffe nicht möglich ist. Aus diesem Grund wurde bei der Definition der Aspektmetamodelle ein modularer Ansatz gewählt und eine Prozedur zur

Erweiterung beschrieben.

In Abschnitt 8 wurde anhand verschiedener Anwendungsszenarien aufgezeigt, welche neuen Informationen durch die Vernetzung der Daten zugänglich gemacht werden konnten. Dank der nun explizit modellierten Beziehungen zwischen den Daten aus den unterschiedlichen Quellen ist eine einfache und effiziente Erkundung dieser möglich.

Zusammenfassend bietet das Konzept dieser Arbeit folgende drei Vorteile:

1. Kern der Arbeit ist ein einfaches, leicht zu verstehendes und allgemeingültiges Beziehungsmodell, das die Verknüpfung verschiedener Entitäten unter der Berücksichtigung zeitlicher Aspekte ermöglicht und so Zusammenhänge zwischen Anlagenstruktur, Zeitreihen und Ereignissen erkundbar macht.
2. Die Arbeit beschreibt eine klare Vorgehensweise zur Umsetzung des entwickelten Konzepts. Dies ist insbesondere aufgrund der hohen Individualität der einzelnen prozesstechnischen Anlagen relevant. Je nach Unternehmen, Anlage oder Standort werden unterschiedliche Werkzeuge mit verschiedenen Informationsmodellen eingesetzt. Diese Diversität erschwert die Definition einer allgemeingültigen Lösung. Allerdings ist aufgrund der beschriebenen Vorgehensweise, des modularen Ansatzes und der Einfachheit des Beziehungsmodells eine Adaption des Konzeptes an eine konkrete prozesstechnische Anlage als realistisch einzustufen.
3. Die im Rahmen dieser Arbeit erfolgte Implementierung zeigt, dass eine praktische Umsetzung des vorgestellten Konzepts aufgrund der vollständigen Modellbasierung realistisch und mit begrenztem Aufwand möglich ist. Es müssen ausschließlich entsprechende Modelltransformationen für die bestehenden Informationsmodelle entwickelt werden. Die Implementierung zeigt, wie Beziehungen zur Laufzeit erfasst werden können, damit der Kontext für spätere Analysen verfügbar ist und nicht aufwändig im Nachhinein rekonstruiert werden muss.

Eine große Herausforderung bei der Modellierung einer prozesstechnischen Anlage mithilfe des vorgestellten Konzepts stellt die Unterscheidung zwischen Änderungen durch Ereignisse und Änderungen durch kontinuierliche Eigenschaftsaussagen (siehe Abschnitt 6.4.9) dar. An dieser Stelle ist es nicht möglich eine einheitliche Lösung für diese Fragestellung zu geben. Vielmehr handelt es sich dabei um eine Designentscheidung, die für jede Anlage individuell getroffen werden muss. Beispielsweise kann das Verändern einer Ventilposition sowohl ein Ereignis als auch eine Eigenschaftsaussage darstellen. Anhaltspunkt für die Art und Weise der Modellierung kann hier die Häufigkeit der Änderung geben. Wird die Ventilposition in unregelmäßigen Abständen durch das Bedienpersonal verändert, bietet sich die Modellierung als Ereignis an. Wird die Position dagegen durch einen geschlossenen Regelkreis in regelmäßigen Abständen vorgegeben, scheint die Modellierung als kontinuierliche Eigenschaftsaussage zweckmäßig.

Die prototypische Implementierung dieser Arbeit kopiert alle Daten aus den unterschiedlichen Datenquellen in die Graphdatenbank. Dadurch sind anschließend alle Daten an zwei Stellen verfügbar. Dies hat einen unnötigen Ressourcenverbrauch und eine doppelte Datenhaltung zur Folge. Eine Lösung könnte hier sein, dass die Daten nicht in die Graphdatenbank kopiert, sondern verschoben werden. Im Anschluss werden alle Daten ausschließlich in der Graphdatenbank verwaltet. Alternativ könnte ein dezentraler Ansatz verfolgt werden. Ein Konzept dazu beschreiben Kampert et. al. in [118]. Sie entwerfen ein Konzept zur Integration der Außenperspektive in das innere einer Systemmodellierung, um so

das System zu befähigen, die eigenen Beziehungen mit anderen Systemen zu erkunden. Grundsätzlich muss das vorgestellte Konzept in eine ganzheitliche Datenhaltungsstruktur einer prozesstechnischen Anlage integriert werden. Für diese Integration des Konzepts in eine übergeordnete Struktur kann die Abbildung der entwickelten Modellhierarchie auf eine bestehende Ontologie (z. B. ISO 15926-2), das OPC UA-Metamodell oder die Modellierung mithilfe von OWL hilfreich sein. Allerdings ist dies stark von der Implementierung der übergeordneten Struktur abhängig, weil diese die entsprechende Ontologie unterstützen muss. Da zum jetzigen Zeitpunkt eine mögliche übergeordnete Struktur unbekannt ist, wurde im Rahmen dieser Arbeit keine entsprechende Abbildung vorgenommen.

Zur Umsetzung weiterer Anwendungsszenarien sollten in Zukunft weitere Aspektmetamodelle entwickelt werden. So könnte zum Beispiel für das in [6] beschriebene Anwendungsgebiete des Online-Energiemanagements ein Aspektmetamodell zur Beschreibung von Bilanzen erstellt werden. Auch könnten Aspektmetamodelle zur Integration von Informationen aus ERP- oder MES-Systemen in den Nachbarschaftsgraphen entwickelt werden.

Ein weiterer offener Punkt ist die Entwicklung einer Benutzerschnittstelle, die auch nicht datenbankerfahrenen Nutzenden die einfache Interaktion mit dem modellierten Kontext ermöglicht. Eine Möglichkeit hierzu stellt das in Abschnitt 8 vorgestellte Assistenzsystem dar.

Eine weitere Herausforderung, die im Rahmen dieser Arbeit nicht betrachtet wurde, stellt die Versionsverwaltung und Änderungsdetektion in den verschiedenen Datenquellen dar. Ändert sich zum Beispiel die Anlagen- oder Steuerungsstruktur, muss dies in dem Nachbarschaftsgraphen widergespiegelt werden. Dazu wird eine entsprechende Versionsverwaltung und Änderungsdetektion benötigt. Es ist davon auszugehen, dass diese Funktionalitäten in Zukunft zum Beispiel von der Verwaltungsschale bereitgestellt werden.

In Zukunft sollte untersucht werden, wie Gewichtung der Beziehungen erfolgen und damit eine Modellierung der Stufe V der in Abbildung 3.9 dargestellten Kontextabstufung erreicht werden kann.

A Anhang

A.1 Pandix-Datei des Pumpwerks

Listing A.1: PandIX-CEAX-Datei des Pumpwerks

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <CAEXFile xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:
3   noNamespaceSchemaLocation="./CAEX_ClassModel_V2.15.xsd" xmlns:xs="http://www.w3.org
4   /2001/XMLSchema" xmlns:pix="http://www.plt.rwth-aachen.de/perma/PandIX_FP"
5   SchemaVersion="2.15" FileName="Pumpwerk.xml">
6   <!-- General File Information -->
7   <Version>6.0</Version>
8   <Copyright>Copyright (C) 2014 Lehrstuhl fuer Prozessleittechnik, RWTH Aachen</
9   Copyright>
10  <!-- External References -->
11  <ExternalReference Alias="NE150" Path="./PLT-Stelle_Version_2.aml"/>
12  <ExternalReference Alias="PandIX_ICL" Path="./PandIX_ICL_V600.xml"/>
13  <ExternalReference Alias="PandIX_PPE_RC" Path="./PandIX_PPE-BasicLib_V600.xml"/>
14  <ExternalReference Alias="PandIX_PPE_SUC" Path="./PandIX_PPE_SUC_V600.xml"/>
15  <ExternalReference Alias="PandIX_PME" Path="./PandIX_PME_V600.xml"/>
16  <ExternalReference Alias="PandIX_PCE" Path="./PandIX_PCE_V600.xml"/>
17  <!-- Instance Hierarchy -->
18  <InstanceHierarchy Name="ACPLT">
19    <InternalElement Name="Pumpwerk" RefBaseSystemUnitPath="PandIX_PME@PandIX_PME/
20      PlantItemClass">
21      <!-- Instance "Pumpwerk" consists of three subplants -->
22      <InternalElement Name="TU10" RefBaseSystemUnitPath="TeilanlagenPumpwerk/TU10"/>
23      <InternalElement Name="TU20" RefBaseSystemUnitPath="TeilanlagenPumpwerk/TU20"/>
24      <InternalElement Name="TU30" RefBaseSystemUnitPath="TeilanlagenPumpwerk/TU30"/>
25      <!-- Connection of the three subplants -->
26      <InternalLink Name="TU2ONE2_TU1ONE1" RefPartnerSideA="TU20:NE2" RefPartnerSideB="
27        TU10:NE1"/>
28      <InternalLink Name="TU3ONE2_TU1ONE2" RefPartnerSideA="TU30:NE2" RefPartnerSideB="
29        TU10:NE2"/>
30      <InternalLink Name="TU1ONE3_TU2ONE1" RefPartnerSideA="TU10:NE3" RefPartnerSideB="
31        TU20:NE1"/>
32      <InternalLink Name="TU1ONE4_TU3ONE1" RefPartnerSideA="TU10:NE4" RefPartnerSideB="
33        TU30:NE1"/>
34    </InternalElement>
35  </InstanceHierarchy>
36  <!-- SystemUnitClassLib -->
37  <SystemUnitClassLib Name="TeilanlagenPumpwerk">
38    <!-- TU10 -->
39    <SystemUnitClass Name="TU10" RefBaseClassPath="PandIX_PME@PandIX_PME/PlantItemClass"
40    >
41      <ExternalInterface Name="NE1" RefBaseClassPath="PandIX_ICL@PandIX_ICL/
42        ProductConnectionPoint"/>
43    </SystemUnitClass>
44  </SystemUnitClassLib>
45 </CAEXFile>
```

```

<ExternalInterface Name="NE2" RefBaseClassPath="PandIX_ICL@PandIX_ICL/
  ProductConnectionPoint"/>
33 <ExternalInterface Name="NE3" RefBaseClassPath="PandIX_ICL@PandIX_ICL/
  ProductConnectionPoint"/>
<ExternalInterface Name="NE4" RefBaseClassPath="PandIX_ICL@PandIX_ICL/
  ProductConnectionPoint"/>
35 <!-- Behaelter Form: liegend, zylindrisch -->
<InternalElement Name="Vessel1">
37   <Attribute Name="Position">
     <Value>S01P2:1.30</Value>
39   </Attribute>
     <Attribute Name="NominalVolume">
41     <Value>0.157</Value>
     </Attribute>
     <Attribute Name="NominalFillelevel" AttributeDataType="pix:Position" Unit="-">
43       <Value>S01P2:0.50</Value>
     </Attribute>
     <Attribute Name="ZeroLevel">
45       <Value>0.0</Value>
     </Attribute>
     <Attribute Name="RestVolume">
47       <Value>0.0</Value>
     </Attribute>
     <Attribute Name="TotalVolume">
49       <Value>0.0</Value>
     </Attribute>
     <Attribute Name="Form">
51       <Value>S01P2:(0.0;0.0);(0.05;0.01869);(0.10;0.05204);(0.15;0.09406)
           ;(0.20;0.14238);(0.25;0.19550);(0.30;0.25232);(0.35;0.31192)
           ;(0.40;0.37353);(0.45;0.43644);(0.50;0.50000);(0.55;0.56356)
           ;(0.60;0.62647);(0.65;0.68808);(0.70;0.74768);(0.75;0.80450)
           ;(0.80;0.85762);(0.85;0.90594);(0.90;0.94796);(0.95;0.98131)
           ;(1.00;1.00000)</Value>
53     </Attribute>
55   </Attribute>
     <ExternalInterface Name="PIn" RefBaseClassPath="PandIX_ICL@PandIX_ICL/
       ProductConnectionPoint">
57       <Attribute Name="Position">
           <Value>S01P2:0.40</Value>
59       </Attribute>
       <Attribute Name="Position">
61       <Value>S01P5:ul;rl</Value>
63       </Attribute>
65     </ExternalInterface>
     <ExternalInterface Name="POut1" RefBaseClassPath="PandIX_ICL@PandIX_ICL/
       ProductConnectionPoint">
67       <Attribute Name="Position">
           <Value>S01P2:0.00</Value>
69       </Attribute>
       <Attribute Name="Position">
71       <Value>S01P5:dw;rc</Value>
73       </Attribute>
     </ExternalInterface>
     <ExternalInterface Name="POut2" RefBaseClassPath="PandIX_ICL@PandIX_ICL/
       ProductConnectionPoint">
75       <Attribute Name="Position">
           <Value>S01P2:0.00</Value>

```



```

77     </Attribute>
    <Attribute Name="Position">
79     <Value>S01P5:dw;rc</Value>
    </Attribute>
81 </ExternalInterface>
    <ExternalInterface Name="L10" RefBaseClassPath="PandIX_ICL@PandIX_ICL/
      MeasurementPoint">
83     <Attribute Name="Position">
      <Value>S01P5:dw;rc</Value>
85     </Attribute>
    </ExternalInterface>
87 <ExternalInterface Name="Q11" RefBaseClassPath="PandIX_ICL@PandIX_ICL/
      MeasurementPoint">
    <Attribute Name="Position">
89     <Value>S01P5:dl;rl</Value>
    </Attribute>
91 </ExternalInterface>
    <ExternalInterface Name="T12" RefBaseClassPath="PandIX_ICL@PandIX_ICL/
      MeasurementPoint">
93     <Attribute Name="Position">
      <Value>S01P1:Q11</Value>
95     </Attribute>
    </ExternalInterface>
97 <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/
      LVesselRequest"/>
</InternalElement>
99 <!-- Apparate -->
    <InternalElement Name="Pump13">
101     <Attribute Name="Position">
      <Value>S01P2:0.40</Value>
103     </Attribute>
      <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/PumpRequest"
        />
105 </InternalElement>
    <InternalElement Name="Pump18">
107     <Attribute Name="Position">
      <Value>S01P2:0.50</Value>
109     </Attribute>
      <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/PumpRequest"
        />
111 </InternalElement>
    <InternalElement Name="Heater43" RefBaseSystemUnitPath="
      PandIX_PPE_SUC@PPE_SystemUnitLib/HeatExchanger">
113     <Attribute Name="Position">
      <Value>S01P2:1.55</Value>
115     </Attribute>
    </InternalElement>
117 <InternalElement Name="Valve16">
    <Attribute Name="Position">
119     <Value>S01P2:1.55</Value>
    </Attribute>
121 <Attribute Name="SafeState">
      <Value>open</Value>
123 </Attribute>
    <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/ValveRequest"
      />

```

```

125     </InternalElement>
126     <InternalElement Name="Valve21">
127         <Attribute Name="Position">
128             <Value>S01P2:1.55</Value>
129         </Attribute>
130         <Attribute Name="SafeState">
131             <Value>open</Value>
132         </Attribute>
133         <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/ValveRequest"
134             />
135     </InternalElement>
136     <InternalElement Name="Valve24">
137         <Attribute Name="Position">
138             <Value>S01P2:1.30</Value>
139         </Attribute>
140         <Attribute Name="SafeState">
141             <Value>close</Value>
142         </Attribute>
143         <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/ValveRequest"
144             />
145     </InternalElement>
146     <InternalElement Name="Valve25">
147         <Attribute Name="Position">
148             <Value>S01P2:1.30</Value>
149         </Attribute>
150         <Attribute Name="SafeState">
151             <Value>close</Value>
152         </Attribute>
153         <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/ValveRequest"
154             />
155     </InternalElement>
156     <!-- Rohrleitungen und Rohrleitungsverzweigungen -->
157     <InternalElement Name="pipe101">
158         <Attribute Name="DN">
159             <Value>25</Value>
160         </Attribute>
161         <Attribute Name="Form">
162             <Value>S01F1:(0.50;0.50);(0.10;0)</Value>
163         </Attribute>
164         <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/PipeRequest"
165             />
166     </InternalElement>
167     <InternalElement Name="pipe111">
168         <Attribute Name="DN">
169             <Value>50</Value>
170         </Attribute>
171         <Attribute Name="Form">
172             <Value>S01F1:(0.80;-0.80);(0.20;0)</Value>
173         </Attribute>
174         <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/PipeRequest"
175             />
176     </InternalElement>
177     <InternalElement Name="pipe112">
178         <Attribute Name="DN">
179             <Value>25</Value>
180         </Attribute>

```

```

177     <Attribute Name="Form">
        <Value>S01F1:(0.35;0.35);(1.50;0);(0.55;0.40);(0.10;0)</Value>
    </Attribute>
179    <ExternalInterface Name="P19" RefBaseClassPath="PandIX_ICL@PandIX_ICL/
        MeasurementPoint">
        <Attribute Name="Position">
            <Value>S01P4:0.91</Value>
        </Attribute>
183    </ExternalInterface>
    <ExternalInterface Name="T20" RefBaseClassPath="PandIX_ICL@PandIX_ICL/
        MeasurementPoint">
        <Attribute Name="Position">
            <Value>S01P4:1.66</Value>
        </Attribute>
187    </ExternalInterface>
    <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/PipeRequest"
        />
</InternalElement>
191 <InternalElement Name="pipe113">
    <Attribute Name="DN">
        <Value>25</Value>
    </Attribute>
195    <Attribute Name="Form">
        <Value>S01F1:(0.80;0)</Value>
197    </Attribute>
    <ExternalInterface Name="F22" RefBaseClassPath="PandIX_ICL@PandIX_ICL/
        MeasurementPoint">
        <Attribute Name="Position">
            <Value>S01P4:0.40</Value>
        </Attribute>
201    </ExternalInterface>
    <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/PipeRequest"
        />
</InternalElement>
205 <InternalElement Name="pipe121">
    <Attribute Name="DN">
        <Value>40</Value>
    </Attribute>
207    <Attribute Name="Form">
        <Value>S01F1:(0.90;-0.90);(0.10;0)</Value>
209    </Attribute>
    <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/PipeRequest"
        />
211 </InternalElement>
213 <InternalElement Name="pipe122">
    <Attribute Name="DN">
        <Value>25</Value>
    </Attribute>
217    <Attribute Name="Form">
        <Value>S01F1:(0.10;0);(0.25;0);(0.40;0);(0.55;0.55);(0.50;0);(0.60;0.60)</
            Value>
        </Attribute>
219    <ExternalInterface Name="P14" RefBaseClassPath="PandIX_ICL@PandIX_ICL/
        MeasurementPoint">
        <Attribute Name="Position">
            <Value>S01P4:0.55</Value>
223    </Attribute>

```

```

    </Attribute>
225 </ExternalInterface>
    <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/PipeRequest"
    />
227 </InternalElement>
<InternalElement Name="pipe123">
229   <Attribute Name="DN">
    <Value>25</Value>
231   </Attribute>
    <Attribute Name="Form">
233     <Value>S01F1:(0.30;0.-0.30);(0.40;0);(0.35;0.25);(0.40;0)</Value>
    </Attribute>
235   <ExternalInterface Name="T15" RefBaseClassPath="PandIX_ICL@PandIX_ICL/
    MeasurementPoint">
    <Attribute Name="Position">
237       <Value>S01P4:0.45</Value>
    </Attribute>
239   </ExternalInterface>
    <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/PipeRequest"
    />
241 </InternalElement>
<InternalElement Name="pipe124">
243   <Attribute Name="DN">
    <Value>25</Value>
245   </Attribute>
    <Attribute Name="Form">
247     <Value>S01F1:(0.55;0)</Value>
    </Attribute>
249   <ExternalInterface Name="F17" RefBaseClassPath="PandIX_ICL@PandIX_ICL/
    MeasurementPoint">
    <Attribute Name="Position">
251       <Value>S01P4:0.30</Value>
    </Attribute>
253   </ExternalInterface>
    <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/PipeRequest"
    />
255 </InternalElement>
<InternalElement Name="pipe131">
257   <Attribute Name="DN">
    <Value>25</Value>
259   </Attribute>
    <Attribute Name="Form">
261     <Value>S01F1:(0.80;0);(0.25;-0.25);(0.25;0)</Value>
    </Attribute>
263   <ExternalInterface Name="T23" RefBaseClassPath="PandIX_ICL@PandIX_ICL/
    MeasurementPoint">
    <Attribute Name="Position">
265       <Value>S01P4:0.60</Value>
    </Attribute>
267   </ExternalInterface>
    <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/PipeRequest"
    />
269 </InternalElement>
<InternalElement Name="pipe151">
271   <Attribute Name="DN">
    <Value>25</Value>

```

```

273     </Attribute>
274     <Attribute Name="Form">
275       <Value>S01F1:(0.10;0);(0.15;0)</Value>
276     </Attribute>
277     <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/PipeRequest"
      />
    </InternalElement>
279    <InternalElement Name="pipe152">
      <Attribute Name="DN">
281        <Value>25</Value>
      </Attribute>
283      <Attribute Name="Form">
        <Value>S01F1:(0.15;0);(0.55;-0.50)</Value>
285      </Attribute>
      <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/PipeRequest"
        />
    </InternalElement>
287    <InternalElement Name="pipe161">
      <Attribute Name="DN">
289        <Value>25</Value>
      </Attribute>
291      <Attribute Name="Form">
        <Value>S01F1:(0.10;0);(0.15;0)</Value>
293      </Attribute>
      <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/PipeRequest"
        />
    </InternalElement>
297    <InternalElement Name="pipe162">
      <Attribute Name="DN">
299        <Value>25</Value>
      </Attribute>
301      <Attribute Name="Form">
        <Value>S01F1:(1.30;0);(0.55;-0.80)</Value>
303      </Attribute>
      <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/PipeRequest"
        />
    </InternalElement>
305    <InternalElement Name="jun100">
      <Attribute Name="Position">
307        <Value>S01P2:1.20</Value>
309      </Attribute>
      <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/
        PipeJunctionRequest"/>
    </InternalElement>
311    <InternalElement Name="jun130">
      <Attribute Name="Position">
313        <Value>S01P2:1.55</Value>
315      </Attribute>
      <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/
        PipeJunctionRequest"/>
    </InternalElement>
317    <InternalElement Name="jun140">
      <Attribute Name="Position">
319        <Value>S01P2:1.30</Value>
321      </Attribute>
      <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/

```

```
        PipeJunctionRequest"/>
323    </InternalElement>
    <!-- PLT-Stellen -->
325    <InternalElement Name="L10" RefBaseSystemUnitPath="PandIX_PCE@PandIX_PCE/
        SensorRequest">
        <Attribute Name="PCEInitial">
            <Value>L</Value>
        </Attribute>
327        <ExternalInterface Name="VAL">
            <Attribute Name="SoftwareSignalType">
                <Value>I</Value>
            </Attribute>
329        </ExternalInterface>
    </InternalElement>
335    <InternalElement Name="Q11" RefBaseSystemUnitPath="PandIX_PCE@PandIX_PCE/
        SensorRequest">
        <Attribute Name="PCEInitial">
            <Value>Q</Value>
        </Attribute>
337        <ExternalInterface Name="VAL">
            <Attribute Name="SoftwareSignalType">
                <Value>I</Value>
            </Attribute>
339        </ExternalInterface>
    </InternalElement>
343    <InternalElement Name="T12" RefBaseSystemUnitPath="PandIX_PCE@PandIX_PCE/
        SensorRequest">
        <Attribute Name="PCEInitial">
            <Value>T</Value>
        </Attribute>
347        <ExternalInterface Name="VAL">
            <Attribute Name="SoftwareSignalType">
                <Value>I</Value>
            </Attribute>
349        </ExternalInterface>
    </InternalElement>
353    <InternalElement Name="N13" RefBaseSystemUnitPath="PandIX_PCE@PandIX_PCE/
        ActuatorRequest">
        <Attribute Name="PCEInitial">
            <Value>N</Value>
        </Attribute>
357        <ExternalInterface Name="W">
            <Attribute Name="SoftwareSignalType">
                <Value>I</Value>
            </Attribute>
359        </ExternalInterface>
    </InternalElement>
363    <InternalElement Name="P14" RefBaseSystemUnitPath="PandIX_PCE@PandIX_PCE/
        SensorRequest">
        <Attribute Name="PCEInitial">
            <Value>P</Value>
        </Attribute>
367        <ExternalInterface Name="VAL">
            <Attribute Name="SoftwareSignalType">
                <Value>I</Value>
            </Attribute>
369        </ExternalInterface>
    </InternalElement>
371    </InternalElement>
```

```

373     </ExternalInterface>
</InternalElement>
375 <InternalElement Name="T15" RefBaseSystemUnitPath="PandIX_PCE@PandIX_PCE/
    SensorRequest">
    <Attribute Name="PCEInitial">
377       <Value>T</Value>
    </Attribute>
379    <ExternalInterface Name="VAL">
    <Attribute Name="SoftwareSignalType">
381       <Value>I</Value>
    </Attribute>
383    </ExternalInterface>
</InternalElement>
385 <InternalElement Name="Y16" RefBaseSystemUnitPath="PandIX_PCE@PandIX_PCE/
    ActuatorRequest">
    <Attribute Name="PCEInitial">
387       <Value>Y</Value>
    </Attribute>
389    <ExternalInterface Name="W">
    <Attribute Name="SoftwareSignalType">
391       <Value>I</Value>
    </Attribute>
393    </ExternalInterface>
</InternalElement>
395 <InternalElement Name="F17" RefBaseSystemUnitPath="PandIX_PCE@PandIX_PCE/
    SensorRequest">
    <Attribute Name="PCEInitial">
397       <Value>F</Value>
    </Attribute>
399    <ExternalInterface Name="VAL">
    <Attribute Name="SoftwareSignalType">
401       <Value>I</Value>
    </Attribute>
403    </ExternalInterface>
</InternalElement>
405 <InternalElement Name="N18" RefBaseSystemUnitPath="PandIX_PCE@PandIX_PCE/
    ActuatorRequest">
    <Attribute Name="PCEInitial">
407       <Value>N</Value>
    </Attribute>
409    <ExternalInterface Name="W">
    <Attribute Name="SoftwareSignalType">
411       <Value>O</Value>
    </Attribute>
413    </ExternalInterface>
</InternalElement>
415 <InternalElement Name="P19" RefBaseSystemUnitPath="PandIX_PCE@PandIX_PCE/
    SensorRequest">
    <Attribute Name="PCEInitial">
417       <Value>P</Value>
    </Attribute>
419    <ExternalInterface Name="VAL">
    <Attribute Name="SoftwareSignalType">
421       <Value>I</Value>
    </Attribute>
423    </ExternalInterface>

```

```
</InternalElement>
425 <InternalElement Name="T20" RefBaseSystemUnitPath="PandIX_PCE@PandIX_PCE/
    SensorRequest">
    <Attribute Name="PCEInitial">
427       <Value>T</Value>
    </Attribute>
429    <ExternalInterface Name="VAL">
        <Attribute Name="SoftwareSignalType">
431             <Value>I</Value>
        </Attribute>
433    </ExternalInterface>
</InternalElement>
435 <InternalElement Name="Y21" RefBaseSystemUnitPath="PandIX_PCE@PandIX_PCE/
    ActuatorRequest">
    <Attribute Name="PCEInitial">
437       <Value>Y</Value>
    </Attribute>
439    <ExternalInterface Name="W">
        <Attribute Name="SoftwareSignalType">
441             <Value>I</Value>
        </Attribute>
443    </ExternalInterface>
</InternalElement>
445 <InternalElement Name="F22" RefBaseSystemUnitPath="PandIX_PCE@PandIX_PCE/
    SensorRequest">
    <Attribute Name="PCEInitial">
447       <Value>F</Value>
    </Attribute>
449    <ExternalInterface Name="VAL">
        <Attribute Name="SoftwareSignalType">
451             <Value>I</Value>
        </Attribute>
453    </ExternalInterface>
</InternalElement>
455 <InternalElement Name="T23" RefBaseSystemUnitPath="PandIX_PCE@PandIX_PCE/
    SensorRequest">
    <Attribute Name="PCEInitial">
457       <Value>T</Value>
    </Attribute>
459    <ExternalInterface Name="VAL">
        <Attribute Name="SoftwareSignalType">
461             <Value>I</Value>
        </Attribute>
463    </ExternalInterface>
    <InternalElement Name="S+" RefBaseSystemUnitPath="NE150@NE150SUCLibrary/Switch">
465         <Attribute Name="LimitValueType">
            <Value>+</Value>
467         </Attribute>
        <ExternalInterface Name="S+" RefBaseClassPath="NE150@NE150InterfaceLibrary/
            SWSignal"/>
469    </InternalElement>
</InternalElement>
471 <InternalElement Name="Y24" RefBaseSystemUnitPath="PandIX_PCE@PandIX_PCE/
    ActuatorRequest">
    <Attribute Name="PCEInitial">
473       <Value>Y</Value>
```



```

475     </Attribute>
    <ExternalInterface Name="W">
      <Attribute Name="SoftwareSignalType">
477        <Value>0</Value>
      </Attribute>
    </ExternalInterface>
  </InternalElement>
481  <InternalElement Name="Y25" RefBaseSystemUnitPath="PandIX_PCE@PandIX_PCE/
    ActuatorRequest">
    <Attribute Name="PCEInitial">
483      <Value>Y</Value>
    </Attribute>
    <ExternalInterface Name="W">
      <Attribute Name="SoftwareSignalType">
487        <Value>0</Value>
      </Attribute>
    </ExternalInterface>
  </InternalElement>
491  <!-- Links -->
  <InternalLink Name="NE3_jun100" RefPartnerSideA=":NE3" RefPartnerSideB="jun100:P1"
    />
493  <InternalLink Name="NE4_jun100" RefPartnerSideA=":NE4" RefPartnerSideB="jun100:P2"
    />
  <InternalLink Name="jun100_pipe101" RefPartnerSideA="jun100:P3" RefPartnerSideB="
    pipe101:PIn"/>
495  <InternalLink Name="pipe101_Vessel1" RefPartnerSideA="pipe101:P0ut"
    RefPartnerSideB="Vessel1:PIn"/>
  <InternalLink Name="Vessel1_L10" RefPartnerSideA="Vessel1:L10" RefPartnerSideB="
    L10:P1"/>
497  <InternalLink Name="Vessel1_Q11" RefPartnerSideA="Vessel1:Q11" RefPartnerSideB="
    Q11:P1"/>
  <InternalLink Name="Vessel1_T12" RefPartnerSideA="Vessel1:T12" RefPartnerSideB="
    T12:P1"/>
499  <InternalLink Name="Vessel1_pipe111" RefPartnerSideA="Vessel1:P0ut1"
    RefPartnerSideB="pipe111:PIn"/>
  <InternalLink Name="Vessel1_pipe121" RefPartnerSideA="Vessel1:P0ut2"
    RefPartnerSideB="pipe121:PIn"/>
501  <InternalLink Name="pipe111_Pump18" RefPartnerSideA="pipe111:P0ut" RefPartnerSideB
    ="Pump18:PIn"/>
  <InternalLink Name="Pump18_pipe112" RefPartnerSideA="Pump18:P0ut" RefPartnerSideB=
    "pipe112:PIn"/>
503  <InternalLink Name="Pump18_N18" RefPartnerSideA="Pump18:N" RefPartnerSideB="N18:P"
    />
  <InternalLink Name="pipe112_P19" RefPartnerSideA="pipe112:P19" RefPartnerSideB="
    P19:P1"/>
505  <InternalLink Name="pipe112_T20" RefPartnerSideA="pipe112:T20" RefPartnerSideB="
    T20:P1"/>
  <InternalLink Name="pipe112_Valve21" RefPartnerSideA="pipe112:P0ut"
    RefPartnerSideB="Valve21:PIn"/>
507  <InternalLink Name="Valve21_Y21" RefPartnerSideA="Valve21:Y" RefPartnerSideB="Y21:
    P"/>
  <InternalLink Name="Valve21_pipe113" RefPartnerSideA="Valve21:P0ut"
    RefPartnerSideB="pipe113:PIn"/>
509  <InternalLink Name="pipe113_F22" RefPartnerSideA="pipe113:F22" RefPartnerSideB="
    F22:P1"/>
  <InternalLink Name="pipe113_Jun130" RefPartnerSideA="pipe113:P0ut" RefPartnerSideB

```

```

    = "jun130:P1"/>
511 <InternalLink Name="pipe121_Pump13" RefPartnerSideA="pipe121:P0ut" RefPartnerSideB
    ="Pump13:PIn"/>
    <InternalLink Name="Pump13_pipe122" RefPartnerSideA="Pump13:P0ut" RefPartnerSideB=
    "pipe122:PIn"/>
513 <InternalLink Name="Pump13_N13" RefPartnerSideA="Pump13:N" RefPartnerSideB="N13:P"
    />
    <InternalLink Name="pipe122_Heater43" RefPartnerSideA="pipe122:P0ut"
    RefPartnerSideB="Heater43:PIn"/>
515 <InternalLink Name="pipe122_P14" RefPartnerSideA="pipe122:P14" RefPartnerSideB="
    P14:P1"/>
    <InternalLink Name="Heater43_pipe123" RefPartnerSideA="Heater43:P0ut"
    RefPartnerSideB="pipe123:PIn"/>
517 <InternalLink Name="pipe123_T15" RefPartnerSideA="pipe123:T15" RefPartnerSideB="
    T15:P1"/>
    <InternalLink Name="pipe123_Valve16" RefPartnerSideA="pipe123:P0ut"
    RefPartnerSideB="Valve16:PIn"/>
519 <InternalLink Name="Valve16_Y16" RefPartnerSideA="Valve16:Y" RefPartnerSideB="Y16:
    P"/>
    <InternalLink Name="Valve16_pipe124" RefPartnerSideA="Valve16:P0ut"
    RefPartnerSideB="pipe124:PIn"/>
521 <InternalLink Name="pipe124_F17" RefPartnerSideA="pipe124:F17" RefPartnerSideB="
    F17:P1"/>
    <InternalLink Name="pipe124_Jun130" RefPartnerSideA="pipe124:P0ut" RefPartnerSideB
    ="jun130:P2"/>
523 <InternalLink Name="jun130_pipe131" RefPartnerSideA="jun130:P3" RefPartnerSideB="
    pipe131:PIn"/>
    <InternalLink Name="pipe131_T23" RefPartnerSideA="pipe131:T23" RefPartnerSideB="
    T23:P1"/>
525 <InternalLink Name="pipe131_Jun140" RefPartnerSideA="pipe131:P0ut" RefPartnerSideB
    ="jun140:P1"/>
    <InternalLink Name="jun140_pipe151" RefPartnerSideA="jun140:P2" RefPartnerSideB="
    pipe151:PIn"/>
527 <InternalLink Name="pipe151_Valve24" RefPartnerSideA="pipe151:P0ut"
    RefPartnerSideB="Valve24:PIn"/>
    <InternalLink Name="Valve24_pipe152" RefPartnerSideA="Valve24:P0ut"
    RefPartnerSideB="pipe152:PIn"/>
529 <InternalLink Name="Valve24_Y24" RefPartnerSideA="Valve24:Y" RefPartnerSideB="Y24:
    P"/>
    <InternalLink Name="pipe152_NE1" RefPartnerSideA="pipe152:P0ut" RefPartnerSideB=":
    NE1"/>
531 <InternalLink Name="jun140_pipe161" RefPartnerSideA="jun140:P3" RefPartnerSideB="
    pipe161:PIn"/>
    <InternalLink Name="pipe161_Valve25" RefPartnerSideA="pipe161:P0ut"
    RefPartnerSideB="Valve25:PIn"/>
533 <InternalLink Name="Valve25_pipe162" RefPartnerSideA="Valve25:P0ut"
    RefPartnerSideB="pipe162:PIn"/>
    <InternalLink Name="Valve25_Y25" RefPartnerSideA="Valve25:Y" RefPartnerSideB="Y25:
    P"/>
535 <InternalLink Name="pipe162_NE2" RefPartnerSideA="pipe162:P0ut" RefPartnerSideB=":
    NE2"/>
</SystemUnitClass>
537 <!-- TU20 -->
<SystemUnitClass Name="TU20" RefBaseClassPath="PandIX_PME@PandIX_PME/PlantItemClass"
    >
539 <ExternalInterface Name="NE2" RefBaseClassPath="PandIX_ICL@PandIX_ICL/

```

```

    ProductConnectionPoint"/>
<ExternalInterface Name="NE1" RefBaseClassPath="PandIX_ICL@PandIX_ICL/
    ProductConnectionPoint"/>
541 <!-- Behaelter Form: stehend, zylindrisch -->
<InternalElement Name="Vessel2">
543   <Attribute Name="Position">
     <Value>S01P2:0.58</Value>
545   </Attribute>
   <Attribute Name="NominalVolume">
547     <Value>0.057</Value>
   </Attribute>
   <Attribute Name="NominalFilllevel" AttributeDataType="pix:Position" Unit="-">
549     <Value>S01P2:0.29</Value>
   </Attribute>
   <Attribute Name="ZeroLevel">
551     <Value>0.0</Value>
   </Attribute>
   <Attribute Name="RestVolume">
553     <Value>0.0</Value>
   </Attribute>
   <Attribute Name="TotalVolume">
555     <Value>0.0</Value>
   </Attribute>
   <Attribute Name="Form">
557     <Value>S01F2:(0.0;0.0);(1.0;1.0)</Value>
   </Attribute>
559   </InternalElement>
<ExternalInterface Name="Pin" RefBaseClassPath="PandIX_ICL@PandIX_ICL/
    ProductConnectionPoint">
565   <Attribute Name="Position">
     <Value>S01P2:0.30</Value>
567   </Attribute>
   <Attribute Name="Position">
569     <Value>S01P5:uw;rl</Value>
   </Attribute>
571 </ExternalInterface>
<ExternalInterface Name="POut1" RefBaseClassPath="PandIX_ICL@PandIX_ICL/
    ProductConnectionPoint">
573   <Attribute Name="Position">
     <Value>S01P2:0.00</Value>
575   </Attribute>
   <Attribute Name="Position">
577     <Value>S01P5:d;w;rl</Value>
   </Attribute>
579 </ExternalInterface>
<ExternalInterface Name="L26" RefBaseClassPath="PandIX_ICL@PandIX_ICL/
    MeasurementPoint">
581   <Attribute Name="Position">
     <Value>S01P5:vx;rc</Value>
583   </Attribute>
</ExternalInterface>
<ExternalInterface Name="Q28" RefBaseClassPath="PandIX_ICL@PandIX_ICL/
    MeasurementPoint">
585   <Attribute Name="Position">
     <Value>S01P5:d;l;rw</Value>
587   </Attribute>
589 </ExternalInterface>

```

```
<ExternalInterface Name="T27" RefBaseClassPath="PandIX_ICL@PandIX_ICL/
MeasurementPoint">
591   <Attribute Name="Position">
     <Value>S01P2:0.12</Value>
593   </Attribute>
     <Attribute Name="Position">
595     <Value>S01P5:dw;rc</Value>
     </Attribute>
597   </ExternalInterface>
     <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/
         LVesselRequest"/>
599 </InternalElement>
<!-- Apparate -->
601 <InternalElement Name="Pump29">
     <Attribute Name="Position">
603     <Value>S01P2:0.28</Value>
     </Attribute>
605     <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/PumpRequest"
         />
     </InternalElement>
607 <InternalElement Name="Valve44">
     <Attribute Name="Position">
609     <Value>S01P2:0.28</Value>
     </Attribute>
611     <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/
         CheckValveRequest"/>
     </InternalElement>
613 <InternalElement Name="Valve30">
     <Attribute Name="Position">
615     <Value>S01P2:0.38</Value>
     </Attribute>
617     <Attribute Name="SafeState">
         <Value>open</Value>
619     </Attribute>
     <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/ValveRequest"
         />
621 </InternalElement>
<!-- Rohrleitungen und Rohrleitungsverzweigungen -->
623 <InternalElement Name="pipe211">
     <Attribute Name="DN">
625     <Value>25</Value>
     </Attribute>
627     <Attribute Name="Form">
         <Value>S01F1:(0.20;-0.20);(0.25;0.00);(0.45;-0.10);(0.25;0.00)</Value>
629     </Attribute>
     <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/PipeRequest"
         />
631 </InternalElement>
<InternalElement Name="pipe212">
633   <Attribute Name="DN">
         <Value>25</Value>
635   </Attribute>
     <Attribute Name="Form">
637     <Value>S01F1:(0.05;0.00)</Value>
     </Attribute>
639     <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/PipeRequest"
```

```

        />
    </InternalElement>
641 <InternalElement Name="pipe213">
    <Attribute Name="DN">
643     <Value>25</Value>
    </Attribute>
645     <Attribute Name="Form">
        <Value>S01F1:(0.12;0.00);(0.12;0.10);(0.12;0.00)</Value>
647     </Attribute>
        <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/PipeRequest"
        />
649 </InternalElement>
    <InternalElement Name="pipe214">
651     <Attribute Name="DN">
        <Value>25</Value>
653     </Attribute>
        <Attribute Name="Form">
655     <Value>S01F1:(2.70;0.00);(0.85;0.85)</Value>
        </Attribute>
657     <ExternalInterface Name="F31" RefBaseClassPath="PandIX_ICL@PandIX_ICL/
        MeasurementPoint">
        <Attribute Name="Position">
659     <Value>S01P4:0.30</Value>
        </Attribute>
661     </ExternalInterface>
        <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/PipeRequest"
        />
663 </InternalElement>
    <!-- PLT-Stellen -->
665 <InternalElement Name="L26" RefBaseSystemUnitPath="PandIX_PCE@PandIX_PCE/
    SensorRequest">
    <Attribute Name="PCEInitial">
667     <Value>L</Value>
    </Attribute>
669     <ExternalInterface Name="VAL">
        <Attribute Name="SoftwareSignalType">
671     <Value>I</Value>
        </Attribute>
673     </ExternalInterface>
    </InternalElement>
675 <InternalElement Name="Q28" RefBaseSystemUnitPath="PandIX_PCE@PandIX_PCE/
    SensorRequest">
    <Attribute Name="PCEInitial">
677     <Value>Q</Value>
    </Attribute>
679     <ExternalInterface Name="VAL">
        <Attribute Name="SoftwareSignalType">
681     <Value>I</Value>
        </Attribute>
683     </ExternalInterface>
    </InternalElement>
685 <InternalElement Name="T27" RefBaseSystemUnitPath="PandIX_PCE@PandIX_PCE/
    SensorRequest">
    <Attribute Name="PCEInitial">
687     <Value>T</Value>
    </Attribute>

```

```
689     <ExternalInterface Name="VAL">
        <Attribute Name="SoftwareSignalType">
691           <Value>I</Value>
        </Attribute>
693     </ExternalInterface>
  </InternalElement>
695 <InternalElement Name="N29" RefBaseSystemUnitPath="PandIX_PCE@PandIX_PCE/
    ActuatorRequest">
    <Attribute Name="PCEInitial">
697       <Value>N</Value>
    </Attribute>
699     <ExternalInterface Name="W">
        <Attribute Name="SoftwareSignalType">
701           <Value>O</Value>
        </Attribute>
703     </ExternalInterface>
  </InternalElement>
705 <InternalElement Name="Y30" RefBaseSystemUnitPath="PandIX_PCE@PandIX_PCE/
    ActuatorRequest">
    <Attribute Name="PCEInitial">
707       <Value>Y</Value>
    </Attribute>
709     <ExternalInterface Name="W">
        <Attribute Name="SoftwareSignalType">
711           <Value>I</Value>
        </Attribute>
713     </ExternalInterface>
  </InternalElement>
715 <InternalElement Name="F31" RefBaseSystemUnitPath="PandIX_PCE@PandIX_PCE/
    SensorRequest">
    <Attribute Name="PCEInitial">
717       <Value>F</Value>
    </Attribute>
719     <ExternalInterface Name="VAL">
        <Attribute Name="SoftwareSignalType">
721           <Value>I</Value>
        </Attribute>
723     </ExternalInterface>
  </InternalElement>
725 <!-- Links -->
  <InternalLink Name="NE2_Vessel2" RefPartnerSideA=":NE2" RefPartnerSideB="Vessel2:
    PIn"/>
727 <InternalLink Name="Vessel2_L26" RefPartnerSideA="Vessel2:L26" RefPartnerSideB="
    L26:P1"/>
  <InternalLink Name="Vessel2_Q28" RefPartnerSideA="Vessel2:Q28" RefPartnerSideB="
    Q28:P1"/>
729 <InternalLink Name="Vessel2_T27" RefPartnerSideA="Vessel2:T27" RefPartnerSideB="
    T27:P1"/>
  <InternalLink Name="Vessel2_pipe211" RefPartnerSideA="Vessel2:P0ut1"
    RefPartnerSideB="pipe211:PIn"/>
731 <InternalLink Name="pipe211_Pump29" RefPartnerSideA="pipe211:P0ut" RefPartnerSideB
    ="Pump29:PIn"/>
  <InternalLink Name="Pump29_pipe212" RefPartnerSideA="Pump29:P0ut" RefPartnerSideB=
    "pipe212:PIn"/>
733 <InternalLink Name="Pump29_N29" RefPartnerSideA="Pump29:N" RefPartnerSideB="N29:P"
    />
```

```

<InternalLink Name="pipe212_Valve44" RefPartnerSideA="pipe212:P0ut"
  RefPartnerSideB="Valve44:PIIn"/>
735 <InternalLink Name="Valve44_pipe213" RefPartnerSideA="Valve44:P0ut"
  RefPartnerSideB="pipe213:PIIn"/>
<InternalLink Name="pipe213_Valve30" RefPartnerSideA="pipe213:P0ut"
  RefPartnerSideB="Valve30:PIIn"/>
737 <InternalLink Name="Valve30_Y30" RefPartnerSideA="Valve30:Y" RefPartnerSideB="Y30:
  P"/>
<InternalLink Name="Valve30_pipe214" RefPartnerSideA="Valve30:P0ut"
  RefPartnerSideB="pipe214:PIIn"/>
739 <InternalLink Name="pipe214_F31" RefPartnerSideA="pipe214:F31" RefPartnerSideB="
  F31:P1"/>
<InternalLink Name="pipe214_NE1" RefPartnerSideA="pipe214:P0ut" RefPartnerSideB=":
  NE1"/>
741 </SystemUnitClass>
<!-- TU30 -->
743 <SystemUnitClass Name="TU30" RefBaseClassPath="PandIX_PME@PandIX_PME/PlantItemClass"
  >
  <ExternalInterface Name="NE1" RefBaseClassPath="PandIX_ICL@PandIX_ICL/
    ProductConnectionPoint"/>
745 <ExternalInterface Name="NE2" RefBaseClassPath="PandIX_ICL@PandIX_ICL/
    ProductConnectionPoint"/>
<!-- Behaelter Form: stehend, zylindrisch -->
747 <InternalElement Name="Vessel3">
  <Attribute Name="Position">
749   <Value>S01P2:0.58</Value>
  </Attribute>
751 <Attribute Name="NominalVolume">
  <Value>0.057</Value>
753 </Attribute>
<Attribute Name="NominalFilllevel" AttributeDataType="pix:Position" Unit="-">
755   <Value>S01P2:0.29</Value>
  </Attribute>
757 <Attribute Name="ZeroLevel">
  <Value>0.0</Value>
759 </Attribute>
<Attribute Name="RestVolume">
761   <Value>0.0</Value>
  </Attribute>
763 <Attribute Name="TotalVolume">
  <Value>0.057</Value>
765 </Attribute>
<Attribute Name="Form">
767   <Value>S01F2:(0.0;0.0);(1.0;1.0)</Value>
  </Attribute>
769 <ExternalInterface Name="PIIn" RefBaseClassPath="PandIX_ICL@PandIX_ICL/
    ProductConnectionPoint">
  <Attribute Name="Position">
771   <Value>S01P2:0.30</Value>
  </Attribute>
773 <Attribute Name="Position">
  <Value>S01P5:uw;rl</Value>
775 </Attribute>
</ExternalInterface>
777 <ExternalInterface Name="P0ut1" RefBaseClassPath="PandIX_ICL@PandIX_ICL/
    ProductConnectionPoint">

```

```

    <Attribute Name="Position">
779     <Value>S01P2:0.00</Value>
    </Attribute>
781    <Attribute Name="Position">
    <Value>S01P5:dw;rl</Value>
783    </Attribute>
  </ExternalInterface>
785  <ExternalInterface Name="L34" RefBaseClassPath="PandIX_ICL@PandIX_ICL/
    MeasurementPoint">
    <Attribute Name="Position">
787     <Value>S01P5:vx;rc</Value>
    </Attribute>
789  </ExternalInterface>
  <ExternalInterface Name="L32" RefBaseClassPath="PandIX_ICL@PandIX_ICL/
    MeasurementPoint">
    <Attribute Name="Position">
791     <Value>S01P5:dl;rl</Value>
    </Attribute>
793  </ExternalInterface>
  <ExternalInterface Name="L35" RefBaseClassPath="PandIX_ICL@PandIX_ICL/
    MeasurementPoint">
    <Attribute Name="Position">
797     <Value>S01P5:vx;rc</Value>
    </Attribute>
799  </ExternalInterface>
  <ExternalInterface Name="T33" RefBaseClassPath="PandIX_ICL@PandIX_ICL/
    MeasurementPoint">
801    <Attribute Name="Position">
    <Value>S01P2:0.06</Value>
803    </Attribute>
    <Attribute Name="Position">
805     <Value>S01P5:dl;rl</Value>
    </Attribute>
807  </ExternalInterface>
  <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/
    LVesselRequest"/>
809 </InternalElement>
<!-- Apparate -->
811 <InternalElement Name="Pump36">
  <Attribute Name="Position">
813    <Value>S01P2:0.27</Value>
  </Attribute>
815  <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/PumpRequest"
    />
</InternalElement>
817 <InternalElement Name="Pump38">
  <Attribute Name="Position">
819    <Value>S01P2:0.27</Value>
  </Attribute>
821  <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/PumpRequest"
    />
</InternalElement>
823 <InternalElement Name="Valve45">
  <Attribute Name="Position">
825    <Value>S01P2:0.27</Value>
  </Attribute>

```



```

827     <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/
        CheckValveRequest"/>
</InternalElement>
829 <InternalElement Name="Valve46">
    <Attribute Name="Position">
831         <Value>S01P2:0.27</Value>
    </Attribute>
833     <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/
        CheckValveRequest"/>
</InternalElement>
835 <InternalElement Name="Valve37">
    <Attribute Name="Position">
837         <Value>S01P2:0.37</Value>
    </Attribute>
839     <Attribute Name="SafeState">
        <Value>none</Value>
841     </Attribute>
    <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/ValveRequest"
        />
843 </InternalElement>
<InternalElement Name="Valve39">
845     <Attribute Name="Position">
        <Value>S01P2:0.37</Value>
847     </Attribute>
    <Attribute Name="SafeState">
849         <Value>none</Value>
    </Attribute>
851     <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/ValveRequest"
        />
</InternalElement>
853 <!-- Rohrleitungen und Rohrleitungsverzweigungen -->
<InternalElement Name="pipe311">
855     <Attribute Name="DN">
        <Value>25</Value>
857     </Attribute>
    <Attribute Name="Form">
859         <Value>S01F1: (0.15;0.15); (0.32;0.00); (0.25;0.00)</Value>
    </Attribute>
861     <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/PipeRequest"
        />
</InternalElement>
863 <InternalElement Name="jun320">
    <Attribute Name="Position">
865         <Value>S01P2:0.43</Value>
    </Attribute>
867     <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/
        PipeJunctionRequest"/>
</InternalElement>
869 <InternalElement Name="pipe331">
    <Attribute Name="DN">
871         <Value>25</Value>
    </Attribute>
873     <Attribute Name="Form">
        <Value>S01F1: (0.15;-0.15); (0.18;0.00)</Value>
875     </Attribute>
    <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/PipeRequest"

```

```

      />
877   </InternalElement>
      <InternalElement Name="pipe332">
879       <Attribute Name="DN">
            <Value>25</Value>
881       </Attribute>
            <Attribute Name="Form">
883       <Value>S01F1:(0.05;0.00)</Value>
            </Attribute>
            <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/PipeRequest"
885       />
      </InternalElement>
887   <InternalElement Name="pipe333">
            <Attribute Name="DN">
889       <Value>25</Value>
            </Attribute>
            <Attribute Name="Form">
891       <Value>S01F1:(0.65;0.10)</Value>
            </Attribute>
            <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/PipeRequest"
893       />
      </InternalElement>
895   <InternalElement Name="pipe334">
            <Attribute Name="DN">
897       <Value>25</Value>
            </Attribute>
            <Attribute Name="Form">
899       <Value>S01F1:(0.10;0.00);(0.10;0.00)</Value>
            </Attribute>
            <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/PipeRequest"
901       />
      </InternalElement>
903   <InternalElement Name="pipe341">
            <Attribute Name="DN">
905       <Value>40</Value>
            </Attribute>
            <Attribute Name="Form">
907       <Value>S01F1:(0.25;0.00);(0.15;-0.15);(0.18;0.00)</Value>
            </Attribute>
            <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/PipeRequest"
909       />
      </InternalElement>
911   <InternalElement Name="pipe342">
            <Attribute Name="DN">
913       <Value>25</Value>
            </Attribute>
            <Attribute Name="Form">
915       <Value>S01F1:(0.05;0.00)</Value>
            </Attribute>
            <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/PipeRequest"
917       />
      </InternalElement>
919   <InternalElement Name="pipe343">
            <Attribute Name="DN">
921       <Value>25</Value>
            </Attribute>
            <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/PipeRequest"
923       />
      </InternalElement>
925   <InternalElement Name="pipe343">
            <Attribute Name="DN">
            <Value>25</Value>
            </Attribute>

```

```

927     <Attribute Name="Form">
          <Value>S01F1:(0.65;0.10)</Value>
929     </Attribute>
    <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/PipeRequest"
      />
931  </InternalElement>
    <InternalElement Name="pipe344">
933     <Attribute Name="DN">
          <Value>25</Value>
935     </Attribute>
    <Attribute Name="Form">
937     <Value>S01F1:(0.10;0.00);(0.10;0.00)</Value>
    </Attribute>
939     <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/PipeRequest"
      />
    </InternalElement>
941  <InternalElement Name="jun350">
    <Attribute Name="Position">
943     <Value>S01P2:0.37</Value>
    </Attribute>
945     <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/
      PipeJunctionRequest"/>
    </InternalElement>
947  <InternalElement Name="pipe351">
    <Attribute Name="DN">
949     <Value>25</Value>
    </Attribute>
951     <Attribute Name="Form">
          <Value>S01F1:(1.00;0.94);(0.13;0.00);(0.25;-0.15);(1.00;0.00);(0.10;0.07)</
            Value>
953     </Attribute>
    <ExternalInterface Name="F40" RefBaseClassPath="PandIX_ICL@PandIX_ICL/
      MeasurementPoint">
955     <Attribute Name="Position">
          <Value>S01P4:0.65</Value>
957     </Attribute>
    </ExternalInterface>
959     <RoleRequirements RefBaseRoleClassPath="PandIX_PPE_RC@PPE_BasicLib/PipeRequest"
      />
    </InternalElement>
961  <!-- PLT-Stellen -->
    <InternalElement Name="L32" RefBaseSystemUnitPath="PandIX_PCE@PandIX_PCE/
      SensorRequest">
963     <Attribute Name="FunctionCode">
          <Value>L</Value>
965     </Attribute>
    <ExternalInterface Name="VAL">
967     <Attribute Name="SoftwareSignalType">
          <Value>I</Value>
969     </Attribute>
    </ExternalInterface>
971  <InternalElement Name="S+" RefBaseSystemUnitPath="NE150@NE150SUCLibrary/Switch">
    <Attribute Name="LimitValueType">
973     <Value>+</Value>
    </Attribute>
975     <Attribute Name="LimitValue">

```

```

    <Value>0.0</Value>
977   </Attribute>
    <Attribute Name="LimitValueHysteresis">
979     <Value>0.0</Value>
    </Attribute>
981   <ExternalInterface Name="S+" RefBaseClassPath="NE150@NE150InterfaceLibrary/
      SWSignal"/>
    </InternalElement>
983 </InternalElement>
    <InternalElement Name="L34" RefBaseSystemUnitPath="PandIX_PCE@PandIX_PCE/
      SensorRequest">
985   <Attribute Name="FunctionCode">
    <Value>L</Value>
987   </Attribute>
    <ExternalInterface Name="VAL">
989   <Attribute Name="SoftwareSignalType">
    <Value>I</Value>
991   </Attribute>
    </ExternalInterface>
993 </InternalElement>
    <InternalElement Name="L35" RefBaseSystemUnitPath="PandIX_PCE@PandIX_PCE/
      SensorRequest">
995   <Attribute Name="FunctionCode">
    <Value>L</Value>
997   </Attribute>
    <ExternalInterface Name="VAL">
999   <Attribute Name="SoftwareSignalType">
    <Value>I</Value>
1001   </Attribute>
    </ExternalInterface>
1003 </InternalElement>
    <InternalElement Name="T33" RefBaseSystemUnitPath="PandIX_PCE@PandIX_PCE/
      SensorRequest">
1005   <Attribute Name="FunctionCode">
    <Value>T</Value>
1007   </Attribute>
    <ExternalInterface Name="VAL">
1009   <Attribute Name="SoftwareSignalType">
    <Value>I</Value>
1011   </Attribute>
    </ExternalInterface>
1013 </InternalElement>
    <InternalElement Name="N36" RefBaseSystemUnitPath="PandIX_PCE@PandIX_PCE/
      ActuatorRequest">
1015   <Attribute Name="FunctionCode">
    <Value>N</Value>
1017   </Attribute>
    <ExternalInterface Name="W">
1019   <Attribute Name="SoftwareSignalType">
    <Value>O</Value>
1021   </Attribute>
    </ExternalInterface>
1023 </InternalElement>
    <InternalElement Name="N38" RefBaseSystemUnitPath="PandIX_PCE@PandIX_PCE/
      ActuatorRequest">
1025   <Attribute Name="FunctionCode">
```

```

    <Value>N</Value>
1027 </Attribute>
    <ExternalInterface Name="W">
1029     <Attribute Name="SoftwareSignalType">
        <Value>0</Value>
1031     </Attribute>
    </ExternalInterface>
1033 </InternalElement>
<InternalElement Name="Y37" RefBaseSystemUnitPath="PandIX_PCE@PandIX_PCE/
    ActuatorRequest">
1035     <Attribute Name="FunctionCode">
        <Value>Y</Value>
1037     </Attribute>
    <ExternalInterface Name="W">
1039     <Attribute Name="SoftwareSignalType">
        <Value>I</Value>
1041     </Attribute>
    </ExternalInterface>
1043 </InternalElement>
<InternalElement Name="Y39" RefBaseSystemUnitPath="PandIX_PCE@PandIX_PCE/
    ActuatorRequest">
1045     <Attribute Name="FunctionCode">
        <Value>Y</Value>
1047     </Attribute>
    <ExternalInterface Name="W">
1049     <Attribute Name="SoftwareSignalType">
        <Value>I</Value>
1051     </Attribute>
    </ExternalInterface>
1053 </InternalElement>
<InternalElement Name="F40" RefBaseSystemUnitPath="PandIX_PCE@PandIX_PCE/
    SensorRequest">
1055     <Attribute Name="FunctionCode">
        <Value>F</Value>
1057     </Attribute>
    <ExternalInterface Name="VAL">
1059     <Attribute Name="SoftwareSignalType">
        <Value>I</Value>
1061     </Attribute>
    </ExternalInterface>
1063 </InternalElement>
<!-- Links -->
1065 <InternalLink Name="NE2_Vessel3" RefPartnerSideA=":NE2" RefPartnerSideB="Vessel3:
    PIn"/>
<InternalLink Name="Vessel_L32" RefPartnerSideA="Vessel3:L32" RefPartnerSideB="L32
    :P1"/>
1067 <InternalLink Name="Vessel3_L34" RefPartnerSideA="Vessel3:L34" RefPartnerSideB="
    L34:P1"/>
<InternalLink Name="Vessel3_L35" RefPartnerSideA="Vessel3:L35" RefPartnerSideB="
    L35:P1"/>
1069 <InternalLink Name="Vessel3_T33" RefPartnerSideA="Vessel3:T33" RefPartnerSideB="
    T33:P1"/>
<InternalLink Name="Vessel3_pipe311" RefPartnerSideA="Vessel3:P0ut1"
    RefPartnerSideB="pipe311:PIn"/>
1071 <InternalLink Name="pipe311_Jun320" RefPartnerSideA="pipe311:P0ut" RefPartnerSideB
    ="jun320:P1"/>

```

```

1073 <InternalLink Name="jun320_pipe331" RefPartnerSideA="jun320:P2" RefPartnerSideB="
    pipe331:PIn"/>
1075 <InternalLink Name="pipe331_Pump36" RefPartnerSideA="pipe331:P0ut" RefPartnerSideB
    ="Pump36:PIn"/>
    <InternalLink Name="Pump36_pipe332" RefPartnerSideA="Pump36:P0ut" RefPartnerSideB=
    "pipe332:PIn"/>
1075 <InternalLink Name="Pump36_N36" RefPartnerSideA="Pump36:N" RefPartnerSideB="N36:P"
    />
    <InternalLink Name="pipe332_Valve45" RefPartnerSideA="pipe332:P0ut"
    RefPartnerSideB="Valve45:PIn"/>
1077 <InternalLink Name="Valve45_pipe333" RefPartnerSideA="Valve45:P0ut"
    RefPartnerSideB="pipe333:PIn"/>
    <InternalLink Name="pipe333_Valve37" RefPartnerSideA="pipe333:P0ut"
    RefPartnerSideB="Valve37:PIn"/>
1079 <InternalLink Name="Valve37_Y37" RefPartnerSideA="Valve37:Y" RefPartnerSideB="Y37:
    P"/>
    <InternalLink Name="Valve37_pipe334" RefPartnerSideA="Valve37:P0ut"
    RefPartnerSideB="pipe334:PIn"/>
1081 <InternalLink Name="pipe334_Jun350" RefPartnerSideA="pipe334:P0ut" RefPartnerSideB
    ="jun350:P1"/>
    <InternalLink Name="jun320_pipe341" RefPartnerSideA="jun320:P3" RefPartnerSideB="
    pipe341:PIn"/>
1083 <InternalLink Name="pipe341_Pump38" RefPartnerSideA="pipe341:P0ut" RefPartnerSideB
    ="Pump38:PIn"/>
    <InternalLink Name="Pump38_pipe342" RefPartnerSideA="Pump38:P0ut" RefPartnerSideB=
    "pipe342:PIn"/>
1085 <InternalLink Name="Pump38_N38" RefPartnerSideA="Pump38:N" RefPartnerSideB="N38:P"
    />
    <InternalLink Name="pipe342_Valve46" RefPartnerSideA="pipe342:P0ut"
    RefPartnerSideB="Valve46:PIn"/>
1087 <InternalLink Name="Valve46_pipe343" RefPartnerSideA="Valve46:P0ut"
    RefPartnerSideB="pipe343:PIn"/>
    <InternalLink Name="pipe343_Valve39" RefPartnerSideA="pipe343:P0ut"
    RefPartnerSideB="Valve39:PIn"/>
1089 <InternalLink Name="Valve39_Y39" RefPartnerSideA="Valve39:Y" RefPartnerSideB="Y39:
    P"/>
    <InternalLink Name="Valve39_pipe344" RefPartnerSideA="Valve39:P0ut"
    RefPartnerSideB="pipe344:PIn"/>
1091 <InternalLink Name="pipe344_Jun350" RefPartnerSideA="pipe344:P0ut" RefPartnerSideB
    ="jun350:P2"/>
    <InternalLink Name="jun350_pipe351" RefPartnerSideA="jun350:P3" RefPartnerSideB="
    pipe351:PIn"/>
1093 <InternalLink Name="pipe351_F40" RefPartnerSideA="pipe351:F40" RefPartnerSideB="
    F40:P1"/>
    <InternalLink Name="pipe351_NE1" RefPartnerSideA="pipe351:P0ut" RefPartnerSideB=":
    NE1"/>
1095 </SystemUnitClass>
    </SystemUnitClassLib>
1097</CAEXFile>

```

A.2 Abbildung der Aspektmetamodelle in Graql

Nachfolgend sind die Definitionen der einzelnen Aspektmetamodelle in der Sprache Graql aufgeführt.

A.2.1 Aspektmetamodell Topologie

Listing A.2: Definition des Aspektmetamodells Topologie in Graql

```
define
  ### Relation ###
  topologyRelation sub baseRelation, abstract,
    relates source,
    relates target,
    relates emergence,
    relates breakup;

  # static perspective
  productConnection sub topologyRelation, abstract,
    relates source,
    relates target,
    relates emergence,
    relates breakup;

  primaryProductConnection sub productConnection,
    relates source,
    relates target,
    relates emergence,
    relates breakup;

  secondaryProductConnection sub productConnection,
    relates source,
    relates target,
    relates emergence,
    relates breakup;

  # dynamic perspective
  productFlow sub topologyRelation, abstract,
    relates source,
    relates target,
    relates emergence,
    relates breakup;

  primaryProductFlow sub productFlow,
    relates source,
    relates target,
    relates emergence,
```

```
    relates breakup;

secondaryProductFlow sub productFlow,
    relates source,
    relates target,
    relates emergence,
    relates breakup;

possibleProductFlow sub topologyRelation, abstract,
    relates source,
    relates target,
    relates emergence,
    relates breakup;

primaryPossibleProductFlow sub possibleProductFlow,
    relates source,
    relates target,
    relates emergence,
    relates breakup;

secondaryPossibleProductFlow sub possibleProductFlow,
    relates source,
    relates target,
    relates emergence,
    relates breakup;

### Events ###
topologyAlteration sub event;

topologyModification sub topologyAlteration;

failure sub topologyAlteration;

actuatorEvent sub topologyAlteration;
pumpOn sub actuatorEvent;
pumpOff sub actuatorEvent;
valveOpen sub actuatorEvent;
valveClose sub actuatorEvent;

### Entities ###
productContainer sub baseEntity,
    plays source,
    plays target;

heatExchanger sub productContainer;
vessel sub productContainer;
pipe sub productContainer;
```



```

reactor sub productContainer;
junction sub productContainer;
checkValve sub productContainer;

externalInterface sub productContainer, abstract;
eductSource sub externalInterface;
productTarget sub externalInterface;

actuator sub productContainer;
valve sub actuator;
pump sub actuator;
rPump sub pump;
dPump sub pump;

```

A.2.2 Aspektmetamodell Messen

Listing A.3: Definition des Aspektmetamodells Messen in Graql

```

define
  ### Relations ###
  measuring sub baseRelation,
    relates measurementPlace,
    relates measurementDevice,
    has measuredValue;

  ### Entities ###
  #Add new role to productContainer
  productContainer sub baseEntity,
    plays measurementPlace;

  sensor sub baseEntity,
    plays measurementDevice;

  levelSensor sub sensor;
  flowSensor sub sensor;
  currentSensor sub sensor;
  pressureSensor sub sensor;
  temperatureSensor sub sensor;
  qualitySensor sub sensor;

  ### Events ###
  sensorEvent sub event;
  calibration sub sensorEvent;

  ### Attribute ###
  measuredValue sub attribute,

```

```
datatype double,  
has timestamp,  
has unit,  
has description;
```

A.2.3 Aspektmetamodell Deployment

Listing A.4: Definition des Aspektmetamodells Deployment in Graql

```
define  
  ### Entities ###  
  computingNode sub baseEntity,  
    plays resourceProvider;  
  
  softwareComponent sub baseEntity,  
    plays resourceConsumer;  
  
  ### Relation ###  
  runsOn sub baseRelation,  
    relates resourceConsumer,  
    relates resourceProvider,  
    relates emergence,  
    relates breakup;  
  
  ### Event ###  
  redeployment sub event;
```

A.2.4 Aspektmetamodell Steuerungsstruktur

Listing A.5: Definition des Aspektmetamodells Steuerungsstruktur in Graql

```
define  
  ### Entities ###  
  operator sub baseEntity,  
    plays occupier;  
  
  measure sub softwareComponent,  
    plays occupier,  
    plays occupee;  
  
  controlUnit sub softwareComponent,  
    plays occupier,  
    plays occupee,  
    plays dependant,  
    plays dependence,  
    has order;
```

```

singleControlUnit sub controlUnit;
groupControlUnit sub controlUnit;

sensorAbstraction sub softwareComponent,
    plays dependence;

### Relations ###
occupation sub baseRelation,
    relates occupee,
    relates occupier,
    relates emergence,
    relates breakup;

controlDependency sub baseRelation,
    relates dependant,
    relates dependence,
    relates emergence,
    relates breakup;

### Events ###
occupationChanged sub event;
freed sub occupationChanged;
occupied sub occupationChanged;

controlDependencyChanged sub event;

### Attributes ###
order sub attribute,
    datatype string;

```

A.2.5 Aspektmetamodell Hardware-Software-Kopplung

Listing A.6: Definition des Aspektmetamodells Hardware-Software-Kopplung in Graql

```

define
    ### Entities ###
    #Add new role to sensor
    sensor sub baseEntity,
        plays hardwareComponent;

    #Add new role to actuator
    actuator sub productContainer,
        plays hardwareComponent;

    #Add new role to sensorAbstraction

```

```

sensorAbstraction sub softwareComponent,
    plays controlComponent;

#Add new role to singleControlUnit
singleControlUnit sub controlUnit,
    plays controlComponent;

### Relation ###
hwSwCoupling sub baseRelation,
    relates controlComponent,
    relates hardwareComponent,
    relates emergence,
    relates breakup;

### Event ###
hwSwCouplingChanged sub event;

```

A.2.6 Aspektmetamodell Wirkung

Listing A.7: Definition des Aspektmetamodells Wirkung in Graql

```

define
    ### Entity ###
    # add now role to baseEntity
    baseEntity sub entity,
        plays origin,
        plays end;

    ### Relation ###
    effect sub baseRelation,
        relates origin,
        relates end,
        has ruleID;

    ### Attribute ####
    ruleID sub attribute,
        datatype string;

```

A.3 GraphML-Dateien der Nachbarschaft von L26

Listing A.8: GraphML-Datei der ersten Phase

```

<ns0:graphml xmlns:ns0="http://graphml.graphdrawing.org/xmlns" xmlns:xsi="http://www.w3
.org/2001/XMLSchema-instance" xsi:schemaLocation="http://graphml.graphdrawing.org/
xmlns http://graphml.graphdrawing.org/xmlns/1.0/graphml.xsd"><ns0:key attr.name="
color" attr.type="string" for="edge" id="d6" />

```

```

2<ns0:key attr.name="dvalidUntil" attr.type="string" for="edge" id="d5" />
  <ns0:key attr.name="dvalidFrom" attr.type="string" for="edge" id="d4" />
4<ns0:key attr.name="validUntil" attr.type="string" for="edge" id="d3" />
  <ns0:key attr.name="ts" attr.type="string" for="edge" id="d2" />
6<ns0:key attr.name="validFrom" attr.type="string" for="edge" id="d1" />
  <ns0:key attr.name="rtype" attr.type="string" for="edge" id="d0" />
8<ns0:graph edgedefault="undirected"><ns0:node id="L26" />
  <ns0:node id="L26_SA" />
10<ns0:node id="Vessel2" />
  <ns0:node id="pipe211" />
12<ns0:node id="pipe162" />
  <ns0:edge id="L26_L26_SA" source="L26" target="L26_SA">
14  <ns0:data key="d0">hwSwCoupling</ns0:data>
    <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
16  <ns0:data key="d2">1588259400.0</ns0:data>
    <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
18  <ns0:data key="d4">-648.0505630970001</ns0:data>
    <ns0:data key="d5">-30946952999.0</ns0:data>
20  <ns0:data key="d6">black</ns0:data>
  </ns0:edge>
22<ns0:edge id="L26_Vessel2" source="L26" target="Vessel2">
  <ns0:data key="d0">measuring</ns0:data>
24  <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
    <ns0:data key="d2">1588259400.0</ns0:data>
26  <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
    <ns0:data key="d4">-648.0505630970001</ns0:data>
28  <ns0:data key="d5">-30946952999.0</ns0:data>
    <ns0:data key="d6">black</ns0:data>
30</ns0:edge>
  <ns0:edge id="Vessel2_pipe211" source="Vessel2" target="pipe211">
32  <ns0:data key="d0">primaryProductFlow</ns0:data>
    <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
34  <ns0:data key="d2">1588259400.0</ns0:data>
    <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
36  <ns0:data key="d4">-648.0505630970001</ns0:data>
    <ns0:data key="d5">-30946952999.0</ns0:data>
38  <ns0:data key="d6">black</ns0:data>
  </ns0:edge>
40<ns0:edge id="Vessel2_pipe162" source="Vessel2" target="pipe162">
  <ns0:data key="d0">primaryProductFlow</ns0:data>
42  <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
    <ns0:data key="d2">1588259400.0</ns0:data>
44  <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
    <ns0:data key="d4">-648.0505630970001</ns0:data>
46  <ns0:data key="d5">-30946952999.0</ns0:data>
    <ns0:data key="d6">black</ns0:data>
48</ns0:edge>
</ns0:graph></ns0:graphml>

```

Listing A.9: GraphML-Datei der zweiten Phase

```

1<ns0:graphml xmlns:ns0="http://graphml.graphdrawing.org/xmlns" xmlns:xsi="http://www.w3
  .org/2001/XMLSchema-instance" xsi:schemaLocation="http://graphml.graphdrawing.org/
  xmlns http://graphml.graphdrawing.org/xmlns/1.0/graphml.xsd"><ns0:key attr.name="
  etype" attr.type="string" for="edge" id="d7" />
  <ns0:key attr.name="color" attr.type="string" for="edge" id="d6" />
3<ns0:key attr.name="dvalidUntil" attr.type="string" for="edge" id="d5" />

```

```
<ns0:key attr.name="dvalidFrom" attr.type="string" for="edge" id="d4" />
5<ns0:key attr.name="validUntil" attr.type="string" for="edge" id="d3" />
<ns0:key attr.name="ts" attr.type="string" for="edge" id="d2" />
7<ns0:key attr.name="validFrom" attr.type="string" for="edge" id="d1" />
<ns0:key attr.name="rtype" attr.type="string" for="edge" id="d0" />
9<ns0:graph edgedefault="undirected"><ns0:node id="L26" />
  <ns0:node id="L26_SA" />
11<ns0:node id="Vessel2" />
  <ns0:node id="pipe211" />
13<ns0:node id="jun140" />
  <ns0:node id="pipe151" />
15<ns0:node id="jun130" />
  <ns0:node id="pipe113" />
17<ns0:node id="pipe124" />
  <ns0:node id="pipe131" />
19<ns0:node id="pipe161" />
  <ns0:node id="Y25" />
21<ns0:node id="pipe162" />
  <ns0:edge id="L26_L26_SA" source="L26" target="L26_SA">
23    <ns0:data key="d0">hwSwCoupling</ns0:data>
    <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
25    <ns0:data key="d2">1588259640.0</ns0:data>
    <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
27    <ns0:data key="d4">-888.0505630970001</ns0:data>
    <ns0:data key="d5">-30946952759.0</ns0:data>
29    <ns0:data key="d6">black</ns0:data>
  </ns0:edge>
31<ns0:edge id="L26_Vessel2" source="L26" target="Vessel2">
  <ns0:data key="d0">measuring</ns0:data>
33    <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
    <ns0:data key="d2">1588259640.0</ns0:data>
35    <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
    <ns0:data key="d4">-888.0505630970001</ns0:data>
37    <ns0:data key="d5">-30946952759.0</ns0:data>
    <ns0:data key="d6">black</ns0:data>
39</ns0:edge>
  <ns0:edge id="Vessel2_pipe211" source="Vessel2" target="pipe211">
41    <ns0:data key="d0">primaryProductFlow</ns0:data>
    <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
43    <ns0:data key="d2">1588259640.0</ns0:data>
    <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
45    <ns0:data key="d4">-888.0505630970001</ns0:data>
    <ns0:data key="d5">-30946952759.0</ns0:data>
47    <ns0:data key="d6">black</ns0:data>
  </ns0:edge>
49<ns0:edge id="Vessel2_pipe162" source="Vessel2" target="pipe162">
  <ns0:data key="d0">primaryProductFlow</ns0:data>
51    <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
    <ns0:data key="d2">1588259640.0</ns0:data>
53    <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
    <ns0:data key="d4">-888.0505630970001</ns0:data>
55    <ns0:data key="d5">-30946952759.0</ns0:data>
    <ns0:data key="d6">black</ns0:data>
57</ns0:edge>
  <ns0:edge id="jun140_pipe151" source="jun140" target="pipe151">
59    <ns0:data key="d0">primaryProductFlow</ns0:data>
```

```

<ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
61 <ns0:data key="d2">1588259640.0</ns0:data>
<ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
63 <ns0:data key="d4">-888.0505630970001</ns0:data>
<ns0:data key="d5">-30946952759.0</ns0:data>
65 <ns0:data key="d6">black</ns0:data>
</ns0:edge>
67<ns0:edge id="jun140_pipe131" source="jun140" target="pipe131">
  <ns0:data key="d0">primaryProductFlow</ns0:data>
69 <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
<ns0:data key="d2">1588259640.0</ns0:data>
71 <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
<ns0:data key="d4">-888.0505630970001</ns0:data>
73 <ns0:data key="d5">-30946952759.0</ns0:data>
<ns0:data key="d6">black</ns0:data>
75</ns0:edge>
<ns0:edge id="jun140_pipe161" source="jun140" target="pipe161">
77 <ns0:data key="d0">primaryProductFlow</ns0:data>
<ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
79 <ns0:data key="d2">1588259640.0</ns0:data>
<ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
81 <ns0:data key="d4">-888.0505630970001</ns0:data>
<ns0:data key="d5">-30946952759.0</ns0:data>
83 <ns0:data key="d6">black</ns0:data>
</ns0:edge>
85<ns0:edge id="jun130_pipe113" source="jun130" target="pipe113">
  <ns0:data key="d0">primaryProductFlow</ns0:data>
87 <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
<ns0:data key="d2">1588259640.0</ns0:data>
89 <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
<ns0:data key="d4">-888.0505630970001</ns0:data>
91 <ns0:data key="d5">-30946952759.0</ns0:data>
<ns0:data key="d6">black</ns0:data>
93</ns0:edge>
<ns0:edge id="jun130_pipe124" source="jun130" target="pipe124">
95 <ns0:data key="d0">primaryProductFlow</ns0:data>
<ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
97 <ns0:data key="d2">1588259640.0</ns0:data>
<ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
99 <ns0:data key="d4">-888.0505630970001</ns0:data>
<ns0:data key="d5">-30946952759.0</ns0:data>
101 <ns0:data key="d6">black</ns0:data>
</ns0:edge>
103<ns0:edge id="jun130_pipe131" source="jun130" target="pipe131">
  <ns0:data key="d0">primaryProductFlow</ns0:data>
105 <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
<ns0:data key="d2">1588259640.0</ns0:data>
107 <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
<ns0:data key="d4">-888.0505630970001</ns0:data>
109 <ns0:data key="d5">-30946952759.0</ns0:data>
<ns0:data key="d6">black</ns0:data>
111</ns0:edge>
<ns0:edge id="pipe161_Y25" source="pipe161" target="Y25">
113 <ns0:data key="d0">primaryProductFlow</ns0:data>
<ns0:data key="d7">valveClose</ns0:data>
115 <ns0:data key="d1">2020-04-30 17:13:54.511673</ns0:data>

```

```

    <ns0:data key="d2">1588259640.0</ns0:data>
117 <ns0:data key="d3">2020-04-30 17:19:24.263013</ns0:data>
    <ns0:data key="d4">-5.4883270263671875</ns0:data>
119 <ns0:data key="d5">-324.26301288604736</ns0:data>
    <ns0:data key="d6">red</ns0:data>
121</ns0:edge>
    <ns0:edge id="Y25_pipe162" source="Y25" target="pipe162">
123 <ns0:data key="d0">primaryProductFlow</ns0:data>
    <ns0:data key="d7">valveClose</ns0:data>
125 <ns0:data key="d1">2020-04-30 17:13:54.511673</ns0:data>
    <ns0:data key="d2">1588259640.0</ns0:data>
127 <ns0:data key="d3">2020-04-30 17:19:24.263013</ns0:data>
    <ns0:data key="d4">-5.4883270263671875</ns0:data>
129 <ns0:data key="d5">-324.26301288604736</ns0:data>
    <ns0:data key="d6">red</ns0:data>
131</ns0:edge>
</ns0:graph></ns0:graphml>

```

Listing A.10: GraphML-Datei der dritten Phase

```

<ns0:graphml xmlns:ns0="http://graphml.graphdrawing.org/xmlns" xmlns:xsi="http://www.w3
.org/2001/XMLSchema-instance" xsi:schemaLocation="http://graphml.graphdrawing.org/
xmlns http://graphml.graphdrawing.org/xmlns/1.0/graphml.xsd"><ns0:key attr.name="
etype" attr.type="string" for="edge" id="d7" />
2<ns0:key attr.name="color" attr.type="string" for="edge" id="d6" />
    <ns0:key attr.name="dvalidUntil" attr.type="string" for="edge" id="d5" />
4<ns0:key attr.name="dvalidFrom" attr.type="string" for="edge" id="d4" />
    <ns0:key attr.name="validUntil" attr.type="string" for="edge" id="d3" />
6<ns0:key attr.name="ts" attr.type="string" for="edge" id="d2" />
    <ns0:key attr.name="validFrom" attr.type="string" for="edge" id="d1" />
8<ns0:key attr.name="rtype" attr.type="string" for="edge" id="d0" />
    <ns0:graph edgedefault="undirected"><ns0:node id="L26" />
10<ns0:node id="L26_SA" />
    <ns0:node id="Vessel2" />
12<ns0:node id="pipe211" />
    <ns0:node id="jun140" />
14<ns0:node id="pipe151" />
    <ns0:node id="jun130" />
16<ns0:node id="pipe113" />
    <ns0:node id="N13" />
18<ns0:node id="pipe121" />
    <ns0:node id="pipe122" />
20<ns0:node id="Heater43" />
    <ns0:node id="pipe123" />
22<ns0:node id="Y16" />
    <ns0:node id="pipe124" />
24<ns0:node id="pipe131" />
    <ns0:node id="pipe161" />
26<ns0:node id="Y25" />
    <ns0:node id="pipe162" />
28<ns0:edge id="L26_L26_SA" source="L26" target="L26_SA">
    <ns0:data key="d0">hwSwCoupling</ns0:data>
30 <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
    <ns0:data key="d2">1588259700.0</ns0:data>
32 <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
    <ns0:data key="d4">-948.0505630970001</ns0:data>
34 <ns0:data key="d5">-30946952699.0</ns0:data>

```



```

    <ns0:data key="d6">black</ns0:data>
36</ns0:edge>
    <ns0:edge id="L26_Vessel2" source="L26" target="Vessel2">
38    <ns0:data key="d0">measuring</ns0:data>
    <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
40    <ns0:data key="d2">1588259700.0</ns0:data>
    <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
42    <ns0:data key="d4">-948.0505630970001</ns0:data>
    <ns0:data key="d5">-30946952699.0</ns0:data>
44    <ns0:data key="d6">black</ns0:data>
    </ns0:edge>
46<ns0:edge id="Vessel2_pipe211" source="Vessel2" target="pipe211">
    <ns0:data key="d0">primaryProductFlow</ns0:data>
48    <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
    <ns0:data key="d2">1588259700.0</ns0:data>
50    <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
    <ns0:data key="d4">-948.0505630970001</ns0:data>
52    <ns0:data key="d5">-30946952699.0</ns0:data>
    <ns0:data key="d6">black</ns0:data>
54</ns0:edge>
    <ns0:edge id="Vessel2_pipe162" source="Vessel2" target="pipe162">
56    <ns0:data key="d0">primaryProductFlow</ns0:data>
    <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
58    <ns0:data key="d2">1588259700.0</ns0:data>
    <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
60    <ns0:data key="d4">-948.0505630970001</ns0:data>
    <ns0:data key="d5">-30946952699.0</ns0:data>
62    <ns0:data key="d6">black</ns0:data>
    </ns0:edge>
64<ns0:edge id="jun140_pipe151" source="jun140" target="pipe151">
    <ns0:data key="d0">primaryProductFlow</ns0:data>
66    <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
    <ns0:data key="d2">1588259700.0</ns0:data>
68    <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
    <ns0:data key="d4">-948.0505630970001</ns0:data>
70    <ns0:data key="d5">-30946952699.0</ns0:data>
    <ns0:data key="d6">black</ns0:data>
72</ns0:edge>
    <ns0:edge id="jun140_pipe131" source="jun140" target="pipe131">
74    <ns0:data key="d0">primaryProductFlow</ns0:data>
    <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
76    <ns0:data key="d2">1588259700.0</ns0:data>
    <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
78    <ns0:data key="d4">-948.0505630970001</ns0:data>
    <ns0:data key="d5">-30946952699.0</ns0:data>
80    <ns0:data key="d6">black</ns0:data>
    </ns0:edge>
82<ns0:edge id="jun140_pipe161" source="jun140" target="pipe161">
    <ns0:data key="d0">primaryProductFlow</ns0:data>
84    <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
    <ns0:data key="d2">1588259700.0</ns0:data>
86    <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
    <ns0:data key="d4">-948.0505630970001</ns0:data>
88    <ns0:data key="d5">-30946952699.0</ns0:data>
    <ns0:data key="d6">black</ns0:data>
90</ns0:edge>

```

```

<ns0:edge id="jun130_pipe113" source="jun130" target="pipe113">
92  <ns0:data key="d0">primaryProductFlow</ns0:data>
    <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
94  <ns0:data key="d2">1588259700.0</ns0:data>
    <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
96  <ns0:data key="d4">-948.0505630970001</ns0:data>
    <ns0:data key="d5">-30946952699.0</ns0:data>
98  <ns0:data key="d6">black</ns0:data>
</ns0:edge>
100<ns0:edge id="jun130_pipe124" source="jun130" target="pipe124">
    <ns0:data key="d0">primaryProductFlow</ns0:data>
102  <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
    <ns0:data key="d2">1588259700.0</ns0:data>
104  <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
    <ns0:data key="d4">-948.0505630970001</ns0:data>
106  <ns0:data key="d5">-30946952699.0</ns0:data>
    <ns0:data key="d6">black</ns0:data>
108</ns0:edge>
    <ns0:edge id="jun130_pipe131" source="jun130" target="pipe131">
110  <ns0:data key="d0">primaryProductFlow</ns0:data>
    <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
112  <ns0:data key="d2">1588259700.0</ns0:data>
    <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
114  <ns0:data key="d4">-948.0505630970001</ns0:data>
    <ns0:data key="d5">-30946952699.0</ns0:data>
116  <ns0:data key="d6">black</ns0:data>
</ns0:edge>
118<ns0:edge id="N13_pipe121" source="N13" target="pipe121">
    <ns0:data key="d0">primaryProductFlow</ns0:data>
120  <ns0:data key="d7">pumpOff</ns0:data>
    <ns0:data key="d1">2020-04-30 17:14:04.686219</ns0:data>
122  <ns0:data key="d2">1588259700.0</ns0:data>
    <ns0:data key="d3">2020-04-30 17:16:41.343148</ns0:data>
124  <ns0:data key="d4">-55.31378102302551</ns0:data>
    <ns0:data key="d5">-101.34314799308777</ns0:data>
126  <ns0:data key="d6">red</ns0:data>
</ns0:edge>
128<ns0:edge id="N13_pipe122" source="N13" target="pipe122">
    <ns0:data key="d0">primaryProductFlow</ns0:data>
130  <ns0:data key="d7">pumpOff</ns0:data>
    <ns0:data key="d1">2020-04-30 17:14:04.686219</ns0:data>
132  <ns0:data key="d2">1588259700.0</ns0:data>
    <ns0:data key="d3">2020-04-30 17:16:41.343148</ns0:data>
134  <ns0:data key="d4">-55.31378102302551</ns0:data>
    <ns0:data key="d5">-101.34314799308777</ns0:data>
136  <ns0:data key="d6">red</ns0:data>
</ns0:edge>
138<ns0:edge id="pipe122_Heater43" source="pipe122" target="Heater43">
    <ns0:data key="d0">primaryProductFlow</ns0:data>
140  <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
    <ns0:data key="d2">1588259700.0</ns0:data>
142  <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
    <ns0:data key="d4">-948.0505630970001</ns0:data>
144  <ns0:data key="d5">-30946952699.0</ns0:data>
    <ns0:data key="d6">black</ns0:data>
146</ns0:edge>

```

```

<ns0:edge id="Heater43_pipe123" source="Heater43" target="pipe123">
148 <ns0:data key="d0">primaryProductFlow</ns0:data>
<ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
150 <ns0:data key="d2">1588259700.0</ns0:data>
<ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
152 <ns0:data key="d4">-948.0505630970001</ns0:data>
<ns0:data key="d5">-30946952699.0</ns0:data>
154 <ns0:data key="d6">black</ns0:data>
</ns0:edge>
156 <ns0:edge id="pipe123_Y16" source="pipe123" target="Y16">
<ns0:data key="d0">primaryProductFlow</ns0:data>
158 <ns0:data key="d7">valveOpen</ns0:data>
<ns0:data key="d1">2020-04-30 17:14:16.874790</ns0:data>
160 <ns0:data key="d2">1588259700.0</ns0:data>
<ns0:data key="d3">2020-04-30 17:16:49.538005</ns0:data>
162 <ns0:data key="d4">-43.12521004676819</ns0:data>
<ns0:data key="d5">-109.53800511360168</ns0:data>
164 <ns0:data key="d6">red</ns0:data>
</ns0:edge>
166 <ns0:edge id="Y16_pipe124" source="Y16" target="pipe124">
<ns0:data key="d0">primaryProductFlow</ns0:data>
168 <ns0:data key="d7">valveOpen</ns0:data>
<ns0:data key="d1">2020-04-30 17:14:16.874790</ns0:data>
170 <ns0:data key="d2">1588259700.0</ns0:data>
<ns0:data key="d3">2020-04-30 17:16:49.538005</ns0:data>
172 <ns0:data key="d4">-43.12521004676819</ns0:data>
<ns0:data key="d5">-109.53800511360168</ns0:data>
174 <ns0:data key="d6">red</ns0:data>
</ns0:edge>
176 <ns0:edge id="pipe161_Y25" source="pipe161" target="Y25">
<ns0:data key="d0">primaryProductFlow</ns0:data>
178 <ns0:data key="d7">valveClose</ns0:data>
<ns0:data key="d1">2020-04-30 17:13:54.511673</ns0:data>
180 <ns0:data key="d2">1588259700.0</ns0:data>
<ns0:data key="d3">2020-04-30 17:19:24.263013</ns0:data>
182 <ns0:data key="d4">-65.48832702636719</ns0:data>
<ns0:data key="d5">-264.26301288604736</ns0:data>
184 <ns0:data key="d6">red</ns0:data>
</ns0:edge>
186 <ns0:edge id="Y25_pipe162" source="Y25" target="pipe162">
<ns0:data key="d0">primaryProductFlow</ns0:data>
188 <ns0:data key="d7">valveClose</ns0:data>
<ns0:data key="d1">2020-04-30 17:13:54.511673</ns0:data>
190 <ns0:data key="d2">1588259700.0</ns0:data>
<ns0:data key="d3">2020-04-30 17:19:24.263013</ns0:data>
192 <ns0:data key="d4">-65.48832702636719</ns0:data>
<ns0:data key="d5">-264.26301288604736</ns0:data>
194 <ns0:data key="d6">red</ns0:data>
</ns0:edge>
196 </ns0:graph></ns0:graphml>

```

Listing A.11: GraphML-Datei der vierten Phase

```

<ns0:graphml xmlns:ns0="http://graphml.graphdrawing.org/xmlns" xmlns:xsi="http://www.w3
.org/2001/XMLSchema-instance" xsi:schemaLocation="http://graphml.graphdrawing.org/
xmlns http://graphml.graphdrawing.org/xmlns/1.0/graphml.xsd"><ns0:key attr.name="
etype" attr.type="string" for="edge" id="d7" />

```

```

2<ns0:key attr.name="color" attr.type="string" for="edge" id="d6" />
  <ns0:key attr.name="dvalidUntil" attr.type="string" for="edge" id="d5" />
4<ns0:key attr.name="dvalidFrom" attr.type="string" for="edge" id="d4" />
  <ns0:key attr.name="validUntil" attr.type="string" for="edge" id="d3" />
6<ns0:key attr.name="ts" attr.type="string" for="edge" id="d2" />
  <ns0:key attr.name="validFrom" attr.type="string" for="edge" id="d1" />
8<ns0:key attr.name="rtype" attr.type="string" for="edge" id="d0" />
  <ns0:graph edgedefault="undirected"><ns0:node id="L26" />
10<ns0:node id="L26_SA" />
  <ns0:node id="Vessel2" />
12<ns0:node id="pipe211" />
  <ns0:node id="jun140" />
14<ns0:node id="pipe151" />
  <ns0:node id="Vessel1" />
16<ns0:node id="pipe101" />
  <ns0:node id="pipe121" />
18<ns0:node id="pipe111" />
  <ns0:node id="N18" />
20<ns0:node id="pipe112" />
  <ns0:node id="Y21" />
22<ns0:node id="pipe113" />
  <ns0:node id="jun130" />
24<ns0:node id="pipe124" />
  <ns0:node id="pipe131" />
26<ns0:node id="pipe161" />
  <ns0:node id="Y25" />
28<ns0:node id="pipe162" />
  <ns0:edge id="L26_L26_SA" source="L26" target="L26_SA">
30  <ns0:data key="d0">hwSwCoupling</ns0:data>
    <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
32  <ns0:data key="d2">1588259840.0</ns0:data>
    <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
34  <ns0:data key="d4">-1088.0505630970001</ns0:data>
    <ns0:data key="d5">-30946952559.0</ns0:data>
36  <ns0:data key="d6">black</ns0:data>
  </ns0:edge>
38<ns0:edge id="L26_Vessel2" source="L26" target="Vessel2">
  <ns0:data key="d0">measuring</ns0:data>
40  <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
    <ns0:data key="d2">1588259840.0</ns0:data>
42  <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
    <ns0:data key="d4">-1088.0505630970001</ns0:data>
44  <ns0:data key="d5">-30946952559.0</ns0:data>
    <ns0:data key="d6">black</ns0:data>
  </ns0:edge>
  <ns0:edge id="Vessel2_pipe211" source="Vessel2" target="pipe211">
48  <ns0:data key="d0">primaryProductFlow</ns0:data>
    <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
50  <ns0:data key="d2">1588259840.0</ns0:data>
    <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
52  <ns0:data key="d4">-1088.0505630970001</ns0:data>
    <ns0:data key="d5">-30946952559.0</ns0:data>
54  <ns0:data key="d6">black</ns0:data>
  </ns0:edge>
56<ns0:edge id="Vessel2_pipe162" source="Vessel2" target="pipe162">
  <ns0:data key="d0">primaryProductFlow</ns0:data>

```

```

58 <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
   <ns0:data key="d2">1588259840.0</ns0:data>
60 <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
   <ns0:data key="d4">-1088.0505630970001</ns0:data>
62 <ns0:data key="d5">-30946952559.0</ns0:data>
   <ns0:data key="d6">black</ns0:data>
64</ns0:edge>
   <ns0:edge id="jun140_pipe151" source="jun140" target="pipe151">
66 <ns0:data key="d0">primaryProductFlow</ns0:data>
   <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
68 <ns0:data key="d2">1588259840.0</ns0:data>
   <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
70 <ns0:data key="d4">-1088.0505630970001</ns0:data>
   <ns0:data key="d5">-30946952559.0</ns0:data>
72 <ns0:data key="d6">black</ns0:data>
   </ns0:edge>
74<ns0:edge id="jun140_pipe131" source="jun140" target="pipe131">
   <ns0:data key="d0">primaryProductFlow</ns0:data>
76 <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
   <ns0:data key="d2">1588259840.0</ns0:data>
78 <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
   <ns0:data key="d4">-1088.0505630970001</ns0:data>
80 <ns0:data key="d5">-30946952559.0</ns0:data>
   <ns0:data key="d6">black</ns0:data>
82</ns0:edge>
   <ns0:edge id="jun140_pipe161" source="jun140" target="pipe161">
84 <ns0:data key="d0">primaryProductFlow</ns0:data>
   <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
86 <ns0:data key="d2">1588259840.0</ns0:data>
   <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
88 <ns0:data key="d4">-1088.0505630970001</ns0:data>
   <ns0:data key="d5">-30946952559.0</ns0:data>
90 <ns0:data key="d6">black</ns0:data>
   </ns0:edge>
92<ns0:edge id="Vessel1_pipe101" source="Vessel1" target="pipe101">
   <ns0:data key="d0">primaryProductFlow</ns0:data>
94 <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
   <ns0:data key="d2">1588259840.0</ns0:data>
96 <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
   <ns0:data key="d4">-1088.0505630970001</ns0:data>
98 <ns0:data key="d5">-30946952559.0</ns0:data>
   <ns0:data key="d6">black</ns0:data>
100</ns0:edge>
   <ns0:edge id="Vessel1_pipe121" source="Vessel1" target="pipe121">
102 <ns0:data key="d0">primaryProductFlow</ns0:data>
   <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
104 <ns0:data key="d2">1588259840.0</ns0:data>
   <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
106 <ns0:data key="d4">-1088.0505630970001</ns0:data>
   <ns0:data key="d5">-30946952559.0</ns0:data>
108 <ns0:data key="d6">black</ns0:data>
   </ns0:edge>
110<ns0:edge id="Vessel1_pipe111" source="Vessel1" target="pipe111">
   <ns0:data key="d0">primaryProductFlow</ns0:data>
112 <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
   <ns0:data key="d2">1588259840.0</ns0:data>

```

```

114 <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
    <ns0:data key="d4">-1088.0505630970001</ns0:data>
116 <ns0:data key="d5">-30946952559.0</ns0:data>
    <ns0:data key="d6">black</ns0:data>
118</ns0:edge>
    <ns0:edge id="pipe111_N18" source="pipe111" target="N18">
120 <ns0:data key="d0">primaryProductFlow</ns0:data>
    <ns0:data key="d7">pumpOff</ns0:data>
122 <ns0:data key="d1">2020-04-30 17:15:27.037692</ns0:data>
    <ns0:data key="d2">1588259840.0</ns0:data>
124 <ns0:data key="d3">2020-04-30 17:19:07.909849</ns0:data>
    <ns0:data key="d4">-112.96230792999268</ns0:data>
126 <ns0:data key="d5">-107.90984892845154</ns0:data>
    <ns0:data key="d6">red</ns0:data>
128</ns0:edge>
    <ns0:edge id="N18_pipe112" source="N18" target="pipe112">
130 <ns0:data key="d0">primaryProductFlow</ns0:data>
    <ns0:data key="d7">pumpOn</ns0:data>
132 <ns0:data key="d1">2020-04-30 17:15:27.037692</ns0:data>
    <ns0:data key="d2">1588259840.0</ns0:data>
134 <ns0:data key="d3">2020-04-30 17:19:07.909849</ns0:data>
    <ns0:data key="d4">-112.96230792999268</ns0:data>
136 <ns0:data key="d5">-107.90984892845154</ns0:data>
    <ns0:data key="d6">red</ns0:data>
138</ns0:edge>
    <ns0:edge id="pipe112_Y21" source="pipe112" target="Y21">
140 <ns0:data key="d0">primaryProductFlow</ns0:data>
    <ns0:data key="d7">valveClose</ns0:data>
142 <ns0:data key="d1">2020-04-30 17:15:39.235964</ns0:data>
    <ns0:data key="d2">1588259840.0</ns0:data>
144 <ns0:data key="d3">2020-04-30 17:19:16.101866</ns0:data>
    <ns0:data key="d4">-100.76403594017029</ns0:data>
146 <ns0:data key="d5">-116.1018660068512</ns0:data>
    <ns0:data key="d6">red</ns0:data>
148</ns0:edge>
    <ns0:edge id="Y21_pipe113" source="Y21" target="pipe113">
150 <ns0:data key="d0">primaryProductFlow</ns0:data>
    <ns0:data key="d7">valveClose</ns0:data>
152 <ns0:data key="d1">2020-04-30 17:15:39.235964</ns0:data>
    <ns0:data key="d2">1588259840.0</ns0:data>
154 <ns0:data key="d3">2020-04-30 17:19:16.101866</ns0:data>
    <ns0:data key="d4">-100.76403594017029</ns0:data>
156 <ns0:data key="d5">-116.1018660068512</ns0:data>
    <ns0:data key="d6">red</ns0:data>
158</ns0:edge>
    <ns0:edge id="pipe113_jun130" source="pipe113" target="jun130">
160 <ns0:data key="d0">primaryProductFlow</ns0:data>
    <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
162 <ns0:data key="d2">1588259840.0</ns0:data>
    <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
164 <ns0:data key="d4">-1088.0505630970001</ns0:data>
    <ns0:data key="d5">-30946952559.0</ns0:data>
166 <ns0:data key="d6">black</ns0:data>
    </ns0:edge>
168<ns0:edge id="jun130_pipe124" source="jun130" target="pipe124">
    <ns0:data key="d0">primaryProductFlow</ns0:data>

```

```

170 <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
    <ns0:data key="d2">1588259840.0</ns0:data>
172 <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
    <ns0:data key="d4">-1088.0505630970001</ns0:data>
174 <ns0:data key="d5">-30946952559.0</ns0:data>
    <ns0:data key="d6">black</ns0:data>
176</ns0:edge>
    <ns0:edge id="jun130_pipe131" source="jun130" target="pipe131">
178 <ns0:data key="d0">primaryProductFlow</ns0:data>
    <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
180 <ns0:data key="d2">1588259840.0</ns0:data>
    <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
182 <ns0:data key="d4">-1088.0505630970001</ns0:data>
    <ns0:data key="d5">-30946952559.0</ns0:data>
184 <ns0:data key="d6">black</ns0:data>
    </ns0:edge>
186<ns0:edge id="pipe161_Y25" source="pipe161" target="Y25">
    <ns0:data key="d0">primaryProductFlow</ns0:data>
188 <ns0:data key="d7">valveClose</ns0:data>
    <ns0:data key="d1">2020-04-30 17:13:54.511673</ns0:data>
190 <ns0:data key="d2">1588259840.0</ns0:data>
    <ns0:data key="d3">2020-04-30 17:19:24.263013</ns0:data>
192 <ns0:data key="d4">-205.4883270263672</ns0:data>
    <ns0:data key="d5">-124.26301288604736</ns0:data>
194 <ns0:data key="d6">red</ns0:data>
    </ns0:edge>
196<ns0:edge id="Y25_pipe162" source="Y25" target="pipe162">
    <ns0:data key="d0">primaryProductFlow</ns0:data>
198 <ns0:data key="d7">valveClose</ns0:data>
    <ns0:data key="d1">2020-04-30 17:13:54.511673</ns0:data>
200 <ns0:data key="d2">1588259840.0</ns0:data>
    <ns0:data key="d3">2020-04-30 17:19:24.263013</ns0:data>
202 <ns0:data key="d4">-205.4883270263672</ns0:data>
    <ns0:data key="d5">-124.26301288604736</ns0:data>
204 <ns0:data key="d6">red</ns0:data>
    </ns0:edge>
206</ns0:graph></ns0:graphml>

```

Listing A.12: GraphML-Datei der fünften Phase

```

<ns0:graphml xmlns:ns0="http://graphml.graphdrawing.org/xmlns" xmlns:xsi="http://www.w3
.org/2001/XMLSchema-instance" xsi:schemaLocation="http://graphml.graphdrawing.org/
xmlns http://graphml.graphdrawing.org/xmlns/1.0/graphml.xsd"><ns0:key attr.name="
etype" attr.type="string" for="edge" id="d7" />
2<ns0:key attr.name="color" attr.type="string" for="edge" id="d6" />
<ns0:key attr.name="dvalidUntil" attr.type="string" for="edge" id="d5" />
4<ns0:key attr.name="dvalidFrom" attr.type="string" for="edge" id="d4" />
<ns0:key attr.name="validUntil" attr.type="string" for="edge" id="d3" />
6<ns0:key attr.name="ts" attr.type="string" for="edge" id="d2" />
<ns0:key attr.name="validFrom" attr.type="string" for="edge" id="d1" />
8<ns0:key attr.name="rtype" attr.type="string" for="edge" id="d0" />
<ns0:graph edgedefault="undirected"><ns0:node id="L26" />
10<ns0:node id="L26_SA" />
<ns0:node id="Vessel2" />
12<ns0:node id="pipe211" />
<ns0:node id="jun320" />
14<ns0:node id="pipe341" />

```

```
<ns0:node id="pipe331" />
16<ns0:node id="pipe311" />
<ns0:node id="Vessel3" />
18<ns0:node id="pipe152" />
<ns0:node id="Y24" />
20<ns0:node id="pipe151" />
<ns0:node id="jun140" />
22<ns0:node id="Vessel1" />
<ns0:node id="pipe101" />
24<ns0:node id="pipe121" />
<ns0:node id="pipe111" />
26<ns0:node id="N18" />
<ns0:node id="pipe112" />
28<ns0:node id="Y21" />
<ns0:node id="pipe113" />
30<ns0:node id="jun130" />
<ns0:node id="pipe124" />
32<ns0:node id="pipe131" />
<ns0:node id="pipe161" />
34<ns0:node id="Y25" />
<ns0:node id="pipe162" />
36<ns0:edge id="L26_L26_SA" source="L26" target="L26_SA">
  <ns0:data key="d0">hwSwCoupling</ns0:data>
  <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
  <ns0:data key="d2">1588259900.0</ns0:data>
  <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
  <ns0:data key="d4">-1148.0505630970001</ns0:data>
  <ns0:data key="d5">-30946952499.0</ns0:data>
  <ns0:data key="d6">black</ns0:data>
44</ns0:edge>
<ns0:edge id="L26_Vessel2" source="L26" target="Vessel2">
  <ns0:data key="d0">measuring</ns0:data>
  <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
  <ns0:data key="d2">1588259900.0</ns0:data>
  <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
  <ns0:data key="d4">-1148.0505630970001</ns0:data>
  <ns0:data key="d5">-30946952499.0</ns0:data>
  <ns0:data key="d6">black</ns0:data>
52</ns0:edge>
54<ns0:edge id="Vessel2_pipe211" source="Vessel2" target="pipe211">
  <ns0:data key="d0">primaryProductFlow</ns0:data>
  <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
  <ns0:data key="d2">1588259900.0</ns0:data>
  <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
  <ns0:data key="d4">-1148.0505630970001</ns0:data>
  <ns0:data key="d5">-30946952499.0</ns0:data>
  <ns0:data key="d6">black</ns0:data>
60</ns0:edge>
<ns0:edge id="Vessel2_pipe162" source="Vessel2" target="pipe162">
  <ns0:data key="d0">primaryProductFlow</ns0:data>
  <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
  <ns0:data key="d2">1588259900.0</ns0:data>
  <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
  <ns0:data key="d4">-1148.0505630970001</ns0:data>
  <ns0:data key="d5">-30946952499.0</ns0:data>
  <ns0:data key="d6">black</ns0:data>
70</ns0:edge>
```



```

</ns0:edge>
72<ns0:edge id="jun320_pipe341" source="jun320" target="pipe341">
  <ns0:data key="d0">primaryProductFlow</ns0:data>
74  <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
  <ns0:data key="d2">1588259900.0</ns0:data>
76  <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
  <ns0:data key="d4">-1148.0505630970001</ns0:data>
78  <ns0:data key="d5">-30946952499.0</ns0:data>
  <ns0:data key="d6">black</ns0:data>
80</ns0:edge>
  <ns0:edge id="jun320_pipe331" source="jun320" target="pipe331">
82  <ns0:data key="d0">primaryProductFlow</ns0:data>
  <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
84  <ns0:data key="d2">1588259900.0</ns0:data>
  <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
86  <ns0:data key="d4">-1148.0505630970001</ns0:data>
  <ns0:data key="d5">-30946952499.0</ns0:data>
88  <ns0:data key="d6">black</ns0:data>
  </ns0:edge>
90<ns0:edge id="jun320_pipe311" source="jun320" target="pipe311">
  <ns0:data key="d0">primaryProductFlow</ns0:data>
92  <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
  <ns0:data key="d2">1588259900.0</ns0:data>
94  <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
  <ns0:data key="d4">-1148.0505630970001</ns0:data>
96  <ns0:data key="d5">-30946952499.0</ns0:data>
  <ns0:data key="d6">black</ns0:data>
98</ns0:edge>
  <ns0:edge id="pipe311_Vessel3" source="pipe311" target="Vessel3">
100  <ns0:data key="d0">primaryProductFlow</ns0:data>
  <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
102  <ns0:data key="d2">1588259900.0</ns0:data>
  <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
104  <ns0:data key="d4">-1148.0505630970001</ns0:data>
  <ns0:data key="d5">-30946952499.0</ns0:data>
106  <ns0:data key="d6">black</ns0:data>
  </ns0:edge>
108<ns0:edge id="Vessel3_pipe152" source="Vessel3" target="pipe152">
  <ns0:data key="d0">primaryProductFlow</ns0:data>
110  <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
  <ns0:data key="d2">1588259900.0</ns0:data>
112  <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
  <ns0:data key="d4">-1148.0505630970001</ns0:data>
114  <ns0:data key="d5">-30946952499.0</ns0:data>
  <ns0:data key="d6">black</ns0:data>
116</ns0:edge>
  <ns0:edge id="pipe152_Y24" source="pipe152" target="Y24">
118  <ns0:data key="d0">primaryProductFlow</ns0:data>
  <ns0:data key="d7">valveOpen</ns0:data>
120  <ns0:data key="d1">2020-04-30 17:18:05.807487</ns0:data>
  <ns0:data key="d2">1588259900.0</ns0:data>
122  <ns0:data key="d3">2020-04-30 17:19:32.465943</ns0:data>
  <ns0:data key="d4">-14.19251298904419</ns0:data>
124  <ns0:data key="d5">-72.46594309806824</ns0:data>
  <ns0:data key="d6">red</ns0:data>
126</ns0:edge>

```

```

<ns0:edge id="Y24_pipe151" source="Y24" target="pipe151">
128 <ns0:data key="d0">primaryProductFlow</ns0:data>
<ns0:data key="d7">valveOpen</ns0:data>
130 <ns0:data key="d1">2020-04-30 17:18:05.807487</ns0:data>
<ns0:data key="d2">1588259900.0</ns0:data>
132 <ns0:data key="d3">2020-04-30 17:19:32.465943</ns0:data>
<ns0:data key="d4">-14.19251298904419</ns0:data>
134 <ns0:data key="d5">-72.46594309806824</ns0:data>
<ns0:data key="d6">red</ns0:data>
136</ns0:edge>
<ns0:edge id="pipe151_jun140" source="pipe151" target="jun140">
138 <ns0:data key="d0">primaryProductFlow</ns0:data>
<ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
140 <ns0:data key="d2">1588259900.0</ns0:data>
<ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
142 <ns0:data key="d4">-1148.0505630970001</ns0:data>
<ns0:data key="d5">-30946952499.0</ns0:data>
144 <ns0:data key="d6">black</ns0:data>
</ns0:edge>
146<ns0:edge id="jun140_pipe131" source="jun140" target="pipe131">
<ns0:data key="d0">primaryProductFlow</ns0:data>
148 <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
<ns0:data key="d2">1588259900.0</ns0:data>
150 <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
<ns0:data key="d4">-1148.0505630970001</ns0:data>
152 <ns0:data key="d5">-30946952499.0</ns0:data>
<ns0:data key="d6">black</ns0:data>
154</ns0:edge>
<ns0:edge id="jun140_pipe161" source="jun140" target="pipe161">
156 <ns0:data key="d0">primaryProductFlow</ns0:data>
<ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
158 <ns0:data key="d2">1588259900.0</ns0:data>
<ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
160 <ns0:data key="d4">-1148.0505630970001</ns0:data>
<ns0:data key="d5">-30946952499.0</ns0:data>
162 <ns0:data key="d6">black</ns0:data>
</ns0:edge>
164<ns0:edge id="Vessel1_pipe101" source="Vessel1" target="pipe101">
<ns0:data key="d0">primaryProductFlow</ns0:data>
166 <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
<ns0:data key="d2">1588259900.0</ns0:data>
168 <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
<ns0:data key="d4">-1148.0505630970001</ns0:data>
170 <ns0:data key="d5">-30946952499.0</ns0:data>
<ns0:data key="d6">black</ns0:data>
172</ns0:edge>
<ns0:edge id="Vessel1_pipe121" source="Vessel1" target="pipe121">
174 <ns0:data key="d0">primaryProductFlow</ns0:data>
<ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
176 <ns0:data key="d2">1588259900.0</ns0:data>
<ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
178 <ns0:data key="d4">-1148.0505630970001</ns0:data>
<ns0:data key="d5">-30946952499.0</ns0:data>
180 <ns0:data key="d6">black</ns0:data>
</ns0:edge>
182<ns0:edge id="Vessel1_pipe111" source="Vessel1" target="pipe111">

```

```

<ns0:data key="d0">primaryProductFlow</ns0:data>
184 <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
<ns0:data key="d2">1588259900.0</ns0:data>
186 <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
<ns0:data key="d4">-1148.0505630970001</ns0:data>
188 <ns0:data key="d5">-30946952499.0</ns0:data>
<ns0:data key="d6">black</ns0:data>
190</ns0:edge>
<ns0:edge id="pipe111_N18" source="pipe111" target="N18">
192 <ns0:data key="d0">primaryProductFlow</ns0:data>
<ns0:data key="d7">pumpOff</ns0:data>
194 <ns0:data key="d1">2020-04-30 17:15:27.037692</ns0:data>
<ns0:data key="d2">1588259900.0</ns0:data>
196 <ns0:data key="d3">2020-04-30 17:19:07.909849</ns0:data>
<ns0:data key="d4">-172.96230792999268</ns0:data>
198 <ns0:data key="d5">-47.90984892845154</ns0:data>
<ns0:data key="d6">red</ns0:data>
200</ns0:edge>
<ns0:edge id="N18_pipe112" source="N18" target="pipe112">
202 <ns0:data key="d0">primaryProductFlow</ns0:data>
<ns0:data key="d7">pumpOn</ns0:data>
204 <ns0:data key="d1">2020-04-30 17:15:27.037692</ns0:data>
<ns0:data key="d2">1588259900.0</ns0:data>
206 <ns0:data key="d3">2020-04-30 17:19:07.909849</ns0:data>
<ns0:data key="d4">-172.96230792999268</ns0:data>
208 <ns0:data key="d5">-47.90984892845154</ns0:data>
<ns0:data key="d6">red</ns0:data>
210</ns0:edge>
<ns0:edge id="pipe112_Y21" source="pipe112" target="Y21">
212 <ns0:data key="d0">primaryProductFlow</ns0:data>
<ns0:data key="d7">valveClose</ns0:data>
214 <ns0:data key="d1">2020-04-30 17:15:39.235964</ns0:data>
<ns0:data key="d2">1588259900.0</ns0:data>
216 <ns0:data key="d3">2020-04-30 17:19:16.101866</ns0:data>
<ns0:data key="d4">-160.7640359401703</ns0:data>
218 <ns0:data key="d5">-56.101866006851196</ns0:data>
<ns0:data key="d6">red</ns0:data>
220</ns0:edge>
<ns0:edge id="Y21_pipe113" source="Y21" target="pipe113">
222 <ns0:data key="d0">primaryProductFlow</ns0:data>
<ns0:data key="d7">valveClose</ns0:data>
224 <ns0:data key="d1">2020-04-30 17:15:39.235964</ns0:data>
<ns0:data key="d2">1588259900.0</ns0:data>
226 <ns0:data key="d3">2020-04-30 17:19:16.101866</ns0:data>
<ns0:data key="d4">-160.7640359401703</ns0:data>
228 <ns0:data key="d5">-56.101866006851196</ns0:data>
<ns0:data key="d6">red</ns0:data>
230</ns0:edge>
<ns0:edge id="pipe113_jun130" source="pipe113" target="jun130">
232 <ns0:data key="d0">primaryProductFlow</ns0:data>
<ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
234 <ns0:data key="d2">1588259900.0</ns0:data>
<ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
236 <ns0:data key="d4">-1148.0505630970001</ns0:data>
<ns0:data key="d5">-30946952499.0</ns0:data>
238 <ns0:data key="d6">black</ns0:data>

```

```

</ns0:edge>
240<ns0:edge id="jun130_pipe124" source="jun130" target="pipe124">
  <ns0:data key="d0">primaryProductFlow</ns0:data>
242  <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
  <ns0:data key="d2">1588259900.0</ns0:data>
244  <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
  <ns0:data key="d4">-1148.0505630970001</ns0:data>
246  <ns0:data key="d5">-30946952499.0</ns0:data>
  <ns0:data key="d6">black</ns0:data>
248</ns0:edge>
  <ns0:edge id="jun130_pipe131" source="jun130" target="pipe131">
250  <ns0:data key="d0">primaryProductFlow</ns0:data>
  <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
252  <ns0:data key="d2">1588259900.0</ns0:data>
  <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
254  <ns0:data key="d4">-1148.0505630970001</ns0:data>
  <ns0:data key="d5">-30946952499.0</ns0:data>
256  <ns0:data key="d6">black</ns0:data>
</ns0:edge>
258<ns0:edge id="pipe161_Y25" source="pipe161" target="Y25">
  <ns0:data key="d0">primaryProductFlow</ns0:data>
260  <ns0:data key="d7">valveClose</ns0:data>
  <ns0:data key="d1">2020-04-30 17:13:54.511673</ns0:data>
262  <ns0:data key="d2">1588259900.0</ns0:data>
  <ns0:data key="d3">2020-04-30 17:19:24.263013</ns0:data>
264  <ns0:data key="d4">-265.4883270263672</ns0:data>
  <ns0:data key="d5">-64.26301288604736</ns0:data>
266  <ns0:data key="d6">red</ns0:data>
</ns0:edge>
268<ns0:edge id="Y25_pipe162" source="Y25" target="pipe162">
  <ns0:data key="d0">primaryProductFlow</ns0:data>
270  <ns0:data key="d7">valveClose</ns0:data>
  <ns0:data key="d1">2020-04-30 17:13:54.511673</ns0:data>
272  <ns0:data key="d2">1588259900.0</ns0:data>
  <ns0:data key="d3">2020-04-30 17:19:24.263013</ns0:data>
274  <ns0:data key="d4">-265.4883270263672</ns0:data>
  <ns0:data key="d5">-64.26301288604736</ns0:data>
276  <ns0:data key="d6">red</ns0:data>
</ns0:edge>
278</ns0:graph></ns0:graphml>

```

Listing A.13: GraphML-Datei der sechsten Phase

```

<ns0:graphml xmlns:ns0="http://graphml.graphdrawing.org/xmlns" xmlns:xsi="http://www.w3
.org/2001/XMLSchema-instance" xsi:schemaLocation="http://graphml.graphdrawing.org/
xmlns http://graphml.graphdrawing.org/xmlns/1.0/graphml.xsd"><ns0:key attr.name="
color" attr.type="string" for="edge" id="d6" />
2<ns0:key attr.name="dvalidUntil" attr.type="string" for="edge" id="d5" />
<ns0:key attr.name="dvalidFrom" attr.type="string" for="edge" id="d4" />
4<ns0:key attr.name="validUntil" attr.type="string" for="edge" id="d3" />
<ns0:key attr.name="ts" attr.type="string" for="edge" id="d2" />
6<ns0:key attr.name="validFrom" attr.type="string" for="edge" id="d1" />
<ns0:key attr.name="rtype" attr.type="string" for="edge" id="d0" />
8<ns0:graph edgedefault="undirected"><ns0:node id="L26" />
<ns0:node id="L26_SA" />
10<ns0:node id="Vessel2" />
<ns0:node id="pipe211" />

```

```

12<ns0:node id="pipe162" />
  <ns0:edge id="L26_L26_SA" source="L26" target="L26_SA">
14  <ns0:data key="d0">hwSwCoupling</ns0:data>
    <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
16  <ns0:data key="d2">1588260000.0</ns0:data>
    <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
18  <ns0:data key="d4">-1248.0505630970001</ns0:data>
    <ns0:data key="d5">-30946952399.0</ns0:data>
20  <ns0:data key="d6">black</ns0:data>
  </ns0:edge>
22<ns0:edge id="L26_Vessel2" source="L26" target="Vessel2">
  <ns0:data key="d0">measuring</ns0:data>
24  <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
    <ns0:data key="d2">1588260000.0</ns0:data>
26  <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
    <ns0:data key="d4">-1248.0505630970001</ns0:data>
28  <ns0:data key="d5">-30946952399.0</ns0:data>
    <ns0:data key="d6">black</ns0:data>
30</ns0:edge>
  <ns0:edge id="Vessel2_pipe211" source="Vessel2" target="pipe211">
32  <ns0:data key="d0">primaryProductFlow</ns0:data>
    <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
34  <ns0:data key="d2">1588260000.0</ns0:data>
    <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
36  <ns0:data key="d4">-1248.0505630970001</ns0:data>
    <ns0:data key="d5">-30946952399.0</ns0:data>
38  <ns0:data key="d6">black</ns0:data>
  </ns0:edge>
40<ns0:edge id="Vessel2_pipe162" source="Vessel2" target="pipe162">
  <ns0:data key="d0">primaryProductFlow</ns0:data>
42  <ns0:data key="d1">2020-04-30 16:59:11.949437</ns0:data>
    <ns0:data key="d2">1588260000.0</ns0:data>
44  <ns0:data key="d3">3000-12-31 23:59:59</ns0:data>
    <ns0:data key="d4">-1248.0505630970001</ns0:data>
46  <ns0:data key="d5">-30946952399.0</ns0:data>
    <ns0:data key="d6">black</ns0:data>
48</ns0:edge>
</ns0:graph></ns0:graphml>

```

Literaturverzeichnis

- [1] FISCH, Jessica ; DIEDRICH, Christian: Methodische Untersuchung des Komplexitätsanstiegs von Produktionssystemen. In: *at – Automatisierungstechnik 2018* (2018), Nr. 66(6), S. 449–455. <http://dx.doi.org/10.1515/auto-2017-0128>. – DOI 10.1515/auto-2017-0128
- [2] KAGERMANN, Henning ; LUKAS, Wolf-Dieter ; WAHLSTER, Wolfgang: Industrie 4.0: Mit dem Internet der Dinge auf dem Weg zur 4. industriellen Revolution. In: *VDI Nachrichten* 2011 (01.04.2011), Nr. 13
- [3] PÖTTER, Thorsten ; FOLMER, Jens ; VOGEL-HEUSER, Birgit: Enabling Industrie 4.0 - Chancen und Nutzen für die Prozessindustrie. In: BAUERNHANSL, Thomas (Hrsg.) ; HOMPEL, Michael ten (Hrsg.) ; VOGEL-HEUSER, Birgit (Hrsg.): *Industrie 4.0 in Produktion, Automatisierung und Logistik*. Wiesbaden : Springer Vieweg, 2014 (SpringerLink). – ISBN 978–3–658–04682–8, S. 159–171
- [4] KLETTI, Jürgen (Hrsg.): *MES - Manufacturing Execution System: Moderne Informationstechnologie unterstützt die Wertschöpfung*. 2. Auflage. Berlin and Heidelberg : Springer Vieweg, 2015. <http://dx.doi.org/10.1007/978-3-662-46902-6>. <http://dx.doi.org/10.1007/978-3-662-46902-6>. – ISBN 978–3–662–46902–6
- [5] BITKOM E.V., VDMA E.V., ZVEI E.V. (Hrsg.): *Umsetzungsstrategie Industrie 4.0: Ergebnisbericht der Plattform Industrie 4.0*. <https://www.plattform-i40.de/PI40/Redaktion/DE/Downloads/Publikation/umsetzungsstrategie-2015.html>. Version: 2015
- [6] ENSTE, Udo ; HARTWICH, Arndt ; BENTHACK, Christina ; WIEMER, Peter ; WEISS, Iris ; HENTER, Michael: Gib den Daten einen Kontext: Echtzeit-Kontextmanagement - ein Beitrag zu NOA. In: *atp magazin* (2020), Nr. 09/2020
- [7] BUNDESMINISTERIUM FÜR WIRTSCHAFT UND ENERGIE (Hrsg.): *Fort-schreibung der Anwendungsszenarien der Plattform Industrie 4.0*. <https://www.plattform-i40.de/PI40/Redaktion/DE/Downloads/Publikation/fortschreibung-anwendungsszenarien.html>. Version: 2016
- [8] SHEARER, Colin: The CRISP-DM model: the new blueprint for data mining. In: *Journal of data warehousing* (2000), Nr. Volume 5 Number 4, 13–19. <https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&cad=rja&uact=8&ved=2ahUKEwikzvXY5rvkAhU03aQKHUO-ATMQFjAAegQIAhAC&url=https%3A%2F%2Fmineracaodedados.files.wordpress.com%2F2012%2F04%2Fthe-crisp-dm-model-the-new-blueprint-for-data-mining-shearer-colin.pdf&usg=AOvVaw0hqDgeGokXrd2xQ7B049GE>

- [9] SONNEVEND, Ilona ; DADHE, Kai: *Vortrag auf der NAMUR Hauptsitzung: Mehr Daten, mehr Konnektivität, mehr Wert*. Bad Neuenahr, 7.11.2019
- [10] DIEDRICH, Christian: Datenidentifikation, -adressierung und sematische Referenzierung. In: *Kommunikation in der Automation - KommA 2019*. 10. Jahreskolloquium - Magdeburg: Otto-von-Guericke-Universität Magdeburg, 2019
- [11] PLATTFORM INDUSTRIE 4.0 (Hrsg.): *Die Verwaltungsschale im Detail: von der Idee zum implementierbaren Konzept*. <https://www.plattform-i40.de/PI40/Redaktion/DE/Downloads/Publikation/verwaltungsschale-im-detail-pr%C3%A4sentation.html>. Version: 19.03.2019
- [12] v. TROTHA, Christian ; EPPEL, Ulrich ; NAZARI, Shaghayegh ; ENGELL, Sebastian ; SONNTAG, Christian ; BEISHEIM, Benedikt ; KRÄMER, Stefan: Betrachtung eines chemischen Verbundstandorts als System of Systems zur dezentralen Optimierung – Das notwendige Informationsmanagement. In: VDI/VDE-GESELLSCHAFT MESS- UND AUTOMATISIERUNGSTECHNIK (Hrsg.): *Automation 2017: Abstracts und Konferenzbeiträge*. Düsseldorf : VDI Verlag GmbH, 2017 (VDI-Berichte). – ISBN 9783180922935
- [13] ROBINSON, Ian ; WEBBER, Jim ; EIFREM, Emil: *Graph databases: [new opportunities for connected data]*. 2nd ed. Sebastopol, CA : O'Reilly, 2015. – ISBN 978-1-491-93200-1
- [14] SINGHAL, Amit: *Official Google Blog: Introducing the Knowledge Graph: things, not strings*. <https://googleblog.blogspot.com/2012/05/introducing-knowledge-graph-things-not.html>. Version: 16.5.2012
- [15] WILLNER, Alexander ; DIEDRICH, Christian ; BEN YOUNES, Raed ; HOHMANN, Stephan ; KRAFT, Andreas: Semantic communication between components for smart factories based on oneM2M. In: *2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, IEEE, 2017. – ISBN 978-1-5090-6505-9, S. 1-8
- [16] HILDEBRANDT, C. ; TORSLEFF, S. ; CAESAR, B. ; FAY, A.: Ontology Building for Cyber-Physical Systems: A domain expert-centric approach. In: *2018 IEEE 14th International Conference on Automation Science and Engineering (CASE)*, IEEE, 20.08.2018 - 24.08.2018. – ISBN 978-1-5386-3593-3, S. 1079-1086
- [17] USCHOLD, Michael ; GRÜNINGER, Michael: Ontologies: Principles, methods and application. In: *The Knowledge Engineering Review* (01.1996), Nr. 11
- [18] JASPERNEITE, Jürgen ; NIGGEMANN, Oliver: Systemkomplexität in der Automation beherrschen: Intelligente Assistenzsysteme unterstützen den Menschen. In: *atp edition* Bd. 9/2012, S. 36-45
- [19] DIN DEUTSCHES INSTITUT FÜR NORMUNG E.V.: *Entwurf: Ergonomie der Mensch-System-Interaktion - Teil 11: Gebrauchstauglichkeit: Begriffe und Konzepte (ISO/DIS 9241-11.2:2016)*. 01.2017

- [20] HINGERL, Viktoria: *Autonomes Fahren: Mehr Stress durch Fahrerassistenzsysteme.* <https://www.heise.de/newsticker/meldung/Autonomes-Fahren-Mehr-Stress-durch-Fahrerassistenzsysteme-3998051.html>. Version: 19.03.2018 09:55 Uhr
- [21] MILLER, Christopher A. ; PARASURAMAN, Raja: Designing for flexible interaction between humans and automation: delegation interfaces for supervisory control. In: *Human factors* 49 (2007), Nr. 1, S. 57–75. <http://dx.doi.org/10.1518/001872007779598037>. – DOI 10.1518/001872007779598037. – ISSN 0018–7208
- [22] TROTHA, Christian V. ; AZARMIPOUR, Mahvar ; EPPLE, Ulrich: Advanced Assistance Systems in the Process Industry: A Classification Attempt. In: *IECON 2018 - 44th Annual Conference of the IEEE Industrial Electronics Society*, IEEE, 2018. – ISBN 978–1–5090–6684–1, S. 3231–3236
- [23] VON TROTHA, Christian ; EPPLE, Ulrich: Assistenzsysteme in der Prozessindustrie: Ein Versuch der Klassifikation. In: *19. Leitkongress der Mess- und Automatisierungstechnik*. Düsseldorf : VDI Verlag GmbH, 2018 (VDI-Berichte). – ISBN 9783180923307, S. 529–542
- [24] KAMPERT, David ; NAZARI, Shaghayegh ; SONNTAG, Christian ; EPPLE, Ulrich ; ENGELL, Sebastian: A Framework for Simulation, Optimization and Information Management of Physically-Coupled Systems of Systems. In: *IFAC-PapersOnLine* 48 (2015), Nr. 3, S. 1553–1558. <http://dx.doi.org/10.1016/j.ifacol.2015.06.307>. – DOI 10.1016/j.ifacol.2015.06.307. – ISSN 24058963
- [25] WAGNER, Constantin ; TROTHA, Christian von ; EPPLE, Ulrich ; METZUL, Alfred ; DEBUS, Kai ; TREIVOUS, Vadim ; TARNOW, Matthias ; HELLE, Christoph: Requirements for the next generation automation solution of rolling mills. In: *IECON 2017 - 43rd Annual Conference of the IEEE Industrial Electronics Society*, IEEE, 29.10.2017 - 01.11.2017. – ISBN 978–1–5386–1127–2, S. 8377–8382
- [26] WAGNER, Constantin ; TROTHA, Christian von ; PALM, Florian ; EPPLE, Ulrich: Fundamentals for the next generation of automation solutions of the fourth industrial revolution. In: *2017 11th Asian Control Conference (ASCC)*, IEEE, 2017. – ISBN 978–1–5090–1573–3, S. 2657–2662
- [27] EPPLE, Ulrich: Operative Führung verfahrenstechnischer Betriebe. In: POLKE, Martin (Hrsg.) ; EPPLE, Ulrich (Hrsg.) ; AHRENS, Wolfgang (Hrsg.): *Prozessleittechnik*. München : Oldenbourg, 1994. – ISBN 3–486–22549–9
- [28] ISO INTERNATIONAL ORGANIZATION FOR STANDARDIZATION ; IEC INTERNATIONAL ELECTROTECHNICAL COMMISSION: *ISO/IEC 25010:2011: Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*. 03.2011
- [29] ISO INTERNATIONAL ORGANIZATION FOR STANDARDIZATION ; IEC INTERNATIONAL ELECTROTECHNICAL COMMISSION: *ISO/IEC25012:2008: Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Data quality model*. 12.2008

- [30] DIN DEUTSCHES INSTITUT FÜR NORMUNG E. V.: *DIN SPEC 40912: Kernmodelle - Beschreibung und Beispiele*. 11.2014
- [31] PALLASKE, U.: Prozessmodelle. In: POLKE, Martin (Hrsg.) ; EPPLE, Ulrich (Hrsg.) ; AHRENS, Wolfgang (Hrsg.): *Prozeßleitetchnik*. München : Oldenbourg, 1994. – ISBN 3-486-22549-9, S. 108–125
- [32] MERSCH, Tina: *Fortschritt-Berichte VDI. Reihe 8, Mess-, Steuerungs- und Regelungstechnik*. Bd. Nr. 1261: *Regelbasierte Modelltransformation in prozessleitetchnischen Laufzeitumgebungen*. Dusseldorf : VDI Verlag GmbH, 2018. – ISBN 9783185261084
- [33] EPPLE, Ulrich: Begriffliche Grundlagen der leitetchnischen Modellwelt: Teil 1: Terminologielehre, Systemmodellierung. In: *atp* 2008 (2008), Nr. 4, S. 83–91
- [34] STACHOWIAK, Herbert: *Allgemeine Modelltheorie*. Wien : Springer, 1973 <https://archive.org/details/Stachowiak1973AllgemeineModelltheorie>. – ISBN 3-211-81106-0
- [35] OMG - OBJECT MANAGEMENT GROUP: *OMG Meta Object Facility (MOF) Core Specification: Version 2.5.1*. <https://www.omg.org/spec/MOF/2.5.1>
- [36] EPPLE, Ulrich: Austausch von Anlagenplanungsdaten auf der Grundlage von Metamodellen. In: *Automatisierungstechnische Praxis* (2003), Nr. 45, S. 61–70
- [37] PRAS, A. ; SCHOENWÄELDER, J.: *RFC 3444: On the Difference between Information Models and Data Models*. <https://tools.ietf.org/html/rfc3444>. Version: 2003
- [38] PRAS, A. ; SCHONWÄLDER, J. ; BURGESS, M. ; FESTOR, O. ; PEREZ, G. M. ; STADLER, R. ; STILLER, B.: Key research challenges in network management. In: *IEEE Communications Magazine* 45 (2007), Nr. 10, S. 104–110. <http://dx.doi.org/10.1109/MCOM.2007.4342832>. – DOI 10.1109/MCOM.2007.4342832. – ISSN 0163-6804
- [39] LEVESQUE, Hector J.: Knowledge representation and reasoning. In: *Annual review of computer science* (1986), 1:255–287. <https://www.annualreviews.org/doi/pdf/10.1146/annurev.cs.01.060186.001351>
- [40] NORTH, Klaus: *Wissensorientierte Unternehmensführung: Wissensmanagement gestalten*. 6., aktualisierte und erweiterte Auflage. Wiesbaden : Springer Gabler, 2016. <http://dx.doi.org/10.1007/978-3-658-11643-9>. <http://dx.doi.org/10.1007/978-3-658-11643-9>. – ISBN 978-3-658-11643-9
- [41] VDI VEREIN DEUTSCHER INGENIEURE: *VDI 5610: Wissensmanagement im Ingenieurwesen: Grundlagen, Konzepte, Vorgehen*. 03.2009
- [42] DIN DEUTSCHES INSTITUT FÜR NORMUNG E. V.: *DIN 2330: Begriffe und Benennungen - Allgemeine Grundsätze*. 07.2013

- [43] REIMER, Ulrich: *Einführung in die Wissensrepräsentation: Netzartige und schema-basierte Repräsentationsformate*. Wiesbaden : Vieweg+Teubner Verlag, 1991 (Leitfäden der angewandten Informatik). <http://dx.doi.org/10.1007/978-3-663-05970-7>. <http://dx.doi.org/10.1007/978-3-663-05970-7>. – ISBN 3519022419
- [44] RICHTER, Michael M.: *Prinzipien der Künstlichen Intelligenz: Wissensrepräsentation, Inferenz und Expertensysteme*. 2., überarbeitete und erweiterte Auflage. Wiesbaden : Vieweg+Teubner Verlag, 1992 (Leitfäden und Monographien der Informatik). <http://dx.doi.org/10.1007/978-3-322-84870-3>. <http://dx.doi.org/10.1007/978-3-322-84870-3>. – ISBN 3519122693
- [45] STOCK, Wolfgang G.: *Wissensrepräsentation: Informationen auswerten und bereitstellen*. München : Oldenbourg, 2008. – ISBN 9783486584394
- [46] SOWA, John F.: *Semantic Networks*. <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.694.700>. Version: 1987
- [47] STUCKENSCHMIDT, Heiner: *Ontologien: Konzepte, Technologien und Anwendungen*. Berlin, Heidelberg : Springer-Verlag Berlin Heidelberg, 2009 (Informatik im Fokus). <http://dx.doi.org/10.1007/978-3-540-79333-5>. <http://dx.doi.org/10.1007/978-3-540-79333-5>. – ISBN 978-3-540-79333-5
- [48] HERDEN, Olaf ; REDENZ, Kevin: Open Source Graphdatenbanken – Konzepte und Klassifikation. Version: 2012. <https://dl.gi.de/handle/20.500.12116/17838>. In: GOLTZ, Ursula (Hrsg.) ; APPELRATH, Hans-Jürgen (Hrsg.): *Informatik 2012*. Bonn : GI, 2012 (GI-Edition lecture notes in informatics proceedings). – ISBN 978-3-88579-602-2, 334-346
- [49] MEIER, Andreas ; KAUFMANN, Michael: *SQL- & NoSQL-Datenbanken*. 8., überarbeitete und erweiterte Auflage 2016. Berlin and Heidelberg : Springer Vieweg, 2016 (eXamen.press). <http://dx.doi.org/10.1007/978-3-662-47664-2>. <http://dx.doi.org/10.1007/978-3-662-47664-2>. – ISBN 978-3-662-47664-2
- [50] GORECKY, Dominic ; SCHMITT, Mathias ; LOSKYLL, Matthias: Mensch-Maschine-Interaktion im Industrie 4.0-Zeitalter. In: BAUERNHANS, Thomas (Hrsg.) ; HOMPEL, Michael ten (Hrsg.) ; VOGEL-HEUSER, Birgit (Hrsg.): *Industrie 4.0 in Produktion, Automatisierung und Logistik*. Wiesbaden : Springer Vieweg, 2014 (Springer-Link). – ISBN 978-3-658-04682-8, S. 525-542
- [51] ABOWD, Gregory D. ; DEY, Anind K. ; BROWN, Peter J. ; DAVIES, Nigel ; SMITH, Mark ; STEGGLES, Pete: Towards a Better Understanding of Context and Context-Awareness. Version: 1999. http://dx.doi.org/10.1007/3-540-48157-5_{_}29. In: GOOS, Gerhard (Hrsg.) ; HARTMANIS, Juris (Hrsg.) ; VAN LEEUWEN, Jan (Hrsg.) ; GELLERSEN, Hans-W (Hrsg.): *Handheld and Ubiquitous Computing* Bd. 1707. Berlin, Heidelberg : Springer Berlin Heidelberg, 1999. – DOI 10.1007/3-540-48157-5_{_}29. – ISBN 978-3-540-66550-2, S. 304-307
- [52] STEPHAN, Peter ; LOSKYLL, Matthias ; STAHL, Christian ; SCHLICK, Jochen: Optimierung von Instandhaltungsprozessen durch Sematische Technologien. In: DENGEL,

- Andreas (Hrsg.): *Semantische Technologien*. Heidelberg : Spektrum Akademischer Verlag, 2012. – ISBN 9783827426635
- [53] VON TROTHA, Christian ; KLEINERT, Tobias ; EPPLE, Ulrich: Ein Konzept zur Kontextualisierung von Prozessdaten: Der erste Schritt auf dem Weg zur datenzentrierten Handlung. In: *atp magazin* (2020), Nr. 10/2020
- [54] VENKATASUBRAMANIAN, Venkat ; RENGASWAMY, Raghunathan ; YIN, Kewen ; KAVURI, Surya N.: A review of process fault detection and diagnosis: Part I: Quantitative model-based methods. In: *Computers & Chemical Engineering* 27 (2003), Nr. 3, S. 293–311. [http://dx.doi.org/10.1016/S0098-1354\(02\)00160-6](http://dx.doi.org/10.1016/S0098-1354(02)00160-6). – DOI 10.1016/S0098-1354(02)00160-6. – ISSN 00981354
- [55] VENKATASUBRAMANIAN, Venkat ; RENGASWAMY, Raghunathan ; KAVURI, Surya N. ; YIN, Kewen: A review of process fault detection and diagnosis: Part III: Process history based methods. In: *Computers & Chemical Engineering* 27 (2003), Nr. 3, S. 327–346. [http://dx.doi.org/10.1016/S0098-1354\(02\)00162-X](http://dx.doi.org/10.1016/S0098-1354(02)00162-X). – DOI 10.1016/S0098-1354(02)00162-X. – ISSN 00981354
- [56] VENKATASUBRAMANIAN, Venkat ; RENGASWAMY, Raghunathan ; KAVURI, Surya N.: A review of process fault detection and diagnosis: Part II: Qualitative models and search strategies. In: *Computers & Chemical Engineering* 27 (2003), Nr. 3, S. 313–326. [http://dx.doi.org/10.1016/S0098-1354\(02\)00161-8](http://dx.doi.org/10.1016/S0098-1354(02)00161-8). – DOI 10.1016/S0098-1354(02)00161-8. – ISSN 00981354
- [57] THAMBIRAJAH, Jegatheeswaran ; BENABBAS, Lamia ; BAUER, Margret ; THORNHILL, Nina F.: Cause-and-effect analysis in chemical processes utilizing XML, plant connectivity and quantitative process history. In: *Computers & Chemical Engineering* 33 (2009), Nr. 2, S. 503–512. <http://dx.doi.org/10.1016/j.compchemeng.2008.10.002>. – DOI 10.1016/j.compchemeng.2008.10.002. – ISSN 00981354
- [58] BAUER, M.: Cause and effect analysis of chemical processes - analysis of a plant-wide disturbance. In: *IEEE Seminar on Control Loop Assessment and Diagnosis*, IEEE, 2005. – ISBN 0 86341 497 4, S. 23–30
- [59] BAUER, Margret ; COX, John W. ; CAVENESS, Michelle H. ; DOWNS, James J. ; THORNHILL, Nina F.: Finding the Direction of Disturbance Propagation in a Chemical Process Using Transfer Entropy. In: *IEEE Transactions on Control Systems Technology* 15 (2007), Nr. 1, S. 12–21. <http://dx.doi.org/10.1109/TCST.2006.883234>. – DOI 10.1109/TCST.2006.883234. – ISSN 1063-6536
- [60] ARROYO, Esteban ; FAY, Alexander ; CHIOUA, Moncef ; HOERNICKE, Mario: Integrating plant and process information as a basis for automated plant diagnosis tasks. In: *Proceedings of the 2014 IEEE Emerging Technology and Factory Automation (ETFA)*, IEEE, 2014. – ISBN 978-1-4799-4845-1, S. 1–8
- [61] BUNTE, Andreas ; DIEDRICH, Alexander ; NIGGEMANN, Oliver: Integrating semantics for diagnosis of manufacturing systems. In: *2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA)*, IEEE, 06.09.2016 - 09.09.2016. – ISBN 978-1-5090-1314-2, S. 1–8

- [62] PETERSEN, Niklas ; HALILAJ, Lavdim ; GRANGEL-GONZÁLEZ, Irlán ; LOHMANN, Steffen ; LANGE, Christoph ; AUER, Sören: Realizing an RDF-Based Information Model for a Manufacturing Company – A Case Study. Version: 2017. http://dx.doi.org/10.1007/978-3-319-68204-4_31. In: D'AMATO, Claudia (Hrsg.) ; FERNANDEZ, Miriam (Hrsg.) ; TAMMA, Valentina (Hrsg.) ; LECUE, Freddy (Hrsg.) ; CUDRÉ-MAUROUX, Philippe (Hrsg.) ; SEQUEDA, Juan (Hrsg.) ; LANGE, Christoph (Hrsg.) ; HEFLIN, Jeff (Hrsg.): *The Semantic Web – ISWC 2017* Bd. 10588. Cham : Springer International Publishing, 2017. – DOI 10.1007/978-3-319-68204-4_31. – ISBN 978-3-319-68203-7, S. 350–366
- [63] POMP, André ; PAULUS, Alexander ; KIRMSE, Andreas ; KRAUS, Vadim ; MEISEN, Tobias: Applying Semantics to Reduce the Time to Analytics within Complex Heterogeneous Infrastructures. In: *Technologies* 6 (2018), Nr. 3, S. 86. <http://dx.doi.org/10.3390/technologies6030086>. – DOI 10.3390/technologies6030086
- [64] POMP, Andre ; LIPP, Johannes ; MEISEN, Tobias: You are Missing a Concept! Enhancing Ontology-Based Data Access with Evolving Ontologies. In: *2019 IEEE 13th International Conference on Semantic Computing (ICSC)*, IEEE, 30.01.2019 - 01.02.2019. – ISBN 978-1-5386-6783-5, S. 98–105
- [65] GRAUBE, Markus ; URBAS, Leon: Modeling and Transformation of systems of systems using Linked Data. In: *IFAC-PapersOnLine* 48 (2015), Nr. 1, S. 930–931. <http://dx.doi.org/10.1016/j.ifacol.2015.05.164>. – DOI 10.1016/j.ifacol.2015.05.164. – ISSN 24058963
- [66] GRAUBE, Markus ; PFEFFER, Johannes ; ZIEGLER, Jens ; URBAS, Leon: Linked Data as integrating technology for industrial data. In: BAROLLI, Leonard (Hrsg.): *14th International Conference on Network-Based Information Systems (NBIS)*, 2011. Piscataway, NJ : IEEE, 2011. – ISBN 978-0-7695-4458-8, S. 162–167
- [67] OCKER, Felix ; PAREDIS, Christiaan J. J. ; VOGEL-HEUSER, Birgit: Applying knowledge bases to make factories smarter. In: *at - Automatisierungstechnik* 67 (2019), Nr. 6, S. 504–517. <http://dx.doi.org/10.1515/auto-2018-0138>. – DOI 10.1515/auto-2018-0138. – ISSN 0178-2312
- [68] MOSER, Thomas ; BIFFL, Stefan: Semantic Integration of Software and Systems Engineering Environments. In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 42 (2012), Nr. 1, S. 38–50. <http://dx.doi.org/10.1109/TSMCC.2011.2136377>. – DOI 10.1109/TSMCC.2011.2136377. – ISSN 1094-6977
- [69] MORDINYI, Richard ; SCHINDLER, Philipp ; BIFFL, Stefan: Evaluation of NoSQL Graph Databases for Querying and Versioning of Engineering Data in Multidisciplinary Engineering Environments. In: *2015 IEEE 20th Conference on Emerging Technologies & Factory Automation (ETFA)*. Piscataway, NJ : IEEE, 2015. – ISBN 978-1-4673-7929-8
- [70] MARQUARDT, Wolfgang ; MORBACH, Jan ; WIESNER, Andreas ; YANG, Aidong: *OntoCAPE: A Re-Usable Ontology for Chemical Process Engineer-*

- ring. Berlin, Heidelberg : Springer-Verlag Berlin Heidelberg, 2010 (RWTHeDition). <http://dx.doi.org/10.1007/978-3-642-04655-1>. <http://dx.doi.org/10.1007/978-3-642-04655-1>. – ISBN 9783642046544
- [71] YANG, Aidong ; MORBACH, Jan ; MARQUARDT, Wolfgang: From conceptualization to model generation: the roles of ontologies in process modeling. In: *Proceedings of the ... International Conference on Foundations of Computer-Aided Process Design / C. A. Floudas, R. Agrarwal / FOCAPD*. 2004, S. 591–594
- [72] WIESNER, Andreas ; MORBACH, Jan ; MARQUARDT, Wolfgang: Semantic data integration for process engineering design data. In: *Proceedings of the Tenth International Conference on Enterprise Information Systems*, SciTePress - Science and Technology Publications, 12.06.2008 - 16.06.2008. – ISBN 978-989-8111-36-4, S. 190–195
- [73] NATARAJAN, Sathish ; GHOSH, Kaushik ; SRINIVASAN, Rajagopalan: An ontology for distributed process supervision of large-scale chemical plants. In: *Computers & Chemical Engineering* 46 (2012), S. 124–140. <http://dx.doi.org/10.1016/j.compchemeng.2012.06.009>. – DOI 10.1016/j.compchemeng.2012.06.009. – ISSN 00981354
- [74] QUIRÓS, Gustavo: *AC PLT Aachener Prozessleittechnik*. Bd. 1183: *Model based decentralised automatic management of product flow paths in processing plants: Zugl.: Aachen, Techn. Univ., Diss., 2010*. Als Ms. gedr. Düsseldorf : VDI-Verl., 2011 <http://nbn-resolving.de/urn/resolver.pl?urn=urn:nbn:de:hbz:82-opus-36468>. – ISBN 9783185183089
- [75] DIN DEUTSCHES INSTITUT FÜR NORMUNG E. V.: *DIN EN 61512-1: Chargenorientierte Fahrweise - Teil 1: Modelle und Terminologie (IEC 61512-1:1997); Deutsche Fassung EN 61512-1:1999*. 01.2000
- [76] ANSI AMERICAN NATIONAL STANDARDS INSTITUTE: *ANSI/ISA 88.00.01: Batch Control Part 1: Models and Terminology*. 2010
- [77] AHRENS, W. ; POLKE, M.: Informationsstrukturen in der Leittechnik. In: POLKE, Martin (Hrsg.) ; EPPLE, Ulrich (Hrsg.) ; AHRENS, Wolfgang (Hrsg.): *Prozeßleittechnik*. München : Oldenbourg, 1994. – ISBN 3-486-22549-9
- [78] ISO INTERNATIONAL ORGANIZATION FOR STANDARDIZATION: *ISO 15926-2: Industrial automation systems and integration - Integration of life-cycle data for process plants including oil and gas production facilities - Part 2: Data model*. 15/12/2003
- [79] ISO INTERNATIONAL ORGANIZATION FOR STANDARDIZATION: *ISO 15926-1: Industrial automation systems and integration - Integration of life-cycle data for process plants including oil and gas production facilities - Part 1: Overview and fundamental principles*. 15.07.2004
- [80] HOLM, Thomas ; CHRISTIANSEN, Lars ; GORING, Markus ; JAGER, Tobias ; FAY, Alexander: ISO 15926 vs. IEC 62424 — Comparison of plant structure modeling concepts. In: *Proceedings of 2012 IEEE 17th International Conference on Emerging*

- Technologies & Factory Automation (ETFA 2012)*, IEEE, 2012. – ISBN 978–1–4673–4737–2, S. 1–8
- [81] ISO INTERNATIONAL ORGANIZATION FOR STANDARDIZATION: *ISO 15519-1: Specification for diagrams for process industry - Part 1: General rules*. 03.2010
- [82] ISO INTERNATIONAL ORGANIZATION FOR STANDARDIZATION: *ISO 15519-2: Specifications for diagrams for process industry - Part 2: Measurement and control*. 06.2015
- [83] EPPLE, Ulrich ; REMMEL, Markus ; DRUMM, Oliver: Modellbasiertes Format für RI-Informationen: Verbesserter Datenaustausch für das PLT-Engineering. In: *atp edition 2011* (2011), Nr. 1-2, S. 62–71
- [84] SCHÜLLER, Andreas ; EPPLE, Ulrich: PandIX — Exchanging P&I diagram model data. In: *Proceedings of 2012 IEEE 17th International Conference on Emerging Technologies & Factory Automation (ETFA 2012)*, IEEE, 2012. – ISBN 978–1–4673–4737–2
- [85] MERSCH, Henning ; BEHNEN, Daniel ; SCHMITZ, Dominik ; EPPLE, Ulrich ; BRECHER, Christian ; JARKE, Matthias: Gemeinsamkeiten und Unterschiede der Prozess- und Fertigungstechnik. In: *at - Automatisierungstechnik* 59 (2011), Nr. 1, S. 34. <http://dx.doi.org/10.1524/auto.2011.0891>. – DOI 10.1524/auto.2011.0891. – ISSN 0178–2312
- [86] DIN DEUTSCHES INSTITUT FÜR NORMUNG E. V.: *DIN EN ISO 10628-1: Schemata für die chemische und petrochemische Industrie - Teil 1: Spezifikation der Schemata (ISO 10628-1:2014)*. 03.2015
- [87] DIN DEUTSCHES INSTITUT FÜR NORMUNG E. V.: *DIN EN ISO 10628-2: Schemata für die chemische und petrochemische Industrie - Teil 2: Graphische Symbole (ISO 10628-2:2012)*. 03.2013
- [88] IEC INTERNATIONAL ELECTROTECHNICAL COMMISSION: *IEC 62424: Representation of process control engineering - Requests in P&I diagrams and data exchange between P&ID tools and PCE-CAE tools*. 07.2016
- [89] EPPLE, Ulrich ; REMMEL, Markus ; DRUMM, Oliver: *PandIX Modellbeschreibung: Version 5.01*. <http://www.plt.rwth-aachen.de/cms/PLT/Forschung/Projekte2/~ejwz/PandIX/>. Version: 2010
- [90] TEMMER, Heiner ; ARGAST, Ulrich ; WELKE, Richard ; THEISSEN, Manfred ; WIEDAU, Michael: Intelligenter R&I-Austausch: DEXPI hat einen wesentlichen Meilenstein erreicht. In: *atp edition 2016* (2016), Nr. 3, S. 54–65
- [91] WIEDAU, Michael ; WEDEL, Lars von ; TEMMEN, Heiner ; WELKE, Richard ; PAPA-KONSTANTINOU, Nikolaos: ENPRO Data Integration: Extending DEXPI Towards the Asset Lifecycle. In: *Chemie Ingenieur Technik* 91 (2019), Nr. 3, S. 240–255. <http://dx.doi.org/10.1002/cite.201800112>. – DOI 10.1002/cite.201800112. – ISSN 0009286X

- [92] JOREWITZ, Reiner: *Fortschrittberichte VDI Reihe 8, Mess-, Steuerungs- und Regelungstechnik*. Bd. 1201: *Eine strukturelle Beschreibungsmethodik zur automatisierten Erzeugung von Prozessbewertungen in der operativen Prozessleittechnik*: Zugl.: Aachen, Techn. Hochsch., Diss., 2011. Als Ms. gedr. Düsseldorf : VDI Verl., 2011. – ISBN 9783185201080
- [93] VOGEL-HEUSER, Birgit ; DIEDRICH, Christian ; FAY, Alexander ; JESCHKE, Sabine ; KOWALEWSKI, Stefan ; WOLLSCHLAEGER, Martin ; GÖHNER, Peter: Challenges for Software Engineering in Automation. In: *Journal of Software Engineering and Applications* 07 (2014), Nr. 05, S. 440–451. <http://dx.doi.org/10.4236/jsea.2014.75041>. – DOI 10.4236/jsea.2014.75041. – ISSN 1945–3116
- [94] IEC INTERNATIONAL ELECTROTECHNICAL COMMISSION: *IEC 61131-3: Speicher-programmierbare Steuerungen - Teil 3: Programmiersprachen*. 02.2013
- [95] IEC INTERNATIONAL ELECTROTECHNICAL COMMISSION: *IEC 61499: Funktionsbausteine für industrielle Leitsysteme*. 11.2012
- [96] WAGNER, Constantin ; EPPLE, Ulrich: Sprechende Kommandos als Grundlage moderner Prozessführungsschnittstellen. In: *Automation 2015*. Düsseldorf : VDI-Verl., 2015 (VDI-Berichte). – ISBN 978–3–18–092258–4
- [97] GROTHOFF, Julian A. ; WAGNER, Constantin A. ; EPPLE, Ulrich: *BaSys 4.0: Metamodell der Komponenten und Ihres Aufbaus; 1st ed.* <http://dx.doi.org/10.18154/RWTH-2018-225880>
- [98] PLATTFORM INDUSTRIE 4.0 ; BUNDESMINISTERIUM FÜR WIRTSCHAFT UND ENERGIE (Hrsg.): *Struktur der Verwaltungsschale: Fortentwicklung des Referenzmodells für die Industrie 4.0-Komponente*. 04.2016
- [99] DIN DEUTSCHES INSTITUT FÜR NORMUNG E. V.: *DIN SPEC 91345: Referenzarchitekturmodell Industrie 4.0 (RAMI4.0)*. <http://www.beuth.de/de/technische-regel/din-spec-91345/250940128>. Version: 04.2016
- [100] *Definition: Abnormal Situation*. <http://www.asmconsortium.net/defined/definition/Pages/default.aspx>
- [101] BAUER, Margret ; SCHLAKE, Jan C.: Changes to the automation architecture: Impact of technology on control systems algorithms. In: *2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, IEEE, 2017. – ISBN 978–1–5090–6505–9, S. 1–8
- [102] DIN DEUTSCHES INSTITUT FÜR NORMUNG E. V.: *DIN EN 62682: Alarmmanagement in der Prozessindustrie*. 02.2016
- [103] *ASM Defined: Sources*. <https://www.asmconsortium.net/defined/sources/Pages/default.aspx>
- [104] ELFAHAAM, Haitham: *Aachner Prozessleittechnik*. Bd. Nr. 1267: *A runtime adaptation concept to reinforce versatility in industrial automation*. Als Manuskript gedruckt. Düsseldorf : VDI Verlag GmbH, 2019. – ISBN 978–3–18–526708–6

- [105] GRÜNER, Sten: *Ressourcenadaptive Anwendungen für die operative Prozessleittechnik*, RWTH Aachen, Dissertation, 2017
- [106] WAGNER, Constantin: *Konzept zur Unterstützung der dezentralen Wiederverwendung in komponentenbasierten Systemen der operativen Leittechnik*, RWTH Aachen, Dissertation, 2019
- [107] AZARMIPOUR, Mahyar ; ELFAHAM, Haitham ; GRIES, Caspar ; EPPLE, Ulrich: PLC 4.0: A Control System for Industry 4.0. In: *IECON 2019 - 45th Annual Conference of the IEEE Industrial Electronics Society*, IEEE, 14.10.2019 - 17.10.2019. – ISBN 978-1-7281-4878-6, S. 5513–5518
- [108] VOGEL-HEUSER, Birgit ; DIEDRICH, Christian ; BROY, Manfred: Anforderungen an CPS aus Sicht der Automatisierungstechnik. In: *at - Automatisierungstechnik* 61 (2013), Nr. 10, 669–676. <http://dx.doi.org/10.1515/auto.2013.0061>. – DOI 10.1515/auto.2013.0061. – ISSN 0178-2312
- [109] FREYTAG, M. ; HARTMANN, K. ; HEIM, M. ; POLKE, M.: Prozeßsensorsystemtechnik. In: POLKE, Martin (Hrsg.) ; EPPLE, Ulrich (Hrsg.) ; AHRENS, Wolfgang (Hrsg.): *Prozeßleittechnik*. München : Oldenbourg, 1994. – ISBN 3-486-22549-9, S. 263–388
- [110] *Abnormal Situation Management – Effective Automation to Improve Operator Performance*. <http://www.asimconsortium.net/Documents/2007%20ASM%20Overview%20--%20MKO%20Symposium.pdf>
- [111] FELLEISEN, Michael: *Prozeßleittechnik für die Verfahrensindustrie*. München : Oldenbourg Industrieverl., 2001. – ISBN 3486270125
- [112] REISS, Thomas: *Industrie 4.0 - Zehn Thesen aus Sicht der Innovationsforschung*. https://www.isi.fraunhofer.de/content/dam/isi/dokumente/schnelleinstiege/Industrie_4_0-Thesen.pdf. Version: 12.2015
- [113] NAMUR ARBEITSKREIS: 4.20: *NE 161: Grundlagen für Remote Operation und autonomen Anlagenbetrieb*. 09.2019
- [114] DIN DEUTSCHES INSTITUT FÜR NORMUNG E. V.: *DIN SPEC 92000: Datenaustausch auf der Grundlage von Eigenschaftsausprägungsaussagen*. 09.2019
- [115] KRÜNING, Kai: Diplomarbeit: Integration eines allgemeinen Objektverwaltungssystems in die leittechnische Umgebung - Am Beispiel des Meldesystems von SIMATIC PCS7.
- [116] KRÜNING, Kai: Technischer Bericht: Implementierungsbeispiel zur Diplomarbeit “Integration eines allgemeinen Objektverwaltungssystems in die leittechnische Umgebung - Am Beispiel des Meldesystems von SIMATIC PCS7”.
- [117] *GRAKN.AI: Meet Grakn and Graql*. <https://dev.grakn.ai/docs/general/quickstart>

- [118] KAMPERT, David ; EPPLE, Ulrich: Outside-in: Simplifying systems by integrating the outside perspective. Version: 2014. <http://dx.doi.org/10.1109/SSD.2014.6808869>. In: *11th International Multi-Conference on Systems, Signals & Devices (SSD), 2014*. Piscataway, NJ : IEEE, 2014. – DOI 10.1109/SSD.2014.6808869. – ISBN 978-1-4799-3866-7, S. 1–6



**BEST
MATCH**
for
**BEST
TALENTS**

INGENIEUR.de
BEST  MATCH

powered by 

So findet Sie Ihr Traumjob!

Ingenieure aller Fachrichtungen, Absolventen und wechselwillige Professionals aufgepasst:

Sagen Sie uns, was Sie können, wollen und lieben – dann bieten Ihnen die besten Unternehmen den passenden Job für Ihr Talent. Schnell, unkompliziert, ohne Aufwand.

DAS SIND IHRE VORTEILE:

Einfache Profilerstellung | Persönliche Beratung | Passgenaue Job-Angebote |
Keine aufwändige Job-Suche | Unternehmen bewerben sich bei Ihnen | Kostenfreie Nutzung |

Transparenz: alle wichtigen Informationen zum Traumjob |

Sicher: Ihr Arbeitgeber hat keine Einsicht in Ihr Profil

JETZT ALS TALENT REGISTRIEREN:

BESTMATCH.INGENIEUR.DE

Die Reihen der Fortschritt-Berichte VDI:

- 1 Konstruktionstechnik/Maschinenelemente
 - 2 Fertigungstechnik
 - 3 Verfahrenstechnik
 - 4 Bauingenieurwesen
- 5 Grund- und Werkstoffe/Kunststoffe
 - 6 Energietechnik
 - 7 Strömungstechnik
- 8 Mess-, Steuerungs- und Regelungstechnik
 - 9 Elektronik/Mikro- und Nanotechnik
 - 10 Informatik/Kommunikation
 - 11 Schwingungstechnik
- 12 Verkehrstechnik/Fahrzeugtechnik
 - 13 Fördertechnik/Logistik
- 14 Landtechnik/Lebensmitteltechnik
 - 15 Umwelttechnik
 - 16 Technik und Wirtschaft
 - 17 Biotechnik/Medizintechnik
 - 18 Mechanik/Bruchmechanik
 - 19 Wärmetechnik/Kältetechnik
- 20 Rechnerunterstützte Verfahren (CAD, CAM, CAE CAQ, CIM ...)
 - 21 Elektrotechnik
 - 22 Mensch-Maschine-Systeme
 - 23 Technische Gebäudeausrüstung

ISBN 978-3-18-347620-6