Institut für Produktion
und Industrielles Informationsmanagement

Universität Duisburg-Essen, Campus Essen
Fakultät für Wirtschaftswissenschaften
Universitätsstraße 9, 45141 Essen
Tel.: +49 (0) 201 18 34007

Arbeitsbericht Nr. 56

zugleich

KI-LiveS-Projektbericht Nr. 10

Ein Vorgehensmodell zur Entwicklung ontologiegestützter Case-based-Reasoning-Systeme

Bornemann, J. • Heeb, T. • Zelewski, S.

– unter Mitarbeit von J. P. Schagen –

Verbundprojekt KI-LiveS: KI-Labor für verteilte und eingebettete Systeme
Förderkennzeichen: 01IS19068

GEFÖRDERT VOM

Bundesministerium
für Bildung
und Forschung

E-Mail: stephan.zelewski@pim.uni-due.de

Internet: https://www.pim.wiwi.uni-due.de/team/stephan-zelewski/

ISSN 1614-0842

Essen 2022

Zusammenfassung

Das BMBF-Forschungsprojekt „KI-LiveS“ (KI-Labor für verteilte und eingebettete Systeme) verfolgt primär das Transferziel („Third Mission“), Erkenntnisse aus der universitären Erforschung Künstlicher Intelligenz (KI) besser in der gewerblichen Wirtschaft zu verankern, um dort Entwicklungen von innovativen Produkten, insbesondere Dienstleistungen anzuregen, die den Wirtschaftsstandort Deutschland nachhaltig stärken. In diesem Kontext befasst sich der vorliegende Projektbericht Nr. 10 des KI-LiveS-Projekts mit der Erstellung eines Vorgehensmodells zur Entwicklung eines ontologiegestützten Case-based-Reasoning-Systems. Das Vorgehensmodell beruht auf der Modellierungssprache BPMN (Business Process Model and Notation), die für die betriebswirtschaftliche Modellierung von Geschäftsprozessen weit verbreitet ist und den Status eines „Quasi-Standards“ erlangt hat.

Abstract

The BMBF research project ‘KI-LiveS’ (AI laboratory for distributed and embedded systems) pursues primarly the third-mission-based aim of a more effective implementation of the university research of Artificial Intelligence (AI) into trade and industry in order to stimulate the development of innovative products, especially services, which strenghten the business location Germany sustainably. In this context, this project report no. 10 of the project “AI LiveS” deals with the construction of a process model for a ontology-supported case-based reasoning system. The process model is based on the modeling language BPMN (Business Process Model and Notation), which is widely used for business process modeling and has achieved the status of a “quasi-standard”.

Danksagung

Dieser Projektbericht entstand durch die Kooperation verschiedener Personen, die am KI-LiveS-Projekt mitwirkten. Dazu zählen neben den Verfassern des Projektberichts vor allem studentische Mitarbeiter des Instituts für Produktion und Industrielles Informationsmanagement, die mit großartigem Engagement die Verfasser bei der Erstellung dieses Projektberichts unterstützt haben. Eine besondere Hervorhebung verdient Frau Svenja Fink, die mit unermüdlichem Einsatz zum Gelingen des Projektberichts beigetragen hat.

Darüber hinaus fühlen sich die Mitglieder des KI-LiveS-Projektkonsortiums („Universitätspartner“) dem BMBF als Förderer des Drittmittel-Verbundprojekts sowie dem Deutschen Zentrum für Luft- und Raumfahrt e. V. (DLR) als zuständigem Projektträger für die großzügige finanzielle Projektförderung bzw. für die professionelle Projektbegleitung zu großem Dank verbunden.

Inhaltsverzeichnis

Abkürzungs-, Akronym- und Symbolverzeichnis

AHP	Analytic Hierarchy Process
AI	Artificial Intelligence
AM	Anforderungsmanagement
API	Application Programming Interface
Aufl.	Auflage
BMBF	Bundesministerium für Forschung und Bildung
BPMN	Business Process Model and Notation
bspw.	beispielsweise
bzw.	beziehungsweise
CBR	Case-based Reasoning
CBR-Abfrage	Case-based-Reasoning-Abfrage
CBR-Anfrage	Case-based-Reasoning-Anfrage
CBR-Anwendung	Case-based-Reasoning-Anwendung
CBR-Tool	Case-based-Reasoning-Tool
CBR-System	Case-based-Reasoning-System
CBR-Zyklus	Case-based-Reasoning-Zyklus
C2	Command and Control
DAML	DARPA Agent Markup Language
DARPA	Defense Advanced Research Projects Agency
DMN	Decision Model and Notation
DOC	Document
DOC(X)	Document (X)
Dr.	Doktor
E-Commerce	Electronic Commerce
EI	Der Eisenbahningenieur
et al.	et alii
e. V.	eingetragener Verein
f.	folgende
ff.	fortfolgende
OOD	Ontology On Demand
ggf.	gegebenenfalls
GPM	Deutsche Gesellschaft für Projektmanagement e. V.
Hrsg.	Herausgeber
html	hypertext markup language
https	hypertext transfer protocol secure

ICDM	Industrial Conference on Data Mining
IEEE	Institute of Electrical and Electronics Engineers
IR	Information Retrieval
IT	Informationstechnik
IT-Projekt	Informationstechnik-Projekt
jCORA	java based Case- and Ontology-based-Reasoning-Application
Jg.	Jahrgang
KE	Knowledge Engineering
KI	Künstliche Intelligenz
KI-LiveS	KI-Labor für verteilte und eingebettete Systeme
KI-Techniken	Künstliche-Intelligenz-Techniken
KOWIEN	Kooperatives Wissensmanagement in Engineering-Netzwerken
LKW	Lastkraftwagen
MS-Office	Microsoft-Office
Nr.	Nummer
No.	Number
Nos.	Numbers
NY	New York
OE	Ontology Engineering
OIL	Ontology Inference Layer
OrGoLo	Organisatorische Innovationen mit Good Governance in Logistik-Netzwerken
OWL	Web Ontology Language
OWL Framework	Web Ontology Language Framework
OWL Ontologie	Web Ontology Language Ontologie
OWL Reasoner	Web Ontology Language Reasoner
PA	Pennsylvania
PAKDD	Pacific-Asia Conference on Knowledge Discovery and Data Mining
PIM	Institut für Produktion und Industrielles Informationsmanagement
PPT	Powerpoint
PPT(X)	Powerpoint (X)
Prof.	Professor
RAM	Random-Access Memory
RDF	Resource Description Framework
RDFS	Resource Description Framework Schema
S.	Seite
SE	Software Engineering
s. g.	sogenannte

u. a.	unter anderem
URL	Uniform Resource Locator
VDI	Verein Deutscher Ingenieure
vgl.	vergleiche
VM	Virtual Machine
Vol.	Volume
WWW	World Wide Web
W3C	World Wide Web Consortium
XLS	Excel
XLS (X)	Excel (X)
XML	Extensible Markup Language
z. B.	zum Beispiel
%	Prozent

Abbildungsverzeichnis

1 Einführung in den Problembereich von Vorgehensmodellen zur Entwicklung ontologiegestützter Case-based-Reasoning-Systeme

1.1 Motivation für die Erstellung eines Vorgehensmodells zur Entwicklung ontologiegestützter Case-based-Reasoning-Systeme

Seit einigen Jahrzehnten gilt Wissen als einer der wichtigsten Produktionsfaktoren.[1] Insbesondere im Projektmanagement als wissenintensive Managementaufgabe spielt Wissen, vor allem in Form von Erfahrungswissen, eine herausragende Rolle.[2]

Etwa 80 % des verfügbaren Wissens in Unternehmen ist in Form von Textdokumenten abgelegt.[3] Bedingt durch die natürlichsprachliche Form der Dokumentation lässt sich das Wissen mittels konventioneller Methoden[4] nur dann mit wirtschaftlich vertretbarem Aufwand wiederverwenden, wenn sich der Umfang des abgelegten Wissens auf lediglich einige wenige Projekte erstreckt.[5] Zudem existiert das projektspezifische Erfahrungswissen in Unternehmen überwiegend „in den Köpfen der beteiligten Akteure“, sodass es nur schwer in Zukunft genutzt und wiederverwendet werden kann.

Mit zunehmendem Umfang des angesammelten Wissens entsteht ein Dilemma: Je mehr Wissen gespeichert wird, desto wertvoller wird die Menge an gespeichertem Wissen für das jeweilige Unternehmen.[6] Durch die immer größer werdende Menge an Wissen ist es jedoch gleichzeitig auch immer schwieriger, Wissen mit konventionellen Methoden für dessen Wiederverwendung aufzubereiten.

Wissen lässt sich jedoch nur in expliziter Form unternehmensweit technisch nutzen und transferieren. Implizites Wissen, wie etwa „in den Köpfen der beteiligten Akteure“ kann dagegen nur schwer transferiert und mit Methoden der elektronischen Datenverarbeitung nicht verarbeitet werden.

Um eine Wiederverwendung von Wissen im Projektmanagement zu ermöglichen, wird in der einschlägigen Fachliteratur seit geraumer Zeit der Ansatz des ontologiegestützten Case-based Reasonings – eine Kombination der Techniken Case-based Reasoning und Ontologien[7] aus der Erforschung Künstlicher Inteligenz (KI) – verfolgt. Obwohl ontologiegestütztes Case-based Reasoning in der ein-

---

1) Vgl. NONAKA/TAKEUCHI (2012), S. 21 ff.

2) Für die herausragende Rolle von Erfahrungswissen im Projektmanagement vgl. z.B. ZELEWSKI/SCHAGEN (2022), S. 1; DE TONI/PESSOT (2021), S. 541-542, 544, 546-547, 550-551 u. 553; RUIZ/TORRES/CRESPO (2021), S. 54-56 u. 61; MARTIN/EMMENEGGER/HINKELMANN et al. (2017), S. 551-552.

3) Vgl. TAN (1999), S. 1.

4) Unter konventionellen Methoden wird das „einfache“ Durchsuchen von elektronischen Medien mithilfe einzelner, rein syntaktisch definierter Wörter verstanden. Diese Volltextsuche führt zwar schnell und ohne großen Aufwand zu Ergebnissen, jedoch liefert sie keine weiteren Informationen, die im Kontext mit dem gesuchten Wort stehen. Selbst wenn die zu durchsuchenden Daten in Datenbanken abgelegt sind, können sie nur begrenzte Antworten auf Fragen liefern, da auch hier lediglich nach der syntaktischen Übereinstimmung zwischen Wörtern oder Sätzen gesucht wird, jedoch kein inhaltliches Verständnis für die Bedeutungen (Semantik) der gesuchten Begriffe bzw. Aussagen vorliegt.

5) Vgl. BEIßEL (2011), S. 2.

6) Vgl. BEIßEL (2011), S. 2. Dank der Wiederverwendung von Wissen können Erwartungen an neue Projekte besser und schneller spezifiziert werden. Hinzu kommt, dass so auch bekannte Fehlerquellen aus durchgeführten Projekten in der Zukunft eliminiert oder zumindest eingedämmt werden können.

7) Innerhalb der Wirtschaftswissenschaften existieren zahlreiche Anwendungsmöglichkeiten für Ontologien, wie etwa im Absatz-, Produktions- und Beschaffungsbereich oder bei der Integration von technischen und betriebswirtschaftlich geprägten Informationssystemen. Auch bei der überbetrieblichen Integration von Informationssystemen (beispielsweise zwischen Kunden- und Lieferantensystemen), der computergestützten Gruppenarbeit, bei elektronischen Marktplätzen und Multi-Agenten-Systemen existieren Anwendungsszenarien für Ontologien, vgl. ZELEWSKI (2002), S. 65. Ein Beispiel für eine solche Ontologie stellt die Verpackungsontologie des Verbundprojekts OrGoLo dar, vgl. KOWALSKI/ZELEWSKI (2015), S. 593 ff. Das Wissen innerhalb von Ontologien reflektiert immer die subjektiv wahrgenommene Welt von Akteuren und kann deshalb keinen objektiven Ausschnitt der Realität widerspiegeln, vgl. WAND/MONARCHI/PARSONS et al. (1995), S. 287.

schlägigen Fachliteratur zunehmend Beachtung findet, werden in der betrieblichen Praxis das ontologiegestützte Case-based Reasoning und entsprechende Ontologien und ontologiegestützte Case-based-Reasoning-Systeme (CBR-Systeme) oftmals als abstrakt und theoretisch angesehen, sodass sie in der betrieblichen Praxis kaum Beachtung finden. Die Werkzeuge des computergestützten Wissensmanagements sind zum gegenwärtigen Zeitpunkt wenig entwickelt und folglich in einem sehr geringen Umfang in der betrieblichen Praxis etabliert. Nutzer wissen oftmals nicht, wie moderne KI-Techniken im alltäglichen Projektmanagement produktiv eingesetzt werden können.

Als Realproblem stellt sich also die Herausforderung, dass ein „Bindeglied“ zwischen den akademischen Erkenntnissen hinsichtlich des ontologiegestützten Case-based Reasonings einerseits und ihrer Anwendung in Form von praxistauglichen ontologiegestützten CBR-Systemen andererseits fehlt. Der vorliegende Projektbericht[8] erörtert, wie diese Herausforderung mithilfe von Vorgehensmodellen für die Entwicklung ontologiegestützter CBR-Systeme bewältigt werden kann.

1.2 Betriebswirtschaftliche Desiderate hinsichtlich Vorgehensmodelle zur Entwicklung ontologiegestützter Case-based-Reasoning-Systeme

Um ein „Bindeglied“ zwischen vorhandenen akademischen Erkenntnissen und ihrer tatsächlichen Anwendung in Form von praxistauglichen ontologiegestützten CBR-Systemen zu schaffen, ist die Existenz eines Vorgehensmedells zur Konzipierung eines ontologiegestützten CBR-Systems wünschenswert. Mithilfe eines solchen Vorgehensmodells ist es Unternehmen möglich, ein ontologiegestütztes CBR-System zu entwickeln, zu nutzen und dessen Funktionalität in der Zukunft mindestens[9] zu erhalten.

Das Vorgehensmodell soll durch seine Benutzerfreundlichkeit einerseits gewährleisten, dass eine möglichst große Anzahl verschiedener interessierter Akteure in Unternehmen möglichst ohne Expertenwissen[10] Kenntnisse darüber erlangen kann, welche Schritte für den Aufbau eines ontologiegestützten CBR-Systems unternommen werden müssen. Andererseits sollte das Vorgehensmodell eine intuitiv verständliche Visualisierung, beispielsweise mittels der Modellierungssprache „Business Process Model and Notation“ (BPNM), bieten, sodass sämtliche Details des Vorgehensmodells unmittelbar plausibel definiert sind und das „Verständnis“, bedingt durch die globale Verbreitung einer solchen Modellierungssprache, in der Praxis weit verbreitet ist.

8) Der Projektbericht beruht im Wesentlichen auf der Masterarbeit von Herrn BORNEMANN. Herr BORNEMANN wird daher als (Haupt-)Autor des Projektberichts an erster Stelle genannt. Er wurde von der erstgenannten Koautorin des Projektberichts – Frau HEEB – während seiner Masterarbeit aus universitärer Sicht maßgeblich „gefördert und gefordert“. Der zweitgenannte Koautor – Herr ZELEWSKI – steuerte mit der Begutachtung der Masterarbeit und mit der Endredaktion des vorliegenden Projektberichts weitere Impulse bei. Hinsichtlich der inhaltlichen und redaktionellen Überarbeitung von Projektberichtsentwürfen wurde er von Herrn SCHAGEN tatkräftig unterstützt.

9) Durch die zunehmende Menge an Wissen im Zeitverlauf innerhalb eines CBR-Systems steigt dessen Funktionalität stetig an, vgl. FREUDENTHALER (2008), S. 1.

10) Als Experte wird ein Akteuer bezeichnet, der überdurchschnittliche Problemlösungsfähigkeiten innerhalb eines speziellen Fachgebiets besitzt, vgl. BEIERLE/KERN-ISBERNER (2019), S. 11.

1.3 State of the Art der verfügbaren Techniken zur Erfüllung der betriebswirtschaftlichen Desiderate hinsichtlich Vorgehensmodellen zur Entwicklung ontologiegestützter Case-based-Reasoning-Systeme

In der einschlägigen Fachliteratur gibt es mittlerweile einige prototypische Anwendungen des ontologiegestützten Case-based Reasonings für die Wiederverwendung von Erfahrungswissen im Projektmanagement.[11] Diese Prototypen zeigen für verschiedene Domänen des Projektmangements die grundsätzliche Eignung des ontologiegestützten Case-based Reasonings für die Wiederverwendung von Erfahrungswissen auf.

Weiterhin exisitieren bereits vereinzelt Ansätze für Vorgehensmodelle für die Erstellung[12] von Ontologien. Die Erstellung von Ontologien ist keine triviale Aufgabe, sondern erfordert ein systematisches Vorgehen.[13] Im Forschungsgebiet des Ontology Engineerings existiert eine Reihe von Vorgehensmodellen, die jeweils einen unterschiedlichen Fokus auf verschiedene Entwicklungsaspekte legen. Eine größere Prominenz innerhalb des Ontology Engineerings genießen die Vorgehensmodelle von NOY/MCGUINNESS[14], GRÜNINGER/FOX[15] und FERNÁNDEZ/GÓMEZ-PÉREZ/JURISTO[16]. Überblicke über verschiedene Vorgehensmodelle innerhalb des Ontology Engineerings finden sich z. B. bei GÓMEZ-PÉREZ/FERNÁNDEZ-LÓPEZ/CORCHO[17] und STUART[18].[19]

Hinsichtlich der Entwicklung von ontologiegestützten CBR-Systemen wird vor allem auf das Vorgehensmodell von BEIßEL verwiesen, das die Entwicklung eines CBR-Systems anhand eines Wasserfallmodells aufzeigt.[20] Das Vorgehensmodell besteht aus sieben Phasen („Initalisierung“, „Konzept“, „Entwurf“, „Implementierung“, „Test“, „Installation“ und „Wartung“). Diese sieben Phasen werden im Hinblick auf das ontologiegestützte Case-based Reasoning spezifiziert und beinhalten z. B. Aktivitäten wie die Auswahl von Ontologie- und CBR-Tools sowie die Erstellung eines Ähnlichkeitsalgorithmus und die Erstellung einer Ontologie. Es lässt sich kritisch anmerken, dass das Vorgehensmodell von BEIßEL hinsichtlich seiner Visualisierung nicht darüber hinaus geht, die einzelnen Phasen „lediglich“ darzustellen, Abhängigkeiten innerhalb des sequenziellen Kontrollflusses aufzuzeigen sowie mögliche Rückkopplungen zu verdeutlichen. Zudem werden einzelne Aktivitäten, wie z. B. die Erstellung der Ontologie, nicht näher spezifiziert, sodass einem Anwender des Vorgehensmodells keine weiteren Anleitungen für die konkrete Erstellung einer Ontologie offeriert werden.

11) Vgl. z. B. WEBER/HEEB/SETHUPATHY et al. (2021), S. 12-23; MARTIN/EMMENEGGER/HINKELMANN et al. (2017), S. 551-552, 560-564 (insbesondere mit S. 562) u. 567-569; KOWALSKI/BERGENRODT/ZELEWSKI (2015), S. 422-471; BEIßEL (2011), S. 9-12 u. 133-213; CHOU (2009), S. 2951-2956 u. 2958-2959.

12) Die Begriffe „Erstellung“ und „Entwicklung“ werden in diesem Projektbericht synonym verwendet.

13) Vgl. HOLSAPPLE/JOSHI (2002), S. 42.

14) Vgl. NOY/MCGUINNESS (2001), S. 4 ff.

15) Vgl. GRÜNINGER/FOX (1995), S. 2 ff.

16) Vgl. FERNÁNDEZ-LÓPEZ/GÓMEZ-PÉREZ/JURISTO (1997), S. 33 ff.

17) Vgl. GÓMEZ-PÉREZ/FERNÁNDEZ-LÓPEZ/CORCHO (2004), S. 113-153.

18) Vgl. STUART (2016), S. 97-100.

19) Eine umfangreiche Untersuchung zu ausgewählten Ansätzen des Ontology Engineerings mit einer anschließenden Bewertung der Ansätze in Hinblick auf mehrere Beurteilungskriterien bieten DITTMANN/APKE (2005), S. 277 ff.

20) Für detailiierte Ausführungen vgl. BEIßEL (2011), S. 41 ff. Die nachfolgenden zwei Sätze beziehen sich ebenfalls auf diese Quelle.

1.4 Wissenschaftliche Problemstellung im Hinblick auf Vorgehensmodelle zur Entwicklung ontologiegestützter Case-based-Reasoning-Systeme

Die Gegenüberstellung der in Kapitel 1.2 identifizierten Desiderate (in Bezug auf das in Kapitel 1.1 skizzierte Realproblem) und des in Kapitel 1.3 dargestellten State of the Art zeigt, dass zwar bereits einige Ansätze für Vorgehensmodelle zur Entwicklung von Ontologien und vereinzelt auch Ansätze für Vorgehensmodelle zur Entwicklung ontologiegestützter CBR-Systeme existieren, diese Ansätze die in Kapitel 1.2 angesprochenen betriebswirtschaftlichen Desiderate jedoch nicht vollends erfüllen. Diese Diskrepanz beruht auf mindestens zwei Aspekten. Erstens zeigt sich, dass die genannten Vorgehensmodelle nicht mittels einer weit verbreiteten Modellierungssprache, wie z. B. BPMN, spezifiziert wurden. Zweitens weist insbesondere das Vorgehensmodell von BEIßEL einen zu geringen Spezifizierungsgrad hinsichtlich der einzelnen, konkret vorzunehmenden Aktivitäten auf.

Es zeigt sich, dass aus wissenschaftlicher Sicht ein Bedarf besteht, ein neues Vorgehensmodell zur Entwicklung ontologiegestützter CBR-Systeme zu erstellen.

1.5 Ziele der Untersuchung

Aus der angeführten wissenschaftlichen Problemstellung können die folgenden Ziele für diesen Projektbericht abgeleitet werden. Es soll ein Vorgehensmodell für die Entwicklung eines ontologiegestützten CBR-Systems mittels der Modellierungssprache BPMN konzipiert werden. Dieses Vorgehensmodell wird hinsichtlich der Nutzung des Ontologie-Editors Protégé und des ontologiegestützten CBR-Systms jCORA ausgelegt. Aus der Perspektive der unternehmerischen Praxis soll ein benutzerfreundliches Vorgehensmodell so gestaltet werden, dass es in der betrieblichen Praxis des Projektmanagements auch ohne Expertenwissen angewendet werden kann.

1.6 Aufbau des Projektberichts

Der Aufbau des Projektberichts wird in der Abbildung 1 auf der nächsten Seite veranschaulicht. Das zweite Kapitel dient zur grundsätzlichen Festlegung aller Definitionen, die für die Themengebiete Vorgehensmodelle, Ontologien und Case-based Reasoning benötigt werden. Anschließend wird in Kapitel drei der konzeptionelle Hintergrund eines Vorgehensmodells zur Entwicklung eines ontologiegestützten CBR-Systems erarbeitet. Nachfolgend werden in Kapitel vier sämtliche zuvor geschilderte Wissensgebiete zusammengeführt. Außerdem wird ein allgemeines („generisches“) Vorgehensmodell für die Entwicklung ontologiegestützter CBR-Systeme konzipiert. In Kapitel fünf folgen ein Fazit und ein Ausblick auf zukünftigen Forschungsbedarf.

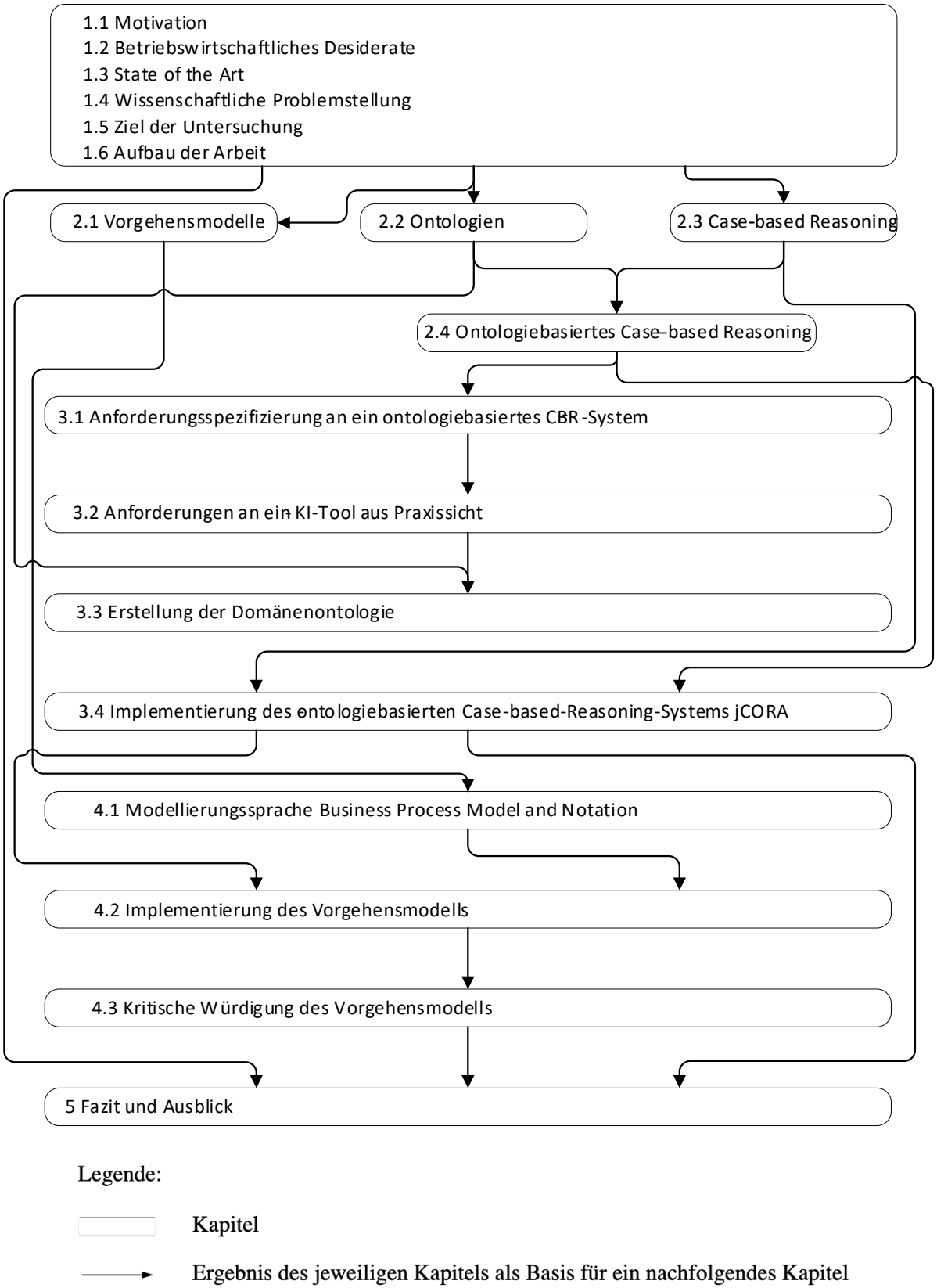

Abbildung 1: Aufbau des Projektberichts[21]

21) In der vorliegenden Abbildung werden aus optischen Gründen verkürzte Kapitelüberschriften verwendet.

2 Grundlagen für die Problembearbeitung

2.1 Vorgehensmodelle

2.1.1 Data Science als übergeordneter Rahmen von Vorgehensmodellen

Um das Themenfeld der Vorgehensmodelle definieren zu können, muss zunächst festgelegt werden, auf welche Fachgebiete sich die Vorgehensmodelle beziehen.[22] Das Fachgebiet, in dem ein Vorgehensmodell im Verlauf dieses Projektberichts erstellt werden soll, ist das Fachgebiet der Data Science. „*Data Science ist ein interdisziplinäres Fachgebiet, in welchem mit Hilfe eines wissenschaftlichen Vorgehens, semiautomatisch und unter Anwendung bestehender oder zu entwickelnder Analyseverfahren Erkenntnisse aus teils komplexen Daten extrahiert und unter Berücksichtigung gesellschaftlicher Auswirkungen nutzbar gemacht werden*“[23]. Die Interdisziplinarität ist dadurch begründet, dass häufig verschiedene Forschungsdisziplinen wie Informatik, Wirtschaftsinformatik, Künstliche Intelligenz, Linguistik, Mathematik und Betriebswirtschaftslehre, miteinander kooperieren.[24] Aufgrund verschiedener Aspekte des technischen Fortschritts[25] ist es möglich, einerseits komplexe Daten[26] zu erhalten und diese andererseits durch zunehmend komplexere Analyseverfahren auszuwerten.[27]

2.1.2 Grundlegende Charakteristika von Vorgehensmodellen

Vorgehensmodelle gelten als allgemeine Anleitungen für die Erfüllung von bestimmten, jeweils zueinander ähnlichen Aufgaben.[28] Sie dienen zur Erleicherung der Planung, Durchführung und Kontrolle von Prozessen zur Aufgabenerfüllung sowie zur Wiederverwendung von Wissen hinsichtlich dieser aufgabenerfüllenden Prozesse. Mittels Vorgehensmodellen wird der organisatorische Rahmen von Prozessen inklusive der anfallenden Aktivitäten, ihrer Reihenfolge, entstehender Artefakte, beteiligter Akteure und benötigter Ressourcen festgelegt.[29]

Das Ziel von Vorgehensmodellen lässt sich wie folgt charakterisieren: „Das *Vorgehen* bei der Entwicklung von betrieblichen Anwendungen, also der gesamte Systementwicklungsprozeß, wird auf Basis von Beschreibungen und Anleitungen durch Strukturierung aus verschiedenen Sichten als *Modell* abgebildet und somit transparent und planbar.“[30]

Vorgehensmodelle unterliegen einer statischen und einer dynamischen Sicht.[31] Die statische Sicht erfasst die interne Architektur[32] und die Dokumentationsform eines Vorgehensmodells. Die dynami-

22) Vgl. FISCHER/BISKUP/MÜLLER-LUSCHNAT (1998), S. 15.

23) SCHULZ/NEUHAUS/KAUFMANN et al. (2020), S. 6.

24) Vgl. ansatzweise SCHULZ/NEUHAUS/KAUFMANN et al. (2020), S. 7 (ohne Nennung von Wirtschaftsinformatik und Betriebswirtschaftslehre).

25) An dieser Stelle sind vor allem Teilaspekte wie die Steigerung der Rechenleistung und der Preisverfall bei Speicherbausteinen relevant.

26) Vor allem durch das Internet werden sehr große Datenmengen erzeugt. Im Jahr 2012 belief sich die tägliche Produktion bereits auf ungefähr 2,5 Exabyte (ein Exabyte entspricht einer Milliarde Gigabyte), vgl. MCAFEE/BRYNJOLFSSON (2012), S. 62.

27) Vgl. MCAFEE/BRYNJOLFSSON (2012), S. 66.

28) Vgl. DITTMANN (2007), S. 31. Der nachfolgende Satz bezieht sich ebenfalls auf diese Quellenangabe.

29) Vgl. BOEHM (1988), S. 61; DITTMANN (2007), S. 31.

30) FISCHER/BISKUP/MÜLLER-LUSCHNAT (1998), S. 3.

31) Vgl. FISCHER/BISKUP/MÜLLER-LUSCHNAT (1998), S. 16 f. Die nachfolgenden drei Sätze beziehen sich ebenfalls auf diese Quellenangabe.

32) Die Ausprägungen dieses Merkmals sind beispielsweise die Strukturierungstiefe in Phasen und Aktivitäten.

sche Sicht betrifft die spezielle, konkretisierende Anwendung eines Vorgehensmodells auf ein Projekt. Solche Konkretisierungen umfassen z. B. strukturelle Anpassungen[33] und die Individualisierung der beschriebenen Aktivitätstypen durch konkrete Projektaktivitäten.

Ein Ordnungsschema bildet die Struktur eines konkreten Vorgehensmodells ab.[34] Dies wird anhand der nachfolgenden Abbildung 2 verdeutlicht. Den Kern eines konkreten Vorgehensmodells bildet ein Regelwerk, das sowohl für das Vorgehensmodell selbst als auch für seine Strukturkomponenten Regeln definiert. Ein häufig anzutreffendes Strukturmerkmal ist die Sicht auf die verschiedenen Tätigkeitsbereiche[35]. Die Tätigkeitsbereiche werden als Aktivitäten und Ergebnisse definiert und beschreiben die wesentlichen Komponenten eines Vorgehensmodells. Weiterhin gibt das Vorgehensmodell die zu verwendenden Methoden und Werkzeuge zur Erarbeitung der intendierten Ergebnisse vor und definiert die Rollen der Akteure in den verschiedenen Tätigkeitsbereichen.

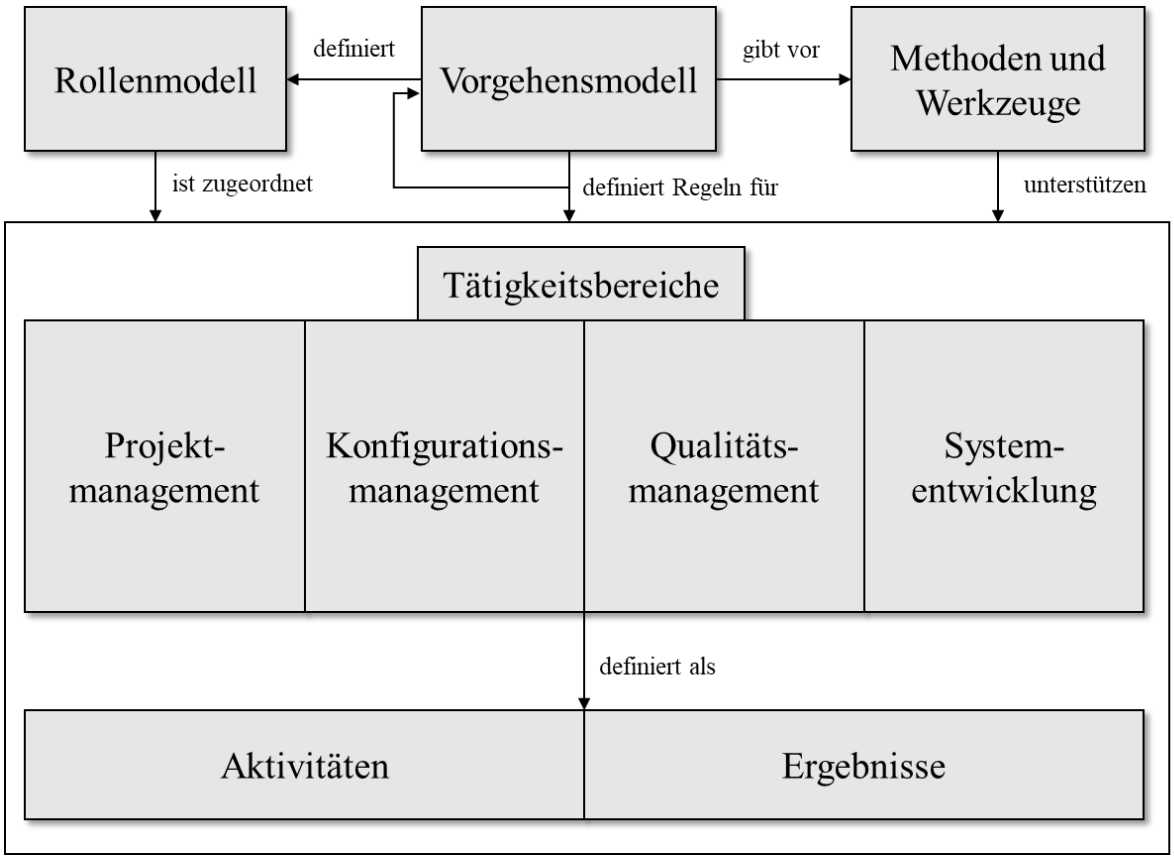

Abbildung 2: Ordnungsschema für ein Vorgehensmodell[36]

33) Diese strukturellen Anpassungen können beispielsweise in Form der genauen Spezifizierung von Aktivitäten und Ereignissen erfolgen.

34) Vgl. FISCHER/BISKUP/MÜLLER-LUSCHNAt (1998), S. 4 f. Der gesamte Absatz bezieht sich auf diese Quellenangabe.

35) Die Tätigkeitsbereiche umfassen das Projektmanagement, das Konfigurationsmanagement, das Qualitätsmanagement sowie die Systementwicklung, vgl. Abbildung 2.

36) Eigene Darstellung in Anlehnung an FISCHER/BISKUP/MÜLLER-LUSCHNAT (1998), S. 4.

2.1.3 Arten von Vorgehensmodellen

Vorgehensmodelle existieren auf allen Ebenen des Software Engineerings. Die Disziplin des Software Engineerings beinhaltet allgemein die Entwicklung von Software auf der Grundlage ingenieurmäßiger[37] Entwicklungsmethoden.[38] Das Knowledge Engineering beschreibt eine Disziplin innerhalb des Software Engineerings und konzentriert sich auf die Entwicklung wissensbasierter Systeme, die das Ziel verfolgen, computergestützt Wissen zu verarbeiten. Eine Teildisziplin des Knowledge Engineerings ist das Ontology Engineering. Es verfolgt das Ziel, Wissen spezieller Domänen zu konzeptualisieren.[39] Mit der Wissenskonzeptualisierung ist hier – bewusst im Gegensatz zum oftmals oberflächlichen Gebrauch des Ontologiebegriffs sogar in der einschlägigen Fachliteratur – keine Repräsentation des Wissens über einen Realitätsausschnitts (einer Domäne) mittels eines „repräsentationalen" Modells gemeint. Vielmehr wird als Wissenskonzeptualisierung die Spezifizierung derjenigen sprachlichen Ausdrucksmittel verstanden, die für die Konstruktion von wissensrepräsentierenden Modellen erforderlich sind oder zumindest als hilfreich empfunden werden.

Vorgehensmodelle lassen sich hinsichtlich der Art und Weise unterscheiden, wie der Weg zwischen dem Start und dem Ziel der aufgabenerfüllenden Prozesse verläuft.[40] Es lassen sich lineare, iterative und auf Prototypen basierende Vorgehensmodelle unterscheiden. Die nachfolgende Abbildung 3 verdeutlicht die zuvor genannte Unterscheidung.

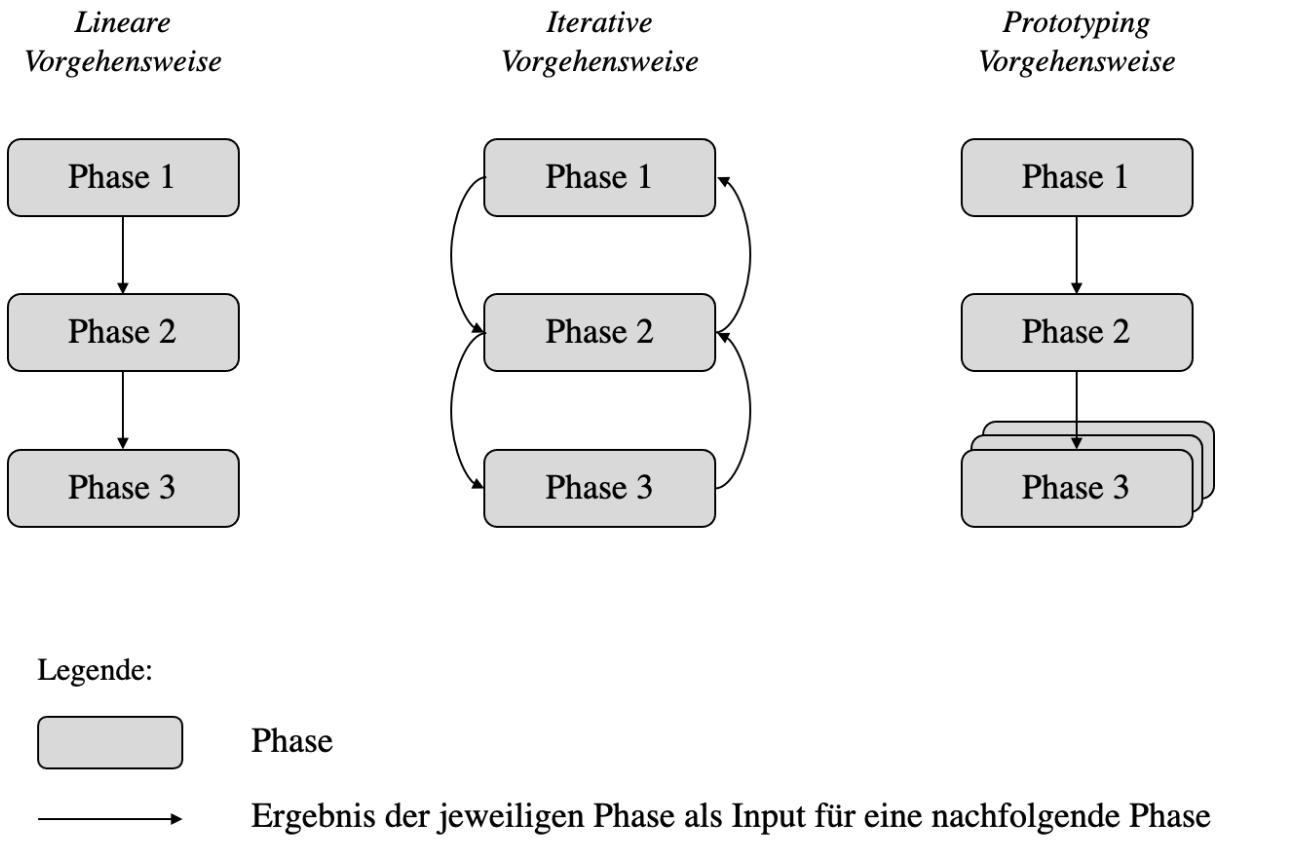

Abbildung 3: Typen von Vorgehensmodellen[41]

37) Vgl. BULLINGER/FÄHNRICH (1997), S. 11. Als Voraussetzung für eine ingenieurmäßige Softwareentwicklung sind explizit nachvollziehbare und variable Vorgehensmodelle zur Steuerung von Softwareentwicklungsprozessen zu betrachten.

38) Vgl. DITTMANN/APKE (2005), S. 277. Die nachfolgenden zwei Sätze beziehen sich ebenfalls auf diese Quelle.

39) Detaillierte Erläuterungen zum Themenbereich Ontologien befinden sich in Kapitel 2.2 ab S. 10.

40) Vgl. SCHNEIDER/DAUN/BEHRENS et al. (2006), S. 117.

41) Eigene Darstellung in Anlehnung an SCHNEIDER/DAUN/BEHRENS et al. (2006), S. 117.

Lineare Vorgehensmodelle beschreiben die zu durchlaufenden Phasen in einer sequenziellen Abfolge: Die nachfolgende Phase startet erst dann, wenn die betrachtete Phase vollständig abgeschlossen ist und deren zuvor festgelegte Ergebnisse als Input für die nachfolgende Phase vorliegen.[42] Das finale Produkt oder Projekt wird von Phase zu Phase im Sinne eines Top-Down-Ansatzes[43] immer weiter konkretisiert.[44] Vorzüge dieser Ausprägungsform sind die einfache Darstellung von Meilensteinen[45] und eine hohe Prozesstransparenz.[46] Als nachteilhaft gilt das hohe Maß an Starrheit, weil Rückkopplungen, etwa aufgrund von veränderten Rahmenbedingungen, nicht vorgesehen oder zumindest „unerwünscht“ (vgl. Abbildung 4) sind. Ein populäres Beispiel für ein lineares Vorgehensmodell[47] ist das Wasserfall-Modell in der Version von BOEHM[48], welches in der nachfolgenden Abbildung 4 dargestellt wird:

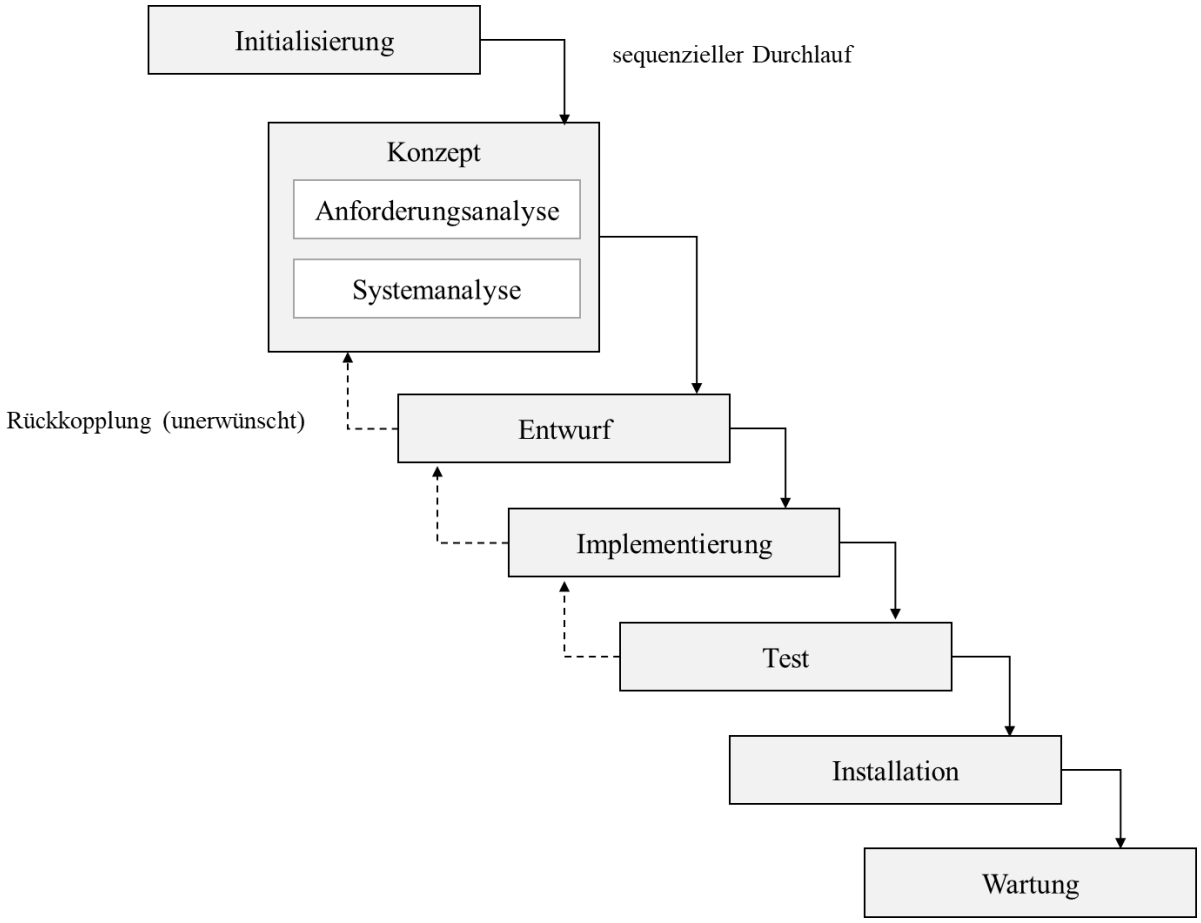

Abbildung 4: Wasserfallmodell[49]

42) Vgl. SCHNEIDER/DAUN/BEHRENS et al. (2006), S. 117.

43) Vgl. BODENDORF (1990), S. 75.

44) Vgl. GROB/SEUFERT (1996), S. 1 f.

45) Meilensteine sind im Kontext des Projektmanagements als wichtige Teilerfolge oder Etappenziele auf dem Weg zum Ziel eines Projekts zu verstehen. Jeder Meilenstein bezieht sich auf ein Ergebnis, das sich hinsichtlich seines Vorliegens versus Nichtvorliegens konkret überprüfen lässt, und das zu einem konkret festgelegten Zeipunkt eingetreten sein soll.

46) Vgl. SCHNEIDER/DAUN/BEHRENS et al. (2006), S. 117. Der Inhalt des nachfolgenden Satzes bezieht sich ebenfalls auf diese Quellenangabe.

47) Ein weiteres Beispiel ist das das sequenzielle Software-Life-Cycle-Modell. Dieses Modell steht stellvertretend für eine Reihe von Modellen, deren Gemeinsamkeit es ist, die Zeitspanne von der Entwicklung bis hin zur Absetzung eines Softwareprodukts zu erfassen, vgl. DITTMANN/APKE (2005), S. 279.

48) Vgl. BOEHM (1988), S. 61 ff.

49) Eigene Darstellung in Anlehnung an BEIßEL (2011), S. 42.

Das Wasserfallmodell beinhaltet die sequenziell nacheinander angeordneten Phasen Initialisierung, Konzept, Entwurf, Implementierung, Test, Installation und Wartung.[50] Die Entwicklung startet mit der Initialisierungsphase, in der die Ziele und erste Vorbereitungen, wie die Festlegung der notwendigen Personengruppen zur Durchführung, festgelegt werden. Die Konzeptphase beinhaltet die Anforderungs- und die Systemanalyse. Während der Anforderungsanalyse werden die funktionalen und nicht-funktionalen Anforderungen an eine Software sowie die Randbedingungen festgelegt.[51] Die Systemanalyse legt fest, was durch das Softwareprodukt geleistet werden soll. Die Entwurfphase beinhaltet die Spezifizierung aller zum fertigen System notwendigen Einzelkomponenten. Während der Implementierungsphase wird ein funktionsfähiger Prototyp einer Software erstellt, der in der Testphase daraufhin untersucht wird, ob er den zuvor festgelegten Anforderungen entspricht. Während der Installationsphase wird die fertige Software in den Praxiseinsatz überführt und innerhalb der Wartungsphase fortlaufend revidiert, um eine dauerhafte Funktionsfähigkeit garantieren zu können.

Iterative Vorgehensmodelle ergänzen lineare Vorgehensmodelle um einen Rückkopplungsmechanimus, der nachträgliche Änderungen an bereits durchlaufenden Phasen zulässt, sobald bei der Überprüfung von Zwischenergebnissen unerwartete Fehler auftreten.[52] Die betroffenen Phasen werden nach der Behebung der Fehlerursache erneut durchlaufen.[53] Ein Beispiel für ein iteratives Vorgehensmodell ist das Spiralmodell aus dem Bereich der Softwareentwicklung, das sich auch in der VDI-Richtlinie 2221 wiederfindet und ein Standardmodell im Bereich der Softwareentwicklung darstellt.

Prototypische Vorgehensmodelle versuchen Evaluationsrisiken[54] dadurch entgegenzuwirken, dass sie bereits in einer sehr frühen Phase des Entwicklungszyklus eine funktionierende Vorabversion der geplanten Software erstellen.[55] Der Prototyp dient als Grundlage für die weitere Entwicklung, indem alle Prozessbeteiligten mögliche Probleme und Wünsche diskutieren können. Diese Diskussionsphase dient als Grundlage für die nächste Generation von Prototypen, bis schließlich die finale Software vorliegt und das Entwicklungsprojekt zum Abschluss kommt.[56]

2.2 Ontologien

2.2.1 Definition von Ontologien

Der Ontologiebegriff lässt sich grundsätzlich in zwei verschiedenen wissenschaftlichen Disziplinen wiederfinden: in der Philosophie und in der Informatik (einschließlich der KI-Forschung).[57] In der Philosophie wird die Ontologie als die Lehre vom Sein betrachtet, welche schon in Aristoteles' Metaphysik Erwähnung findet.[58] Aus Sicht der Informatik werden Ontologien für Wissensrepräsentationszwecke eingesetzt, um komplexe Sachverhalte – mittels sprachlicher Ausdrucksmittel (siehe oben) –

50) Vgl. BEIßEL (2011), S. 41 ff. Der gesamte Absatz bezieht sich auf diese Quellenangabe.

51) Vgl. Kapitel 3.1

52) Vgl. GROB/SEUFERT (1996), S. 2.

53) Vgl. SCHNEIDER/DAUN/BEHRENS et al. (2006), S. 118. Der nachfolgende Satz bezieht sich ebenfalls auf diese Quellenangabe.

54) Fällt die Evaluierung der Benutzeranforderungen negativ aus, sind die Korrekturkosten umso höher, je später die Fehler im Verlauf des Entwicklungsprozesses erkannt werden.

55) Vgl. GROB/SEUFERT (1996), S. 2.

56) Vgl. BUDDE/KAUTZ/KUHLENKAMP et al. (1990), S. 90 f. Die finale Software bezieht sich in diesem Punkt nur auf das Vorgehensmodell. Software wird in den meisten Fällen während ihres Produktlebenszyklus laufend an sich verändernde Umweltbedingungen angepasst.

57) Vgl. BERGENRODT/KOWALSKI (2015), S. 8.

58) Vgl. PFUHL (2003), S. 93 f.

vollständig und realitätsnah erfassen zu können.[59] Innerhalb dieses Projektberichts wird stets der Ontologiebegriff basierend auf dem Verständnis der Informatik verwendet.

Eine häufig zu findende Definition des informationstechnischen Ontologiebegriffs[60] stammt von GRUBER: „An ontology is an explicit specification of a conceptualization.“[61] Ebenso weit verbreitet ist eine leicht erweiterte Variante der Definitions von GRUBER: „An ontology is a formal, explicit specification of a shared conceptualization.“[62] Diese beiden Definitionen benennen zwei wichtige Eigenschaften von Ontologien: Zum einen sind Ontologien Konzeptualisierungen, also abstrakte, vereinfachte Beschreibungen der Realität, und zum anderen sind diese Konzepte formal eindeutig beschrieben.[63] Diese Definitionen sind jedoch dahingehend zu kritisieren, dass nicht eindeutig definiert wird, ob unter dem Begriff der Spezifikation die Spezifikation der sprachlichen Ausdrucksmittel oder die Spezifikation einer Konzeptualisierung zu verstehen ist.[64] Da die für die Anwendung des ontologiegestützten Case-based Reasonings die Spezifikation sprachlicher Ausdrucksmittel von zentraler Bedeutung ist, wird in diesem Projektbericht die Definition von ZELEWSKI, die die Bedeutung von sprachlichen Ausdrucksmitteln betont, als Arbeitsdefinition zugrunde gelegt:„ Eine Ontologie ist eine explizite und formalsprachliche Spezifikation derjenigen sprachlichen Ausdrucksmittel, die nach Maßgabe einer von mehreren Akteuren gemeinsam verwendeten Konzeptualisierung von realen Phänomenen für die Konstruktion repräsentationaler Modelle als erforderlich erachtet werden. Die Konzeptualisierung erstreckt sich auf jene realen Phänomene, die in einem subjekt- und zweckabhängig eingegrenzten Realitätsausschnitt von den Akteuren als wahrnehmbar oder vorstellbar angesehen werden und für die Kommunikation zwischen den Akteuren benutzt oder benötigt werden.“[65]

2.2.2 Ontologiekomponenten

Ontologien bestehen aus Klassen, Instanzen, Relationen und Attributen.[66]

Um die Begriffe zur Strukturierung eines betrachteten Realitätsausschnitts zu beschreiben, werden Klassen[67] verwendet.[68] Klassen repräsentieren jeweils eine Menge von Instanzen, die allesamt Eigenschaften haben, die sie als zugehörig zu dieser Klasse charakterisieren.[69] Instanzen stehen für konkrete Elemente[70] einer Klasse.[71] Ein Beispiel für eine Klasse ist die Klasse „Projekt“, die die Menge sämtlicher konkreter Projekte umfasst. Dementsprechend stellt das „KI-LiveS-Projekt“ eine konkrete Instanz der Klasse „Projekt“ dar.

---

59) Vgl. SCHUHBAUER/FUHR/WITTMANN (2008), S. 97 (allerdings ohne den expliziten Bezug auf sprachliche Ausdrucksmittel).

60) Vgl. ZELEWSKI/BRUNS/KOWALSKI (2012), S. 158; PFUHL (2003), S. 96.

61) GRUBER (1995), S. 908.

62) STUDER/BENJAMINS/FENSEL (1998), S. 184. Für eine nahezu identische Definition siehe auch JAKUS/MILUTINOVIĆ/OMEROVIĆ et al. (2013), S. 29.

63) Vgl. BERGENRODT/KOWALSKI (2015), S. 8.

64) Vgl. ZELEWSKI/BRUNS/KOWALSKI (2012), S. 158 f.; PFUHL (2003), S. 96.

65) ZELEWSKI (2015), S. 122.

66) Vgl. BERGENRODT/KOWALSKI (2015), S. 9; JAKUS/MILUTINOVIĆ/OMEROVIĆ et al. (2013), S. 31.

67) Klassen werden im Kontext von Ontologien oftmals in synonymer Weise auch als Konzepte bezeichnet.

68) Vgl. DENGEL/BERNARDI/VAN ELST (2012), S. 65.

69) Vgl. GANDON (2010), S. 10 (dort allerdings in Bezug auf ein Konzept anstelle einer Klasse).

70) Diese Elemente werden im Kontext von Ontologien oftmals in synonymer Weise auch als Instanzen (wie bereits geschehen) oder Individuen bezeichnet.

71) Vgl. SCHUHBAUER/FUHR/WITTMANN (2008), S. 99. In der Praxis kann eine eindeutige Differenzierung zwischen Klassen und Instanzen schwierig sein. Objekte mit dem höchsten Detaillierungsgrad sollten immer als Instanzen behandelt werden, abstraktere Objekte als Klassen, vgl. KOWALSKI/ZELEWSKI (2015), S. 604.

Durch Relationen werden Beziehungen zwischen Klassen oder – im Fall von Relationselementen – zwischen klassenzugehörigen Instanzen ausgedrückt. Relationen bestehen zumeist aus einem Tripel aus Subjekt, Prädikat und Objekt.[72] Subjekt und Objekt werden durch Klassen bzw. Instanzen repräsentiert. Das Prädikat drückt die Relation im engeren Sinne aus. Es wird grundsätzlich zwischen taxonomischen und nicht-taxonomischen Relationen unterschieden.[73]

Taxonomische Relationen („ist_ein“) stellen Über- und Unterordnungsbeziehungen zwischen Klassen dar. Ein Beispiel für eine taxonomische Relation in Tripel-Form lautet z. B. „IT-Projekt ist_ein Projekt“. Der taxonomischen Relation „ist_ein“ ist zu entnehmen, dass die Klasse „IT-Projekt“ der Klasse „Projekt“ untergeordnet ist, weil ein IT-Projekt eine spezielle Ausprägung eines Projekts darstellt. In die andere Richtung gilt diese Relation jedoch nicht, weil nicht jedes Projekt ein IT-Projekt ist. Man spricht in diesem Zusammenhang auch von einer Subsumptions-Relation.[74] Über die „ist_ein“-Relation wird der Klassenzusammenhang taxonomisch strukturiert: Je tiefer sich die Ebene einer Klasse innerhalb einer Taxonomie befindet, desto spezifischer ist die Menge der von der Klasse beschriebenen Instanzen.[75] An der Spitze der Hierarchie steht die „maximal“ abstrakte Klasse „Ding“[76] oder „Sachverhalt“, die alle anderen Klassen der Ontologie subsumiert.[77]

Mittels nicht-taxonomischer Relationen können jegliche andere denkmögliche Beziehungen zwischen Klassen oder Instanzen ausgedrückt werden.

Attribute beschreiben die Eigenschaften der jeweiligen Instanzen einer bestimmten Klasse[78] und werden durch Attributwerte, z. B. in Form von Zeichenketten sowie boolschen oder numerischen Werten, konkretisiert.[79] In dem zuvor genannten Beispiel könnte ein Attribut für die Klasse „Projekt“ beispielsweise „hatGesamtkosten“ lauten. Attribute werden wie Relationen in Tripel-Form formuliert. Im Gegensatz zu Relationen wird das Objekt dieser Tripel-Form durch einen Datentyp repräsentiert. Für das Beispiel „hatGesamtkosten“ könnte als Datentyp z. B. „Integer“ gewählt werden, um auszudrücken, dass als Attributwert ein ganzzahliger numerischer Wert zu verwenden ist.

Relationen und Attribute müssen nicht für jede neue Klasse oder jede neue Instanz individuell festgelegt werden, weil ähnliche Objekte in den meisten Fällen auch ähnliche Eigenschaften miteinander teilen.[80] Wenn in einer Ontologie für die Klasse „Projekt“ z. B. das bereits vorgestellte Attribut „hatGesamtkosten“ spezifiziert wurde, steht dieses Attribut als sprachliches Ausdrucksmittel aufgrund der sogenannten „Vererbung“ ebenso für alle Subklassen der Klasse „Projekt“ und für die Menge aller Instanzen der Klasse „Projekt“ und der zugeordneten Subklassen zur Verfügung.

72) Vgl. LIM/LIU/LEE (2011), S. 56.

73) Vgl. ZELEWSKI (2005), S. 159.

74) Vgl. BERGENRODT/KOWALSKI (2015), S. 10.

75) Vgl. BERGENRODT/KOWALSKI (2015), S. 10.

76) Diese Klasse wird im späteren Verlauf dieses Projektberichts „OWL:Thing“ heißen, da dies die Standardbezeichnung in der verwendeten Software Protégé zur Erstellung von Ontologien ist. Andere Bezeichnungen sind „Entität“ oder „Seiendes“, wobei die zu erfüllende Bedingung die des zuvor erwähnten „maximalen“ Abstraktionsgrads sein muss. Zu den Bezeichnungen vgl. ABDOULLAEV (2008), S. 13.

77) Vgl. STUCKENSCHMIDT (2011), S. 186; BERGENRODT/KOWALSKI (2015), S. 10.

78) Vgl. ZELEWSKI (2005), S. 157.

79) Vgl. SCHUHBAUER/FUHR/WITTMANN (2008), S. 99. Es existiert eine Reihe von Standard-Werttypen, wie z. B. „Integer“ und „String“. Hinsichtlich weiterer Ausführungen zu den Standard-Werttypen siehe Kapitel 3.3.

80) Vgl. HAARMANN (2014), S. 34. Der gesamte Absatz bezieht sich auf diese Quellenangabe.

2.2.3 Ontologiearten

Aufgrund der vielfältigen Gegenstandsbereiche, deren Wissen sich durch Ontologien repräsentieren lassen, resultieren verschiedene Ontologiearten. Neben den Domänenontologien[81], die das Wissen eines bestimmten Ausschnitts der Realität[82] repräsentieren, existieren nach ZELEWSKI[83] noch vier weitere Arten von Ontologien. Die Commonsense-Ontologien repräsentieren Wissen, das sich nicht einer speziellen Branche zuordnen lässt, sondern als „selbstverständliches Hintergrundwissen" vorausgesetzt wird. Das durch eine Commonsense-Ontologie repräsentierte Wissen kann so umfangreich[84] sein, dass es in der KI-Forschung große Erfassungsprobleme bereitet. Daneben existieren Repräsentations- oder Meta-Ontologien, die die Ausdrucksmöglichkeiten von Repräsentations- oder Modellierungssprachen spezifizieren, Aufgaben-Ontologien zur Spezifikation von generischen Aufgabentypen sowie Methoden-Ontologien, die den Termvorrat und die Syntax zulässiger Termverknüpfungen zur Verfügung stellen, die für die Lösung einer Problemlösungsmethode notwendig sind. Neben den bereits genannten Ontologiearten führt FENSEL noch Metadaten-Ontologien an, die das Vokabular zur Beschreibung von Online-Informationsquellen repräsentieren.[85]

2.2.4 Ontologie-Repräsentationssprachen

Als Repräsentationssprache wird die maschinenlesbare Form bezeichnet, mit der ein Ontologie-Tool (die sprachlichen Ausdrucksmittel für) das in einer Domäne vorhandene Wissen repräsentiert.[86] Die Repräsentationssprache kann entweder standardisiert oder proprietär sein. Die Ausdrucksstärke einer Repräsentationssprache wird durch die Menge der behauptbaren Propositionen festgelegt. Als Propositionen zur Beurteilung der Ausdrucksstärke kann die Unterstützung von Axiomen, Meta-Klassen und Instanzen genutzt werden.[87]

Eine weit verbreitete Repräsentationssprache ist die Web Ontology Language (OWL), die auf einer Spezifikation des WORLD WIDE WEB CONSORTIUMS (W3C) beruht und eine Überarbeitung von DAML+OIL darstellt.[88] Mit OWL lassen sich Inhalte von Informationen computergestützt verarbeiten, anstatt sie nur für menschliche Akteure anzubieten. Mithilfe von zusätzlichem, von OWL bereitgestelltem Vokabular können Web-Inhalte durch Maschinen besser interpretiert werden als Web-Inhalte, welche die weit verbreiteten Datenformate XML, RDF und RDF-Schema (RDFS) verwenden.

81) Vgl. MCDANIEL/STOREY/SUGUMARAN (2018), S. 32.

82) Der wissenschaftlich etablierte Begriff des Realitätsausschnitts lässt sich umgangssprachlich auch als „Branchenbegriff" bezeichnen und erhöht so das Verständnis praxisorientierter Leser, vgl. ZELEWSKI (1999), S. 12.

83) Vgl. ZELEWSKI (1999), S. 11 f.

84) Die Commonsense-Ontologie OntoSenticNet umfasst beispielsweise 100.000 Klassen, vgl. DRAGONI/PORIA/CAMBRIA (2018), S. 77 f.

85) Vgl. FENSEL (2004), S. 5.

86) Vgl. BEIßEL (2011), S. 24. Die nachfolgenden zwei Sätze beziehen sich ebenfalls auf diese Quellenangabe.

87) Vgl. SU/ILEBREKKE (2002), S. 763.

88) Vgl. BEIßEL (2011), S. 24 ff. Der gesamte Absatz bezieht sich auf diese Quellenangabe.

2.3 Case-based Reasoning

2.3.1 Definition von Case-based Reasoning

Das Case-based Reasoning (zu Deutsch: Fallbasiertes Schließen) beschreibt eine zyklische Problemlösungsmethode, die bereits vorhandenes Wissen über die Lösung von alten Problemen wiederverwendet und auf dieser Basis Lösungsvorschläge für neue Probleme unterbreitet.[89] „A case-based reasoner solves new problems by adapting solutions that were used to solve old problems".[90] Das Case-based Reasoning ist eine Methode aus dem Umfeld der KI-Forschung[91], durch die sich beispielsweise Expertensysteme[92] – oder in „modernerer Diktion": wissensbasierte Systeme – implementieren lassen.[93] Ein Expertensystem ist ein System, *„[…] das in einem gegebenen Spezialisierungsbereich menschliche Experten in Bezug auf ihr Wissen und ihre Schlussfolgerungsfähigkeit nachbildet*."[94]

Die Basis des Expertenwissens liegt im Case-based Reasoning als Erfahrungswissen aus der Vergangenheit[95] vor und wird in der sogenannten Fallbasis[96] als Fall abgelegt. Im Kontext des Case-based Reasonings wird ein Fall als Tripel aus Fallbeschreibung, Falllösung und Fallbewertung gespeichert.[97] Die Fallbeschreibung beinhaltet die Problemstellung des Falls, während die Falllösung eine Lösung für die in der Fallbeschreibung geschilderte Problemstellung umfasst.[98] Die Fallbewertung beurteilt die Falllösung in Hinblick auf die aufgetretenen Erfolgs- und Misserfolgsfaktoren.

2.3.2 Case-based-Reasoning-Zyklus

Der von AAMODT und PLAZA entwickelte Case-based-Reasoning-Zyklus (CBR-Zyklus) besteht aus den Phasen Retrieve, Reuse, Revise und Retain.[99] Die Abfolge der genannten Schritte führt über eine neue Problemstellung zu einer Falllösung, die wiederum zu einer Erweiterung der Fallbasis führt, womit ein Zyklus vorläufig abgeschlossen ist.[100] Nachfolgend werden die einzelnen Phasen genauer erläutert und in einer BPMN-Abbildung zusammengefasst. Unter der BPMN-Abbildung findet sich eine entsprechende Legende zu den verwendeten Grafik-Symbolen.

89) Vgl. FREUDENTHALER (2008), S. 1.

90) RIESBECK/SCHANK (2013), S. 25.

91) Vgl. LIAO (2005), S. 93.

92) Vgl. BEIERLE/KERN-ISBERNER (2019), S. 11 f.

93) Vgl. BERGENRODT/KOWALSKI (2015), S. 18.

94) BEIERLE/KERN-ISBERNER (2019), S. 12.

95) Vgl. SLADE (1991), S. 42.

96) Vgl. FREUDENTHALER (2008), S. 6.

97) Vgl. BEIERLE/KERN-ISBERNER (2019), S. 171 f.; BERGENRODT/KOWALSKI (2015), S. 19; FREUDENTHALER (2008), S. 5 f.

98) Vgl. BERGENRODT/KOWALSKI (2015), S. 19. Diese Quellenangabe bezieht sich ebenfalls auf den nachfolgenden Satz.

99) Vgl. AAMODT/PLAZA (1994), S. 44. Im Deutschen bedeuten die Begriffe für die Phasen des CBR-Zyklus sinngemäß in der genannten Reihenfolge: abrufen, wiederverwenden, überarbeiten und ablegen.

100) Vgl. PFUHL (2003), S. 12.

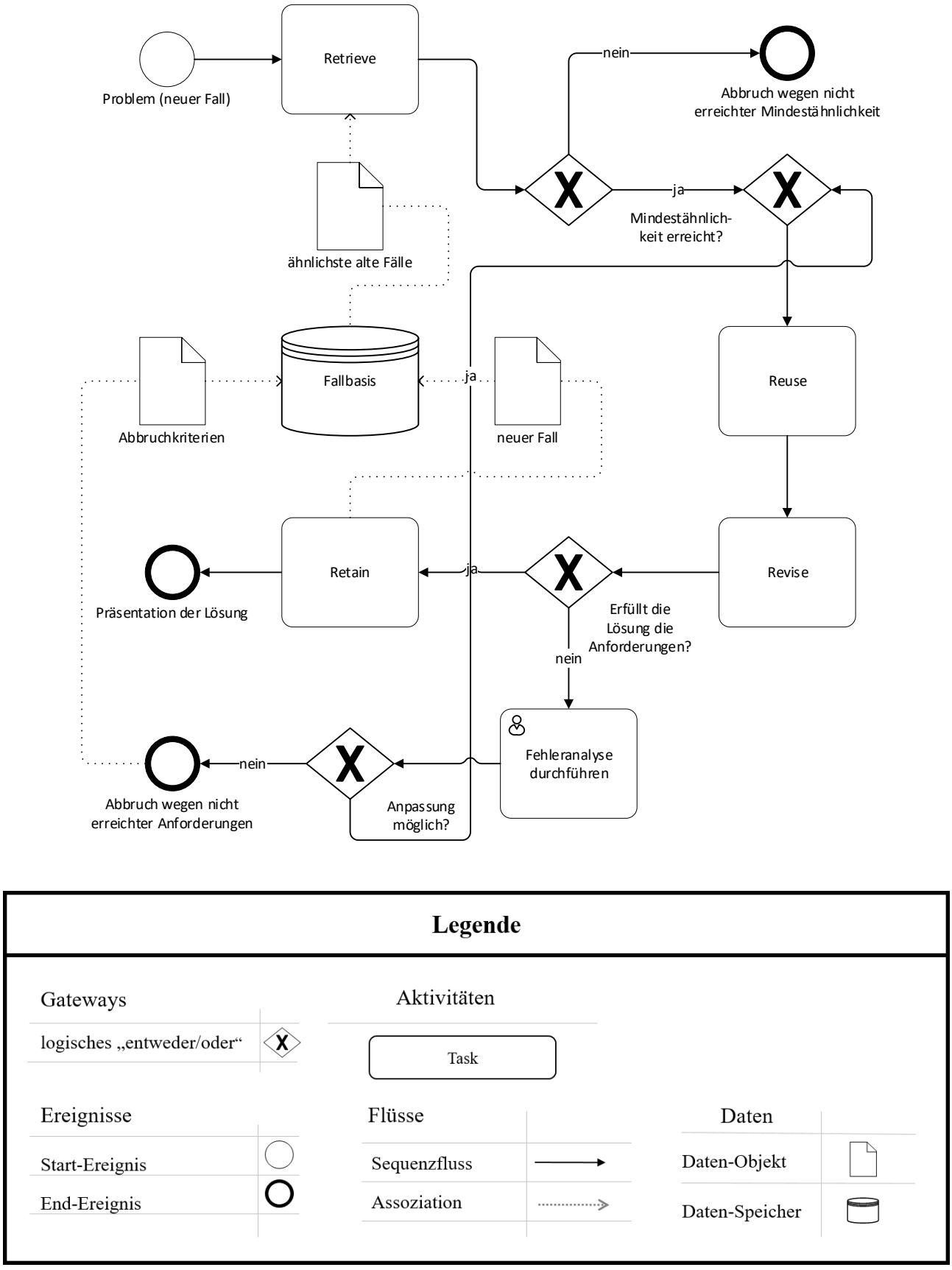

Abbildung 5: CBR-Zyklus[101]

101) In Anlehung an AAMODT/PLAZA (1994), S. 44. Für die Erweiterungen des ursprünglichen CBR-Zyklus, insbesondere hinsichtlich verwendeter Abbruchskriterien, vgl. KOWALSKI/KLÜPFEL/ZELEWSKI et al. (2012), S. 83; KOWALSKI/ZELEWSKI/GÜNES et al. (2011), S. 50.

Retrieve-Phase

Der CBR-Zyklus beginnt mit der Retrieve-Phase, also dem Ermitteln des ähnlichsten Falls oder der ähnlichsten Fälle, der bzw. die sich in der Fallbasis befindet bzw. befinden.[102] Zuvor muss der Nutzer seinen neuen, zu bearbeitenden Fall in das CBR-System eingeben.[103] Durch das Gewichten der relativen Bedeutungen von Klassen, Attributen und Relationen für die Ähnlichkeitsermittlung kann seitens des Nutzers auch eine subjektive Komponente in seine Anfrage an das CBR-System eingebracht werden.[104]

Eine bewährte Methode zur analytischen Bestimmung der Gewichtung qualitativer Faktoren ist beispielsweise der Analytic Hierarchy Process (AHP).[105] Diese Methode basiert auf Paarvergleichen qualitativer Aspekte auf verschiedenen Ebenen einer Hierarchiestufe mithilfe einer festgelegten Skala und bietet die Möglichkeit, nachvollziehbare und konsistente Entscheidungen zu treffen. Zusätzlich ist der AHP auch als ein Instrument zur Entscheidungsfindung in Gruppen geeignet, falls mehr als ein Entscheidungsträger an der Bestimmung der Gewichtung beteiligt ist. Denkbar wäre ebenso, ein Entscheidungsfindungstool auf Basis des AHP in ein CBR-System zu integrieren. Potenzielle Nutzer müssten lediglich die zu gewichtenden Aspekte (in der Regel Klassen, Attribute und Relationen) festlegen und würden im Anschluss aufgefordert, Paarvergleiche auf der zugrunde liegenden Skala zu tätigen. Nach automatischer Prüfung der Konsistenz wird eine Gewichtung errechnet, die in die Anfrage an das CBR-System einfließen kann.

Der zu bearbeitende Fall wird mit den in der Fallbasis befindlichen Fällen anhand von quantitativen und qualitativen Merkmalen mithilfe eines Ähnlichkeitsalgorithmus verglichen. Ein solcher Algorithmus zur Ermittlung der Ähnlichkeit zwischen Fallbeschreibungen greift in ontologiegestützten CBR-Systemen stets auf eine Domänen-Ontologie als wesentliche Komponente der Ähnlichkeitsermittlung zurück. Als Ergebnis erhält der Benutzer je nach verwendeten Einstellungen den ähnlichsten Fall (oder die ähnlichsten Fälle).[106] Die Ähnlichkeit wird mithilfe einer Zahl im Intervall [0,1] angegeben. Ein Wert von Null entspricht einer vollkommenen Verschiedenheit der Fallbeschreibungen und ein Wert von Eins einer vollkommenen Gleichheit sämtlicher relevanter, fallbeschreibender Merkmale.[107]

Reuse-Phase

Nachdem der Nutzer zu seiner Beschreibungung für einen neuen Fall den ähnlichsten Fall (oder die ähnlichsten Fälle) aus der Fallbasis erhalten hat, folgen in der Reuse-Phase eine Betrachtung der Unterschiede zwischen dem alten und dem neuen Fall sowie die Feststellung derjenigen Aspekte, die sich von der Falllösung für den ähnlichsten alten Fall auf den neuen Fall – das aktuelle Problem – unverändert übertragen lassen.[108] Stimmt die Fallbeschreibung für den gefundenen ähnlichsten alten Fall „weitgehend“ mit der Fallbeschreibung für den neuen Fall überein, kann ein einfaches Kopieren der Falllösung für den ähnlichsten alten Fall geschehen, ohne sie an den neuen Fall anzupassen (dieser Vorgang wird auch als Null-Adaption[109] bezeichnet).[110] Falls dies nicht möglich ist, weil sich alter und neuer Fall hinsichtlich ihrer Fallbeschreibungen zu stark voneinander unterscheiden, muss eine

102) Vgl. AAMODT/PLAZA (1994), S. 44. Der Einfachheit halber wird im Folgenden nur auf einen ähnlichsten Fall eingegangen.

103) Vgl. BERGENRODT/KOWALSKI (2015), S. 21.

104) Vgl. BEIERLE/KERN-ISBERNER (2019), S. 189.

105) Vgl. SAATY (1994), S. 22 ff.

106) Vgl. PFUHL (2003), S. 14.

107) Vgl. BERGENRODT/KOWALSKI (2015), S. 21.

108) Vgl. AAMODT/PLAZA (1994), S. 51.

109) Vgl. PFUHL (2003), S. 14; WILKE/BERGMANN (1998), S. 500; AAMODT/PLAZA (1994), S. 51.

110) Vgl. WATSON (1998), S. 16.

Anpassung („Adaption“) der Falllösung für den ähnlichsten alten Fall an die Fallbeschreibung des neuen Falls erfolgen.

WILKE/BERGMANN beschreiben verschiedene Adaptionstechniken.[111] Dazu gehören die transformierende, die substituierende, die strukturelle und die generative Adaption. Sie unterscheiden sich hinsichtlich verschiedener Aspekte, wie z. B. in Bezug auf den Grad der Nähe zwischen den Fallbeschreibungen für den ähnlichsten alten Fall und für den neuen Fall oder in Bezug auf das Ausmaß des benötigten Wissens über die spezifische Domäne der betrachteten Fälle.

Revise-Phase

Die vom CBR-System vorgeschlagene Falllösung muss hinsichtlich ihrer „Praxistauglichkeit“ überprüft werden, indem entweder Simulationen durchgeführt, Experten befragt werden oder die Problemlösung in der Praxis implementiert wird.[112] Das Ergebnis dieser Praxistauglichkeitsprüfung wird in die Fallbewertung aufgenommen und im Falle eines Fehlschlags um eine Analyse der Fehler erweitert. Die so entdeckten Fehler und zugrunde liegenden Erfahrungen führen zu einer Anpassung der Falllösung durch den Nutzer, um so eine hohe Güte der Eignung der Falllösung für den neuen Fall sicherzustellen.

Retain-Phase

In der letzten Phase des CBR-Zyklus werden die zuvor genannten Elemente eines Falls – die Fallbeschreibung, die Falllösung und die Fallbewertung – zusammengefasst und in einem neuen Fall in der Fallbasis abgelegt.

AAMODT/PLAZA sehen die Erweiterung des Index-Vokabulars[113] als einen wichtigen Teil eines CBR-Systems an, da so die Wahrscheinlichkeit zufriedenstellender Lösungsvorschläge über die Eingabe neuer Fälle über die Zeit hinweg steigt.[114]

2.4 Ontologiegestütztes Case-based Reasoning

Ein CBR-System kann sein Wissen in den vier Wissenscontainern Vokabular, Ähnlichkeitsermittlung, Adaption und Fallbasis speichern.[115] Innerhalb jedes Wissenscontainers lässt sich eine Ontologie zur Repräsentation von Wissen verwenden und liefert das benötigte Hintergrundwissen (streng genommen die sprachlichen Ausdrucksmittel für dieses Hintergrundwissen) für den jeweils betroffenen Systemteil.[116] Insbesondere geben Ontologien das für ein CBR-System notwendige Vokabular[117] vor und können auch die Fallstruktur des CBR-Systems definieren.[118] Sie bieten je nach implementiertem Algorithmus das Potenzial, eine Messung der Ähnlichkeit zwischen den Fällen in der Fallbasis zu ermöglichen und auch eine mögliche Adaption der Falllösungen zu verbessern.[119]

111) Vgl. WILKE/BERGMANN (1998), S. 500 ff.

112) Vgl. AAMODT/PLAZA (1994), S. 52; PFUHL (2003), S. 14. Der nächste Satz bezieht sich ebenfalls auf diese Quellenangaben.

113) Durch das Index-Vokabular werden alle für einen Vergleich relevanten Informationen der gespeicherten Fälle des CBR-Systems abgebildet, vgl. BEIERLE/KERN-ISBERNER (2019), S. 179.

114) Vgl. AAMODT/PLAZA (1994), S. 53; BEIERLE/KERN-ISBERNER (2019), S. 172.

115) Vgl. ROTH-BERGHOFER (2004), S. 394 f.

116) Vgl. RICHTER/WEBER (2013), S. 281.

117) Vgl. AMAILEF/LU (2013), S. 79. Das vorgegebene Vokabular einer Ontologie dient je nach CBR-System als „weitreichendes Grundgerüst“, jedoch können z. B. Instanzen als Konkretisierungen von Klassen hinzugefügt werden.

118) Vgl. ASSALI/LENNE/DEBRAY (2010), S. 102 f. Je nach Aufbau und Funktionsweise eines CBR-Systems müssen bestimmte Klassen innerhalb einer Ontologie existieren, um eine Ähnlichkeitsberechnung durchführen zu können.

119) Vgl. ASSALI/LENNE/DEBRAY (2010), S. 107 ff.

Innerhalb der Fallbasis eines CBR-Systems kann zwischen einer homogenen und einer heterogenen Fallstruktur unterschieden werden. Eine homogene Fallstruktur liegt vor, wenn alle Fälle dieselbe Datenstruktur aufweisen, sie also dieselben Klassen, Relationen und Attribute besitzen.[120] Eine solche Fallstruktur kommt jedoch nur für einen sehr kleinen Ausschnitt der Realität in Betracht. In der betrieblichen Praxis lassen sich strukturelle Unterschiede zwischen Fällen, bedingt durch den größeren Ausschnitt der Realität, oftmals nicht vermeiden.[121] Dementsprechend liegt eine heterogene Fallstruktur vor, wenn sich die (Daten-)Strukturen der Fälle zumindest teilweise unterscheiden.[122] Heterogene Fallstrukturen lassen sich mithilfe von Ontologien realisieren, die das verwendbare Vokabular mit entsprechender Flexibilität zur Verfügung stellen.[123]

120) Vgl. WATSON (2003), S. 27 f.

121) Als Beispiel sei ein Autoradio genannt: Dieses technische Gerät kann als kleinste Einheit einer homogenen Fallstruktur betrachtet werden. Jedoch besteht es aus schätzungsweise mehr als einhundert Einzelteilen. Soll in Fällen näher definiert werden, warum ein Autoradio ausgefallen ist, muss es auf seine Einzelteile heruntergebrochen werden. Eine homogene Fallstruktur lässt sich dann nur gewährleisten, wenn jedes denkmögliche Einzelteil zuvor in einer entsprechenden Ontologie spezifiziert wurde. Dies ist oftmals praktisch unmöglich, weil sich der technische Aufbau von Autoradios häufiger ändert, wie z. B. im Hinblick auf eingesetzte „integrierte“ Schaltkreise, die früher verwendete einzelne elektronische Komponenten ersetzen, oder in Bezug auf „Features“, über die zwar einzelne, aber nicht alle Autoradios verfügen (wie z. B. Ortungssysteme zwecks Diebstahlsprävention).

122) Vgl. WATSON (2003), S. 27 f.

123) Vgl. ASSALI/LENNE/DEBRAY (2010), S. 100 ff.

3 Konzipierung des Vorgehensmodells für die Entwicklung ontologiegestützter CBR-Systeme

3.1 Anforderungsspezifizierung für ein ontologiegestütztes CBR-System

Für die erfolgreiche Durchführung von Projekten stellt die Anforderungsanalyse einen wichtigen Baustein dar.[124]

Um den Begriff des Anforderungsmanagements[125] definieren zu können, muss zunächst eine Definition des Begriffs Anforderung erfolgen. Nach dem Standard IEEE 610.12-1990 wird der Begriff wie folgt definiert:[126] „Eine Anforderung ist:

(1) Eine Bedingung oder Eigenschaft, die ein System oder eine Person benötigt, um ein Problem zu lösen oder ein Ziel zu erreichen

(2) Eine Bedingung oder Eigenschaft, die ein System oder eine Systemkomponente aufweisen muss, um einen Vertrag zu erfüllen oder einem Standard, einer Spezifikation oder einem anderen formell auferlegten Dokument zu genügen

(3) Eine dokumentierte Repräsentation einer Bedingung oder Eigenschaft wie in (1) oder (2) definiert."

Nach dieser Definition kann eine Anforderung sowohl in dokumentierter Form[127] als auch in undokumentierter Form vorliegen.

Anforderungen lassen sich in drei Arten[128] klassifizieren, die nachfolgend kurz erläutert werden. Funktionale Anforderungen definieren die Funktionen (Aufgaben), die von einem System bereitzustellen sind (erfüllt werden sollen). Nicht-funktionale Anforderungen[129] definieren allgemeine Qualitätsmerkmale eines Systems wie z. B. die Zuverlässigkeit, die Ausfallsicherheit und vor allem auch die Benutzerfreundlichkeit. Rahmenbedingungen sind organisatorische oder technologische Anforderungen, die schwer bis gar nicht veränderbar sind. Sie werden auch als Restriktionen bezeichnet. Rahmenbedingungen können dahingehend differenziert werden, ob sie für den Entwicklungsprozess oder für das fertige System gelten.

Im Rahmen des BMBF-Forschungsprojekts „KI-LiveS" (KI-Labor für verteilte und eingebettete Systeme) greift der vorliegende Projektbericht hinsichtlich der Anforderungen direkt auf den Projektbericht Nr. 1 des KI-LiveS-Projekts zurück, in dem eine systematische Anforderungsanalyse im Hinblick auf KI-Tools für die „intelligente" Wiederverwendung von Erfahrungswissen im Bereich des betrieblichen Projektmanagements dokumentiert ist. Das Vorgehen bei der Anforderungsanalyse unterteilt sich in dem Projektbericht Nr. 1 in fünf Phasen und wird hier im vorliegenden

---

124) Hinsichtlich der Bedeutung des Anforderungsmanagements für den Projekterfolg scheint es mittlerweile einen Konsens sowohl in der Forschung als auch in der Praxis zu geben, vgl. EBERT (2017), S. 4 ff.; VON KOCEMBA/BELZ (2015), S. 22 f.; RUPP (2014), S. 10; NIEBISCH (2013), S. 9 f.; POHL (2007), S. 8 ff. Nicht zuletzt durch den „Chaos Report" der Standish Group von 1994 ist das Themengebiet des Anforderungsmanagements, das vormals ein überwiegend theoriegeprägtes Forschungsgebiet darstellte, für die Praxis durch „einfach" zu verstehende Zahlenwerte greifbarer geworden. Nicht zuletzt aus diesem Grund gibt es an dem „Chaos Report" eine anhaltende Kritik von Seiten der Wissenschaft, u. a. weil die Ergebnisse nicht reliabel seien und die Vorhersagequalität deshalb nicht belastbar sei, vgl. EVELEENS/VERHOEF (2010), S. 30 ff.; GLASS (2006), S. 15 f.

125) Im deutschen Sprachgebrauch ist die englische Bezeichnung Requirements Engineering ebenso geläufig.

126) POHL (2007), S. 13.

127) Dokumentierte Anforderungen werden auch als Anforderungsartefakte bezeichnet, vgl. POHL (2007), S. 13.

128) Vgl. POHL (2007), S. 14 ff. Der gesamte folgende Absatz bezieht sich auf diese Quellenangabe.

129) Abweichend von dem Begriff „Qualitätsanforderungen" in der genannten Fachliteratur wird in diesem Projektbericht der Begriff „nicht-funktionale Anforderungen" verwendet, um die Einheitlichkeit zum vorgenannten KI-LiveS-Projektbericht Nr. 1 zu wahren.

Projektbericht um eine sechste Phase (Evaluation der Ergebnisse mit den Stakeholdern) und eine siebte Phase (Erstellung des finalen Anforderungskatalogs) erweitert.

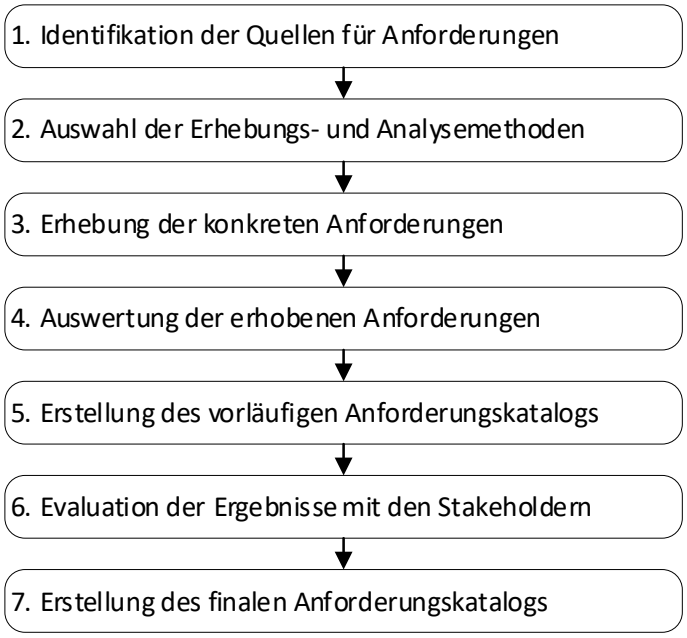

Abbildung 6: Vorgehen einer Anforderungsanalyse[130]

Identifikation der Quellen fürt Anforderungen

Um die Anforderungen an ein System ermitteln zu können, müssen zunächst die Quellen für Anforderungen an das zu entwickelnde System identifiziert werden.[131] In der Fachliteratur werden zumeist drei Quellen für Anforderungen genannt: Stakeholder[132], Dokumente und betriebliche Informationssysteme.[133] Unter den Begriff „Dokumente" fallen z. B. Normen und Standards sowie organisationsspezifische Dokumente, wie z. B. Prozessdokumentationen und Fehlerberichte der Vorgängersysteme. Der Begriff „betriebliche Informationssysteme" umfasst alle (Software-)Systeme, die sich gegenwärtig in Benutzung durch ein Unternehmen befinden. Aus ihnen können während der praktischen Arbeit Anforderungen an ein neues System abgeleitet und dokumentiert werden.

Im Rahmen des KI-LiveS-Projekts wurden Stakeholder als ausschließliche Anforderungsquelle genutzt, weil das Projekt vor allem ein praxisorientiertes Transferziel verfolgt.[134]

130) Eigene Darstellung in Anlehnung an SCHAGEN/ZELEWSKI/HEEB (2020), S. 3.

131) Vgl. RUPP (2014), S. 77 f.

132) Unter dem Begriff Stakeholder werden in dieser Arbeit sämtliche Akteure subsumiert, die einen (direkten oder indirekten) Einfluss auf die Anforderungen haben, vgl. RUPP (2014), S. 77. Akteure sind in diesem Zusammenhang nicht als Individuen zu sehen, sondern als Gruppen von individuellen Akteuren, die hinsichtlich ihrer Interessen gegenüber einem Unternehmen als annähernd gleichartig betrachtet werden, vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 3 f.

133) Vgl. POHL/RUPP (2015), S. 21.

134) Vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 4. Im Rahmen dieses Projektberichts werden im Vorgehensmodell alle drei Anforderungsquellen dargestellt. Der Fokus liegt jedoch auch hier auf der Anforderungsquelle der Stakeholder. Dies beruht darauf, dass sich die beiden anderen Quellen auf bereits in einem Unternehmen existierende KI-Systeme beziehen, die jedoch in der Praxis bisher nur in einer geringen Anzahl vorhanden sind. Da vornehmlich die einzelnen Schritte zur Konzipierung eines ontologiegestützten CBR-Systems dargestellt werden, wird nicht davon ausgegangen, dass bereits so ein System in einem Unternehmen existiert und dort genutzt wird.

Auswahl der Erhebungs- und Analysemethoden

Die Erhebungsmethoden sollen die bewussten, unbewussten und unterbewussten Anforderungen der Stakeholder an ein System dokumentieren.[135] Mithilfe von Analysemethoden sollen die identifizierten Anforderungen im Anschluss systematisiert werden, um so den Kriterien „Strukturiertheit", „Orthogonalität" und „Kompatibilität" zu genügen.[136] Das Ziel der Systematisierung von Anforderungen liegt in der Erstellung eines Anforderungskatalogs, der die Anforderungen unter Berücksichtigung der zuvor genannten Kriterien systemisch strukturiert.

Im Bereich des Anforderungsmanagements existiert eine Vielzahl von Erhebungs- und Analysemethoden, deren Eignung jeweils von den Rahmenbedingungen der Anforderungsanalyse abhängt.[137]

Erhebung der konkreten Anforderungen

Im Projektbericht Nr. 1 des KI-LiveS-Projekts wurden die drei Erhebungsmethoden Experteninterview, Use Cases und Storytelling eingesetzt.

Das Ziel von Experteninterviews ist es, domänenspezifische Informationen (hier Anforderungen) von Experten einer oder mehrerer Domänen zu erhalten.[138] Es kann zwischen explorativen und standardisierten Interviews unterschieden werden. Dem explorativen Interview liegen einige Leitfragen oder -gedanken zugrunde, auf deren Basis der Interviewer vertiefende, qualitative Fragen stellt, die sich auch aus den vorherigen Antworten des Interviewten ergeben können. Dem standardisierten Interview liegt hingegen ein Fragenkatalog zugrunde, der Abweichungen nur sehr eingeschränkt zulässt. Dier Aufbau gewährleistet eine gute Vergleichbarkeit der Ergebnisse, jedoch sind die möglichen Erkenntnisse zum großen Teil[139] auf die intendierten Antworten des Erstellers des Fragenkatalogs beschränkt.

Use Cases repräsentieren Anwendungsfälle für das System, für das die Anforderungen erhoben werden sollen.[140] Das Erkenntnisziel besteht speziell in der Ermittlung funktionaler Anforderungen, im hier betrachteten konkreten Fall die funktionalen Anforderungen der Wirtschaftspraxis an ein KI-Tool zur „intelligenten" Wiederverwendung von Erfahrungswissen im betrieblichen Projektmanagement.[141] Die Durchführung der Use Cases im KI-LiveS-Projekt findet auf Basis einer Use-Case-Schablone statt, die u. a. folgende Aspekte umfasst: Akteure und Ereignisse zum Auslösen des Use Case, eine Kurzbeschreibung des Use Case, die vorhandenen Vorbedingungen und Aktivitäten des Use Case sowie die Ausnahmefälle und Nachbedingungen des Use Case.

---

135) Vgl. POHL/RUPP (2015), S. 26.

136) Vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 4. Der nächste Satz bezieht sich ebenfalls auf diese Quelle. Das Kriterium der „Strukturiertheit" bezieht sich auf die inhaltlichen Über- oder Unterordnungsbeziehungen zwischen Anforderungen. Das Kriterium „Orthogonalität" erstreckt sich auf die inhaltliche Einzigartigkeit ohne Überschneidungen von Anforderungen. Das Kriterium „Kompatibilität" adressiert die notwendige Konsistenz der Anforderungen untereinander.

137) Vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 4 f. An dieser Stelle wird auf eine Erläuterung der spezifischen Erhebungs- und Analysemethoden verzichtet, da die Ergebnisse der Anforderungserhebung in diesem Projektbericht im Vordergrund stehen. Einen Überblick sowie eine Evaluation der Erhebungs- und Analysemethoden im Rahmen der Anforderungsanalyse des KI-LiveS-Projekts finden sich im zitierten Projektbericht Nr. 1 (SCHAGEN/ZELEWSKI/HEEB (2020)) auf S. 4-14; weiterführende Literaturangaben werden auf S. 4 dieses Projektberichts angeführt.

138) Vgl. POHL (2007), S. 325.

139) Der Übergang zwischen einem explorativen und einem standardisierten Interview ist fließend. Es bietet sich an, gewisse Teilbereiche eines Interviews, wie beispielsweise die persönlichen Daten und den groben Aufbau, zu standardisieren, um einerseits eine Basis-Vergleichbarkeit und andererseits möglichst umfangreiche und zielführende Antworten zu erlangen.

140) Vgl. RUPP (2014), S. 116.

141) Vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 21 f. Der nachfolgende Satz bezieht sich ebenfalls auf diese Quellenangabe.

Durch die Methode des Storytellings soll Erfahrungswissen von Mitarbeitern über einschneidende Ereignisse[142] in einem Unternehmen erfasst und ausgewertet werden, um auf dieser Basis die gesammelten Erfahrungen zu dokumentieren und im Unternehmen nutzbar zu machen.[143] Dadurch sollen funktionale und nicht-funktionale Anforderungen, insbesondere diejenigen, die in implizierter Form „in den Köpfen von Experten" verankert sind, gewonnen werden.[144]

Auswertung der konkreten Anforderungen

Die Auswertung der konkreten Anforderungen wird im Projektbericht Nr. 1 des KI-LiveS-Projekts detailliert geschildert.[145] Sie erfolgt in drei Rubriken im Hinblick auf die Experteninterviews, die Use Cases und das Storytelling. Hinsichtlich der Experteninterviews wird zwischen den Auswertungen in Bezug auf die persönlichen Daten der befragten Experten, die funktionalen Anforderungen, die nicht-funktionalen Anforderungen sowie – als Besonderheit dieser Anforderungsanalye – auch in Bezug auf die Erwartungen an Veränderungen der Arbeitsplätze im betrieblichen Projektmanagement beim Einsatz von KI-Tools unterschieden.

Erstellung des vorläufigen Anforderungskatalogs

Nach der Erhebung und Auswertung der Anforderungen an die „intelligente" Wiederverwendung von Erfahrungswissen im betrieblichen Projektmanagement ist die Grundlage für die Konstruktion eines Anforderungskatalogs geschaffen. Die Systematik orientiert sich im Falle des Projektberichts Nr. 1 des Verbundprojekts KI-LiveS an dem Interviewleitfaden.[146] Demzufolge werden zuerst die funktionalen und nachfolgend die nicht-funktionalen Anforderungen aufgeführt, jeweils in Clustern von eng miteinander zusammenhängenden Anforderungen gruppiert. Zum Schluss werden die Anforderungen aufgrund von Randbedingungen aufgeführt.

Evaluation der Ergebnisse mit den Stakeholdern

Nachdem der vorläufige Anforderungskatalog erstellt wurde, besteht die Möglichkeit, diesen vorläufigen Anforderungskatalog mit den Stakeholdern zu evaluieren. Die Stakeholder können hierbei überprüfen, ob die von ihnen explizit formulierten und die von den Studiendurchführenden implizit interpretierten Anforderungen korrekt in den vorläufigen Anforderungskatalog übernommen wurden – oder ob gegebenenfalls Korrekturen vorgenommen werden sollten. Insbesondere die Transformation von impliziten zu expliziten Anforderungen fordert immer auch eine subjektive Interpretation der Studiendurchführenden. Diese Interpretation wird durch den Wissensstand über die betrachtete Domäne und das Wissen der Studiendurchführenden über die Anforderungsquellen beeinflusst. Ebenso gilt es, im Verlauf der Evaluation konfliktäre Anforderungen zu diskutieren und solche Konflikte nach Möglichkeit zu beheben. Es ist zu berücksichtigen, dass eine Evaluation des Anforderungskatalogs auch neue, „sekundäre" Anforderungen der Stakeholder hervorbringen kann.

Da ein nachträgliches Aufarbeiten von Anforderungen in späteren Phasen eines Entwicklungsprojekts um ein vielfaches höher ausfällt als eine frühe Korrektur der Anforderungen im Verlauf der Anforderungsanalyse, unterstreicht dies die besondere Bedeutung der Evaluation des vorläufigen Anforderungskatalogs.[147]

142) Einschneidende Ereignisse stellen in diesem Zusammenhang Ereignisse dar, deren Bedeutung über die tägliche Geschäftstätigkeit hinaus geht und die zumindest in Teilaspekten einmalig sind. Dabei kann es sich beispielsweise um Pilotprojekte, eine Fusion oder Spaltung, die Einführung neuer Produkte oder neuer Informationssysteme, wie z. B. eines ontologiegestützten CBR-Systems, handeln.

143) Vgl. THIER (2017), S. 21.

144) Vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 25.

145) Vgl. SCHAGEN/ZELEWSKI/HEEB (2020), Kapitel 3 (S. 28-85).

146) Vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 86. Der nachfolgende Satz bezieht sich ebenfalls auf diese Quelle.

147) Vgl. speziell zur Qualitätssicherung der Erhebung von Anforderungen EBERT (2017), S. 181 ff.

Erstellung des finalen Anforderungskatalogs

Wurde mindestens eine Evaluation durchgeführt und sind keine grundsätzlichen Qualitätsmängel aufgedeckt worden, kann der vorläufige Anforderungskatalog – unter Beachtung eventuell geringfügiger, z. B. redaktioneller Überarbeitungen – in den finalen Anforderungskatalog überführt werden. Andernfalls, wenn grundsätzliche Qualitätsmängel hinsichtlich des vorläufigen Anforderungskatalogs aufgedeckt wurden, bedarf es einer Rückkopplung zu früheren Phasen der Anforderungsanalyse, die so lange zu wiederholen sind, bis ein überarbeiteter vorläufiger Anforderungskatalog ohne grundsätzliche Qualitätsmängel resultiert.[148]

3.2 Anforderungen an ein KI-Tool aus Praxissicht

Auf Basis des Anforderungskatalogs an KI-Tools zur intelligenten, computergestützten Wiederverwendung von Erfahrungswissen im betrieblichen Projektmanagement aus dem KI-LiveS-Projektbericht Nr. 1 erfolgt nun ein kurzer Überblick über ausgewählte funktionale und nicht-funktionale Anforderungen sowie über Anforderungen aufgrund von Randbedingungen.

Funktionale Anforderungen[149]

Anforderungen hinsichtlich der Berücksichtigung von Erfahrungswissen

- Domänenunterstützungsfunktion[150]
- Phasen- und Aufgabenunterstützungsfunktion[151]
- Projektbeschreibungsfunktion[152]
- Projektbewertungsfunktion[153]

148) Von der „theoretischen“ Denkmöglichkeit einer „Endlosschleife“ sei hier der Einfachheit und Praxisnähe halber abgesehen.

149) Für den vollständigen Anforderungskatalog vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 88 ff.

150) Die Effektivität der Erfüllung von Projektmanagementaufgaben im Kontext eines ontologiegestützten CBR-Systems hängt wesentlich von der Güte der Abdeckung des Anwendungsbereichs durch die Ontologie ab, vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 88.

151) Folgende Phasen wurden nach der Häufigkeit der Nennung in der Kategorie „sehr wichtig“ innerhalb der Experteninterviews genannt (die Auflistung ist gekürzt): Ausarbeitung von Angeboten (Vorkalkulation), Suche nach Projektpartnern, Suche nach relevanten Daten bereits durchgeführter Projekte für die Planung neuer Projekte, Zusammenstellung von Projektteams für neue Projekte sowie Projektplanung; vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 89.

152) Diese Funktion umfasst qualitative, natürlichsprachliche Projektmerkmale in unstrukturierten, projektbezogenen Dokumenten; vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 90 ff. Die nachfolgend ausgewählten Aspekte beziehen sich ebenso auf die Projektbearbeitung und -finalisierung. Die Unterscheidung zwischen Soll- und Ist-Größen erfolgt mithilfe des CBR-Tools jCORA im späteren Verlauf des Projektberichts. Ausgewählte Aspekte lauten: Projektgegenstand, Projektart, Projektbranche sowie räumlicher und inhaltlicher Projektumfang („Scope“).

153) Unter der Projektbewertungsfunktion sind alle Unteranforderungen subsumiert, die bereits bei der zuvor dargestellten Projektbeschreibungsfunktion erfasst wurden. Der Fokus liegt bei dieser Funktion jedoch auf Ist-Größen und deren Abweichungen von Soll-Größen; vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 95 ff. Zusätzliche Anforderungen sind Formen der Projektbewertung wie die Unterscheidung zwischen unstrukturierten, natürlichsprachlichen Dokumenten wie Lessons learned, Debriefings und Projekt Reports sowie die Inhalte der Projektbewertung wie kritische Erfolgs- und Misserfolgsfaktoren.

- Datenquellenerschließungsfunktion[154]
- Wissensspeicher-, Wissensbereitstellungs- und Wissenswiederverwendungsfunktion[155]

Anforderungen hinsichtlich der Wirtschaftlichkeit des Projektmanagements

- Effektivitätsfunktion[156]
- Effizienzfunktion[157]

Anforderungen hinsichtlich der Benutzung eines CBR-Systems, in dem Projekte als Fälle repräsentiert und bearbeitet werden

- Falleingabefunktion[158]
- Fallauswahlfunktion[159]
- Ähnlichkeitsberechnungsfunktion[160]
- Fallbewertungsfunktion[161]
- Gewichtungsfunktion[162]

Ein Teil der ermittelten funktionalen Anforderungen lässt sich auch als Grundlage für die Formulierung der im folgenden Kapitel 3.3 dargestellten Competency Questions heranziehen.

---

154) Diese funktionale Anforderung wird im Projektbericht Nr. 1 als „Datenerschließungsfunktion" bezeichnet. Die hier bevorzugte Bezeichnung „Datenquellenerschließungsfunktion" lässt direkt auf den Inhalt schließen, weshalb sie hier verwendet wird. Vor allem wurden die folgenden Datenquellen ermittelt: MS-Office-Dateien [PPT(X), DOC(X), XLS(X)], Erfahrungswissen in den „Köpfen" von Projektmanagern und anderen projektbezogenen Ansprechpartnern sowie PDF-Dateien und Projektmanagementdatenbanken; vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 103 ff.

155) Das KI-Tool soll erfasstes Erfahrungswissen aus früheren Projekten sowie neu generiertes Erfahrungswissen für eine spätere Wiederverwendung speichern und bereitstellen; vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 105 ff.

156) Das KI-Tool soll die Effektivität des Projektmanagements als Verhältnis zwischen dem Ist- und dem Soll-Output der Projektmanagementaktivitäten steigern, vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 107 ff. Diese Anforderung umfasst eine Ergebnisfunktion, wie z. B. Handlungsempfehlungen für neue Projekte, das Aufzeigen von möglichen Verbesserungspotenzialen und eine Schätzung der Projektkosten.

157) Die Produktivität des Projektmanagements als Verhältnis zwischen dem Output und dem Input (Ressourceneinsatz) der Projektmanagementaktivitäten soll gesteigert werden; vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 109 ff. Eine Produktivitätssteigerung kann z. B. durch einen geringen zusätzlichen Ressourceneinsatz bei der Implementierung des KI-Tools sowie durch eine Minderung des Ressourceneinsatzes bei gleichbleibender Leistungsfähigkeit während der Nutzung des KI-Tools realisiert werden.

158) Alte, in der Vergangenheit bereits durchgeführte Projekte sollen mithilfe einer Anleitung innerhalb des KI-Tools als alte Fälle eingegeben, gespeichert und zur Verfügung gestellt werden; vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 110.

159) Das KI-Tool soll in der Lage sein, nach Benutzervorgabe eine – jeweils näher zu spezifizierende – Anzahl möglichst ähnlicher Altfälle als Referenzfälle für einen neuen Fall zu präsentieren, vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 110 f.

160) Das KI-Tool soll die Ähnlichkeit zwischen Fällen berechnen und hierfür einen numerischen Wert ausgeben können. Im Vorfeld der automatisierten Ähnlichkeitsberechnung sollen Benutzer die Gewichtung der ähnlichkeitsrelevanten Fallmerkmale, die in die Ähnlichkeitsberechnung einbezogen werden, subjektiv beeinflussen können; vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 111 f.

161) Das KI-Tool soll den Benutzern die Möglichkeit bieten, ein empfohlenes Fallresultat kritisch zu hinterfragen und bei Bedarf zu verändern. Dies soll verhindern, dass mögliche neue und „innovative" Problemlösungsmöglichkeiten nicht in die Wissensbasis des KI-Tools aufgenommen werden; vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 112.

162) Der Benutzer soll projektspezifische Einflussgrößen je nach Perspektive der Priorität im aktuell zu bearbeitenden Fall gewichten können. Die Einflussgrößen müssen durch die Projektmitarbeiter festgelegt werden, da sie als Benutzer des KI-Tools die zu vergleichenden Einflussgrößen für eine Ähnlichkeitsberechnung bestimmen; vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 112.

Über die zuvor zusammengefassten Anforderungen hinaus existieren noch weitere funktionale Anforderungen[163] an ein KI-Tool, die unabhängig von der Wiederverwertung von Erfahrungswissen im betrieblichen Projektmanagement sind. Auszugsweise handelt es sich um die Plattformfunktion, die Kategorisierungsfunktion, die Budgetauskunftsfunktion, die Planungsfunktion und die Datensicherheitsfunktion.

Nicht-funktionale Anforderungen

Anforderungen hinsichtlich der Benutzerfreundlichkeit der Benutzerschnittstelle eines KI-Tools

- Benutzerkomfort[164]
- Eingabemasken[165]
- Dateiselektion[166]
- Ergebnisanzeige[167]

Weitere Anforderungen, die sich nicht auf die Benutzerfreundlichkeit der Benutzerschnittstelle eines KI-Tools beziehen

- Funktionalität des KI-Tools[168]
- Effizienz und Zuverlässigkeit des KI-Tools[169]
- Interoperabilität des KI-Tools in Bezug auf andere Software
- Kompatibilität des KI-Tools[170]
- Anpassbarkeit des KI-Tools
- Überprüfbarkeit hinsichtlich der Ergebnisse des KI-Tools

163) Der vollständige Anforderungskatalog befindet sich in SCHAGEN/ZELEWSKI/HEEB (2020), S. 114 ff.

164) Unter Benutzerkomfort wird eine gute Verständlichkeit des KI-Tools hinsichtlich seiner Benutzung verstanden. Darunter werden u. a. eine einfache und intuitive Benutzerführung, eine hohe Plausibilität der Empfehlungen des KI-Tools, das Vorhandensein von Schulungsmaterial für die Bedienung des Tools, eine Suchfunktion, eine Feedbackmöglichkeit für Benutzer und eine Anpassbarkeit der Benutzerschnittstelle des KI-Tools an die Bedürfnisse unterschiedlicher Benutzergruppen (z. B. Projektmitarbeiter versus Projektmanager) verstanden; vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 117 ff.

165) Für die Dateneingabe sollen Templates oder Eingabemasken durch das KI-Tool zur Verfügung gestellt werden. Die Eingabemasken sollen sich, je nach aktueller Projektphase, verändern. Weiterhin soll der Benutzer durch eine einfache Sichtkontrolle („Ampelsystem“) ein Feedback über die Qualität oder die Vollständigkeit seiner Dateneingabe erhalten; vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 119.

166) Das KI-Tool soll Dateien aus einem Pool mit projektrelevantem Wissen eigenständig identifizieren und erschließen, sodass keine Kapazitäten von Mitarbeitern durch diese „Fleißarbeit“ absorbiert werden; vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 119.

167) Das KI-Tool soll alte Fälle mit dem numerisch berechneten Ähnlichkeitswert nach einer vorgegebenen Mindestähnlichkeit oder nach vorgegebenen Schlagwörtern anzeigen; vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 119 f.

168) Die dem KI-Tool zugeordneten Aufgaben sollen korrekt erfüllt werden. Die Funktionalität bezieht sich in diesem Zusammenhang auf die übergeordnete Fähigkeit einer Software (oder allgemein: eines Systems), die ihm zugeteilten Aufgaben auf einer Metaebene korrekt zu erfüllen, und bezieht sich nicht auf den Inhalt einzelner Aufgaben; vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 120.

169) Der Effizienzbegriff in Bezug auf das Leistungsniveau des KI-Tools und die eingesetzten Ressourcen bezieht sich in diesem Fall beispielsweise auf die aufgewandte Arbeitszeit für die Benutzer und den Betriebsmitteleinsatz in Bezug auf Hard- und Software; vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 120 f.

170) Die geforderte Kompatibilität erstreckt sich auf die Unterstützung des KI-Tools im Hinblick auf branchenübliche Standards im Bereich des Wissensmanagements; vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 121.

- Das Vorhandensein von Schulungsmaterial im Umgang mit dem KI-Tool[171]

Anforderungen aufgrund von Randbedingungen

Kompetenzunterstützung der Mitarbeiter im Projektmanagement durch ein KI-Tool

- Unterstützung allgemeiner projektmanagementbezogener Kompetenzen[172]
- Unterstützung spezieller projektmanagementbezogener Kompetenzen[173]
- Unterstützung „weicher" Kompetenzen[174]

Datenbereitstellungsunterstützung[175]

- spezifische Softwareschnittstellen „gängiger" Projektmanagement-Software[176]
- allgemeine Softwareschnittstellen „gängiger" IT-Software

Arbeitsplatzunterstützung

- Unterstützung der Teilung von Fach- und Projektwissen[177]
- Unterstützung des Sicherheitsgefühls hinsichtlich Arbeitsplatz und Arbeitstätigkeit[178]

171) Dies bezieht sich sowohl auf integrierte Tutorials innerhalb des KI-Tools als auch auf externes, online bereitgestelltes Schulungsmaterial; vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 122.

172) Die Unterstützung umfasst die Identifizierung ähnlichster bereits durchgeführter Projekte (Fälle) in Bezug auf ein aktuelles Projekt, die Anpassung der Resultate (Fallresultate) von bereits durchgeführten Projekten an die Beschreibungen (Fallbeschreibungen) aktueller Projekte und die Identifizierung kritischer, projektbezogener Erfolgs- und Misserfolgsfaktoren; vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 122.

173) Unter speziellen projektmanagementbezogenen Kompetenzen werden zum einen die Anwendung agiler Projektmanagementmethoden und zum anderen das Festlegen situativ- und projektbedingter Meilensteine subsumiert; vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 123.

174) Die Anforderung meint die Förderung der Kommunikation zwischen den Projektmitarbeitern und eine Unterstützung des Teambuildings in Bezug auf zukünftige Projekte (siehe auch die funktionalen Anforderungen der Phasen- und Aufgabenunterstützungsfunktion); vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 123.

175) Ein KI-Tool soll Unternehmen im Allgemeinen und Projektmitarbeiter im Speziellen dazu motivieren, das „in ihren Köpfen" eingeschlossene Erfahrungswissen hinsichtlich des betrieblichen Projektmanagements in das KI-Tool einzubringen. In diesem Zusammenhang sollen hemmende Faktoren, wie z. B. die Angst vor einer „Wissensenteignung" oder vor einem „Kontrollverlust", abgebaut werden, vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 123.

176) Mögliche Softwares stellen das SAP-Modul PS (Projekt System), Microsoft Project und Altassian Jira dar; vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 123 f.

177) Die Anforderung bezieht sich sowohl auf eine unternehmensinterne als auch -externe Wissensteilung; vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 124. Eine Möglichkeit der Einschränkung der Wissensweitergabe aus wettbewerbsstrategischen Gründen ist ebenfalls denkbar und wirkt „hemmenden" Faktoren entgegen.

178) Mitarbeitern des Projektmanagements soll glaubhaft vermittelt werden, dass ihre Arbeitsplätze durch den Einsatz eines KI-Tools nicht obsolet werden, sondern sie lediglich eine Unterstützung im Treffen ihrer Entscheidungen sowie eine Befreiung von Routinetätigkeiten erfahren. Weiterhin sollen Vorkehrungen getroffen werden können, die eine stärkere Überwachung der Projektmanagement-Tätigkeiten der Projektmitarbeiter glaubhaft ausschließen, vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 125 f.

3.3 Erstellung der Domänenontologie

Im Bereich der Ontologie-Erstellung[179] haben sich bislang keine Standard-Vorgehensmodelle etablieren können.[180] Es besteht jedoch ein Konsens darüber, dass die Ontologie-Erstellung zumeist ein iterativer Prozess ist.

Vorhandene Entwicklungsmethoden lassen sich grob in zwei Gruppen[181] unterteilen: Auf der einen Seite existieren erfahrungsbasierte Methoden, wie z. B. die Entwicklungsmethoden von USCHOLD/KING (1995) oder GRÜNINGER/FOX (1995). Auf der anderen Seite existieren evolutionäre Methoden, wie z. B. „Methontology" von FERNÁNDEZ-LÓPEZ/GÓMEZ-PÉREZ/JURISTO (1997) und die „101-Methode" von NOY/MCGUINNESS (2001).[182] Die verwendete Entwicklungsmethode hängt von vielen Einflussgrößen ab, wie z. B. von den Absichten und Kompetenzen der Entwickler sowie von projektspezifischen Einflussgrößen, wie z. B. den Anforderungen an die Detailliertheit der Ontologie und der Implementierungssprache.[183]

Das Vorgehen zur Erstellung einer Domänenontologie im Bereich des Projektmanagements orientiert sich in diesem Projektbericht an der Entwicklungsmethode von NOY/MCGUINNESS.[184] Diese Entwicklungsmethode wurde gewählt, weil sie sich zum einen ausdrücklich an Erstentwickler von Ontologien richtet und deshalb verhältnismäßig einfach anzuwenden ist und weil sie sich zum anderen einer relativ weiten Verbreitung[185] erfreut.[186] Die 101-Methode zur Erstellung einer Domänenontologie wird mithilfe des Ontologie-Editors Protégé eingesetzt.[187]

Protégé ist eine im Jahr 2000 von der Stanford Medical Informatics Group der Stanford University entwickelte Open-Source-Software, die kostenlos installiert und verwendet werden kann sowie einen der am häufigsten genutzte Ontologie-Editoren darstellt.[188] In Protégé können Domänen-Ontologien in der Ontologierepräsentationssprache OWL konstruiert werden.[189] Mithilfe des Plug-ins OWLViz ist es möglich, eine vorhandene Ontologie zu visualisieren und sich so einen intuitiv leicht zugänglichen Überblick über die Ontologie zu verschaffen.

Je umfangreicher eine Ontologie ist, desto schwieriger ist die Vermeidung formaler und konzeptioneller Inkonsistenzen bei ihrer Erstellung.[190] Ontologie-Editoren, wie auch Protégé, unterstützen deshalb automatische Inferenzmechanismen, welche die formale und konzeptionelle Konsistenz einer Ontologie überprüfen und den Anwender bei der Detektion von Fehlern unterstützen. Protégé bietet eine Vielzahl von Reasonern an, die zur Konsistenzprüfung verwendet werden können.

---

179) Wie bereits an früherer Stelle angemerkt, werden in diesem Projektbericht die Begriffe „Erstellung" und „Entwicklung" synonym verwendet .

180) Vgl. STUCKENSCHMIDT (2011), S. 158 ff. Der nachfolgende Satz bezieht sich ebenfalls auf diese Quellenangabe.

181) Vgl. BRUSA/CALIUSCO/CHIOTTI (2006), S. 7 f.

182) Einen Überblick über verschiedene Methoden zur Ontologieerstellung bieten GÓMEZ-PÉREZ/FERNÁNDEZ-LÓPEZ/CORCHO (2004), S. 113 ff.

183) Vgl. KOWALSKI/ZELEWSKI (2015), S. 601; BRUSA/CALIUSCO/CHIOTTI (2006), S. 7 f.

184) Vgl. NOY/MCGUINNESS (2001), S. 4 ff.

185) Beispiele sind die „Command & Control Ontology" von CURTS/CAMPBELL (2005).

186) Vgl. KOWALSKI/ZELEWSKI (2015), S. 600 f.

187) Die Wahl des Ontologie-Editors Protégé hat mehrere Gründe. Erstens ist Protégé der bevorzugte Ontologie-Editor innerhalb des KI-LiveS-Projekt-Konsortiums. Zweitens hat BEIßEL mittels der AHP-Methode im Vergleich von fünf Ontologie-Editoren im Hinblick auf verschiedene Kriterien Protégé als den „besten" Ontologie-Editor bewertet, vgl. BEIßEL (2011), S. 88 ff.. Drittens wird Protégé im Leitfaden der 101-Methode benutzt; vgl. NOY/MCGUINNESS (2001), S. 2.

188) Vgl. NOY/CRUBÉZY/FERGERSON et al. (2003), S. 953.

189) Vgl. KOWALSKI/ZELEWSKI (2015), S. 601. Der nachfolgende Satz bezieht sich ebenfalls auf diese Quelle.

190) Vgl. STUCKENSCHMIDT (2009), S. 158 ff. Der nachfolgende Satz bezieht sich ebenfalls auf diese Quelle.

Nachfolgend wird eine Domänenontologie anhand der sieben Schritte gemäß der 101-Methode von NOY/MCGUINNESS vorgestellt.

Schritt 1: Bestimmung der Domäne und des Umfangs der Ontologie

Als Domäne wird das betriebliche Projektmanagement festgelegt, weil dieser Projektbericht innerhalb des Verbundprojekts KI-LiveS erstellt wird und der Fokus auf der Wiederverwertung von Erfahrungswissen im betrieblichen Projektmanagement liegt.

Um den Umfang einer Ontologie festzulegen, wird das Mittel der Competency Questions[191] gewählt. Diese Fragen sollen sich mithilfe der zu erstellenden Ontologie beantworten lassen und werden zunächst von Stakeholdern, z. B. mittels der Technik des Brainstormings[192], gesammelt.[193] Idealerweise sollen die Fragen sich auf alle Ebenen der Ontologie beziehen, sodass die Fragen schon in diesem Stadium hierarchisch in Beziehungen zueinander stehen.[194]

Competency Questions im Bereich einer Projektmanagementontologie könnten z. B. lauten:[195]

- Welche Projektteams haben welche Kompetenzen?
- Auf welchen Märkten ist das Unternehmen unterrepräsentiert?
- Wie groß ist die durchschnittliche Budgetabweichung?
- Welche Strukturierungsaspekte besitzt ein Projekt?
- Welche Risiken existieren in bestimmten Ländern?

Die Festlegung von Competency Questions kann als eine Art der Anforderungsspezifizierung betrachtet werden, weil verschiedene Stakeholder ihre Erwartungen an das zu erstellende Endprodukt formulieren.

Schritt 2: Wiederverwendung existierender Ontologien

Eine Wiederverwendung von bereits bestehenden Ontologien oder ihrer Teile kann den Aufwand für die Erstellung einer neuen Ontologie erheblich verringern.[196] Im Zuge einer Wiederverwendung kann von den Erfahrungen bei der Modellierung bestimmter Domains profitiert werden, sodass sich Fehler hinsichtlich der Ontologie-Erstellung vermeiden lassen. Auch kann sich durch Basisontologien bestimmter Domänen eine einheitliche Terminologie entwickeln, die durch ihren Normcharakter den Austausch von Informationen zwischen verschiedenen Anwendern unterstützt. Nichtsdestoweniger können bei der Wiederverwendung bestehender Ontologien erhebliche Probleme entstehen: Zum einen erfüllen bestehende Ontologien in den seltensten Fällen sämtliche Anforderungen in Hinblick auf die zuvor formulierten Competency Questions. Dies führt zu einer Abwägung, ob der Aufwand zur Anpassung an die Anforderungen geringer ist als die komplette Modellierung einer neuen Ontologie. Dies ist besonders hervorzuheben, weil die Analyse einer unbekannten Ontologie mitunter sehr aufwendig sein kann.

191) Das Konzept der Competency Questions findet sich auch in anderen Vorgehensmodellen zur Konstruktion von Ontologien, beispielsweise im Projekt TOVE (Toronto Virtual Enterprise) von GRÜNINGER und FOX; vgl. DITTMANN/APKE (2005), S. 300 ff.

192) Vgl. USCHOLD/GRUNINGER (1996), S. 19.

193) Vgl. NOY/MCGUINNESS (2001), S. 5; STUCKENSCHMIDT (2011), S. 159 f.

194) Vgl. USCHOLD/GRUNINGER (1996), S. 29.

195) Die Aufzählung der beispielhaften Competency Questions ist nicht erschöpfend und kann im zeitlichen Verlauf der Entwicklung einer Domänenontologie variieren. So können bestimmte Fragen im Verlauf als nicht mehr relevant angesehen werden und neue Aspekte an Wichtigkeit gewinnen; vgl. STUCKENSCHMIDT (2011), S. 160.

196) Vgl. STUCKENSCHMIDT (2011), S. 160 f. Der gesamte Absatz bezieht sich auf diese Quelle.

Bereits bestehende Ontologien existieren in der Fachliteratur, in Online-Bibliotheken[197] und können mit dedizierten Suchmaschinen[198] gefunden werden.

Bei der Wiederverwendung bestehender Ontologien ist nicht nur die formale Repräsentationssprache von Bedeutung, sondern auch die zugrunde gelegte Umgangssprache (natürliche Sprache).[199] Die überwiegende Mehrheit der veröffentlichten Ontologien ist in englischer Sprache verfasst. Die durch Klassen repräsentierten Konzepte sind zum Teil nicht ohne weitere Interpretation zu übersetzen, weil sie sonst ihre Bedeutung verlieren oder ändern könnten. Gleiches gilt für Instanzen, Relationen und Attribute. Es bietet sich daher an, bei der Erstellung einer Ontologie zumindest eine Zweisprachigkeit der verwendeten Begrifflichkeiten anzustreben, um einerseits die Verbreitung von Wissen zu fördern und andererseits dem internationalen Umfeld der Domäne des Projektmanagements zu genügen.

Schritt 3: Identifikation relevanter Begriffe

Die Identifikation der relevanten Begriffe kann für eine Domänenontologie auf mehrere Arten erfolgen. Ein weit verbreiteter Ansatz ist die Befragung von Experten aus der jeweils zugunde liegenden Anwendungsdomäne.[200] Dieses Vorgehen ergibt eine große Überschneidung mit den zuvor formulierten Competency Questions aus Schritt 1, sodass diese Competency Questions zumindest als Grundlage für relevante Begriffe benutzt werden können. Weitere Quellen sind Dokumente über oder aus dem Anwendungsbereich der Ontologie und andere Datenquellen, wie z. B. Projektdatenbanken und -berichte. Hierbei lassen sich Wortarten direkt Ontologiebestandteilen zuordnen: Substantive repräsentieren zumeist Klassen, Verben können auf Relationen und Adjektive auf Attribute hindeuten.

Im Bereich einer Projektmanagement-Ontologie können z. B. die folgenden Begriffe relevant sein[201]:

- Projektbeschreibung
- Projektlösung
- Projektbewertung
- Projektziel
- Stakeholder
- Kompetenzen

Durch die Technik des Textminings können Ontologien aus unstrukturierten Dokumenten automatisch angereichert werden. Zunächst bietet sich eine Unterscheidung zwischen Textmining und Datamining an. Datamining wird in der Fachliteratur zumeist auf stark strukturierte Daten, wie z. B. Tabellen und generell numerische Werte, beschränkt.[202] Textmining ist hingegen in der Lage, Informationen aus schwach strukturierten Dokumenten zu extrahieren, die einer vorgegebenen Domäne angehören. In der Fachliteratur existieren verschiedene Sichtweisen bezogen auf die Fachgebiete Textmining und Datamining.[203]

197) Beispiele für Online-Bibliotheken sind die der Stanford University (http://www.ksl.stanford.edu/software/ontolingua/) und die DAML ontology library (http://www.daml.org/ontologies/). Des Weiteren stehen einige kommerzielle Ontologien zur Verfügung; vgl. NOY/MCGUINNESS (2001), S. 6.

198) Beispiele für solche Suchmaschinen sind Watson (http://kmi.open.ac.uk/technologies/name/watson/) und Swoogle (swoogle.umbc.edu); vgl. STUCKENSCHMIDT (2011), S. 162. Das Projekt Swoogle wurde 2010 eingestellt.

199) Vgl. STUCKENSCHMIDT (2011), S. 160 ff. Der nachfolgende Satz bezieht sich ebenfalls auf diese Quellenangabe.

200) Vgl. STUCKENSCHMIDT (2011), S. 162 ff.; NOY/MCGUINNESS (2001), S. 6. Der gesamte Absatz bezieht sich auf diese Quellenangaben.

201) Weitere Begriffe werden später in Kapitel 4.2.3 in Abbildung 19 als taxonomisch angeordnete Klassen vorgestellt.

202) Vgl. HAARMANN (2014), S. 43 ff. Der nachfolgende Satz bezieht sich ebenfalls auf diese Quellenangabe.

203) Vgl. MEHLER/WOLFF (2005), S. 9. Massendatengetriebene Ansätze wie das Data Mining und wissensbasierte Ansätze schließen sich nicht aus.

Das Textmining kann zum einen Informationen aus Ontologien verwenden, um in Texten Konzepte der zugrunde liegenden Ontologie wiederzufinden und somit Wissen zu extrahieren und nutzbar zu machen.[204] Zum anderen existieren Techniken computergestützter Ontologie-Erstellung, wie z. B. „Ontology On Demand (OOD)".[205] Diese Technik beinhaltet eine Fachtextsammlung, die als Basis zur vollautomatischen Erstellung einer Domänen-Ontologie dient. Dabei werden zuerst syntaktische und anschließend grammatische Strukturen extrahiert, die im Anschluss zur Erkennung semantischer Zusammenhänge verwendet werden.

Schritt 4: Festlegung der Klassen und der Klassenhierarchie

Bei der Festlegung der Klassenhierarchie mittels der taxonomischen „ist_ein"-Relation kann zwischen drei elementaren Ansätzen unterschieden werden: Top-Down, Bottom-Up und Middle-Out.[206]

Der Top-Down-Ansatz startet mit der Erstellung der „maximal generalisierten", abstraktesten Klasse, die üblicherweise als „Thing" bezeichnet wird. Darauffolgend werden detailliertere Subklassen konzipiert. Der Top-Down-Ansatz gestattet eine gute Kontrolle über den Detaillierungsgrad der Ontologie auf den jeweiligen Ebenen. Er kann jedoch dazu führen, dass eine gewisse Instabilität, vor allem im Hinblick auf die in der Klassenhierarchie höher angeordneten Klassen, zustande kommt.

Beim Bottom-Up-Ansatz ist das Vorgehen genau gegenteilig: Zuerst werden die detailliertesten und konkretesten (Sub-)Klassen erstellt. Im weiteren Verlauf werden die in der Taxonomie höher angesiedelten Klassen mit größerem Abstraktionsgrad spezifiziert. Dieses Vorgehen hat den Effekt, dass die Ontologie einen sehr hohen Detaillierungsgrad besitzt. Dies bedingt einen hohen Konzipierungsaufwand, der dadurch entsteht, dass es schwierig ist, Gemeinsamkeiten zusammenhängender Klassen zu finden, ohne Inkonsistenzen zu verursachen.

Der Middle-Out-Ansatz bildet eine Kombination der beiden vorgenannten Ansätze, indem zuerst die Klassen gebildet werden, die aus Sicht der Modellierer am prägnantesten erscheinen. So können sowohl abstrakte Klassen wie „Eigenschaft" ein prägnantes Konzept darstellen und einen wichtigen „Ankerpunkt" bilden, als auch Subklassen wie „Projektleiter" oder „Projektmitarbeiter", denen im Nachhinein eine Klasse „Akteur" übergeordnet wird. Der Middle-Out-Ansatz bietet einen guten Kompromiss aus einem ausreichenden Detaillierungsgrad, der Konzentration auf essenzielle Klassen und einer hohen Konsistenz der Ontologie. Aus diesen Gründen wird der Middle-Out-Ansatz bei der Konzipierung einer Domänenontologie empfohlen.[207]

Um eine Klassenhierarchie aufzustellen, muss zunächst festgelegt werden, welche der zuvor identifizierten Begriffe als Klassen verwendet werden sollen. Die relevanten Begriffe[208] aus Schritt 3 werden dahingehend untersucht, ob sie sich anderen Begriffen als Instanzen zuordnen lassen. Ist dies der Fall , so werden sie als Instanzen identifiziert; andernfalls handelt es sich um (Sub-)Klassen.[209] Einer Instanz sind keine weiteren Objekte untergeordnet, deshalb befinden sich diese Elemente ausschließlich auf der niedrigsten Stufe einer Ontologie und weisen den höchsten Detaillierungsgrad auf.[210]

Die identifizierten Objekte – Klassen und Instanzen – werden nun mithilfe der taxonomischen „ist_ein"-Relation in Bezug zueinander gesetzt und gehen somit Über- und Unterordnungsbeziehungen ein.[211]

---

204) Vgl. HAARMANN (2014), S. 48 ff.

205) Vgl. HAARMANN (2014). Die nachfolgenden zwei Sätze beziehen sich ebenfalls auf diese Quellenangabe.

206) Vgl. NOY/MCGUINNESS (2001), S. 6 f. Zu den Vor- und Nachteilen der jeweiligen Ansätze vgl. USCHOLD/GRUNINGER (1996), S. 20 ff.

207) Vgl. NOY/MCGUINNESS (2001), S. 7; USCHOLD/GRUNINGER (1996), S. 21.

208) Die Begriffe werden in diesem Zusammenhang auch als Entitäten bezeichnet.

209) Vgl. BEIßEL (2011), S. 143.

210) Vgl. KOWALSKI/ZELEWSKI (2015), S. 604; NOY/MCGUINNESS (2001), S. 18.

211) Vgl. KOWALSKI/ZELEWSKI (2015), S. 604.

Eine wichtige Prüfung, die streng genommen bereits in Schritt 3 stattfinden sollte, ist die Beschränkung oder strenge Fokussierung auf die darzustellende Domäne: Jede Klasse der Ontologie sollte diese kritischen Prüfung bestehen, da die Ontologie sonst einerseits unnötig komplex wird und andererseits für eine Wiederverwendung durch andere Nutzer schwer in ihrer Gänze zu erfassen ist.[212]

Ontologie-Editoren bieten die Möglichkeit, Klassen als disjunkt zueinander festzulegen. Diese Klassen können keine gemeinsamen Instanzen besitzen. Dies ermöglicht, eine Ontologie mithilfe eines Reasoners innerhalb des Ontologie-Editors auf Modellierungsfehler hin computergestützt zu prüfen.[213] Die Möglichkeit dieser Abgrenzung von Klassen ist auch deshalb vorteilhaft, weil der Ontologierepräsentationssprache OWL die sogenannte Open World Assumption zugrunde liegt. Durch eine Anreicherung der Semantik einer Ontologie, z. B. durch disjunkte Klassen, wird die Aussagekraft einer Ontologie zweckmäßig eingeschränkt.

Schritt 5: Definition der nicht-taxonomischen Relationen und Attribute

Nachdem die Klassen und die Taxonomie definiert wurden, folgt die Definition der nicht-taxonomischen Relationen und Attribute der Ontologie. Während der Erstellung der Klassen wird durch die systembedingte Hierarchisierung „automatisch" die taxonomische Relation „ist_ein" zwischen den erstellten Klassen gebildet.[214] Zur Steigerung der Ausdrucksstärke einer Ontologie werden weiterführende nicht-taxonomische Relationen und Attribute ergänzt.

Diejenigen Begriffe, die nach der Durchführung von Schritt 4 bei der Identifikation relevanter Begriffe aus Schritt 3 nicht als Klassen definiert wurden, müssen Attribute, Instanzen oder nicht-taxonomische Relationen darstellen.[215] Die Definiton von nicht-taxonomischen Relationen und Attributen richtet sich nach der Maßgabe, welche sprachlichen Ausdrucksmittel für die Repräsentation des betrachteten Realitätsausschnitts als notwendig erachtet werden.

Sämtliche Attribute und nicht-taxonomische Relationen einer Klasse werden auf alle Subklassen und Instanzen dieser Klasse vererbt. Daraus folgt, dass Attribute und nicht-taxonomische Relationen immer der in der Klassenhierarchie höchstmöglichen Klasse[216] zugeordnet werden sollten, für die sie noch passen.[217] Das Ausnutzen der Vererbung ermöglicht es, die Anzahl nicht-taxonomischer Relationen und Attribute möglichst gering und übersichtlch zu halten und zugleich eine möglichst große Ausdrucksstärke zu ermöglichen.

Die 101-Methode von NOY und MCGUINNESS bezieht sich in diesem Schritt auf die Erstellung von „Slots". Darunter werden nicht-taxonomische Relationen und Attribute subsumiert. Der Begriff „Slot" wird in Protégé seit der Version 5 nicht mehr verwendet, stattdessen besitzen Attribute und nicht-taxonomische Relationen seitdem keine Oberkategorie mehr.

212) Vgl. NOY/MCGUINNESS (2001), S. 19 f.

213) Vgl. NOY/MCGUINNESS (2001), S. 20.

214) Vgl. BEIßEL (2011), S. 147.

215) Vgl. NOY/MCGUINNESS (2001), S. 8. Begriffe, die nach der Durchführung von Schritt 5 weder als Klasse noch als Attribut oder als nicht-taxonomische Relation in die Ontologie eingebracht wurden, können entweder als nicht relevant eingestuft und ignoriert werden oder aber einen Teil der Schritte noch einmal durchlaufen. Im letztgenannten Fall bietet es sich an, mit diesen Begriffen die Schritte 3 bis 5 zu wiederholen, um einen möglichen Informationsverlust zu vermeiden und eine möglichst zielführende Konstruktion der sprachlichen Ausdrucksmittel für die Beschreibung des in der Domäne vorhandenen Wissens durch eine Ontologie zu ermöglichen.

216) Je höher eine Klasse in der Klassenhierarchie angesiedelt ist, desto allgemeiner ist die Klasse und je geringer ist die Granularität der Klasse.

217) Vgl. NOY/MCGUINNESS (2001), S. 9.

Schritt 6: Definition der Wertebereiche und Kardinalitäten von Attributen und Relationen

Die Definition der Wertebereiche lässt sich direkt aus den für plausibel erachteten Werten derjenigen Attribute und nicht-taxonomischen Relationen ableiten, die in Schritt 3 identifiziert wurden.[218] Die Wertebereiche sind so zu definieren, dass sie mindestens alle plausiblen Werte umfassen.

Der Ontologie-Editor Protégé bietet eine Auswahl an Wertebereichen für Attribute. Nachfolgend werden die am häufigsten auftretenden[219] Wertebereiche kurz erläutert: *String* wird für Attribute verwendet, die eine endliche Folge von Zeichen als Werte annehmen können. Ein Beispiel ist eine Seriennummer (bestehend aus Buchstaben und Ziffern) eines Bauteils. *Float* und *Integer* werden für Attribute verwendet, die Zahlenwerte annehmen können. Im Falle von *Float* handelt es sich um Gleitkommazahlen, also angenäherte reelle Zahlen (mit einer endlichen Zahl von Dezimalstellen). *Integer* bezeichnet ganzzahlige Werte. Beide Wertebereiche können durch die Angabe eines Minimal- und Maximalwerts begrenzt werden. *Boolean* wird dann verwendet, wenn der Wert eines Attributes entweder „wahr“ oder „falsch“ lauten kann.

Die Kardinalität von Attributen und nicht-taxonomischen Relationen definiert, wie viele Werte ein Attribut bzw. eine nicht-taxonomische Relation gleichzeitig anzunehmen vermg.[220] Der Bereich erstreckt sich von einer einfachen Kardinalität mit genau einem erlaubten Wert bis hin zu einer mehrfachen Kardinalität mit einer beliebig großen, aber jeweils endlichen Anzahl an Werten. Beispielsweise weist das Attribut „ProjektName“ der Klasse „Projekt“ eine Kardinalität von 1 auf, da ein Projekt zur eindeutigen Identifikation nur genau einen und somit eindeutigen Namen aufweisen darf. Hingegen besitzt das Attribut „Transportmittel“ der Klasse „Transportlogistik“ eine Kardinalität von 3, wenn in der zugrunde liegenden Ontologie „Lkw“, „Schiff“ und „Flugzeug“ als zulässige Werte existieren.

Je höher die maximale Kardinalität von Attributen und nicht-taxonomischen Relationen in einer Ontologie ist, desto komplexer gestaltet sich eine Berechnung vo Ähnlichkeiten zwischen Projekten durch einen Ähnlichkeitsalgorithmus.[221] In diesem Fall müssen verschiedene Szenarien mithilfe verschiedener Attributswerte berechnet werden, die einen optimistischen, einen durchschnittlichen und einen pessimistischen Ansatz unterstellen.[222] Je nach Wahrscheinlichkeit von z. B. angenommenen Umsätzen werden unterschiedliche Ähnlichkeitswerte berechnet.

Schritt 7: Erzeugung von Instanzen

In diesem letzten Schritt werden den Klassen der Ontologie Instanzen zugeordnet.[223] Die Entscheidung, ob eine Klasse oder eine Instanz vorliegt, kann nicht pauschal getroffen werden.[224] Eine Instanz besitzt immer eine höhere Granularität als ihre Klasse. Die Frage, ob eine Instanz oder eine Klasse vorliegt, hängt von der Domäne der Ontologie und von dem individuellen Schwerpunkt ab, den der Ontologie-Ersteller festlegt. In einer Ontologie mit der Domäne Projektmanagement kann beispielsweise „CBR-System“ eine Instanz der Klasse „Projektmanagementsystem“ darstellen. In einer anderen Ontologie lässt sich diese Instanz als eine Klasse deklarieren, weil der Fokus beispielsweise auf Wissensmanagementsystemen liegt, sodass die Klasse „CBR-System“ in eine Vielfalt spezieller Varianten von CBR-Systemen als Instanzen ausdifferenziert wird.

218) Vgl. BEIßEL (2011), S. 152. Der nachfolgende Satz bezieht sich ebenfalls auf diese Quelle.

219) Vgl. NOY/MCGUINNESS (2001), S. 9 f.

220) Vgl. NOY/MCGUINNESS (2001), S. 9. Der gesamte Absatz bezieht sich auf diese Quelle.

221) Vgl. BEIßEL (2011), S. 153.

222) Vgl. BERGMANN (1998), S. 8 f. Der nachfolgende Satz bezieht sich ebenfalls auf diese Quelle

223) Vgl. NOY/MCGUINNESS (2001), S. 11.

224) Vgl. NOY/MCGUINNESS (2001), S. 18 f. Die nachfolgenden zwei Sätze beziehen sich ebenfalls auf diese Quelle.

3.4 Implementierung des ontologiegestützten CBR-Systems jCORA

Die Erstellung einer domänenspezifischen Ontologie leistet für sich genommen nur einen Beitrag[225] zur Wiederverwendung von projektspezifischem Erfahrungswissen. Das zuvor teilweise implizit vorhandene Wissen ist durch die Explikation mithilfe der sprachlichen Ausdrucksmittel einer Ontologie zwar „anschaulicher" und „greifbarer" geworden, jedoch kommt eine wirkliche Nutzung dieses Erfahrungswissens, speziell computergestützt, ohne weitere Softwareunterstützung noch nicht in Betracht. Dieses Erfahrungswissen kann vor allem mithilfe eines CBR-Systems computergestützt genutzt und wiederverwendet werden.

In der Fachliteratur existiert eine Vielzahl von CBR-Systemen, von denen das Tool jCORA[226] im KI-LiveS-Projekt verwendet wird.[227] Auf dieses spezielle CBR-System jCORA fokussiert sich der vorliegende Projektbericht, der ebensfalls im Kontext des KI-LiveS-Projekts entstanden ist.

Um das CBR-System jCORA implementieren zu können, muss zunächst ein passendes OWL-Framework ausgewählt werden.[228] OWL-Frameworks bieten üblicherweise die folgenden Funktionen:[229]

- Bereitstellen von Speicherplatz im Arbeitsspeicher (RAM[230]), im Festspeicher oder in Datenbanken für das Speichern von Ontologien;
- Füllen des Speichers mit Daten aus Dateien, Netzwerkspeichern, Datenbanken oder eigens konstruierten Aussagen;
- Verknüpfen der Daten durch die Bildung von Schnitt- und Vereinigungsmengen in Bezug auf Klassen;
- Erzeugen zusätzlicher Informationen durch internes und externes Reasoning und das Aufdecken von Problemen zwischen Datensätzen;
- Stellen von Anfragen an das System und Visualisierung der Ergebnispfade innerhalb der Ontologie;
- Exportieren der Ontologiedaten in verschiedene Standardformate[231];
- Löschen der Daten und Freigeben des Speichers.

---

225) Dieser Beitrag ist z. B. auf das Wissen begrenzt, das durch eine Fluktuation von Mitarbeitern generell oder innerhalb der Bearbeitungszeit eines Projekts ansonsten das Unternehmen verlassen hätte. Dank der Durchführung von Experteninterviews und der Wiederverwendung von Projektberichten konnte dieses zum Teil implizite Wissen der Mitarbeiter in einer expliziten Form in Gestalt einer Ontologie konserviert werden.

226) Bei jCORA handelt es sich um ein Akronym aus den Worten „java based Case- and Ontology-based Reasoning Application".

227) Der Prototyp jCORA wurde im Rahmen des Verbundprojekts „Organisatorische Innovationen mit Good Governance in Logistik-Netzwerken" (OrGoLo) mit dem Ziel entwickelt, ontologiegestütztes Case-based Reasoning mit heterogenen und zeitlich variablen Fallstrukturen durchführen zu können und der Anforderung zu genügen, einen möglichst „breit aufgestellten" Algorithmus zur Ähnlichkeitsbestimmung zwischen Fällen zu bieten; vgl. BERGENRODT/KOWALSKI/ZELEWSKI (2015), S. 480.

228) Vgl. BERGENRODT/KOWALSKI (2015), S. 34.

229) Vgl. HEBELER/FISHER/BLACE et al. (2009), S. 267 ff. Die nachfolgende Aufzählung bezieht sich ebenfalls auf diese Quellenangabe.

230) Das Akronym RAM steht für „Random-Access Memory" und bezeichnet einen Direktzugriffsspeicher, der z. B. in Computern aktuell benötigte Daten so lange speichert, bis die Komponenten des Computers diese nicht mehr benötigen oder der Speicher stromlos geschaltet wird.

231) Zu den Standardformaten zählen RDF (Resource Description Framework), XML (Extensible Markup Language), OWL (Web Ontology Language), Turtle (Terse RDF Triple Language), die Manchester-Syntax und die funktionale Syntax, vgl. BERGENRODT/KOWALSKI/ZELEWSKI (2015), S. 483.

Die in der Praxis meist verbreiteten Semantic Web Frameworks sind Apache Jena, Sesame und OWL API.[232] Im Rahmen der Implementierung von jCORA wird das Framework OWL API verwendet, da der Ontologie-Editor Protégé darauf basiert und somit sowohl OWL-Ontologien als auch diverse OWL-Reasoner unterstützt.[233]

OWL API zeichnet sich im Gegensatz zu den Frameworks Apache Jena und Sesame dadurch aus, dass es auf der Methodenebene eine geringere Granularität besitzt.[234] Das Framework OWL API gestattet ein einfaches und schnelles Manipulieren von OWL-Dateien und ist dabei weitestgehend unabhängig von der zugrundeliegenden Technologie des Frameworks. Im Gegensatz dazu bietet Apache Jena Entwicklern einen vollständigen Zugriff auf Technologien und Datenstrukturen, was das Bearbeiten von OWL-Dateien zwar aufwändiger, aber auch flexibler gestaltet. Im Rahmen der Implementierung von jCORA fiel die Entscheidung auf das Framework Apache Jena, da es mehr Möglichkeiten für eine spätere Erweiterung der Ähnlichkeitsbestimmung bietet.

In jCORA wird innerhalb der Architektur der Fallbasis zwischen einer globalen Domänenontologie und lokalen Fallontologien unterschieden.[235] Fallspezifische Instanzen, Relations- und Attributwerte werden in den lokalen Fallontologien abgelegt. Dagegen werden Klassen, Instanzen, Relationen und Attribute, die allgemeine sprachliche Ausdrucksmittel einer Domäne repräsentieren, in der globalen Domänenontologie abgelegt. Die globale Domänenontologie und eine lokale Fallontologie bilden zusammen eine Anwendungs- oder Fallontologie. Durch diese Architektur ergeben sich vier Schritte beim Laden eines Falls:

- Laden des Falls aus einer Datei von der lokalen Festplatte,
- Laden der lokalen Fallontologie aus einem Tripel-Speicher[236],
- Zusammenführen der globalen Domänen- und der lokalen Fallontologie zur Anwendungs- oder Fallontologie,
- Erstellen des Inferenzmodells[237] mittels Pellet Reasoner[238] und Präsentation des Inferenzmodells für den Nutzer.

Sämtliche durch Nutzer in jCORA durchgeführten Veränderungen werden in die lokale Fallontologie aufgenommen.[239] Wird der Fall gespeichert, so wird der veränderte Fall in den Tripel-Speicher übernommen und steht dort für die zukünftige Wiederverwendung des Wissens zur Verfügung. Vorteilhaft bei diesem Vorgehen ist, dass die globale Domänenontologie jederzeit einfach durch einen Ontologie-Editor modifiziert werden kann und dass sich Änderungen direkt auf alle Fälle in der Fallbasis auswirken.

---

232) Vgl. HEBELER/FISHER/BLACE et al. (2009), S. 153 f.; PÖHLAND/KORZETZ/SCHLEGEL (2013), S. 75 f. Eine Zusammenstellung weiterer Semantic Web Frameworks findet sich in HEBELER/FISHER/BLACE et al. (2009), S. 153 ff.

233) Vgl. WATERFELD/WEITEN/HAASE (2011), S. 64.

234) Vgl. BERGENRODT/KOWALSKI (2015), S. 35. Der gesamte Absatz bezieht sich auf diese Quellenangabe.

235) Vgl. BERGENRODT/KOWALSKI (2015), S. 36. Der gesamte Absatz bezieht sich auf diese Quellenangabe.

236) Der Tripel-Speicher ist eine spezifische Komponente des OWL Frameworks Apache Jena. In diesem Speicher werden Ontologien als eine Liste von Tripeln gespeichert; vgl. BERGENRODT/KOWALSKI/ZELEWSKI (2015), S. 513.

237) Ein Inferenzmodell stellt eine „erweiterte" Ontologie dar, die neben der eigentlichen Ontologie sämtliche Aussagen-Tripel in der Standard-Form „Subjekt – Prädikat – Objekt" beinhaltet, die mithilfe eines Reasoners aus der Ontologie abgeleitet wurden. Mittels Transitivitätsregeln können weitere Aussagen abgeleitet werden; vgl. BERGENRODT/KOWALSKI/ZELEWSKI (2015), S. 515.

238) Der „Pellet Reasoner" ist eine Softwarekomponente, die Inferenzen auf Ontologien durchführen kann, wenn die Ontologien im OWL-DL-Standard erstellt worden sind. Das Akronym DL steht hier für „Description Logic", deren Ausdrucksreichtum etwas unterhalb der Prädikatenlogik 1. Stufe einzuordnen ist. Der „Pellet Reasoner" kann als Plugin in Protégé eingesetzt werden. Vgl. BERGENRODT/KOWALSKI/ZELEWSKI (2015), S. 515.

239) Vgl. BERGENRODT/KOWALSKI/ZELEWSKI (2015), S. 515. Der gesamte Absatz bezieht sich auf diese Quellenangabe.

Nutzer müssen bei der Modifikation ein hohes Maß an Aufmerksamkeit darauf legen, dass die Komponenten (Klassen, Attribute, Relationen und Instanzen) innerhalb der globalen Domänenontologie nicht umbenannt oder gelöscht werden dürfen, wenn diese in den lokalen Fallontologien verwendet werden. Andernfalls kann es zu Inkonsistenzen, Restriktionsverletzungen oder anderen unerwünschten Fehlern kommen.

Das CBR-System jCORA stellt Anforderungen an die globale Domänenontologie, wie z. B. das Vorhandensein einer dreiteiligen Fallstruktur.[240] Innerhalb der Ontologie müssens neben der Klasse „Projekt“ die drei strukturkonstituierenden Klassen „Projektbeschreibung[241]“, „Projektlösung“ und „Projektbewertung“ vorhanden sein. Eine weitere Anforderung ist, dass die drei zuletzt genannten Klassen mittels entsprechender Relationen mit der Klasse „Projekt“ in Beziehung gesetzt werden.[242] Die globale Domänen-Ontologie darf auch keine Instanzen der Klassen „Projekt“, „Projektbeschreibung“, „Projektlösung“ und „Projektbewertung“ beinhalten, da ansonsten nicht festgestellt werden kann, welche zwei Instanzen die Basis für eine Ähnlichkeitsberechnung darstellen.[243] Dieses Problem korrespondierender Instanzen wird als „Ontology-Alignment-Problem“ diskutiert[244] und in jCORA durch die zuvor genannten Anforderungen an die Domänenontologie und durch die Separierung der lokalen Fallontologien von der globalen Domänenontologie verhindert.[245]

240) Vgl. ASSALI/LENNE/DEBRAY (2010), S. 102 f.; KURBEL/DORNHOFF (1993), S. 1052 ff. Die Fallstruktur der Ontologie von ASSALI/LENNE/DEBRAY umfasst jedoch nur die zwei Komponenten „Problem“ und „Lösung“. Eine dreiteilige Fallstruktur entspricht der üblichen Fallstruktur im Bereich des Case-based Reasonings. Eine umfassende Aufzählung entsprechender Literatur findet sich in ZELEWSKI/KOWALSKI/BERGENRODT (2015), S. 242.

241) Analog zu dem Begriff „Projekt“ kann auch der Begriff „Fall“ benutzt werden, vgl. BERGENRODT/KOWALSKI/ZELEWSKI (2015), S. 516. Wichtig ist, dass eine übereinstimmende Terminologie zwischen den drei zuvor erwähnten Klassen und den Einstellungen in jCORA besteht. Das CBR-Tool kann keine Ähnlichkeitsberechnung vornehmen, wenn beispielsweise in der Domänenontologie der Terminus „Projektbeschreibung“ benutzt wird und in der Fallontologie der Terminus „Fallbeschreibung“.

242) Vgl. BERGENRODT/KOWALSKI/ZELEWSKI (2015), S. 516.

243) Vgl. BERGENRODT/KOWALSKI/ZELEWSKI (2015), S. 516 f.

244) Vgl. SCHARFFE/FENSEL (2008), S. 83 ff.; FOSSATI/GHIDONI/DI EUGENIO et al. (2006), S. 51 ff.

245) Vgl. BERGENRODT/KOWALSKI (2015), S. 38.

4 Erstellung des Vorgehensmodells

4.1 Modellierungssprache Business Process Model and Notation

4.1.1 Grundlagen

Ein Vorgehensmodell soll in einer benutzerfreundlichen Darstellungsweise wiedergeben, wie hinsichtlich der Erfüllung komplexer, in der Regel neuartiger Aufgaben systematisch vorzugehen ist. Ein Vorgehensmodell zur Erstellung eines ontologiegestützten CBR-Systems wird dadurch erforderlich, dass die Schritte bis zur Inbetriebnahme eines solchen Systems in der betrieblichen Praxis noch nicht etabliert sind und es sich diesbezüglich aus betriebswirtschaftlicher Perspektive nicht um allgemein verfügbares Wissen, sondern um „Expertenwissen" handelt. Diese Schritte bedürfen daher einer ausführlichen Erläuterung, insbesondere in Hinblick auf die Themenfelder Case-based Reasoning und Ontologien. Sie werde im Folgenden mithilfe der Modellierungssprache Business Process Model and Notation (BPMN) dargestellt und mit Screenshots der verwendeten Software „Microsoft Visio" visualisiert, die über ein benutzerfreundliches Plug-in für die Grafiksymbole der Modellierungssprache BPMN verfügt.

Um sich in die Modellierungssprache BPMN hineinzuversetzen, ist es hilfreich, zunächst den Aufgabenbereich „Business Process Management" (BPM) zu definieren.[246] Für Business Process Management existiert jedoch keine einheitliche Definition.[247]

In der deutschen Fassung des Referenzwerks „BPM Common Body of Knowledge" definiert die European Association of BPM (EABPM) „Business Process Management" als synonyme Bezeichnung für „Geschäftsprozessmanagement" und „Prozessmanagement".[248] Ein Prozess wird als eine Reihe von Aktivitäten verstanden, die von Menschen oder Maschinen ausgeführt werden, um Ziele zu erreichen. Im Vordergrund steht dabei die Generierung eines Kundennutzens und somit auch die Wertsteigerung für das betroffene Unternehmen. „Business Process Management (BPM) ist ein systematischer Ansatz, um sowohl automatisierte als auch nicht automatisierte Prozesse zu erfassen, zu gestalten, auszuführen, zu dokumentieren, zu messen, zu überwachen und zu steuern und damit nachhaltig die mit der Unternehmensstrategie abgestimmten Ziele zu erreichen."[249]

Business Process Management wird überwiegend eingesetzt, um:[250]

1) bestehende Prozesse organisatorisch oder informationstechnisch zu verbessern,
2) bestehende Prozesse zu dokumentieren und
3) neue Prozesse einzuführen.

Der letztgenannte, dritte Einsatzbereich steht im Vordergrund dieses Projektberichts.

Die Zielsetzung der Modellierungssprache BPMN besteht darin, eine standardisierte grafische Prozessnotation bereitzustellen. Zu diesem Standardisierungsziel trug vor allem bei, dass diese Modellierungssprache vom Unternehmen IBM (International Business Machines Corporation) übernommen und seitens der Object Management Group (OMG) seit dem Jahr 2005 propagiert wurde.[251] Die OMG ist in der IT-Welt unter anderem durch den Modellierungsstandard Unified Modeling Language (UML) bekannt und verhalf der Modellierungssprache BPMN durch deren Standardisierung zu weltweiter Bekanntheit und Akzeptanz.

246) Vgl. FREUND/RÜCKER (2019), S. 1.

247) Vgl. BANDARA/HARMON/ROSEMANN (2011), S. 759 f.

248) Vgl. FREUND/RÜCKER (2019), S. 1. Die nachfolgenden zwei Sätze beziehen sich ebenfalls auf diese Quelle.

249) FREUND/RÜCKER (2019), S.1.

250) Vgl. FREUND/RÜCKER (2019), S. 2.

251) Vgl. FREUND/RÜCKER (2019), S. 7. Der nachfolgende Satz bezieht sich ebenfalls auf diese Quelle.

Das Regelwerk der Modellierungssprache BPMN, das aus allen verfügbaren Grafik-Symbolen, ihren Bedeutungen und ihren Kombinationsregeln besteht, lassen sich als BPMN-Spezifikation auf der Website der OMG abrufen.[252]

BPMN-Prozessmodelle können seit der Version 2.0 direkt in Workflow Engines[253] ausgeführt werden.[254] Dies hat zu einer gesteigerten Verwendung der Modellierungssprache geführt. Zuvor mussten BPMN-Prozessmodelle für Workflow Engines konvertiert werden, was aufgrund teils fehlender technischer Attribute zu Fehlern führen konnte.

4.1.2 Notationselemente

Dieses Kapitel dient der Vorstellung der Notationselemente der Modellierungssprache BPMN, die in diesem Projektbericht verwendet werden.[255] Die nachfolgende Abbildung 7 gewährt einen Überblick über die typischen Notationselemente der Modellierungssprache BPMN, auch wenn sie in diesem Projektbericht nicht vollständig ausgeschöpft werden.

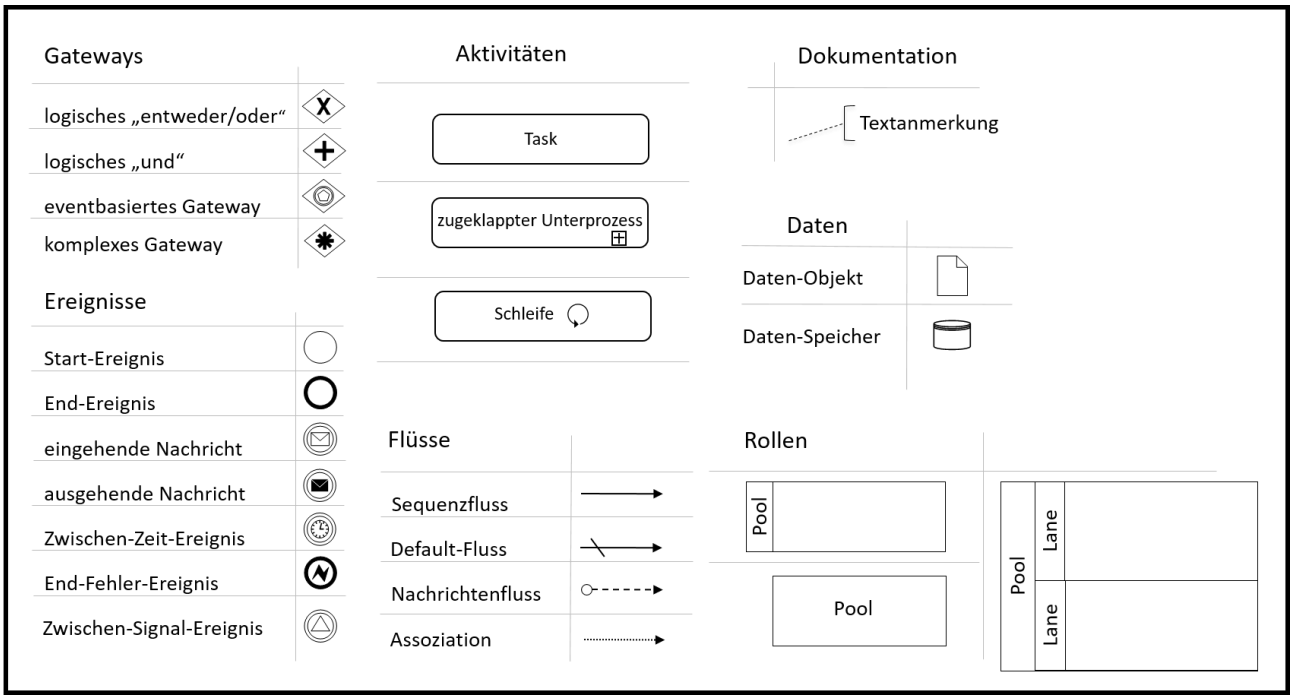

Abbildung 7: Notationselemente der Modellierungssprache BPMN

252) Vgl. Object Management Group (2013). Die Spezifikation der aktuellen Version 2.0.2 lässt sich im Internet unter der URL https://www.omg.org/spec/BPMN/2.0.2/PDF aufrufen.

253) Eine Workflow Engine ist eine Software zur Steuerung von Prozessen. Sie dient als Schnittstelle zwischen technischen und menschlichen Systemen.

254) Vgl. FREUND/RÜCKER (2019), S. 8. Die beiden nachfolgenden Sätze beziehen sich ebenfalls auf diese Quellenangabe.

255) Eine Übersicht über die wichtigsten Notationselemente findet sich in der BPMN-Spezifikation, vgl. OBJECT MANAGEMENT GROUP (2013), S. 26 ff. Einen einfachen Zugang zur Modellierungssprache BPMN bietet z. B. FREUND/RÜCKER (2019), S. 28 ff. Die nachfolgende Ausführung über die Notationselemente der BPMN bezieht sich ebenfalls auf diese Quellenangabe.

Um die Wirkungszusammenhänge der Prozesse zu verdeutlichen, die mit der Modellierungssprache BPMN dargestellt werden, kann das Token-Konzept verwendet werden.[256] Ein Token ist ein abstraktes Konstrukt, das alle zulässigen Prozesspfade durchlaufen kann und somit auch komplexe Prozesse verständlich macht.[257] Ein Token wird in einem Startereignis „geboren“ und bewegt sich anschließend entlang der Prozesspfade, bis es in einem Endereignis „konsumiert“ wird. Sind alle während eines Prozesses entstandene Token konsumiert worden, endet der betrofene Prozess. Das Token-Prinzip ist ebenfalls hilfreich, um Modellierungsfehler innerhalb von Prozessen aufzudecken. Sollte z. B. ein Token aufgrund eines Modellierungsfehlers nicht konsumiert werden können, tritt ein Laufzeitfehler auf.

Aufgaben

Aufgaben legen fest, was von den Prozessteilnehmern ausgeführt werden soll. Im vorliegenden Projektbericht wird stets versucht, Aufgaben nach dem Objekt-Verrichtungsprinzip[258] zu formulieren. Das bedeutet, dass Aufgaben stets aus einem Tupel aus einem Objekt und einem Verb (für die Verrichtung, die am Objekt vollzogen wird) bestehen.

Ereignisse

Ereignisse stellen wichtige Geschehnisse innerhalb eines Prozesses dar. Die Modellierungssprache BPMN schreibt das Vorhandensein von Start- und Endereignissen nicht verbindlich vor, jedoch sind Endereignisse am Ende eines jeden Pfads dann verpflichtend einzufügen, wenn mindestens ein Startereignis formuliert wurde. Zwischenereignisse können überall dort eingefügt werden, wo ein Prozess einen beachtenswerten Status erreicht hat, der explizit festgehalten werden soll. Zwischen- und Endereignisse lassen sich oftmals als Meilensteinereignisse auffassen, die im betrieblichen Projektmanagement vertraut sind.

Ereignisse lassen sich in eintretende und ausgelöste Ereignisse unterteilen. Startereignisse sind stets eintretende Ereignisse, die unabhängig vom eigentlichen Prozess eintreten. Zwischenereignisse können sowohl eintretende Ereignisse als auch ausgelöste Ereignisse sein. In der Blanko-Form ist das Zwischenereignis dadurch gekennzeichnet, dass es durch den Prozessfortschritt erreicht wird. Endereignisse können nur durch einen Prozess ausgelöst werden und nicht unabhängig von diesem Prozess eintreten. Ereignisse werden in diesem Projektbericht aus einem Objekt gefolgt von einem „passivierten Verb“ (gemeint ist ein von einem Verb abgeleitetes Partizip Perfekt Passiv) formuliert.[259]

Sequenzflüsse

Durch Sequenzflüsse wird die zeitlich-logische Reihenfolge dargestellt, in der die Notationselemente (Aufgaben und Ereignisse) zueinander stehen. Die Token bewegen sich entlang der Sequenzflüsse und visualisieren so die Durchführung der betroffenen Prozesse.

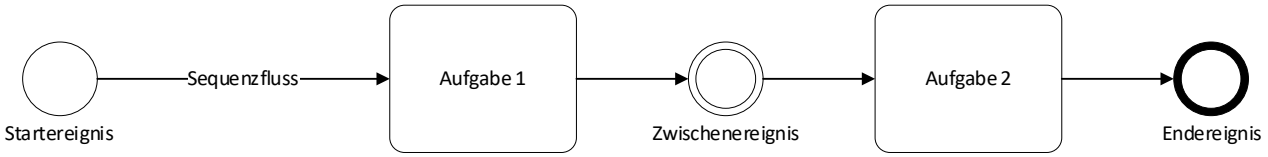

Abbildung 8: BPMN-Notation für einen Sequenzfluss

256) Vgl. FREUND/RÜCKER (2019), S. 31. Der gesamte Absatz bezieht sich auf diese Quellenangabe.

257) Ein Token der Modellierungssprache BPMN gleicht strukturell einer „Marke“ (oder einem „dot“), die in PETRI-Netzen zur Visualisierung von Prozessdurchführungen, insbesondere zur Visualisierung des prozessbedingten Flusses von (Daten-)Objekten durch ein System, verwendet werden.

258) Vgl. FREUND/RÜCKER (2019), S. 33.

259) Vgl. zum Vorgehen FREUND/RÜCKER (2019), S. 33 f.

Datenbasierte exklusive Gateways[260]

Das datenbasierte exklusive Gateway (XOR[261]-Gateway) teilt einen eingehenden Sequenzfluss in mindestens zwei ausgehende Sequenzflüsse auf, von denen nur einer durchlaufen werden kann. Welcher Pfad durchlaufen wird, entscheidet sich aufgrund der Entscheidung, mit der auf die durch das Gateway gestellte Entscheidungsfrage geantwortet wird. Die ausgehenden Pfade sind mit den möglichen, sich gegenseitig ausschließenden Antworten („Daten“) auf die Entscheidungsfrage beschriftet. Für das Beispiel in der folgenden Abbildung 9 bedeutet dies, dass ein Token nur über genau einen der beiden ausgehenden Pfade geleitet werden kann, und zwar je nach der getroffenen Entscheidung am XOR-Gateway. Nachdem entweder die Aufgabe 2 oder aber die Aufgabe 3 erfüllt wurde, gelangt der Token über ein zweites, zusammenführendes XOR-Gateway weiter zum Endereignis, durch das er konsumiert wird.

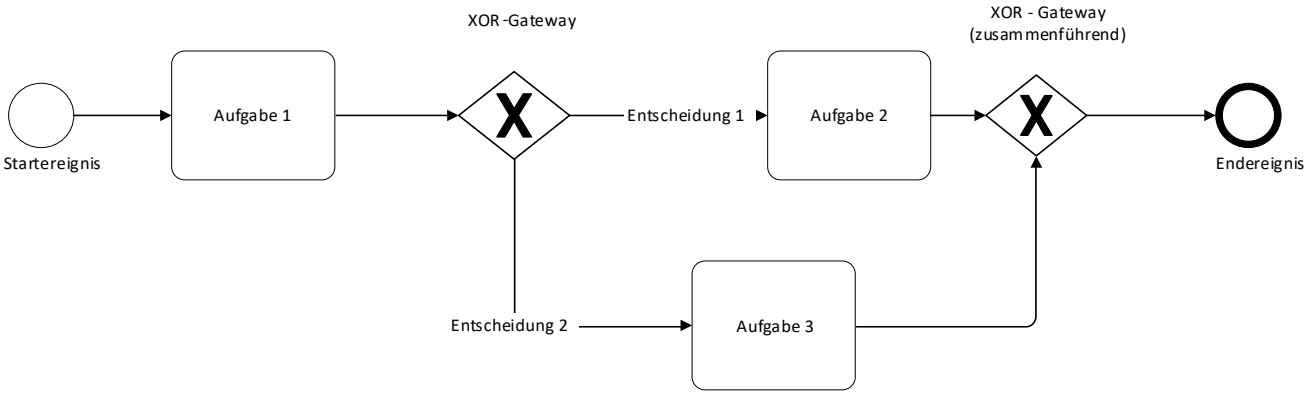

Abbildung 9: BPMN-Notation für ein datenbasiertes exklusives Gateway

Ereignisbasierte exklusive Gateways

Das ereignisbasierte exklusive Gateway teilt den eingehenden Sequenzfluss in mindestens zwei ausgehende Sequenzflüsse auf, von denen genau einer durchlaufen werden kann. Im Gegensatz zum datenbasierten exklusiven Gateway wird der ausgehende Pfad des Tokens nicht mithilfe einer Entscheidungsfrage ermittelt, sondern durch Zwischenereignisse, die eintreten können, aber nicht müssen. Der Token wartet so lange am Gateway, bis ein Zwischenereignis eingetreten ist, und bewegt sich dann innerhalb des Sequenzflusses desjenigen ausgehenden Pfads weiter, dessen Zwischenereignis zuerst eingetreten ist.[262] Vgl. dazu die Abbildung 10 auf der nächsten Seite.

260) Vgl. FREUND/RÜCKER (2019), S. 34 ff. Der nachfolgende Absatz bezieht sich ebenfalls auf diese Quellenangabe.

261) Das exklusive OR (XOR) steht aus logischer Sicht für das exklusive „oder“.

262) Der denkmögliche Fall, dass mindestens zwei der Zwischenereignisse zum selben Zeitpunkt eintreten, wird in der Modellierungssprache BPMN nicht präzise geregelt. In diesem Fall ist nicht eindeutig, welcher der ausgehenden Pfade, deren Zwischenereignisse zeitgleich eingetreten sind, vom Token weiter durchlaufen wird.

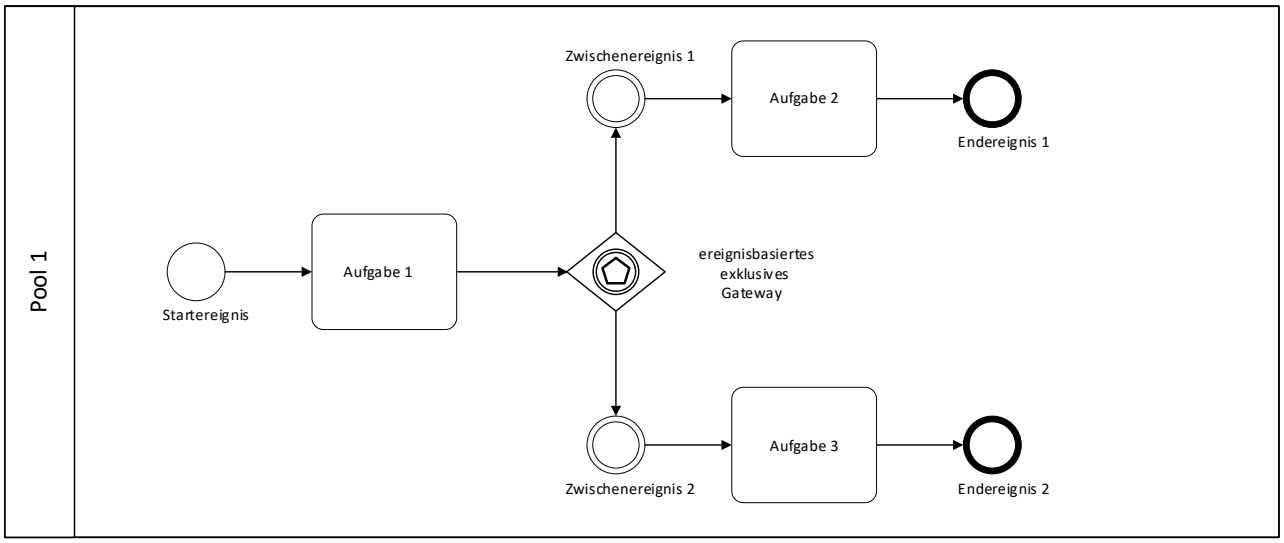

Abbildung 10: BPMN-Notation für ein ereignisbasiertes exklusives Gateway

Parallele Gateways

Das parallele Gateway (AND-Gateway) teilt ebenfalls einen eingehenden in mehrere ausgehende Sequenzflüsse auf, allerdings ohne eine Entscheidungsfrage. Stattdessen werden nach dem Gateway alle ausgehenden Pfade durchlaufen; somit findet eine Parallelisierung des Sequenzflusses statt. Im Beispiel der nachfolgenden Abbildung 11 werden die beiden ausgehenden Pfade nach der Erfüllung der Aufgaben 2 und 3 durch ein weiteres paralleles Gateway wieder zusammengeführt und endet mit dem Endereignis.

Das Token-Prinzip verdeutlicht in diesem Zusammenhang eine Besonderheit des parallelen Gateways: Nachdem der Token durch das Startereignis geboren wurde, wird er durch das parallele Gateway in so viele Kopien vermehrt, wie ausgehende Pfade existieren. In diesem Beispiel „wandert" je ein Token durch Aufgabe 2 und durch Aufgabe 3.

Zur Verdeutlichung der Funktionsweise des zusammenführenden, parallelen Gateways wurde in diesem Beispiel ein weiteres Notationselement eingeführt: die (Text-)Anmerkung. Dieses kann nach Belieben an sämtliche Notationselemente angebracht werden, um Prozessdetails zu erläutern oder zu konkretisieren. Im Beispiel wurde eine Angabe der Durchlaufzeit der Aufgaben 2 und 3 ergänzt, um die Funktionsweise des zusammenführenden, parallelen Gateways zu veranschaulichen. Nachdem die Aufgabe 2 nach fünf Minuten erfüllt wurde, wandert der Token zum zusammenführenden, parallelen Gateway und wartet dort so lange, bis dort auch der andere Token angekommen ist, nachdem die Aufgabe 1 nach sieben Minuten erfüllt wurde. Sobald dies der Fall ist, werden die beiden Token zu nur noch einem Token fusioniert und wandern weiter zum Endereignis, wo der Token konsumiert und der Prozess nach sieben Minuten beendet wird.

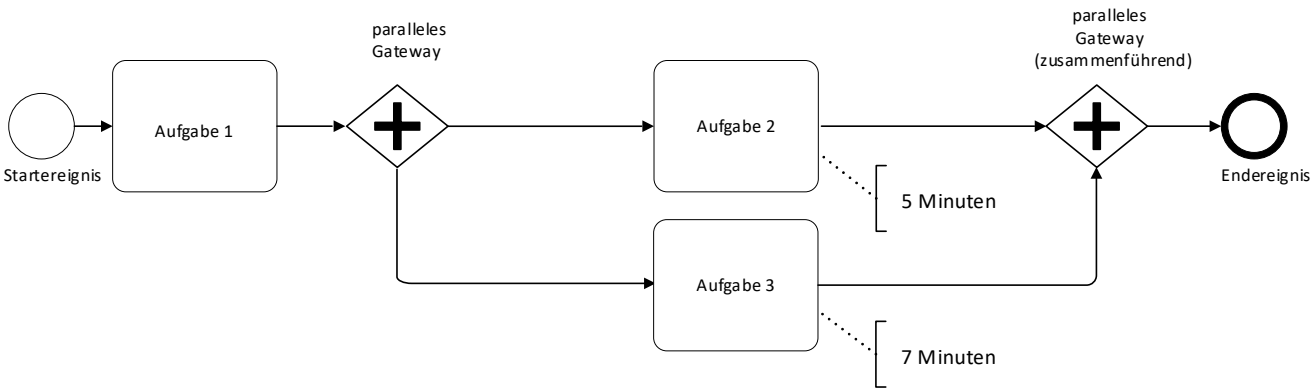

Abbildung 11: BPMN-Notation für ein paralleles Gateway

Lanes

Mit den bisherigen Notationselementen können Prozesse bereits modelliert werden. Mithilfe von Lanes lassen sich Verantwortlichkeiten für Prozesse hinzufügen. Dies können sowohl konkrete Personen als auch Abteilungen (z. B. Buchhaltung), allgemeine Rollen (z. B. Lieferant) oder ein IT-Systeme (z. B. ein CBR-System) sein.

Pools

Pools bilden den übergeordneten Rahmen für Lanes. Sie fassen beispielsweise die Lanes der konkreten Personen „Frau M." und „Herr B." zum Pool „Geschäftsführung" zusammen.

Nachrichtenflüsse

Sequenzflüsse dürfen die Poolgrenzen nicht überschreiten. Damit poolübergreifende Aktivitäten trotzdem koordiniert werden können, werden Nachrichtenflüsse eingesetzt. Nachrichtenflüsse können einer Aufgabe oder einem (Zwischen-)Ereignis entspringen oder in diese hineinlaufen.

Assoziationen

Über Assoziationen können Datenobjekte mit Aufgaben oder Sequenzflüssen verknüpft werden.

Datenobjekte

Informationen oder Dokumente, die während eines Prozesses erzeugt oder verwendet werden, werden als Datenobjekte[263] bezeichnet. Sie können sowohl in elektronischer (digitaler) als auch in physischer Form vorliegen.

Das Zusammenspiel von Lanes, Pools, Nachrichtenflüssen, Assoziationen und Datenobjekte wird in der Abbildung 12 auf der nächsten Seite exemplarisch verdeutlicht.

263) Vgl. FREUND/RÜCKER (2019), S. 100 ff.

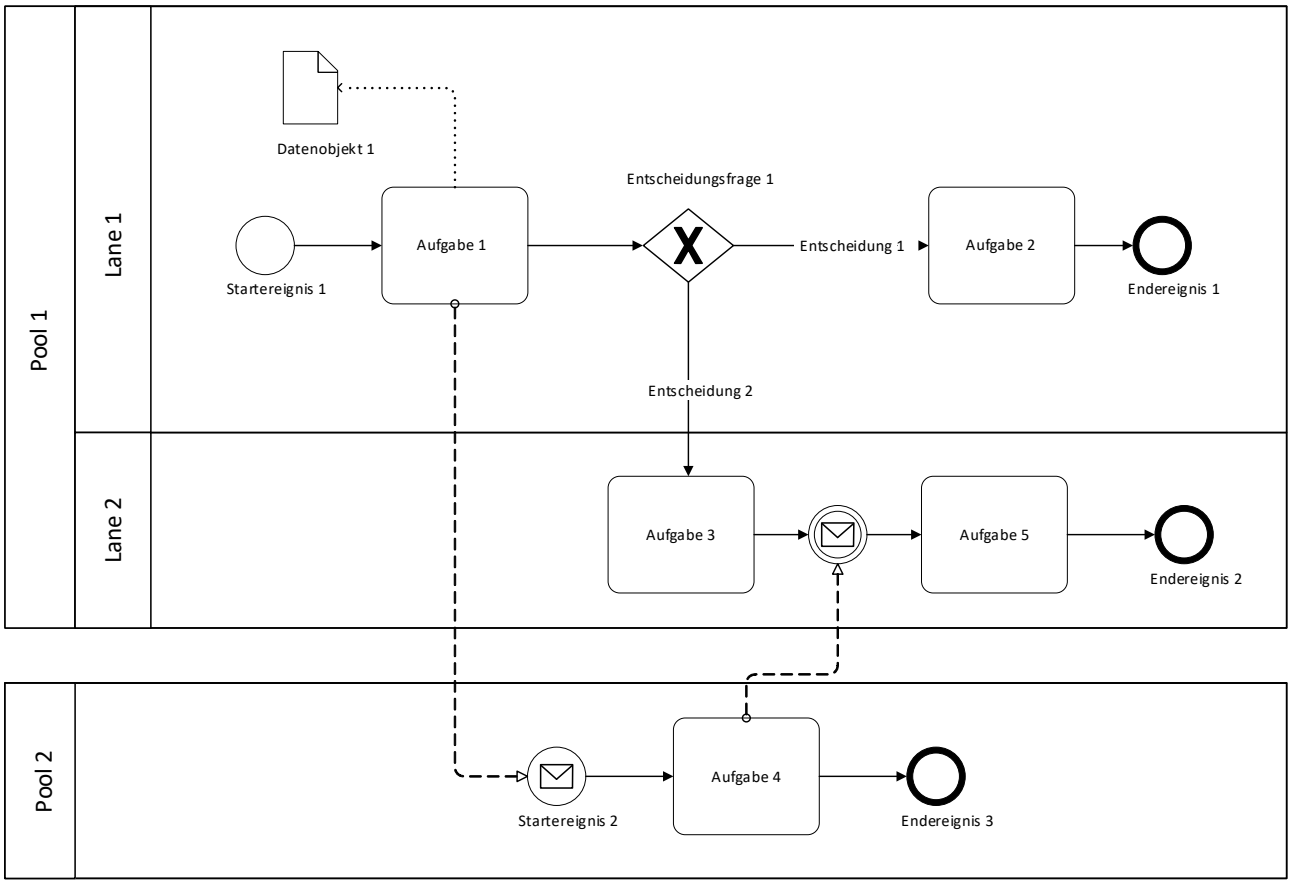

Abbildung 12: BPMN-Notationen für Pools und Lanes sowie Nachrichtenflüsse, Assoziationen und Datenobjekte

4.2 Präsentation des Vorgehensmodells

4.2.1 Überblick über das Vorgehensmodell

In diesem Kapitel wird ein benutzerfreundliches Vorgehensmodell für die Erstellung eines ontologiegestützten CBR-Systems vorgestellt, das auf die Bedürfnisse der betrieblichen Praxis hinsichtlich Begrifflichkeit und Übersichtlichkeit zugeschnitten ist. Das Vorgehensmodell wird mithilfe der semigrafischen Modellierungssprache BPMN visualisiert und an geeigneten Stellen durch Screenshots der verwendeten KI-Tools – Protégé als Ontologie-Editor und jCORA als ontologiegestütztes CBR-System – ergänzt.

Einen Überblick über das *vollständige Vorgehensmodell* bietet die Abbildung 13 auf der nächsten Seite. Wegen der Größe des Vorgehensmodells und der hieraus resultierenden Schwierigkeit, es in BPMN-Notation auf einer DIN-A4-Seite mit noch leicht lesbaren Details darzustellen, wird das Vorgehensmodell in den nächsten Kapiteln in drei Phasen untergliedert. Es handelt sich um die Phasen der Anforderungsspezifizierung an ein ontologiegestütztes CBR-System, der Erstellung der zugehörigen Ontologie mithilfe des Ontologie-Editors Protégé sowie des Anlegens und Vergleichens von Fällen im ontologiegestützten CBR-System jCORA. Diese drei Phasen sind aus der Perspektive der betrieblichen Praxis für jedes Projekt zu durchlaufen, in dem ein ontologiegestütztes CBR-System zur Wiederverwendung von Erfahrungswissen im betrieblichen Projektmanagement entwickelt und implementiert werden soll. Für die beiden ersten Phasen wird in den nachfolgenden Kapiteln jeweils ein separater, phasenspezifischer *Ausschnitt* aus dem vollständigen Vorgehensmodell präsentiert, der sich mittels der Modellierungssprache BPMN übersichtlich darstellen lässt. Für die dritte Phase werden vor allem Screenshots des ontologiegestützten CBR-Systems jCORA genutzt, um zu verdeutlichen, wie sich mit seiner Hilfe Fälle anlegen und miteinander vergleichen lassen.

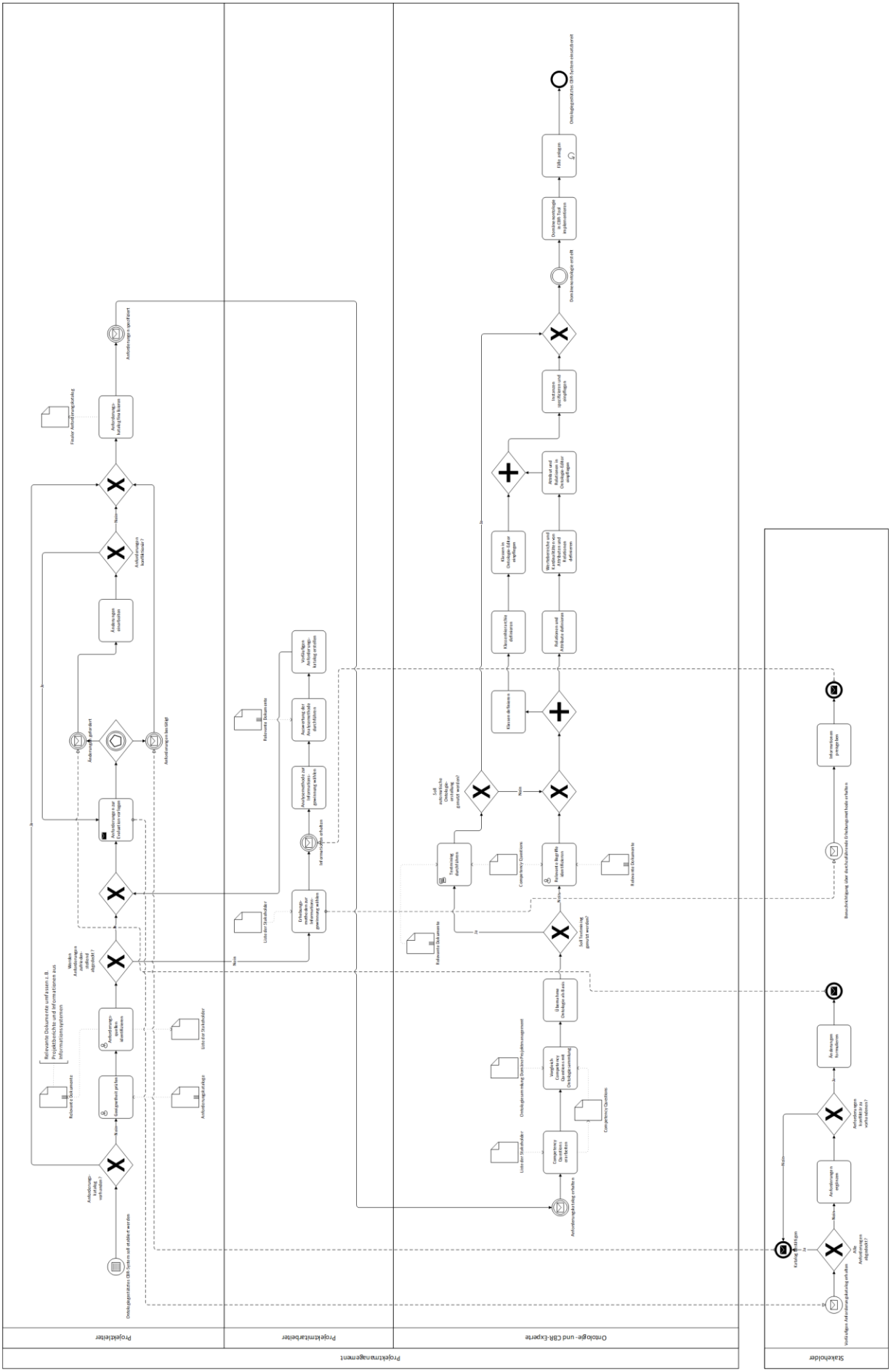

Abbildung 13: vollständiges Vorgehensmodell

4.2.2 Vorgehensmodell Phase 1: Anforderungsspezifizierung

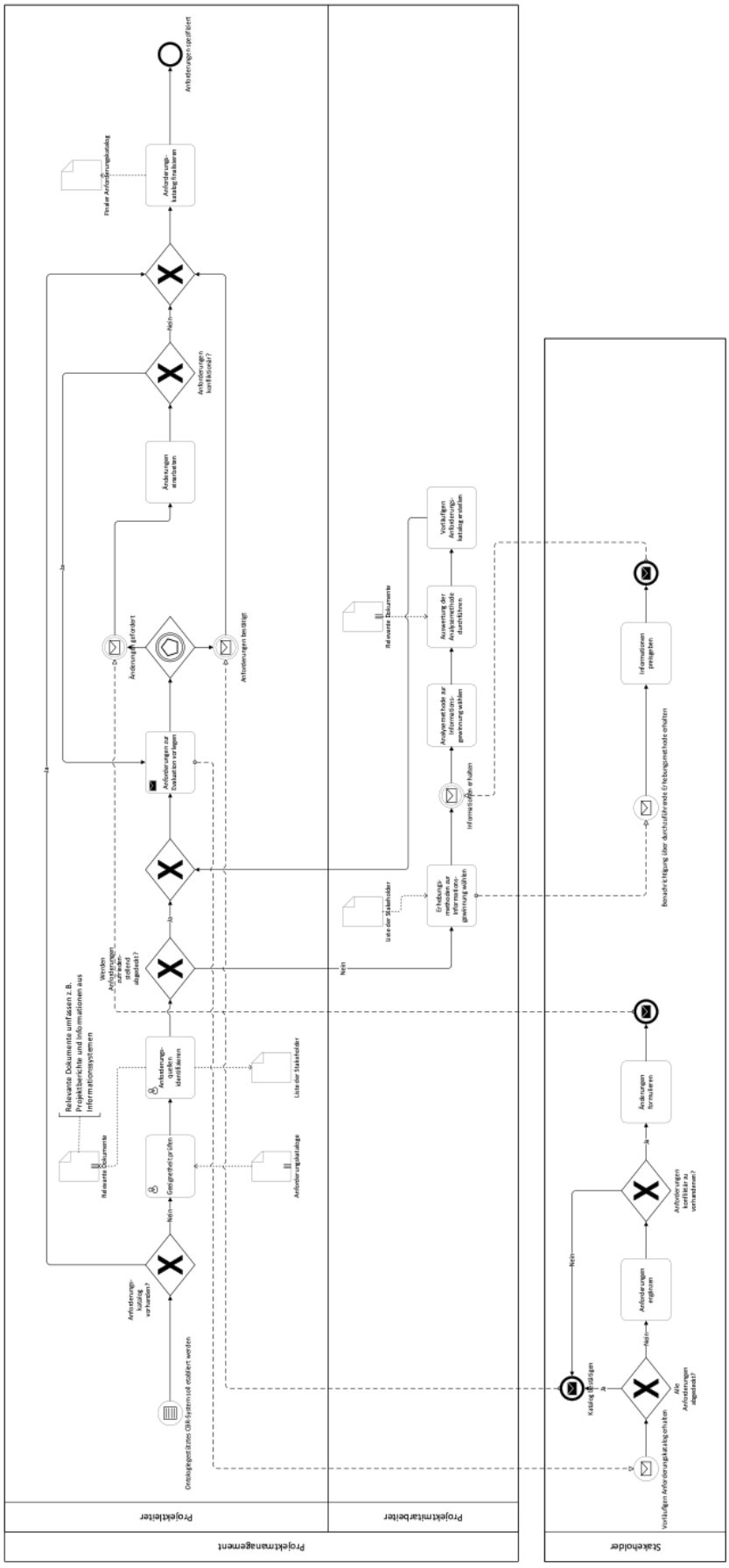

Abbildung 14: Ausschnitt aus dem Vorgehensmodell für die Anforderungsanalyse

Der Ausschnitt aus dem Vorgehensmodell, der sich in der voranstehenden Abbildung 14 auf die Phase der Anforderungsspezifizierung bezieht, gibt die Schritte wieder, die in Kapitel 3.1 hinsichtlich der Spezifizierung von Anforderungen an ein ontologiegestütztes CBR-System erläutert wurden.

4.2.3 Vorgehensmodell Phase 2: Erstellung einer Domänenontologie

In diesem Kapitel wird die Phase 2 näher betrachtet, die sich der Erstellung der Domänenontologie widmet. Der zugehörige Ausschnitt aus dem Vorgehensmodell wird in der Abbildung 15 auf der nächsten Seite präsentiert.

Bevor mit dem Ontologie-Editor Protégé gearbeitet werden kann, muss das (Erfahrungs-)Wissen, das mit den sprachlichen Ausdrucksmitteln einer Ontologie in einem Modell des jeweils relevanten Realitätsausschnitts repräsentiert und „nutzbar“ gemacht werden soll, in expliziter Form vorliegen und aufbereitet werden.

Die Festlegung einer Domäne ist der erste Schritt während der Erstellung einer Domänenontologie. Aus betriebswirtschaftlicher Perspektive entspricht die Domäne zumeist der Branche, in der das jeweils betroffene Unternehmen operiert. Unter Umständen kann die Domäne jedoch auch spezieller eingegrenzt werden, um den Aufwand bei der Erstellung zu reduzieren. Der im Folgenden aufgezeigte Weg resultiert in der Erstellung einer Projektmanagement-Domäne, die ihrerseits weiter spezialisiert werden kann. Für eine solche Domänenspezialisierung kommt beispielsweise eine Einschränkung auf Projekte in sicherheitskritischen Bereichen („Risikoprojekte“) oder auf Straßenbauprojekte in Betracht.

Nachdem die Domäne bestimmt wurde, muss der Umfang der geplanten Ontologie mithilfe von sogenannten „Competency Questions“ abgesteckt werden. Bei dieser Methode werden – wenn möglich – sämtliche Stakeholder in mehreren Zusammenkünften dahingehend befragt, welche kompetenzbezogenen Fragestellungen sich durch das ontologiegestützte CBR-System beantworten lassen sollen. Hierbei ist es erst einmal nicht relevant, einen Konsens zwischen den Teilnehmern der Brainstormings zu erhalten. Mithilfe der Brainstormingmethode können Ideen „reifen“, weil verschiedene Teilnehmer mitunter einen unterschiedlichen Grad an Erfahrungswissen aufweisen und die Visualisierung von Competency Questions anderer Teilnehmer auch zum eigenen Erkenntnisgewinn beitragen kann. Nachdem der Brainstormingprozess abgeschlossen ist, wird der Fragenkatalog finalisiert und dient als Anforderungskatalog für zukünftige Schritte.

Falls in einem Unternehmen noch nicht viele Erfahrungen mit dem Management von Erfahrungswissen, insbesondere mit dem Umgang mit Ontologien, vorliegen, kann ein Rückgriff auf bereits existierende Ontologien vorteilhaft sein, um einerseits den generellen Aufbau und die „Herangehensweise“ an Ontologien besser zu verstehen und andererseits von dem bereits vorhandenen Erfahrungswissen zu profitieren. Das in den Ontologien verwendete Vokabular kann durch eine bestehende Wiederverwendung von zumindest Teilen einer Ontologie dazu beitragen, dass sich ein gewisses „Standard-Vokabular“ etabliert. Hiervon profitiert das allgemeine begriffliche Verständnis zwischen verschiedenen Ontologie-Erstellern.

Zu prüfen ist, wie groß die Übereinstimmung zwischen den sprachlichen Ausdrucksmitteln bereits existierender Ontologien einerseits und den sprachlichen Ausdrucksmitteln für die zuvor ermittelten Competency Questions andererseits ist. Falls nur ein „Grundgerüst“ einer existierenden Ontologie als zweckmäßig für die eigene Ontologie angesehen wird, kann dies bereits den Aufwand der Ontologie-Erstellung wesentlich reduzieren. Existierende Ontologien können in der Fachliteratur, in Online-Bibliotheken oder mithilfe von dedizierten Suchmaschinen gefunden werden.

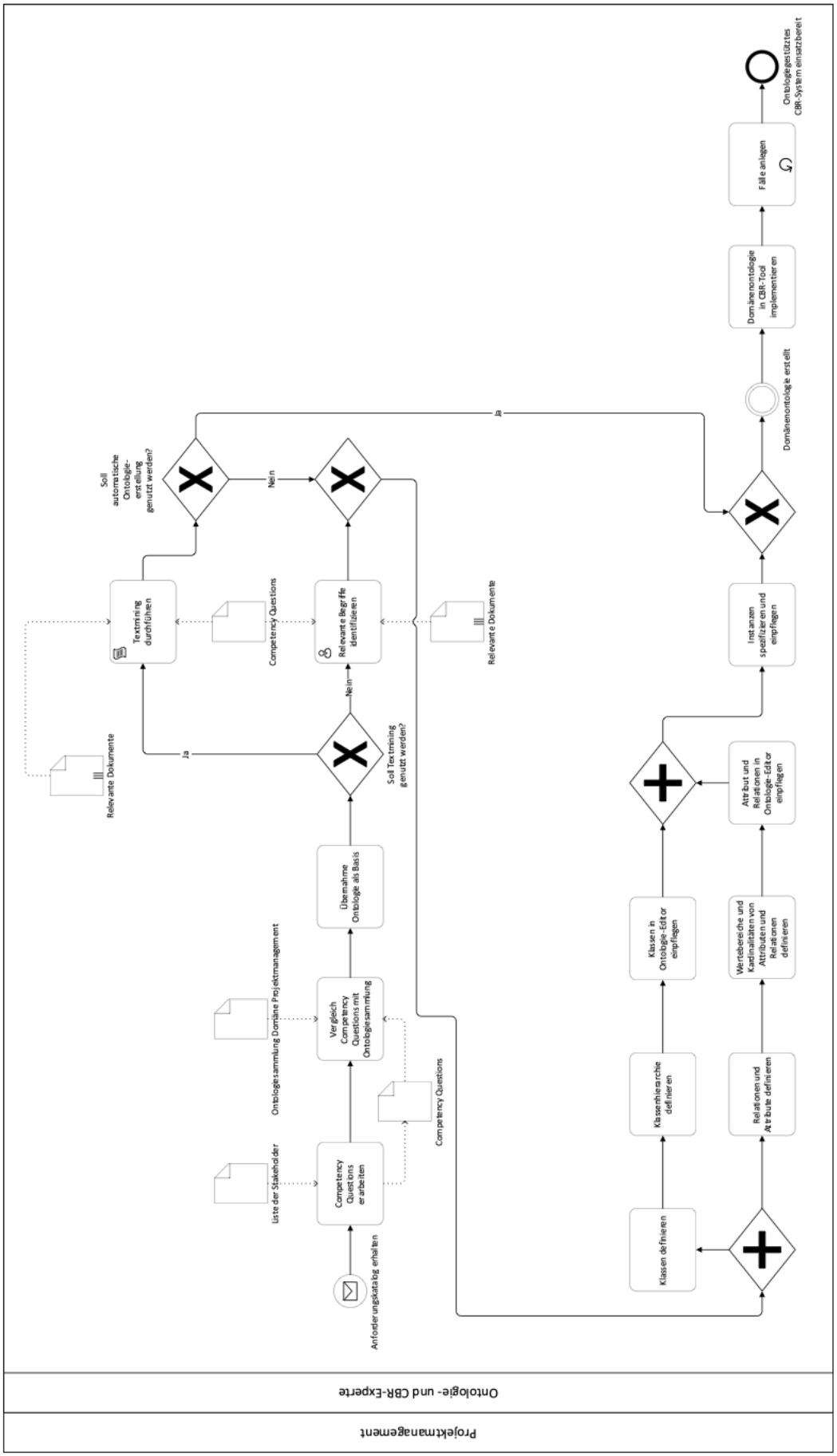

Abbildung 15: Ausschnitt aus dem Vorgehensmodell für die Erstellung einer Domänenontologie

Im nächsten Schritt folgt die Identifikation von Begriffen, die sich für die Domänenontologie als relevant erweisen. Ein weit verbreiteter Ansatz ist die Befragung von Experten der jeweiligen Domäne. Da sich in dieser Hinsicht eine große Überschneidung mit den zuvor formulierten Competency Questions ergibt, können diese Competency Questions als Grundlage für die Expertenbefragung genutzt werden. Weitere Quellen für relevante Begriffe sind Projektdatenbanken, Dokumente mit „Lessons learned“ sowie Project Reports sowohl in digitaler als auch in analoger Form. Die Wortarten können nach dem Sammeln der Begrifflichkeiten direkt verschiedenen Ontologiebestandteilen zugeordnet werden: Substantive entsprechen Klassen in Ontologien, Verben lassen sich zumeist in Relationen überführen und Adjektive stimmen in der Regel mit Attributen überein.

Für den nächsten Schritt empfiehlt sich bereits die Verwendung des Ontologie-Editors Protégé, dessen Installation nachfolgend kurz erläutert wird.

Nach dem kostenlosen Download der Protégé-Software[264] öffnet sich der Installations-Wizard nach einem Doppelklick auf protégé.exe:

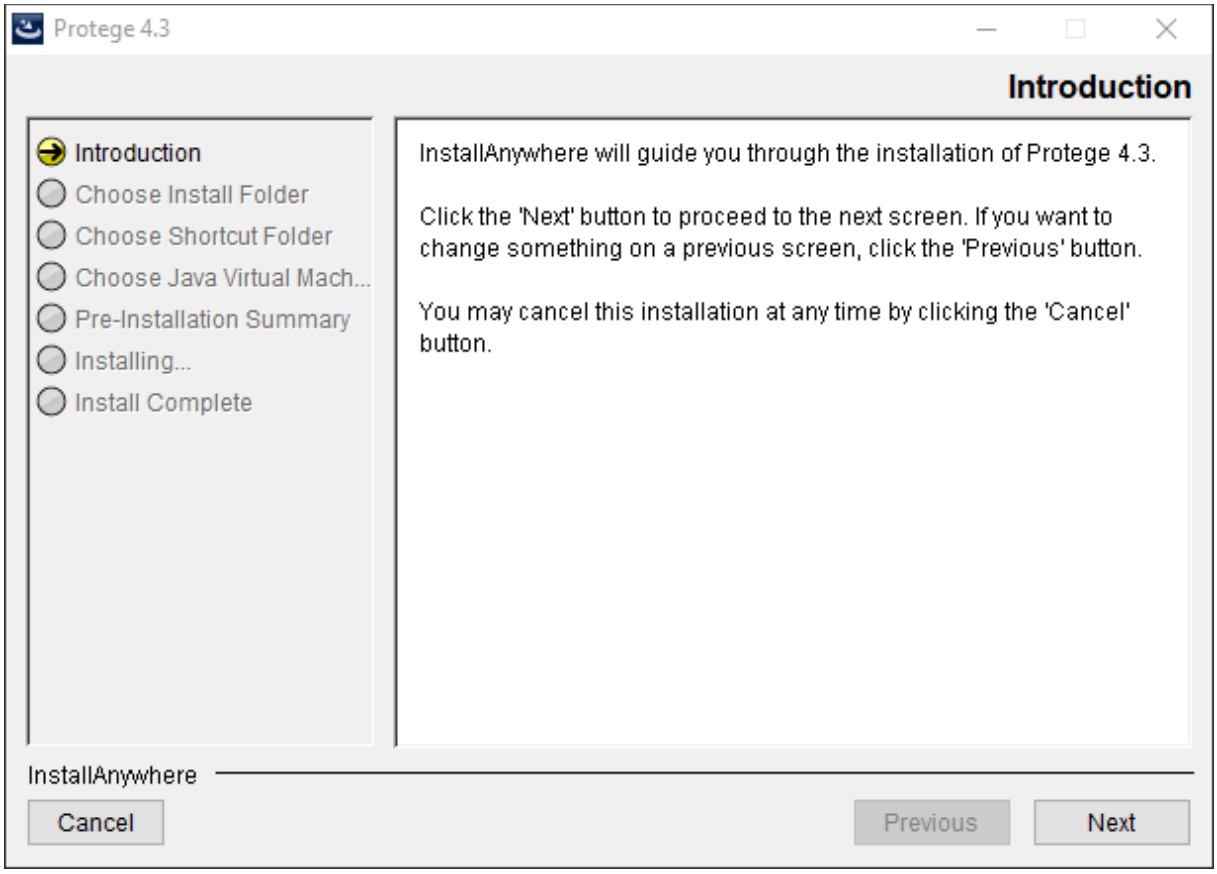

Abbildung 16: Installierung der Software Protégé

Nach der Wahl des Speicherorts auf dem Computer muss festgelegt werden, ob eine bereits installierte Java Virtual Machine (Java VM) verwendet werden soll oder ob sie vom Protégé-Wizard auf dem Computer neu installiert werden soll. Da Protégé auf die Programmiersprache Java zurückgreift, wird sie zum Betrieb dieser Software benötigt. Nach der erfolgreichen Installierung gelangt man zur Startoberfläche des Ontolgie-Editors Protégé; vgl. die nachfolgende Abbildung 17.

264) Die Software Protégé steht u. a. für das Betriebssystem Windows unter https://protege.stanford.edu zum kostenlosen Download bereit. Alternativ kann auch die Online-Version WebProtégé verwendet werden.

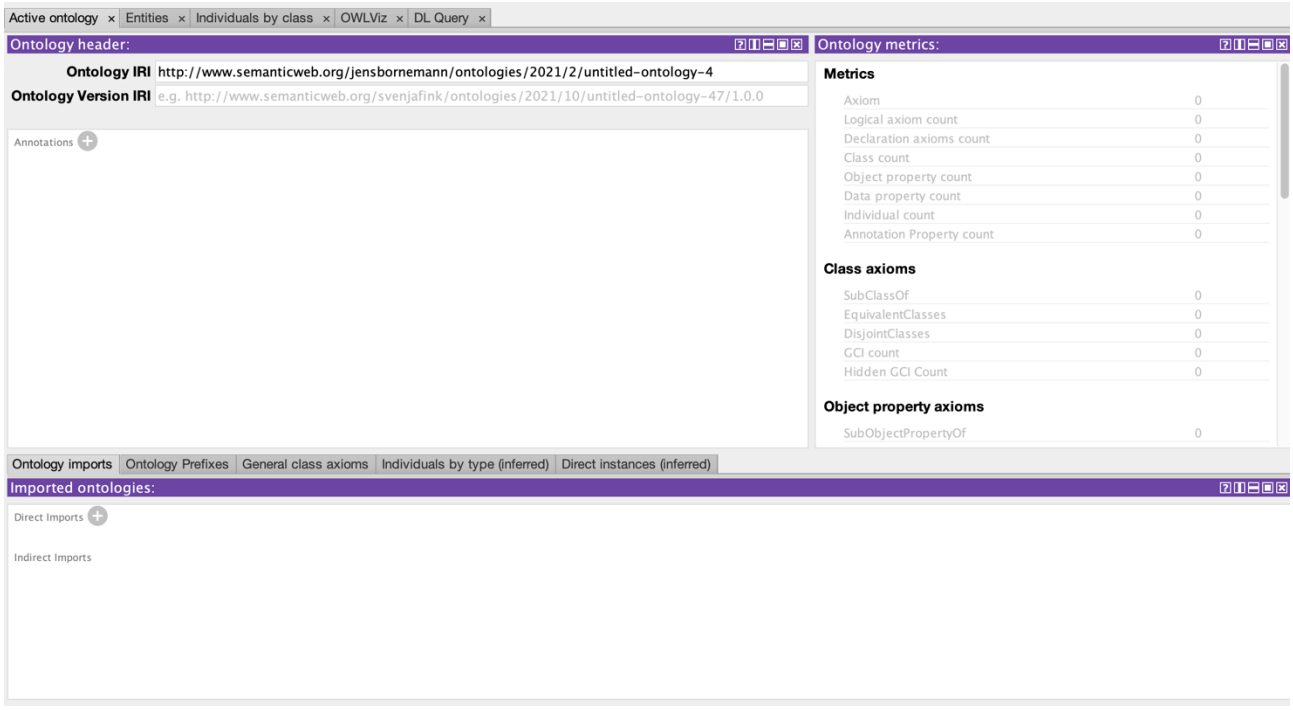

Abbildung 17: Startoberfläche des Ontologie-Editors Protégé

Nachfolgend erfolgt die Erstellung der Klassen und der Klassenhierarchie. Die zuvor ermittelten Substantive repräsentieren Klassen, die in Protégé hierarchisch unter der maximal abstrakten Klasse „Thing“ angeordnet werden. Über ein Kontextmenü lassen sich neue Subklassen einfügen (vgl. Abbildung 18), umbenennen und löschen.

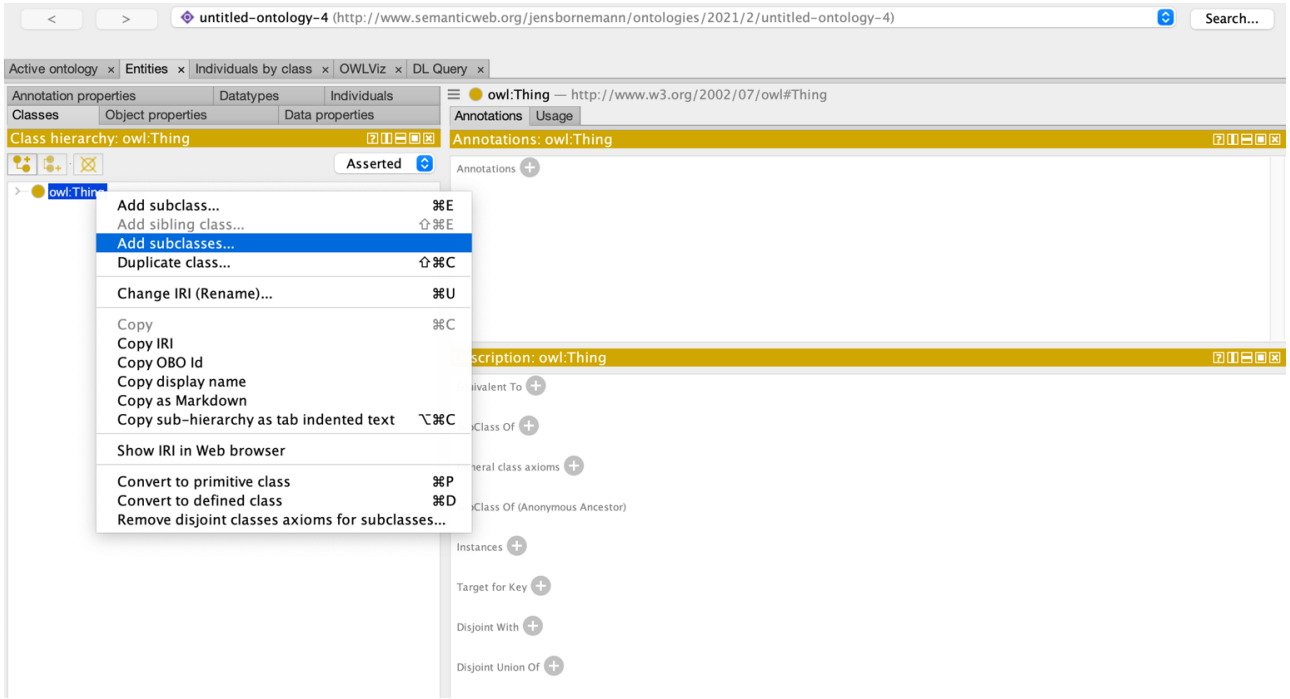

Abbildung 18: Erstellung einer neuen Subklasse

Auf diese Weise werden alle Klassen und Subklassen so lange erstellt, bis die Taxonomie aus der Sicht des Ontologie-Erstellers fertiggestellt ist. Die im Kontext dieses Projektberichts erstellten Klassen der Ontologie werden in der nachfolgenden Abbildung 19 wiedergegeben.

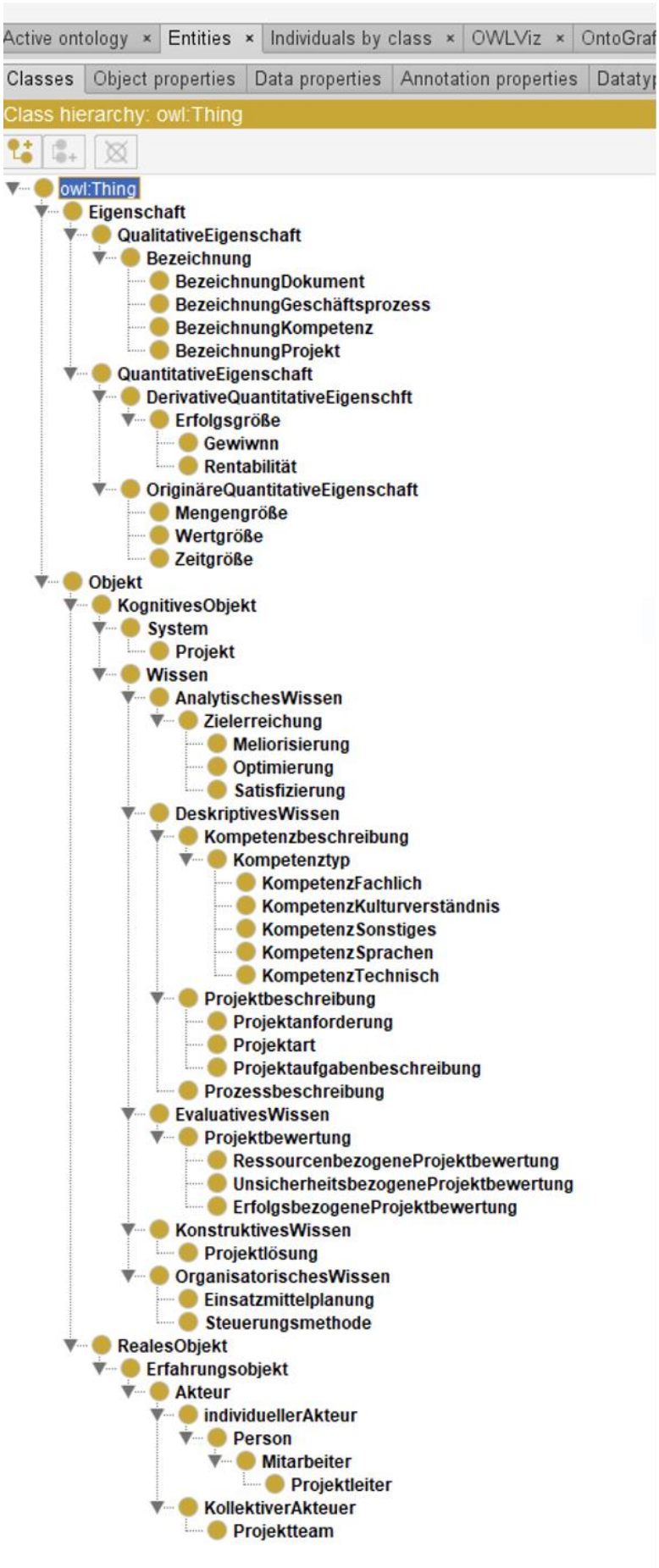

Abbildung 19: Klassen einer Projekt-Taxonomie in Protégé

Anschließend können die benötigten nicht-taxonomischen Relationen im Ontologie-Editor Protégé angelegt werden. Dies erfolgt über den Reiter *Object properties*. Über ein Kontextmenü lassen sich nicht-taxonomische Relationen anlegen, umbenennen und löschen; vgl. die nachfolgende Abbildung 20.

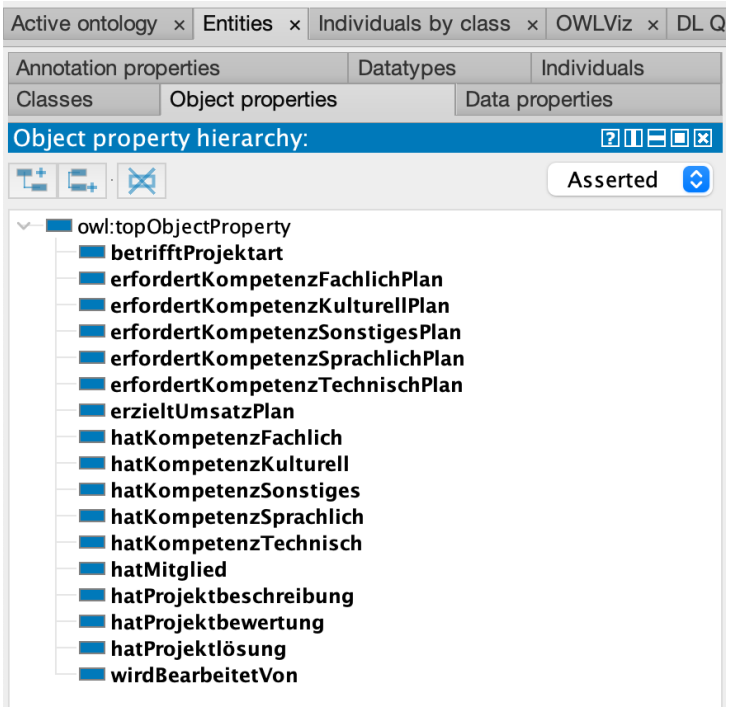

Abbildung 20: Überblick über angelegte nicht-taxonomische Relationen

Die einzelnen nicht-taxonomischen Relationen lassen sich näher spezifizieren. Dies erfolgt durch das Anklicken einer nicht-taxonomischen Relation. Dabei öffnet sich auf der rechten Seite das folgende Fenster; vgl. die Abbildung 21. Wie aus dieser ersichtlich, werden für eine nicht-taxonomische Relation deren Domain und Range durch das Anklicken des Plus-Zeichens hinzugefügt.[265]

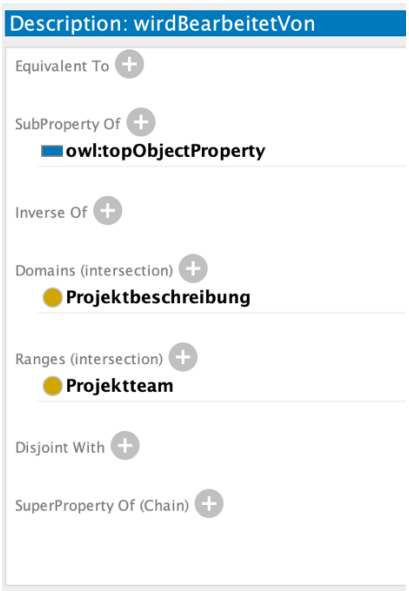

Abbildung 21: Festlegung der Domäne und Range einer nicht-taxonomischen Relation

265) In Abbildung 21 werden Domain und Range jeweils im Plural formuliert, weil in Protégé grundsätzlich die Möglichkeit besteht, mehrere Klassen als Domain bzw. Range einer Relation zu spezifizieren.

In einem weiteren Schritt können in Protégé Attribute erstellt werden. Dies erfolgt über den Reiter *Data properties*. Auch hier lassen sich über ein Kontextmenü Attribute anlegen, umbenennen und löschen. Die nachfolgende Abbildung 22 verdeutlicht, wie ein Attribut anglegt wird.

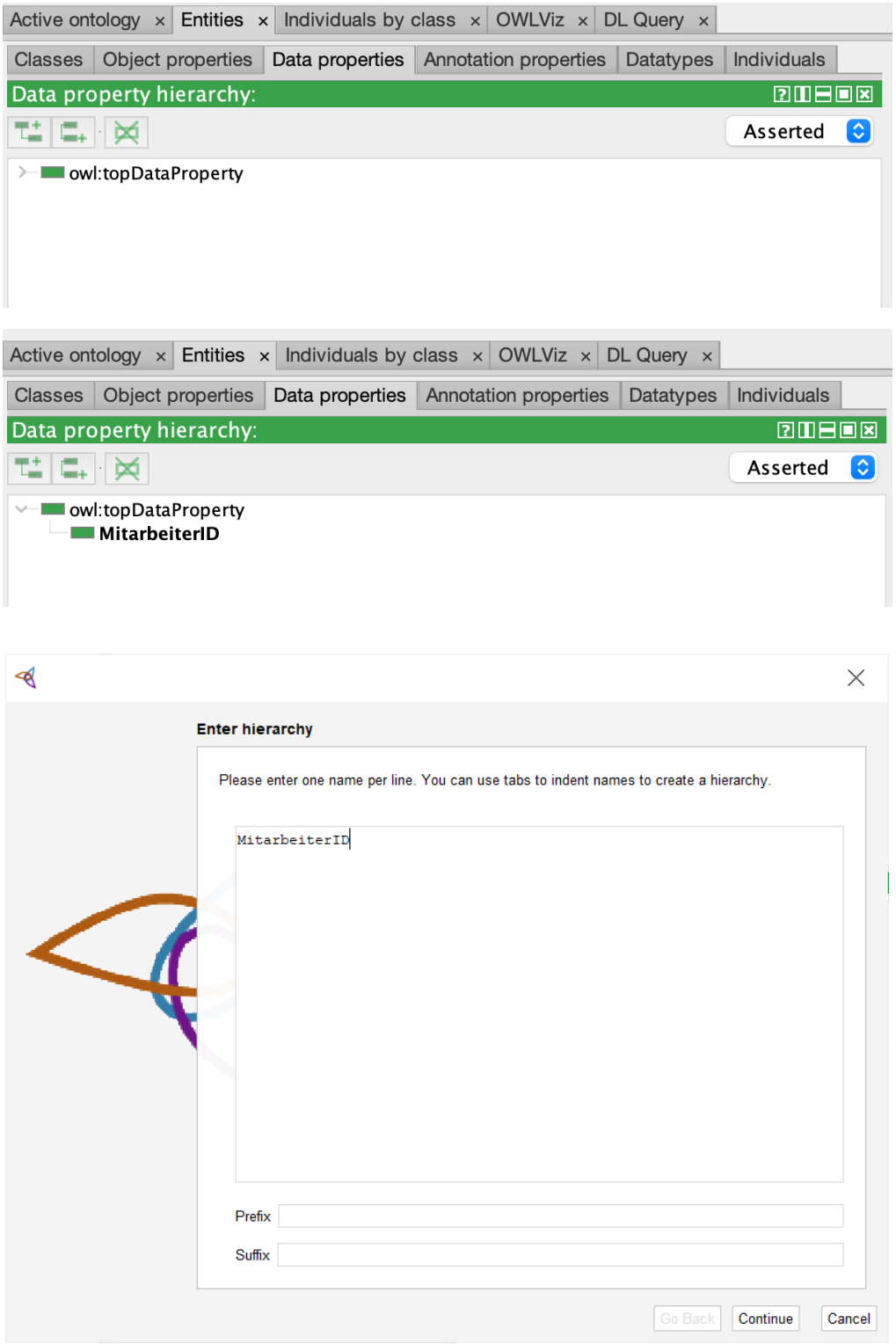

Abbildung 22: Anlegen eines Attributs

Wie aus der Abbildung 22 ersichtlich, werden Attribute analog zur Erstellung nicht-taxonomischer Relationen erstellt. Ebenso gilt es, für die angelegten Attribute ihre Domains und Ranges zu spezifizieren. Die nachfolgende Abbildung 23 verdeutlicht diesen Prozess.

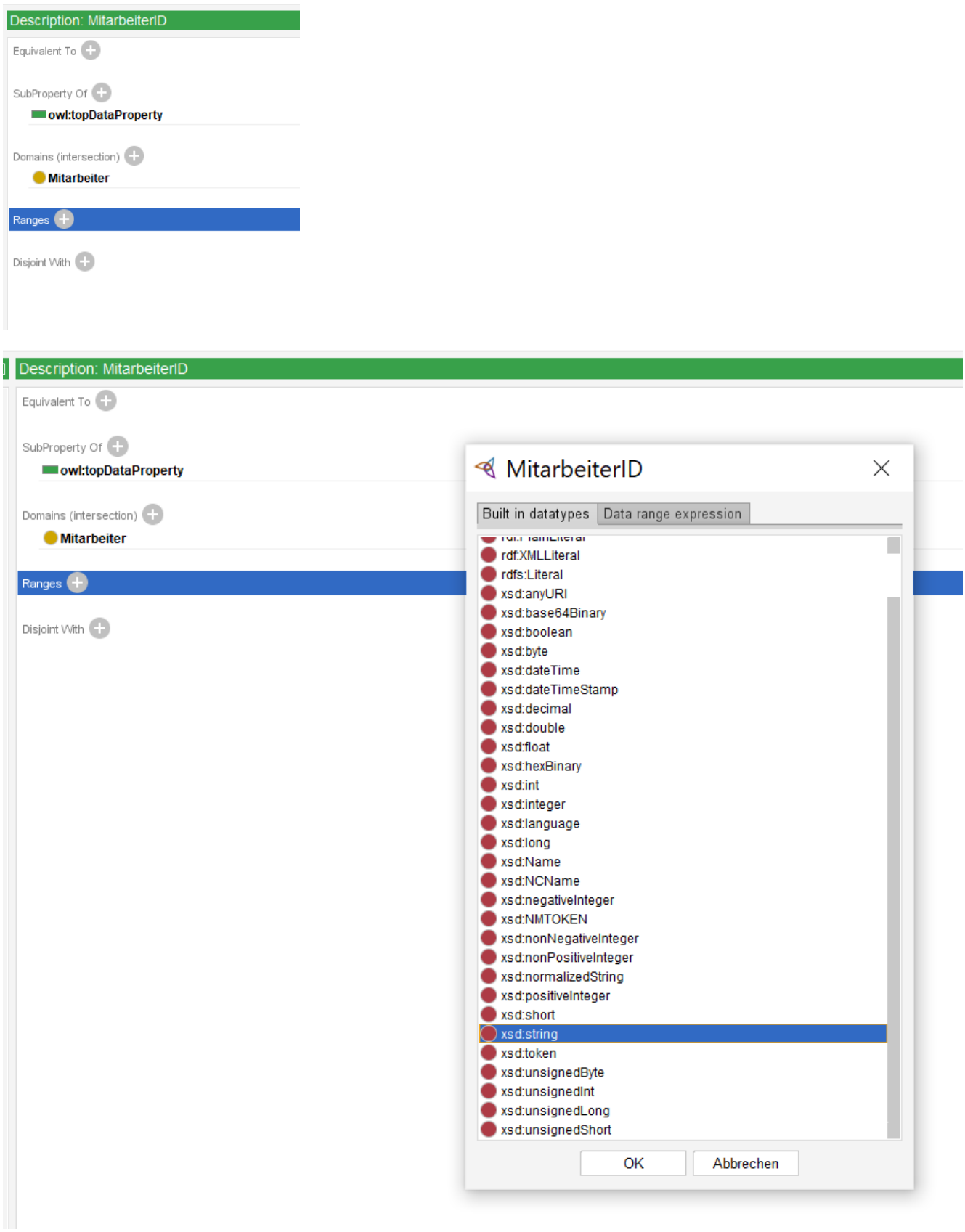

Abbildung 23: Spezifizierung von Domain und Range eines Attributs

Um die Domain eines Attributs zu spezifizieren, wird zunächst die gewünschte Klasse aus der Taxonomie hinzugefügt. Anschließend gilt es, als Range den gewünschten Datentyp hinzuzufügen. Die voranstehende Abbildung 23 zeigt einen Ausschnitt der in Protégé verfügbaren Datentypen. In dem

betrachteten Beispiel wird zur Repräsentation einer MitarbeiterID eine Zeichenkette („String“) benötigt. Sofern ein Attributwert – wie in diesem Beipiel – als Zeichenkette dargestellt werden soll, wird der Datentyp „xsd:string“ ausgewählt. Durch Drücken des Buttons „OK“ wird der gewählte Datentyp als Range hinzugefügt.

Nach Erstellung der Domänenontologie mithilfe des Ontologie-Editors Protégé kann diese vorläufige Ontologie in jCORA geladen werden.

4.2.4 Vorgehensmodell Phase 3: Anlegen und Vergleichen von Fällen im ontologiegestützten CBR-System jCORA

Nach dem Starten des ontologiegestützten CBR-Systems jCORA[266] öffnet sich der nachfolgend dargestellte Startbildschirm.

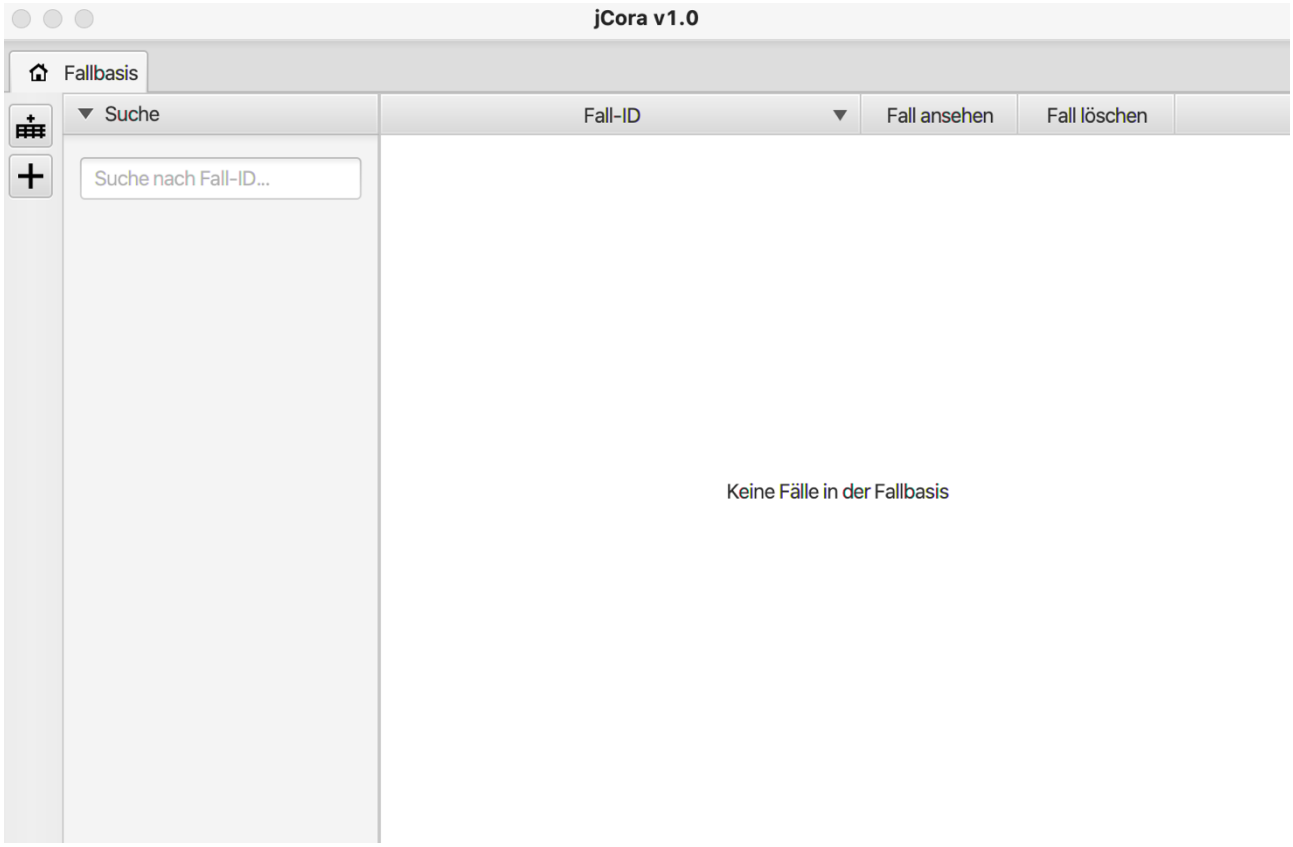

Abbildung 24: Startbildschirm des CBR-Systems jCORA

Nach dem erstmaligen Öffnen von jCORA gilt es zunächst, die zuvor erstellte Ontologie in jCORA hochzuladen. Dazu ist es notwendig, den Menüpunkt Einstellungen zu öffnen. Es öffnet sich das nachfolgende Fenster mit den drei Reitern „Sprach“, „Fallbasis“ und „Fallstruktur“. Der Reiter „Sprache“ öffnet sich zuerst.

266) Die Funktionen von jCORA sind in ZELEWSKI/SCHAGEN (2022), S. 31 ff., und BERGENRODT/KOWALSKI/ZELEWSKI (2015), S. 475 ff., genauer beschrieben.

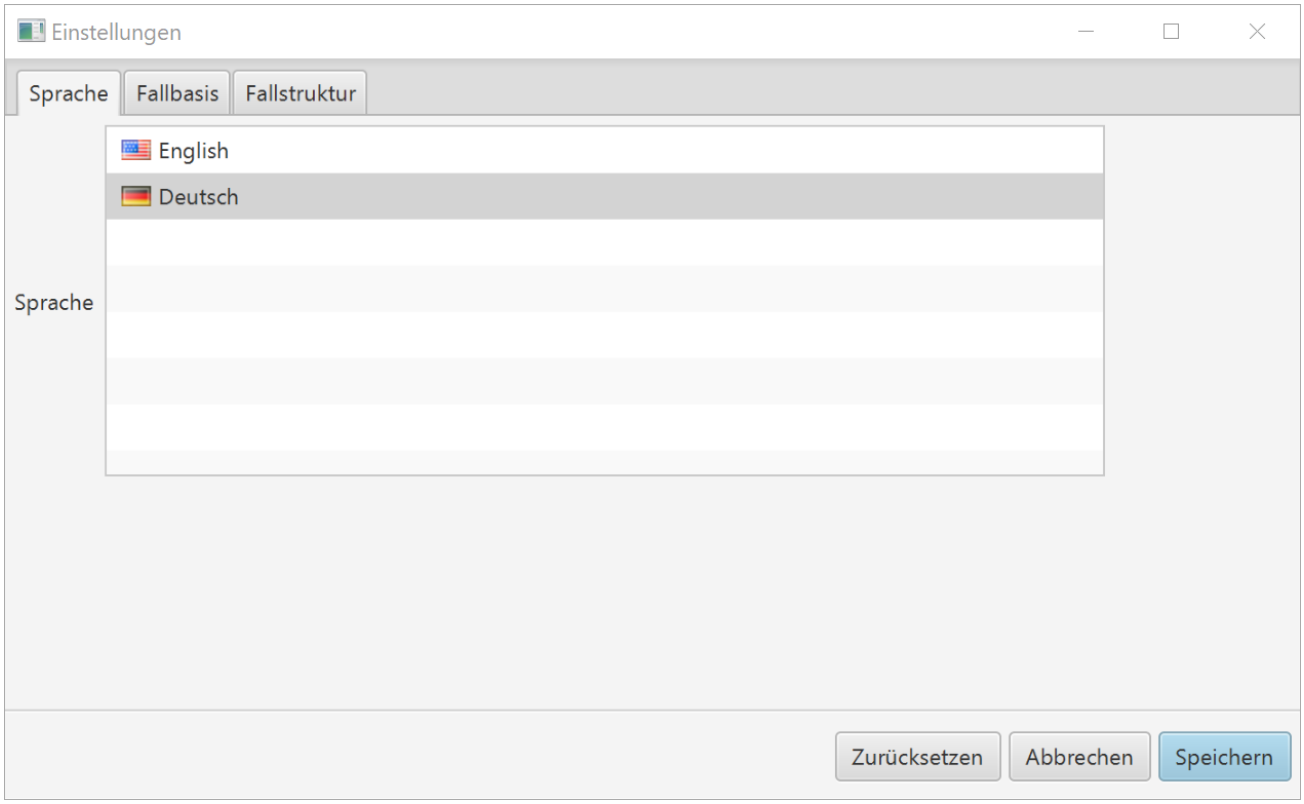

Abbildung 25: Auswahl der Sprache in jCORA

Über den ersten Reiter „Sprache“ kann die gewünschte Sprache ausgewählt werden. Es kann zwischen Deutsch und Englisch ausgewählt werden.

Über den zweiten Reiter „Fallbasis“ wird zunächst ein Ablageort für die Fallbasis gewählt. Dieser Ablageort kann ein beliebiger Ordner auf dem lokalen Computer des Benutzers sein. Anschließend erfolgt der eigentliche Upload der Domänenontologie. Dazu wird die zuvor in Protégé erstellte Domänenontologie vom lokalen Computer des Benutzers in Form einer OWL-Datei ausgewählt. Zudem wird der spezifische Ontologiename aus Protégé übernommen. Es muss beachtet werden, dass nach dem Eintragen des Namensraums eine Raute „#“ dem Namen hinzugefügt wird. Die nachfolgende Abbildung 26 verdeutlicht diesen zweiten Reiter.

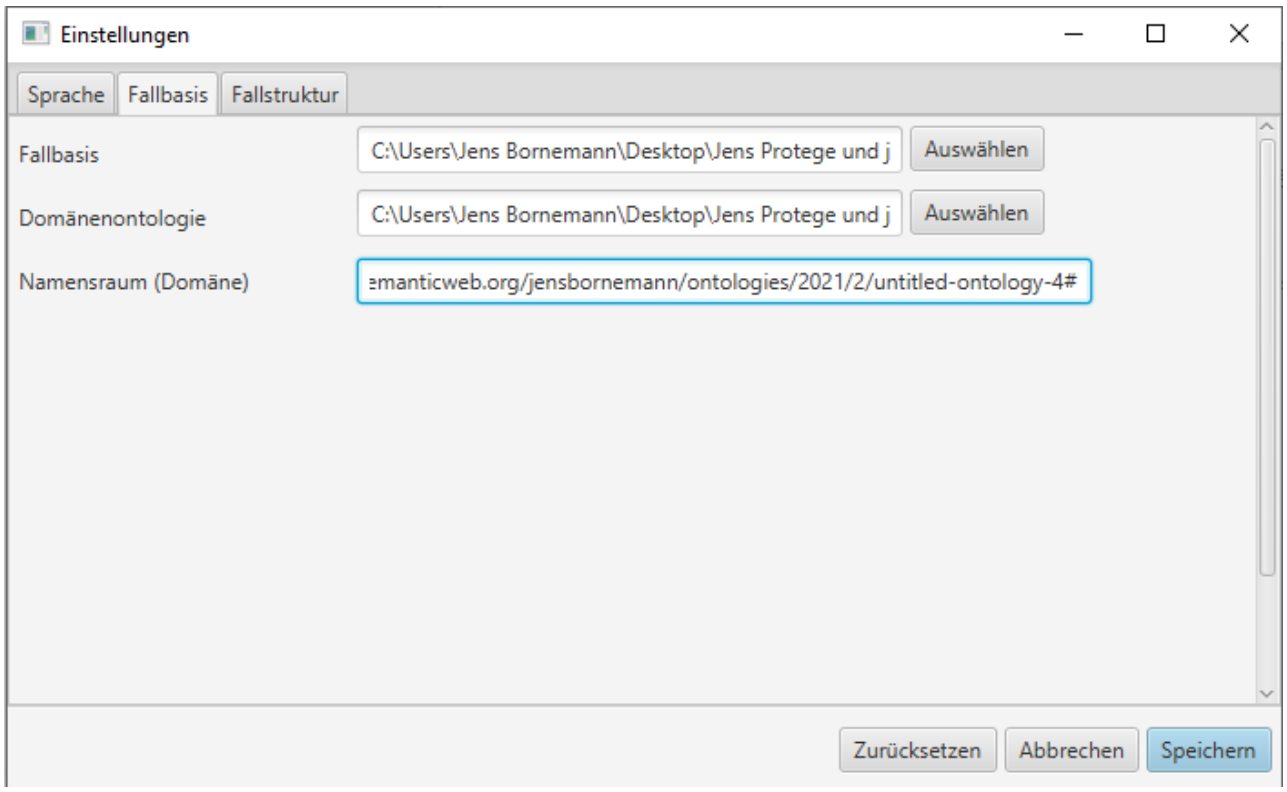

Abbildung 26: Fallbasis in jCORA eingeben

Im dritten Reiter „Fallstruktur“ wird spezifiziert, wie die für CBR-Systeme typische Dreiteilung in Fallbeschreibung, Falllösung und Fallbewertung in der zugrunde liegenden Domänenontologie auf Klassenebene konkretisiert wird. Für Domänenontologien mit Bezug zum Projektmanagement wird diese Dreiteilung zumeist durch die Klassen „Projektbeschreibung“, „Projektlösung“ und „Projektbewertung“ spezifiziert. Analog gilt es zudem, eine Klasse „Projekt“ zu spezifizieren. Neben den genannten Klassen müssen die in der Domänenontologie spezifizierten nicht-taxonomischen Relationen zwischen der Klasse „Projekt“ und den Klassen „Projektbeschreiibung“, „Projektlösung“, „Projektbewertung“ eingetragen werden. Dies sind in der Regel die folgenden nicht-taxonomischen Relationen: „hatProjektbeschreibung“, „hatProjektlösung“ und „hatProjektbewertung“.

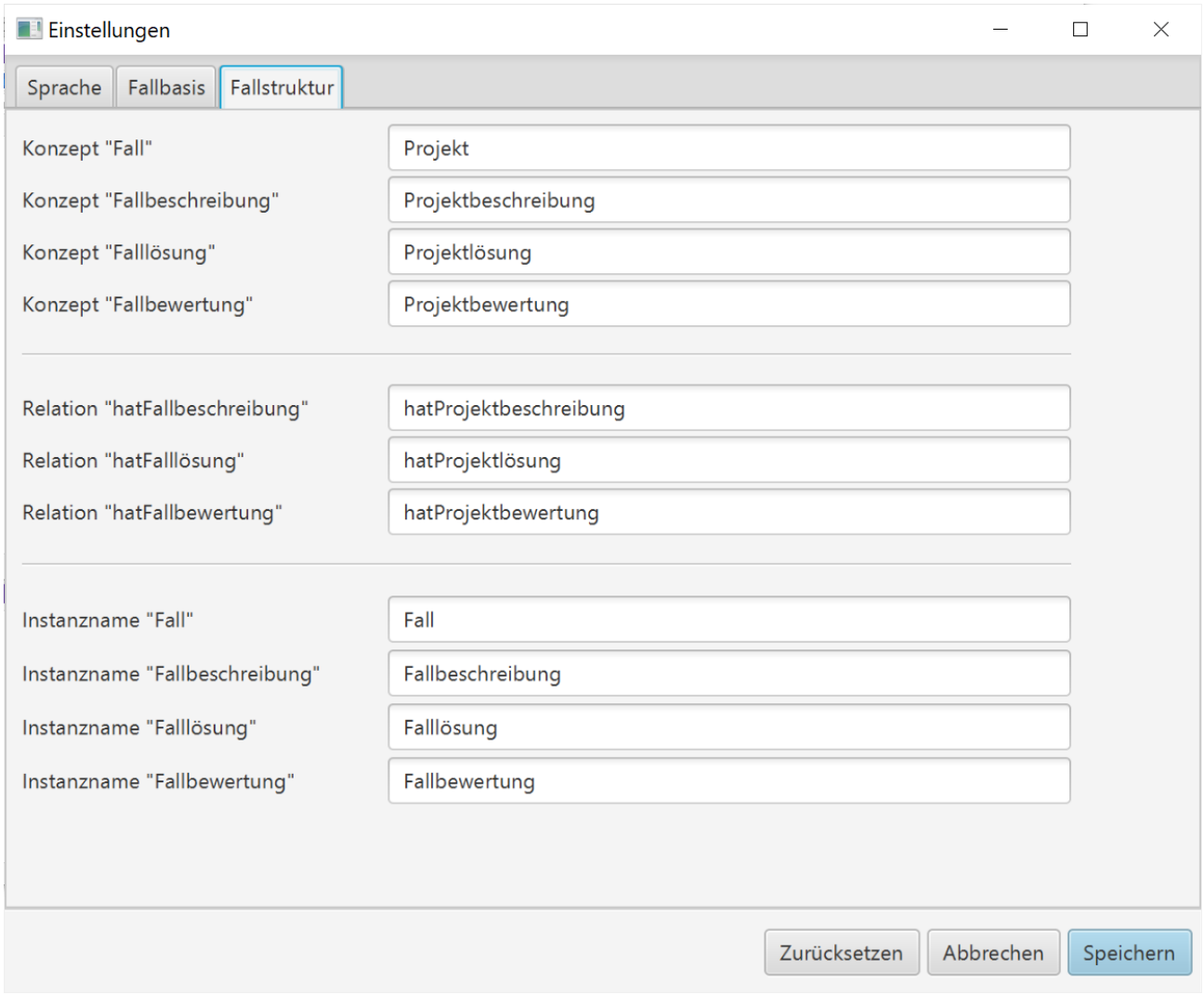

Abbildung 27: Einstellen der Fallstruktur in jCORA

Diese Fallstruktur bildet bei jeder neuen Fallerstellung die Basis eines Falls.

Nachdem die Domänenontologie in jCORA hochgeladen und die Fallstruktur gemäß der zugrunde liegenden Domänenontologie konkretisiert wurde, kann mit der Fallerstellung in jCORA begonnen werden.

Die nachfolgende Abbildung 28 zeigt den konstituiven Beginn eines Fallgraphen, bestehend aus der zuvor beschriebenen dreiteiligen Fallstruktur, in jCORA.

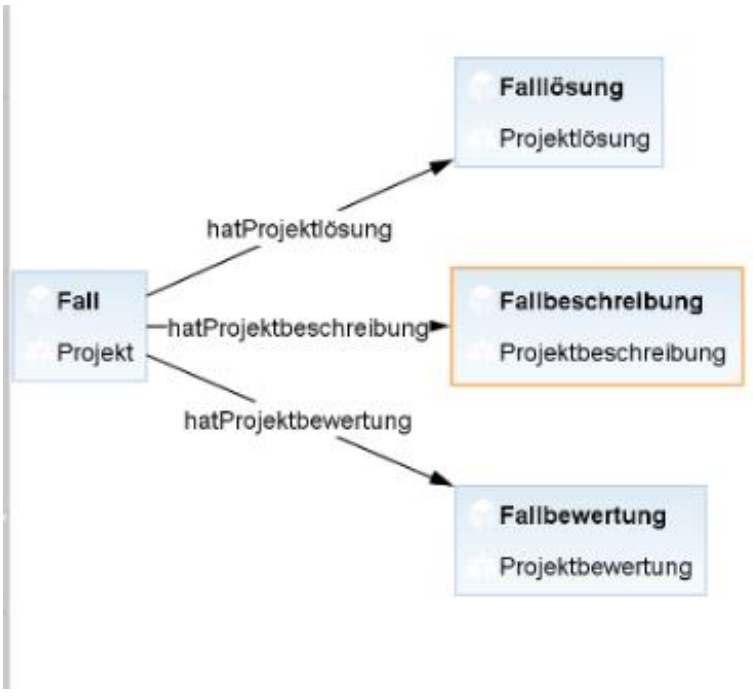

Abbildung 28: Dreiteilige Fallstruktur eines Fallgraphs in jCORA

Um den Fallgraphen zu erweitern, kann durch einen Rechts-Klick auf die Instanz „Fallbeschreibung“ nicht-taxonomische Relationen[267] hinzugefügt werden.[268] Es öffnet sich das nachfolgende Fenster.

267) Nicht-taxonomische Relationen werden in jCORA „nur“ als Relationen bezeichnet, da der Fallgraph lediglich um nicht-taxonomische Relationen erweitert wird.

268) Die Erweiterung des Fallgraphen kann grundsätzlich durch einen Rechtsklick auf eine beliebige Instanz erfolgen. Vorliegend wird dies am Beispiel der Instanz „Fallbeschreibung“ verdeutlicht, da diese Instanz der „natürliche“ Ausgangspunkt einer Fallgrapherweiterung ist.

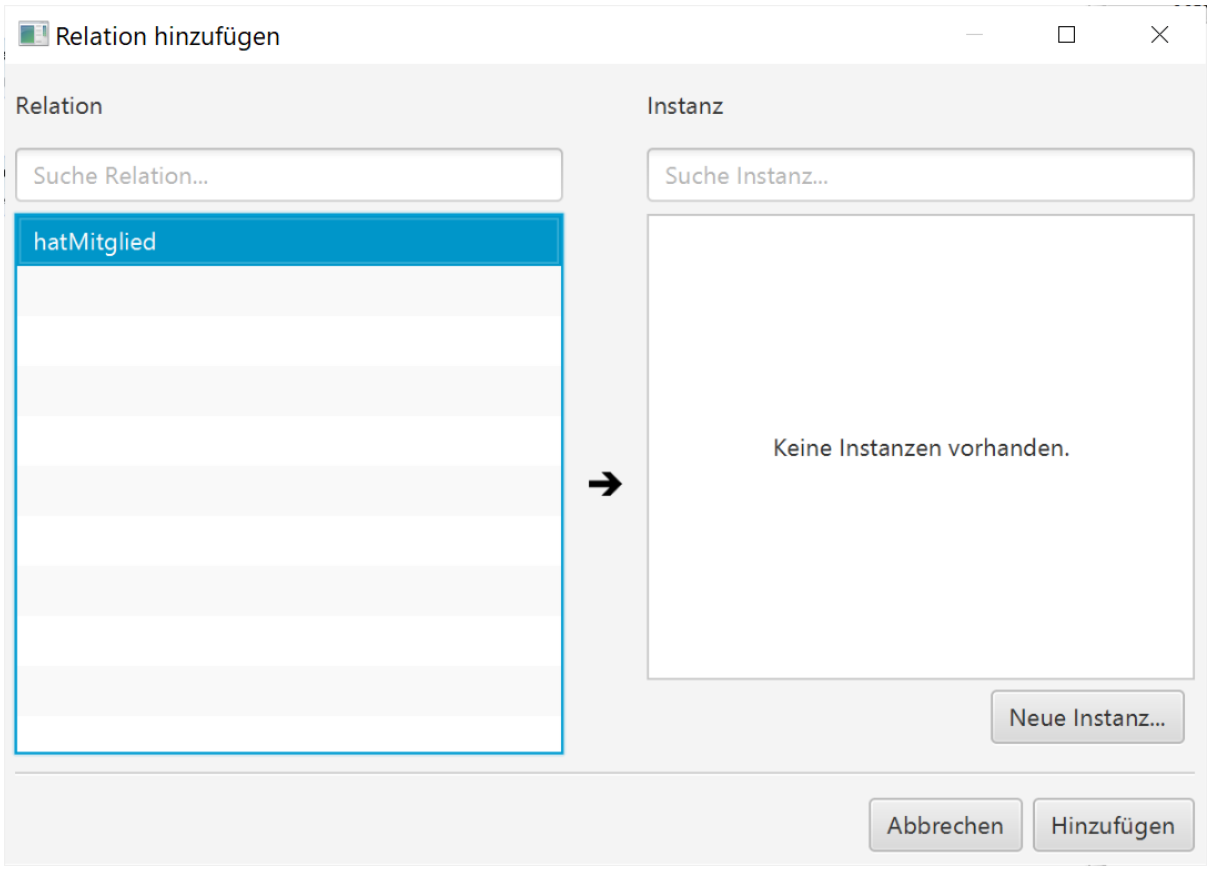

Abbildung 29: Relationen hinzufügen in jCORA

Das in Abbildung 29 zu sehende Fenster ist wie folgt zu interpretieren: Auf der linken Seite befinden sich alle verfügbaren nicht-taxonomischen Relationen. Die Menge der verfügbaren nicht-taxonomischen Relationen ist instanzabhängig. Für die im Einzelfall jeweils ausgewählte Instanz stehen alle nicht-taxonomischen Relationen zur Verfügung, die in ihrer Domain jene Klasse besitzen, zu der die im Einzelfall ausgewählte Instanz gehört.

Neue Instanzen können durch das Anklicken der Schaltfläche „Neue Instanz“ nach Belieben hinzugefügt werden. Hierfür werden ein eindeutiger Name – in diesem Fall der Mitarbeitername – vergeben und eine Klassenzuordnung ausgewählt. Die nachfolgende Abbildung 30 verdeutlicht diese Aktivitäten.

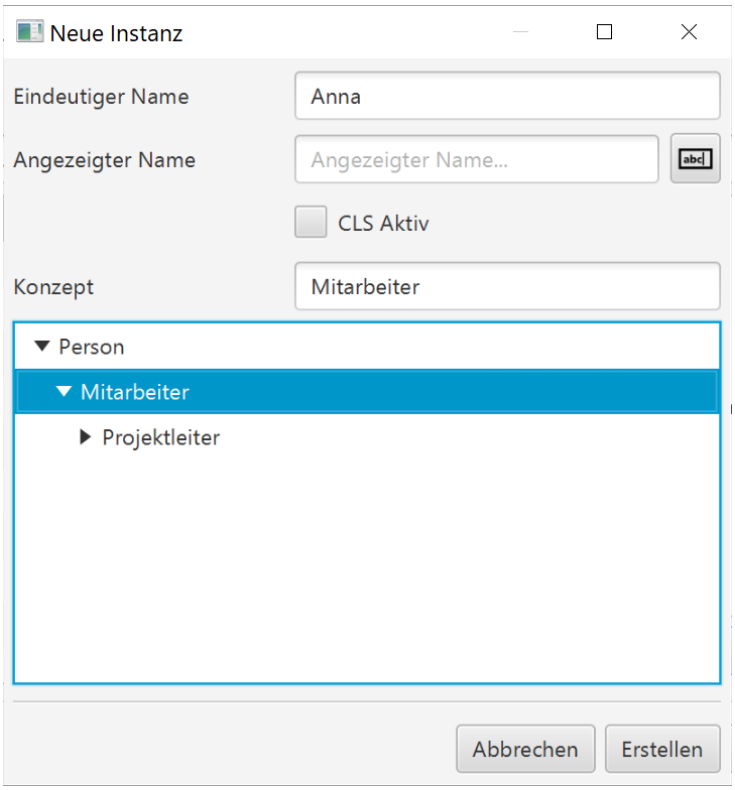

Abbildung 30: Neue Instanz in jCORA anlegen – Teil 1

Beim Anlegen einer Instanz ist zu berücksichtigen, dass die Instanzen nur zu jenen Klassen gehören dürfen, die in der zugrunde liegenden Domänenontologie die Range der jeweils betrachteten nicht-taxonomischen Relation bilden.

Nach dem Anlegen einer Instanz erscheint sie auf der rechten Seite des in Abbildung 31 dargestellten Fensters. Sie kann durch Anklicken der Schaltfläche „Hinzufügen“ zur Fallbasis hinzugefügt werden.

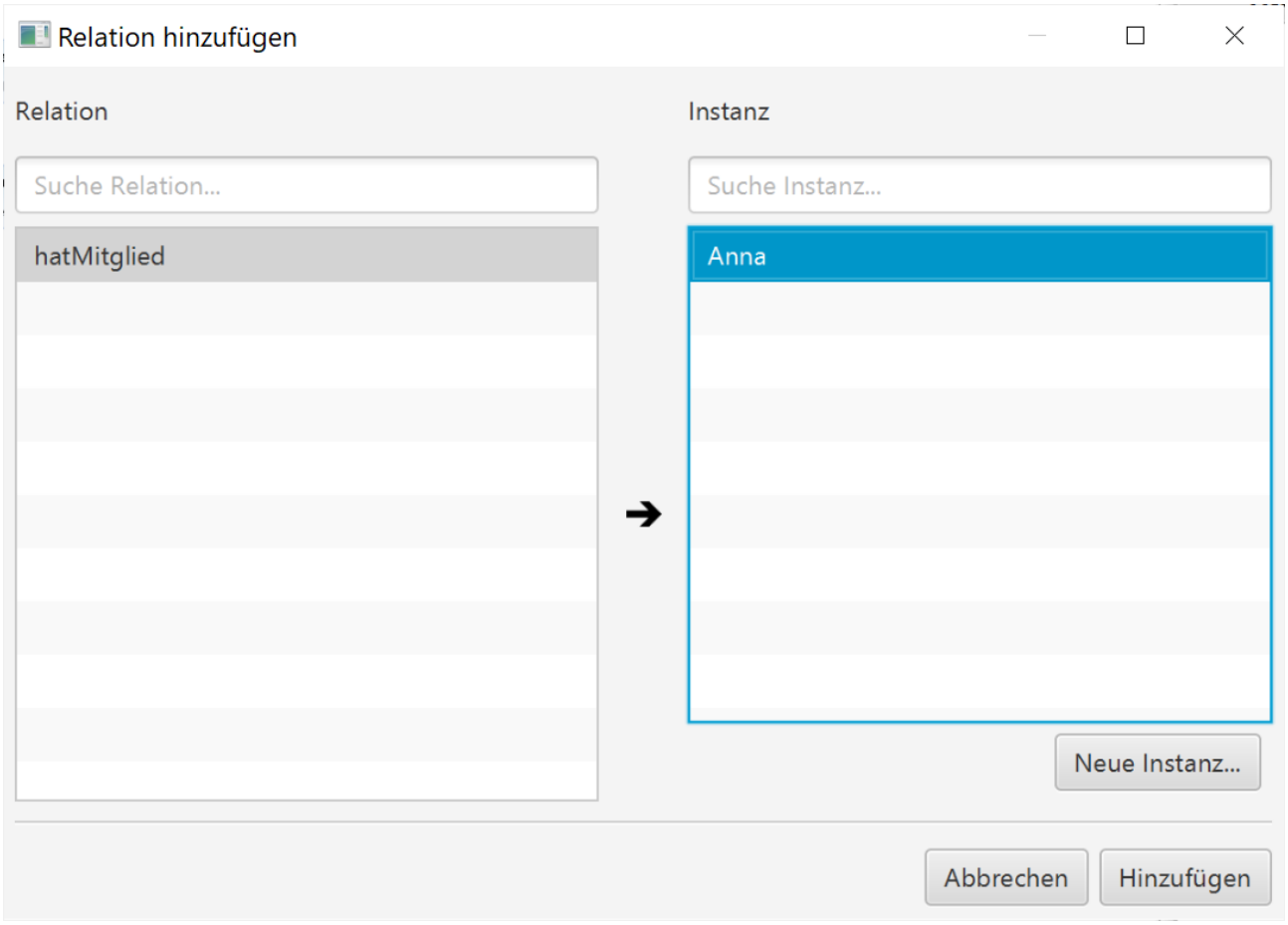

Abbildung 31: Neue Instanz in jCORA anlegen – Teil 2

Neben dem Erstellen fallspezifischer Instanzen als konkrete Relationswerte erlaubt jCORA auch das Hinzufügen von Attributwerten. Diese können nach Anklicken einer Instanz auf der rechten Seite über das Plus-Symbol hinzugefügt werden; siehe die Abbildung 32 auf der nächsten Seite.

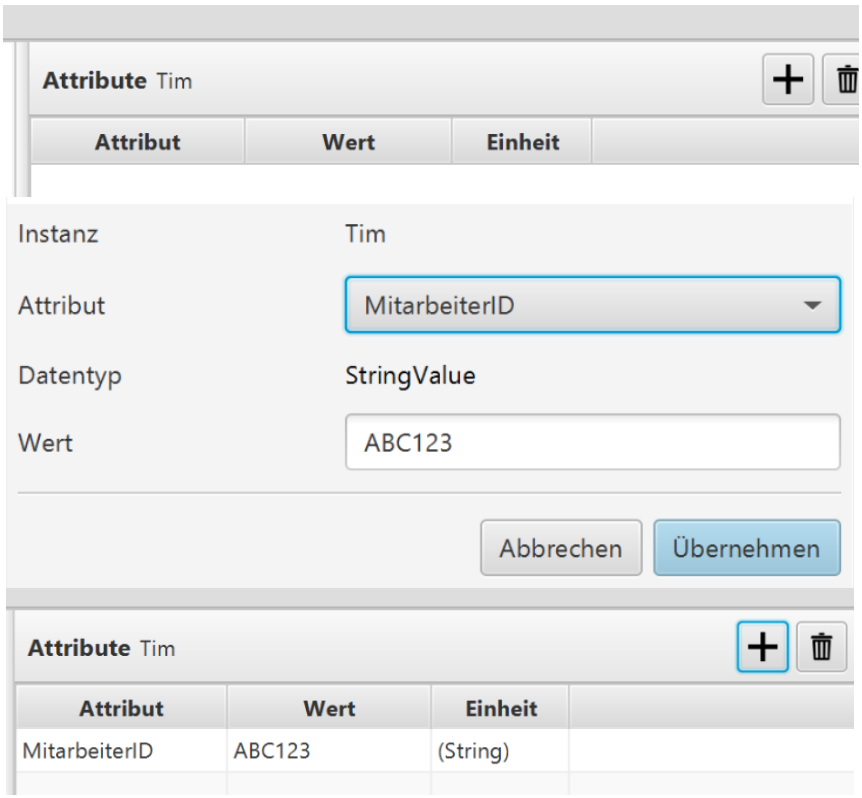

Abbildung 32: Hinzufügen von Attributen in jCORA

Die hinzugefügten Klassen, Instanzen, Relationen und Attribute werden innerhalb eines Fallgraphen in jCORA graphisch dargestellt. Die nachfolgenden Abbildungen 33 und 34 verdeutlichen zwei vollständige[269] Fallgraphen in jCORA.

269) Die Vollständigkeit eines Fallgraphens richtet sich nach dem subjektiven Empfinden des Erstellers und kann von Fall zu Fall unterschiedlich angesehen werden.

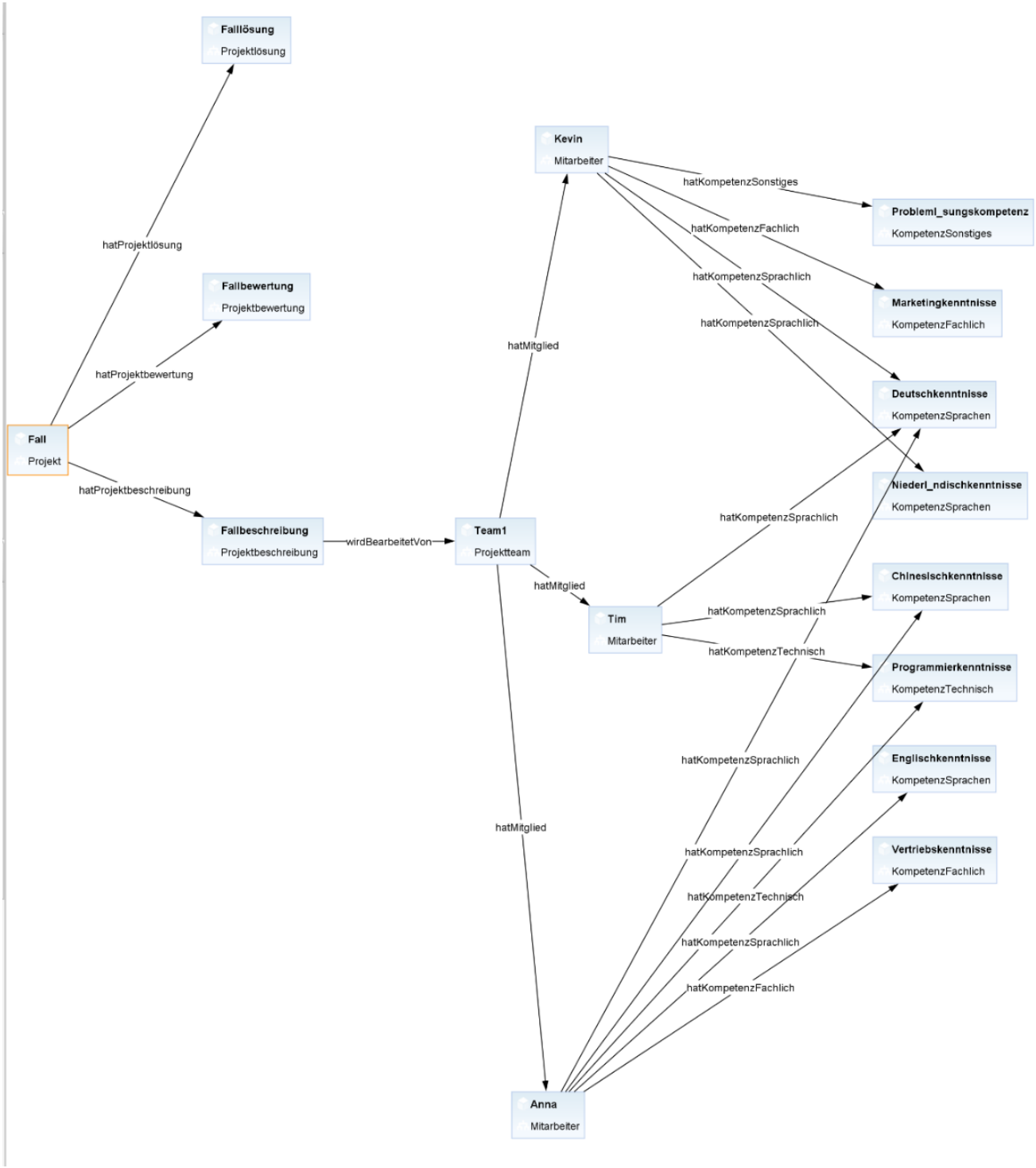

Abbildung 33: Fallgraph jCORA Fall 1

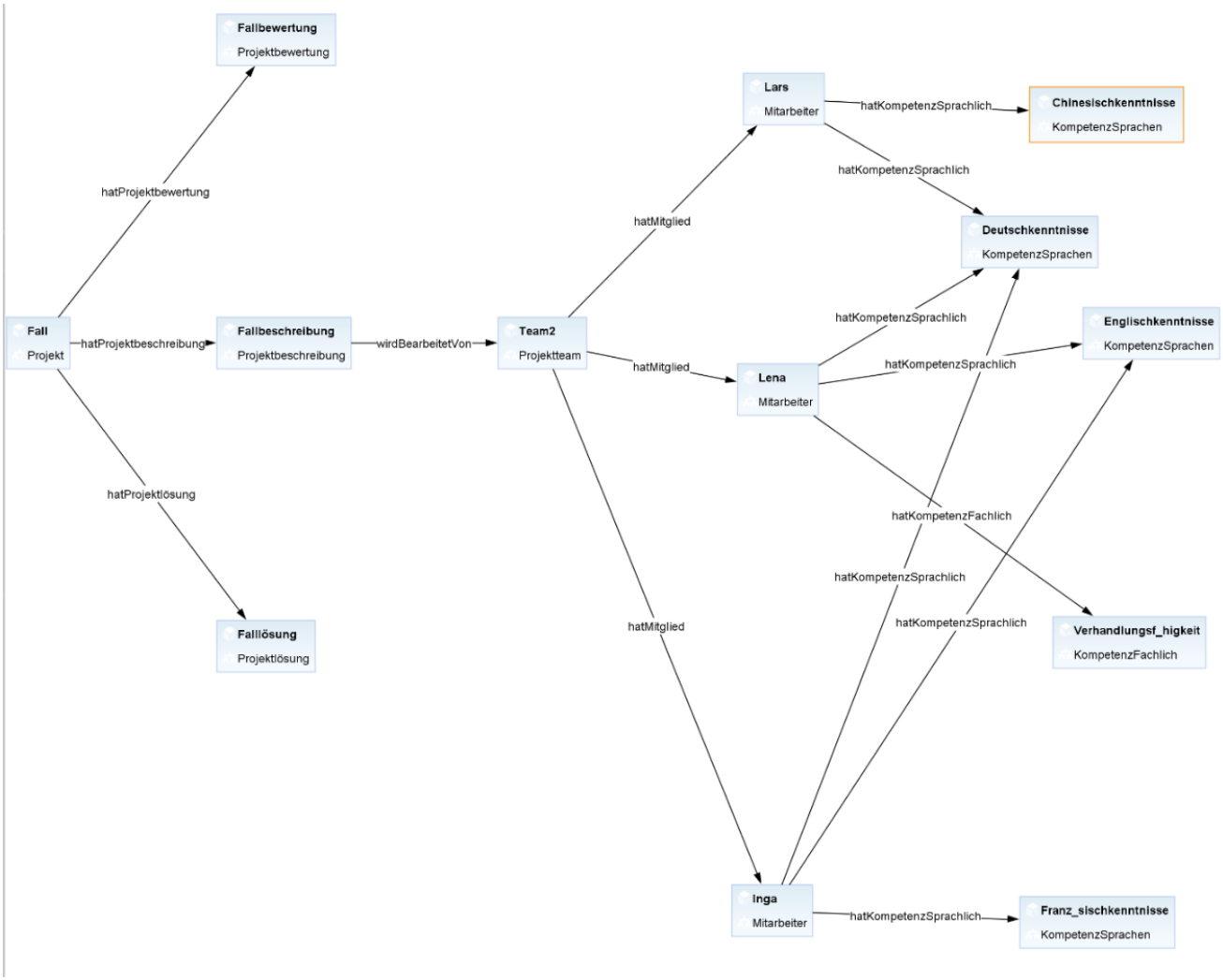

Abbildung 34: Fallgraph jCORA Fall 2

Aus einem angelegten Fall heraus kann über das Kontextmenü durch Klick auf das Symbol eine Anfrage an das CBR-System jCORA (kurz: CBR-Anfrage) gestartet werden. In diesem Projektbericht wird nur der „Standardfall“ betrachtet, für den ein ontologiegestütztes CBR-System im Bereich des betrieblichen Projektmanagements maßgeblich vorgesehen ist. Es handelt sich um die Anfrage an das CBR-System jCORA, welche *ähnlichsten alten Fälle* (Projekte) in seiner Falldatenbank (Projektwissensbank) in Bezug auf einen jeweils betrachteten, neuen Fall (neues Projekt) vorhanden sind, deren Erfahrungswissen, das in den Falllösungen und Fallbewertungen (Projektlösungen bzw. Projektbewertungen) der ähnlichsten alten Fälle gespeichert ist, sich zur Bearbeitung des neuen Falls wiederverwenden lässt. Die Initiierung einer solchen CBR-Anfrage wird in der nachfolgenden Abbildung 35 dargestellt.

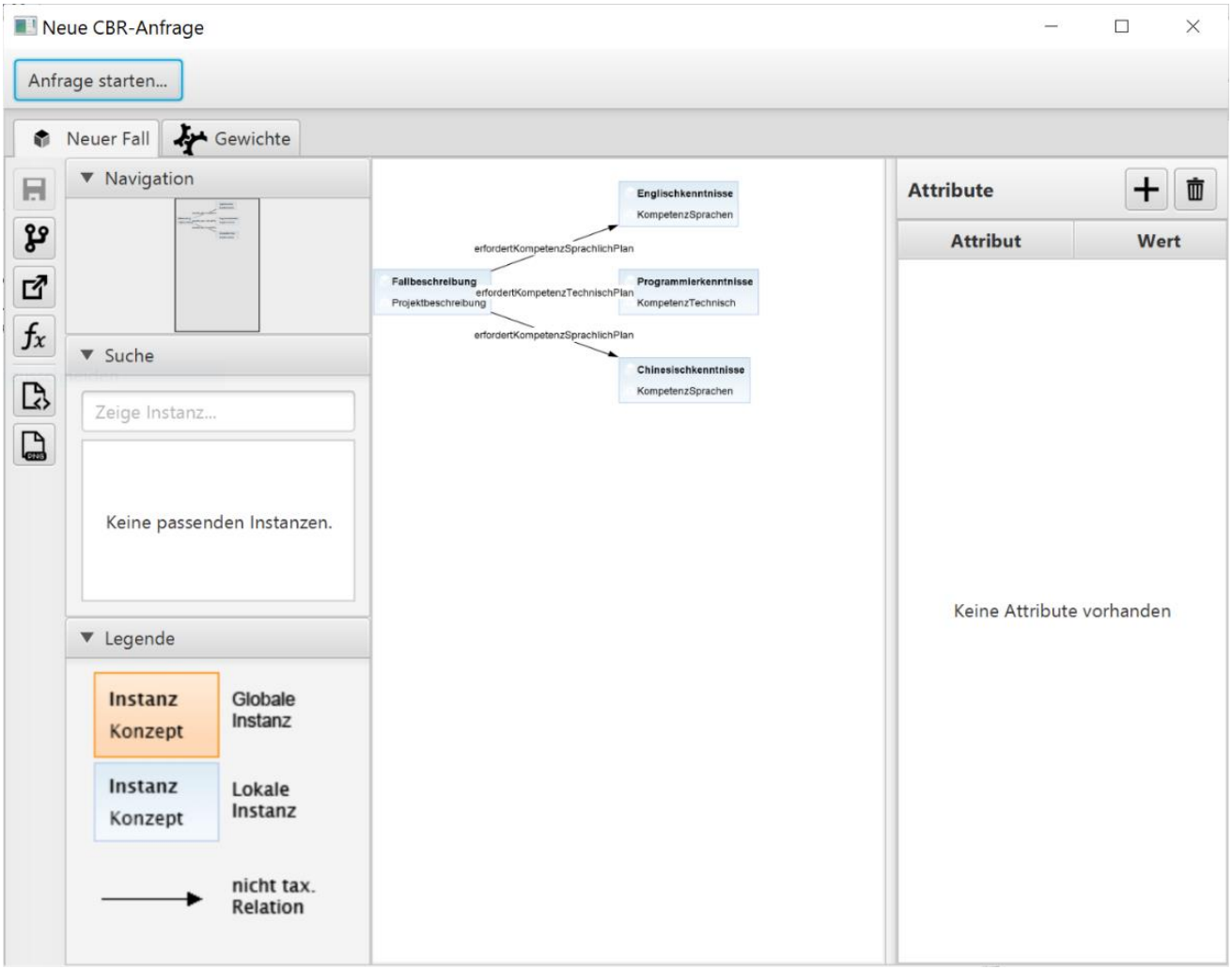

Abbildung 35: CBR-Anfrage in jCORA starten

Zudem können an dieser Stelle individuelle Gewichtungen für alle Relationen und Attribute aus der zugrunde liegenden Domänenontologie vergeben werden. Mittels dieser Gewichtungen ist es möglich, individuelle Präferenzen des Benutzers eines ontologiegestützten CBR-Systems hinsichtlich der projektspezifischen Bedeutungen von Relationen und Attributen auszudrücken. Mit diesen projektspezifischen Bedeutungen ist gemeint, welche Wichtigkeit der Benutzer eines ontologiegestützten CBR-Systems den Relationen und Attributen aus der zugrunde liegenden Domänenontologie im Einzelfall für die Erfüllung einer speziellen Projektmanagementaufgabe zumisst.

Alle Gewichtungen sind, wie in der nachfolgenden Abbildung 36 dargestellt, tabellarisch aufgeführt. Als Default-Wert sind alle Gewichtungen auf jeweils 100 % eingestellt, sodass alle Relationen und Attribute gleichgewichtet in die Ermittlung ähnlichster alter Fälle (Projekte) eingehen, solange dieser Default-Wert für einzelne (oder alle) Relationen und Attribute nicht abgeändert wird.

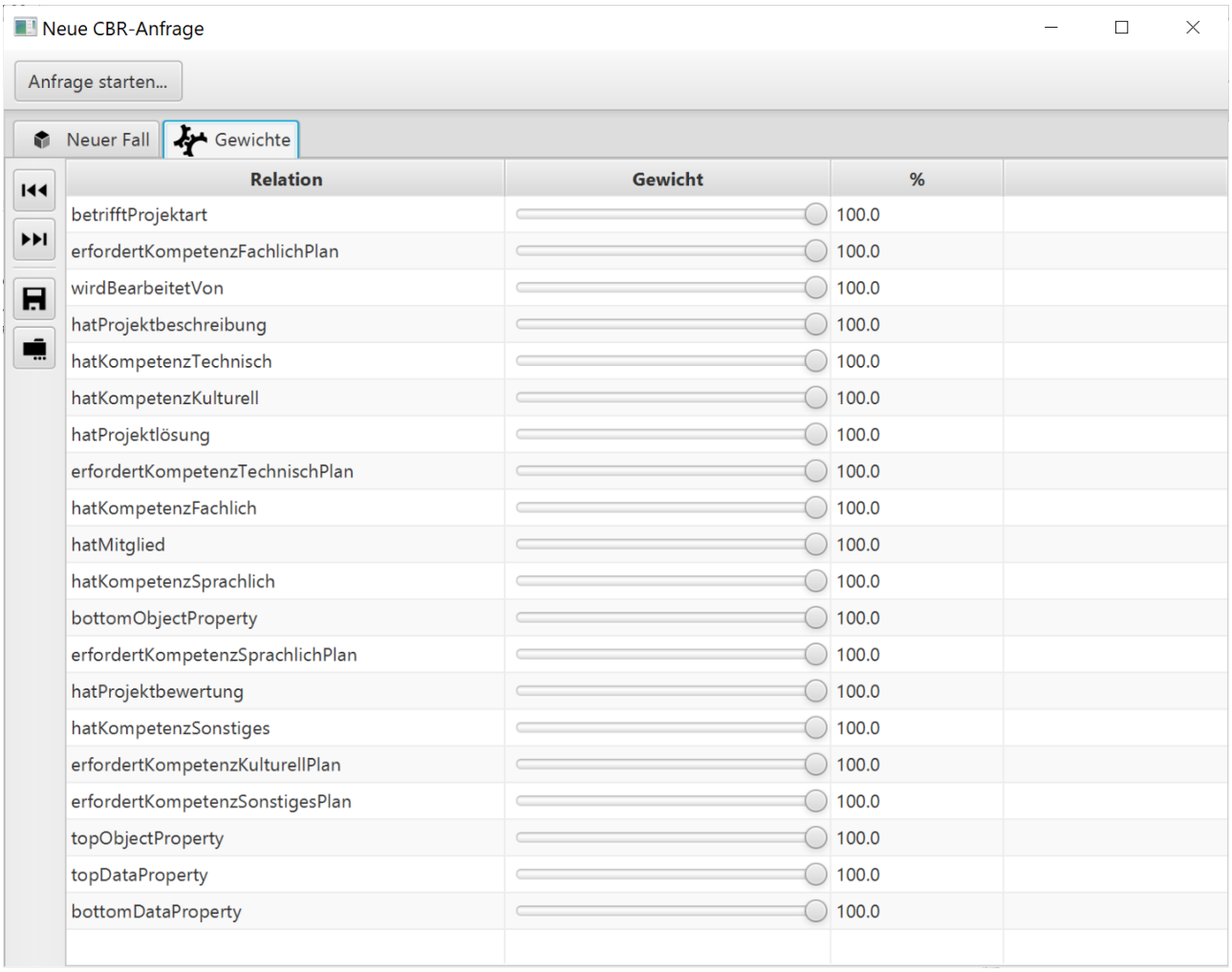

Abbildung 36: Einstellen von Gewichtungen in jCORA

Sofern die vorgenommen Gewichtung den Präferenzen des Benutzers entspricht, kann durch den Button „Anfrage starten…“, der sich in Abbildung 36 oben befindet, die Ermittlung ähnlichster alter Fälle als „CBR-Anfrage“ gestartet werden.

Die nachfolgende Abbildung 37 präsentiert in exemplarischer Weise die tabellarische Darstellung der Ergebnisse, die aus der Ähnlichkeitsermittlung für drei Fälle (Fall 3 ist der vorgegebene, neue Fall) resultieren.

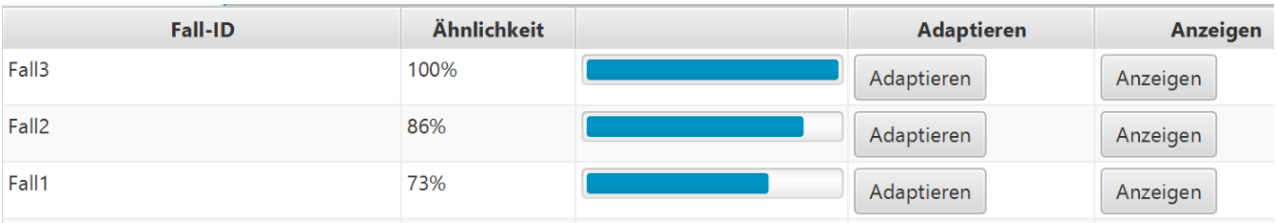

Abbildung 37: Ergebnis der CBR-Anfrage in jCORA

Die einzelnen Ergebnisse der CBR-Anfrage können anschließend angezeigt oder angepasst („adaptiert“)[270] werden.

270) Eine automatische Adaption ist in jCORA gegenwärtig wegen mangelnder Adaptionsregeln auf eine „Nulladaption“ (also ein reines Kopieren der Falllösungen alter Fälle für die Lösung eines neuen Falls) beschränkt. Neben einer (gegenwärtig stark eingeschränkten) automatischen Adaption ist in jCORA auch eine „manuelle“ Adaption möglich.

4.3 Kritische Würdigung des Vorgehensmodells

Das ontologiegestützte CBR-System jCORA befindet sich in einem experimentellen, prototypischen Stadium.

Die Domänenunterstützungsfunktion innerhalb der funktionalen Anforderungen hängt zu einem überwiegenden Teil von der zugrunde liegenden Domänenontologie ab.[271] jCORA kann diese wichtige funktionale Anforderung somit erfüllen.

Eine weitere wichtige Anforderung besteht in der Phasen- und Aufgabenunterstützungsfunktion. Die wichtigsten Bausteine sind die Ausarbeitung von Angeboten für ausgeschriebene Projekte und die Suche nach relevantem Erfahrungswissen aus bereits durchgeführten Projekten.[272] Diese Anforderungen werden von jCORA zur Zeit nicht erfüllt, da es sich bei dem Prototyp um ein reines auf dem Vergleich von Fällen basierendes CBR-System handelt.[273] Zwar kann eine Adaption bereits durchgeführter Fälle aus der Fallbasis vorgenommen werden, um die Fallstruktur für einen aktuellen Fall zu adaptieren. Jedoch entspricht dies nicht der geforderten Unterstützungsfunktion, die sich eher auf eine Prognosefunktion als auf eine reine Vergleichsfunktion beschränkt.

Die funktionalen Anforderungen einer Projektbeschreibungs- und bewertungsfunktion werden von jCORA erfüllt, da das CBR-System die dreiteilige Fallstruktur mit einer Projektbeschreibung, einer Projektlösung und einer Projektbewertung vorgibt.

Hinsichtlich des vorgestellten Vorgehensmodells ist zudem anzumerken, dass sich dessen Anwendungsreichweite „lediglich“ auf die Arbeit mit dem Ontologie-Editor Protégé und dem ontologiegestützten CBR-System jCORA beschränkt. Für andere Ontologie-Editoren bzw. andere ontologiegestützte CBR-Systeme sind gegebenenfalls Anpassungen des vorgestellten Vorgehensmodells vorzunehmen. Außerdem schöpft das Vorgehensmodell die semantische Ausdrucksfähigkeit des Ontologie-Editors Protégé nicht vollständig aus. Unberücksichtigt bleiben beispielsweise die konkrete Nutzung von Kardinalitäten, von Reasonern und auch von Semantic Web Rules.

271) Vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 88.

272) Vgl. SCHAGEN/ZELEWSKI/HEEB (2020), S. 89.

273) Vgl. BERGENRODT/KOWALSKI (2015), S. 107 f.

5 Fazit und Ausblick

Die Sicherung und Wiederverwendung von betrieblichem Erfahrungswissen stellt in einer globalisierten Welt, in der viele Unternehmen auf diversen Märkten im Wettbewerb zueinander stehen, einen nicht zu unterschätzenden Wettbewerbsvorteil dar. Erfahrungswissen liegt in expliziter und impliziter Form in Unternehmen vor. Die explizite Form umfasst z. B. Projektberichte, Debriefings und Lessons learned und wird, wenn Projekte eine längere Zeitspanne in der Vergangenheit liegen, oftmals nicht mehr aktiv genutzt, sondern lediglich archiviert. Das implizite Wissen, vor allem „in den Köpfen" der Projektbeteiligten, verweilt entweder so lange im Unternehmen, bis der betreffende Mitarbeiter das Unternehmen verlässt oder bis das Wissen „in den Köpfen" verblasst ist und vergessen wird.

Um die wertvolle Unternehmensressource des Erfahrungswissens zum einen zu erhalten und zum anderen wiederverwenden zu können, gibt es unterschiedliche Vorgehensweisen. Ontologien eignen sich insbesondere, um nicht nur Wissen über stark strukturierte Daten, wie sie z. B. in Projektdatenbanken vorliegen, zu speichern und wiederzuverwenden, sondern vor allem auch für inhaltliches Wissen abseits von numerischen Werten. Das implizite Wissen der Mitarbeiter des Projektmanagements über eine bestimmte Domäne lässt sich mithilfe einer Domänenontologie explizieren und wiederverwenden. Gleichzeitig schränkt dieser Vorgang die Machtposition der Mitarbeiter des Projektmanagements („Wissen ist Macht") ein, weil ein Teil ihres humanspezifischen Kapitals in Form von Erfahrungswissen durch die Aufnahme in ein ontologiegestütztes CBR-System in das Eigentum des Unternehmens übergeht. Diesem Machtverlust stehen aus der Mitarbeitersicht die Vorteile der Verwendung eines Wissensmanagementsystems gegenüber. Sie erstrecken sich vor allem auf eine Entlastung des Arbeitsaufwands bei der Planung, Durchführung und Steuerung neuer Projekte bei gleichzeitiger Befreiung von Routinetätigkeiten und einer qualitativ besseren – mit mehr Erfahrungswissen fundierten – Entscheidungsfindung im Projektmanagement.

Durch ein CBR-System ist es möglich, explizites Erfahrungswissen mithilfe von Ontologien für die betriebliche Praxis nutzbar zu machen, insbesondere wiederzuverwenden. Ein ontologiegestütztes CBR-System kann seinen Benutzer beim Management neuer Projekte unterstützen, indem das in der Fallbasis (Projektwissensbank) hinterlegte Erfahrungswissen aus alten, bereits durchgeführten Projekten für Anfragen in Bezug auf die neuen Projekte zur Verfügung gestellt wird. Dadurch erhöht sich nicht nur die Qualität der getroffenen Entscheidungen des Projektmanagements, sondern auch dessen Wirtschaftlichkeit.

Da den KI-Techniken der Ontologien und des Case-based Reasonings in der betrieblichen Praxis oftmals noch nicht die wünschenswerte Aufmerksamkeit zuteil wird, erhalten Unternehmen mithilfe des in diesem Projektbericht vorgestellten Vorgehensmodells die Möglichkeit, die einzelnen Schritte zur Erstellung eines ontologiegestützten CBR-Systems systematisch nachzuvollziehen und einen Überblick darüber zu erhalten, inwiefern die beiden vorgenannten KI-Techniken Beiträge zur täglichen Aufgabenbewältigung im Bereich des Projektmanagements leisten können.

Für die Zukunft ist es denkbar und wünschenswert, ein Softwarepaket zu entwickeln, das die derzeit noch recht komplizierte und wenig intuitive Benutzung eines Ontologie-Editors wie Protégé und eines CBR-Systems wie jCORA vereinheitlicht und soweit wie möglich vereinfacht. Zusätzlich ist ein auf dieses Softwarepaket zugeschnittenes Vorgehensmodell, inklusive eines Vorgehensmodells zur Erstellung und zum Betrieb eines ontologiegestützten CBR-Systems, wünschenswert, um eine höhere Akzeptanz dieser KI-Techniken in der betrieblichen Praxis zu erreichen.

Literaturverzeichnis

Vorbemerkungen:

- Alle Quellen werden im Literaturverzeichnis wie folgt aufgeführt: In der ersten Zeile wird der *Referenztitel* der Quelle angegeben. Er entspricht der Form, die im Text Verwendung findet, wenn auf die Quelle hingewiesen wird.
- Bei der Vergabe der Referenztitel wird bei *einem* Autor dessen Nachname, gefolgt von dem Erscheinungsjahr der Quelle in Klammern, verwendet. Existieren *zwei* oder *drei* Autoren, werden diese getrennt von einem Schrägstrich („/“) aufgeführt. Bei mindestens *vier* Autoren werden nur die ersten drei Autoren mit dem Zusatz „et al.“ aufgeführt.
- Die Quellen werden lexikografisch nach Maßgabe der Namen ihrer Autoren geordnet.
- Bei Quellen mit gleichen Autoren werden Quellen mit früheren Erscheinungsdaten vor Quellen mit neueren Erscheinungsdaten angeführt.
- Zwischen Quellen, die sich hinsichtlich ihrer Autoren und Erscheinungsdaten nicht unterscheiden, wird durch Zusätze wie „a“ und „b“ unterschieden.
- Zu *Internetquellen* wird die dafür verantwortliche Instanz aufgeführt. Dies können sowohl natürliche als auch juristische Personen sein. Für Internetquellen werden die zum Zugriffsdatum gültige Internetadresse (URL) und das Zugriffsdatum angegeben.

AAMODT/PLAZA (1994)

Aamodt, A.; Plaza, E.: Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. In: AI Communications, Vol. 7 (1994), No. 1, S. 39-59.

ABDOULLAEV (2008)

Abdoullaev, A.: Reality, universal ontology, and knowledge systems – Toward the intelligent world. Hershey 2008.

AMAILEF/LU (2013)

Amailef, K.; Lu, J.: Ontology-supported case-based reasoning approach for intelligent m-Government emergency response services. In: Decision Support Systems, Vol. 55 (2013), No. 1, S. 79-97.

ASSALI/LENNE/DEBRAY (2010)

Assali, A. A.; Lenne, D.; Debray, B.: Heterogeneity in Ontological CBR Systems. In: Montani, S.; Jain, L. C. (Hrsg.): Successful Case-Based reasoning Applications. Berlin 2010, S. 97-116.

BANDARA/HARMON/ROSEMANN (2011)

Bandara, W.; Harmon, P.; Rosemann, M.: Professionalizing Business Process Management: Towards a Body of Knowledge for BPM. In: Su, J.; zur Muehlen, M. (Hrsg.): Business Process Management Workshops, BPM 2010 international workshops and education track, Revised selected papers. 13.-15.09. 2010 in Hoboken, NJ, USA. Heidelberg 2011, S. 759-774.

BEIERLE/KERN-ISBERNER (2019)

Beierle, C.; Kern-Isberner, G.: Methoden wissensbasierter Systeme – Grundlagen, Algorithmen, Anwendungen. 6. Auflage, Wiesbaden 2019.

BEIßEL (2011)

Beißel, S.: Ontologiegestütztes Case-Based Reasoning – Entwicklung und Beurteilung semantischer Ähnlichkeitsindikatoren für die Wiederverwendung natürlichsprachlich repräsentierten Projektwissens. Dissertation, Universität Duisburg-Essen. Wiesbaden 2011.

BERGENRODT/KOWALSKI (2015)

Bergenrodt, D.; Kowalski, M.: Konzipierung, Implementierung und kritische Evaluierung einer Projektwissensbank auf Basis von semantischen Methoden der Künstlichen Intelligenz. OrGoLo-Projektbericht Nr. 31, Institut für Produktion und Industrielles Informationsmanagement, Universität Duisburg-Essen (Campus Essen). Essen 2015.

BERGENRODT/KOWALSKI/ZELEWSKI (2015)

Bergenrodt, D.; Kowalski, M.; Zelewski, S.: Prototypische Implementierung des ontologiegestützten CBR-Systems jCORA. In: Zelewski, S.; Akca, N.; Kowalski, M. (Hrsg.): Organisatorische Innovationen mit Good Governance und Semantic Knowledge Management in Logistik-Netzwerken – Wissenschaftliche Grundlagen und Praxisanwendungen. Berlin 2015, S. 475-553.

BERGMANN (1998)

Bergmann, R.: On the Use of Taxonomies for Representing Case Features and Local Similarity Measures. Technische Universität Kaiserslautern. Kaiserslautern 1998.

BODENDORF (1990)

Bodendorf, F.: Computer in der fachlichen und universitären Ausbildung. München et al. 1990.

BOEHM (1988)

Boehm, B.W.: A spiral model of software development and enhancement. In: Computer, Vol. 21 (1988), No. 5, S. 61-72.

BRUSA/CALIUSCO/CHIOTTI (2006)

Brusa, G.; Caliusco, L.; Chiotti, O.: A Process for Building a Domain Ontology: an Experience in Developing a Government Budgetary Ontology. In: Orgun, M.A.; Meyer, T. (Hrsg.): Advances in Ontologies 2006, Proceedings of the Second Australasian Ontology Workshop (AOW 2006), 05.12.2006 in Hobart. Sydney 2006, S. 7-15.

BUDDE/KAUTZ/KUHLENKAMP et al. (1990)

Budde, R.; Kautz, K.; Kuhlenkamp, K.; Züllighoven, H.: What is prototyping? In: Information Technology & People, Vol. 6 (1990), Nos. 2/3, S. 89-95.

BULLINGER/FÄHNRICH (1997)

Bullinger, H.-J.; Fähnrich, K.-P.: Betriebliche Informationssysteme – Grundlagen und Werkzeuge der methodischen Softwareentwicklung. Berlin et al. 1997.

CHOU (2009)

Chou, J.: Web-based CBR system applied to early cost budgeting for pavement maintenance project. In: Expert Systems with Applications, Vol. 36 (2009), No. 2, S. 2947-2960.

CURTS/CAMPBELL (2005)

Curts, R.J.; Campbell, D.E.: Building An Ontology For Command & Control – 10th International Command and Control Research and Technology Symposium – The Future of C2. Online-Publikation im Internet unter der URL: https://apps.dtic.mil/dtic/tr/fulltext/u2/a464304.pdf, veröffentlicht am 17.03.2005, letzter Zugriff am 22.03.2021.

DENGEL/BERNARDI/VAN ELST (2012)

Dengel, A.; Bernardi, A.; van Elst, L.: Wissensrepräsentation. In: Dengel, A. (Hrsg.): Semantische Technologien. Grundlagen – Konzepte – Anwendungen. Heidelberg 2012, S. 21-72.

DE TONI/PESSOT (2021)

De Toni, A.F.; Pessot, E.: Investigating organisational learning to master project complexity: An embedded case study. In: Journal of Business Research, Vol. 129 (2021), S. 541-554.

DITTMANN (2007)
Dittmann, L.U.: OntoFMEA – Ontologiebasierte Fehlermöglichkeits- und Einflussanalyse. Dissertation, Universität Duisburg-Essen. Wiesbaden 2007.

DITTMANN/APKE (2005)
Dittmann, L.U.; Apke, S.: Vorgehensmodelle zur Konstruktion von Ontologien – Eine darstellende Untersuchung. In: Zelewski, S. (Hrsg.): Ontologiebasierte Kompetenzmanagementsysteme – Grundlagen, Konzepte, Anwendungen. Berlin 2005, S. 277-319.

DRAGONI/PORIA/CAMBRIA (2018)
Dragoni, M.; Poria, S.; Cambria, E.: OntoSenticNet: A Commonsense Ontology for Sentiment Analysis. In: IEEE Intelligent Systems, Vol. 33 (2018), No. 3, S. 77-85.

EBERT (2017)
Ebert, C.: Systematisches Requirements Engineering – Anforderungen ermitteln, dokumentieren, analysieren und verwalten. 6. Aufl., Heidelberg 2017.

EVELEENS/VERHOEF (2010)
Eveleens, J.L.; Verhoef, C.: The Rise and Fall of the Chaos Report Figures. In: IEEE Software, Vol. 27 (2010), No. 1, S. 30-36.

FENSEL (2004)
Fensel, D.: Ontologies – A Silver Bullet for Knowledge Management and Electronic Commerce. Berlin - Heidelberg 2004.

FERNÁNDEZ-LÓPEZ/GÓMEZ-PÉREZ/JURISTO (1997)
Fernández-López, M.; Gómez-Pérez, A.; Juristo, N.: Methontology: from ontological art towards ontological engineering. In: Farquhar, A.; Grüninger, M. (Hrsg.): Ontological Engineering: Papers from the AAAI Spring Symposium. Menlo Park 1997, S. 33-40.

FISCHER/BISKUP/MÜLLER-LUSCHNAT (1998)
Fischer, T.; Biskup, H.; Müller-Luschnat, G.: Begriffliche Grundlagen für Vorgehensmodelle. In: Kneuper, R.; Müller-Luschnat, G.; Oberweis, A. (Hrsg.): Vorgehensmodelle für die betriebliche Anwendungsentwicklung. Stuttgart et al. 1998, S. 13-31 (beitragsindividuelle Paginierung: S. 1-17).

FOSSATI/GHIDONI/DI EUGENIO et al. (2006)
Fossati, D.; Ghidoni, G.; Di Eugenio, B.; Cruz, I.; Xiao, H.; Subba, R.: The problem of ontology alignment on the web: a first report. In: Kilgarriff, A.; Baroni, M. (Hrsg.): WAC'06: Proceedings of the 2nd International Workshop on Web as Corpus, 01.04.2006 in Trento. East Stroudsburg 2006, S. 51-58.

FREUDENTHALER (2008)
Freudenthaler, B.: Case-based Reasoning (CBR) – Grundlagen und ausgewählte Anwendungsgebiete des fallbasierten Schließens. Saarbrücken 2008.

FREUND/RÜCKER (2019)
Freund, J.; Rücker, B.: Praxishandbuch BPMN – Mit Einführung in DMN. 6. Auflage. München 2019.

GANDON (2010)
Gandon, F.L.: Ontologies in Computer Science: These New "Software Components" of Our Information Sysetms. In: Gargouri, F.; Jaziri, W. (Hrsg.): Ontology Theory, Management, and Design: Advanced Tools and Models. Hershey 2010, S. 1-26.

GLASS (2006)
Glass, R.L.: The Standish Report: Does It Really Describe a Software Crisis? In: Communications of the ACM, Vol. 49 (2006), No. 8, S. 15-16.

GÓMEZ-PÉREZ/FERNÁNDEZ-LÓPEZ/CORCHO (2004)
Gómez-Pérez, A.; Fernández-López, M.; Corcho, O.: Ontological engineering – With examples from the areas of knowledge management, e-commerce and the semantic web. London 2004.

GRÜNINGER/FOX (1995)
Grüninger, M.; Fox, M.S.: Methodology for the design and evaluation of ontologies. In: Skuce, D. (Hrsg.): IJCAI95 Workshop on Basic Ontological Issues in Knowledge Sharing. Montreal 1995, S. 6.1-6.10 (beitragsindividuelle Paginierung: S. 1-10).

GROB/SEUFERT (1996)
Grob, H.L.; Seufert, S.: Vorgehensmodelle bei der Entwicklung von CAL-Software. Münster 1996.

GRUBER (1995)
Gruber, T.R.: Toward principles for the design of ontologies used for knowledge sharing. In: International Journal of Human-Computer Studies, Vol. 43 (1995), Nos. 5-6, S. 907-928.

HAARMANN (2014)
Haarmann, B.: Ontology On Demand – Vollautomatische Ontologieerstellung aus deutschen Texten mithilfe moderner Textmining-Prozesse. Dissertation, Universität Bochum, Fakultät für Philologie. Berlin 2014.

HEBELER/FISHER/BLACE et al. (2009)
Hebeler, J.; Fisher, M.; Blace, R.; Perez-Lopez, A.: Semantic Web Programming. Indianapolis 2009.

HOLSAPPLE/JOSHI (2002)
Holsapple, C.W.; Joshi, K.D.: A Collaborative Approach to Ontology Design. In: Communications of the ACM, Jg. 45 (2002), No. 2, S. 42-47.

JAKUS/MILUTINOVIĆ/OMEROVIĆ et al. (2013)
Jakus, G.; Milutinović, V.; Omerović, S.; Tomažič, S.: Concepts, ontologies, and knowledge representation. New York 2013.

KOWALSKI/BERGENRODT/ZELEWSKI (2015)
Kowalski, M.; Bergenrodt, D.; Zelewski, S.: Prototypische Implementierung des ontologiegestützten CBR-Tools mit jColibri. In: Zelewski, S.; Akca, N.; Kowalski, M. (Hrsg.): Organisatorische Innovationen mit Good Governance und Semantic Knowledge Management in Logistik-Netzwerken – Wissenschaftliche Grundlagen und Praxisanwendungen. Berlin 2015, S. 415-474.

KOWALSKI/KLÜPFEL/ZELEWSKI et al. (2012)
Kowalski, M.; Klüpfel, H.; Zelewski, S.; Bergenrodt, D.; Becker, A.: Implementation of an Ontology-driven CBR-System for the Intelligent Reuse Project-Related Knowledge. In: Perner, P. (Hrsg.): Advances in Data Mining, 12th Industrial Conference on Data Mining ICDM 2012, 16.-20.07.2012 in Berlin, Poster and Industry Proceedings. Fockendorf 2012, S. 80-88.

KOWALSKI/ZELEWSKI/GÜNES et al. (2011)
Kowalski, M.; Zelewski, S.; Günes, N.; Kühn, T.: Kostenschätzungen für die Reaktivierung passiver Gleisanschlüsse – Eine neue Methode für Kostenschätzungen mithilfe von Case-based Reasoning (CBR) basiert auf der Wiederverwendung von historischem Projektwissen. In: EI – Der Eisenbahningenieur, 62. Jg. (2011), Heft 6, S. 49-54.

KOWALSKI/ZELEWSKI (2015)

Kowalski, M.; Zelewski, S.: Erstellung einer Verpackungsontologie mithilfe des Ontologie-Editors Protégé. In: Zelewski, S.; Akca, N.; Kowalski, M. (Hrsg.): Organisatorische Innovationen mit Good Governance und Semantic Knowledge Management in Logistik-Netzwerken – Wissenschaftliche Grundlagen und Praxisanwendungen. Berlin 2015, S. 593-614.

KURBEL/DORNHOFF (1993)

Kurbel, K.; Dornhoff, P.: Aufwandschätzung für Software-Entwicklungsprojekte mit Hilfe fallbasierter Wissensverarbeitung. In: Zeitschrift für Betriebswirtschaft, Jg. 63 (1993), Heft 10, S. 1047-1065.

LIAO (2005)

Liao, S.-H.: Expert system methodologies and applications—a decade review from 1995 to 2004. In: Expert Systems with Applications, Vol. 28 (2005), No 1, S. 93-103.

LIM/LIU/LEE (2011)

Lim, E.H.Y.; Liu, J.N. K.; Lee, R.S.T.: Knowledge Seeker – Ontology Modelling for Information Search and Management: A Compendium. Berlin - Heidelberg 2011.

MARTIN/EMMENEGGER/HINKELMANN et al. (2017)

Martin, A.; Emmenegger, S.; Hinkelmann, K.; Thönssen, B.: A viewpoint-based case-based reasoning approach utilising an enterprise architecture ontology for experience management. In: Enterprise Information Systems, Vol. 11 (2017), No. 4, S. 551-575.

MCAFEE/BRYNJOLFSSON (2012)

McAfee, A.; Brynjolfsson, E.: Big Data: The Management Revolution. In: Harvard Business Review, Vol. 90 (2012), No. 10, S. 60-68.

MCDANIEL/STOREY/SUGUMARAN (2018)

McDaniel, M.; Storey, V.C.; Sugumaran, V.: Assessing the quality of domain ontologies: Metrics and an automated ranking system. In: Data & Knowledge Engineering, Vol. 115 (2018), S. 32-47.

MEHLER/WOLFF (2005)

Mehler, A.; Wolff, C.: Einleitung: Perspektiven und Positionen des Text Mining. In: Zeitschrift für Computerlinguistik und Sprachtechnologie, Vol. 20 (2005), No. 1, S. 1-18.

NIEBISCH (2013)

Niebisch, T.: Anforderungsmanagement in sieben Tagen: Der Weg vom Wunsch zur Konzeption. Berlin et al. 2013.

NONAKA/TAKEUCHI (2012)

Nonaka, I.; Takeuchi, H.: Die Organisation des Wissens: Wie japanische Unternehmen eine brachliegende Ressource nutzbar machen. 2. Aufl., Frankfurt am Main 2012.

NOY/CRUBÉZY/FERGERSON et al. (2003)

Noy, N.F.; Crubézy, M.; Fergerson, R.W.; Knublauch, H.; Tu, W.S.; Vendetti, J.; Musen, M. A.: Protégé-2000: An Open-Source Ontology-Development and Knowledge-Acquisition Environment. In: AMIA Annual Symposium Proceedings Archive, Vol. 2003, S. 953.

NOY/MCGUINNESS (2001)

Noy, N.F.; McGuinness, D.L.: Ontology Development 101: A Guide to Creating Your First Ontology. Online-Publikation im Internet unter der URL: https://protege.stanford.edu/publications/ontology_development/ontology101.pdf, veröffentlicht am 01.01.2001, letzter Zugriff am 21.03.2021.

OBJECT MANAGEMENT GROUP (2013)
Object Management Group: Business Process Model and Notation (BPMN) – Version 2.0.2, Online-Publikation im Internet unter der URL: https://www.omg.org/spec/BPMN/2.0.2/PDF, letzter Zugriff am 02.13.2021.

PFUHL (2003)
Pfuhl, M.: Case-Based Reasoning auf der Grundlage Relationaler Datenbanken: Eine Anwendung zur strukturierten Suche in Wirtschaftsnachrichten. Dissertation, Universität Marburg. Wiesbaden 2003.

POHL (2007)
Pohl, K.: Requirements engineering – Grundlagen, Prinzipien, Techniken. Heidelberg 2007.

POHL/RUPP (2015)
Pohl, K.; Rupp, C.: Basiswissen Requirements Engineering: Aus- und Weiterbildung zum Certified Professional for Requirements Engineering Foundation Level. 4. Aufl., Heidelberg 2015.

PÖHLAND/KORZETZ/SCHLEGEL (2013)
Pöhland, R.; Korzetz, M.; Schlegel, T.: Semantische Modellierung und Klassifikation von Touch-Interaktionen. In: Schlegel, T. (Hrsg.): Multi-Touch: Interaktion durch Berührung. Berlin - Heidelberg 2013, S. 69-88.

RICHTER/WEBER (2013)
Richter, M.M.; Weber, R.O.: Case-based reasoning: A textbook. Berlin - Heidelberg 2013.

RIESBECK/SCHANK (2013)
Riesbeck, C.K.; Schank, R.C.: Inside Case-Based Reasoning. Hoboken 2013.

ROTH-BERGHOFER (2004)
Roth-Berghofer, T.R.: Explanations and Case-Based Reasoning: Fundational Issues. In: Funk, P. (Hrsg.): Advances in case-based reasoning – 7th European conference, ECCBR 2004, 30.-08.-02.09.2004 in Madrid, Proceedings. Berlin 2004, S. 389-403.

RUIZ/TORRES/CRESPO (2021)
Ruiz, J.G.; Torres, J.M.; Crespo, R.G.: The Application of Artificial Intelligence in Project Management Research: A Review. In: International Journal of Interactive Multimedia and Artificial Intelligence, Vol. 6 (2021), No. 6, S. 54-65.

RUPP (2014)
Rupp, C.: Requirements-Engineering und -Management – Aus der Praxis von klassisch bis agil. 6. Aufl., München 2014.

SAATY (1994)
Saaty, T.L.: How to Make a Decision: The Analytic Hierarchy Process. In: Interfaces, Vol. 24 (1994), No. 6, S. 19-43.

SCHAGEN/ZELEWSKI/HEEB (2020)
Schagen, J.P.; Zelewski, S.; Heeb, T.: Erhebung und Analyse der Anforderungen an ein KI-Tool aus der Perspektive der betrieblichen Praxis – mit Fokus auf der Wiederverwendung von Erfahrungswissen im Bereich des betrieblichen Projektmanagements. Institut für Produktion und Industrielles Informationsmanagement, Arbeitsbericht Nr. 47, zugleich KI-LiveS-Projektbericht Nr. 1. Universität Duisburg-Essen (Campus Essen). Essen 2020.

SCHARFFE/FENSEL (2008)
Scharffe, F.; Fensel, D.: Correspondence Patterns for Ontology Alignment. In: Gangemi, A.; Euzenat, J. (Hrsg.): Knowledge Engineering: Practice and Patterns – 16th International Conference, EKAW 2008, 29.09.-02.10.2008 in Acitrezza, Proceedings. Berlin 2008, S. 83-92.

SCHNEIDER/DAUN/BEHRENS et al. (2006)
Schneider, K.; Daun, C.; Behrens, H.; Wagner, D.: Vorgehensmodelle und Standards zur systematischen Entwicklung von Dienstleistungen. In: Bullinger, H.-J.; Scheer, A.-W. (Hrsg.): Service Engineering – Entwicklung und Gestaltung innovativer Dienstleistungen. 2. Aufl., Berlin - Heidelberg 2006, S. 113-138.

SCHUHBAUER/FUHR/WITTMANN (2008)
Schuhbauer, H.; Fuhr, T.; Wittmann, S.: Flexible Informationsstrukturen mit Ontologien. In: HMD: Praxis der Wirtschaftsinformatik, Jg. 45 (2008), Nr. 262, S. 97-105.

SCHULZ/NEUHAUS/KAUFMANN et al. (2020)
Schulz, M.; Neuhaus, U.; Kaufmann, J.; Badura, D.; Kerzel, U.; Welter, F.; Prothmann, M.; Kühnel, S.; Passlick, J.; Rissler, R.; Badewitz, W.; Dann, D.; Gröschel, A.; Kloker, S.; Alekozai, E. M.; Felderer, M.; Lanquillon, C.; Brauner, D.; Gölzer, P.; Binder, H.; Rhode, H.; Gehrke, N.: DASC-PM v1.0 – Ein Vorgehensmodell für Data-Science-Projekte. Elmshorn 2020.

SLADE (1991)
Slade, S.: Case-Based Reasoning: A Research Paradigm. In: AI Magazine, Vol. 12 (1991), No. 1, S. 42-55.

STUART (2016)
Stuart, D.: Practical Ontologies for Informtion Professionals. London 2016.

STUCKENSCHMIDT (2011)
Stuckenschmidt, H.: Ontologien – Konzepte, Technologien und Anwendungen. 2. Aufl., Berlin 2011.

STUDER/BENJAMINS/FENSEL (1998)
Studer, R.; Benjamins, R.V.; Fensel, D.: Knowledge Engineering: Principles and Methods. In: Data & Knowledge Engineering, Vol. 25 (1998), S. 161-197.

SU/ILEBREKKE (2002)
Su, X.; Ilebrekke, L.: A Comparative Study of Ontology Languages and Tools. In: Pidduck, A.B.; Mylopoulos, J.; Ozsu, M.T.; Woo, C.C. (Hrsg.): Advanced Information Systems Engineering: 14th International Conference, CAiSE 2002, 27.-31.05.2002 in Toronto, Proceedings. Berlin - Heidelberg 2002, S. 761-765.

TAN (1999)
Tan, A.-H.: Text Mining: The state of the art and the challenges. In: Proceedings of the PAKDD'99 Workshop on Knowledge Disocovery from Advanced Databases (KDAD'99), 26.04.1999 in Beijing. Berlin - Heidelberg 1999, S. 65-70 (beitragsindividuelle Paginierung: S. 1-6).

THIER (2017)
Thier, K.: Storytelling: Eine Methode für das Change-, Marken-, Projekt- und Wissensmanagement. 3. Aufl., Berlin et al. 2017.

USCHOLD/GRUNINGER (1996)
Uschold, M.; Gruninger, M.: Ontologies: Principles, Methods and Applications. Edinburgh 1996.

VON KOCEMBA/BELZ (2015)
von Kocemba, D.F.; Belz, H.J. (Teamleitung): Ergänzung und Veränderung von Erfolgsfaktoren im Projektmanagement bei zunehmender Internationalisierung. Herausgegeben von der GPM – Deutsche Gesellschaft für Projektmanagement e. V. Nürnberg o. J. (Dokument laut Metadaten erstellt am 11.12.2015).

WAND/MONARCHI/PARSONS et al. (1995)
Wand, Y.; Monarchi, D.E.; Parsons, J.; Woo, C.C.: Theoretical foundations for conceptual modelling in information systems development. In: Decision Support Systems, Vol. 15 (1995), No. 4, S. 285-304.

WATERFELD/WEITEN/HAASE (2011)
Waterfeld, W.; Weiten, M.; Haase, P.: Ontology Management Infrastructures. In: Hepp, M.; De Leenheer, P.; de Moor, A.; Sure, Y. (Hrsg.): Ontology Management – Semantic Web, Semantic Web Services, and Business Applications. New York 2011, S. 59-87.

WATSON (1998)
Watson, I.D.: Applying Case-Based Reasoning: Techniques for Enterprise Systems. San Francisco 1998.

WATSON (2003)
Watson, I.D.: Applying Knowledge Management: Techniques for Building Corporate Memories. San Francisco 2003.

WEBER/HEEB/SETHUPATHY et al. (2021)
Weber, L.; Heeb, T.; Sethupathy, G.; Schagen, J.P.; Zelewski, S.: „Intelligente“ Wiederverwendung von Erfahrungswissen im betrieblichen Projektmanagement mithilfe von KI-Techniken bei sicherheitskritischen IT-Projekten mit Fokus auf PRINCE2 und Risikomanagement. Arbeitsbericht Nr. 50, Institut für Produktion und Industrielles Informationsmanagement, Universität Duisburg-Essen (Campus Essen), zugleich KI-LiveS-Projektbericht Nr. 4. Essen 2021.

WILKE/BERGMANN (1998)
Wilke, W.; Bergmann, R.: Techniques and Knowledge Used for Adaptation During Case-Based Problem Solving. In: del Pobil, A. P.; Mira, J.; Ali, M. (Hrsg.): Tasks and Methods in Applied Artificial Intelligence: 11th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, IEA/AIE-98, 01.-04.06.1998 in Castellón, Proceedings. 2. Aufl., Berlin et al.1998, S. 497-506.

ZELEWSKI (1999)
Zelewski, S.: Ontologien zur Strukturierung von Domänenwissen – Ein Annäherungsversuch aus betriebswirtschaftlicher Perspektive. Institut für Produktion und Industrielles Informationsmanagement, Arbeitsbericht Nr. 3. Universität Duisburg-Essen (Campus Essen). Essen 1999.

ZELEWSKI (2002)
Zelewski, S.: Wissensmanagement mit Ontologien. In: Essener Unikate, Jg. 11 (2002), Heft 18, S. 62-73.

Zelewski (2005)
Zelewski, S.: Einführung in das Themenfeld „Ontologien“ aus informations- und betriebswirtschaftlicher Perspektive. In: Zelewski, S.; Alan, Y; Alparslan, A.; Dittmann, L.; Weichelt, T. (Hrsg.): Ontologiebasierte Kompetenzmanagementsysteme – Grundlagen, Konzepte, Anwendungen. Berlin 2005, 115-228.

ZELEWSKI/BRUNS/KOWALSKI (2012)
Zelewski, S.; Bruns, A. S.; Kowalski, M.: Ontologies for Guaranteeing the Interoperability in e-Business: A Business Economics Point of View. In: Kajan, E.; Dorloff, F.-D.; Bedini, I. (Hrsg.): Handbook of Research on E-Business Standards and Protocols: Documents, Data and Advanced Web Technologies. Hershey 2012, S. 154-184.

ZELEWSKI/KOWALSKI/BERGENRODT (2015)

Zelewski, S.; Kowalski, M.; Bergenrodt, D.: Management von Erfahrungswissen aus internationalen Logistik-Projekten mithilfe von Case-based Reasoning. In: Zelewski, S.; Akca, N.; Kowalski, M. (Hrsg.): Organisatorische Innovationen mit Good Governance und Semantic Knowledge Management in Logistik-Netzwerken – Wissenschaftliche Grundlagen und Praxisanwendungen. Berlin 2015, S. 229-268.

ZELEWSKI/SCHAGEN (2022)

Zelewski, S.; Schagen, J. P.: Case-based Reasoning als KI-Technik zur „intelligenten", computergestützten Wiederverwendung von Erfahrungswissen im Projektmanagement. Arbeitsbericht Nr. 55, Institut für Produktion und Industrielles Informationsmanagement, Universität Duisburg-Essen (Campus Essen), zugleich KI-LiveS-Projektbericht Nr. 9. Essen 2021.

Institut für Produktion und Industrielles Informationsmanagement Universität Duisburg-Essen / Campus Essen

Verzeichnis der Arbeitsberichte (ISSN 1614-0842)

Nr. 1: Zelewski, S.: Stickels theoretische Begründung des Produktivitätsparadoxons der Informationstechnik. Universität Essen, Essen 1999.

Nr. 2: Zelewski, S.: Flexibilitätsorientierte Koordinierung von Produktionsprozessen. Universität Essen, Essen 1999.

Nr. 3: Zelewski, S.: Ontologien zur Strukturierung von Domänenwissen. Universität Essen, Essen 1999.

Nr. 4: Siedentopf, J.; Schütte, R.; Zelewski, S.: Wirtschaftsinformatik und Wissenschaftstheorie. Universität Essen, Essen 1999.

Nr. 5: Fischer, K.; Zelewski, S.: Ontologiebasierte Koordination von Anpassungsplanungen in Produktions- und Logistiknetzwerken mit Multi-Agenten-Systemen. Universität Essen, Essen 1999.

Nr. 6: Weihermann, A. E.; Wöhlert, K.: Gentechnikakzeptanz und Kommunikationsmaßnahmen in der Lebensmittelindustrie. Universität Essen, Essen 1999.

Nr. 7: Schütte, R.: Zum Realitätsbezug von Informationsmodellen. Universität Essen, Essen 2000.

Nr. 8: Zelewski, S.: Erweiterungen eines Losgrößenmodells für betriebliche Entsorgungsprobleme. Universität Essen, Essen 2000.

Nr. 9: Schütte, R.: Wissen, Zeichen, Information, Daten. Universität Essen, Essen 2000.

Nr. 10: Hemmert, M.: The Impact of Internationalization and Externalization on the Technology Acquisition Performance of High-Tech Firms. Universität Essen, Essen 2001.

Nr. 11: Hemmert, M.: Erfolgswirkungen der internationalen Organisation von Technologiegewinnungsaktivitäten. Universität Essen, Essen 2001.

Nr. 12: Hemmert, M.: Erfolgsfaktoren der Technologiegewinnung von F&E-intensiven Großunternehmen. Universität Essen, Essen 2001.

Nr. 13: Schütte, R.; Zelewski, S.: Epistemological Problems in Working with Ontologies. Universität Essen, Essen 2001.

Nr. 14: Peters, M. L.; Zelewski, S.: Analytical Hierarchy Process (AHP). Universität Essen, Essen 2002.

Nr. 15: Zelewski, S.: Wissensmanagement mit Ontologien. Universität Essen, Essen 2002.

Nr. 16: Klumpp, M.; Krol, B.; Zug, S.: Management von Kompetenzprofilen im Gesundheitswesen. Universität Essen, Essen 2002.

Nr. 17: Zelewski, S.: Der „non statement view“ – eine Herausforderung für die (Re-) Konstruktion wirtschaftswissenschaftlicher Theorien. Universität Essen, Essen 2002.

Nr. 18: Peters, M. L.; Zelewski, S.: A heuristic algorithm to improve the consistency of judgments in the Analytical Hierarchy Process (AHP). Universität Duisburg-Essen (Campus Essen), Essen 2003.

Nr. 19: Peters, M. L.; Zelewski, S.: Fallstudie zur Lösung eines Standortplanungsproblems mit Hilfe des Analytical Hierarchy Process (AHP). Universität Duisburg-Essen (Campus Essen), Essen 2003.

Nr. 20: Zelewski, S.: Konventionelle versus strukturalistische Produktionstheorie. Universität Duisburg-Essen (Campus Essen), Essen 2003.

Nr. 21: Alparslan, A.; Zelewski, S.: Moral Hazard in JIT Production Settings. Universität Duisburg-Essen (Campus Essen), Essen 2004.

Nr. 22: Dittmann, L.: Ontology-based Skills Management. Universität Duisburg-Essen (Campus Essen), Essen 2004.

Nr. 23: Peters, M. L.; Zelewski, S.: Ein Modell zur Auswahl von Produktionsaufträgen unter Berücksichtigung von Synergien. Universität Duisburg-Essen (Campus Essen), Essen 2004.

Nr. 24: Peters, M. L.; Zelewski, S.: Ein Modell zur Zuordnung ähnlicher Kundenbetreuer zu Kunden. Universität Duisburg-Essen (Campus Essen), Essen 2004.

Nr. 25: Zelewski, S.: Kooperatives Wissensmanagement in Engineering-Netzwerken – (vorläufiger) Abschlussbericht zum Verbundprojekt KOWIEN. Universität Duisburg-Essen (Campus Essen), Essen 2004.

Nr. 26: Siemens, F.: Vorgehensmodell zur Auswahl einer Variante der Data Envelopment Analysis. Universität Duisburg-Essen (Campus Essen), Essen 2005.

Nr. 27: Alan, Y.: Integrative Modellierung kooperativer Informationssysteme – Ein Konzept auf der Basis von Ontologien und Petri-Netzen. Dissertation, Universität Duisburg-Essen (Campus Essen), Essen 2005.

Nr. 28: Akca, N.; Ilas, A.: Produktionsstrategien – Überblick und Systematisierung. Universität Duisburg-Essen (Campus Essen), Essen 2005.

Nr. 29: Zelewski, S.: Relativer Fortschritt von Theorien – ein strukturalistisches Rahmenkonzept zur Beurteilung der Fortschrittlichkeit wirtschaftswissenschaftlicher Theorien (Langfassung). Universität Duisburg-Essen (Campus Essen), Essen 2005.

Nr. 30: Peters, M. L.; Schütte, R.; Zelewski, S.: Erweiterte Wirtschaftlichkeitsanalyse mithilfe des Analytic Hierarchy Process (AHP) unter Berücksichtigung des Wissensmanagements zur Beurteilung von Filialen eines Handelsunternehmens. Universität Duisburg-Essen (Campus Essen), Essen 2006.

Nr. 31: Zelewski, S.: Beurteilung betriebswirtschaftlichen Fortschritts – ein metatheoretischer Ansatz auf Basis des „non statement view“ (Langfassung). Universität Duisburg-Essen (Campus Essen), Essen 2006.

Nr. 32: Kijewski, F.; Moog, M.; Niehammer, M.; Schmidt, H.; Schröder, K.: Gestaltung eines Vorgehensmodells für die Durchführung eines Promotionsprojekts am Fachbereich Wirtschaftswissenschaften der Universität Duisburg-Essen, Campus Essen, zum Erwerb des „Dr. rer. pol.“ mithilfe von PETRI-Netzen. Universität Duisburg-Essen (Campus Essen), Essen 2006.

Nr. 33: Peters, M. L.; Zelewski, S.: Effizienzanalyse unter Berücksichtigung von Satisfizierungsgrenzen für Outputs – Die Effizienz-Analysetechnik EATWOS. Universität Duisburg-Essen (Campus Essen), Essen 2006.

Nr. 34: Häselhoff, I.; Meves, Y.; Munsch, D.; Munsch, S.; Schulte-Euler, D.; Thorant, C.: Anforderung an eine verbesserte Lehrqualität – Qualitätsplanung mittels House of Quality. Universität Duisburg-Essen (Campus Essen), Essen 2007.

Nr. 35: Zelewski, S.: Das ADL-Modell der Prinzipal-Agent-Theorie für die Just-in-Time-Produktionssteuerung – Darstellung, Analyse und Kritik. Universität Duisburg-Essen (Campus Essen), Essen 2008.

Nr. 36: Peters, M. L.; Zelewski, S.: Analyse der Effizienzentwicklung von Bankfilialen mithilfe des Operational Competitiveness Ratings (OCRA). Universität Duisburg-Essen (Campus Essen), Essen 2010.

Nr. 37: Peters, M. L.; Zelewski, S.: Fallstudie zu Porters generischen Wettbewerbsstrategien im Kontext nachhaltigen Wirtschaftens. Universität Duisburg-Essen (Campus Essen), Essen 2010.

Nr. 38: Peters, M. L.; Zelewski, S.: Erweiterung von EATWOS um die Berücksichtigung von Satisfizierungsgrenzen für Inputs. Universität Duisburg-Essen (Campus Essen), Essen 2012.

Nr. 39: Bergenrodt, D.; Jene, S.; Zelewski, S.: Implementierung des Tau-Werts. Universität Duisburg-Essen (Campus Essen), Essen 2013.

Nr. 40: Millan-Torres, J.; Arndt, C.: Erstellung eines Businessplans zur Existenzgründung des Unternehmens Cowdy! – Anwendung des „Fast-Casual"-Konzepts auf ein systemgastronomisch organisiertes Restaurant mit dem Schwerpunkt der Steakzubereitung. Universität Duisburg-Essen (Campus Essen), Essen 2014.

Nr. 41: Klumpp, M.; Oeben, M.; Zelewski, S.: Evaluation internationaler Bildungstransfer – Konzeptioneller Rahmen und Diskurs zur wissenschaftlichen Bewertung im Forschungs- und Transferprojekt OpporTUNIty. Universität Duisburg-Essen (Campus Essen), Essen 2018.

Nr. 42: Oeben, M.; Gerlach, A.-T.; Akdogan, D.; Arabaci, T.; Bagbasi, F.; Gudieva, A.; Klumpp, M.: Evaluation von Bildungsleistungen in Deutschland und Tunesien – das Beispiel des Hochschulsektors. Universität Duisburg-Essen (Campus Essen), Essen 2018.

Nr. 43: Oeben, M.; Klumpp, M.: Die Berufsschulsysteme in Tunesien und Deutschland – Ein systematischer Vergleich im Rahmen der wissenschaftlichen Evaluation des Projektes OpporTUNIty. Universität Duisburg-Essen (Campus Essen), Essen 2018.

Nr. 44: Peters, M. L.; Zelewski, S.: Adaption der Efficiency Analysis Technique With Input and Output Satisficing (EATWIOS) zur Berücksichtigung von unteren und oberen Satisfizierungsgrenzen. Universität Duisburg-Essen (Campus Essen), Essen 2018.

Nr. 45: Oeben, M.; Klumpp, M.: Export von Expertise im Bereich der Berufsausbildung – Erfolgsfaktoren und Hemmnisse für den Aufbau und Betrieb eines technischen Berufsschulzentrums in Tunesien im Forschungs- und Transferprojekt OpporTUNIty. Universität Duisburg-Essen (Campus Essen), Essen 2019.

Nr. 46: Oeben, M.; Klumpp, M.; Zelewski, S.: Internationaler Bildungstransfer – Internationaler Quervergleich als komparativer Ansatz zu Erfahrungen im Bildungstransfer in Richtung Tunesien. Universität Duisburg-Essen (Campus Essen), Essen 2019.

Nr. 47: Schagen, J. P.; Zelewski, S.; Heeb, T.: Erhebung und Analyse der Anforderungen an ein KI-Tool aus der Perspektive der betrieblichen Praxis – mit Fokus auf der Wiederverwendung von Erfahrungswissen im Bereich des betrieblichen Projektmanagements. Zugleich KI-LiveS-Projektbericht Nr. 1. Universität Duisburg-Essen (Campus Essen), Essen 2020.

Nr. 48: Schagen, J. P.; Zelewski, S.; Haselhoff, T.; Schmitz, S.; Heeb, T.: Überblick über potenzielle Quellen für Test- und Evaluierungsdaten eines KI-Labors im Rahmen des KI-LiveS-Projekts. Zugleich KI-LiveS-Projektbericht Nr. 2. Universität Duisburg-Essen (Campus Essen), Essen 2021.

Nr. 49: Fink, S.; Röhrig, K.; Heeb, T. (Mitarbeit Schagen, J. P.; Zelewski, S.): Konzipierung und Implementierung eines ontologiegestützten Case-based-Reasoning-Systems für die Wiederverwendung von projektbezogenem Erfahrungswissen. Zugleich KI-LiveS-Projektbericht Nr. 3. Universität Duisburg-Essen (Campus Essen), Essen 2021.

Nr. 50: Weber, L.; Heeb, T.; Sethupathy, G. (Mitarbeit Schagen, J. P.; Zelewski, S.): „Intelligente“ Wiederverwendung von Erfahrungswissen im betrieblichen Projektmanagement mithilfe von KI-Techniken bei sicherheitskritischen IT-Projekten mit Fokus auf PRINCE2 und Risikomanagement. Zugleich KI-LiveS-Projektbericht Nr. 4. Universität Duisburg-Essen (Campus Essen), Essen 2021.

Nr. 51: Allam, S.; Heeb, T.; Zelewski, S.: Konzipierung und Implementierung eines E-Learning-Moduls für ein ontologiegestütztes Case-based Reasoning Tool zur Unterstützung des Projektmanagements im Rahmen des KI-LiveS-Projekts. Zugleich KI-LiveS-Projektbericht Nr. 5. Universität Duisburg-Essen (Campus Essen), Essen 2021.

Nr. 52: Weber, L.; Allam, S.; Camgöz, A. (Mitarbeit Heeb, T.; Zelewski, S.): Erstellung eines E-Learning-Moduls für den Ontologie-Editor Protégé. Zugleich KI-LiveS-Projektbericht Nr. 6. Universität Duisburg-Essen (Campus Essen), Essen 2021.

Nr. 53: Fink, S.; Hauke, M.; Ye, B. (Mitarbeit Schagen, J. P.; Zelewski, S.): Erstellung und kritische Analyse von Use Cases für Anwendungen von KI-Tools im betrieblichen Projektmanagement – mit Fokussierung auf der „intelligenten“ Wiederverwendung von projektbezogenem Erfahrungswissen. Zugleich KI-LiveS-Projektbericht Nr. 7. Universität Duisburg-Essen (Campus Essen), Essen 2021.

Nr. 54: Schagen, T.; Heeb, T.; Zelewski, S. (Mitarbeit Schagen, J. P.): Entwicklung eines E-Learning-Moduls für ein ontologiegestütztes Case-based-Reasoning-System für das betriebliche Projektmanagement. Zugleich KI-LiveS-Projektbericht Nr. 8. Universität Duisburg-Essen (Campus Essen), Essen 2021.

Nr. 55: Zelewski, S.; Schagen, J. P.: Case-based Reasoning als KI-Technik zur „intelligenten“, computergestützten Wiederverwendung von Erfahrungswissen im Projektmanagement. Zugleich KI-LiveS-Projektbericht Nr. 9. Universität Duisburg-Essen (Campus Essen), Essen 2022.

Nr. 56: Bornemann, J.; Heeb, T.; Zelewski, S. (Mitarbeit Schagen, J. P.): Ein Vorgehensmodell zur Entwicklung ontologiegestützter Case-based-Reasoning-Systeme. Zugleich KI-LiveS-Projektbericht Nr. 10. Universität Duisburg-Essen (Campus Essen), Essen 2022.