Objektbasierte Generierung von Geschäftsprozessmodellen

Zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften (Dr.-Ing.)

von der KIT-Fakultät für Wirtschaftswissenschaften
des Karlsruher Instituts für Technologie (KIT)

genehmigte

Dissertation

von

M. Sc. Selina Schüler

Tag der mündlichen Prüfung: 08. Mai 2025
Erster Gutachter: Prof. Dr. Andreas Oberweis
Zweiter Gutachter: Prof. Dr. Sascha Alpers

Karlsruhe (2025)

Kurzfassung

Obwohl Geschäftsprozessmodellen eine große Bedeutung zugesprochen wird, werden viele Geschäftsprozesse in der Praxis nicht oder nur unzureichend modelliert. Gründe hierfür sind beispielsweise die zeit- und ressourcenintensive manuelle Modellierung sowie fehlende Modellierungsexpertise. Diese Probleme sollen durch die automatisierte Prozessmodellierung gelöst werden.

Die Ausführung eines Geschäftsprozesses basiert auf der kontinuierlichen Interaktion mit Objekten (wie beispielsweise Bestellungen, Rechnungen, Lieferscheine): Aktivitäten in Geschäftsprozessen erzeugen, lesen, bearbeiten und verbrauchen Objekte, die beispielsweise in Dokumenten oder Datenbanken gespeichert sind. Die Zustände dieser Objekte werden während der Ausführung der Geschäftsprozesse durch Aktivitäten verändert. Somit enthalten Objekte sowohl Aktivitäts- als auch Objektinformationen. Diese enge Verzahnung von Objekten und Geschäftsprozessen unterstreicht die Bedeutung einer integrierten Betrachtung von Objekt- und Kontrollflussaspekten. Dennoch wurde die Nutzung von Objekten für die automatisierte Prozessmodellierung bisher kaum untersucht. Daher war das Ziel der Arbeit, eine Methode zur automatisierten Prozessmodellierung zu entwickeln, die auf der Analyse von Objekten basiert. Grundlage dieser Methode sind Objekte, die in den Ausführungen des zu modellierenden Geschäftsprozesses erzeugt, gelesen, bearbeitet oder verbraucht wurden.

Die Methode umfasst ein zweistufiges Vorgehen. In der ersten Phase, der *Informationsextraktion*, werden die Objektinstanzen zunächst Prozessinstanzen zugeordnet. Anschließend werden die Objektinstanzen nach Objekttypen klassifiziert. Aus den inhaltlichen und zeitlichen Zusammenhängen der Objektinstanzen werden Aktivitätstypen abgeleitet. In der zweiten Phase, der *Transformation*, werden die Objekttypen und Aktivitätstypen in die Elemente eines höheren Petri-Netzes transformiert. Objekttypen werden durch Stellen, Aktivitätstypen durch Transitionen und der Kontrollfluss durch Kanten dargestellt. Zusätzlich werden auch Objektstrukturen und Objektmanipulation abgebildet.

Die praktische Anwendbarkeit der Methode wird durch einen Software-Prototyp demonstriert. Die Evaluation zeigt, dass die Methode für die gegebenen Szenarien sequenzielle, alternative und nebenläufige Ordnungsbeziehungen zwischen Aktivitätstypen erkennt. Durch die integrierte Darstellung von Objekt- und Kontrollfluss werden die Zusammenhänge kompakter in den Petri-Netzen abgebildet als bei traditionellen Process Discovery-Algorithmen. Die Berücksichtigung inhaltlicher Zusammenhänge führt im Vergleich zu den zeitlichen Zusammenhängen zur Erkennung von zusätzlichen Aktivitätstypen.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

7

7PMG	Seven Process Modeling Guidelines

A

AD	Aktivitätsdiagramme
AG	Anforderung an die Generierung
AI	Anforderungen an die Input-Daten
AM	Anforderung an das generierte Modell
AST	Abstract Syntax Tree

B

B/E-Netz	Bedingungs-/Ereignis-Netz
BIK	Beschriftungs- und Inschriftenkonventionen
BPM	Business Process Management
BPMN	Business Process Model and Notation

C

CNN	Convolutional Neural Network
CRM-System	Customer-Relationship-Management-System
CSV	Comma Separated Values

D

DMN	Decision Model and Notation
DQ	Datenqualität
DTD	Dokumenttypdefinition

E

EPK	Ereignisgesteuerte Prozesskette
ER-Modell	Entity-Relationship-Modell

G

GCN	Graph Convolutional Neural Network
GoM	Grundsätze ordnungsgemäßer Modellierung
GPM	Geschäftsprozessmanagements

H

HTML	Hypertext Markup Language

J

java	Java Source Code
js	JavaScript Source Code
JSON	JavaScript Object Notation

K

KDM-Modell	Knowledge Discovery Meta-Modell
KI	Künstliche Intelligenz

L

LDA	Latent Dirichlet Allocation
LR	Literaturrecherche

M

MVC	Model-View-Controller

N

NR/T-Netz	Nested Relation-/Transitions-Netz

O

OCR	Optical Character Recognition

P

PNML	Petri Net Markup Language
Pr/T-Netz	Prädikats-/Transitions-Netz
py	Python Source Code

R

RPA	Robotic Process Automation

S

S/T-Netz	Stellen-/Transitions-Netz
S-BPM	Subjektorientiertes Business Process Management
SBVR	Semantics of Business Vocabulary and Business Rules
SGML	Standard Generalized Markup Language
SIQ	***S**imple enough to be practically applicable, yet **I**ntegrates the most relevant insights from the BPM field, while it deals with **Q**uality*
StDN-Netz	Strukturiertes-Daten-Netz

U

UML	Unified Modeling Language

V

VDU	Visual Document Understanding

X

XML	Extensible Markup Language

1 Einleitung

Die *Modellierung von Geschäftsprozessen* ist ein zentrales Instrument des Geschäftsprozessmanagements in Unternehmen [Wesk19]. Sie bildet die Grundlage für vielfältige Aspekte: von der systematischen Dokumentation und Analyse bestehender Geschäftsprozesse über deren kontinuierliche Verbesserung bis hin zur operativen Überwachung und Steuerung. Durch die Modellierung können die Geschäftsprozesse in Unternehmen kontinuierlich analysiert sowie (bei Bedarf) angepasst und verbessert werden. Durch die Verbesserung von Geschäftsprozessen sollen beispielsweise die Wettbewerbsfähigkeit gesteigert, die Mitarbeiterzufriedenheit erhöht oder die Erwartungen von Kunden erfüllt werden [Wesk19, ŠkTr13, BBSG+16].

Geschäftsprozessmodelle werden zur Entscheidungsfindung verwendet. Sie können Grundlage für eine (Teil-)Automatisierung der Geschäftsprozesse sein und dienen auch zur Unterstützung der Entwicklung oder Anpassung von IT-Infrastrukturen, um serviceorientierte IT-Systeme zu gestalten [Agui04, RiMa21]. Diese vielfältigen Einsatzzwecke unterstreichen die Bedeutung von Geschäftsprozessmodellen. Trotz dieser Bedeutung werden Geschäftsprozesse in der Praxis nicht oder nur unzureichend modelliert [DLMR18, BBMB+11].

Diese Diskrepanz zwischen Bedeutung und Verwendung von Geschäftsprozessmodellen ist auf verschiedene Herausforderungen zurückzuführen, die in Abschnitt 1.1 näher betrachtet werden. Als möglicher Lösungsansatz wird in Abschnitt 1.2 die Automatisierung der Prozessmodellierung diskutiert. Abschnitt 1.3 gibt einen Überblick über den Aufbau der Arbeit.

1.1 Herausforderungen der Geschäftsprozessmodellierung

Ein Geschäftsprozessmodell bildet die relevanten Aspekte eines Geschäftsprozesses ab. Dabei soll die Komplexität eines Geschäftsprozesses durch geeignete Abstraktion vereinfacht dargestellt werden. Gleichzeitig sollen valide Aussagen über den Geschäftsprozess ermöglicht werden. Geschäftsprozessmodelle sind entweder zur abstrahierten Repräsentation der Realität, als Dokumentation von Prozesswissen oder als Vorlage für die Realität relevant [Kobl10].

Damit Geschäftsprozesse mittels Geschäftsprozessmodellen dokumentiert, analysiert, überwacht und gesteuert werden können, sind geeignete Beschreibungssprachen erforderlich. Für einfache Beschreibungsmodelle sind Darstellungsformen wie tabellarische und textuelle Beschreibungen oder grafische Abbildungen, mit beispielsweise Rechtecken und Pfeilen, möglich. Um jedoch einheitlich komplexere Geschäftsprozesse zu modellieren, sind diese unzureichend [Kobl10]. Daher sollte ein Geschäftsprozess mithilfe einer ausreichend ausdrucksstarken Modellierungssprache modelliert werden. Für die Abbildung der Aktivitätenreihenfolge haben sich verschiedene Modellierungssprachen etabliert: Petri-Netze [Reis13] (nach [Petr62]), Business Process Model and Notation (BPMN) [Obje13b], Ereignisgesteuerte Prozessketten (EPK) [KeNS92], UML-Aktivitätsdiagramme (UML-AD) [Obje17] oder Varianten dieser Sprachen.

Die Modellierung von Geschäftsprozessen erfordert sowohl fundierte Kenntnisse in der gewählten Modellierungssprache als auch umfassendes Domänenwissen über den abzubildenden Geschäftsprozess. Eine Herausforderung liegt in der Erfassung aller relevanten Informationen, da Geschäftsprozesse typischerweise verschiedene Akteure, Abteilungen und Systeme involvieren [Wesk19]. Diese Herausforderung wird durch die Komplexität von Geschäftsprozessen verstärkt, die durch das Zusammenspiel zahlreicher Objekte wie Bestellungen, Rechnungen und Lieferscheine charakterisiert sind. Aufgrund dieser Komplexität der Geschäftsprozesse und der unstrukturierten Informationsverwertung erfolgt die Modellierung bisher häufig manuell [BBMB+11, NaAK15].

Die manuelle Erstellung und Aktualisierung von Geschäftsprozessmodellen sind zeit- und ressourcenintensive Aufgaben. Empirische Studien zeigen, dass Geschäftsprozessmodelle häufig syntaktisch oder semantisch fehlerhaft sind [Mend08, Mend09, RSBR14]. Diese resultieren oft aus unzureichender Modellierungsexpertise oder fehlendem Domänenwissen. Sind Modellierer und Prozessexperte unterschiedliche Personen, können subjektive Interpretationen sowie Kommunikationsprobleme zwischen Prozessexperten und Modellierern zu fehlerhaften Modellen führen [VöSB13, MeRv10].

Fehlerhafte Geschäftsprozessmodelle können bei der Verwendung zu Problemen führen. Als Kommunikationsgrundlage können sie zu Missverständnissen zwischen verschiedenen Stakeholdern führen. Bei der Entscheidungsfindung besteht die Gefahr, dass auf Basis fehlerhafter Modelle falsche Schlussfolgerungen gezogen und Fehlentscheidungen getroffen werden. In der Prozessanalyse können fehlerhafte Modelle die Identifikation von Verbesserungspotenzialen erschweren und bei der automatisierten Prozessausführung zu Compliance-Verstößen oder ineffizienter Ressourcennutzung führen [Mood05].

Die automatisierte Prozessmodellierung ist ein vielversprechender Ansatz zur Überwindung dieser Herausforderungen. Sie ermöglicht nicht nur Zeit- und Qualitätsvorteile gegenüber der manuellen Modellierung [AaWe04], sondern auch eine systematische und

objektive Prozessabbildung [HSSS20]. Automatisch generierte Geschäftsprozessmodelle können bezüglich der syntaktischen und semantischen Korrektheit sowie semantischen Vollständigkeit manuell erstellte Modelle übertreffen. Allerdings bestehen auch bei der automatisierten Generierung Herausforderungen: Unzureichend validierte Algorithmen, mangelnde Durchsetzung von Qualitätsanforderungen oder qualitativ minderwertige Eingabedaten können die Modellqualität beeinträchtigen [HSSS20].

1.2 Automatisierte Prozessmodellierung

Die automatisierte Modellierung von Geschäftsprozessen ist seit über zwei Jahrzehnten Gegenstand aktiver Forschung [ScAl24]. Neue technologische Entwicklungen beispielsweise im Bereich generativer Transformermodelle [FoSc24] sowie die zunehmende Verfügbarkeit digitaler Prozessdaten [RVPv+17] eröffnen kontinuierlich neue Perspektiven für dieses Forschungsfeld.

Bestehende Ansätze zur automatisierten Modellierung basieren überwiegend auf Eventlogs, Prozessbeschreibungen oder Softwarecode. Diese Ansätze beschränken sich meist auf die Abbildung zeitlicher Zusammenhänge [ScAl24]. Process Discovery-Algorithmen analysieren die durchgeführten Prozessabläufe unter Verwendung der Eventlogs aus IT-Systemen, um mithilfe der Algorithmen ein Geschäftsprozessmodell (beispielsweise ein Petri Netz) zu erstellen [AaCa22]. Eine zentrale Herausforderung ist die Identifikation der am Geschäftsprozess beteiligten Systeme. Die Eventlogs dieser Systeme müssen extrahiert, aufbereitet und gegebenenfalls umformatiert werden. Dabei hängen Lokalisierung und Identifizierung der relevanten Eventlogs maßgeblich vom Wissen über Datenstrukturen und Prozessabläufe ab. Die Qualität der Eventlogs ist für das resultierende Geschäftsprozessmodell entscheidend. Sind die relevanten Reihenfolgen der Aktivitäten nicht im Eventlog repräsentiert, dann bildet ein generiertes Modell diese auch nicht ab [Aals22]. Wird jedoch durch umfangreiche Eventlogs sichergestellt, dass alle Ausführungspfade eines Geschäftsprozesses im Eventlog repräsentiert sind, können die generierten Modelle zu komplex sein [SuBa16b].

Für die Prozessmodellierung herangezogene Prozessinformationen beschränken sich jedoch nicht nur auf Eventlogs. Beispielsweise durch Expertenbefragungen, Workshops und Umfragen können Geschäftsprozesse reflektiert und natürlichsprachlich beschrieben werden. Bei der Verarbeitung natürlichsprachlicher Beschreibungen erschweren Mehrdeutigkeiten, unterschiedliche Detaillierungsgrade und unvollständige Informationen allerdings die korrekte automatisierte Modellierung [MLTA19, LWZZ10].

Um die Komplexität der automatisierten Prozessmodellierung zu reduzieren, arbeiten existierende Ansätze häufig mit Einschränkungen. Dafür wird beispielsweise der Input

auf sequenzielle Beschreibungen oder einfache Sätze beschränkt. Außerdem werden beim Output nur ausgewählte Modellelemente und grundlegende Kontrollflussmuster fokussiert. Viele der Ansätze beschränken sich auf einzelne Datenquellen, obwohl Geschäftsprozesse typischerweise in einer Vielzahl von verschiedenen Dokumenten beschrieben werden. Während solche Einschränkungen die Umsetzung vereinfachen können, fehlt oft eine fundierte Begründung für die getroffene Auswahl [ScAl24].

Weitere Prozessinformationen liegen in den zugehörigen Prozessartefakten wie beispielsweise Dokumenten in unterschiedlichen Formaten vor (Word-Dokument, E-Mail, PDF, XML-basierte Dokumente aus Systemen). Geschäftsprozesse bzw. Prozessschritte, die auf Dokumenten basieren, werden häufig von verschiedenen IT-Systemen unterstützt, was beispielsweise die Anwendung von Process Mining erschwert. Wird ein Bestellprozess betrachtet, umfasst dieser verschiedene Objekttypen wie Bestellungen, Wareneingänge und Rechnungen, die in vielfältigen Beziehungen zueinander stehen. Die Abbildung dieser Objektbeziehungen in einem Eventlog führt zu Informationslücken durch die notwendige Aggregation und zu einer verzerrten Prozesssicht durch die künstliche Trennung zusammengehöriger Objekte [PGEM+20].

Die Ausführung von Geschäftsprozessen basiert auf der kontinuierlichen Interaktion mit Objekten – Aktivitäten in Geschäftsprozessen lesen, verarbeiten und erzeugen Objekte, die beispielsweise in Dokumenten oder Datenbanken gespeichert sind. Die Zustände der Objekte werden während der Ausführung der Geschäftsprozesse durch deren Aktivitäten verändert. Diese Objekte enthalten somit sowohl Aktivitäts -, als auch Objektinformationen [ScAl24]. Daher können durch die Analyse der Zustandsänderungen der Objekte und der zeitlichen und inhaltlichen Zusammenhänge zwischen den Objekten sowohl Informationen über den Kontrollfluss als auch zu Entscheidungslogiken und Geschäftsregeln abgeleitet werden. Diese enge Verzahnung von Objekten und Geschäftsprozessen unterstreicht die Bedeutung einer integrierten Betrachtung von Objekt- und Kontrollflussaspekten. Wenn die Zustandsänderungen der Objekte korrekt im Geschäftsprozessmodell abgebildet und die Konsistenz zwischen Objekt- und Geschäftsprozessmodell sichergestellt ist, können Geschäftsprozesse effektiv analysiert und gesteuert werden [PGEM+20].

Objekte in Geschäftsprozessen bieten im Vergleich zu Eventlogs und Systemprotokollen zusätzliche Informationen über manuelle Aktivitäten. Dazu kann auf bereits vorliegende Daten zu Objekten aufgebaut werden, die im Rahmen der Prozessausführung erzeugt, gelesen, bearbeitet oder verbraucht werden, ohne von der Repräsentation einer Aktivität in einem IT-System abhängig zu sein [AaWe04]. Neue gesetzliche Regelungen wie die verpflichtende Nutzung elektronischer Rechnungen ab 2025 [BGB24] fördern zusätzlich die standardisierte digitale Verfügbarkeit von solchen Daten zu Objekten. Während die technische Extraktion von Eventlogs spezifisches System-Know-how erfordert, können

die Objekte in Geschäftsprozessen direkt von den Prozessbeteiligten bereitgestellt werden.

Eine Methode zur automatisierten Modellierung von Geschäftsprozessen aus Objekten, die sowohl inhaltliche als auch zeitliche Zusammenhänge berücksichtigt und in Geschäftsprozessmodellen integriert, existiert bislang nicht. Dabei gilt es, die Beziehungen zwischen den Objekten und den Aktivitäten zu bestimmen, die diese Objekte erzeugen und deren Zustände ändern.

Die vorliegende Arbeit setzt sich daher zum Ziel, diese Lücke zu schließen. Es soll eine Methode zur automatisierten Modellierung von Geschäftsprozessen in Form von höheren Petri-Netzen aus Objektinstanzen entwickelt werden. Höhere Petri-Netze ermöglichen die Integration der objektorientierten Perspektive in die kontrollflussorientierte Geschäftsprozessmodellierung. Das resultierende höhere Petri-Netz bildet damit nicht nur die zeitlichen Zusammenhänge zwischen den Objekten ab, sondern integriert durch Manipulationsanweisungen und Schaltbedingungen auch weiteres Prozesswissen in das Geschäftsprozessmodell. Höhere Petri-Netze wie XML- oder JSON-Netze ermöglichen die formale Abbildung von Objektflüssen sowie die präzise Modellierung von Operationen auf Objektstrukturen.

Eine automatisierte Prozessmodellierung, die direkt auf Prozessdaten und -artefakten basiert, ermöglicht eine zeitnahe Aktualisierung der Geschäftsprozessmodelle bei Änderungen am Geschäftsprozess. Dadurch wird eine flexiblere und schnellere Anpassung von Geschäftsprozessmodellen möglich. Dies kann nicht nur den manuellen Modellierungsaufwand reduzieren, sondern auch die Qualität und Wartbarkeit der resultierenden Geschäftsprozessmodelle verbessern und eine konsistente und nachvollziehbare Transformation von Prozessdaten in Modelle ermöglichen. Geschäftsprozessmodelle, die sowohl die Prozesslogik als auch die Datenmanipulationen formal spezifizieren, können zur Prozessautomatisierung verwendet werden [LeOb03]. Dabei könnten die extrahierten Objektstrukturen und -beziehungstypen direkt für die Konfiguration von IT-Systemen genutzt werden [LBWM+20]. Durch Eingaben wie JSON- oder XML-basierte Dateien soll die Methode eine schnelle und korrekte Anpassung der Geschäftsprozessmodelle an veränderte Geschäftsprozesse und ggf. Prozessautomatisierungen ermöglichen.

1.3 Aufbau der Arbeit

Ausgehend von den in diesem Kapitel beschriebenen Herausforderungen der Geschäftsprozessmodellierung sowie der Berücksichtigung der automatisierten Prozessmodellierung werden in Kapitel 2 zunächst die Grundlagen der Geschäftsprozessmodellierung betrachtet. Dafür werden die für die vorliegende Arbeit zentralen Begriffe *Modell*,

Geschäftsprozess und *Geschäftsprozessmodellierung* definiert und deren Zusammenhänge erläutert. Es werden kontrollfluss- und objektorientierten Modellierungsperspektiven betrachtet, da diese für die spätere Methodenentwicklung verwendet werden. Zusätzlich werden Qualitätsaspekte von Modellen diskutiert, um diese bei der zu entwickelnden Prozessmodellgenerierung zu gewährleisten.

Kapitel 3 analysiert dann die verschiedenen Datenquellen, die für eine automatisierte Prozessmodellierung genutzt werden können. Diese Analyse ist notwendig, um die Eignung der Daten hinsichtlich Struktur, Qualität und Informationsgehalt zu bewerten. Ein besonderer Schwerpunkt liegt auf der Analyse von Prozessartefakten wie Dokumenten und Datenbankeinträgen, die den Zustand und die Zustandsänderungen von Objekten (z.B. Bestellungen, Rechnungen, Lieferscheine) dokumentieren. Die Analyse der Objekte ermöglicht sowohl die Ableitung des Kontrollflusses als auch die Extraktion prozessrelevanter Objektinformationen. Zudem werden verschiedene Modellierungssprachen miteinander verglichen, um eine geeignete Notation für die Repräsentation der extrahierten Informationen zu identifizieren.

Darauf aufbauend werden in Kapitel 4 die Ergebnisse einer systematischen Literaturrecherche zu existierenden Ansätzen der Prozessmodellgenerierung betrachtet und analysiert. Die detaillierte Analyse von relevanten Publikationen dient dazu, den aktuellen Forschungsstand zu erfassen, Forschungslücken zu identifizieren und Möglichkeiten und Herausforderungen der bestehenden Ansätze einzubeziehen. Unter Einbezug dieser Ansätze soll (bei Bedarf) eine neue Lösung entworfen und in einem Software-Prototyp umgesetzt werden.

Nach den Richtlinien des Methoden-Engineerings [Brin96] werden in Kapitel 5 basierend auf den vorangegangenen Kapiteln sowie der Zielsetzung der Arbeit zunächst Anforderungen an die neue Methode der Prozessmodellgenerierung gesammelt. Dies umfasst sowohl Anforderungen an die Eingabedaten und das resultierende Modell als auch an den Generierungsprozess selbst. Diese Anforderungen bilden die Basis für die Methodenentwicklung. Außerdem soll anhand dieser Anforderungen in einem Software-Prototyp die Qualität der Methode überprüft werden. Zur systematischen Spezifizierung der Anforderungen wird ein Anwendungsfall der Prozessmodellgenerierung betrachtet.

In Kapitel 6 wird eine neue zweistufige Methode zur objektbasierten Generierung von Petri-Netzen beschrieben. In der Informationsextraktion werden die Objektinstanzen Objekttypen zugeordnet. Anschließend werden die Prozessinstanzen, die diese Objektinstanzen betreffen, definiert und aus inhaltlichen sowie zeitlichen Zusammenhängen zwischen Objektinstanzen innerhalb einer Prozessinstanz die Aktivitätstypen abgeleitet. Die anschließende Transformation überführt die extrahierten Informationen in Petri-Netze, wobei Objekttypen als Stellen und Aktivitätstypen als Transitionen abgebildet

werden. Zusätzlich werden Objektstrukturen für die Marken in den Stellen und Manipulationsregeln als Inschriften von Transitionen integriert.

Die Umsetzbarkeit der Methode wird durch die in Kapitel 7 beschriebene prototypische Implementierung demonstriert. Die modulare Architektur des Software-Prototyps ermöglicht künftige Erweiterungen.

Kapitel 8 evaluiert die entwickelte Methode anhand des Software-Prototyps. Dabei wird sowohl die Erfüllung der definierten Anforderungen überprüft als auch die korrekte Extraktion der Aktivitätstypen und deren Ordnungsbeziehung basierend auf unterschiedlichen Kontrollflussmustern gezeigt. Zusätzlich sollen auch die Grenzen der Methode sowie der Beitrag und das Potential erläutert werden.

Die Arbeit schließt in Kapitel 9 mit einer Zusammenfassung der Ergebnisse sowie einem Ausblick auf zukünftige Forschungsrichtungen, die sich aus den gewonnenen Erkenntnissen ergeben.

Der Aufbau der Arbeit ist in Abbildung 1-1 visualisiert.

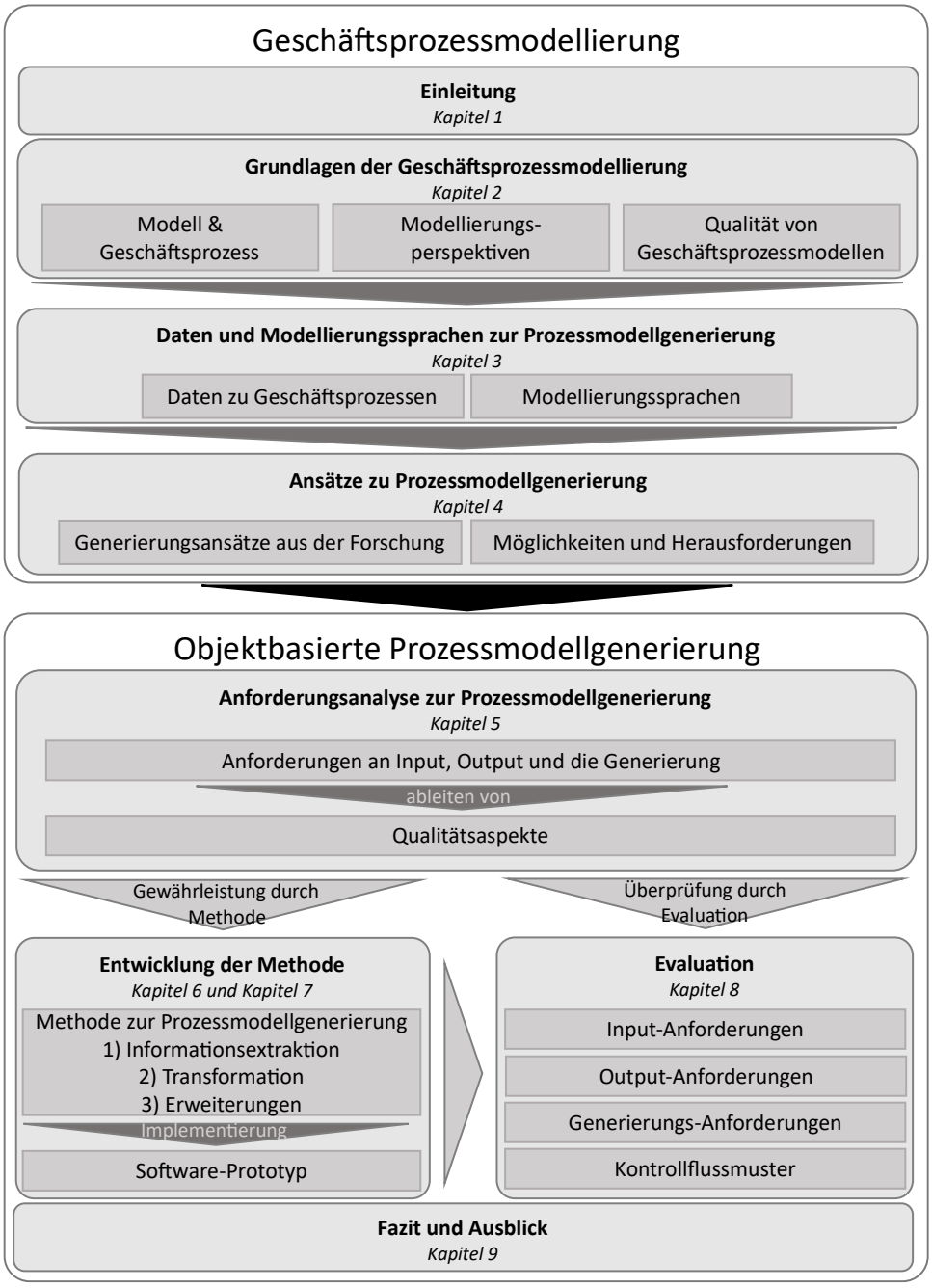


Abbildung 1-1: Aufbau der Arbeit

2 Modellierung von Geschäftsprozessen

In dieser Arbeit soll eine Methode zur automatisierten Erstellung von Geschäftsprozessmodellen (die *Geschäftsprozessmodellierung*) erarbeitet werden, die auf der Analyse von Objektinstanzen eines Geschäftsprozesses basiert. Daher wird im Folgenden zunächst auf die Modellierung im Kontext von Geschäftsprozessen eingegangen (siehe Abschnitt 2.1), dann werden Modellierungsperspektiven definiert, welche die Aspekte von Geschäftsprozessen fokussieren (siehe Abschnitt 2.2) und anschließend werden ausgewählte relevante Qualitätsaspekte für Geschäftsprozessmodelle diskutiert (siehe Abschnitt 2.3).

2.1 Modellierung im Kontext von Geschäftsprozessen

Um die Modellierung im Kontext von Geschäftsprozessen zu definieren, ist zunächst der allgemeine Begriff *Modellierung* relevant (siehe Abschnitt 2.1.1). Basierend darauf wird der Begriff anschließend im Kontext von Geschäftsprozessen weiter betrachtet. Dazu wird zunächst in Abschnitt 2.1.2 der Begriff *Geschäftsprozess* definiert. Anschließend wird in Abschnitt 2.1.3 auf die *Geschäftsprozessmodellierung* eingegangen.

2.1.1 Begriffsklärung: Modellierung

Die *Modellierung* wird in dieser Arbeit zunächst im Sinne der Allgemeinen Modelltheorie von [Stac73] betrachtet. [Stac73] beschreibt den Modellbegriff unter Verwendung von drei Hauptmerkmalen: Das Abbildungsmerkmal, das Verkürzungsmerkmal und das pragmatische Merkmal. Diese Merkmale werden folgend aus der Arbeit von [Stac73] zitiert und anschließend weiter beschrieben.

Abbildungsmerkmal: "*Modelle sind stets Modelle von etwas, nämlich Abbildungen, Repräsentationen natürlicher oder künstlicher Originale, die selbst wieder Modelle sein können*" [Stac73]. Ein Modell ist daher ein Abbild des Modelloriginals (*deskriptiv*) oder ein Vorbild für die Realität, wenn der betrachtete Modellierungsgegenstand noch nicht existiert (*präskriptiv*). Basierend auf deskriptiven Modellen können durch Analysen Aussagen über das Modelloriginal getroffen werden, um beispielsweise Verbesserungspotentiale zu identifizieren. Ebenfalls dokumentieren sie den Status quo zur Erhöhung der Transparenz oder zum Einhaltungsnachweis von Gesetzen, Richtlinien oder Verhaltens-

maßregeln. Präskriptive Modelle unterstützen die Spezifikation eines Systems über die Darstellung der für den Modellierungszweck relevanten Eigenschaften und Verhaltensmuster des Systems [Kobl10]. Oftmals wird unter einem *Modell* die Vorstellung/das Konzept über das Modelloriginal verstanden und eine konkrete Darstellung der Vorstellung als *Schema* bezeichnet [vgl. Ober96, Kasc99]. In anderen Literarturstellen werden die Begriffe *Modell* und *Schema* synonym verwendet [vgl. Kobl10, FrRü17]. Da in dieser Arbeit der allgemein gebräuchliche Begriff *Geschäftsprozessmodell* verwendet werden soll, wird von dieser Unterscheidung abgesehen.

Verkürzungsmerkmal: "*Modelle erfassen im allgemeinen nicht alle Attribute des durch sie repräsentierten Originals, sondern nur solche, die den jeweiligen Modellerschaffern und/oder Modellbenutzern relevant scheinen*" [Stac73]. Als *Modell* wird demnach eine Abstraktion eines Modelloriginals zur Darstellung der in einem bestimmten Kontext wichtigen Eigenschaften oder Verhaltensmuster eines Systems oder Prozesses bezeichnet [Ober96]. Die Eigenschaften oder Verhaltensmuster eines Systems als Ausschnitt eines Modelloriginals sollten in dem Modell widerspruchsfrei, in Bezug zum Kontext und dem damit verbundenen definierten Ziel vollständig, redundanzfrei und dem Original bzw. der Vorstellung vom Original entsprechend abgebildet werden [Thal22].

Pragmatisches Merkmal: "*Modelle sind ihren Originalen nicht per se eindeutig zugeordnet. Sie erfüllen ihre Ersetzungsfunktion a) für bestimmte – erkennende und/oder handelnde, modellbenutzende – Subjekte, b) innerhalb bestimmter Zeitintervalle und c) unter Einschränkung auf bestimmte gedankliche oder tatsächliche Operationen.*" [Stac73]. Die Zuordnung zwischen einem Modell und Modelloriginal bzw. dessen Vorstellung davon wird durch eine Abbildungsrelation beschrieben [Rose96]. Da Modelle für jemanden und für einen bestimmten Zeitpunkt erstellt werden, können unterschiedliche Modelle für den gleichen Zweck modelliert werden.

Des Weiteren sollten Modelle änderbar, auf unterschiedlichen Genauigkeitsebenen und aus unterschiedlichen Sichten darstellbar sein. Die Modellierungsperspektive eines Modellierungsvorhabens beschreibt den fokussierten Aspekt von dem Modelloriginal, der in einem Modell dargestellt werden soll. Eine mögliche Perspektive ist die kontrollflussorientierte Modellierungsperspektive (siehe auch Abschnitt 2.2). Zur Unterstützung der Erfüllung dieser Forderungen werden zur Modellierung Modellierungssprachen eingesetzt [Gada17]. Die Verwendung einer Modellierungssprache zur Beschreibung des Modelloriginals wird als *Modellierung* bezeichnet. Eine Modellierungssprache stellt ein Beschreibungsmodell zur Modellierung bereit [Ober96]. In dieser Arbeit soll eine Methode entwickelt werden, deren Ergebnis ein Modell (im Speziellen: ein Geschäftsprozessmodell (siehe Abschnitt 2.1.3)) in einer ausgewählten Modellierungssprache bereitstellt. Als Modellierungssprache kann eine künstliche Sprache verwendet werden. Künstliche Sprachen sind, im Gegensatz zu natürlichen Sprachen, für bestimmte Zwecke

(hier: die Beschreibung von einem Modell) neu entwickelt worden. Die *Sprache* wird als ein System von Zeichen, die der Vermittlung von Informationen dienen, definiert [SpAc80]. In Bezug auf Modelle sind besonders grafische Symbole oder textuelle Elemente als Zeichen von Bedeutung [OFFD14]. Die daraus entstehenden Zeichensysteme bzw. die zugrundeliegende Lehre von Zeichen werden im Rahmen der wissenschaftlichen Disziplin *Semiotik* betrachtet. Die Semiotik wird in die drei aufeinander aufbauenden Dimensionen Syntax, Semantik und Pragmatik unterteilt [LiSS94]. Diese lassen sich wie folgt definieren:

- Bei der *Syntax* wird zwischen der konkreten und abstrakten Syntax unterschieden. Die konkrete Syntax definiert das konkrete Aussehen der Elemente, die in einer Modellierungssprache verwendet werden. Die *abstrakte* Syntax beschreibt die Regeln zur Verknüpfung der Syntaxelemente des Modells. Die *konkrete* Syntax wird auch *Notation* genannt und die abstrakte Syntax Grammatik.
- Die *Semantik* legt die Bedeutung der Syntaxelemente fest. Sie kann außerdem die Syntax ergänzen, indem sie syntaktisch richtige Konstruktionen als semantisch unzulässig kennzeichnet.
- Die *Pragmatik* beschreibt das Verständnis der Zeichenverwendung und -bedeutung durch den Anwendenden der Sprache [ReLR11].

Je nachdem ob Syntax und/oder Semantik einer Modellierungssprache in einer natürlichen oder eindeutigen, künstlichen Sprache definiert werden, handelt es sich um eine informale, semi-formale oder formale Modellierungssprache [Pati06]. Eine formale Modellierungssprache besteht aus einer eindeutig festgelegten Menge von Syntaxelementen, deren Semantik ebenfalls eindeutig festgelegt ist. Semi-formale Modellierungssprachen sind hinsichtlich ihrer Syntax präzise beschrieben, jedoch nicht hinsichtlich ihrer Semantik. Informale Modellierungssprachen weisen weder eine eindeutige Syntax noch eine präzise Semantik auf. Die Syntax einer Modellierungssprache kann durch ein formales Metamodell definiert werden. Die Semantik kann durch das Festlegen der Ausführung einzelner Modellierungselemente oder ihrer Kombinationen präzisiert werden. Modelle, die durch eine semi-formale oder formale Modellierungssprache beschrieben wurden, können auf syntaktische und/oder semantische Richtigkeit anhand der festgelegten Regeln und Ausführungsinterpretationen dieser Modellierungssprache überprüft werden [OFFD14].

2.1.2 Begriffsklärung: Geschäftsprozess

Wird die Modellierung im Kontext von Geschäftsprozessen betrachtet, bedarf es verschiedener Anforderungen an eine Modellierungssprache, um die relevanten Eigenschaften oder Verhaltensmuster des Modelloriginals, den *Geschäftsprozess*,

entsprechend abbilden zu können. Diese Anforderungen sind zunächst insbesondere von dem Gegenstand, der modelliert werden soll, beeinflusst. Daher wird zunächst der Begriff *Geschäftsprozess* definiert. Anschließend werden dazu *Geschäftsprozesstyp* und *Prozessinstanz* betrachtet. Entsprechend dem Modellierungszweck wird festgelegt, welche Eigenschaften eines Geschäftsprozesses abgebildet werden sollen (Abschnitt 2.1.3).

[Dave93] beschreibt einen Geschäftsprozess als eine Reihe von logisch zusammenhängenden Aktivitäten, um ein definiertes Output-Objekt für einen bestimmten Kunden oder Markt zu erreichen. Die Aktivitäten unterliegen einer definierten Ordnung über Zeit und Ort hinweg, mit einem Anfang, einem Ende und klar identifizierten Input- und Output-Objekten [Dave93]. Im Vergleich zu Davenport, definiert [Ober96] die Aktivitäten weiter und verallgemeinert das Ziel: "*Ein betrieblicher Ablauf ist eine Menge von manuellen, teil-automatisierten oder automatisierten Aktivitäten, die in einem Betrieb nach bestimmten Regeln, auf ein bestimmtes Ziel hin ausgeführt werden. Die Aktivitäten hängen über betroffene Personen, Maschinen, Dokumente, Betriebsmittel u.ä. miteinander zusammen.*" [Ober96]. Ein Geschäftsprozess wird dabei begrifflich mit einem betrieblichen Ablauf gleichgesetzt. Geschäftsprozesse sind in ein organisatorisches Umfeld eingebettet und unterschiedliche Geschäftsprozesse können "*durch Datenaustausch, Kommunikation, gemeinsame Aufgabenträger oder gemeinsam genutzte Ressourcen vielfältig miteinander verbunden"* sein [Ober96]. Nach [HaCh93] wird durch einen Geschäftsprozess eine wertschöpfende Tätigkeit ausgeführt und mittels Input-Objekten Output-Objekte erzeugt [HaCh93].

Je nach Detailierungsgrad der Regeln bezüglich der Reihenfolge der Aktivitäten, gibt es strukturierte, teilstrukturierte oder unstrukturierte Abläufe. Unstrukturierte Abläufe können aufgrund der nicht eindeutig festgelegten Reihenfolge von Aktivitäten nur begrenzt geplant werden. [Ober96] definiert zusätzlich zu den Input-Objekten, die einem Ablauf von außen zugefügt werden, und den Output-Objekten, die ein Ablauf für die Umgebung erzeugt, die internen Objekte, die innerhalb desselben Ablaufs von Aktivitäten erzeugt und verbraucht werden. Aktivitäten werden durch *personelle* bzw. *nichtpersonelle Aufgabenträger* ausgeführt. Die Erfüllung einer Aufgabe erfolgt durch Ausführen einer oder mehrerer Aktivitäten. Wenn mindestens zwei Aufgabenträger an der Ausführung von Prozessaktivitäten beteiligt sind, ist es ein *kooperativer* oder *arbeitsteiliger Geschäftsprozess*. Bei einem *verteilten Geschäftsprozess* werden die Aktivitäten an geografisch unterschiedlichen Orten ausgeführt [Ober96].

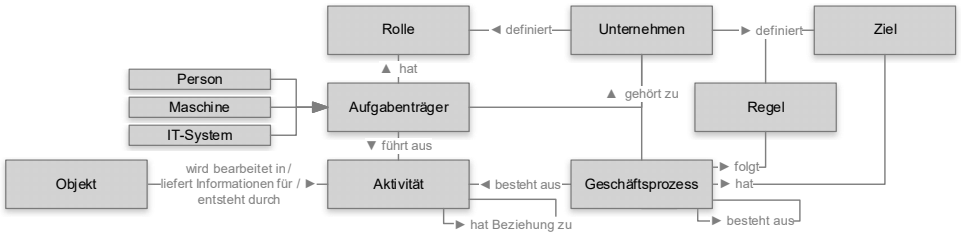
Nach [Booc00] sind Objekte greifbare und/oder sichtbare Dinge oder intellektuell wahrnehmbare Einheiten, auf die sich menschliches Denken und Handeln bezieht. Im Kontext von Geschäftsprozessen können Objekte sowohl physische Entitäten (wie Dokumente oder Produkte) als auch abstrakte Konzepte (wie Bestellungen oder Verträge) repräsentie-

ren. Nach [Ober96] kann des Weiteren zwischen statischen (das heißt objektbezogenen) und dynamischen (d.h. aktivitätsbezogenen) Aspekten differenziert werden. Als Objekte nennt er "*Personen, Belege, Formulare, Betriebsmittel, Maschinen u.ä., die über Eigenschaften wie Alter, Größe, Gewicht, Fähigkeiten etc. verfügen*" [Ober96]. Der Zustand einer Objektinstanz beschreibt die Gesamtheit aller Eigenschaften einer Objektinstanz zu einem bestimmten Zeitpunkt. Durch die Änderung solcher Eigenschaften mittels Aktivitäten werden dynamische Aspekte miteinbezogen[1]. Somit werden Objekte in dieser Arbeit definiert mit [Ober96, Booc00]:

1. **Zustand**: Objekte haben beobachtbare Eigenschaften, durch die sie beschrieben werden. Die Gesamtheit aller Eigenschaften ist der Zustand eines Objektes. Der Zustand eines Objekts umfasst die *Attribute* und die aktuellen *Werte* dieser Attribute zu einem bestimmten Zeitpunkt.

2. **Verhalten**: Das Verhalten eines Objekts wird differenziert in: *Zustandsveränderungen* mittels Aktivitäten und *Aktionen*, bei denen das Objekt selbst andere Objekte durch Nachrichten zu Operationen veranlasst. In dieser Arbeit werden die *Zustandsveränderungen* betrachtet. Eine Zustandsveränderung ist eine Veränderung eines Objekts beziehungsweise die Modifikation der Attribute bzw. Werte. Der aktuelle Zustand eines Objekts repräsentiert dabei das kumulierte Ergebnis seines bisherigen Verhaltens.

In Abbildung 2-1 werden die ausgewählten betrachteten Eigenschaften eines Geschäftsprozesses sowie Zusammenhänge zwischen diesen dargestellt, die potenziell in einem Modell abgebildet werden könnten.



in Anlehnung an [JaLi96, Ober96]

Abbildung 2-1: Eigenschaften eines Geschäftsprozesses sowie deren Zusammenhänge

[1] Jedes Objekt besitzt eine eindeutige Identität, die es von allen anderen Objekten unterscheidbar macht [Booc00].

Wird der Gegenstand *Geschäftsprozess* betrachtet, wird zwischen der Typ- und Instanz-Ebene eines Geschäftsprozesses unterschieden[2]. Als Geschäftsprozesstyp wird der allgemeinen Rahmen eines Geschäftsprozesses bezeichnet, der als Beschreibung der Aktivitäten und Objekte eines Ablaufs sowie der Zusammenhänge zwischen diesen gegeben ist, ohne jedoch auf konkrete Objekte und Aktivitäten einzugehen. Die Ausprägung eines Geschäftsprozesstyps, in der jeder Aktivität konkrete Objekte zugeordnet sind, die diese Aktivität eindeutig identifizieren, wird als Prozessinstanz bezeichnet. Eine Prozessinstanz besteht daher aus Aktivitätsinstanzen. Zu einem Geschäftsprozesstyp können zu einem Zeitpunkt mehrere konkrete Prozessinstanzen vorliegen [Ober96]. In Abbildung 2-2 wird die Unterscheidung zwischen dem Begriff sowie der Typ- und Instanz-Ebene verdeutlicht, wobei für den Begriff eine reduzierte Darstellung von Abbildung 2-1 verwendet wurde.

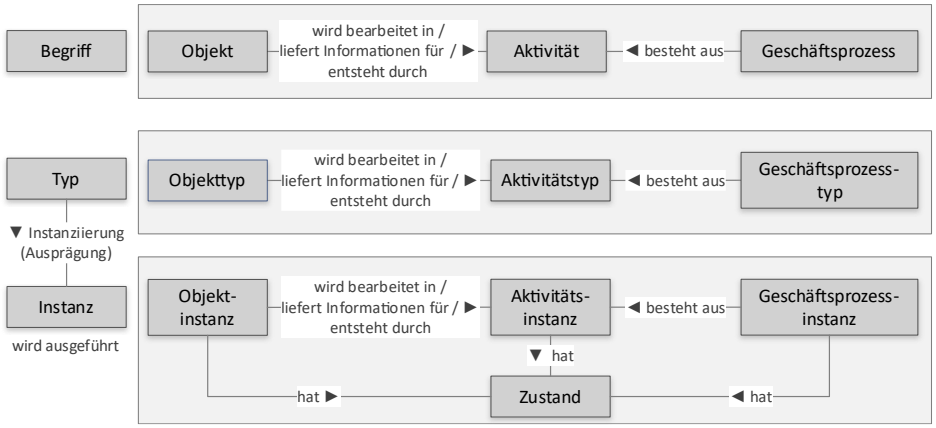


Abbildung 2-2: Begriff, Typ- und Instanz-Ebene eines Geschäftsprozesses

Der Zusammenhang zwischen Objekten und Aktivitäten ist in Abbildung 2-3 auf Instanz-Ebene dargestellt. Da eine Aktivität durch einen Aufgabenträger ausgeführt werden kann und Teil eines Geschäftsprozesses ist, der ein definiertes Ziel hat und definierten Regeln folgt, (siehe Abbildung 2-1), gilt dies ebenfalls für eine Aktivitätsinstanz und wird mit gestrichelten Kanten dargestellt.

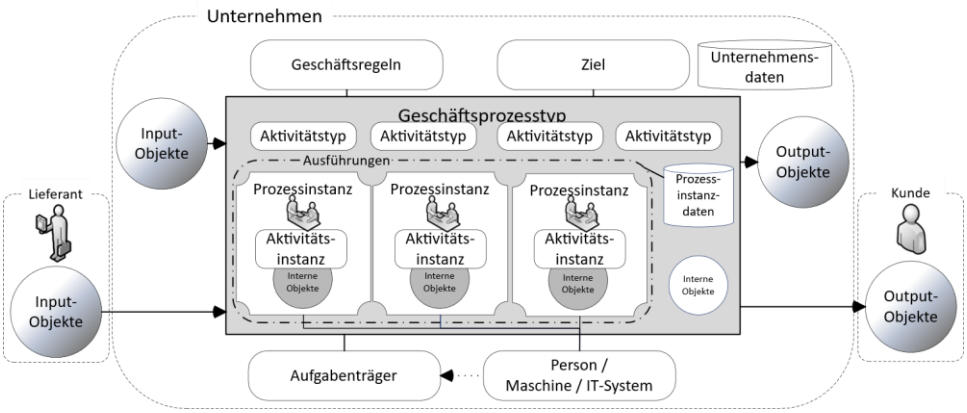
[2] In der Arbeit werden i. d. R. die Begriffe *Geschäftsprozess*, *Aktivität* und *Objekt* verwendet, wenn sich Aussagen dazu sowohl auf den Typ als auch die Instanz (und damit auf das Konzept des Begriffes) beziehen.

in Anlehnung an [Ober96, Joch09]

Abbildung 2-3: Zusammenhang Objekt und Aktivität

Einer Prozessinstanz sowie den Aktivitätsinstanzen innerhalb dieser Prozessinstanz können ebenfalls Zustände zugeordnet werden. Mögliche Zustände sind nach [Ober96] "*vor-Ausführung*", "*aktiv*", "*wartend*", "*gestört*", "*abgebrochen*" und "*regulär beendet*". Zustandsübergänge werden durch ein oder mehrere menschliche oder maschinelle Aufgabenträger als Kontrollinstanz(en) einer Aktivität bzw. eines Geschäftsprozesses ausgelöst. Der Zustand einer Prozessinstanz kann durch die Zustände der Aktivitätsinstanzen impliziert werden [Ober96]. Ebenso können Zeitaspekte für Geschäftsprozesse berücksichtigt werden, beispielsweise durch Zeitdauern, zeitbezogene Regeln, Zeitpunkte oder objektbezogene Lebenszeiten [Ober90]. Ein digitales Modell, das die Eigenschaften, Funktionsweise und Leistung eines physischen Objekts oder Systems (beispielsweise einer Prozessinstanz) abbildet, wird als *Digitaler Schatten* [StFL19] bezeichnet. Bildet das Modell die Eigenschaften, Funktionsweise und Leistung des physischen Objekts oder Systems in Echtzeit ab, wird es als *Digitaler Zwilling* [StFL19] bezeichnet.

Die Betrachtung eines *Geschäftsprozesstyps* sowie seiner konkreten Ausprägungen (*Prozessinstanzen*) ist in Abbildung 2-4 zusammengefasst.



in Anlehnung an [Ober96, Dave93, Wesk19]

Abbildung 2-4: Geschäftsprozesstyp und -instanz

Die genannten Input-, internen und Output-Objekte in Abbildung 2-4 stellen jeweils das Konzept des Begriffs *Objekt* dar, da alle in der Abbildung dargestellten Objekte sowohl als Typ als auch als Instanz definiert werden könnten. In Abbildung 2-4 wird jedoch differenziert zwischen Objekten, die im direkten Zusammenhang mit einer Prozessinstanz stehen (grau gefärbt – im Folgenden *Prozessinstanzobjekte*) und Objekten, die unabhängig von einer Prozessinstanz sind (weiß gefärbt – im Folgenden *prozessinstanzunabhängige Objekte*). Die prozessinstanzunabhängigen Objekte können über das Unternehmen, in das der Geschäftsprozess eingebettet ist, mit einer (aber auch weiteren) Prozessinstanz(en) in Beziehung gesetzt werden. Da Input- und Output-Objekte Prozessinstanzobjekte oder prozessinstanzunabhängige Objekte sein können, sind diese mit Farbverlauf abgebildet. Folgendes Beispiel verdeutlicht diese Differenzierung: Während Einträge im ERP-System wie Materialinformationen unabhängig von einer einzelnen Prozessinstanz sind, steht die Rechnung für ein in einer Prozessinstanz produziertes Produkt im direkten Bezug zu dieser Prozessinstanz. Da einige Daten nicht konkreten Objektinstanzen zugeordnet werden können, wurden zusätzlich die Unternehmens- und Prozessinstanzdaten ergänzt. Dazu zählen beispielsweise Logs zu Datenbank-Operationen als Unternehmensdaten und Logs zu IT-Systemen, die Aktivitätsinstanzen in den Instanzen zu einem Geschäftsprozesstyp ausführen, zu Prozessinstanzdaten.

Folgender Text beschreibt einen Geschäftsprozesstyp, der die in Abbildung 2-1 dargestellten Eigenschafen beispielhaft verdeutlicht:

Ein Unternehmen, das Smartphones produziert, definiert seine Ziele, Rollen und Geschäftsregeln folgendermaßen: Ein Ziel des Unternehmens ist, die Smartphones kostengünstig zu produzieren und gewinnmaximierend zu verkaufen. Daher ist in diesem Unternehmen ein Geschäftsprozess zum Verkauf der Smartphones eingebettet. Der Geschäftsprozesstyp *Smartphone-Verkauf* besteht aus Teilprozessen (entsprechen wiederum Geschäftsprozesstypen) und Aktivitätstypen, wie *Kunde anrufen*, *Vertrag aufsetzen*, *Rechnung erstellen*, *Rechnung drucken, Rechnung senden, Lieferschein drucken, Smartphone liefern*, *Zahlungseingang erhalten*. Die Instanzen dieser Aktivitätstypen werden von Aufgabenträgern ausgeführt. Beispielsweise wird die Aktivitätsinstanz *Kunde anrufen* für die Kundin Frau Musterfrau in der Prozessinstanz *Smartphone-Verkauf* an Frau Musterfrau von dem Aufgabenträger Herr Mustermann in der Rolle *Vertriebsmitarbeiter* ausgeführt. Dieser Aufgabenträger gehört damit zu dem Unternehmen und hat eine definierte Rolle. Während die in einem Customer-Relationship-Management-System (CRM-System) gespeicherten Kundendaten Objektinstanzen von dem prozessinstanzunabhängigen Objekttyp *Kunde* darstellen, ist der Objekttyp *Vertrag*, der gegebenenfalls instanziiert wird, ein Prozessinstanzobjekt. Der Geschäftsprozesstyp *Smartphone-Verkauf* wird durch Teilprozesse weiter unterteilt. Diese Teilprozesse sind durch Aktivitätstypen weiter beschrieben. Einzelne Aktivitätstypen des Unternehmens folgen einer oder mehreren Geschäftsregeln (z. B. bei einer Verkaufshöhe von mehr als 100.000 € wird bei dem

Aktivitätstyp *Rechnung erstellen* ein Rabatt in Höhe von 5% gewährt). Für die Ausführung der Aktivitätsinstanzen werden durch Beziehungen zwischen diesen Aktivitätstypen Ablaufreihenfolgen definiert: Während die Rechnung und der Lieferschein nebenläufig gedruckt werden können, muss zuvor der Vertrag gestaltet und für das Drucken der Rechnung die Rechnung erstellt worden sein. Wird innerhalb eines instanziierten, aber noch nicht beendeten Geschäftsprozesstyps der Vertrag aufgesetzt, haben sowohl die Aktivitätsinstanz *Vertrag aufsetzen* als auch die Prozessinstanz als Ganze den Zustand *aktiv*. Die Objekte des Geschäftsprozesstyps sind unter anderem die Smartphones, Rechnungen und Einträge in den IT-Systemen. Diese Objekte werden im Rahmen einer Aktivität bearbeitet. Beispielsweise wird der Zustand der Objektinstanz *Rechnung von Frau Musterfrau* in der Aktivitätsinstanz des Aktivitätstyp *Rechnung erstellen* verändert, indem die Rechnungsempfängerin und der Gesamtpreis eingetragen werden.

In dieser Arbeit wird die folgende Definition 2-1 für Geschäftsprozesse (in Anlehnung an [Ober96], [Dave93] und [Wesk19]) zugrunde gelegt:

Definition 2-1: Geschäftsprozess

"Ein [Geschäftsprozess] ist eine Menge von manuellen, teil-automatisierten oder automatisierten Aktivitäten, die in einem Betrieb nach bestimmten Regeln, auf ein bestimmtes Ziel hin ausgeführt werden. Die Aktivitäten hängen über betroffene Personen, Maschinen, Dokumente u. ä. miteinander zusammen."

[Ober96]

Ein Geschäftsprozess kann nach weiteren Charakteristika spezifiziert werden:

- nach den Prozessinstanz- und prozessinstanzunabhängigen Objekten, die in einem Geschäftsprozess als Input-Objekt (von außen zugefügt), Output-Objekt (für Umgebung erzeugt) oder internes Objekt (innerhalb eines Geschäftsprozesses erzeugt und verbraucht) auftreten
- nach den Zustandsdefinitionen der Instanzen von Objekten, Aktivitäten und des Geschäftsprozess selbst
- nach den Aufgabenträgern, die Aktivitäten ausführen und über eine Rolle zugehörig zu einem Unternehmen sind

Das Geschäftsprozessmanagement (GPM, englisch: Business Process Management (BPM)) umfasst Methoden und Technologien zur kontinuierlichen Anpassung und Verbesserung von Geschäftsprozessen. Dies erfolgt durch Identifikation, Priorisierung, Dokumentation, Analyse und Überwachung der Geschäftsprozesse [Wesk19]. Das GPM unterscheidet zwischen existierenden und neu einzuführenden Geschäftsprozessen. Ein existierender Geschäftsprozess wird durch das GPM zuerst dokumentiert. Diese Dokumentation kann sowohl die einzelnen auszuführenden Aufgaben als auch die beteiligten Rollen und IT-Systeme enthalten. Anhand dieser Dokumentation können ein Ist-Geschäftsprozess analysiert und Schwachstellen identifiziert werden. Das Ergebnis dieser Analyse ist Grundlage für die Neugestaltung des Geschäftsprozesses, der anschließend

umgesetzt werden soll. Im Unterschied dazu wird bei einem neu einzuführenden Geschäftsprozess mit der Geschäftsprozesskonzeption begonnen. Dieser entworfene Geschäftsprozess wird danach in einen realen Geschäftsprozess umgesetzt. Das Ergebnis der Prozessumsetzung ist ein IST-Geschäftsprozess, der dem zuvor konzipierten SOLL-Geschäftsprozess entsprechen soll. Die laufenden dokumentierten Geschäftsprozesse werden anschließend kontinuierlich überwacht, um mögliche Schwachstellen frühzeitig zu erkennen und Maßnahmen zu ihrer Behebung ergreifen zu können [AnDQ22, FrRü17]. In dieser Arbeit liegt der Fokus auf bereits existierenden, aber ggf. noch nicht (vollständig) modellierten Geschäftsprozessen. Hierdurch können Geschäftsprozessmodelle auf aktuelle Prozessinstanz- und Unternehmensdaten aufbauen und dadurch weitere Informationen miteinbeziehen.

Das GPM ist demnach der Steuerungsprozess von Analyse, Design, Modellierung, Implementierung, Ausführung und Management von Geschäftsprozessen [FrRü17]. [BePV12] sehen als oberstes Ziel des GPM die Unterstützung bei der Erreichung der Unternehmensziele. [ÖzDr17] hingegen bezeichnen die Standardisierung der Aktivitäten im Unternehmen als übergeordnetes Ziel, um eine Transparenzerhöhung, Kosteneinsparung, Harmonisierung der Prozesslandschaft, Verbesserung der Qualität und Steigerung der Kundenzufriedenheit anzustreben. [BeMW09] sehen den Nutzen vom GPM darin, die eigenen Abläufe effektiv an die Markterfordernisse anpassen zu können. Nach diesen Ansichten verfolgt das GPM strategische und operative Ziele. Einzelne Geschäftsprozesse zu untersuchen und zu verbessern, zählt zu den operativen Zielen. Geht die Betrachtung über die einzelnen Geschäftsprozesse hinaus, um die festgelegten Ziele eines Unternehmens zu erreichen, so ist dies den strategischen Zielen zuzuordnen. Hierfür wird die Gestaltung jedes einzelnen und das Zusammenspiel aller Geschäftsprozesse bewusst gesteuert und geführt [VoHa18]. Das GPM betont damit, dass die einzelnen Geschäftsprozesse in Unternehmen nicht isoliert, sondern als wertsteigernde Folge von Funktionen und Aufgaben, die über mehrere organisatorischen Einheiten verteilt sein können, zu betrachten sind [ScSe13]. Dies wurde bereits durch Definition 2-1 verdeutlicht.

Das GPM adressiert somit Geschäftsprozesse, die mithilfe von Modellen dokumentiert, analysiert und gestaltet werden können. Zusätzlich können Geschäftsprozessmodelle zur Ressourcen-Einsatzplanung, zur Unterstützung der Ablaufausführung durch Workflow-Managementsysteme und zur Überwachung und Steuerung von Abläufen herangezogen werden [vgl. Ober96, Wesk19, ScSe13]. Ein Workflow-Managementsystem ist ein System, das die Ausführung von Arbeitsabläufen durch den Einsatz von Software definiert, verwaltet und deren Ausführungsreihenfolge durch eine Computerdarstellung der Arbeitslogik (einem Geschäftsprozessmodell) gesteuert wird [Work96].

2.1.3 Begriffsklärung: Geschäftsprozessmodellierung

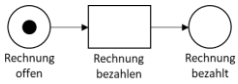
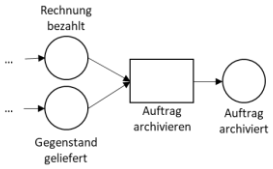
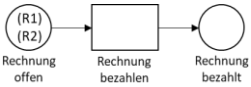
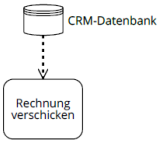
Grundlage des GPM ist ein Geschäftsprozessmodell. Geschäftsprozessmodelle werden von [Foth10] als konzeptuelle Abbildungen der Reihenfolge von Aktivitäten definiert. Die Reihenfolge von Aktivitäten ergibt sich durch die Beziehungen zwischen den Aktivitäten innerhalb eines Geschäftsprozesses. Diese Beziehungen können kausale Zusammenhänge oder organisatorische Vorgaben sein. [Ober96] schreibt, dass "[...] *zwei Aktivitäten, die sich kausal bedingen, [...] in einer bestimmten Reihenfolge stattfinden [müssen]. Kausale Zusammenhänge zwischen den Aktivitäten eines Ablaufs können beispielsweise durch das Fließen von Belegen, Formularen, Akten(ordnern) oder Vorgangsmappen gegeben sein. Aktivitäten können sich außerdem innerhalb eines Ablaufs gegenseitig ausschließen - z. B. wenn sie auf dieselbe Ressource zurückgreifen. Schließlich können Aktivitäten auch kausal unabhängig voneinander ("nebenläufig") stattfinden. Von der Nebenläufigkeit zu unterscheiden ist die (zeitliche) Parallelität zweier Aktivitäten im Sinne von deren Gleichzeitigkeit"* [Ober96]. Dies entspricht in dem Beispielprozess in Abschnitt 2.1.2 dem kausalen Zusammenhang, dass eine Rechnung erst versendet werden kann, nachdem sie erstellt wurde, und der kausalen Unabhängigkeit zwischen dem Drucken der Rechnung und dem Drucken des Lieferscheins, da keine kausal implizierte Beziehung zwischen ihnen besteht [Ober90]. Eine organisatorische Vorgabe, die Beziehungen zwischen den Aktivitäten definiert, kann in einem Geschäftsprozess beispielsweise durch ein physikalisches Signal zwischen zwei Aktivitäten umgesetzt werden, indem das Beenden einer Aktivität dieses Signal auslöst, das dann eine Vorbedingung für den Start einer anderen Aktivität ist. Dies entspricht in dem Beispielprozess in Abschnitt 2.1.2 einer Einschränkung in einem Abrechnungs-System, dass eine Rechnung im System erst erstellt werden kann, wenn ein Vertrag hinterlegt und von einem Freigabeverantwortlichen bestätigt wurde.

Da ein Geschäftsprozessmodell alle relevanten Aspekte eines Geschäftsprozesses unter Verwendung eines Beschreibungsmodells abbilden soll, können in einem Geschäftsprozessmodell neben Reihenfolge von Aktivitäten auch weitere Aspekte wie Rollenzuweisungen oder Kommunikationsflüsse definiert werden. Für einfache Beschreibungsmodelle sind Methoden wie tabellarische und textuelle Beschreibungen oder grafische Abbildungen, mit beispielsweise Rechtecken und Pfeilen, möglich. Um jedoch einheitlich komplexere Geschäftsprozesse zu modellieren, sind diese unzureichend [Kobl10]. Daher sollte ein Geschäftsprozess mithilfe einer ausreichend ausdrucksstarken Modellierungssprache modelliert werden (vgl. Abschnitt 2.1.1). Liegt der Fokus auf der Abbildung der Reihenfolge von Aktivitäten, werden häufig Petri-Netze [Reis13] (nach [Petr62]), Ereignisgesteuerte Prozessketten (EPK) [KeNS92], Business Process Model and Notation (BPMN) [Obje13b], UML-Aktivitätsdiagramme (UML-AD) [Obje17] oder Erweiterungen dieser Sprachen als Modellierungssprache verwendet. Für spezielle Modellierungs-

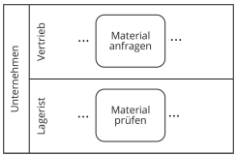
zwecke stehen weitere unterschiedliche Modellierungssprachen zur Verfügung, beispielsweise für Datenmodelle werden Entity-Relationship-Schema [Chen76] oder UML-Klassendiagramme verwendet oder für Zustandsdiagramme UML-Zustandsdiagramme herangezogen.

Beispiele für komplexere Sachverhalte insbesondere bezüglich der Reihenfolge von Aktivitäten innerhalb eines Geschäftsprozesses, die sich durch Modellierungssprachen in einem Geschäftsprozessmodellausschnitt abbilden lassen, sind in Tabelle 2-1 beschrieben. Spalte 1 beschreibt dabei den abzubildenden Sachverhalt, der durch eine Modellierungssprache abgebildet werden kann, Spalte 2 erläutert einen Ausschnitt eines Prozessbeispiels, der diesen Sachverhalt beinhaltet und Spalte 3 zeigt je ein ausgewähltes Beispiel, das den Sachverhalt in einer Modellierungssprache abbildet. Diese Beispiele sollen nicht weiter erklärt werden, sondern lediglich die grundsätzliche Umsetzungsmöglichkeit zeigen. Auf die konkrete Modellierung wird in Abschnitt 3.2 eingegangen.

Tabelle 2-1: Prozessausschnitte und deren Abbildung durch eine Modellierungssprache[3]

Sachverhalt	Prozessausschnitt	Abbildung
Aktivitäten, Zustände und Zustandsänderung	Durch die Aktivität *Rechnung bezahlen*, ändert sich der Zustand der Rechnung von offen in bezahlt.	
Vorbedingungen, die vor Erledigung einer Aktivität zutreffen müssen	Erst wenn sowohl die Rechnung bezahlt als auch der bestellte Gegenstand geliefert wurde, kann ein Auftrag archiviert werden.	
Unterscheidung von Objekten innerhalb eines Geschäftsprozesses	Wenn eine bestimmte Rechnung bezahlt wird, verändert sich nur deren Zustand in *bezahlt*.	
notwendige Dateneingaben	Für das Verschicken von Rechnungen werden die Daten aus dem CRM-System benötigt.	

[3] Verwendete Modellierungssprachen: (Höhere) Petri-Netze, BPMN

Sachverhalt	Prozessausschnitt	Abbildung
verantwortliche Organisationseinheiten	Während die Materialanfragen von dem Vertrieb erfolgen, ist für die Prüfung der Materialien der Lagerist zuständig.	

Basierend auf den Erörterungen wird der Begriff Geschäftsprozessmodell für diese Arbeit wie folgt definiert:

Definition 2-2: Geschäftsprozessmodell

Ein *Geschäftsprozessmodell* ist die Repräsentation eines Geschäftsprozesses unter Verwendung einer (in dieser Arbeit: grafischen) Modellierungssprache. Es beschreibt alle für einen definierten Zweck relevanten Aspekte des Geschäftsprozesses, wobei insbesondere die Reihenfolge von Aktivitäten fokussiert wird. Zur Modellierung wird eine Modellierungssprache herangezogen, die bezüglich des Modellierungszwecks ausgewählt wird.
Ein Geschäftsprozessmodell definiert i. d. R. einen Geschäftsprozesstyp. Eine Prozessinstanz stellt eine konkrete Ausprägung des modellierten Geschäftsprozesstyps dar.
[Ober96, Fran10]

Das GPM basiert auf Geschäftsprozessmodellen. Daher lassen sich in Tabelle 2-2 zusammenfassend folgende Einsatzzwecke und damit auch Zwecke von Geschäftsprozessmodellen je nach GPM-Phase in Anlehnung an [Ober96, Kobl10, AnDQ22] definieren:

Tabelle 2-2: Einsatzzwecke von Geschäftsprozessmodellen nach GPM-Phasen

Phase	Einsatzzweck
Prozessidentifikation	Hilfsmittel zum Verstehen eines Ablaufs bzw. Systems inklusive Erhöhung der Transparenz
Prozessdokumentation	
Prozessanalyse	Analyse und Simulation von Eigenschaften/Verhaltensmustern, die am Original nicht oder nur schwer untersucht werden können Durchführung von Experimenten
Prozessredesign	Vergleich alternativer Prozessdesigns und Auswahlunterstützung

Phase	Einsatzzweck
Prozessimplementierung	Grundlage zur Kommunikation zwischen Mensch und Maschine / Maschinen / Menschen Anforderungsspezifikation zur Gestaltung eines Systems sowie zur Leistungsvereinbarung
Prozessmonitoring und -steuerung	Automatisierte Ausführung und Steuerung (Einführung von Workflow Management Systemen) Kontinuierliches Prozessmanagement inkl. Qualitätsmanagement

Auf Grundlage dieser Definitionen, Beispiele und Einsatzzwecke lassen sich die Anforderungen an Modellierungssprachen, die zur Modellierung von Geschäftsprozessen verwendet werden können, definieren. Abhängig von dem Einsatzzweck sind unterschiedliche Aspekte eines Geschäftsprozesses relevant, die in einem Geschäftsprozessmodell abgebildet werden sollen.

Werden die in Abbildung 2-1 dargestellten Eigenschaften und deren Zusammenhänge eines Geschäftsprozesses betrachtet, sind folgende Aspekte zu erwähnen: Aktivitäten und deren Reihenfolge, Aufgabenträger, Objekte, Beziehungen, Rollen und weitere Geschäftsregeln. Durch die Abbildung dieser Aspekte können Geschäftsprozessmodelle Transparenz über Geschäftsprozesse, Objekte und Aufgabenträger liefern und damit weitere Einsatzzwecke verfolgen. Die Modellierung von Geschäftsprozessen erfolgt häufig manuell durch Modellierer, die beispielsweise durch Interviews mit Wissensträgern, Analysen von Unterlagen oder durch eigene Beobachtungen zunächst detaillierte Informationen über den zu modellierenden Geschäftsprozess sammeln, oder kann durch Algorithmen unterstützt werden (vgl. Abschnitt 4.2.) [AaCa22].

Durch den Geschäftsprozess als Gegenstand der Modellierung ergeben sich Anforderungen an Modellierungssprachen, wie beispielsweise, dass diese über eine entsprechende Ausdrucksmächtigkeit verfügen müssen, damit alle relevanten Aspekte des (gedachten) Modelloriginals darstellbar sind und diese Darstellung über eine präzise Semantik verfügt. Dennoch sollte eine angemessene Darstellung möglich sein, d. h., dass ein einfacher Sachverhalt nicht durch ein unnötig komplexes Modell dargestellt werden muss. Zusätzlich, um ein Geschäftsprozessmodell auch in den späteren Phasen des GPM verwenden zu können, sollten Methoden zur Analyse und Simulation existieren. Zur Erstellung und grundsätzlichen Verwendung des Geschäftsprozessmodells sollte die Modellierungssprache als leicht erlernbar von den Modellierenden und den Anwendenden empfunden werden. Anwendende eines Modells sind Personen, die das Modell lesen und je nach Modellierungszweck verwenden. Die genannten Anforderungen werden in [Ober96] detailliert beschrieben. Weitere Anforderungen an Modellierungssprachen wie die Modellierung von Zielen werden in [FrLa03] betrachtet.

Die definierten Anforderungen sollten durch eine Modellierungssprache für die Geschäftsprozessmodellierung erfüllt werden, um ein Geschäftsprozessmodell im Sinne des GPM mit Fokus auf der Dokumentation, Verbesserung und Automatisierung von Geschäftsprozessen verwenden zu können. Der Erfüllungsgrad der Anforderungen ist je nach Modellierungssprache unterschiedlich. Dies lässt sich insbesondere auf die verwendete Modellierungsperspektive zurückführen. Daher ist bei der Auswahl einer Modellierungssprache bezüglich des Einsatzzweckes eines Geschäftsprozessmodells unter anderem die zugrundeliegende fokussierte Perspektive der Modellierungssprache entscheidend.

2.2 Modellierungsperspektiven für Geschäftsprozesse

Eine Modellierungsperspektive eines Modellierungsvorhabens beschreibt den fokussierten Aspekt von dem Modelloriginal, der in einem Modell dargestellt werden soll. Entsprechend inhomogen sind die Forderungen an Inhalte und Erscheinungsformen und die damit verbundenen Vorstellungen über erstrebenswerte Eigenschaften dieser Modelle [Kobl10]. Liegt der fokussierte Aspekt bei dem Vorhaben einen Geschäftsprozess zu modellieren, auf der Reihenfolge von Aktivitäten, wird zur Modellierung von Geschäftsprozessen häufig die kontrollflussorientierte Modellierungsperspektive gewählt. Stehen bei einem Modellierungsvorhaben jedoch andere Aspekte des Geschäftsprozesses im Fokus, wie beispielsweise die verwendeten IT-Systeme, ist eine andere Modellierungsperspektive zu wählen.

Eine perspektivenorientierte Modellierung kann die Gestaltung verbessern und die Missverständnisse zwischen den Modellierern und Anwendern des Modells verringern. Durch unterschiedliche Modellierungsperspektiven kann der Modellierer sich an die Bedürfnisse der Anwender anpassen [JaGo08]. Folgend werden im Kontext von Geschäftsprozessen relevante Modellierungsperspektiven genannt und anschließend Modellierungsperspektiven, die in dieser Arbeit in der Methode zur Prozessmodellgenerierung aufgegriffen werden sollen, weiter beschrieben.

Modellierungsperspektiven können nach den zu beantwortenden Kernfragen eines Geschäftsprozesses, die in einem Modell beantwortet werden sollen, eingeteilt werden [OFFD14]. Hierzu werden drei Fragen unterschieden:

- *Wer ist Aufgabenträger*? Die Modelle, die diese Frage beantworten sollen, liefern ein Überblick über die Subjekte eines Geschäftsprozesses (Analyse der Subjekte).

- *Welche Objekte oder Hilfsmittel benötigt der Aufgabenträger*? (Analyse der Objekte) Bei der Fokussierung auf die Objekte eines Geschäftsprozesses können in einem Modell die Funktionen und/oder Daten zu den Objekten integriert werden [Gada17].
- *Was macht der Aufgabenträger*? (Analyse der Prädikate) Die Prädikatanalyse stellt einen Überblick über die Aufgaben eines Geschäftsprozesses bereit. Die aufgabenorientierte Modellierungsperspektive fokussiert beispielsweise die Reihenfolge der Aktivitäten, die ausgeführt werden, um eine Aufgabe zu erfüllen.

Neben der Modellierungsperspektive in der Prädikatsanalyse, die die Reihenfolge der Aktivitäten fokussiert, können auch die Aufgaben unabhängig ihrer zur Erfüllung auszuführenden Aktivtäten und deren zeitlichen Zusammenhänge untereinander betrachtet werden. Es können der Zweck einer Aufgabe oder auch die Bedingungen für die Ausführung einer Aufgabe fokussiert werden [SZNS06]. Damit ergeben sich diese Detail-Modellierungsperspektiven:

- *Kontrollflussbezogen*: Die Reihenfolge der Aktivitäten steht im Vordergrund. Die Reihenfolge ergibt sich durch die kausalen Zusammenhänge zwischen den Aktivitäten oder organisatorischen Vorgaben [vgl. Ober96, FrLa03, JaGo08, Gada17].
- *Funktional*: Die Aufgaben eines Geschäftsprozesses und deren Zwecke sollen dargestellt werden. Dazu zählen die zu erreichenden Ziele der Aufgaben, wie beispielsweise die Transformation der Input-Leistungen zu Output-Leistungen. Dies kann über Modellierungssprachen, die Ziele und deren Zusammenhänge darstellen, fokussiert werden [vgl. Wesk19, SZNS06, JaLi96].
- *Verhaltensbezogen*: Die Bedingungen für die Erfüllung der Aufgaben bzw. der Ausführung der Aktivitäten werden fokussiert [vgl. SZNS06]. Eine Bedingung ist eine Aussage, die, wenn sie sich als *wahr* erweist, eine andere Aussage impliziert. Der Wahrheitswert einer Bedingung kann sich mit den Prozessinstanzen bzw. während der Lebenszeit einer Instanz ändern. Durch die Änderungen der Wahrheitswerte wird die Durchführung eines Geschäftsprozesses beeinflusst [FrLa03].

Bei der Objektanalyse lässt sich zwischen den Objekten und dem dynamischen Verhalten bezogen auf diese Objekte unterscheiden. Diese Modellierungsperspektiven können ebenfalls für die Betrachtung eines Geschäftsprozesses relevant sein.

- *Objekte:* Die Objekte eines Geschäftsprozesses können durch die Objektstruktur beschrieben werden. Ebenso können konkrete Ausprägungen dieser Objekte beschrieben werden [Obje17]. Objekte werden durch Daten im Geschäftsprozess abgebildet. Objektdaten können sich auf eine Aktivität, einen Geschäftsprozess oder das Unternehmen beziehen [Wesk19].

- *Objektfluss* (Datenfluss): Fokussiert werden die Input-Output-Objekttransformation im Geschäftsprozess. Dabei werden die im Rahmen des Geschäftsprozesses erzeugten oder verwendeten Objekte betrachtet. Diese Perspektive beschreibt nicht den Prozessablauf an sich, sondern den Objekt- bzw. Datenfluss, also den Verlauf der Objekte im Zusammenspiel der Aktivitäten [vgl. JaLi96, Gada17]. Der Prozessablauf ergibt sich nur indirekt aus den Darstellungen, wobei der Ablauf der Prozessschritte nur schwer aus den Diagrammen herauszulesen sein kann [Gada17]. Werden die Aktivitäten der Abläufe mit Bürodokumenten bzw. -formularen durchgeführt, können formularorientierte Sprachen zur Ablaufmodellierung verwendet werden [Ober96].

Die Analyse der Subjekte kann nach der organisatorischen und operativen Perspektive differenziert werden:

- Die *organisatorische* Perspektive fokussiert die am Geschäftsprozess beteiligten Unternehmenseinheiten sowie deren Rollen [Wesk19].
- Die *operative* Modellierungsperspektive fokussiert die Systeme, die zur Ausführung der Aufgaben notwendig sind [Wesk19]. In der operativen Perspektive werden diese Systeme sowie die Unternehmens- und Geschäftsprozess-Daten, die über Schnittstellen in diese Systeme hinein- und aus ihnen herausgeführt werden, fokussiert [AHKB03]. Weiter kann die operative Perspektive die Aufrufumgebung von Systemen und die Zuordnung der Eingabe- und Ausgabeparameter der Systeme zu den Eingabe- und Ausgabeparametern von Geschäftsprozessaktivitäten umfassen [Wesk19].

In Tabelle 2-3 werden die beschriebenen Modellierungsperspektiven sowie deren fokussierte Aspekte zusammengefasst. Es zeigen sich dabei Überschneidungen zwischen den Modellierungsperspektiven, da diese Modellierungsperspektiven die Aspekte eines Geschäftsprozesses nicht trennscharf aufgreifen.

Tabelle 2-3: Übersicht ausgewählter Modellierungsperspektiven

Modellierungsperspektive	**Fokussierter Aspekt**
Subjektorientiert	Wer ist Aufgabenträger? (Analyse der Subjekte)
Operativ	Operationen, die zur Ausführung der Aktivitäten aufgerufen werden sowie dafür verwendete Dienste (beispielsweise Werkzeuge und Applikationen)
Organisatorisch	Beziehungen zwischen den Mitarbeitern (Benutzer, Rollen), die für die Ausführung von verschiedenen Aufgaben qualifiziert sind

Modellierungsperspektive	Fokussierter Aspekt
Aufgabenorientiert	Was macht der Aufgabenträger? Welche Aufgaben sind zu erfüllen und welche Aktivitäten werden ausgeführt? (Analyse der Prädikate)
Funktional	Aufgaben des Geschäftsprozesses und deren Zweck
Verhaltensbezogen	Bedingungen für die Erfüllung der Aufgaben und Ausführungen der Aktivitäten
Kontrollflussbezogen	Reihenfolge der Aktivitäten durch kausale Zusammenhänge oder organisatorischen Vorgaben
Objektorientiert	Welche Objekte oder Hilfsmittel benötigt der Aufgabenträger? (Analyse der Objekte)
Objekte	Konsumierte oder produzierte Objektdaten bezüglich der Aktivitäten, des Geschäftsprozesses oder des Unternehmens
Objektfluss	Verwendete Objekte und Objektfluss zwischen den Aktivitäten des Geschäftsprozesses. Darstellen der Beziehung zwischen Prozessdaten und Datenmodellen

Die Modellierungsperspektiven unterscheiden sich demnach in den fokussierten Aspekten, aber auch in den geeigneten Modellierungsmethoden und in den *zugeordneten* Modellierungssprachen. Eine Modellierungssprache, die einer Modellierungsperspektive zugeordnet ist, ist dafür geeignet, den gewünschten fokussierten Aspekt des Modelloriginals in einem Modell abzubilden. Durch unterschiedliche Ansätze zur Einteilung der Modellierungsperspektiven und den unterschiedlichen Ausdrucksstärken einer Modellierungssprache oder den Erweiterungen zu Modellierungssprachen können Modellierungssprachen mehreren Modellierungsperspektiven zugeordnet werden.

Die in Abschnitt 2.1.3 bereits genannten Modellierungssprachen EPK, BPMN, Petri-Netze oder die UML-Aktivitätsdiagramme sind der kontrollflussorientierten Modellierungsperspektive zugeordnet. Die subjektorientierte Business Process Management-Modellierung (S-BPM) stellt eine Modellierungssprache zur subjektorientierten Modellierungsperspektive bereit [FSSO+14]. Dabei liegt der Schwerpunkt auf den beteiligten Personen und auf den Schnittstellen im Geschäftsprozess, um beispielsweise Applikationslösungen zu erstellen und reibungslose Integrationen in die bestehende IT-Struktur zu ermöglichen. Zu den Modellierungssprachen, die eine objektorientierte Modellierungsperspektive ermöglichen, zählen die Klassen- und Objektdiagramme von UML (UML-OD). Die OMG schlägt dabei vor, die notwendigen Objekte zu identifizieren und dann das dynamische Verhalten zu definieren [OMG17]. Einige Modellierungssprachen werden dabei auch verwendet, um Aspekte mehrerer Modellierungsperspektiven abzubil-

den: Die Modellierungssprache BPMN bietet die Möglichkeit, zusätzlich zum Kontrollfluss sowohl Objekte als auch Rollen und IT-Systeme abzubilden [Obje17]. Dennoch ist hierbei meist ein Aspekt mehr im Fokus und bildet die Grundlage des Modells.

Die Eigenschaften eines Geschäftsprozesses aus Abbildung 2-1 werden in unterschiedlichen Modellierungsperspektiven in unterschiedlichem Maß aufgegriffen. Auf Grundlage der ausgeprägten Eigenschaften der Prozessinstanzen eines Geschäftsprozesstyps kann ein Geschäftsprozessmodell deskriptiv erstellt werden. In dieser Arbeit soll dieses Geschäftsprozessmodell aus Daten, die diese Eigenschaften beschreiben, generiert werden. In diesem Zusammenhang sind insbesondere die aus den Daten ableitbaren Aspekte zu den Aktivitäten und den Objekten eines Geschäftsprozesses relevant, die in den kontrollflussorientierten und objekt(-fluss-)orientierten Modellierungsperspektiven fokussiert werden. Daher werden diese Modellierungsperspektiven im Folgenden betrachtet.

2.2.1 Kontrollflussorientierte Modellierungsperspektiven

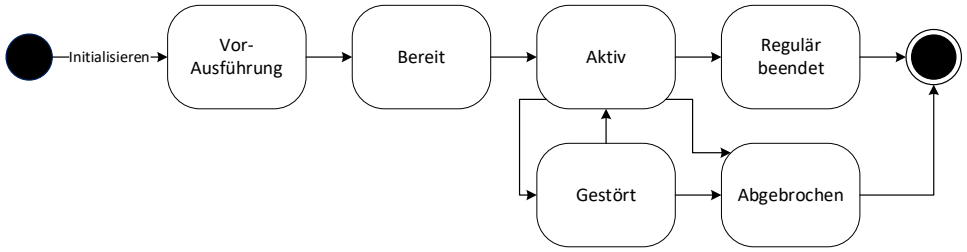
Die kontrollflussorientierte Modellierungsperspektive fokussiert die Aktivitäten[4] und ihre Ausführungsreihenfolge. In einem Modell können die Aktivitäten und ihre Ausführungsreihenfolge durch verschiedene Kontrollflussmuster abgebildet werden. Innerhalb der Kontrollflussmuster können Konstruktoren zur Darstellung verwendet werden, um den Kontrollfluss aufzuspalten oder zu vereinigen [AHKB03]. In [Wesk19] und [AHKB03] werden einige Kontrollflussmuster definiert. Die Darstellungsform eines Kontrollflussmusters in einem Modell unterscheidet sich je nach gewählter Modellierungssprache. Des Weiteren ist es von einem Modellierer abhängig, ob und wie eine Ausführungsreihenfolge von Aktivitäten eines Geschäftsprozesses in einem Geschäftsprozessmodell dargestellt wird [Agui04].

Wie in Abschnitt 2.1.3 beschrieben, ist die Reihenfolge von zwei Aktivitäten in einem Geschäftsprozess durch vorhandene oder nicht-vorhandene kausale Zusammenhänge zwischen diesen Aktivitäten sowie organisatorische Vorgaben definiert. Die verwendeten Kontrollflussmuster in einem Geschäftsprozessmodell geben daher die Ausführungssemantik der einzelnen Prozessinstanzen vor, denn diese beziehen sich auf die Ausführungen der Aktivitätstypen, die Aktivitätsinstanzen. Daher wird zur Definition verschiedener Kontrollflussmuster zunächst der Zustandsraum für Aktivitätsinstanzen definiert. Wie in Abschnitt 2.1.2 beschrieben, kann einer Aktivitätsinstanz zu jedem Zeitpunkt ein Zustand zugeordnet werden [Ober96]. Es können folgende Zustände unterschieden werden:

[4] Im Folgenden wird der Begriff Aktivität verwendet, wobei die Erfüllung einer Aufgabe durch Ausführen einer oder mehreren Aktivitäten erfolgt (vgl. Kapitel 2.1.2).

- *Vor-Ausführung*: Die Aktivitätsinstanz wurde initialisiert, aber noch nicht gestartet. Diesen Zustand erreicht eine Aktivitätsinstanz, wenn durch die Initialisierung einer Prozessinstanz die Aktivitätsinstanzen erzeugt werden.
- *Bereit*: Die Aktivitätsinstanz ist zur Ausführung bereit, aber noch nicht aktiv. Demnach wurde die Ausführung der Aktivitätsinstanz noch nicht begonnen.
- *Aktiv*: Die Aktivitätsinstanz wurde gestartet und befindet sich in der Ausführung.
- *Gestört*: Die Aktivitätsinstanz ändert den Zustand von *aktiv* in *gestört* aufgrund eines unterbrechenden Fehlerereignisses.
- *Abgebrochen*: Die Aktivitätsinstanz wurde abgebrochen.
- *Regulär beendet*: Die Ausführung der Aktivitätsinstanz wurde erfolgreich abgeschlossen.

Das Zustandsdiagramm in Abbildung 2-5 zeigt die möglichen Übergänge zwischen den Zuständen einer Aktivitätsinstanz.



in Anlehnung an [Ober96, Wesk19]

Abbildung 2-5: Zustandsübergänge von Aktivitätsinstanzen

Im Folgenden werden die fünf Basis-Kontrollflussmuster bezüglich der Zustände von Aktivitätsinstanzen beschrieben (Abschnitt 2.2.1.1, 2.2.1.2 und 2.2.1.3). Weitere Kontrollflussmuster werden in Abschnitt 2.2.1.4 kurz aufgegriffen. Die Definitionen dieser Kontrollflussmuster basieren auf den Arbeiten von [Wesk19] und [AHKB03]. Bei den vorgestellten Kontrollflussmustern wird ebenfalls exemplarisch darauf eingegangen, wann diese Muster in einem Modell auf Grundlage der Eigenschaften eines Geschäftsprozesses gewählt werden könnten. Dazu werden Prozessinstanzen betrachtet und zeitliche Zusammenhänge zwischen den Aktivitätsinstanzen relativ zu der Beobachtung beschrieben. Dies sind Zusammenhänge der Art: Aktivitätsinstanz A_i wird vor Aktivitätsinstanz B_i beobachtet. Da diese Beobachtungen mit den kausalen Zusammenhängen verträglich sein müssen und mit den organisatorischen Vorgaben verträglich sein sollten, können darauf basierend Kontrollflussmuster vermutet werden. Verträglichkeit heißt dabei, dass die Ausführung des Aktivitätstyp A nicht nach der Ausführung des Aktivitätstyp B beobachtet werden kann, wenn Aktivitätstyp A durch einen kausalen Zusammenhang der Vorgänger von dem Aktivitätstyp B sein muss [Ober90]. Die aufgrund der Beobachtung vermuteten Kontrollflussmuster sind nicht mit Sicherheit bestimmbar, da

der beobachtete Ausführungszeitpunkt auch lediglich zufällig so sein kann, oder nicht alle vorhandenen Zusammenhänge in den ausgewählten beobachteten Prozessinstanzen abgebildet sind. Eine detaillierte Betrachtung, wie aufgrund verschiedener Prozessinstanzen Rückschlüsse auf den in einem Modell abzubildenden Kontrollfluss gezogen werden können, erfolgt in Kapitel 6.

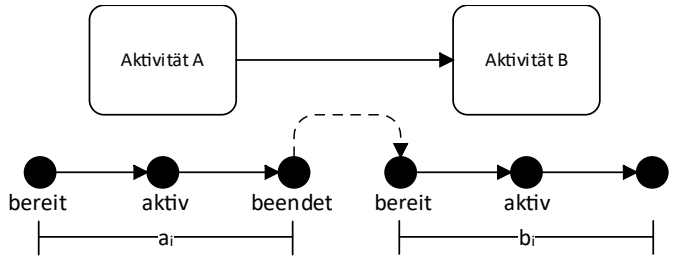
Zur Beschreibung der zeitlichen Zusammenhänge werden die folgenden Begriffe verwendet:

- Prozessinstanz g_i: Eine Prozessinstanz ist eine konkrete Ausführung eines Geschäftsprozesstyps, wobei eine Prozessinstanz einen konkreten Kontrollfluss des Geschäftsprozesses durchläuft.
- Aktivität X und Aktivitätsinstanz x_i wobei $X = \{A, B, C, \dots\}$, $x = \{a, b, c, \dots\}$ und $i = \{1, 2, 3, \dots\}$: Eine Aktivität X bildet einen Aktivitätstyp in einem Geschäftsprozesstyp ab. Die Aktivitätsinstanz x_i bezeichnet eine konkrete Ausprägung der Aktivität X.

2.2.1.1 Sequenz

Die Aktivitäten A und B eines Geschäftsprozesses sind in sequenzieller Reihenfolge im Geschäftsprozessmodell angeordnet, wenn in der Prozessinstanz g_i eine Aktivitätsinstanz b_i erst nach der Beendigung der Aktivitätsinstanz a_i durchgeführt wird. Unter Verwendung der definierten Zustände kann der Zustand von b_i erst in *bereit* übergehen, wenn der Zustand von a_i *regulär beendet* ist. Daher kann b_i nicht ausgeführt werden, wenn a_i übersprungen wurde. Wobei eine Instanz der Aktivität B (b_j) nicht von den Instanzen der Aktivität A anderer Prozessinstanzen (a_i, $i \neq j$) sequenziell abhängig ist.

In Abbildung 2-6 werden das Kontrollflussmuster Sequenz sowie die Zustände der Aktivitätsinstanzen abgebildet.



[Wesk19]

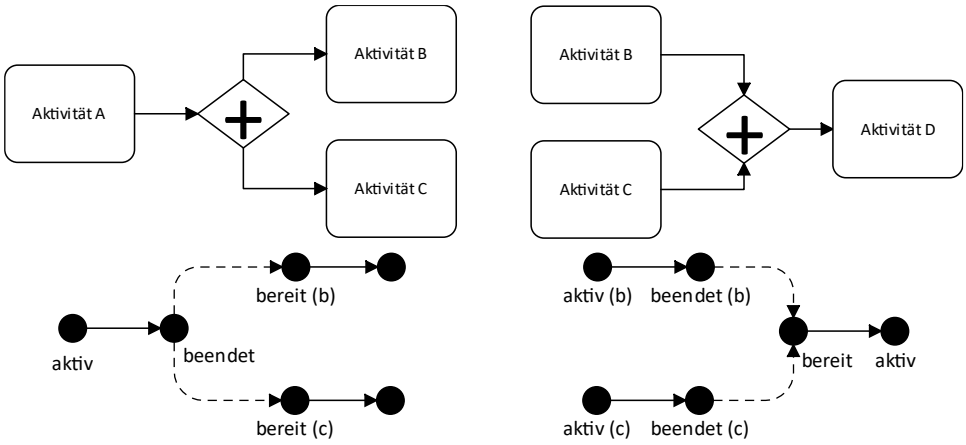
Abbildung 2-6: Darstellung des Kontrollflussmusters Sequenz

Bei der Prozessmodellgenerierung kann eine sequenzielle Reihenfolge von den zwei Aktivitäten A und B vermutet werden, wenn beobachtet wird, dass der Ausführungszeitpunkt der Aktivitätsinstanz B_i immer nach der erfolgreich beendeten Ausführung der Aktivitätsinstanz A_i in derselben Prozessinstanz g_i ist. Zusätzlich ist nie eine Ausführung der Aktivitätsinstanz von Aktivität B aufgetreten, wenn innerhalb einer Prozessinstanz die Aktivitätsinstanz von Aktivität A nicht erfolgreich beendet wurde. Jedoch könnten diese genannten Zusammenhänge auch zufällig, d. h. nicht kausal, auftreten. Werden jedoch nicht nur die Ausführungszeitpunkte der Aktivitäten betrachtet, sondern auch zugehörige Daten, könnte eine Abhängigkeit der einen Aktivität von der anderen Aktivität bestätigt werden. Im Beispielprozess aus Abschnitt 2.1.2 ist die Vertragsgestaltung vor der Rechnungsstellung durchzuführen. Dementsprechend sind innerhalb einer Prozessinstanz die Ausführungszeitpunkte der Rechnungsstellung nach der Beendigung der Vertragsgestaltung. Dieser Zusammenhang könnte durch die Objektbetrachtung weiter bestätigt werden, da hier eine Rechnung auf den zugehörigen Vertrag sowie die darin aufgelisteten gewünschten Produkte verweist.

2.2.1.2 Nebenläufigkeit (AND-Aufspaltung; AND-Zusammenführung)

Die Aktivitäten B und C eines Geschäftsprozesses sind nebenläufig nach einer Aktivität A angeordnet, wenn in einer Prozessinstanz g_i nach der Beendigung der Aktivitätsinstanz a_i sowohl die Aktivitätsinstanz b_i als auch die Aktivitätsinstanz c_i bereit werden. Zusätzlich sind die zwei Aktivitätsinstanzen der Aktivitäten B und C unabhängig voneinander. Zur Darstellung der Nebenläufigkeit wird eine AND-Aufspaltung als Kontrollflussmuster verwendet. Eine AND-Zusammenführung repräsentiert die Zusammenführung von den Aktivitäten B und C, die einen sequenziellen Zusammenhang zur Aktivität D haben. Erst wenn sowohl die Aktivitätsinstanz b_i als auch die Aktivitätsinstanz c_i beendet sind, erreicht die Aktivitätsinstanz d_i den Zustand bereit.

In Abbildung 2-7 werden die Kontrollflussmuster der Nebenläufigkeit und die Zustände der Aktivitätsinstanzen abgebildet. Zusätzlich zeigt das Kontrollflussmuster die sequenzielle Reihenfolge von der Aktivität A und B und der Aktivität A und C auf der linken Seite sowie von der Aktivität B und B und von der Aktivität C und D auf der rechten Seite der Abbildung.



[Wesk19]

Abbildung 2-7: Darstellung des Kontrollflussmusters Nebenläufigkeit

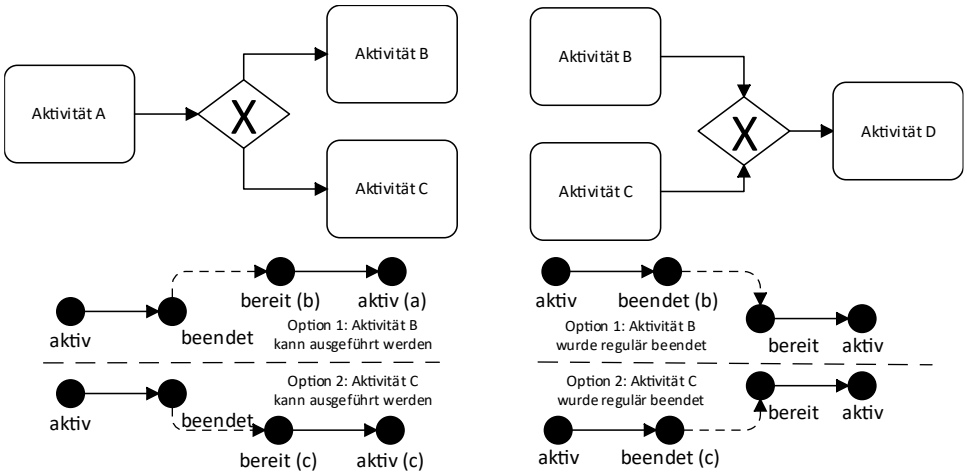
Bei der Prozessmodellgenerierung kann eine nebenläufige Ausführung von zwei Aktivitäten vermutet werden, wenn beobachtet wird, dass die Ausführungszeitpunkte der Aktivitäten B und C in unterschiedlichen Reihenfolgen über die Prozessinstanzen hinweg auftreten. Die gegebenenfalls umgebenen sequenziellen Abhängigkeiten sind wie bei dem Kontrollflussmuster Sequenz zu beobachten: die Ausführung der Aktivitäten B und C starten immer erst nach der erfolgreich beendeten Ausführung von Aktivität A innerhalb einer Prozessinstanz und es existiert keine Aktivitätsinstanz von B und C, wenn innerhalb einer Prozessinstanz die Aktivitätsinstanz von Aktivität A nicht erfolgreich beendet wurde. Jedoch können diese genannten Zusammenhänge auch zufällig auftreten. Der Verdacht auf eine nebenläufige Ausführung auf Grund der genannten Beobachtungen kann bekräftigt werden, wenn die zugehörigen Objekte, die in den Aktivitäten bearbeitet oder erstellt werden, ebenfalls keine Abhängigkeiten aufzeigen. Im Beispielprozess aus Abschnitt 2.1.2 können die Rechnung und das verkaufte Produkt nebenläufig versendet werden, da zwar in der Rechnung Bezug zum Produkt durch eine Produkt-ID genommen wird, diese Information jedoch auf dem zuvor erstellten Vertrag basiert und somit die Objekte *Rechnung* und *Produkt* unabhängig voneinander sind. Beide Objekte weisen jedoch Abhängigkeiten zum Vertrag auf und somit sind die Ausführungszeitpunkte der beiden Aktivitäten zur Bearbeitung und zum Versand der Rechnung bzw. des Produktes erst nach der Fertigstellung des Vertrages.

2.2.1.3 Alternative Ausführung (XOR-Aufspaltung; XOR-Zusammenführung)

Zwei Aktivitäten B und C werden alternativ zueinander ausgeführt, wenn nach der erfolgreichen Beendigung einer vorgelagerten Aktivitätsinstanz ai ausschließlich eine der beiden nachfolgenden Aktivitätsinstanzen (b_i oder c_i) ausgeführt wird. Dies bedeutet, dass sich die Aktivitäten B und C in einer Prozessinstanz gegenseitig ausschließen

Zur Darstellung der alternativen Ausführung in einem Kontrollflussmuster wird eine XOR-Aufspaltung verwendet. Eine XOR-Zusammenführung bedeutet, dass der Prozessfluss nach Beendigung einer der alternativen Aktivitätsinstanzen (a_i oder b_i) mit der nachfolgenden Aktivitätsinstanz c_i fortgesetzt wird. Da bei einer alternativen Ausführung die Aktivitäten A und B nie nebenläufig ausgeführt werden, erfolgt keine Zusammenführung verschiedener aktiver Ausführungspfade in der Instanz.

In Abbildung 2-8 werden die Kontrollflussmuster der alternativen Ausführung sowie die Zustände der Aktivitätsinstanzen abgebildet.



[Wesk19]

Abbildung 2-8: Darstellung des Kontrollflussmusters Alternative Ausführung

Bei der Prozessmodellgenerierung kann eine alternative Ausführung von zwei Aktivitäten vermutet werden, wenn beobachtet wird, dass die Ausführung der zwei Aktivitäten B und C immer nach der erfolgreich beendeten Ausführung der Aktivität A ist und die Aktivitätsinstanzen B und C nie zusammen in derselben Prozessinstanz auftreten. Zusätzlich wird keine Ausführung der beiden Aktivitäten beobachtet, wenn innerhalb der Prozessinstanzen die Aktivitätsinstanz von Aktivität A nicht erfolgreich beendet wurde. Jedoch könnten diese genannten Zusammenhänge auch zufällig auftreten. Ebenso kann eine alternative Ausführung auch Teil eines Zyklus sein, sodass die Aktivitäten B und C zwar nie nebenläufig ausgeführt werden, jedoch beide in einer Prozessinstanz vorkommen können. Dies könnte erkannt werden, wenn die Aktivitäten B und C in wiederholenden Sequenzen nur alternativ auftreten, wie beispielsweise bei den hintereinander folgenden Sequenzen: ABD und ACD. Wie bereits bei den anderen Kontrollflussmustern diese genannten Zusammenhänge durch die Betrachtung der Objekte unterstützt werden. Wird das Prozessbeispiel aus Abschnitt 2.1.2 betrachtet, lassen sich zwei Alternativen durch die zugehörigen Objekte bestätigen. Es wird entweder die Aktivität *Rechnungstellung mit*

Rabatt oder die Aktivität *Rechnungstellung ohne Rabatt* ausgeführt. Bei beiden Aktivitäten wird das Objekt *Rechnung* erzeugt, das entweder einen Preis mit oder ohne Rabatt definiert. Bei den Prozessinstanzen wird somit immer ein Objekt *Rechnung* erzeugt, das entweder einen reduzierten oder einen nicht reduzierten Preis enthält. Ebenso könnte bei der Aktivität *Kunde anrufen* die Aktivität *Kunde per Mail kontaktieren* als Alternative ergänzt werden. Würde es als Alternative gehandhabt werden, würde immer nur eine E-Mail oder die Telefondaten in einer Prozessinstanz auftreten.

2.2.1.4 Weitere Kontrollflussmuster

Neben den Basis-Kontrollflussmustern können weitere Muster bezüglich des Kontrollflusses in einem Geschäftsprozess existieren. Diese beziehen sich beispielsweise auf Mehrfachinstanzen oder die implizite Beendigung einer Prozessinstanz. Bei der Prozessmodellgenerierung stellen bestimmte Kontrollflussmuster besondere Herausforderungen dar, da die zugrundeliegenden Zusammenhänge nicht eindeutig abgeleitet werden können oder mehrdeutig interpretiert werden können. Folgend sollen insbesondere die für die Prozessmodellgenerierung als problematisch angesehenen Kontrollflussmuster kurz erläutert werden:

Mehrfachauswahl:
Wenn nach einer Aktivität A sowohl eine Und- als auch XOR-Aufspaltung zu den Aktivitäten B und C auftreten kann, ist der Zusammenhang zwischen den Aktivitäten A, B und C als Mehrfachauswahl definiert. So erreicht entweder nur Aktivitätsinstanz b_i oder die Aktivitätsinstanz c_i durch das reguläre Beenden der Aktivitätsinstanz a_i den Zustand *bereit* (alternative Ausführung) und damit kann nur eine der beiden Aktivitäten ausgeführt werden. Oder beide erreichen zusammen diesen Zustand und somit könnten auch beide ausgeführt werden. Dieses Kontrollflussmuster wird durch eine OR-Aufspaltung (bzw. Zusammenführung) gekennzeichnet. Dieses Muster weist Schwierigkeiten in der präzisen Semantik auf, da unklar sein kann, wann welcher Pfad zu wählen ist bzw. gewählt werden kann [AHKB03].

Zyklus:
Ein Zyklus repräsentiert, dass Aktivitäten mehrfach in einer Prozessinstanz ausgeführt werden können. Das Enden eines Zyklus kann durch eine vorgegebene Bedingung, die erfüllt sein muss, begrenzt werden. Dies kann die Vorgabe einer absoluten Anzahl von Zyklen sein. Zusätzlich zu diesem Zyklus kann auch nur eine einzelne Aktivität mehrfach ausgeführt werden. Die erneute Ausführung kann auch ohne reguläre Beendigung der vorherigen Aktivitätsinstanz der gleichen Aktivität beginnen. Mit BPMN wird diese Mehrfachinstanziierung durch einen entsprechenden Marker in einer abgebildeten Aktivität verdeutlicht. Zyklen und Mehrfachinstanziierungen ohne Endbedingung können problematisch in einem Geschäftsprozess sein, außerdem führen sie zu erhöhter Komplexität [RHAM06].

Ausnahmen:
Ausnahmen wie Terminüberschreitungen, Krankheit eines Sachbearbeiters, Störung einer Maschine oder Fehlen eines Betriebsmittels können in einer Prozessinstanz auftreten. Für als relevant erachtete Ausnahmen werden Behandlungsmechanismen definiert. Wird beispielsweise eine Bestellung storniert, werden alle bereits erstellten Dokumente vernichtet. Treten unvorhergesehene Situationen auf, muss über den weiteren Ablauf der Prozessinstanz erst entschieden werden [FrLa03]. Für die Darstellung solcher Behandlungsmechanismen bedarf es weiterer Modellierungsmöglichkeiten, wenn die Behandlung nicht der bereits genannten Kontrollflussmuster entspricht. Daher kann die Feststellung einer Ausnahme sowie deren Modellierung problematisch sein.

2.2.2 Objekt(-fluss-)orientierte Modellierungsperspektiven

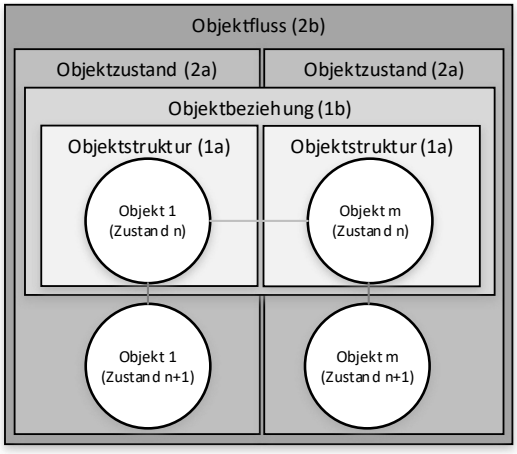
Sollen die Objekte bzw. der Objektfluss eines Geschäftsprozesses fokussiert werden, können unterschiedliche Modellierungsperspektiven herangezogen werden. Wie in Abschnitt 2.1.2 beschrieben, können Objekte beispielsweise Aufträge und Produkte sein. Bei der Modellierung kann zwischen den statischen (in Abbildung 2-9 (1a) und (1b)) und dynamischen (in Abbildung 2-9 (2a) und (2b)) Aspekten der Objekte unterschieden werden [Vett98].

Die *statischen* Aspekte der Objekte können als Entitäten mit Attributen in Datenmodellen dargestellt werden (1a). Hierzu wird beispielsweise ein Auftrag als Entitätstyp *Auftrag* klassifiziert und mit Attributen wie *Auftragsnummer*, *Datum*, *Menge* und *Betrag* beschrieben. Diese Objektstrukturen können beispielsweise in UML-Klassendiagramme und die Instanzen dieser Objekte zu einem bestimmten Zeitpunkt in UML-Objektdiagrammen beschrieben werden [Obje17]. Der Zustand eines Objektes zu einem bestimmten Zeitpunkt beschreibt die Belegung der Eigenschaften mit einem bestimmten Wert [Ober96]. Objekte können aus atomaren Werten bestimmter Datentypen konstruiert werden und sind einfach strukturiert, wenn sie sich ausschließlich aus atomaren Werten zusammensetzen. Setzen sich die Objekte nicht nur aus atomaren Werten sondern auch selbst wieder aus strukturierten Objekten zusammen, sind es komplex strukturierte Objekte [Obje17].
Weiter können in ER-Modellen und UML-Klassendiagrammen nicht nur die Objekte, sondern auch deren Zusammenhänge durch Beziehungen beschrieben werden (1b) [Chen76, Obje17]. Dadurch können beispielsweise die Abhängigkeiten der Objekte in einem Geschäftsprozess durch die Modellierung der Entitätstypen dargestellt werden [Wesk19].

Ebenfalls können die *dynamischen* Aspekte der Objekte in einem Modell beschrieben werden. Ein dynamischer Aspekt sind die Zustandsänderungen einzelner Objekte (2a).

Zustandsänderungen spiegeln sich bei der Objektfokussierung im Verhalten wieder, d. h. wie ein Objekt agiert und reagiert, bestimmt durch die Menge der Operationen, die bezüglich eines Objekts durchführt werden können sowie durch die Kenntnis seiner Beziehungen und Funktionen [Agui04].
Ein weiterer dynamischer Aspekt ist der Objektfluss durch die Übertragung der Objekte zwischen Aktivitäten und (Teil-)Prozessen (2b). Hierdurch werden auch die Objektabhängigkeiten abgebildet. Objekte können zwischen Aktivitäten und Teilprozessen derselben Prozessinstanz und auch zwischen Aktivitäten verschiedener Prozessinstanzen ausgetauscht werden. Für Aktivitäten, die Objekte verändern oder austauschen, können Ein- und Ausgabeparameter definiert werden. Diese Parameter können mit Objekten oder Attributwerten von Objekten (beispielsweise durch die Übergabe von Referenzen auf die Objekte) belegt werden [Wesk19]. Bei der Ausführung einer Aktivität werden die Eingabeparameter gelesen und bei der regulären Beendigung der Aktivität die Ausgabeparameter belegt. Diese Ausgabeparameter können sowohl dem Typ der Eingabeparametern entsprechen als auch einen neuen Typ haben. Anschließend können diese Parameter wieder Eingabeparameter für nachfolgende Aktivitäten darstellen. Die Ausführungsbedingung einer Aktivität kann daher in Abhängigkeit von Objekten definiert werden [MeSW11]. Bei dieser Objektflussmodellierung überlagert die objektorientierte Perspektive die Kontrollflussperspektive. Dabei werden in den Modellen Daten zu Objekten zur Erfüllung von Bedingungen in Geschäftsprozessen betrachtet [AHKB03] oder Aktivitätssequenzen aus Daten-Abhängigkeiten definiert [HGAE+22].

Abbildung 2-9 stellt diese Modellierungsansätze der objekt(-fluss-)orientierten Modellierungsperspektive im Zusammenhang zu den betrachteten Objekten dar.



in Anlehnung an [MeSW11]

Abbildung 2-9: Modellierungsperspektiven der Objektmodellierung

2.3 Qualitätsaspekte für Geschäftsprozessmodelle

Geschäftsprozessmodelle, die manuell oder (teil-)automatisiert erstellt werden, können fehlerhaft sein. Bei der manuellen Modellierung sind Modellierer und Prozessexperte oftmals unterschiedliche Personen. Dies kann zu Kommunikationsproblemen bzw. Transformationseffekten führen. Transformationen finden bei der Abbildung von einem Modelloriginal in einem Modell durch die Vereinfachung und Abbildung sowie bei der Interpretation des Modells statt [VöSB13]. Außerdem können auch eine fehlende Modellier- oder Prozessexpertise zu unzureichender Qualität bei Geschäftsprozessmodellen führen. Die Qualität eines Geschäftsprozessmodells bezieht sich in Anlehnung an die Qualitätsdefinition der ISO 9000 auf den Erfüllungsgrad der definierten Anforderungen bezüglich der Gesamtheit aller Eigenschaften des Geschäftsprozessmodells [ISO15]. Bei der (teil-)automatisierten Erstellung von Geschäftsprozessmodellen kann die Qualität der generierten Geschäftsprozessmodelle unzureichend sein, wenn der Algorithmus zur Generierung qualitativ unzureichende Ergebnisse liefert (beispielsweise, weil er nicht ausreichend überprüft wurde), die gewählten Anforderungen an ein Geschäftsprozessmodell nicht sichergestellt werden, oder bereits die Qualität der Datengrundlage unzureichend ist. Fehlerhafte Geschäftsprozessmodelle können jedoch bei deren Verwendung beispielsweise als Kommunikations- oder Entscheidungsgrundlage, zur Prozessanalyse oder zur Ausführung zu Problemen führen [Mood05].

Daher ist es wichtig, die Qualität von Geschäftsprozessmodellen bei dem Erstellungsprozess zu *gewährleisten* bzw. das erstellte Geschäftsprozessmodell anschließend auf dessen Qualität zu *überprüfen* [ReMR15]. Bei der Qualitätssicherung werden Maßnahmen definiert, mit dem Ziel, dass ein Produkt oder eine Dienstleistung die definierten Anforderungen erfüllt [Balz11]. In Anlehnung daran können diese Maßnahmen in der Geschäftsprozessmodellierung in die *konstruktive* und *analytische* Qualitätssicherung eingeteilt werden:

- *konstruktive* Qualitätssicherung: Maßnahmen, die während der Modellerstellung *gewährleisten* sollen, dass ein Geschäftsprozessmodell den festgelegten Qualitätsanforderungen entspricht.
- *analytische* Qualitätssicherung: Maßnahmen, die nach der Modellerstellung *überprüfen* sollen, ob bzw. in welchem Grad ein Geschäftsprozessmodell die festgelegten Qualitätsanforderungen erfüllt.

Bei der *konstruktiven* Qualitätssicherung werden daher präventive Maßnahmen während der Modellierung wie den unterstützenden Aufbau eines Geschäftsprozessmodells bei der Modellierung angewandt. Die Modellierer ergänzen ein Geschäftsprozessmodell nur mit einfachen Operationen wie „füge parallele Aktivität hinzu“ und unterbinden dadurch syntaktische Fehler im Geschäftsprozessmodell. Dies kann die Modellierungsfreiheit

stark einschränken [Kobl10]. Eine weitere Maßnahme zur konstruktiven Qualitätssicherung ist, wenn die Regeln zur Syntax bereits während der Modellierung eines Modells vorgeschrieben sind und deren Einhaltung erzwungen werden. Dazu stellen einige Modellierungseditoren nur einen bestimmten Zeichenvorrat für die Modellerstellung zur Verfügung oder lassen nur bestimmte Beziehungen zwischen Elementen zu [OvBS12].

Bei der *analytischen* Qualitätssicherung werden ex-post-Analysen von Geschäftsprozessmodellen durchgeführt [ReMR15]. Bei der Überprüfung der Qualität von Geschäftsprozessmodellen wird demnach die Erfüllung oder der Erfüllungsgrad von Qualitätsaspekten bezüglich definierter Anforderungen im Geschäftsprozessmodell überprüft. Das Ergebnis der Betrachtung eines Qualitätsaspektes kann demnach binär sein, indem der Aspekt erfüllt wurde oder nicht, oder es kann angegeben werden, in welchem Grad die Aspekte erfüllt werden. Nachdem einzelne Qualitätsaspekte im Geschäftsprozessmodell überprüft wurden, kann in Einbezug aller überprüften Qualitätsaspekte bewertet werden, ob das Geschäftsprozessmodell in einer ausreichenden Qualität vorliegt.

Bei der Betrachtung der Qualität von Geschäftsprozessmodellen können zwei Betrachtungsweisen unterschieden werden [Kobl10]:

- *Betrachtung der Qualität des modellierten Geschäftsprozesses*: Das Geschäftsprozessmodell wird in Vertretung des abgebildeten Modelloriginals (dem Geschäftsprozess) auf definierte Qualitätsaspekte analysiert.
- *Betrachtung der Modellqualität*: Das Geschäftsprozessmodell wird als eigenständiges Objekt auf definierte Qualitätsaspekte analysiert.

Für diese beiden Betrachtungen sind unterschiedliche Qualitätsaspekte zu definieren. Für die *Betrachtung der Qualität des modellierten Geschäftsprozesses* existieren beispielsweise Qualitätsmodelle für die System- und Softwareentwicklung nach der ISO/IEC 25010:2011 [ISO11] oder für die Geschäftsprozesse nach [Kneu15]. Ziel dieser Qualitätsmodelle ist unter anderem die Einhaltung von Qualitätsaspekten zu Prozesszielen und Anforderungen, zur Prozessmodellierung, Wirksamkeit, Effizienz, Prozessfähigkeit, Konformität und Änderbarkeit [Kneu15]. [JLKR07] nennen die Performance eines Geschäftsprozesses als Ziel, die durch die vier Qualitätsdimensionen Zeit, Kosten, Flexibilität und Qualität bewertet werden kann. Diese vier Aspekte können in Konkurrenz stehen [JLKR07]. Mithilfe eines Modells könnten dort ausgewählte definierte Qualitätsaspekte eines Qualitätsmodells für das betrachtete Modelloriginal überprüft werden. Die Betrachtung der Qualität eines Modelloriginals basierend auf dem zugehörigen Modell wird in der Arbeit von [Kobl10] für Geschäftsprozesse und in der Arbeit von [FiHR08] für Software fokussiert. [FiHR08] unterteilen die Anforderungen für Modelle zur Qualitätssicherung in innere und äußere Qualität. Während die innere Qualität sich auf ein einzelnes Modell bezieht, wird bei der äußeren Beziehung die Umgebung des

Modells bzw. des abgebildeten Modelloriginals betrachtet, um ein Indikator für die Softwarequalität zu liefern [FiHR08]. Die Betrachtung der Prozessqualität anhand eines Modells ist kein Teil dieser Arbeit. In dieser Arbeit wird die *Betrachtung der Modellqualität* fokussiert, da diese in der Prozessmodellgenerierung gewährleistet sein sollte und grundlegend für die Betrachtung der Prozessqualität am Modell ist. Zudem beeinflusst ein Lösungsansatz zur Prozessmodellgenerierung – wie es in dieser Arbeit vorgestellt wird – höchstens indirekt oder nachträglich die Qualität eines Modelloriginals.

Die *Betrachtung der Modellqualität* soll nach den Anforderungen an ein Modell (Abschnitt 2.3.1), nach den Anforderungen an eine Modellierungssprache (Abschnitt 2.3.2) sowie nach den Anforderungen an die Modellierung (Abschnitt 2.3.3) betrachtet werden. Anhand dieser Anforderungen können die zu berücksichtigenden Qualitätsaspekte abgeleitet werden.

In Abbildung 2-10 wird die Unterscheidung nach der Betrachtung der Prozessqualität (wenn das Modelloriginal ein Geschäftsprozess ist) und der Betrachtung der Modellqualität dargestellt.

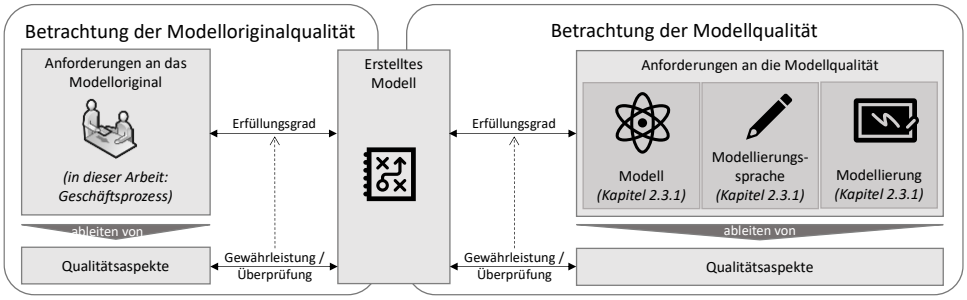


Abbildung 2-10: Betrachtung Modellqualität

Die Betrachtung der Modellqualität wird in dieser Arbeit beschrieben, da aufbauend auf den definierten Qualitätsaspekten an ein Geschäftsprozessmodell, abgeleitet von der Betrachtung der Modellqualität, Anforderungen an die generierten Geschäftsprozessmodelle gestellt werden können. Diese Anforderungen sollen anschließend in der Methode berücksichtigt werden. Diese Vorgehensweise unterstützt zum einen die Einhaltung der definierten Qualitätsaspekte bei der Generierung der Geschäftsprozessmodelle. Zum anderen können die generierten Geschäftsprozessmodelle in Bezug zu den definierten Qualitätsaspekten im Rahmen der Evaluation bewertet werden. Dadurch können die generierten Geschäftsprozessmodelle nach der Erfüllung der definierten Qualitätsaspekte eingeordnet und diese Einordnung für Rückschlüsse auf die Nützlichkeit der generierten Modelle verwendet werden.

Die Vorgehensweise zur Einbeziehung der Qualitätsaspekte in die Prozessmodellgenerierung ist in Abbildung 2-11 dargestellt. In Kapitel 5 werden aus diesen Qualitätsaspekten die Anforderungen für die Prozessmodellgenerierung abgeleitet.

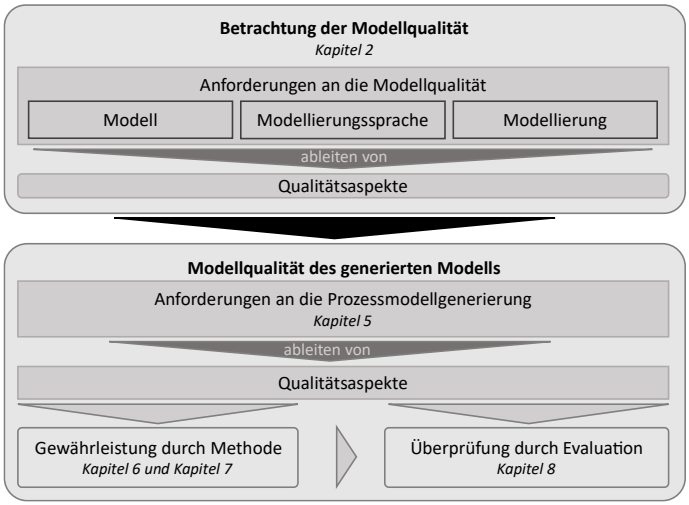


Abbildung 2-11: Vorgehensweise zur Einbeziehung der Modellqualität in die Prozessmodellgenerierung

2.3.1 Qualität nach den Anforderungen an ein Modell

Die Betrachtung der Qualität nach den Anforderungen an ein (Geschäftsprozess-)Modell ist abhängig von dem Zeitpunkt der Betrachtung. Diese zeitpunktabhängigen Anforderungen können durch Maßnahmen der konstruktiven und analytischen Qualitätssicherungen während bzw. nach der Modellierung gewährleistet werden. Als Betrachtungszeitpunkte sind die Modellerstellung, die Modellverwendung und die Modellverwaltung zu nennen [Kobl10, Aals22]:

- *Modellerstellung*: Ziel der manuellen oder (teil-)automatisierten Modellerstellung ist ein den Anforderungen genügendes Geschäftsprozessmodell. Anhand dieser Anforderungen kann die Qualität des Geschäftsprozessmodells bestimmt werden. Dazu wird das Geschäftsprozessmodell zur Entwurfszeit betrachtet.
- *Modellverwendung*: Nachfolgend sollen drei typische Einsatzzwecke von Geschäftsprozessmodellen betrachtet werden (siehe auch Tabelle 2-2).
    - *Zur Auswahlentscheidung*: Geschäftsprozessmodelle können zur Unterstützung einer Auswahl bzgl. der Erreichung eines Ziels dienen, indem die entsprechenden Geschäftsprozessmodelle bewertet und verglichen oder simuliert werden. Außerdem können Referenzmodelle herangezo-

gen werden, um aus den verfügbareren abstrahierten Referenzlösungen, die Referenzlösung auszuwählen, die für diese Entscheidung geeignet ist. Durch die Bewertung der Qualität der Modelle (und der von den Modellen abgebildeten Lösung – die Prozessqualität) kann eine Auswahlentscheidung transparent und intersubjektiv nachvollziehbar getroffen werden.

  - *Zur Ausführung*: Geschäftsprozessmodelle können zur Ausführung in Workflowmanagement-Systemen verwendet werden. Dazu muss das Geschäftsprozessmodell weiteren Anforderungen genügen, um von diesen Systemen interpretiert werden zu können. Dabei kann das Geschäftsprozessmodell die Entscheidungen innerhalb einer Prozessinstanz beeinflussen und Interaktivitäten zu Modellnutzern bereitstellen. Hierbei kann durch Analysen zur Laufzeit die Qualität des Geschäftsprozessmodells aufgezeigt werden oder Einschränkungen definiert werden, um beispielsweise die Fehleranfälligkeit bei der Ausführung zu verringern.
  - *Zur Überprüfung bzw. Analyse:* Geschäftsprozessmodelle können zur Überprüfung des Geschäftsprozesses herangezogen werden, um Abweichungen zwischen dem Ist- und Soll-Prozess zu entdecken oder aktuelle Performanceanalysen zu berechnen.

- *Modellverwaltung*: Wurden zu Geschäftsprozessen mehrere Geschäftsprozessmodelle erstellt, da Geschäftsprozessmodelle beispielsweise stetig an den abgebildeten Geschäftsprozess angepasst werden sollten, ist ein Management der Geschäftsprozessmodelle sinnvoll. Dazu können zwischen diesen Geschäftsprozessmodellen Veränderungen erkannt, protokolliert und dahingehen beurteilt werden, ob diese eine Verbesserung bzgl. der Qualität des Geschäftsprozessmodells bzw. des abgebildeten Geschäftsprozesses darstellen.

Der Zeitpunkt der Modellerstellung entspricht dem der in dieser Arbeit fokussierten Prozessmodellgenerierung. Daher sind insbesondere die Anforderungen an ein Modell zur Betrachtung der Modellqualität entscheidend. Die zu definierenden Qualitätsaspekte bzgl. der Anforderungen an ein Modell sollen daher ausgehend von der Begriffsklärung *Modell* in Abschnitt 2.1 definiert werden. Ein Modell ist eine im Hinblick auf einen bestimmten Zweck reduzierte Darstellung eines Modelloriginals. Um dieses Ziel zu erreichen, werden in verschiedenen Arbeiten einige Anforderungen zur Qualität definiert [DOJC+93, FiHR08, ISO11] Einige dieser Anforderungen lassen sich dabei aus dem Softwarequalitätsmanagement ableiten, da diese Disziplin ebenfalls modellbasierte Verfahren zur Qualitätsförderung einsetzt.

Insbesondere werden folgende Anforderungen genannt: Konsistenz, Vollständigkeit, Korrektheit, Eindeutigkeit, Änderbarkeit, Redundanzfreiheit, entsprechende Wahl der

Genauigkeitsebene. Aus diesen Anforderungen können zu überprüfende Qualitätsaspekte abgeleitet werden. Tabelle 2-4 zeigt die Anforderungen und die zu überprüfenden Qualitätsaspekte.

Tabelle 2-4: Qualitätsaspekte nach den Anforderungen an ein Modell

Anforderung	Qualitätsaspekt
Konsistenz	Das Modell bildet keine widersprüchlichen Informationen ab.
Vollständigkeit	Das Modell bildet alle Aspekte eines Geschäftsprozesses ab, die für den Zweck relevant sind.
Redundanzfreiheit	Das Model bildet einen Aspekt eines Geschäftsprozesses genau einmal ab.
Änderbarkeit	Das Modell ist änderbar.
Korrektheit	Das Model bildet den Geschäftsprozess entsprechend der Vorstellung semantisch richtig ("valide") ab. Die abgebildeten Aspekte im Modell sind rückverfolgbar zum Geschäftsprozess.
Eindeutigkeit	Das Modell lässt keine mehrdeutige, ungewünschte Interpretation zu.
Genauigkeitsebene	Die Abstraktionsebene wurde für den Zweck entsprechend gewählt und kann ggf. verändert werden.

Einige dieser Anforderungen sind abhängig von der gewählten Modellierungssprache (und dem verwendeten Modellierungseditor). Daher können diese Anforderungen durch die Anforderungen an eine Modellierungssprache weiter spezifiziert werden. Die Qualität nach den Anforderungen an eine Modellierungssprache wird in Abschnitt 2.3.2 betrachtet.

Abhängig von der Verwendung eines Modells und der verwendeten Modellierungssprache, können für Geschäftsprozessmodelle auch Anforderungen bezüglich der Eigenschaften dieser Modelle gestellt werden. Eigenschaften, die beispielsweise für Petri-Netze gefordert werden können, sind *Lebendigkeit*, *Beschränktheit*, *Zusammenhängend* und *Soundness* [Aals98]. Solche Eigenschaften für Modelle werden daher erst nach der Auswahl und Beschreibung der Modellierungssprache in dem nachfolgenden Abschnitt 3 betrachtet.

Die in Tabelle 2-4 genannten Anforderungen an ein Modell können bei einem generierten Modell teilweise basierend auf der Datengrundlage überprüft werden. Beispielsweise kann die Vollständigkeit eines Modells überprüft werden, in dem das abgebildete Verhalten mit dem Verhalten, das die Datengrundlage vorgibt, verglichen wird. [Aals11] führt vier Qualitätsaspekte ein, um die Repräsentativität eines generierten Geschäftsprozessmodells basierend auf Eventlogs zu bewerten [Aals11]:

- *Fitness*: das Geschäftsprozessmodell soll das im Eventlog abgebildete Verhalten abbilden,
- *Precision*: das Geschäftsprozessmodell soll nur das vorhandene Verhalten im Eventlog abbilden,
- *Generalization*: das Geschäftsprozessmodell soll das Verhalten im Eventlog verallgemeinern und
- *Simplicity*: das Geschäftsprozessmodell soll so einfach wie möglich sein.

Diese vier genannten Qualitätsaspekte beeinflussen sich gegenseitig, da beispielsweise eine Generalisierung des Verhaltens zur Verminderung der Präzision führen kann. Daher stehen diese Aspekte in Konkurrenz zueinander [Aals11].

2.3.2 Qualität nach den Anforderungen an eine Modellierungssprache

Die in Abschnitt 2.1.3 beschriebenen Anforderungen an eine Modellierungssprache sind bei der Wahl der Modellierungssprache zu beachten sowie als funktionale Anforderungen an die Prozessmodellgenerierung sowie die zu entwickelnde Software aufzunehmen. Durch eine entsprechende Wahl der Modellierungssprache und der Erfüllung dieser Anforderungen bei der Entwicklung einer Software können die aus diesen Anforderungen resultierenden Qualitätsaspekte in einem Geschäftsprozessmodell gewährleistet werden.

Eine Modellierungssprache wird nach der Semiotik in die drei aufeinander aufbauenden Dimensionen Syntax, Semantik und Pragmatik unterteilt (siehe Abschnitt 2.1.1). Die Qualität eines Modells wird nach der Semiotik durch die Nichterfüllung einer Anforderung (d. h. ein Fehler), die bezüglich dieser Dimensionen definiert wurde, beeinflusst. Ein Fehler innerhalb eines Modells hat daher Einfluss auf die syntaktische, semantische oder pragmatische Qualität [LiSS94]. Dementsprechend kann ein Fehler in die folgenden drei Fehlerarten eingeordnet werden:

Tabelle 2-5: Fehlerarten in Geschäftsprozessmodellen nach der Semiotik

Fehler	Bedeutung	Beispiel
Syntax-fehler	Fehler im Aufbau des Modells	Zwei unmittelbar aufeinanderfolgende Transitionen bei Petri-Netzen.
Semantik-fehler	Falsche / Fehlende Verwendung von Modellierungselementen innerhalb von Konstrukten	Fehlende Bedingung, d.h. im Falle von Petri-Netzen eine fehlende Stelle im Vorbereich einer Transition.

Fehler	Bedeutung	Beispiel
Pragmatischer Fehler	Interpretation des modellierten Konstruktes entspricht nicht der Anforderung bzw. Realität	Ein Petri-Netz ist durch seine Größe und seine überschneidenden Kanten missverständlich.

Das SIQ-Framework unterteilt die Gewährleistung und Überprüfung der Qualität eine Modells nach diesen drei Dimensionen, die demnach abhängig von der Spezifikation der verwendeten Modellierungssprache, der Semantik von dem zu beschreibenden Geschäftsprozess und Geschäftsprozessmodell sowie von der Modellinterpretation eines Anwenders, der Pragmatik, sind. SIQ steht für "***S**imple enough to be practically applicable, yet **I**ntegrates the most relevant insights from the BPM field, while it deals with **Q**uality*". Dies bedeutet, dass das Framework einfach genug ist, um praktisch anwendbar zu sein und dennoch die relevantesten Erkenntnisse aus dem GPM-Feld integriert. Die Überprüfung der drei Dimensionen zur analytischen Qualitätssicherung der syntaktischen, semantischen und pragmatischen Qualität unterteilt sich in *Verifikation*, *Validierung* und *Zertifizierung.* Die konstruktive Qualitätssicherung dieser drei Dimensionen wird durch konstruktive Maßnahmen empfohlen [ReMR15]:

Die *Verifikation* entspricht der ex-post analytischen Qualitätssicherung zur Überprüfung der syntaktischen Qualität. Durch eine *Verifizierung* wird überprüft, ob das Geschäftsprozessmodell Verstöße gegen die syntaktischen Regeln der Modellierungssprache aufweist. Es wird zwischen statischen und verhaltensbezogenen Eigenschaften unterschieden. Statische Eigenschaften können sich auf die Verwendung der Elemente einer Modellierungssprache und Verhaltenseigenschaften auf die in Abschnitt 2.3.1 bereits genannten Eigenschaften, wie *Soundness* beziehen.

Durch die *Validierung* wird überprüft, dass ein Modell den Geschäftsprozess semantisch zutreffend beschreibt. Die semantische Qualität ist demnach abhängig davon, inwieweit die semantische Interpretation nach der Spezifikation einer Modellierungssprache, den gewünschten abzubildenden Aspekten des Geschäftsprozesses entspricht. Validität bedeutet, dass alle im Modell abgebildeten Aspekte richtig und relevant für den Zweck sind. Zusätzlich sollten alle relevanten Aspekte des Geschäftsprozesses in dem Modell abgebildet sein. [Ritt10] definiert dazu die vier Aspekte Korrektheit (siehe Tabelle 2-4), Vollständigkeit (siehe Tabelle 2-4), Relevanz und Authentizität. Authentizität bedeutet, dass das Modell ein realistisches Abbild des Geschäftsprozesses ist. Dieser Aspekt ist insbesondere von den drei ersteren genannten Aspekten beeinflusst [Ritt10].

Die *Zertifizierung* bezieht sich auf die Interpretation der Anwender des Modells. Dementsprechend wird geprüft, ob der Geschäftsprozess durch das Modell verständlich abgebil-

det wird. Die pragmatische Qualität ist daher abhängig von dem Anwender, der die Verständlichkeit des Modells bewertet.

Ziel des SIQ-Frameworks ist, ein Modell zu erreichen, das keine Syntaxfehler, Semantikfehler oder pragmatischen Fehler aufweist. Zur Überprüfung der semantischen und pragmatischen Qualität sollte das Geschäftsprozessmodell syntaktisch korrekt sein [ReMR15]. Bei einer Modellgeneration kann die Qualität des Modells durch definierte Einschränkungen bzgl. der definierten Anforderungen an ein generiertes Modell im Generierungsalgorithmus gewährleistet werden. Tabelle 2-6 fasst die Qualitätsüberprüfungen nach den Anforderungen an eine Modellierungssprache zusammen.

Tabelle 2-6: Qualitätsaspekte nach den Anforderungen an eine Modellierungssprache

Dimension	Qualitätsaspekt
Syntax	Das Modell verstößt gegen keine Syntaxregeln, die für die verwendete Modellierungssprache definiert sind.
Semantik	Die semantische Interpretation des Geschäftsprozessmodells nach der definierten Semantik der verwendeten Modellierungssprache entspricht der des abgebildeten Geschäftsprozesses. Das Modell ist valide und vollständig.
Pragmatik	Die Anwenderinterpretation des Geschäftsprozessmodells entspricht der des abgebildeten Geschäftsprozesses.

Detaillierte Vorschläge zur Überprüfung dieser Dimensionen beschreibt das 3QM-Framework von [OvBS12]. Geschäftsprozessmodelle, die durch eine semi-formale oder formale Modellierungssprache beschrieben wurden, können auf syntaktische und/oder semantische Richtigkeit anhand der definierten Regeln und definierten Ausführungsinterpretationen dieser Modellierungssprache überprüft werden [OFFD14]. Da die pragmatische Qualität von Geschäftsprozessen abhängig von der Anwenderinterpretation ist, werden von einer Modellierungssprache häufig keine Anforderungen an die pragmatische Qualität definiert. Um unter anderen die pragmatische Qualität in Geschäftsprozessen zu verbessern, existieren oftmals Richtlinien zur Modellierung. Die Anforderungen an die Modellierung, die von solchen Richtlinien definiert werden, werden im Folgenden Abschnitt 2.3.3 betrachtet.

2.3.3 Qualität nach den Anforderungen an die Modellierung

Die Qualität eines Geschäftsprozessmodell kann zusätzlich zu den genannten Anforderungen aus Abschnitt 2.3.1 und 2.3.2, nach den Anforderungen an die Modellierung

betrachtet werden. Dazu wurden die Grundsätze ordnungsgemäßer Modellierung (GoM.) von [BePV12] und die sieben Prozess-Modellierungs-Richtlinien (Seven Process Modeling Guidelines (7PMG.)) von [MeRv10] definiert. Durch die Überprüfung der Einhaltung dieser Grundsätze und Richtlinien kann ebenfalls die Qualität eines Geschäftsprozessmodells bezüglich der resultierenden Qualitätsaspekte beeinflusst werden. Diese beinhalten beide sowohl Qualitätsaspekte aus Abschnitt 2.3.1 als auch Abschnitt 2.3.2.

Die Grundsätze ordnungsgemäßer Modellierung beinhalten [BePV12]:

- *Grundsatz der Korrektheit*: Ein Modell muss sowohl syntaktisch als auch semantisch korrekt sein (siehe Abschnitt 2.3.2).
- *Grundsatz der Relevanz*: Es sollten nur genau die für den Modellierungszweck relevanten Aspekte modelliert werden (siehe Abschnitt 2.1.3 und 2.2).
- *Grundsatz der Wirtschaftlichkeit*: Wirtschaftlichkeit bedeutet in diesem Zusammenhang, dass der Aufwand für die Erstellung des Modells den zu erwarteten Nutzen nicht übersteigt. Daher sollte das Dokumentationsziel mit möglichst geringem Aufwand erreicht werden. Der Aufwand wird beispielsweise auch durch die Lebensdauer des Modells beeinflusst. Durch Modellierungsunterstützungen kann der Aufwand gemindert und der Nutzen gesteigert werden.
- *Grundsatz der Klarheit*: Das Modell sollte leicht lesbar und anschaulich und dadurch verständlich sein (siehe Abschnitt 2.3.2). Beispielsweise sollten keine überschneidenden Kanten oder Elemente existieren und eine einheitliche Flussrichtung vorhanden sein.
- *Grundsatz der Vergleichbarkeit*: Werden Modelle mit unterschiedlichen Methoden erstellt, müssen diese vergleichbar sein (beispielsweise Ist- und Soll-Modelle). Dies setzt voraus, dass die verwendeten Modellierungssprachen ähnliche Strukturen verwenden.
- *Grundsatz des systematischen Aufbaus*: Ein Modell gilt als systematisch strukturiert, wenn verschiedene Sichten in eine Gesamtsicht integriert werden können. Daher sollten beispielsweise Sachverhalte, die in unterschiedliche Perspektiven beschrieben werden und gleich sind, auch gleich bezeichnet werden.

Tabelle 2-7 zeigt die Anforderungen und die zu überprüfenden Qualitätsaspekte nach den GoM.

Tabelle 2-7: Qualitätsaspekte nach den Anforderungen an die Modellierung – GoM

Grundsatz	Qualitätsaspekt
Korrektheit	Das Modell ist syntaktisch und semantisch bezüglich der verwendeten Modellierungssprache korrekt.
Relevanz	Das Modell abstrahiert den abgebildeten Geschäftsprozess entsprechende des Zwecks des Modells.

Grundsatz	Qualitätsaspekt
Wirtschaftlichkeit	Die Kosten für den Aufwand zur Modellerstellung sind geringer als der Nutzen des Modells.
Klarheit	Das Modell ist strukturiert, übersichtlich und lesbar.
Vergleichbarkeit	Die syntaktische und semantische Vergleichbarkeit von unterschiedlichen Modellen ist gegeben.
Systematischer Aufbau	Bei der isolierten Modellierung wurden die Konsequenzen auf andere Sichten beachtet.

Die sieben Prozess-Modellierungs-Richtlinien beinhalten [MeRv10]:

- *Elementanzahl*: Es sollen so wenig Elemente verwendet werden wie möglich, da die Größe des Modells Einfluss auf die Verständlichkeit und Fehlerwahrscheinlichkeit hat.
- *Pfadmöglichkeiten*: Ein Element sollte möglichst wenig eingehende und ausgehende Kanten haben da dies die Verständlichkeit und Fehlerwahrscheinlichkeit beeinflusst.
- *Start- und ein Endereignis*: Es sollte nur ein Start- und ein Endereignis verwendet werden. Dies ist häufig auch eine Anforderung von Workflowmanagement-Systemen.
- *Strukturiertheit*: Ein Geschäftsprozessmodell ist strukturiert, wenn jeder Aufspaltungs-Konnektor mit einem entsprechenden Zusammenführungs-Konnektor desselben Typs übereinstimmt.
- *OR-Konnektor*: OR-Aufspaltungen und -Zusammenführungen sollten vermieden werden, da die Semantische Interpretation uneindeutig sein kann. Des Weiteren kann die Fehlerwahrscheinlichkeit bei Geschäftsprozessmodellen, die nur nebenläufige und alternative Ausführungen abbilden, reduziert werden.
- *Aktivitätsbezeichnungen*: Aktivitäten sollten mit dem Verb-Objekt-Stil bezeichnet werden, um Mehrdeutigkeiten zu vermeiden und die Verständlichkeit zu erhöhen.
- *Modellgröße*: Ein Modell mit mehr als 50 Elementen sollte aufgeteilt werden, da die Größe des Modells Einfluss auf die Fehler und die Verständlichkeit haben. Beispielsweise könnten Teilprozesse verwendet werden.

Tabelle 2-8 zeigt die Anforderungen und die zu überprüfenden Qualitätsaspekte nach den 7PMG

Tabelle 2-8: Qualitätsaspekte nach den Anforderungen an die Modellierung – 7PMG

Richtlinie	Qualitätsaspekt
Elementanzahl	Das Modell enthält eine verständliche Anzahl an Elementen.

Richtlinie	Qualitätsaspekt
Pfadmöglichkeiten	Ein Element enthält möglichst wenige Kontrollflussaufspaltungen.
Start- und ein Endereignis	Es wird nur ein Start- und ein Endereignis verwendet.
Strukturiertheit	Die Aufspaltungen werden mit dem gleichen Typ zusammengeführt.
OR-Konnektor	Es werden keine OR-Aufspaltungen und -Zusammenführungen verwendet.
Aktivitäts-bezeichnungen	Es werden Verb-Objekt-Aktivitäts-Bezeichnungen verwendet.
Modellgröße	Ein Modell hat nicht mehr als 50 Elemente.

Aufbauend auf diesen Richtlinien definieren [CFFG+18] weitere Anforderungen. Dazu gehören:

- Bei einer alternativen Ausführung sollte durch die Benennung der Kanten von aufspaltenden XOR-Gateways die Bedingung der Auswahl definiert werden.
- Subprozesse sollten verwendet werden, um Aktivitäten die das gleiche Ziel zu gruppieren. Insbesondere ist dies sinnvoll, wenn es Aktivitäten sind, die (i) nicht den adressierten Prozessbeteiligten betreffen, die (ii) ein anderes Ziel als der abgebildete Geschäftsprozess verfolgen oder die (iii) in einem anderen Geschäftsprozess wiederverwendet werden.

Die in diesen Kapiteln vorgestellten Anforderungen werden in Kapitel 5 weiter diskutiert, um die Anforderungen an ein generiertes Modell zu definieren. Zur Umsetzung von den Anforderungen an die automatisierte Überprüfung bezüglich der Qualität in Modellen existieren weitere Arbeiten [vgl. UFHP+21, SFHP+21, CFFG+18, KoOr20].

2.4 Zusammenfassung

In diesem Kapitel wurde die Modellierung im Kontext von Geschäftsprozessen inklusive zugehörigen Modellierungsperspektiven betrachtet. Unterschiedliche Qualitätsaspekte für Geschäftsprozessmodelle wurden beschrieben. Dafür wurden zunächst die Begriffe *Modell*, *Geschäftsprozess* und *Geschäftsprozessmodellierung* definiert. In der allgemeinen Modelltheorie wird der *Modellbegriff* unter Verwendung von den drei Hauptmerkmalen *Abbildungsmerkmal*, *Verkürzungsmerkmal* und *pragmatische Merkmal* beschrieben. Zusammenfassend ist ein Modell eine im Hinblick auf einen bestimmten Zweck reduzierte Darstellung eines Modelloriginals. Bei einem *Geschäftsprozess* werden eine Menge

von Aktivitäten nach bestimmten Regeln auf ein bestimmtes Ziel hin ausgeführt. Wird die Modellierung im Kontext von Geschäftsprozessen betrachtet, bedarf es verschiedener Anforderungen an eine Modellierungssprache, um die relevanten Eigenschaften oder Verhaltensmuster des Modelloriginals, den *Geschäftsprozess*, entsprechend abbilden zu können. Daher wurden die Eigenschaften von Geschäftsprozessen betrachtet. Die bezüglich eines definierten Ziels relevanten Aspekte eines Geschäftsprozesses können dann unter Verwendung einer grafischen Modellierungssprache in einem Geschäftsprozessmodell abgebildet werden.

Um in Geschäftsprozessmodellen unterschiedliche Aspekte zu fokussieren, können bei der Modellierung unterschiedliche Modellierungsperspektiven eingenommen werden. In der Geschäftsprozessmodellierung ist dabei insbesondere die kontrollfluss- und objekt(-fluss-)orientierte Modellierungsperspektive wichtig. Bei der kontrollflussorientierten Modellierungsperspektive steht die Reihenfolge der Aktivitäten im Vordergrund. Die Reihenfolge ergibt sich durch die kausalen Zusammenhänge zwischen den Aktivitäten. Bei der Fokussierung auf die Objekte eines Geschäftsprozesses, können in einem Modell die Funktionen und/oder Daten zu den Objekten integriert werden.

Geschäftsprozessmodelle, die manuell oder automatisiert erstellt werden, können fehlerhaft sein. Bei der manuellen Modellierung sind zum einen Modellierer und Prozessexperte oftmals unterschiedliche Personen und dies kann zu Kommunikationsproblemen bzw. Transformationseffekten führen. Zum anderen können auch eine fehlende Modellier- oder Prozessexpertise zu unzureichender Qualität bei Geschäftsprozessmodellen führen. Bei der (teil-)automatisierten Erstellung von Geschäftsprozessmodellen kann die Qualität der generierten Geschäftsprozessmodelle unzureichend sein, wenn der Algorithmus zur Generierung qualitativ unzureichende Ergebnisse liefert (beispielsweise, weil er nicht ausreichend überprüft wurde), die gewählten Anforderungen an ein Geschäftsprozessmodell nicht sichergestellt werden, oder bereits die Qualität der Datengrundlage unzureichend ist. Fehlerhafte Geschäftsprozessmodelle können jedoch bei deren Verwendung als Kommunikations- oder Entscheidungsgrundlage, zur Prozessanalyse oder zur Ausführung zu Problemen führen. Die Qualität eines Geschäftsprozessmodells bezieht sich auf den Erfüllungsgrad von definierten Anforderungen bezüglich der Gesamtheit aller Eigenschaften des Geschäftsprozessmodells. Daher wurden in diesem Kapitel ausgewählte relevante Qualitätsaspekte für Geschäftsprozessmodelle diskutiert. Bei der Betrachtung der Qualität eines Geschäftsprozessmodells wird zwischen der *Betrachtung der Modellqualität* und *Betrachtung der Prozessqualität* unterschieden. Zur Definition der Qualitätsaspekte zur Betrachtung der Modellqualität wurden die Qualitätsaspekte nach den Anforderungen an ein Modell, an eine Modellierungssprache sowie an die Modellierung betrachtet.

3 Daten für die Geschäftsprozessmodellierung

Zur Geschäftsprozessmodellierung wird insbesondere die kontrollflussorientierte Modellierungsperspektive verwendet (siehe Abschnitt 2.1.3). Bei der kontrollflussorientierten Modellierungsperspektive steht die Reihenfolge der Aktivitäten im Vordergrund. Die Reihenfolge der Aktivitäten ergibt sich durch vorhandene oder nicht-vorhandene kausale Zusammenhänge zwischen diesen Aktivitäten sowie organisatorische Vorgaben. Bei der Fokussierung auf die Objekte eines Geschäftsprozesses können in einem Modell die statischen und/oder dynamischen Aspekte der Objekte fokussiert werden (siehe Abschnitt 2.2.2). Die mit Hilfe der zu entwickelnden Methode generierten Geschäftsprozessmodelle sollen die kontrollfluss- und objekt(-fluss-)orientierte Modellierungsperspektive miteinander kombinieren. Um sowohl Informationen für den Kontrollfluss als auch für die Objekte bzw. den Objektfluss zu erhalten, sollen insbesondere Daten zu Prozessinstanzobjekten und zu prozessinstanzunabhängigen Objekten (siehe Abschnitt 2.1.2) für die Generierung betrachtet werden. Um zu analysieren, wie und welche Daten mit welcher Modellierungssprache abgebildet werden können, wird in diesem Kapitel zunächst auf die Daten zu Geschäftsprozessen (Abschnitt 3.1) und anschließend auf mögliche Modellierungssprachen (Abschnitt 3.2) eingegangen.

3.1 Daten zu Geschäftsprozessen

Um ein Geschäftsprozessmodell generieren zu können, werden in diesem Abschnitt die relevanten Daten eines Geschäftsprozesses betrachtet. Zu den relevanten Daten zählen sowohl Daten, die spezifisch einem Geschäftsprozess zugeordnet werden können, als auch Unternehmensdaten, die nicht explizit einem Geschäftsprozess eines Unternehmens zugeordnet werden können. Die Relevanz der Daten für die Prozessmodellgenerierung ergibt sich aus ihrer Bedeutung für den zu modellierenden Geschäftsprozess. Die Bedeutung der Daten für die Modellierung ist in dieser Arbeit abhängig von den daraus ableitbaren Informationen über den Kontrollfluss und über die Objekte bzw. den Objektfluss. Um die Bedeutung der Daten für die Prozessmodellgenerierung zu bewerten, werden somit die zwei Dimensionen *Informationsträger* und *getragene Information* betrachtet [EnGr11].

Informationsträger sind physische oder digitale Medien zur Datenspeicherung, beispielsweise Datenbanken oder Dateien. Eine Datenbank dient zur Speicherung und

Organisation von Daten. Eine Datei stellt eine logische Einheit von Daten dar. Die *Informationsträger* unterscheiden sich insbesondere nach den unterstützen *Datenformaten*. Ein *Datenformat* ist ein festgelegtes Schema für die Strukturierung und Darstellung von Daten. Es legt fest, wie die Daten in einem bestimmten Kontext gespeichert, verarbeitet und übertragen werden sollen. Daher geben Datenformate Konventionen zur Ordnung der Daten beispielsweise innerhalb einer Datei vor. Ein Datenformat kann die Syntax und die Semantik der Daten festlegen [HaKP11]. Da ein Datenformat vorgibt, wie die Daten strukturiert sein müssen, können Daten nach der Strukturiertheit unterschieden werden: Es gibt strukturierte, semi-strukturierte und unstrukturierte Daten.

- *Strukturierte* Daten sind Daten, die in einer festgelegten, vorstrukturierten Form gespeichert werden und einem Datenmodell unterliegen. Ein Beispiel für strukturierte Daten sind CSV-Dateien (Comma Separated Values-Dateien). Beim CSV-Datenformat stellt jede Zeile einen Datensatz dar und die Werte innerhalb eines Datensatzes sind durch Kommas getrennt. Durch diese einheitliche Form können sie beispielsweise effizient durch relationale Datenbanken verwaltet werden. In Datenbanken kann das Datenbankschema festgelegt werden, wodurch das Speichern und Verwalten dieser strukturierten Tabellen/Relationen verbessert wird. Ebenso zählen zu den strukturierten Daten auch Dateien, die den Quellcode einer Programmiersprache enthalten, wie z. B. Java Source Code (.java), Python Source Code (.py) oder JavaScript Source Code (.js).
- *Semi-strukturierte* Daten unterliegen keinem vollständig vordefinierten Datenmodell und es ist daher nur eine teilweise strukturierte Form ersichtlich. Beispiele für semi-strukturierte Daten sind XML-Dateien (Extensible Markup Language), JSON-Dateien (JavaScript Object Notation) und HTML-Dokumente (Hypertext Markup Language).
- *Unstrukturierte* Daten sind Daten, die keinem Datenmodell unterliegen und bei denen daher keine festgelegte Struktur ersichtlich ist. Datenformate für unstrukturierte Daten sind beispielsweise Textverarbeitungsformate wie Microsoft-Word-Dateien.

Für eine automatische Verarbeitung werden als *Informationsträger* oftmals strukturierte Daten bevorzugt, da diese nach festen Regeln verarbeitet werden können. Zu diesen zählen insbesondere Daten, die in formalen, künstlichen Sprachen wie Programmiersprachen oder formalen Modellierungssprachen vorliegen. Welche Daten jedoch zur Verarbeitung ausgewählt werden sollten, ist insbesondere von der Relevanz der *getragenen Information* für das Ziel der Verarbeitung abhängig.

Die relevanten *getragenen Informationen* zur Generierung eines Geschäftsprozessmodells können nach Definition 2-1 und Abbildung 2-4 Informationen, die durch Daten zu Prozessinstanzobjekten, prozessinstanzunabhängigen Objekten, Prozessinstanzen sowie weiteren Daten, die im Unternehmen vorliegen sein. Um die Relevanz der *Informations-*

träger und der *getragenen Information* für die Generierung eines Geschäftsprozessmodells und deren Abbildung darin zu bewerten, wird die *Zweckeignung* der Informationsträger und der getragenen Information zur Geschäftsprozessmodellierung untersucht. Da die Informationsträger sich insbesondere nach den unterstützten *Datenformaten* unterscheiden, wird bei der Zweckeignung der Informationsträger dieses analysiert. Zusätzlich sind jedoch auch weitere Dimensionen zu berücksichtigen, welche durch die Analyse der *Datenqualität* aufgegriffen werden. Bei der Zweckeignung der getragenen Information wird (zusätzlich zu einigen Dimensionen der Datenqualität, die sich auch auf die getragene Information beziehen) der *Informationsgehalt* analysiert. Durch diese Berücksichtigung werden auch die Einflussfaktoren der Zweckeignung von Information nach [Esch85] aufgegriffen: Informationsinhalt, formale Gestalt, Informationsverhalten, technologische Einflüsse, situative Einflüsse und Informationskosten.

Die *Datenqualität* (DQ) wird nach ausgewählten Dimensionen der DQ-Dimensionen von [WaSt96] betrachtet. Die DQ-Dimensionen werden in die vier Kategorien eingeteilt. Die Kategorien sowie die zugeteilten Dimensionen sind [WaSt96]:

- *intrinsische DQ*: Genauigkeit, Objektivität, Glaubwürdigkeit und Ansehen,
- *kontextuelle DQ*: Relevanz, Mehrwert, Aktualität, Vollständigkeit und angemessene Menge an Daten,
- *repräsentative DQ*: Interpretierbarkeit, Verständlichkeit, einheitliche Darstellung und Übersichtlichkeit,
- *Zugänglichkeits-DQ*: Zugänglichkeit und Zugriffssicherheit (in dieser Arbeit nicht betrachtet).

Die Bewertung dieser Dimensionen kann sowohl auf die Daten an sich (Informationsträger) als auch auf die getragene Information der Daten bezogen sein [RKMP+11]. Die Gesamteinschätzung anhand dieser Dimensionen wird in Tabelle 3-1 vermerkt, indem die Schwächen (○) und/oder Stärken (●) im Vergleich zu den anderen Einträgen genannt werden. Auch wenn die genannten Stärken bzw. Schwächen nur auf ausgewählte Informationsträger einer Kategorie zutreffen, werden diese dennoch genannt, da diese prinzipiell als mögliche Stärke bzw. Schwäche auftreten können. Folgend sind die Dimensionen in Bezug zu dieser Arbeit in Anlehnung an [WaSt96] und [RKMP+11] erläutert:

1. *Genauigkeit:* Daten gelten als genau, wenn sie fehlerfrei und präzise sind. Die Genauigkeit hängt davon ab, ob die in den Daten enthaltenen Informationen der Realität entsprechen. Dies wird ebenfalls dadurch beeinflusst, ob und wie mögliche Fehler identifizierbar sind.
2. *Objektivität*: Daten (bzw. Informationen) sind objektiv, wenn sie sachlich und wertfrei sind.

3. *Glaubwürdigkeit*: Daten sind glaubwürdig, wenn beispielsweise Zertifikate einen hohen Qualitätsstandard ausweisen. Informationen sind glaubwürdig, wenn die Informationsgewinnung beispielsweise transparent ist.
4. *Ansehen*: Daten bzw. daraus resultierende Informationen sind hoch angesehen, wenn die Datenherkunft, das Transportmedium und das verarbeitende System mit einer hohen Vertrauenswürdigkeit bewertet werden.
5. *Relevanz*: Daten sind relevant, wenn sie für den Anwendenden notwendige und nutzbare Informationen liefern[1].
6. *Mehrwert*: Daten haben einen Mehrwert, wenn sie bzw. ihre Information wertschöpfend sind. Damit haben Daten einen Mehrwert, wenn sie für den Verwendungszweck (neue) relevante Informationen liefern.
7. *Aktualität*: Daten sind aktuell, wenn sie die tatsächliche Eigenschaft des beschriebenen Objektes zeitnah abbilden und zeitnah verwendet werden können.
8. *Vollständigkeit*: Daten sind vollständig, wenn sie keine fehlenden Einträge haben und zu den festgelegten Zeitpunkten zur Verfügung stehen.
9. *Angemessener Umfang*: Daten sind von angemessenem Umfang, wenn die Menge der verfügbaren Daten den gestellten Anforderungen genügt.
10. *Interpretierbarkeit*: Daten sind eindeutig auslegbar, wenn sie in gleicher, fachlich korrekter Art und Weise begriffen werden.
11. *Verständlichkeit*: Daten sind verständlich, wenn sie unmittelbar von den Anwendern verstanden und für deren Zwecke eingesetzt werden können.
12. *Einheitliche Darstellung*: Daten sind einheitlich dargestellt, wenn die Daten fortlaufend auf dieselbe Art und Weise abgebildet werden.
13. *Übersichtlichkeit*: Daten sind übersichtlich, wenn genau die benötigten Daten und zugehörigen Informationen in einem passenden und leicht fassbaren Format dargestellt sind.
14. *Zugänglichkeit*: Daten bzw. Informationen sind zugänglich, wenn sie anhand einfacher Verfahren und auf direktem Weg für den Anwender abrufbar sind.

Für die Bewertung der Daten anhand der DQ-Dimensionen in Tabelle 3-1 werden einige Datenträgertypen zu Datenträgerkategorien aggregiert. Als ihre Bewertungsgrundlage sind die Standards für die jeweils genannten Beispiele verwendet worden.

Beim *Informationsgehalt* wird zunächst der Informationsgehalt zur *Prozessmodellierung* betrachtet. Der *Informationsgehalt zur Prozessmodellierung* wird durch die Funktion der

[1] Es wird hier der konkrete Zusammenhang von Daten und deren notwendigen und nutzbaren Informationen *für den Anwender* betrachtet. Der Anwender ist die Person oder der Roboter, der ein Geschäftsprozessmodell generiert haben möchte. Im Unterschied dazu steht die Gesamtbetrachtung der Relevanz der Informationsträger und der getragenen Information *für die Prozessmodellgenerierung und Abbildung* der Daten und Information durch Betrachten der Zweckeignung dieser.

Information für die Prozessmodellgenerierung bewertet. Es wird dabei betrachtet, über welche zu modellierende Aspekte eines Geschäftsprozesses die Daten Informationen enthalten. Demnach unterteilen sich die Funktionen der Informationen nach den betrachteten Aspekten eines Geschäftsprozesses in Abschnitt 2.1.2:

- *Zielinformation*: Die Daten geben Informationen zu dem Ziel eines Unternehmens, eines Geschäftsprozesses oder eines Produktes bzw. einer Serviceleistung. Abgeleitet davon können beispielsweise technische Anforderungen definiert werden. Ebenso beeinflussen die ökonomischen oder ökologischen Ziele, wie beispielsweise bzgl. der Zeit oder Qualität, die Anforderungen an einen Geschäftsprozess.
- *Aktivitätsinformation*: Die Daten geben Informationen zum Geschäftsprozess, indem beispielsweise Geschäftsregeln definiert, oder auch Software-Tools festgelegt werden. Ebenso können die Daten Informationen zur Zeit- oder Ressourcenplanung geben. Dazu zählen beispielsweise auch Prozessbeschreibungen oder Prozessdaten wie Eventlogs, die entstehen, wenn in einem Geschäftsprozess beispielsweise IT-Systeme verwendet werden.
- *Objektinformation*. Die Daten geben Informationen zu den Objekten, die im Geschäftsprozess erzeugt, gelesen, verbraucht oder bearbeitet werden. Objektinformationen können dabei über den Geschäftsprozess hinaus vorliegen. Dazu zählen beispielsweise bei einem Produkt als Objekt auch Informationen zum Produktlebenszyklus, die sich auf das Objekt vor oder nach dem Geschäftsprozess beziehen.

Da ein Geschäftsprozessmodell generiert werden soll, das die Kontrollfluss- und Objekt-Modellierungsperspektive miteinander kombiniert, wird zusätzlich der *Informationsgehalt zur Objektmodellierung* bewertet. Dazu wird betrachtet, inwiefern die Daten bzw. die Information das Objekt eines Geschäftsprozesses betreffen. Der Informationsgehalt wird nach den in Abschnitt 2.2.2 definierten Perspektiven der Objektmodellierung kategorisiert: Objektstruktur, Objektbeziehung, Objektzustand, Objektfluss.

Die beschriebene Bewertung der Informationsträger und der getragenen Information wird in Abbildung 3-1 verdeutlicht.

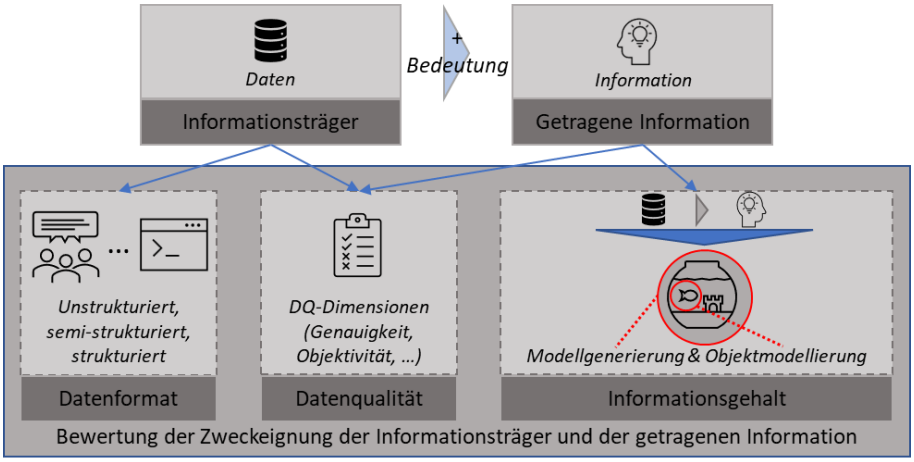


Abbildung 3-1: Bewertungsstruktur der Zweckeignung zur Prozessmodellgenerierung

In Tabelle 3-1 werden einige Daten-Kategorien als *Informationsträger* inklusive unterstützter Datenformate sowie deren getragene Informationen für die Prozessmodellgenerierung betrachtet und bezüglich der Zweckeignung für die Geschäftsprozessmodellierung bewertet[2].

Tabelle 3-1: Informationsträger und getragene Informationen von Daten für Geschäftsprozesse sowie deren Bewertung

Legende: *Objektstruktur*: OS, *Objektbeziehung*: OB, *Objektzustand*: OZ, *Objektfluss*: OF; *Stärken: ●; Schwächen: ○*

Informationsträger (Datenformate)	**Getragene Information**	**Bewertung der Zweckeignung**
Dokument zu Objektinstanzen (Word, PDF, XML, PPT, …)	Bestellung, Rechnung, Lieferschein, Auftragsvertrag bzgl. Produkt, Software-Code als Produkt, …	**Datenformat:** Unstrukturiert bis semi-strukturiert **Datenqualität:** ● *Relevanz, Aktualität, Objektivität, einheitliche Darstellung, Mehrwert* ○ *Ansehen, Zugänglichkeit* **Informationsgehalt zur Prozessmodellierung:** Objektinformationen, Aktivitätsinformation **Informationsgehalt zur Objektmodellierung:** OS, OB, OZ, OF

[2] Eine weitere Betrachtung von Daten zur Modellgenerierung findet sich in [Reim23].

Legende: *Objektstruktur*: OS, *Objektbeziehung*: OB, *Objektzustand*: OZ, *Objektfluss*: OF; *Stärken: ●; Schwächen: ○*

Informationsträger (Datenformate)	**Getragene Information**	**Bewertung der Zweckeignung**
Dokument zu Objekttyp (Word, PDF, XML, PPT, …)	Produktlebenszyklusbetreffende-Dokumente	**Datenformat:** Unstrukturiert bis semi-strukturiert **Datenqualität:** ● *Aktualität, Objektivität* ○ *Relevanz, Ansehen, Mehrwert* **Informationsgehalt zur Prozessmodellierung:** Objektinformationen **Informationsgehalt zur Objektmodellierung:** OS, OB, OZ, OF
Prozessinstanzbetreffendes Dokument (Excel, Word, …)	Zeitplanung, Ressourcenplanung, …	**Datenformat:** überwiegend semi-strukturiert **Datenqualität:** ● *Aktualität, Ansehen, Zugänglichkeit, Verständlichkeit, Relevanz* ○ *Objektivität, Interpretierbarkeit, angemessener Umfang, Mehrwert* **Informationsgehalt zur Prozessmodellierung:** Aktivitätsinformation, Objektinformationen, Zielinformationen **Informationsgehalt zur Objektmodellierung:** OZ, OF
Prozessinstanzbetreffende Daten (XML, JSON)	Eingabe- und Ausgabendaten beispielsweise für Maschineneinstellungen, Sensordaten, Eventlogs von Prozess-IT-Systemen	**Datenformat:** semi-strukturiert bis strukturiert **Datenqualität:** ● *Relevanz, Aktualität, Objektivität, einheitliche Darstellung, Glaubwürdigkeit* ○ *Ansehen, Zugänglichkeit, Verständlichkeit* **Informationsgehalt zur Prozessmodellierung:** Aktivitätsinformation, (Objektinformationen) **Informationsgehalt zur Objektmodellierung:**

Legende: *Objektstruktur*: OS, *Objektbeziehung*: OB, *Objektzustand*: OZ, *Objektfluss*: OF;
Stärken: ●; Schwächen: ○

Informationsträger (Datenformate)	**Getragene Information**	**Bewertung der Zweckeignung**
		(OS, OB, OZ, OF)
Geschäftsprozesstyp-betreffendes Dokument (Word, PDF, XML, PPT, JPEG…)	Textuelle oder grafische Prozessbeschreibungen, Use-Case-Beschreibungen, Geschäftsregeln, …	**Datenformat:** Unstrukturiert bis semi-strukturiert **Datenqualität:** ● *Relevanz, Mehrwert* ○ *Aktualität, Ansehen, Objektivität, Genauigkeit, Verständlichkeit, Vollständigkeit* **Informationsgehalt zur Prozessmodellierung:** Aktivitätsinformation, Zielinformationen **Informationsgehalt zur Objektmodellierung:** OS, OB, OZ, OF
Unternehmenszugehöriges Dokument (Word, PDF, XML, PPT, …)	Unternehmensrichtlinien, Gesetze, Anforderungen, Rahmenverträge zu Lieferanten, Beschaffungsaufträge, Serviceaufträge, System-Architekturmodelle, Organigramme, …	**Datenformat:** Überwiegend unstrukturiert **Datenqualität:** ● *Relevanz* ○ *Aktualität, Ansehen, Objektivität, angemessener Umfang, Mehrwert, Verständlichkeit, einheitliche Darstellung* **Informationsgehalt zur Prozessmodellierung:** Zielinformationen, Aktivitätsinformationen **Informationsgehalt zur Objektmodellierung:** -
Unternehmensdaten (Code-, Datenbank-, Eventlogformate)	Eventlogs oder Code von Unternehmens-IT-Systemen oder der Web-Präsenz des Unternehmens, Daten zu Kunden, Produkten, Logistik, Bestellungen	**Datenformat:** semi-strukturiert bis strukturiert **Datenqualität:** ● *Relevanz, Aktualität, Objektivität, einheitliche Darstellung, Glaubwürdigkeit, Mehrwert* ○ *Ansehen, Zugänglichkeit, Verständlichkeit, angemessener Umfang* **Informationsgehalt zur Prozessmodellierung:**

Legende: *Objektstruktur*: OS, *Objektbeziehung*: OB, *Objektzustand*: OZ, *Objektfluss*: OF; *Stärken: ●; Schwächen: ○*

Informationsträger (Datenformate)	**Getragene Information**	**Bewertung der Zweckeignung**
		Aktivitätsinformation, (Objektinformationen) **Informationsgehalt zur Objektmodellierung:** OS, OB, OZ, OF
Kommunikationsdaten	E-Mails, Chats, Kommentare	**Datenformat:** Unstrukturiert **Datenqualität:** ● *Relevanz, Aktualität* ○ *Ansehen, Objektivität, Zugänglichkeit, Verständlichkeit, angemessener Umfang, einheitliche Darstellung, Übersichtlichkeit, Genauigkeit* **Informationsgehalt zur Prozessmodellierung:** Aktivitätsinformation, Objektinformationen, Zielinformationen **Informationsgehalt zur Objektmodellierung:** OZ, OF

Die Bewertung der Informationsträger und getragenen Informationen von Daten für die Prozessmodellgenerierung in Tabelle 3-1 sind danach zu betrachten, wie sie zum einen die Prozessmodellgenerierung unterstützen und zum anderen, ob sie in einem Geschäftsprozessmodell abgebildet werden sollten, um zusätzliche relevante Aspekte entsprechend dem Verwendungszweck abzubilden. Tabelle 3-1 zeigt, dass hierfür insbesondere Daten zu Objektinstanzen geeignet sein könnten, da diese sowohl Aktivitäts- als auch Objektinformationen sowie Informationen zu allen Objektmodellierungsperspektiven enthalten können. Zusätzlich könnte mit ausgewählten Anforderungen die Datenqualität der Daten zu Objektinstanzen für eine Generierung ausreichend sein. Die zu treffenden Anforderungen werden in Kapitel 5 betrachtet. Zusätzlich zu den Daten zu Objektinstanzen könnten jedoch auch noch weitere Daten als Informationsbasis für die Prozessmodellgenerierung verwendet werden, um beispielsweise den beobachtbaren Kontrollfluss zu ermitteln. Ob und ggf. welche weiteren Daten relevant für die Prozessmodellgenerierung sind, wird in Kapitel 6 betrachtet.

3.2 Objektorientierte Modellierungssprachen zur Geschäftsprozessmodellierung

Liegt bei der Geschäftsprozessmodellierung der Fokus auf der Reihenfolge von Aktivitäten, wie in Kapitel 2 beschrieben, wird eine kontrollflussorientierte Modellierungsperspektive verwendet. Eine kontrollflussorientierte Modellierungsperspektive konzentriert sich auf die Darstellung der Reihenfolge von Aktivitäten, während die Modellierung von Objekten weniger fokussiert wird.

Zur Modellierung von Geschäftsprozessen werden insbesondere graphbasierte Sprachen eingesetzt [Wesk07]. Zu diesen Modellierungssprachen zählen Petri-Netze, EPK, UML-AD und BPMN (siehe Abschnitt 2.1.3). Diese Modellierungssprachen unterscheiden sich hinsichtlich der angebotenen Notationselemente, der grafischen Darstellung dieser Notationselemente, der syntaktischen Regeln sowie ihrer Ausführungssemantik. Um in dieser Arbeit ein Geschäftsprozessmodell zu generieren, das sowohl den Kontrollfluss als auch Informationen zu Objekten abbildet, sind in diesem Abschnitt Modellierungssprachen im Hinblick auf ihre Fähigkeit zur Abbildung von Objektstrukturen, Objektzuständen, Objektbeziehungen und Objektflüssen zu betrachten. Wie in Abschnitt 3.1 erarbeitet, sollen insbesondere Daten zu den Objektinstanzen eines Geschäftsprozesses betrachtet werden.

Um die kontrollflussorientierte Modellierungsperspektive mit der objektorientierten Perspektive zu kombinieren, berücksichtigen Geschäftsprozessmodelle Eigenschaften von Objekten zur Erfüllung von Bedingungen in Geschäftsprozessen [AHKB03] oder zur Definition von Aktivitätsfolgen aus Objektabhängigkeiten [MeSW11]. Die Ausführungsbedingung einer Aktivität kann dann als Funktion mit Bezug auf ein Objekt definiert werden [MeSW11].

Die Möglichkeiten solcher Geschäftsprozess-Modellierungssprachen, Objekte abzubilden, unterscheiden sich in Syntax und Semantik. BPMN stellt Elemente zur Darstellung von Dokumenten sowie Datenbanken bereit [Obje13b], während die erweiterte EPK Informationsobjekte bereitstellt, um mittels Daten Objekte abzubilden, die für die Durchführung von Funktionen benötigt werden. Zusätzlich können diese Informationsobjekte mittels eines Informationsflusses verbunden werden, um den Datenfluss zwischen den Informationsobjekten darzustellen [NüRu02]. Bei den UML-AD können ebenfalls Objekt-Elemente modelliert werden, die mit einer Aktivität über eine Kante verbunden wird, um den Datenfluss abzubilden. Ein Objekt-Element enthält Marken, die einzelne Objekte in einem bestimmten Zustand darstellen [Obje17]. Diese Objekt-Elemente werden nur mit einem Namen und beispielsweise einen Zustand für die Marken beschrieben. Eine weitere Beschreibung der Objekte ist nicht möglich. [RDHM11] erweitern die UML so, dass Relationen zwischen den Geschäftsprozessmodellen (als UML-AD) und den Da-

tenmodellen (als UML-KD) spezifiziert werden können. Durch diese Relationen werden die Modelle lose gekoppelt, ohne die Definition von Abhängigkeiten zu ermöglichen. Auch für BPMN existiert eine Erweiterung, um die dargestellten Datenbanken mit Datenmodellen (als UML-KD) zu spezifizieren und zusätzlich auf Daten-Operationen bei einer Aktivität einzugehen [COWZ21]. Bei den genannten Modellierungssprachen werden jedoch nur wenig formale semantische und syntaktischen Regeln definiert, die erforderlich sind, um den Objektfluss präzise zwischen Aktivitäten und die Objektzustände abzubilden [MeSW11]. In markierten Petri-Netzen können Marken als Repräsentationen von Objekten definiert werden. Des Weiteren können in höheren Petri-Netzen durch die unterscheidbaren Marken detaillierte Informationen zu Objekten beschrieben werden. Durch die Transitionen können zusätzlich Regeln für das Erzeugen, Lesen, Bearbeiten oder Löschen von Objekten definiert werden [vgl. GeLa79, Jens81, ObSa96, Weit98, LeOb03, BaHM15].

Eine Gegenüberstellung der Modellierungssprachen, bezogen auf die Darstellung von den Aspekten eines Geschäftsprozesses, ist in Tabelle 3-2 gegeben. Die betrachteten Aspekte sind Aktivitäten, Ereignisse, Objekte sowie der Kontroll- und Objektfluss. Die Aspekte wurden anhand der Sprach-Spezifikationen und anderen Arbeiten zur Sprach-Bewertung betrachtet [vgl. WKKL19].

Tabelle 3-2: Darstellung der Aspekte eines Geschäftsprozesses durch Modellierungssprachen

Aspekt	**Petri-Netze**	**eEPK**	**UML-AD**	**BPMN**
Aktivität	Transition	Aktivität	Aktivität	Aktivität
Ereignis	Stelle[3]	Ereignis	Signal	Ereignis
Kontrollfluss	Kante	Kante / Konnektor	Kante/ Entscheidungsknoten	Kante/ Gateway
Objekt	Marke	Informations-Objekt	Objektknoten	Artefakt
Objektfluss	Kante	Nichtstrukturbildende Kante	Kante	Assoziation

Dieser Vergleich zeigt, dass alle ausgewählten Modellierungssprachen Objekte darstellen können und es einige Ansätze gibt, die die kontrollflussorientierte Modellierungsperspek-

[3] Bei Bedingungs-/Ereignis-Netzen (siehe Kapitel 3.2.1) sind Ereignisse durch Transitionen abgebildet und Stellen repräsentieren Bedingungen. Bei der Geschäftsprozessmodellierung können Stellen eingetroffene Ereignisse repräsentierten (beispielsweise *Bestellungsanfrage erhalten*).

tive mit der objektorientierten Modellierungsperspektive kombinieren. Diese Kombination erfolgt oftmals durch die Integration von Daten in ein Geschäftsprozessmodell. Solche Ansätze werden mit den Begriffen *data-aware*, *data-centric* oder *data-driven process management* beschrieben [PWOB19, SMAL+19]. Jedoch gibt es insbesondere in der Ausdrucksmächtigkeit Unterschiede. Hier bieten die höheren Petri-Netze eine formale Definition an. Des Weiteren können durch die unterscheidbaren Marken detaillierte Informationen zu Objekten beschrieben und durch die Transitionen auch verändert werden. Dies ist bei den anderen Modellierungssprachen nicht in vergleichbarer Granularität möglich [MeSW11][4]. In höheren Petri-Netzen können sowohl Objektstrukturen, Objektbeziehungen, Objektzustandsänderungen und Objektflüsse spezifiziert werden. Durch die unterscheidbaren Marken und die Beschreibung von Kontroll- und Objektfluss kann ein Petri-Netz sowohl einen Geschäftsprozesstyp als auch die Instanzen des Geschäftsprozesstyps abbilden.

Daher sollen in dieser Arbeit die Geschäftsprozessmodelle in Form von höheren Petri-Netzen generiert werden. Im Folgenden wird zunächst auf Petri-Netze (Abschnitt 3.2.1) und anschließend auf höhere Petri-Netze (Abschnitt 3.2.2), in denen Objekte repräsentiert werden können, eingegangen.

3.2.1 Petri-Netze

Petri-Netze werden als graphbasierte Modellierungssprache zur grafischen Modellierung von Geschäftsprozessen verwendet. Ein Petri-Netz ist ein Graph, der aus zwei Arten von Knoten besteht, den *Stellen* und den *Transitionen*. Stellen (dargestellt durch Kreise) und Transitionen (dargestellt durch Rechtecke) sind durch gerichtete Kanten miteinander verbunden. Damit ergibt sich die folgende Definition 3-1.

Definition 3-1: Petri-Netz

Ein Tripel $N = (S, T, F)$ heißt Petri-Netz, falls gilt:

i. S und T sind disjunkte Mengen
ii. $S \cup T \neq \emptyset$
iii. $F \subseteq (S \times T) \cup (T \times S)$ ist eine zweistellige Relation, die Flussrelation von N.

Die Elemente aus S heißen Stellen, die aus T Transitionen. Für $x \in S \cup T$ heißt

- $\bullet x = \{y \in S \cup T \mid (y, x) \in F\}$ der Vorbereich und
- $x \bullet = \{y \in S \cup T \mid (x, y) \in F\}$ der Nachbereich von x.

[Reis86]

[4] Bei den von [MeSW11] betrachteten Modellierungssprachen unterstützt lediglich *Corepro* [MüRH08] die in dieser Arbeit gewünschte kombinierte Modellierungsperspektive (da keine höheren Petri-Netzen betrachtet wurden). Diese Modellierungssprache fokussiert bei der Modellierung jedoch den Objekt-Lebenszyklus und vernachlässigt daher die Objektstruktur. Daher wurde sie in dieser Arbeit nicht ausgewählt.

Bei *markierten* Petri-Netzen weist die Markierung $M: S \longrightarrow \mathbb{N}_0$ jeder Stelle eine Anzahl von abstrakten, nicht unterscheidbaren Marken (dargestellt durch schwarze Punkte) zu. $M(s)$ beschreibt die Anzahl der Marken in einer Stelle *s* unter einer Markierung M. Die Anfangsmarkierung wird mit M_0 bezeichnet [Reis86]. Bei markierten Petri-Netzen wird zwischen den Bedingungs-/Ereignis-Netzen (B/E-Netze) und Stellen-/Transitions-Netzen (S/T-Netze) differenziert:

- *B/E-Netze*: Die Stellen werden als Bedingungen und Transitionen als Ereignisse interpretiert. Eine Stelle *s* ist markiert ($M(s) = 1$; Bedingung trifft zu) oder unmarkiert ($M(s) = 0$; Bedingung trifft nicht zu). Wenn für eine Markierung M jede Stelle im Vorbereich einer Transition *t* markiert ($\forall s \in {}^\bullet t: M(s) = 1$) und jede Stelle im Nachbereich dieser Transition unmarkiert ($\forall s \in t^\bullet: M(s) = 0$) ist, ist die Transition *t* aktiviert unter M. Wenn eine Transition aktiviert ist, kann sie gemäß der Schaltregel schalten: im Vorbereich der Transition werden die Marken der Stellen entnommen und im Nachbereich der Transition wird jeweils eine neue Marke eingefügt. Somit ermöglicht die Anwendung der Schaltregel das Simulieren des Markenflusses in einem Netz [Reis86].
- *S/T-Netze*: Die Stellen werden beispielsweise als Objektspeicher und Transitionen als Operationen interpretiert. Für Stellen können daher Kapazitäten ($K: S \longrightarrow \mathbb{N} \cup \{\infty\}$) und für Kanten Gewichte ($W: F \longrightarrow \mathbb{N}$) definiert werden. Die Markierungen eines Netzes müssen diese Kapazitäten einhalten. Eine Transition *t* ist unter M aktiviert, falls gilt: $\forall s \in {}^\bullet t: M(s) \geq W(s,t)$ und $\forall s \in t^\bullet: M(s) \leq K(s) - W(t,s)$. Beim Schalten einer aktivierten Transition *t* werden den Stellen im Vorbereich gemäß dem jeweiligen Kantengewicht Marken entnommen ($\forall s \in {}^\bullet t: M(s) - W(s,t)$) und den Stellen im Nachbereich gemäß dem jeweiligen Kantengewicht Marken eingefügt ($\forall s \in t^\bullet: M(s) + W(t,s)$) [Reis86].

Der in Abschnitt 2.1.2 beschriebene Geschäftsprozess ist in Abbildung 3-2 als markiertes Petri-Netz (S/T-Netz) dargestellt. Die Marke in der Stelle *Neue Anfrage* repräsentiert damit eine neue Anfrage. Würden in der Stelle mehrere Marken (und damit Anfragen) liegen, wären diese Marken jedoch nicht voneinander unterscheidbar. Um diese Objekte, die durch Marken abgebildet werden können, unterscheidbar darzustellen, werden Petri-Netze erweitert [vgl. GeLa79, Jens81, ObSa96, Weit98, LeOb03, BaHM15]. Erweiterte Petri-Netze werden als höhere Petri-Netze beschrieben. In Abschnitt 3.2.2. werden Varianten höherer Petri-Netze betrachtet, die sich besonders zur Abbildung von Objekten eignen.

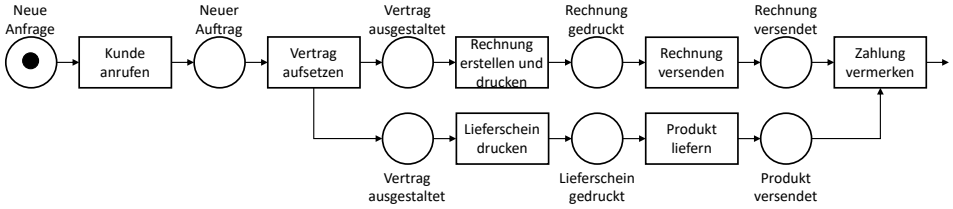
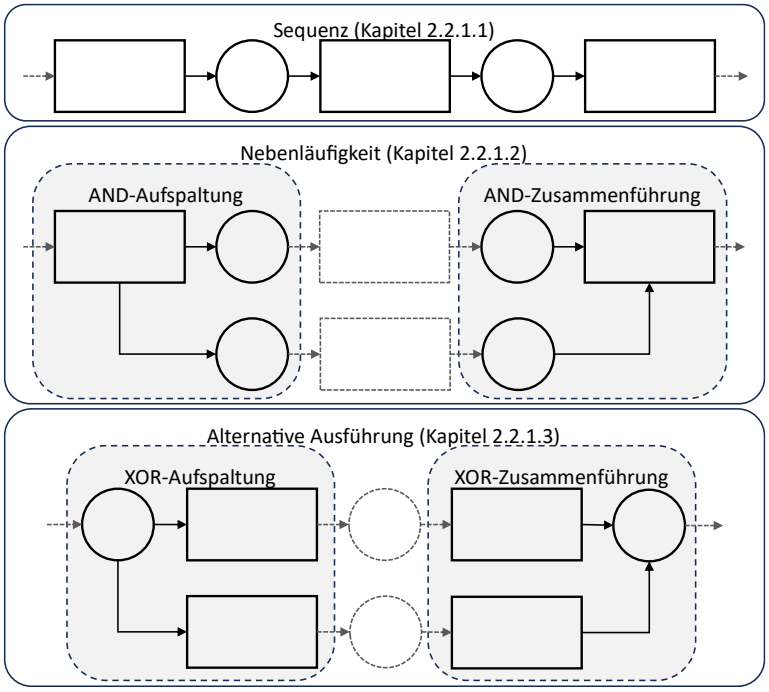


Abbildung 3-2: Beispiel eines S/T-Netzes

Die in Abschnitt 2.2.1 genannten Kontrollflussmuster *Sequenz*, *Nebenläufigkeit* und *Alternative Ausführung* sind in Abbildung 3-3 als Petri-Netze dargestellt.



[AHKB03]

Abbildung 3-3: Abbildung der Kontrollflussmuster mit Petri-Netzen

Workflow-Netze schränken Petri-Netze in ihrer Allgemeinheit strukturell ein und lassen nur Strukturen zu, die typischen Charakteristika von Geschäftsprozessen entsprechen [Aals98]. Ein Workflow-Netz (siehe Definition 3-2) hat einen eindeutigen Startpunkt (Quelle) und Endpunkt (Senke). Zudem liegt jeder Knoten des Netzes auf einem Pfad von der Quelle zur Senke.

Definition 3-2: Workflow-Netz

Ein Workflow-Netz ist ein markiertes S/T-Netz $N = (S, T, F)$, für welches gilt:
N hat zwei Stellen *i* und *o* mit folgenden Eigenschaften:

- Stelle i ist eine Quelle, d. h. $\bullet i = \emptyset$
- Stelle o ist eine Senke, d. h. $o\bullet = \emptyset$
- Wenn zu N eine neue Transition *t* hinzugefügt wird, die *o* und *i* mit $\bullet t = o$ und $t \bullet = i$ verbindet, so ist das resultierende, auch Netz $N = (S, T \cup \{t\}, F \cup \{(o, t),(t, i)\})$, stark zusammenhängend.

[Aals98]

Diese strukturellen Anforderungen verbessern die Analysierbarkeit und Verifizierbarkeit der modellierten Geschäftsprozesse. Ein wichtiges Qualitätskriterium für Workflow-Netze ist die Eigenschaft der *Soundness*. Diese formale Eigenschaft soll die korrekte Ausführbarkeit des modellierten Geschäftsprozesses gewährleisten. Ein Workflow-Netz erfüllt die *Soundness*-Eigenschaft genau dann, wenn die drei Kriterien erfüllt sind [Aals98]:

- Von jedem erreichbaren Zustand aus existiert ein Ausführungspfad zum definierten Endzustand.
- Wenn die Senke eine Marke enthält, sind alle anderen Stellen des Netzes unmarkiert.
- Für jede Transition existiert mindestens eine mögliche Ausführungssequenz, in der diese Transition schaltet.

3.2.2 Höhere Petri-Netze

Werden höhere Petri-Netze zur Repräsentation von Geschäftsprozessen verwendet, können Objekte durch Stellen und Marken im Petri-Netz weiter beschrieben werden. Bei diesen Ansätzen werden die Elemente der höheren Petri-Netze folgendermaßen interpretiert:

- *Stellen* repräsentieren Objektspeicher, bei denen das Schema vorgegeben werden kann, um Einschränkungen für die Objektstruktur und einzelne Werte festzulegen (Abbildung von Objekttypen).
- *Marken* repräsentieren die Objekte zu einem bestimmten Zeitpunkt im Prozess, entsprechend dem definierten Stellenschema (Abbildung von Objektinstanzen).
- *Transitionen* repräsentieren Aktivitäten, die Bedingungen beschreiben können, die sich auf das definierte Schema der adjazenten Stellen beziehen. Außerdem definieren sie Operationen, die eine Manipulation der durch Marken dargestellten Ob-

jekte wie erzeugen, bearbeiten und löschen ermöglichen (Abbildung von Aktivitätstypen sowie bei der Ausführung bzw. Simulation von Aktivitätsinstanzen).
- *Kanten* verbinden Stellen mit Transitionen und umgekehrt. Sie haben die Möglichkeit, Einschränkungen für die Objektstruktur und einzelne Werte anzugeben, um Transitionen zu aktivieren.

Die beschriebene Idee dieser höheren Petri-Netze wurde in mehreren Arbeiten aufgegriffen. Im Folgenden werden ausgewählte höhere Petri-Netze vorgestellt, die zur Abbildung von (semi-)strukturierten Dokumenten (die Objektinstanzen beschreiben) als Marken verwendet werden können.

In *gefärbten Petri-Netzen* haben Marken verschiedene Typen, die Farben repräsentieren (ähnlich einem Schema). Diese Farben stellen oft verschiedene Objekte wie Dokumente, Güter oder Menschen im modellierten System dar. Jede Marke einer bestimmten Farbe hat Werte, die durch Transitionen geändert werden können. Farben können mit Kanten verknüpft werden, um anzuzeigen, dass nur Marken mit einer bestimmten Farbe durch sie entnommen oder eingefügt werden. Gefärbte Petri-Netze können mit verschiedenen Merkmalen erweitert werden, um ihre Ausdrucksmächtigkeit zu erhöhen. Dazu zählt beispielsweise die Darstellungsmöglichkeit verschiedener Element-Merkmale oder Fälle, die an der Ausführung von Aktivitäten beteiligt sind, oder die Darstellungsmöglichkeit von Zeitaspekten durch die Verknüpfung von Zeit mit Marken, Stellen oder Transitionen [Jens81].

Bei *Prädikats-/Transitions-Netzen* (Pr/T-Netze) repräsentieren Stellen (Prädikate) Relationsschemata in erster Normalform Die Markierung eines Pr/T-Netzes stellt einen globalen Systemzustand dar und weist den Prädikaten entsprechend ihres Relationsschemas eine Relation zu. Transitionen repräsentieren eine Klasse von Operationen und logischen Ausdrücken bezüglich der Relationen der adjazenten Prädikate. Eine Marke wird als Werte-Tupel beschrieben, die den Relationsschemata entsprechen. In erster Normalform haben diese nur atomare Attribute. Beispielsweise könnte ein Auftrag als Relation *Auftrag(KundenId, AuftragsId, Bestellung)* beschrieben sein. Eine Marke wäre dann ein spezifischer Auftrag, beispielsweise abgebildet durch den Werte-Tupel *<k1, a1, smartphone-YX32>*. Eine Transition ist unter einer gegebenen Markierung aktiviert, wenn bestimmte Tupel (d. h. Marken) im Vorbereich vorhanden und bestimmte Tupel im Nachbereich noch nicht vorhanden sind und die Bedingung der Transition erfüllt ist [GeLa79]. In Business Objects Petri-Netzen wird das Konzept der Pr/T-Netze auf Objekte in abstrahierter Weise angewandt, um die Interaktionen zwischen Objekten zu betrachten [LoKR09]. *Nested Relation-/Transitions-Netze* (NR/T-Netze) sind eine Erweiterung von Pr/T-Netzen, die die Darstellung komplexer, hierarchischer Strukturen als Schema ermöglichen. NR/T-Netze verwenden Filtertabellen, die den Stellen im Netz zugeordnet sind, um festzulegen, welche Marken von Stellen entnommen oder in Stellen eingefügt

werden sollen. NR/T-Netze basieren auf unnormalisierten ("verschachtelten") Relationen und ermöglichen den direkten Zugriff auf Substrukturen innerhalb von Marken [ObSa96].

Bei *Strukturierten-Daten-Netzen* (StDN-Netze) wird von strukturierten Dokumenten als Marken ausgegangen, die durch Transitionen mittels Muster und Anfragen manipuliert werden können. Diese strukturierten Dokumente werden als Bäume definiert, deren Knoten in Form von Attribut-Wert-Paaren gegeben [BaHM15].

Dahingegen werden in *SGML-Netzen* Stellen mit Hilfe einer SGML-Dokumenttypdefinition (DTD) typisiert, die die Struktur eines Dokuments definiert. SGML steht für Standard Generalized Markup Language. Daher ist eine Marke in einer Stelle eine DTD-konforme Dokumentinstanz. Transitionen spezifizieren Operationen auf die Marken der Dokumentenspeicher. Stellen sind mit Dokumentenvorlagen beschriftet. Die eingehenden Kanten einer Transition wählen eine Menge von Instanzen aus, die aus den Eingabestellen gelesen werden sollen, während ausgehende Kanten Einfügungen in die Stellen im Nachbereich definieren [Weit98]. Ähnlich wie SGML-Netze repräsentieren Marken in *XML-Netzen* XML-basierte Dokumente [LeOb03] und in *JSON-Netze* JSON-beschriebene Dokumente [FSFO23]. Damit repräsentieren bei XML- sowie JSON-Netzen Marken unterscheidbare, komplex strukturierte Dokumente. Wenn eine Transition schaltet, entnimmt sie die ausgewählten Dokumente in ihrem Vorbereich und fügt neue Dokumente in ihrem Nachbereich ein. Stellen werden als Dokumentenspeicher interpretiert, die durch ein Dokumentschema typisiert sind, um die Datenintegrität zu gewährleisten. Der Dokumentenfluss wird durch diese Schemata sowie durch Dokumentenfilter und Prüffunktionen kontrolliert.

Exemplarisch wird die Struktur eines JSON-Netzes in Definition 3-3 angegeben. Ähnlich dazu sind auch die SGML-, NR/T- und XML-Netze definiert, die jedoch auf anderen Dokumentformaten basieren.

Definition 3-3: JSON-Netz

Ein JSON-Netz ist ein Tupel $JSN = (S, T, R, C, O, SI, TI, AI, M_0)$ mit

i. S, T sind endliche Mengen mit $S \cap T = \emptyset, S \cup T \neq \emptyset$.
ii. $R \cup C$ ist die Menge der eingehenden Kanten der Transitionen, wobei $R \subseteq S \times T$ die Menge der lesenden Kanten und $C \subseteq S \times T$ die Menge der konsumierenden Kanten ist und $R \cap C = \emptyset$. Für jede Transition $t \in T$ gilt:
    - $\bullet t_{read} = \{s \in S \mid (s, t) \in R\}$ entspricht der Menge aller Stellen im Vorbereich der Transition t, mit lesender Kante[5] zur Transition hin.
    - $\bullet t_{consume} = \{s \in S \mid (s, t) \in C\}$ entspricht der Menge aller Stellen im Vorbereich

[5] Lesende Kanten entsprechen nach [VoSY98] nicht-konsumierenden Kanten.

der Transition t, mit konsumierender Kante zur Transition hin.
- $•t = \{s \in S \mid (s, t) \in R \cup C\}$ entspricht der Menge aller Stellen mit einer Kante zur Transition t hin.

iii. $O \subseteq T \times S$ ist die Menge der ausgehenden Kanten der Transitionen und $t• = \{s \in S \mid (t, s) \in O\}$ entspricht der Menge aller Stellen mit einer Kante von Transition t weg.

iv. Die Stelleninschrift SI weist jeder Stelle $s \in S$ ein Dokumentenschema p_s zu.

v. Die Transitionsinschrift TI weist einer Teilmenge der Transitionen $T' \subseteq T$ eine Check-Funktion c_t mit maximal n Parametern zu, wobei n der Anzahl an Stellen im Vorbereich der Transition t entspricht ($n = |•t|$).

vi. Die Kantenbeschriftung AI weist
- jeder eingehenden Kante einer Transition $(s, t) \in R \cup C$ eine Filterbeschreibung als Dokumentfilter $p_{s,t.}$ zu.
- jeder ausgehenden Kante einer Transition $(t, s) \in O$ eine Dokument-Erzeugungsfunktion $q_{t,s}$ mit n Parametern zu, wobei n der Anzahl an Stellen im Vorbereich der Transition t entspricht ($n = |•t|$).

vii. M_0 ist die Anfangsmarkierung der Stellen in S mit JSON-Dokumenten. M(s) beschreibt die Menge der Dokumente in einer Stelle s unter einer Markierung M.

[FSFO23]

Bei einem JSON-Netz, das den in Abschnitt 2.1.2 beschriebene Geschäftsprozess abbildet, entsprechen die Stellen und Transitionen zunächst dem in Abbildung 3-2 dargestellten Petri-Netz. Zusätzlich werden den Stellen jeweils ein Schema und den Transitionen und Kanten Inschriften zugewiesen. Außerdem bilden die Marken unterscheidbare JSON-Dokumente ab. Die Stelle (*Neue) Anfrage* wird demnach durch ein *JSON-Schema* [WAHD22] beschrieben, das festlegt, dass eine Anfrage durch eine Anfrage-ID, eine Kunden-ID, den Vor- und Nachnamen einer Person, eine Anschrift einer Person, eine Auflistung der gewünschten Produkte und eine E-Mail-Adresse beschrieben werden kann (siehe Listing 3-1).

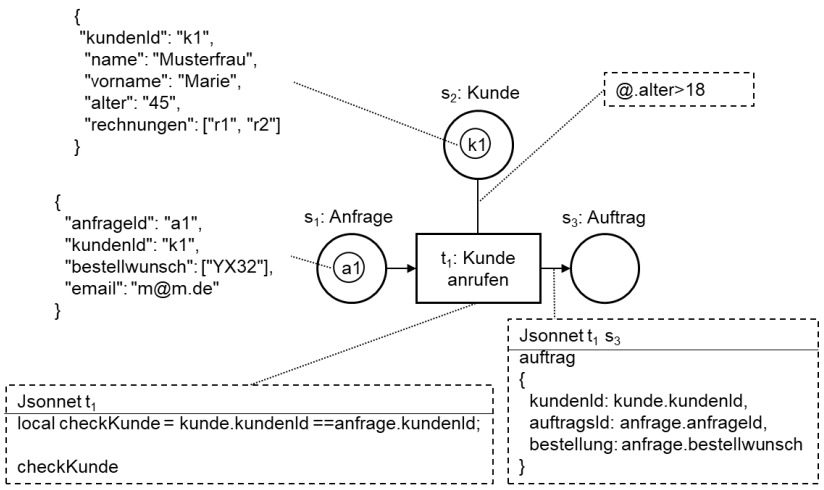
Listing 3-1: JSON-Schema einer Kundenanfrage

```
{
      "title": "Anfrage",
      "description": "Kundenanfrage zu einem Produkt",
      "type": "object",
      "properties": {
         "anfrageId": {"type": "string"},
         "kundenId": {"type": "string"},
         "nachname": {"type": "string"},
         "vorname": {"type": "string"},
         "anschrift": {"type": "string"},
         "bestellwunsch": {
            "type": "array", items": {"type": "string"}
         },
         "email": {
            "type": "string", "format": "email"
         },
```

```
17          },
18          "required": ["nachname", "email"]
19  }
```

Eine Marke in der Stelle (*Neue) Anfrage* würde einem JSON-Dokument, das zu diesem Schema valide ist, entsprechen. Zusätzlich könnten weitere Stellen eingefügt werden, um beispielsweise Datenbanken abzubilden. Durch die Abbildung der Kundendatenbank mit den Einträgen zu Kunden als Marken könnten weitere Informationen im Geschäftsprozessmodell vorliegen. Sollen in einer Datenbank die Einträge durch eine Aktivität nicht entfernt, bearbeitet oder hinzugefügt werden, ist die Stelle, die diese Datenbank repräsentiert, durch eine lesende Kante mit der Transition, die die Aktivität repräsentiert, verbunden. In der Transition *t_1: Kunde anrufen* wird daher geprüft, ob die Kunden-ID eines Kunden der Kunden-ID einer Anfrage entspricht. Zusätzlich könnte die Geschäftsregel, dass nur Anfragen bearbeitet werden, bei denen der Kunde/die Kundin über 18 Jahre alt ist, über einen Kantenfilter mittels eines *JSONPath-Ausdrucks* (@.alter > 18) [Göss07] abgebildet werden. Die Erzeugung einer Marke (beispielsweise ein Auftrag als erfolgreicher Abschluss des Kundengesprächs) wird mittels *Jsonnet* [Goog14] auf den ausgehenden Kanten einer Transition beschrieben.

Ein Ausschnitt des ursprünglichen Petri-Netzes aus Abbildung 3-2 ist in Abbildung 3-4 dargestellt.



in Anlehnung an [FSFO23]

Abbildung 3-4: Beispiel eines JSON-Netzes

3.3 Zusammenfassung

In diesem Kapitel wurde zunächst auf Daten eingegangen, die zur Geschäftsprozessmodellierung in Unternehmen existieren und zur Prozessmodellgenerierung verwendet werden können. Die Relevanz der Daten für die Prozessmodellgenerierung ergibt sich aus ihrer Bedeutung für den zu modellierenden Geschäftsprozess. Die Bedeutung der Daten für die Modellierung ist in dieser Arbeit abhängig von den daraus ableitbaren Informationen über den Kontrollfluss und über die Objekte bzw. den Objektfluss. Um die Bedeutung der Daten für die Prozessmodellgenerierung zu bewerten, wurden die zwei Dimensionen *Informationsträger* und *getragene Information* betrachtet Informationsträger sind Speichermedien wie Datenbanken oder Dateien, die sich insbesondere nach dem unterstützten Datenformat unterscheiden. Die Daten wurden bezüglich ihrer Datenstruktur, ihrer Datenqualität und ihres Informationsgehalts bewertet. Der Informationsgehalt wurde differenziert zwischen dem Informationsgehalt für die Prozessmodellgenerierung und dem für die Objektmodellierung, da ein Geschäftsprozessmodell generiert werden soll, das die kontrollfluss- und objektfluss-Modellierungsperspektive miteinander kombiniert. Durch die Bewertung der möglichen Daten wurde herausgestellt, dass insbesondere die Daten zu Prozessinstanzobjekten relevante Informationen für die Prozessmodellgenerierung bereitstellen und zur Abbildung relevanter Objekt-Aspekte in einem Geschäftsprozessmodell geeignet sind. Um die Informationen, die solche Daten liefern, ebenfalls in den Geschäftsprozessmodellen abbilden zu können, wurden einige Modellierungssprachen im Hinblick auf ihre Fähigkeit zur Modellierung von Objektstrukturen, Objektzuständen, Objektbeziehungen und Objektflüssen betrachtet. Es gibt bereits einige Modellierungssprachen, die Objekte darstellen können, und es gibt dazu auch weitere Ansätze, welche die kontrollflussorientierte Modellierungsperspektive mit der objektorientierten Modellierungsperspektive kombinieren. Jedoch gibt es insbesondere in der Ausdrucksmächtigkeit Unterschiede. Hier bieten Petri-Netze eine formale Definition an. Durch die unterscheidbaren Marken können detaillierte Informationen zu Objekten beschrieben und durch die Transitionen auch verändert werden. Dies ist bei den anderen Modellierungssprachen nicht in vergleichbarer Granularität möglich. Somit können sowohl Objektstrukturen, Objektbeziehungen, Objektzustandsänderungen und Objektflüsse repräsentiert werden. Durch die unterscheidbaren Marken und die Beschreibung von Kontroll- und Objektfluss kann ein Petri-Netz sowohl einen Geschäftsprozesstyp als auch die Instanzen des Geschäftsprozesstyps abbilden. Aufgrund dieser Ausdrucksmächtigkeit sollen in dieser Arbeit die Geschäftsprozessmodelle in Form eines Petri-Netzes generiert werden. Daher wurden in diesem Kapitel Petri-Netze definiert und Varianten höherer Petri-Netze vorgestellt, die sich zur Abbildung von Objekten eignen. Zu diesen Varianten zählen insbesondere die XML- und JSON-Netze, deren Marken XML-basierte bzw. in JSON-beschriebene Dokumente abbilden.

4 Generierungsansätze in der Modellierung

Die Modellierung von Geschäftsprozessen ist wie in Abschnitt 2.1 beschrieben wesentlich für das Geschäftsprozessmanagement. Neben der "richtigen" Wahl der Modellierungssprache, die unter anderem von dem Modellierungszweck abhängig ist, ist auch die Qualität des Modells und der Zeitaufwand für die Erstellung eines Geschäftsprozessmodells entscheidend für das Geschäftsprozessmanagement. Bei der Modellierung werden nach der Auswahl des zu modellierenden Geschäftsprozesses (das Modelloriginal) und der Definition des Modellierungszwecks, Informationen über das Modelloriginal und den Modellierungszweck gewonnen. Diese Informationen werden anschließend in einem Modell auf geeignete Weise bezüglich des Modellierungszwecks abgebildet. Anschließend kann das Modell mithilfe des Wissens über das Modelloriginal validiert werden. Daher werden im Wesentlichen zwei Rollen in der Modellierung von Geschäftsprozessen unterschieden: Der *Wissensträger*, der das Wissen über das Modelloriginal hat und für die Modellierung bereitstellt, und der *Modellierer*, der für die Modellierung des Geschäftsprozesses zuständig ist. Die Rollen können von mehreren Personen belegt sein oder eine Person kann, entsprechendes Wissen und Fähigkeiten vorausgesetzt, auch beide Rollen innehaben. Zusätzlich gibt es noch den Anwender, der ein Modell liest und verwendet [Wesk19].

Es existieren in der Forschung und Praxis einige Ansätze, die die Unterstützung und/oder (teilweise) Automatisierung der Modellierung betrachten, um beispielsweise die Zeitaufwände bei der Modellierung zu reduzieren oder eine einheitliche Qualität bei den Geschäftsprozessmodellen zu erhalten. Dabei kann die Informationsbereitstellung durch den Wissensträger durch ausgewählte verwendete Daten innerhalb eines Geschäftsprozesses unterstützt oder repräsentiert werden. Die Auswahl des zu betrachtenden Geschäftsprozesses kann beispielsweise durch eine Bewertung, welcher Geschäftsprozess automatisiert und damit auch modelliert werden sollte [JRBD19] und die Validierung eines Geschäftsprozessmodells durch die automatische Qualitätsbewertung unterstützt werden (siehe Abschnitt 2.3).

Auch die Modellierungstätigkeit des Modellierers kann durch ein Software-Tool unterstützt oder automatisiert werden, das Daten als Input erhält und entsprechend als Output ein Geschäftsprozessmodell generiert. Um diesen Schritt weiter zu betrachten, sollen

vorhandene Ansätze zur Generierung von Geschäftsprozessmodellen mittels einer systematischen Literaturrecherche (LR) gesucht und analysiert werden[1]. Diese LR orientiert sich an den Richtlinien von [KiCh07] zur Durchführung einer LR. Hierzu werden drei aufeinander aufbauende Schritte durchgeführt:

1. Planung der LR
2. Durchführung der LR
3. Betrachtung der Resultate der LR

Diese entsprechen einer Anpassung des dreistufigen Ansatzes von [TrDS03] aus dem Bereich der Medizinforschung für den Kontext der angewandten Informatik. Um weitere relevante Literatur über Quellen und Zitaten aus den ersten Resultaten der LR zu sichten, wird die Vorwärts- und Rückwärtssuche nach [WeWa02] angewandt. Anhand der Ergebnisse der LR sollen in Abschnitt 4.1 zunächst die Ansätze zur Generierung von Geschäftsprozessmodellen aus der Forschung identifiziert und darauf aufbauend in Abschnitt 4.2 die Möglichkeiten und Herausforderungen in der Prozessmodellgenerierung definiert werden.

4.1 Generierungsansätze aus der Forschung

Die LR, die in dieser Arbeit durchgeführt wird, soll relevante Ansätze zur Prozessmodellgenerierung finden. Hierdurch sollen zum einen durch die Identifikation der bestehenden Ansätze die bisherigen Möglichkeiten zur Prozessmodellgenerierung erarbeitet und in der Methode in Kapitel 6 (ggf. in adaptierter Form) berücksichtigt werden. Zum anderen sollen die Herausforderungen in der Prozessmodellgenerierung durch die gefundenen Ansätze aufgezeigt werden. Durch die Berücksichtigung der Möglichkeiten und Herausforderungen der betrachteten vergleichbaren Ansätze zur Prozessmodellgenerierung soll sichergestellt werden, dass der in dieser Arbeit entwickelte Ansatz mindestens diesbezüglich einen Mehrwert für die Prozessmodellgenerierung bietet.

Abschnitt 4.1.1 beschreibt die Planung und Durchführung der LR sowie die Organisation der Resultate und beinhaltet eine deskriptive Analyse über die Resultate. In Abschnitt 4.1.2 werden die Ergebnisse der LR beschrieben. Es gibt damit einen Überblick über Inhalte der aktuellen Forschung hinsichtlich der Prozessmodellgenerierung. Auf Grundlage der LR können in Abschnitt 4.2 die Möglichkeiten und Herausforderungen der Prozessmodellgenerierung erarbeitet werden.

[1] Die beschriebenen Schritte und Ergebnisse der LR wurden in [ScAl24] veröffentlicht.

4.1.1 Planung, Durchführung und Resultatsorganisation der LR

Bei einer LR wird in der Planung (Abschnitt 4.1.1.1) zunächst der Forschungszweck präzisiert, um damit auch die Notwendigkeit der LR nachzuweisen. Anschließend wird ein Suchprotokoll erstellt, das durch probeweises Anwenden in verschiedenen Datenbanken getestet wird. Falls erforderlich, soll insbesondere die Suchanfrage angepasst werden, um die Ergebnisse zu verbessern. Mit Abschluss dieser Planung erfolgt die Durchführung der LR (Siehe Abschnitt 4.1.1.2), bei der durch Anwenden der Suchanfrage die entsprechenden Forschungsarbeiten (im Folgenden *Publikationen* genannt) in den ausgewählten Datenbanken extrahiert werden. Basierend auf den gefundenen Publikationen werden anhand der im Suchprotokoll definierten Ein- und Ausschlusskriterien die relevanten Publikationen ausgewählt. Auf deren Basis wird eine Referenzsuche durchgeführt, um mögliche weitere relevante Publikationen zu identifizieren [WeWa02]. Diese resultierenden Publikationen werden anschließend bezüglich ihrer Qualität bewertet und ausgeschlossen, falls sie definierte Qualitätskriterien nicht einhalten. Basierend auf den verbleibenden Publikationen erfolgt die Organisation der Resultate (siehe Abschnitt 4.1.1.3). Aufbauend auf dem Ergebnis der Planung und Durchführung der LR werden die Resultate anschließend deskriptiv analysiert, klassifiziert und diskutiert (siehe Abschnitt 4.1.2). Zur Planung, Durchführung und Resultatsorganisation der LR wurde das von [HZFD12] entwickelte Softwarewerkzeug *StArt* benutzt.

4.1.1.1 Planung der LR

Der Forschungszweck der LR ist, Ansätze zur Prozessmodellgenerierung zu identifizieren, zu analysieren und zu bewerten. Außerdem sollen die Herausforderungen in der Prozessmodellgenerierung ermittelt werden. Die Ergebnisse der LR sollen bei der Entwicklung einer Methode zur Prozessmodellgenerierung berücksichtigt werden.

Ein Suchprotokoll soll die Herausforderung, die Gesamtzahl der verfügbaren Publikationen auf eine überschaubare Menge einzugrenzen lösen, möglichst ohne relevante Publikationen fälschlicherweise nicht zu identifizieren (d.h. nicht aus den Datenbanken zu extrahieren) oder in späteren Schritten auszuschließen. In diesem Suchprotokoll wird die Suchanfrage definiert: Zunächst werden die relevantesten Suchbegriffe aus dem Forschungszweck und themennahen bereits bekannten Publikationen abgeleitet. Dabei werden die geeigneten Suchbegriffe aus den Titeln, den angegebenen Schlüsselwörtern und dem Abstract identifiziert. Die relevantesten Suchbegriffe sind anschließend auf alternative Schreibweisen, Synonyme und verwandte Begriffe zu überprüfen, um ein möglichst breites Spektrum des Themenkomplexes abzudecken und auch die Variationen in der Bezeichnung der einzelnen Begriffe zu berücksichtigen [BKBT+07]. Um den semantisch richtigen Bezug zwischen den herausgearbeiteten Suchbegriffen herzustellen,

sind diese durch die booleschen Operatoren AND und OR zu verbinden. Die daraus resultierende Suchanfrage wird anschließend für die Abfrage verschiedener ausgewählter Datenbanken benutzt.

In dieser LR sollte sich die Suchanfrage sowohl auf Geschäftsprozesse, auf Modelle und auf die Automatisierung der Modellierung beziehen. Die herausgearbeiteten relevantesten Begriffe dieser Bereiche sind in Tabelle 4-1 in Deutsch und Englisch definiert. In einer Zelle befinden sich die Wörter, die oftmals synonym zueinander verwendet werden. Soll nach unterschiedlichen Wortformen gesucht werden, sind diese innerhalb einer Zelle durch Kommas getrennt.

Tabelle 4-1: Suchbegriffe der Bereiche Geschäftsprozess, Modell und Automatisierung der Modellierung

Geschäftsprozess	**Modell**	**Automatisierung der Modellierung**	
Business Process Workflow	Model Graph Flowchart Diagram Declarative	Discovery Learning Generate, Generation	
		Automated, Automation, automatically Algorithm Script Program	Constructed, Construction Support, Supporting adaption, adapted correction, corrected
Geschäftsprozess Workflow	Modell Graph, Graphbasiert, Flowchart Diagramm Deklarativ	Discovery Learning Generierung, Generation	
		Automatisierung, automatisierte, automatische Algorithmus Skript, Programm	Konstruktion, konstruieren Modellierung Unterstützen, Unterstützung Ändern, Änderung Verbessern, Verbesserung Erweitern, Erweiterung

Die zu berücksichtigenden Publikationen sollten aus allen drei Bereichen mindestens einen Begriff enthalten, um sicherzustellen, dass diese sich auf die Prozessmodellgenerierung und nicht auf angrenzende Themen beziehen. Daher wurde die Kombination der Begriffe durch eine AND-Verknüpfung zwischen den Spalten und eine OR-Verknüpfungen innerhalb einer Zelle festgelegt. Bei einer Anwendung der Suchanfrage

auf Titel, Abstrakt und Schlüsselwörtern in der Datenbank Scopus ergab dies über 35.000 Publikationen. Daher sind weitere Einschränkungen getroffen worden. Bei den Ergebnissen der ersten Überprüfung ist es auffällig, dass insbesondere Literatur aus anderen Fachgebieten angezeigt wurde. Durch die Anwendung der Abfrage auf Titel, Abstract und Schlüsselwörter sind die Begriffe ggf. nicht im direkten Zusammenhang erwähnt. Beispielsweise tritt bei der Publikation von [HALD11] mit dem Titel "*Reinforcement **learning** based resource allocation in **business process** management*" der Begriff *Modell* lediglich im Abstract bei der Nennung eines Markov-Modells zur Optimierung der Ressourcenplanung im Geschäftsprozessmanagement auf, wodurch dennoch die Bedingung der Suchanfrage erfüllt wäre, der Forschungsschwerpunkt jedoch nicht die Geschäftsprozessmodellierung ist. Daher könnte die hohe Anzahl an Publikationen aus anderen Fachgebieten in der Suchanfrage darauf zurückzuführen sein, dass die Begriffe der Spalten ohne Zusammensetzung auch in weiteren Fachgebieten verwendet werden. Insbesondere der Modellbegriff wird beispielsweise auch in den Disziplinen Physik und Medizin benutzt. Wie auch das Beispiel zeigt, wird zu Geschäftsprozessen nicht nur im Bereich der Modellierung geforscht. Um sicherzustellen, dass die Publikationen sich auf den Bereich der Geschäftsprozessmodellierung beziehen, wird die Abfrage der Begriffe aus den Bereichen *Geschäftsprozess* und *Modell* auf den Titel eingeschränkt. Die Abfrage der Begriffe aus dem Bereich der *Automatisierung der Modellierung* erfolgt weiterhin im Titel, im Abstract und in den Schlüsselwörtern. Da wie bereits in Tabelle 4-1 ersichtlich, eine Vielzahl an unterschiedlichen Wortverwendungen existiert, ist gerade in diesem Bereich eine Vollständigkeit der Begriffe unwahrscheinlich. Da im Titel eines Papiers i.d.R. nur eine Wortform des Begriffs verwendet wird, ermöglicht die Erweiterung auf Abstract, dass durch den vermehrten Textumfang ggf. mehrere Wortformen des gleichen Ausdrucks verwendet werden.

Tabelle 4-2 und Tabelle 4-3 zeigen die Suchanfragen, die sich aus dem Vorgehen ergab. Da der Großteil der relevanten Publikationen in Englisch verfasst wurde, werden die Suchanfragen sowohl in englischer als auch in deutscher Sprache spezifiziert. Die einzelnen Spalten werden mit AND verknüpft.

Tabelle 4-2: Suchanfrage in Englisch

Titel	Titel	Titel, Abstract, Keywords
Business Process OR Workflow	Model OR Graph* OR Flowchart OR	Discovery OR Learning OR generat* OR

Titel	Titel	Titel, Abstract, Keywords	
	Diagram OR declarative	Automat* OR Algorithm OR Script OR program	Construct* OR Support OR adapt* OR correct*

Tabelle 4-3: Suchanfrage in Deutsch

Titel	Titel	Titel, Abstract, Keywords	
		Discovery (Entdecken) OR Learning OR Gener* OR	
Geschäftsprozess OR Workflow	Modell OR Graph* OR Flussdiagramm OR Diagramm OR Deklarativ	Automat* OR Algorithm* OR Skript OR Script OR Programm*	Konstru* OR Modell* OR Gener* OR Unterstützen OR Ändern OR Verbessern OR Erweitern

Durch die Analyse anderer LR im Bereich der Geschäftsprozessmodellierung [vgl. FHSB+22, StDT19, MSRR15], wurden folgende relevante Literaturdatenbanken für die LR identifiziert: Scopus, ACM Digital Library, IEEE Xplore, SpringerLink, Science Direct, Web of Science, DBLP und Google Scholar.

Trotz der bereits definierten Einschränkungen war eine große Menge an Publikationen, die der Suchanfrage entsprechen, zu erwarten, die dennoch nicht dem Forschungszweck entsprechen. Daher sollte anschließend eine Bewertung der Relevanz erfolgen. Explizite Ein- und Ausschlusskriterien helfen dabei, diesen Schritt nachvollziehbar zu gestalten [PFMM08]. Jede Publikation, die mindestens ein Einschlusskriterium und kein Aus-

schlusskriterium erfüllt, wurde inkludiert. Mit den Ausschlusskriterien sollen Publikationen ausgeschlossen werden, die nicht direkt zur Bearbeitung des Forschungszwecks beitragen.

Zu den formellen Ein- und Ausschlusskriterien gehören Kriterien bezüglich der Sprache, des Dokumententyps und des Forschungsgebiets. Die Sprache der Publikationen wird auf Englisch und Deutsch eingeschränkt. Publikationen, die in Journalen, Konferenzen oder Workshops veröffentlicht werden, werden in der LR berücksichtigt. Um jedoch auch aktuelle noch nicht veröffentlichte oder unvollständige Literatur zu betrachten, werden diese unter Einhaltung ausgewählter Qualitätsaspekte ebenfalls einbezogen. Jedoch werden Reviews, Meta-Analysen oder Kommentare ausgeschlossen. Verwandte LR, die gefunden werden, werden auf deren betrachtete Resultate geprüft, jedoch nicht selbst in die Resultate miteinbezogen.

Mit den inhaltlichen Kriterien soll sichergestellt werden, dass in die LR nur themenrelevante Publikationen mit aufgenommen werden. Daher werden die Ergebnisse der LR auf relevante Forschungsgebiete eingeschränkt und auf den Zusammenhang zum Forschungszweck der LR überprüft. Des Weiteren werden die Publikationen anhand des Titels, des Abstracts und/oder des Volltextes auf das Forschungsthema geprüft: Publikationen, die sich nicht mit der Prozessmodellgenerierung befassen, werden ausgeschlossen. Damit soll die Anzahl der Resultate eingeschränkt und der Fokus der Publikationen klarer herausgestellt werden.

Insgesamt ergeben sich damit die in Tabelle 4-4 formellen und inhaltlichen Ein- und Ausschlusskriterien.

Tabelle 4-4: Ein- und Ausschlusskriterien

Einschlusskriterien	**Ausschlusskriterien**
Sprache: Englisch, Deutsch.	*Sprache*: Andere Sprachen.
Dokumententyp: Publikationen in Journalen, Konferenzveröffentlichungen und Workshopveröffentlichungen, White-Paper.	*Dokumententyp*: Sonstige Veröffentlichungen wie z. B. LR, Reviews, Meta-Analysen oder Kommentare.
Forschungsgebiete: Publikationen aus dem Fachbereich der Informatik oder der Geschäftsprozessmodellierung/ dem Geschäftsprozessmanagement.	*Forschungsgebiete*: Publikationen aus anderen Forschungsbereichen, bei denen nicht die Erfüllung des Forschungszwecks zu erwarten ist, wie z. B. dem Bioengineering, Medizin, Biochemie oder Psychologie.

Einschlusskriterien	Ausschlusskriterien
Forschungsthema: Publikationen, die Ansätze/Algorithmen für die (Teil-) Automatisierung der Erstellung von kontrollflussorientierten Geschäftsprozessmodellen vorstellen.	*Forschungsthema*: Publikation, die keine Ansätze/Algorithmen für die (Teil-) Automatisierung der Erstellung von kontrollflussorientierten Geschäftsprozessmodellen oder reine Transformationen von Modell zu Modell in der gleichen Modellierungsperspektive vorstellen.
Version: Publikation, die die aktuelle Version der Autoren darstellt.	*Version*: Publikationen, die einen Ansatz vorstellen, der bereits in einer neueren Version in einer anderen Publikation vorgestellt wird.

4.1.1.2 Durchführung der LR

Im zweiten Schritt der LR wird die Suche und Sichtung der Literatur durchgeführt. Hierfür werden die ausgewählten Literaturdatenbanken mit den in der Planung festgelegten Suchanfragen durchsucht. Gefundene Publikationen werden extrahiert und nach den Ein- und Ausschlusskriterien im Hinblick auf Relevanz bewertet.

Die LR wurde im September 2022 in den ausgewählten Datenbanken durchgeführt. Die Literaturrecherche mit der Suchanfrage aus Abschnitt 4.1.1.1 ergab eine Anzahl von 3617 Publikationen, welche die Basis für die anschließenden Überprüfungen bilden. Diese Basis wurde im Folgenden auf Duplikate untersucht, sodass 3030 verschiedene Publikationen verblieben. Anschließend wurden anhand der Ausschlusskriterien Publikationen ausgeschlossen, deren Sprache nicht deutsch oder englisch ist, deren Thema nicht dem betrachteten Forschungsgebiet zugeordnet wird (z.B. Medizin, Biochemie oder Psychologie) oder deren Thema nicht den Forschungszweck betrifft. Es verblieben 382 Publikationen, die anschließend anhand des Volltextes weiter auf Relevanz geprüft wurden. Mit den resultierenden 60 Publikationen wurde als abschließender Schritt eine Vorwärts- und Rückwärtssuche durchgeführt[2]. Dies führte zu weiteren 65 relevanten Publikationen. Insgesamt werden somit 125 Publikationen aus dem Forschungsgebiet der Prozessmodellgenerierung in die vorliegende Arbeit einbezogen[3].

Abbildung 4-1 zeigt den Ablauf der Durchführungsphase.

[2] Die Publikationen, die basierend auf Eventlogs Process Mining-Algorithmen anwenden und keine weitere Relevanz zu der vorliegenden Arbeit aufweisen, wurden bei der Vorwärts- und Rückwärtssuche ausgeschlossen. Zusätzlich wurden die für diese Arbeit relevanten Publikationen zu Process Mining-Algorithmen, die in [Rybi21] und [Emme22] betrachtet wurden, in die LR aufgenommen.

[3] Eine Übersicht der Publikationen der LR wurde in [Schü23] veröffentlicht.

Abbildung 4-1: Ablauf der Durchführungsphase

4.1.1.3 Organisation der Resultate

In diesem Abschnitt soll eine Struktur für die Bewertung und Einordnung der Resultate der LR festgelegt werden. Anhand dieser können in Abschnitt 4.1.2 die Resultate strukturiert und nachvollziehbar bewertet und eingeordnet werden, um anschließend eine Klassifikation der Resultate zu ermöglichen und vorzustellen. Da das Festlegen der Organisationsstrategie ebenfalls abhängig von den tatsächlichen relevanten Publikationen ist, erfolgt dies erst nach der Planung und Durchführung der LR.

Die Qualität der ausgewählten Publikationen der LR soll anhand ausgewählter Qualitätskriterien bewertet werden. Dazu wurden basierend auf ähnlichen Bewertungsmodellen von [AdGD21] und [DiTC21] sieben Fragen als Qualitätskriterium definiert. Die Fragen sind für die relevanten Publikationen mit *erfüllt*, *teilweise erfüllt* und *nicht erfüllt* zu beantworten. Die Interpretation dieser drei Antwortkategorien ist für jede der sechs Fragen in Tabelle 4-5 definiert.

Tabelle 4-5: Qualitätskriterien LR

	Qualitätskriterium		
	Erfüllt (1)	**Teilweise erfüllt (0,5)**	**Nicht erfüllt (0)**
	Wird eine definierte Methode verwendet?		
Q1	Die Publikation folgt einer eindeutig in der Publikation beschriebenen Methode.	Die Publikation folgt einer Methode, die aber nicht eindeutig in der Publikation beschrieben ist.	Die Publikation beruht auf keiner ersichtlichen Methode.

	Qualitätskriterium		
	Erfüllt (1)	**Teilweise erfüllt (0,5)**	**Nicht erfüllt (0)**
Q2	Ordnen die Autoren ihre Publikation eindeutig bezüglich des aktuellen Stands der Forschung ein?		
	Die Autoren stellen einen eindeutigen Bezug zum aktuellen Stand der Forschung her, der in einem eigenen Kapitel über verwandte Arbeiten ausführlich dokumentiert ist.	Die Autoren stellen Bezug zum aktuellen Stand der Forschung her, aber dies wird nicht in einem Kapitel über verwandte Arbeiten ausführlich dokumentiert.	Die Autoren stellen keinen Bezug zum aktuellen Stand der Forschung her.
Q3	Wird der Zweck oder das Ziel der Publikation erläutert und erfüllt?		
	Der Zweck/Das Ziel der Publikation ist eindeutig definiert und wird innerhalb der Publikation erfüllt.	Der Zweck/Das Ziel der Publikation ist eindeutig definiert, wird aber nicht innerhalb der Publikation hinreichend erfüllt.	Der Zweck/Das Ziel der Publikation ist nicht definiert.
Q4	Wird die Publikation evaluiert/validiert?		
	Die Publikation wird anhand einer Fallstudie (oder ähnliches) evaluiert/validiert.	Die Publikation wird anhand eines einfachen Beispiels evaluiert/validiert.	Die Publikation wird nicht evaluiert/validiert.
Q5	Zeigen die Autoren die Limitationen der eigenen Forschung in ihrer Publikation auf?		
	Die Limitationen der eigenen Forschung werden klar in einem eigenen Kapitel oder in der Diskussion der Ergebnisse aufgezeigt.	Die Limitationen der eigenen Forschung werden partiell beschrieben, aber nicht eindeutig herausgestellt.	Auf die Limitationen der eigenen Forschung wird nicht eingegangen.
Q6	Wird die Publikation in einem hochrangingen Journal veröffentlicht? / Hatte die Konferenz/der Workshop eine zur Geschäftsprozessmodellierung passenden Schwerpunkt mit einem peer-review Verfahren und einem guten Rang nach [Scha10]?		
	Die Publikation wurde in einem Q1 Journal veröffentlicht. / Die Publikation wurde auf einer/m passenden Konferenz/Workshop	Die Publikation wurde in einem Q2/Q3 Journal veröffentlicht. / Die Publikation wurde auf einer/m Konferenz/Workshop mit	Die Publikation wurde in einem Q4 Journal veröffentlicht. / Die Publikation wurde auf einer/m Konferenz/Workshop mit nur minimalen

	Qualitätskriterium		
	Erfüllt (1)	**Teilweise erfüllt (0,5)**	**Nicht erfüllt (0)**
	veröffentlicht mit Rang A.	einem Rang B veröffentlicht.	Anforderungen an die Publikationen und Rang C oder keinem Rang veröffentlicht / Konferenz/Workshop hatte keinen passenden Schwerpunkt.
	Wird der in der Publikation genannte Algorithmus technisch beschrieben?		
Q7	Der Algorithmus wurde technisch beschrieben oder als Softwarewerkzeug implementiert, das Open-Source zur Verfügung steht.	Der Algorithmus wurde teilweise technisch beschrieben oder als Softwarewerkzeug implementiert, dessen Code aber nicht Open-Source verfügbar ist.	Der Algorithmus wurde nicht technisch beschrieben, nicht als Softwarewerkzeug implementiert und steht nicht Open-Source zur Verfügung.

Zur quantitativen Gesamtbewertung der Qualität einer Publikation wird jeder Antwortkategorie eine Punktzahl zwischen 0 und 1 zugewiesen: *erfüllt* entspricht 1 Punkt, *teilweise erfüllt* entspricht 0,5 Punkten, *nicht erfüllt* entspricht 0 Punkten. Um nur qualitativ hochwertige Publikationen zu berücksichtigen, deren beschriebene Forschung nachvollziehbar und strukturiert durchgeführt wurde, werden Publikationen mit weniger als 3,5 Punkten ausgeschlossen.

Zusätzlich zur Qualitätsbewertung, sollen einige Eigenschaften der Publikationen geprüft werden, um eine strukturierte Analyse und Bewertung der Resultate zu unterstützen. Hierfür werden verschiedene Kategorien erstellt, in denen es definierte Ausprägungen gibt. Anschließend werden die zutreffenden Ausprägungen für jede einzelne Publikation festgelegt. Jede Publikation hat daher mindestens eine zutreffende Ausprägung je Kategorie, jedoch können in einzelnen Kategorien auch mehrere Ausprägungen zutreffen.

In der Kategorie *Modellierungssprache* werden die Publikationen anhand der verwendeten Modellierungssprache, in der das generierte Geschäftsprozessmodell vorliegt, eingeordnet, z.B. *Petri-Netze, BPMN, Declare*. Tabelle 4-6 gibt einen Überblick über die Ausprägungen der Kategorie Modellierungssprache.

Tabelle 4-6: Kategorie Modellierungssprache

Ausprägung	Beschreibung
Petri-Netze	Petri-Netze sind eine grafisch und mathematisch fundierte Modellierungssprache, bestehend aus Stellen, Transitionen und Kanten (siehe Abschnitt 3.2).
BPMN	Die Business Process Model and Notation (BPMN) ist ein Standard für die Geschäftsprozessmodellierung. Ein Geschäftsprozess wird mit Elementen aus fünf Kategorien dargestellt: Ablaufelemente, Daten, Verbindungselemente, Pools/Lanes und Artefakte [Obje13b].
Declare	Declare ist eine deklarative Geschäftsprozessmodellierungssprache, um auch flexible, schwachstrukturierte Geschäftsprozesse zu modellieren. Dazu wird nicht der exakte Kontrollfluss festgelegt, sondern es werden Bedingungen definiert, die für die Fortsetzung des Kontrollflusses erfüllt sein müssen [PeSA07].
Prozessbaum	Ein Prozessbaum (Process tree) ist ein gerichteter, zusammenhängender Graph ohne Zyklen. Die Blätter des Prozessbaums repräsentieren die Aktivitäten und die Verzweigungsknoten beschreiben den Zusammenhang der Aktivitäten [BuDA12].
Kausale Netze	Ein kausales Netz ist ein Graph, in dem Aktivitäten durch Knoten und die Abhängigkeiten durch gerichtete Kanten dargestellt werden. Hierbei werden nur die kausalen Abhängigkeiten abgebildet [Aals11].
Direkt-Folge-Graph	Bei einem Direkt-Folge-Graph (Directly-follows graph) werden die Aktivitäten durch Knoten dargestellt. Der Kontrollfluss ergibt sich durch gerichtete Kanten zwischen den Knoten [Aals19].
Sonstige	Eine Einordnung in Sonstige erfolgt, wenn keine der erwähnten Modellierungssprachen vorliegt.

Da in dieser Arbeit eine Methode zur Prozessmodellgenerierung entwickelt werden soll, werden die Ansätze zur Prozessmodellgenerierung auch nach dem beschriebenen *Entwicklungsstand* kategorisiert. Dieser wird in Tabelle 4-7 zusammengefasst und beschrieben.

Tabelle 4-7: Kategorie Entwicklungsebene

Ausprägung	Beschreibung
Konzept	Es wird ein Ansatz zur Prozessmodellgenerierung wissenschaftlich fundiert beschrieben.
Umsetzung	Es wird ein wissenschaftlich fundiertes Konzept zur Prozessmodellgenerierung umgesetzt.

Ausprägung	Beschreibung
Evaluation	Es wird ein wissenschaftlich fundiertes Konzept oder dessen Umsetzung zur Prozessmodellgenerierung evaluiert.

In der Kategorie *Input-Daten* werden die Publikationen nach den verwendeten Input-Daten eingeordnet. Basierend auf diesen Input-Daten wird ein Geschäftsprozessmodell generiert. Die Ausprägungen der Kategorie werden in Tabelle 4-8 beschrieben.

Tabelle 4-8: Kategorie Input-Daten

Ausprägung	Beschreibung
Software-Code	Anweisungsfolge, die von einem Computer interpretiert und ausgeführt werden kann. Diese besteht in der Regel aus Text, der in einer bestimmten Programmiersprache geschrieben ist und bestimmte Funktionen oder Berechnungen ausführen soll.
Regeln	Begriffsdefinitionen und Verhaltens- oder Handlungsanweisungen in einer Organisation. Regeln sollen Konsistenzen und Genauigkeiten von Geschäftsprozessen und -entscheidungen sicherstellen.
Tabellen	Strukturierte Darstellung von Daten oder Informationen in Form von Zeilen und Spalten. Jede Zeile entspricht dabei einem Datensatz, während die Spalten bestimmte Kategorien oder Eigenschaften dieser Daten darstellen.
Grafische Modelle	Eine im Hinblick auf einen bestimmten Zweck reduzierte Darstellung eines Modelloriginals unter Verwendung einer grafischen Modellierungssprache.
Wissensbasis	Beschreibungen von Beziehungen und Eigenschaften von Konzepten oder Entitäten in einem bestimmten Bereich. Diese dienen dazu, die Semantik von Informationen zu strukturieren und zu verstehen.
Eventlogs	Von Informationssystemen aufgezeichnete Ereignisse (z. B. die Eingabe einer Bestellung). Eventlogs setzen sich aus einer Menge von Traces zusammen. Ein Trace ist definiert als eine endliche Sequenz von Ereignissen, die derselben Prozessinstanz zugeordnet sind und die gleiche Fallbezeichnung besitzen.
Natürlich-sprachlicher Text	Text, der von einer Person beispielsweise in Deutsch oder Englisch verfasst wurde. Er unterscheidet sich von formalen Texten durch seine (weitgehend) freie Wortwahl und Syntax.

4.1.2 Resultate der LR

Im Folgenden werden die Resultate der LR vorgestellt. Zunächst wird die Qualitätsbewertung der Resultate beschrieben und ausgewählte deskriptive Eigenschaften werden

geprüft. Anschließend wird genauer auf die Publikationen eingegangen, um einen Überblick über die verwandten Arbeiten zu geben.

4.1.2.1 Qualitätsbewertung

Die Qualitätsbewertung der Publikationen ergab, dass alle bewerteten Publikationen die gestellten Anforderungen erfüllten. 31,2% der Publikationen erreichten mindestens 6 Punkte. Die niedrigste Punktzahl, die vergeben wurde, betrug 3,5 Punkte. Da der Schwellwert zur Aufnahme mit 3,5 definiert wurde, wurden alle Publikationen aufgenommen. Insbesondere die Erläuterung des Ziels der Publikationen (abgedeckt durch Frage Q3) wurde von allen Studien beantwortet (darunter von 5 Publikationen nur teilweise). Die Evaluation (abgedeckt durch Frage Q4) und die Limitationen der eigenen Arbeit (abgedeckt durch Q5) wurden in 64% bzw. 86,4% der Publikationen nicht hinreichend für die Erfüllung dieser Kriterien betrachtet.

In Tabelle 4-9 werden die genauen Resultate pro Frage angegeben.

Tabelle 4-9: Übersicht der Qualitätsbewertung

Bewertung	Q1	Q2	Q3	Q4	Q5	Q6	Q7
Erfüllt (1)	116	108	120	45	17	52	56
Teilweise erfüllt (0,5)	9	12	5	43	64	67	60
Nicht erfüllt (0)	0	5	0	37	44	6	6

4.1.2.2 Deskriptive Statistiken

In diesem Abschnitt werden die Resultate der LR nach verschiedenen deskriptiven Eigenschaften betrachtet. In Abbildung 4-2 werden die Resultate nach dem Veröffentlichungsjahr dargestellt. Die erste Veröffentlichung war 1998 zur Prozessmodellgenerierung aus Eventlogs (in der Publikation Workflowlogs genannt) [AgGL98]. Die Herausforderung, frühere Arbeiten zu finden, deren Titel, Keywords bzw. Abstracts nicht den Suchstring abdecken, wurde durch die Rückwärtssuche angegangen. Jedoch ist auch zu beachten, dass nur die jeweils aktuellen Versionen eingeschlossen wurden, wodurch ältere Publikationen derselben Autoren ausgeschlossen werden. Bei den Publikationen bis 2010 beziehen sich 18 von 26 auf Eventlogs oder natürlichsprachliche Texte. 2015 wurden die meisten Publikationen veröffentlicht. Die Statistik zeigt, dass die Forschung zur Prozessmodellgenerierung weiterhin aktiv betrieben wird und kein Rückgang erkennbar ist. Die geringere Anzahl im Jahr 2022 lässt sich darauf zurückführen, dass aufgrund des Untersuchungszeitpunktes nur Publikationen bis August (teilweise September) einbezogen wurden und auch vorher erfolgte Publikationen ggf. noch nicht vollständig in den Datenbanken verfügbar waren.

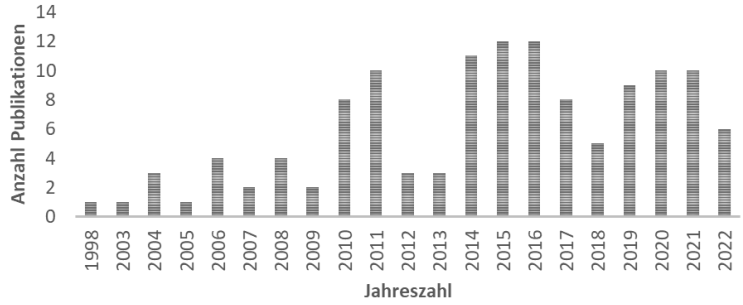


Abbildung 4-2: Publikationen nach Veröffentlichungsjahr

In Abbildung 4-3 wird dargestellt, welchen Entwicklungsebenen die Publikationen zugeordnet werden können. Sie zeigt, dass die meisten Publikationen die entwickelten Konzepte auch technisch umsetzen und evaluieren. Die Qualität der Evaluation wurde hierbei nicht berücksichtigt. Vereinzelte Publikationen stellen nur ein Konzept vor oder evaluieren dieses Konzept nur anhand eines (gedanklichen) Beispiels, ohne das Konzept umgesetzt zu haben. Außerdem existieren einige Publikationen, die trotz Umsetzung des Konzepts keine Evaluierung vornehmen.

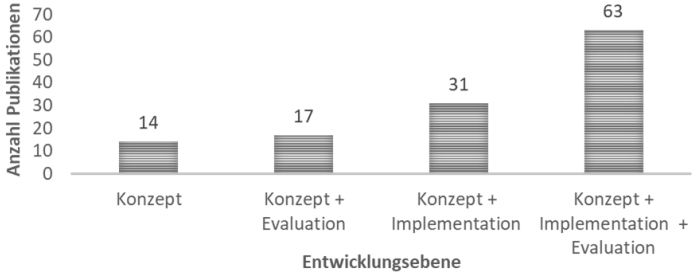


Abbildung 4-3: Publikationen nach der Entwicklungsebene

Abbildung 4-4 zeigt, dass die meisten Publikationen basierend auf Eventlogs und Text(en) Geschäftsprozessmodelle generieren. Dies lässt sich darauf zurückführen, dass diese Input-Daten eine hohe Akzeptanz finden und oftmals in Unternehmen bereits vorliegen. Im Vergleich dazu sind die Input-Daten Software Code und Wissensbasis weniger als Input-Daten vertreten. Insbesondre basierend auf Software-Code existieren jedoch einige Ansätze, die Modelle (beispielsweise Klassen- und Objektdiagramme) generieren. Diese wurden in der LR jedoch nicht berücksichtigt, da diese nicht den Geschäftsprozessmodellen zugeordnet werden können, sondern beispielsweise in der Domäne der Softwarearchitektur bzw. Softwarewartung entstanden sind und angewandt werden. Wissensbasen werden überwiegend zusätzlich oder als Zwischenschritt zur

Prozessmodellgenerierung verwendet und werden daher ebenfalls weniger als Input-Daten verwendet.

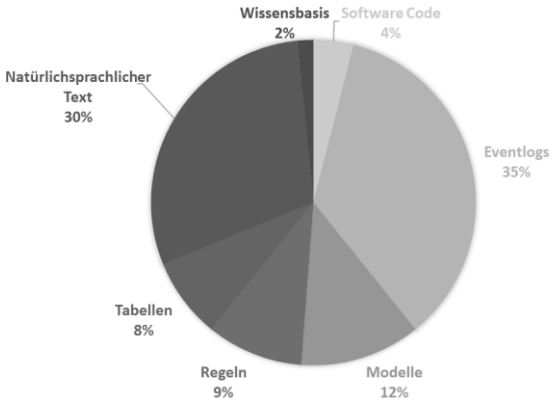


Abbildung 4-4: Publikationen nach den Input-Daten

4.1.2.3 Resultate zur Prozessmodellgenerierung

In diesem Abschnitt werden die einzelnen Ansätze der relevanten Publikationen weiter betrachtet. Um eine Unterscheidung der Ansätze zu ermöglichen, werden diese anhand des verwendeten Input-Datenformats (siehe Abschnitt 3.1 und Tabelle 4-8) strukturiert. Häufig legen Ansätze zur Datenverarbeitung insbesondere das Datenformat fest, um eine fehlerfreie Interpretation der Daten und damit die fehlerfreie Verarbeitung zu ermöglichen [GHvS+98]. Durch die Einteilung der Ansätze nach dem Input-Datenformat werden Publikationen, die ähnliche Verfahren beschreiben, um diese Daten zu verarbeiten, zusammen betrachtet und Unterschiede, in den Verfahren bezüglich der gleichen Input-Datenformate werden verdeutlicht. Außerdem können die Ansätze präziser nach den Input-Datenformaten unterschieden werden, während beispielsweise eine eindeutige Zuteilung zu einem Verfahren nicht in allen Publikationen definiert wird oder mehrere Verfahren angewandt werden. Auch eine Strukturierung nach dem Output, also anhand der Modellierungssprache, in der die Modelle generiert sind, könnte zu einer schlechten Vergleichbarkeit der einzelnen Ansätze führen. Beispielsweise würden Process Mining-Verfahren mit Verfahren, die Modelle aus natürlichsprachlichen Text generieren, gegenübergestellt werden, obwohl die Verfahren größtenteils unterschiedliche Vorgehensweisen verfolgen und daher nur bedingt miteinander vergleichbar sind. Der Output kann jedoch zur weiteren Betrachtung dienen, um die Ansätze, die auf dem gleichen Input-Datenformat basieren, voneinander abgrenzen zu können.

In fünf Publikationen werden Geschäftsprozessmodelle aus *Software-Code* erstellt [CeFB14, PDSP15, PéGP11, Alaw15, LeAa15][4]. Bei der Methode von [CeFB14] wird der Software-Code zunächst basierend auf dem zugehörigen Objektmodell in Topics und Subtopics eingeteilt und es werden deren Verbindungen identifiziert. Anschließend werden für die Topics und Subtopics die Funktionen als Graphen dargestellt. Die identifizierten Verbindungen zwischen den Topics und Subtopics werden dann verwendet, um die erstellten Graphen zu verbinden. Der Ansatz wird durch mehrmonatige Software-Entwicklungsprojekte motiviert, bei denen beispielsweise neue Entwickler eingestellt werden [CeFB14]. [PDSP15] und [PéGP11] generieren mittels Process Discovery-Algorithmus[5] aus Software-Code zunächst ein Knowledge Discovery Meta-Modell (KDM-Modell), das die einzelnen Elemente im Software-Code und deren Verbindungen identifiziert. Mittels heuristischer Regeln werden aus dem KDM-Modell die relevanten Geschäftsprozesselemente identifiziert. Diese heuristischen Regeln entsprechen implementierten Geschäftsprozesspattern, die Einschluss- und Ausschlussregeln beschreiben. Eingeschlossen werden beispielsweise Sequenzen und ausgeschlossen Datenbankzugriffe, da letztere nicht die Prozesslogik beschreiben. Anschließend werden die Geschäftsprozesselemente zu BPMN-Elementen gemapped [PéGP11, PDSP15]. [Alaw15] extrahiert aus den Klassen im Software-Code zunächst einen Abstract Syntax Tree (AST) und leitet daraus Geschäftsregeln ab. Basierend auf den extrahierten Geschäftsregeln wird dann ein Aktivitätsdiagramm und anschließend ein BPMN-Modell generiert. [LeAa15] entwickeln einen zweistufigen Ansatz zur Prozessmodellgenerierung. Im ersten Schritt werden durch die Analyse von Benutzeranfragen an System-Schnittstellen die Interaktionspunkte und deren Beziehungen erfasst. Aus diesen Informationen wird ein Eventlog generiert, der die Reihenfolge der Systeminteraktionen dokumentiert. Im zweiten Schritt wird aus diesem Eventlog mittels eines Process Discovery-Algorithmus ein Geschäftsprozessmodell abgeleitet.

In fünf Publikationen werden Geschäftsprozessmodelle basierend auf *Geschäftsregeln*, die in SBVR (Semantics of Business Vocabulary and Business Rules) beschrieben sind, generiert [RaPH08, WYHX11, TaAk14, IqBa16, KlHo16]. Mit SBVR können Geschäftsregeln und -begriffe formal beschrieben werden. Dadurch sollen Konsistenzen und Genauigkeiten von Geschäftsprozessen und -entscheidungen sichergestellt und verbessert werden [Obje13a]. Zur Generierung der Geschäftsprozessmodelle werden beispielsweise Abbildungsregeln über Metamodelle oder Transformationsregeln zwischen den Regeln und den Modellelementen definiert. [IqBa16] führen außerdem noch eine semantische

[4] In der LR waren einige Publikationen, die basierend auf Software-Code ein Modell als Sequenzdiagramm oder Klassendiagramm generieren. Da diese Publikationen jedoch keine Geschäftsprozesse abbilden, wurden diese Publikationen nicht berücksichtigt. [ShYa21] vergleichen existierende Ansätze in diesem Bereich miteinander.

[5] Process Discovery: Siehe Betrachtung bei der Inputkategorie *Eventlog*

Analyse bezüglich der Regeln durch, um die relevanten Elemente zunächst aus den Regeln zu extrahieren. Diese Ansätze generieren BPMN-Modelle und UML-Aktivitätsdiagramme. Sieben weitere Ansätze generieren ebenfalls Geschäftsprozessmodelle basierend auf Geschäftsregeln, die jedoch nicht in SBVR formuliert sind. In Tabelle 4-10 sind die Ansätze, die basierend auf Regeln Geschäftsprozessmodelle generieren, zusammengefasst.

Tabelle 4-10: Ansätze basierend auf Regeln

Ansatz	Regel-Format	Output	Verfahren
[WYHX11]	SBVR	BPMN-Modell	Metamodell
[TaAk14]	SBVR	BPMN-Modell	Transformationsregeln
[IqBa16]	SBVR	UML-Aktivitätsdiagramm	Element-Extrahierung, Transformationsregeln
[KlHo16]	SBVR	BPMN-Modell	Transformationsregeln
[RaPH08]	SBVR	UML-Aktivitätsdiagramm	Transformationsregeln
[FiAv14]	DEMO-Action-Rules	BPMN-Modell	Metamodell
[AGTW11]	Linear Temporal Logic	BPMN-Modell	Ausführung und Process Mining (Pruning-Heuristik)
[DeCh16] (basierend auf [AGTW11])	Linear Temporal Logic	Prozessbaum	Pruning-Heuristik
[JaHu11]	Defeasible Logic Programming	Eigenes Modell	Ontologie-Reasoning
[GBGO21]	Agent-oriented rule language	Petri-Netz	Transformations- und Merging-Regeln
[MrMB16]	Declare	Prozessbaum	Transformations- und Merging-Regeln
[JWBD15]	S-Declare	BPMN-Modell	Transformationsregeln zu Multi-Ziel-Problem und Suchalgorithmus

Zusätzlich dazu gibt es weitere Ansätze, die basierend auf anderen Daten solche Regeln ableiten. SBVR-Regeln können beispielsweise aus Use-Case-Diagrammen [DaSB19] oder natürlichsprachlichem Text [HJBG21, TaSS22] abgeleitet werden. Daher könnten

solche Ansätze mit den in der Tabelle genannten Ansätzen kombiniert werden, sodass Geschäftsprozessmodelle aus Input-Daten wie Use-Case-Diagrammen oder natürlichsprachlichem Text generiert werden könnten. Jedoch können zusätzliche Transformationen Informationsverluste bedeuten.

In zehn Publikationen werden basierend auf *Tabellen* Geschäftsprozessmodelle generiert. Zu diesen Tabellen zählen beispielsweise Excel-Tabellen, die Werte wie Zahlen oder Strings enthalten. Diese Tabellen bilden Use-Cases [HnCe21, Lübk06, LüSc08], Geschäftsregeln [HePv11, KYYW20], Geschäftsprozessbeschreibungen [KrDe10], Standortdaten [ZMSS18], CRM-Daten [BaBC18], Nachrichten [EAZP+12] und Aktivitätsdaten [WiKL18] ab. Bei dem Ansatz von [WiKL18] werden die Aktivitätsdaten über ein webbasiertes Tool von Prozessbeteiligten manuell eingetragen. Diese eingetragenen Aktivitäten enthalten Informationen zu Input- und Output-Daten und optionalen oder erforderlichen weiteren Aktivitäten. Bei der Generierung aus CRM-Daten werden maschinelle Lernverfahren (Latent Dirichlet Allocation (LDA)) angewandt [BaBC18].

In Tabelle 4-11 sind die Ansätze, die basierend auf Tabellen Geschäftsprozessmodelle generieren, zusammengefasst.

Tabelle 4-11: Ansätze basierend auf Tabellen

Ansatz	Tabellen-Inhalt	Output	Verfahren
[KrDe10]	Geschäftsprozess-beschreibungen	BPMN (und EPK) (Konzept)	Transformationsregeln
[HePv11]	Excel-Tabelle zur Berechnung mit Zellenverknüpfungen	Directed Graph Markup Language (Datenflussmodell)	Transformationsregeln
[Lübk06]	Use-Case	EPK	Metamodell und Regeln
[LüSc08]	Use-Case	BPMN	Metamodell und Regeln
[HnCe21]	Use-Case	UML-Aktivitätsdiagramm	Metamodell und Regeln
[ZMSS18]	Bluetooth-Standortdaten	Block-structured model	Pattern-Matching
[BaBC18]	CRM-Daten	Kausales Netz (Eventlog)	Transformation zu Bedingungen, *Process Mining*

Ansatz	Tabellen-Inhalt	Output	Verfahren
[EAZP+12]	EDIFACT-Nachrichten[6]	(Eventlog)	Relationsanalyse, *Process Mining*
[KYYW20]	Artefaktinformationen und Geschäftsregeln	(Eventlog)	Transformationsregeln *Process Mining*
[WiKL18]	Aktivitäten und zugehörige Daten	BPMN-Modell (Eventlog)	Transformation zu Bedingungen, *Process Mining*

In einigen Publikationen werden Geschäftsprozessmodelle basierend auf anderen grafischen *Modellen* generiert. Dabei werden drei grundlegende Ansätze unterschieden: Bei metamodellbasierten Ansätzen wird die Transformation auf Basis der formalen Definitionen der Modellierungssprachen spezifiziert. Transformationsregeln definieren konkret, wie Modellelemente/Modellausschnitte umgewandelt werden sollen, während Mapping-Regeln die Zuordnung zwischen den Elementen des Quell- und Zielmodells festlegen. Die Transformation wird durch strukturierte Austauschformate ermöglicht, die eine systematische Überführung der Modelle gewährleisten. Diese Ansätze sind in nachfolgender Tabelle 4-12 zusammengefasst.

Tabelle 4-12: Ansätze basierend auf Modellen

Ansatz	Modelltyp/-Format	Output	Verfahren
[SiGa16]	Datenmodell (UML-Objektmodell ähnliches Modell)	BPMN-Modell	Mapping-Regeln
[KüRG07]	Referenz-Objekt-Lifecycle	UML-Aktivitätsdiagramm	Mapping-Regeln
[RDHI08]	Objekt-Verhaltens-Modell	Petri-Netz	Metamodell, Heuristik
[FZDL12]	Konzeptuelles Modell zu Rollen, Ziele, Funktionen (Dependency Network Diagrams)	Petri-Netz	Mapping-Regeln, Pattern
[VRAV10]	Produkt-Daten-Modell (Product Based	Petri-Netz	Mapping-Regeln

[6] "Electronic Data Interchange for Administration, Commerce and Transport" ist ein von der UN festgelegtes elektronisches Datenaustauschformat für Verwaltung, Handel und Transport [Pele21].

Ansatz	Modelltyp/-Format	Output	Verfahren
	Workflow Design)		
[SSFM08]	E-Mail-Aufgaben-Graph (Task Delegation Graph)[7]	Graphbasierte Visualisierung	Mapping-Regeln
[KWZA+22]	Entscheidungsmodell (DMN[8]-Modell)	BPMN-Modell mit DMN	Mapping-Regeln
[KlNa15]	Attribute-Relationship-Diagramm	BPMN-Modell und Entscheidungstabelle	Mapping-Regeln
[DePo14]	Zielmodell (Business-Strategy Context Process)	BPMN-Modell	Metamodell
[NaGE14]	Zielmodell (KAOS Goal Model) und Bedingungen (CTL)	BPMN-Modell	Clustering, Pattern
[GNPP+11]	Zielmodell (Wahrheitswerte), Einschränkungen	BPMN-Modell	Aktivitäts-Ziel-Ontologie, Mapping
[BeHO16]	Zielmodell (Erweitertes i*-Modell)	BPMN-Modell	Mapping-Regeln
[Mirb17]	Zielmodell (Map-Modell)	BPMN-Modell	Ontologie-Reasoning und Mapping-Regeln
[CKSB11]	Formalized Analysis Model	BPMN-Modell	Mapping-Regeln
[DhRa10]	Use Case Modell (erweitertes UML-Use Case Diagramm)	UML-Aktivitätsdiagramm	Metamodell und Regeln

Des Weiteren existieren auch Ansätze, um aus anderen Daten zunächst diese erwähnten Modelle zu generieren wie beispielsweise ein DMN-Modell aus Text [GCPV22].

In zwei Publikationen werden die Geschäftsprozessmodelle als BPMN-Modelle regelbasiert aus Wissensbasen generiert [YaCC19, ViLa22]. Diese Wissensbasen werden in Form von Ontologien realisiert, die Geschäftsregeln bzw. Anforderungen formal reprä-

[7] Ein E-Mail-Aufgaben-Graph (Task Delegation Graph) bildet die Beziehungen zwischen Aufgabenzuweisungen und -weiterleitungen in E-Mail-Kommunikation ab. Die Knoten repräsentieren dabei die Beteiligten und die Kanten die Aufgabenübertragungen zwischen ihnen.

[8] Decision Model and Notation)

sentieren. Eine Ontologie stellt dabei ein formales, maschinenlesbares Modell einer Wissensdomäne dar, das Konzepte, deren Eigenschaften und Beziehungen zueinander definiert. Die Generierung der BPMN-Modelle erfolgt durch Reasoning-Mechanismen auf diesen Ontologien. Beim Reasoning wird aus bestehendem Wissen durch logische Schlussfolgerungen neues Wissen abgeleitet. In dem auf Geschäftsregeln basierenden Ansatz von [JaHu11] (siehe Tabelle 4-10) wird eine Ontologie manuell aus informalen Geschäftsregeln erstellt und aufbauend auf dieser werden manuell formale Geschäftsregeln definiert, auf denen dann die Prozessmodellgenerierung basiert. Dadurch soll die Konsistenz und das Verständnis von Geschäftsregeln verbessert und ihre Wiederverwendbarkeit erhöht werden.

Die meisten Publikationen der LR generieren die Geschäftsprozessmodelle basierend auf Eventlogs (siehe Abschnitt 4.1.2.2). Die durch die LR identifizierten Ansätze, die Eventlogs verarbeiten, wenden entweder Process Discovery-Algorithmen oder maschinelle Lernverfahren an.

Mittels *Process Discovery-Algorithmen* werden prozessbezogene Informationen extrahiert. In diesem Bereich existieren in der Forschung und Praxis unterschiedliche Algorithmen, um abhängig von den Zielen (beispielsweise der gewünschten Modellierungssprache, in der das Modell generiert werden soll) und den Eigenschaften der zu analysierenden Eventlogs, Geschäftsprozessmodelle zu generieren [Aals22].

Resultate der LR, die zur Prozessmodellgenerierung Process Discovery-Algorithmen nutzen, sind aufgrund der Anzahl in Tabelle 9-1 im Anhang aufgelistet. Um die Ansätze weiter zu betrachten, sind diese anhand der Modellierungssprache, in der das Modell generiert wurde, klassifiziert. In den Publikationen werden zur Darstellung des resultierenden Modells Petri-Netze, BPMN-Modelle, Zustandsmaschinen[9], Kausale Netze, Multi-Variant-Prozessmodelle[10], Activity Precedence Graphen[11], Temporal Network Representation[12] oder Direkt-Folge-Graphen gewählt. Publikationen, die Process Discovery-Algorithmen wie beispielsweise den Alpha-Algorithmus vorstellen oder anwenden, die ausschließlich auf Eventlogs basieren, werden nicht im Einzelnen vorgestellt. Anwendungsempfehlungen für diese Ansätze werden in anderen Publikationen ausführlich diskutiert [ACDL+19, WBVB12]. Die verschiedenen Algorithmen werden dabei miteinander verglichen, indem die generierten Geschäftsprozessmodelle auf ihre Qualität hin bewertet werden (siehe Abschnitt 2.3.1). Folgend sollen die Resultate vorgestellt werden, die sich bezüglich der Datengrundlagen oder der generierten Modelle von

[9] Zustandsmaschinen: [LGXX+14].
[10] Multi-Variant-Prozessmodelle: [FoGP15].
[11] Activity Precedence Graphen: [LiDi13].
[12] Temporal Network Representation: [SeWG17].

anderen Arbeiten unterscheiden und insbesondere in der vorliegenden Arbeit relevant sein könnten (siehe Abschnitt 4.2.1).

- [LeMS21] stellen einen Ansatz vor, in dem die Eventlogs durch Pfad-Eingaben der Anwender erzeugt werden. Die Anwender erstellen verschiedene unvollständige Geschäftsprozesspfade und werden dabei von einem Tool unterstützt. Basierend auf diesen Eingaben wird ein Eventlog erzeugt, aus dem anschließend ein Geschäftsprozessmodell mit Process Discovery Algorithmen generiert wird [LeMS21].
- [AaBe20] stellen einen Ansatz zur Generierung von objektzentrischen Petri-Netzen vor, die auf höheren gefärbten Petri-Netzen basieren. Dieser Ansatz wird als Erweiterung des Ansatzes von [BeAa20]gesehen, die einen Ansatz zur Generierung eines Multiple ViewPoint Modell basierend auf einem Datenbank-Log vorstellen. Bei objektzentrischen Petri-Netzen entsprechen die Stellen Objekttypen und die Marken stellen Ausprägungen von diesen Objekttypen dar. Objektzentrische Petri-Netze werden aus objektzentrischen Eventlogs generiert, bei denen sich ein Event auf eine Sammlung von Objekten bezieht, die jeweils einen Objekttyp anstatt eine Case-ID haben.
- [GoRA19] stellen einen Ansatz vor, bei dem basierend auf verschiedenen Änderungslogs von Datenbanken ein Metamodell generiert wird, um daraus Eventlogs mit unterschiedlichen Perspektiven zu generieren.
- Ähnlich dazu stützen [LGHH+16] ihren Algorithmus auf Event- und Tokenlogs, die Zustandsänderungen und Ressourcennutzungen beschreiben. In Ihrer Publikation beschreiben sie daher sowohl die Generierung dieser Tokenlogs aus Workflowmanagement-Systemen als auch die Generierung von Geschäftsprozessmodellen basierend auf diesen Logs [LGHH+16].
- [AMDG17] stellen einen Ansatz zur Generierung von hybriden Petri-Netzen vor. Hybride Petri-Netze sollen die Vorteile von informellen und formellen Geschäftsprozessmodellen kombinieren. Während Prozesspfade, die eindeutig in einem Eventslog definiert sind, in dem generierten Modell präzise definiert werden, werden uneindeutige Prozesspfade, wo Informationen in dem Eventlog dazu fehlen, informaler in dem Modell beschrieben AMFG17. Damit wird hier eine Herausforderung behandelt, die insbesondere bei der Verwendung von realen Daten existiert.
- [AgHE20] beziehen in ihren Process Discovery Algorithmus sowohl Eventlogs als auch Informationen über den Datenaustausch zwischen Organisationsrollen mit ein, die aus einem anderen Modell abgeleitet sind. Die zusätzliche Berücksichtigung des Datenaustauschs ermöglicht eine effizientere Entdeckung komplexer Strukturen im Vergleich zu Algorithmen, die nur auf Eventlogs basieren [AgHE20].

- [HGAE+22] entwickeln einen Ansatz, der einen sich ständig weiterentwickelnden Graphen aus integrierten Datenobjekten und Aufgaben verwendet, um statische und dynamische Aspekte der Verwaltungs- und Wissensarbeit zu modellieren und zu speichern, und testen den Ansatz in einer realen Umgebung im Bereich des geistigen Eigentums. Darüber hinaus stellen sie einen abhängigkeitsbasierten Process Mining-Ansatz vor, um datenabhängige Aufgabensequenzen im graphbasierten Modell zu erlernen [HGAE+22].

Des Weiteren existieren einige Publikationen, die basierend auf anderen Daten Eventlogs generieren, um anschließend Process Discovery Algorithmen anwenden zu können. Beispielsweise generieren

- [LeAa15] basierend auf Code,
- [AGTW11] basierend auf Regeln,
- [BaBC18], [WiKL18], [EAZP+12] und [KYYW20] basierend auf Tabellen,
- [VRAV10] basierend auf Modellen,
- [DiMe13], [JlGB17], [RBFZ20] und [KEKR21] basierend auf E-Mails und [SoTS15], [WKKL19] und [SBCM+21] auf Text-Dokumenten

Eventlogs[13]. [DiMe13] entwickeln einen zweistufigen Ansatz: Zunächst wird eine Wissensbasis aufgebaut, aus der anschließend deklarative Einschränkungen abgeleitet werden. Diese deklarativen Einschränkungen definieren explizit die Regeln und Bedingungen, die der Geschäftsprozess erfüllen muss, anstatt den Prozessablauf prozedural zu beschreiben. Die Generierung aus E-Mails und Text-Dokumenten wird mittels Natural Language Processing (und maschinellen Lernverfahren [RBFZ20]) erreicht (siehe auch bei der Betrachtung der Generierung aus natürlichsprachlichen Texten). Außerdem wenden weitere Ansätze Verfahren der künstlichen Intelligenz (KI) an, um Geschäftsprozessmodelle aus Eventlogs zu generieren.

- [GrMM18] ergänzen Process Discovery Techniken mit Verfahren der künstlichen Intelligenz, um Geschäftsprozessmodelle zu generieren und zu analysieren. Zunächst wird mittels eines Eventlogs ein Modell generiert und anschließend in Komponenten aufgeteilt. Mittels eines Neuronalen Netzes werden neue Geschäftsprozessmodelle erstellt und anhand einer Simulation analysiert. Das Neuronale Netz basiert auf den in dem Eventlog gefundenen Abhängigkeiten, den Beziehungen zwischen den einzelnen Komponenten des Modells sowie auf dem anfänglichen Modell. Das für die bestimmte Geschäftsaufgabe effektivste Modell

[13] Diese Ansätze werden nicht den Ansätzen basierend auf Eventlogs zugeordnet, da die Kategorisierung nach Input und nicht nach der verwendeten Methode vorgenommen wurde.

nach den Simulationsergebnissen, ist das resultierende Geschäftsprozessmodell des Ansatzes [GrMM18].

- [GGPS06] stellen den auf maschinellem Lernen basierenden Algorithmus DWS-Mining vor. Der Algorithmus identifiziert basierend auf Clusterverfahren verschiedene Prozessvarianten und kann diese in dem generierten Geschäftsprozessmodell präziser definieren. [SoMF21] stellen einen auf maschinellem Lernen basierenden Algorithmus vor, der Graph Convolutional Neural Networks (GCNs) verwendet, um Petri-Netze zu generieren. GCNs sind eine spezielle Form künstlicher neuronaler Netze, die auf Graphstrukturen operieren können. Sie verarbeiten die Beziehungen zwischen Knoten in einem Graphen ähnlich wie Convolutional Neural Networks Muster in Bildern erkennen. Der Algorithmus wurde mit synthetisch erzeugten Paaren von Eventlogs und zugehörigen Geschäftsprozessmodellen trainiert [SoMF21].

Maschinelle Lernverfahren werden auch bei Ansätzen verwendet, die basierend auf *natürlichsprachlichen Texten* Geschäftsprozessmodelle generieren. In Unternehmen liegen die Informationen zu Geschäftsprozessen häufig in natürlichsprachlicher Form vor, wodurch diese Informationen unstrukturiert sind. Bei einer Verwendung der natürlichen Sprache zur Informationsbeschreibung ist kein Erlernen der formalen Syntax einer Modellierungssprache notwendig. Sie bietet daher den Vorteil, dass diese von einem größeren Personenkreis gelesen werden kann [BlAt03]. Andererseits können natürlichsprachliche Texte zu Missverständnissen führen und durch die unstrukturierte Form ist eine einheitliche Analyse nicht möglich.

Für Ansätze, die auf natürlichsprachlichen Texten basieren, existieren bereits einige LR, die ebenfalls in der vorliegenden Arbeit berücksichtigt wurden [ATSD+18, MAAB+19]. Die Resultate der LR, die als Input natürlichsprachliche Informationen verwenden, sind in Tabelle 9-2 im Anhang aufgelistet. Diese Ansätze unterscheiden sich insbesondere in den verwendeten Modellierungssprachen für das generierte Geschäftsprozessmodell sowie in dem angewandten Ansatz zur Verarbeitung der natürlichsprachlichen Sprache abhängig von der Struktur des Inputs. Bei einigen Ansätzen stehen zusätzlich zu dem natürlichsprachlichen Text Meta-Informationen zur Verfügung. Beispielsweise können bei einer E-Mail neben dem Inhalt der E-Mail auch Absender, Empfänger, Zeitpunkt und Betreff angegeben sein.

In der LR lassen sich drei verschiedene Ansätze zur Verarbeitung von natürlichsprachlichen Texten identifizieren:

- Sprachregeln: Sprachregelbasierte Verfahren werden insbesondere für Prozess-Extrahierung verwendet, in denen angenommen wird, dass die Sätze bestimmten Regeln folgen.

- Pattern-Matching: Es werden Sprachpattern entwickelt, die grundlegende Sprachmuster, syntaktische Bäume und die Auflösung von Anaphern berücksichtigen. Dieses Verfahren ist jedoch oftmals durch domänenspezifische Wissensbasen begrenzt, sodass sie nicht für domänen- oder themenoffene Szenarien geeignet sind.
- Die Natural Language Processing-Verfahren (NLP-Verfahren) werden angewandt, um die natürlichsprachlichen Texte zu verarbeiten. Es können beispielsweise Subjekt-Verb-Objekt-Konstrukte, Teilnehmer und Aufspaltungswörter (und/oder/…) extrahiert werden, die anschließend verarbeitet werden. Dabei können folgende Verfahren unterschieden werden, die in einigen Publikationen auch kombiniert werden:
    - Bei der *Tokenisierung* werden Texte in Einheiten/Blöcke aufgeteilt, die als Token bezeichnet werden. Meistens entspricht ein Wort einem Token.
    - Beim *Taggen* werden die unterschiedlichen Wörter beispielsweise mit Verb oder Nomen annotiert.
    - *Parsing* ist das Verfahren der Zuordnung von strukturellen Beschreibungen zu Klassifikationen der Wörter in natürlichsprachlichen Texten.
    - *Chunking* ist der Prozess der Kategorisierung von Ideen in einer Hierarchie.
    - *Part-of-Speech* (POS) ist der Prozess der Markierung eines Wortes zu seiner entsprechenden Wortart.

Die meisten Ansätze der LR, die natürlichsprachliche Texte verarbeiten, basieren auf NLP. Als Input werden häufig textuelle Prozessbeschreibungen verwendet, aber auch Use-Case-Beschreibungen, User-Stories, Geschäftsprozessregeln, Richtlinien-Dokumente und Software-Anforderungen. Es wurden auch Kochrezepte verwendet, da sie ähnlich wie Geschäftsprozesse eine Reihenfolge von Aktivitäten mit klar definierten Inputs und Outputs beschreiben. Die generierten Modelle werden in verschiedenen Modellierungssprachen dargestellt, darunter BPMN, Petri-Netze, UML-Aktivitätsdiagramme, Declare, EPK und Prozessstrukturbäume.

4.2 Möglichkeiten und Herausforderungen in der Prozessmodellgenerierung

In diesem Abschnitt sollen die Möglichkeiten zur Prozessmodellgenerierung der bestehenden Ansätze im Bezug zu der vorliegenden Arbeit erarbeitet werden (siehe Abschnitt 4.2.1). Außerdem sollen die Herausforderungen in der Prozessmodellgenerierung durch die gefundenen Ansätze aufgezeigt werden (siehe Abschnitt 4.2.2). Durch die Berücksichtigung der Möglichkeiten und Herausforderungen der betrachteten vergleichbaren

Ansätze zur Prozessmodellgenerierung soll sichergestellt werden, dass der in dieser Arbeit entwickelte Ansatz mindestens diesbezüglich einen Mehrwert für die Prozessmodellgenerierung birgt.

4.2.1 Möglichkeiten der Resultate in Bezug zur Arbeit

Wie in Kapitel 3 beschrieben, können zur Prozessmodellgenerierung insbesondere Dokumente zu Prozessinstanzobjekten geeignet sein, da diese sowohl Aktivitäts- als auch Objektinformationen sowie Informationen zu allen Objektmodellierungsperspektiven enthalten können. Zusätzlich zu den Dokumenten zu Prozessinstanzobjekten könnten jedoch auch noch weitere Daten als Informationsbasis für die Prozessmodellgenerierung verwendet werden, um beispielsweise den beobachtbaren Kontrollfluss zu ermitteln. Daher fasst Tabelle 4-13 zunächst die zu unterscheidenden Input-Daten, die in den Ansätzen verarbeitet wurden, nach der jeweiligen Strukturiertheit zusammen.

Tabelle 4-13: Input-Daten nach Strukturiertheit

Strukturiert	Semi-Strukturiert	Unstrukturiert
Code	Eventlogs, Location Data, Event Data (XES)	Natürlichsprachliche Texte
Strukturierte Regeln (Declare, SBVR)	Regeln (Textmuster)	Unstrukturierte Regeln (natürlichsprachlich)
Wissensbasis	Semi-strukturierter Text (Textmuster, EDIFACT Messages, Emails)	
Tabellen (CSV, XLS)	Modelle	

Weiter können diese Input-Daten nach der in Abschnitt 3.1 beschriebenen Betrachtung analysiert werden. Daher werden die Ansätze nach den Informationsträgerkategorien, der getragenen Information und dem Datenformat unterschieden. In Tabelle 4-14 werden zu den einzelnen Informationsträgerkategorien die getragenen Informationen beschrieben, die in den Ansätzen durch deren verwendete Input-Daten gegeben sind. Zu diesen wird zusätzlich mindestens ein Beispielansatz genannt.

Tabelle 4-14: Ansätze zu den verschiedenen Informationsträgerkategorien

Informationsträger-kategorie	Getragene Informa-tion	Datenformat inklusive Beispiel-ansatz
Dokument zu Ob-jekttyp	Objekt-Lebenszyklus	Modell [KüRG07]

Informationsträger-kategorie	Getragene Information	Datenformat inklusive Beispielansatz
	Produkt	Modell [VRAV10]
	Artefakt	Tabelle [KYYW20]
Prozessinstanz-betreffendes Dokument	Berechnungstabelle	Tabelle [HePv11]
Prozessinstanz-betreffende Daten	Ortsdaten	Tabelle [ZMSS18]
	IT-Systemdaten	Eventlogs, Event Data, Prozesspfade (siehe Tabelle 9-1)
Prozesstyp-betreffendes Dokument	Geschäftsprozess-beschreibung	Text [HoKW18, SoRG22], Modell [GhKC07] , Textmuster [Capo16], Tabelle [KrDe10]
	Geschäfts(prozess)-regeln	Text [FlKM05], Tabelle [FiAv14] Modell [KWZA+22] Formale Sprache (Declare, SBVR, …) [DeCh16]
	Anwendungsfälle	Text [YuBL10], Tabelle [HnCe21], Modell [DhRa10]
	Anforderungen	Text [KaMA15], Modell [DePo14, BeHO16], Ontologie [YaCC19]
	Prozessziele	Modell (Anwendungssicht) [Mirb17], Modell (Unternehmenssicht) [GNPP+11, NaGE14]
Unternehmens-zugehöriges Dokument	Klassen, Attribute und Verhalten	Modell [RDHI08, KlNa15, SiGa16]
	Unternehmens-richtlinien	Text [LWZZ10]
	Ziele, Rollen, Richtlinien, Funktionseinheiten, Abhängigkeiten	Modell [FZDL12]

Informationsträger-kategorie	Getragene Information	Datenformat inklusive Beispielansatz
	Unternehmensziele	Modell [GhKC07]
Unternehmensdaten	Entities und Relationen	SAP-Datenbank [GoRA19], CRM-Datenbank [BaBC18], Datenbank-Logs [BeAa20]
	System-Code	Source-Code [PDSP15, PéGP11]
	IT-Systemdaten	Eventlogs, Event Data (siehe Tabelle 9-1)
	Unternehmensobjekte, -konzepte, und -funktionen	Wissensbasis [ViLa22]
Kommunikationsdaten	Kunden-Service-Kommunikation	E-Mails [KEKR21]
	E-Mail-Verlauf	E-Mails [RBFZ20, DiMe13], E-Mail-Log [JlGB17], EDIFACT-Messages [EAZP+12]

Bei dieser Betrachtung wird ersichtlich, dass bisher keine Ansätze gefunden wurden, die auf Daten zu Objektinstanzen basieren. Jedoch können die Ansätze der LR-Resultate dennoch in der Methode in Kapitel 6 berücksichtigt werden. Zum einen könnten zur Verarbeitung der Daten bzw. zur Informationsextraktion aus den Daten ähnliche Ansätze adaptiert werden. Zum anderen könnten diese Ansätze relevant sein, da bei dieser Methode auch noch weitere Daten als Informationsbasis für die Prozessmodellgenerierung verwendet werden könnten. Daher werden in Tabelle 4-15 ausgewählte Ansätze der LR-Resultate bezüglich der möglichen Berücksichtigung in der Methode beschrieben.

Tabelle 4-15: Übersicht zur möglichen Berücksichtigung von LR-Resultaten

Publikation	Ansatz	Mögliche Berücksichtigung
[LeMS21]	Manuelle Pfad-Definition zur Prozessmodellgenerierung.	Da in dieser Arbeit Informationen zur Reihenfolge der Prozessinstanzobjekte zur Analyse des Kontrollflusses erforderlich sind, bietet sich dieser Ansatz zur Verarbeitung manuell erstellter Pfad-Möglichkeiten an.

Publikation	Ansatz	Mögliche Berücksichtigung
[AaBe20]	Objektzentrierte Petri-Netze basierend auf höheren gefärbten Petri-Netzen.	Der Ansatz weist Verbindungen zu den in Abschnitt 3.2.2 vorgestellten Modellierungssprachen auf und eignet sich daher zur Transformation der Informationen in objektorientierte Modellierungssprachen.
AMFG17	Differenzierte Abbildung eindeutiger und uneindeutiger Prozesspfade.	Dieser Ansatz behandelt eine zentrale Herausforderung bei der Verwendung realer Daten, die auch für diese Arbeit von Bedeutung ist.
[AgHE20]	Integration von Eventlogs und Datenaustausch-Informationen.	Da in dieser Arbeit der Kontrollfluss aus Objektflüssen abgeleitet werden soll, ist der Bezug zum Datentausch relevant.
[BaBC18]	Transformation von CRM-Daten in Geschäftsprozessmodell mittels Process Discovery Algorithmen.	Der Ansatz leitet aus CRM-Daten Event-Klassen und Beschriftungen ab und transformiert diese nach einer Vorverarbeitung in einen Eventlog. Da er für unstrukturierte Daten entwickelt wurde, aus denen Geschäftsprozessmodelle abgeleitet werden, ist er für die vorliegende Arbeit relevant.
[EAZP+12]	Geschäftsprozessmodell-Generierung aus EDIFACT-Nachrichten.	Die Arbeit fokussiert die Herausforderung, Überschneidungen zwischen verschiedenen Nachrichten zu identifizieren, um diese Prozessinstanzen zuzuordnen. Da dies auch bei Prozessinstanzobjekten relevant ist, lässt sich ein vergleichbarer Ansatz in der vorliegenden Arbeit implementieren.
[WiKL18]	Transformation von Aktivitäten und Daten (manuelle Eingabe) in Bedingungen und Eventlog.	Die Transformation von Aktivitäten und zugehörigen Daten zu Bedingungen und einem Eventlog stellt einen möglichen Ansatz für diese Arbeit dar. Insbesondere die definierten allgemeinen Bedingungen für das generierte Modell sind relevant.
[GoRA19]	Metamodellbasierte Eventlog-Generierung aus Änderungslogs von Datenbanken.	Da in dieser Arbeit Kontrollfluss-Informationen aus Änderungen der Prozessinstanzobjekte abgeleitet werden sollen, sind diese objektfokussierten Ansätze anwendbar. Obwohl sich die Ansätze auf Objekttypen beziehen, lassen sich aus Prozessin-
[KYYW20].	Integration von Artefaktinformationen (Objekttypen wie Bestellung, Produkt	

Publikation	Ansatz	Mögliche Berücksichtigung
	und Rechnung) und Geschäftsregeln in einem Artifact-centric Process Model.	stanzobjekten beispielsweise über Datenmodelle entsprechende Informationen ableiten.
[LGHH+16]	Analyse von Zustandsänderungen und Ressourcennutzung.	
[KüRG07].	Generierung basierend auf Objekt-Lebenszyklen. Ein Objekt-Lebenszyklus-Modell beschreibt die möglichen Zustände für ein Objekt sowie die Zustandsänderungen.	
[VRAV10]	Process Discovery auf Basis von Produkt-Datenmodellen. Das Produkt-Datenmodell stellt Datenelemente eines Produktes sowie die Operationen auf diesen Datenelementen dar.	
[SiGa16]	Transformation von Datenmodell zu BPMN.	Falls in dieser Arbeit aus den Prozessinstanzobjekten zunächst ein Datenmodell erstellt wird, ist der Ansatz zur Transformation des Datenmodells zum Geschäftsprozessmodell relevant.
[RDHI08]	Generierung von Petri-Netzen aus Objekt-Verhaltens-Modellen.	Da zum einen in dieser Arbeit ein Modell generiert werden soll, das auf Petri-Netzen basiert und zum anderen in dem Ansatz die Transformation über ein gemeinsames Metamodell durchgeführt wird, könnte in dieser Arbeit ein vergleichbarer Ansatz gewählt werden.

Des Weiteren sind insbesondere regelbasierte Ansätze für die vorliegende Arbeit relevant, da sich entsprechende Regeln durch den Vergleich der Prozessinstanzobjekte innerhalb einer Prozessinstanz und/oder prozessinstanzübergreifend ableiten lassen. Zusätzlich lassen sich Ansätze zur Verarbeitung natürlichsprachlicher Texte einsetzen, um Prozessinstanzobjekte, die beispielsweise als Dokumente vorliegen, vorzuverarbeiten.

4.2.2 Herausforderungen in der Prozessmodellgenerierung

Die vorgestellten Resultate der LR unterscheiden sich in den verwendeten Input-Daten, den ausgewählten Modellierungssprachen und den entwickelten Verfahren. Zusammenfassend werden bereits basierend auf folgenden Input-Daten Geschäftsprozessmodelle generiert: Software-Code, Regeln, Tabellen, Modellen, Eventlogs und natürlichsprachliche Texte. Die Generierung der Modelle wird durch Zeit-, Kostenersparnisse und einheitliche Modellqualität motiviert. Dennoch weisen die Ansätze auch einige Herausforderungen auf, die im Folgenden betrachtet werden sollen. Dabei wird insbesondere auf die Herausforderungen bezüglich der Inputs, der Verfahren und der generierten Modelle eingegangen. In einer angrenzenden LR wird eine Bewertung von NLP-Ansätzen vorgenommen, die ebenfalls auf Herausforderungen eingeht [ShSA22].

Ansätze, die Modelle basierend auf realen Input-Daten generieren, sind auf deren hinreichende Qualität angewiesen. Bei der Prozessmodellgenerierung kann nur das abgebildet werden, was in dem verwendeten Input auch dargestellt wird. So ist es beispielsweise bei den Process Mining Methoden entscheidend, die relevanten Systeme zu identifizieren und bei einer Vorverarbeitung des Inputs keine relevanten Aspekte zu entfernen. Sind die relevanten Reihenfolgen der Aktivitäten nicht in dem verwendeten Input, so bildet ein generiertes Modell diese auch nicht ab [Aals22]. Ist der Input jedoch zu umfangreich, um beispielsweise sicherzustellen, dass alle Pfade, die über einen gewissen Zeitraum anfallen, in dem Eventlog vorkommen, können die generierten Modelle zu komplex sein [vgl. SuBa16b].

Diese genannten Herausforderungen sind nicht nur bei den Ansätzen mit Eventlogs relevant. Bei den natürlichsprachlichen Input können Sätze komplex sein, wenn beispielsweise Aktivitäten auf mehrere Sätze aufgeteilt werden und Relativsätze behandelt werden müssen [vgl. MLTA19]. Auch können Redundanzen und Mehrdeutigkeiten, beispielsweise in Anforderungsdokumenten, die als Input verwendet werden, zu fehlerhaften Modellen führen [vgl. LWZZ10]. Bei natürlichsprachlichem Input können unterschiedliche Wörter für das gleiche verwendet werden, was bei einer Verarbeitung zu unterschiedlich generierten Aktivitäten in einem Modell führen kann. Ist der Detailierungsgrad einer Beschreibung unterschiedlich, so kann es vorkommen, dass auch das Modell in unterschiedlichen Detailierungsgraden Sachverhalte abbildet. Wird der Input durch einen Benutzer erstellt, entstehen potenzielle Fehler und Ungenauigkeiten sowie eine subjektive Wahrnehmung der beschreibenden Person bezüglich des Geschäftsprozesses [vgl. FZDL12]. Dies führt dazu, dass häufig nur die Prozessteile beschrieben werden, die die jeweilige Person betreffen. Zudem werden nicht alle möglichen Prozessinstanzen erfasst, sondern primär die häufigsten oder bekanntesten. Darüber hinaus sind natürlichsprachliche Texte wie Unternehmensrichtlinien oft unvollständig und definieren nicht alle Aspekte eines Geschäftsprozessmodells, was zu Lücken in den generierten

Geschäftsprozessmodellen führt [vgl. LWZZ10]. Bei den natürlichsprachlichen Ansätzen, die auf Prozessbeschreibungen aufbauen, aber auch bei den Modell-Ansätzen, wird häufig nur eine Input-Datei verwendet, obwohl ein Geschäftsprozess oftmals über mehre Dateien beschrieben wird [vgl. MaAb21]. Des Weiteren haben diese Ansätze oftmals Probleme, die sich aus dem Wechsel von Aktiv und Passiv des Eingabetextes, möglichen Umformulierungen und Änderungen der Reihenfolge und Bedingungen ergeben, die explizit vermerkt sind. Ebenso kann es aber auch zu Fehlern führen, wenn Meta-Aussagen und Beispielsätze, die nicht zu Modellelementen führen sollten, transformiert werden [vgl. MLTA19]. Außerdem nutzen die derzeit verwendeten Algorithmen hauptsächlich syntaktische Muster, erfassen aber nur wenig Semantik in den Sätzen [vgl. LWZZ10].

Des Weiteren werden bei einigen Ansätzen Informationen (Textinhalte oder Modellelementen) beim Input ausgeschlossen. Bei natürlichsprachlichen Prozessbeschreibungen werden die Texte beispielsweise auf sequentielle Beschreibungen eingeschränkt [vgl. FrMP11, STAM+19], oder andere Ansätze geben vor, dass es nur einzelne, einfache Sätze, keine Fragen oder nur relevante Inhalte sein dürfen, um komplexe Strukturen zu vermeiden [vgl. HoKW18, FrMP11]. In anderen Publikationen dürfen beispielsweise Objekte nur eine Beziehung zu einem anderen Objekt haben, da bei mehr vorhandenen Beziehungen diese falsch transformiert werden [vgl. MaAb21]. Werden jedoch alle Informationen verwendet, oder sollen weitere Informationen beispielsweise durch Annotationen ergänzt werden, um einen Algorithmus zu verbessern, kann dies zeitaufwändig sein. Außerdem zeigt sich in der Praxis, dass die Input-Daten nicht immer vorgegebene Regeln, beispielsweise bezüglich des Layouts, einhalten [HHMD+20].

Ebenso wie Einschränkungen beim Input getroffen werden, schränken sich einige Publikationen auf ausgewählte Elemente einer Modellierungssprache, in der das Modell generiert werden soll, ein. Dies kann sinnvoll sein, jedoch fehlt oftmals die Begründung [vgl. MaAb21]. Zusätzlich zu den ausgeschlossenen Elementen werden auch komplexe Kontrollflussmuster im generierten Geschäftsprozessmodell ausgeschlossen [vgl. HoKW18]. Bauen Publikationen auf Prozessmodellierungsmustern auf, beschränken sich diese oftmals nur auf gängige Prozessmodellierungsmuster (z. B. nur die in Abschnitt 2.2.1 beschriebenen). Diese Beschränkung ermöglicht jedoch, sich auf die Umsetzung dieser Muster zu fokussieren und viele zusätzliche Einschränkungen zu vermeiden [vgl. FZDL12]. Werden keine solchen Muster verwendet, können die konstruierten Geschäftsprozessmodelle syntaktische Fehler enthalten [vgl. LWZZ10].

Basierend auf diesen Herausforderungen wurden bereits einige Verbesserungsmöglichkeiten vorgeschlagen. Daher werden folgend für die drei hauptsächlichen Herausforderungen (Qualität der Inputdaten, Algorithmus-Komplexität und Qualität des generierten Modells) ausgewählte Verbesserungsmöglichkeiten beschrieben.

- Verbesserung der Input-Datenqualität:
    - durch die Vorverarbeitung von Eventlogs: [BeDG22] stellen einen Ansatz vor, aus natürlichsprachlichen Texten mit dem Sprachmodell GPT Informationen (Aktivität, Rollen, Beziehungen) vorab zu extrahieren und dadurch in strukturierte Prozessbeschreibungen umzuwandeln. [WLXZ+] nutzen GPT, um die Datensätze zu labeln und ermöglichen so eine Zeit- und Kostenersparnis. Ebenso zeigen [FiFK23], dass Sprachmodelle wie GPT zur Prozessmodellgenerierung angewandt werden können, indem natürlichsprachliche Texte in formale Austauschformate transformiert werden.
    - durch die Ergänzung weiterer Informationen: [KovW18] wandeln unterschiedliche Daten wie Aktivitäten, Pfade, Eventlogs und Modelle in Vektoren um. Mittels eines Neuronalen Netzes werden die Aktivitäten anschließend aus dem Eventlog gelernt, in dem der Kontext betrachtet wird.
    - durch die Komposition von Aktivitäten: [TGPV21] schlagen vor, Aktivitäten in Eventlogs durch Teilmodelle zusammenzufassen.
- Verbesserung der Algorithmen:
    - durch angeben von Kontextinformationen: [GhSP10] leiten beispielsweise aus Verhaltensähnlichkeiten von Prozessbeschreibungen und kontextuellen Merkmalen Kontextinformationen ab, um eine verbesserte Relevanzbewertung der Pfade zu ermöglichen.
    - durch besseres Einbeziehen von semantischen Merkmalen: [LWZZ10] schlagen vor, Forschungsergebnisse aus anderen Bereichen wie Semantic Web, Ontologien und Prozessreferenzmodellen weiter zu nutzen.
- Verbesserung der Qualität des generierten Modells:
    - durch Reparatur-Algorithmen basierend auf dem Hinzufügen zusätzlicher Daten: Beispielsweise schlagen [MiLA17] vor, ein Modell basierend auf dem zugehörigen Eventlog zu verbessern, indem das Modell zunächst in Fragmente mit klar definierten Grenzen unterteilt wird. Die Fragmente, die nicht zum Eventlog passen, werden durch neue Fragmente ersetzt, die mit Hilfe von Process Discovery-Algorithmen erstellt werden. Anschließend wird das reparierte Modell aus den Fragmenten zusammengesetzt. Um die Lesbarkeit zu verbessern, repariert diese Methode das Modell anschließend lokal, [MiLA17]. [TiMi19] verbessern diesen Algorithmus, indem die Auswahl der Fragmente mit künstlicher Intelligenz unterstützt wird. Dieser Ansatz ist dadurch jedoch rechenintensiv. [TaHo21] beziehen zur Modellverbesserung zusätzlich Informationen zu Unternehmenszielen mit ein.

- durch Reparatur-Algorithmen basierend auf Modelleigenschaften: Einige Publikationen stellen Algorithmen vor, um Petri-Netze, die die Soundness-Eigenschaft nicht erfüllen, automatisch zu korrigieren. Dabei generiert ein Algorithmus auf Basis von einem Geschäftsprozessmodell, das nicht sound ist, bei jedem Durchlauf eine kleine Menge von alternativen Modellen, die dem Ursprungsmodell ähneln, aber weniger oder gar keine Fehler im Ablaufverhalten enthalten sollen [GLMH11]. Weitere Ansätze beschäftigen sich beispielsweise explizit mit der Einbeziehung von Zyklen [ChKB18] oder der Benennung von Modellelementen [BeDG22].
- durch Reparatur-Algorithmen, die die Komplexität reduzieren sollen: [MüBe17] stellen einen Ansatz vor, der die Komplexität in einem Modell reduzieren soll. Dazu werden die Anzahl der Aktivitäten und Datenelemente sowie die Komplexität des Kontroll- und Datenflusses beachtet und angepasst. [WSLZ19] entwickeln einen Algorithmus, um durch Abstraktion das Modell zu vereinfachen.
- durch Verwenden von Mustern: [SWMW12] generieren aus Modell-Repositories Muster zur Unterstützung der Modellierung, die die Modellqualität verbessern sollen [SWMW12].

4.3 Zusammenfassung

In diesem Kapitel wurde eine LR geplant und durchgeführt und deren Resultate wurden betrachtet. Dabei unterscheiden sich die Resultate in den verwendeten Input-Daten, der verwendeten Methode zur Generierung und den unterstützten Modellierungssprachen. In der LR wurden 125 Veröffentlichungen aus 3617 Arbeiten auf der Grundlage eines Such-Protokolls ausgewählt. Auf der Grundlage der Input-Daten konnten sieben Gruppen identifiziert werden: Software-Code, Tabellen, Regeln, Modelle, Wissensdatenbanken, Eventlogs und natürlichsprachlicher Text. Für diese Gruppen wurden die unterschiedlichen Methoden zur Prozessmodellgenerierung beschrieben. Zusätzlich zeigt die LR, dass auch wenn die Forschung zur Prozessmodellgenerierung bereits mindestens seit 1998 aktiv betrieben wird, es weiterhin viel Potential gibt, um Verbesserungen vorzunehmen. Zum einen werden immer wieder neue Technologien entwickelt, die eine solche Generierung unterstützen können, wie beispielsweise [FiFK23] zeigen. Zum anderen werden durch die Digitalisierung auch immer mehr Daten in Geschäftsprozessen und Unternehmen erzeugt, die zur Generierung verwendet werden können. Dabei zeigt sich in den Resultaten auch, dass bisher keine Ansätze gefunden wurden, die auf Objektinstanzen basieren, obwohl diese wie in Abschnitt 3.1 beschrieben relevante Informationen zu den Geschäftsprozessen enthalten. Beispielweise könnten durch diese Informationen nicht nur

der beobachtete Kontrollfluss dargestellt werden, sondern auch analysiert werden, auf welchen Informationen basierend welcher Pfad gewählt wird. In diesem Kapitel werden zusätzlich die Möglichkeiten zur Prozessmodellgenerierung der bestehenden Ansätze im Bezug zu dieser Arbeit erarbeitet und die Herausforderungen in der Prozessmodellgenerierung durch die gefundenen Ansätze aufgezeigt. Durch die Berücksichtigung der Möglichkeiten und Herausforderungen der betrachteten vergleichbaren Ansätze zur Prozessmodellgenerierung soll sichergestellt werden, dass der in dieser Arbeit entwickelte Ansatz diesbezüglich einen Mehrwert für die Prozessmodellgenerierung mit sich bringt.

5 Anforderungsanalyse zur Prozessmodellgenerierung

Diese Arbeit folgt den Richtlinien des Methoden-Engineerings, das auf Konzepten des *traditionellen* Engineering-Prozesses basiert [Brin96]. Ausgehend von einer Problemstellung wurden in Kapitel 4 bestehende Ansätze in einer Literaturrecherche betrachtet und analysiert. Unter Einbezug dieser Ansätze soll (bei Bedarf) eine neue Lösung konzeptioniert und in einem Software-Prototyp umgesetzt werden. Nach den Richtlinien des Methoden-Engineerings sollen daher in diesem Kapitel basierend auf den vorangegangenen Kapiteln sowie der Zielsetzung der Arbeit zunächst Anforderungen entwickelt werden, mit denen die Methode zur Prozessmodellgenerierung entworfen wird. Damit dient dieses Kapitel der Spezifizierung der Anforderungen an die Prozessmodellgenerierung, um die Qualität der Methode sicherzustellen und diese Qualität ebenfalls mit der Umsetzung in einem Software-Prototyp überprüfen zu können. Zur systematischen Spezifizierung der Anforderungen wird der Anwendungsfall der Prozessmodellgenerierung betrachtet.

Die Vorbedingung für die Prozessmodellgenerierung ist, dass dem Anwender die entsprechenden Daten zu diesem Geschäftsprozess vorliegen. Des Weiteren wird die deutsche Sprache als Ausgangssprache genutzt. Anpassungen für andere Sprachen sollten mit der zu entwickelten Methode mit einem gewissen Aufwand möglich sein, stehen zunächst jedoch nicht im Mittelpunkt der Betrachtung. Falls diese Vorbedingung erfüllt ist, folgt die Vorverarbeitung der Daten und anschließend die Prozessmodellgenerierung. Der Anwendungsfall ist in Abbildung 5-1 dargestellt und wird anschließend über die Ziffern 1 - 7 weiter beschrieben.

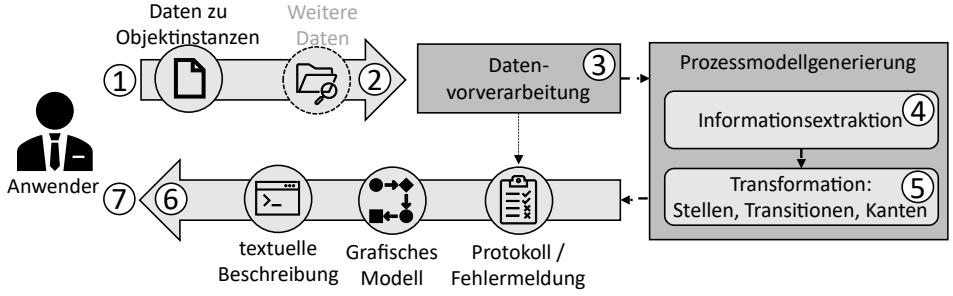


Abbildung 5-1: Anwendungsfall des Prozessmodellgenerators

1. Der Anwender ruft den Prozessmodellgenerator auf.
2. Der Prozessmodellgenerator ermöglicht das Hochladen mehrerer Dateien in unterschiedlichen vorgegebenen Formattypen.
    - Ist eine Datei in einem zulässigen Format, so wird die Datei für die Generierung verwendet.
    - Ist eine Datei nicht in einem zulässigen Format, wird der Ausschluss der Datei zur Prozessmodellgenerierung im Protokoll vermerkt.
    - Liegt keine Datei in einem zulässigen Format vor oder führt das Hochladen zu einem Abbruch des Prozessmodellgenerators, wird eine entsprechende Fehlermeldung angezeigt.
3. Die Dateien in einem zulässigen Format werden vorverarbeitet, um die Dateien anschließend für die Prozessmodellgenerierung verwenden zu können.
    - Kann eine Datei vollständig vorverarbeitet werden, ist die Datei für die Informationsextraktion bereit.
    - Kann eine Datei nicht vollständig vorverarbeitet werden, wird die Datei teilweise vorverarbeitet und der teilweise Ausschluss der Datei zur Prozessmodellgenerierung wird im Protokoll vermerkt.
    - Kann eine Datei nicht vorverarbeitet werden, wird der Ausschluss der Datei zur Prozessmodellgenerierung im Protokoll vermerkt.
    - Kann keine Datei vorverarbeitet werden oder führt eine Vorverarbeitung zu einem Abbruch des Prozessmodellgenerators, wird eine (entsprechende) Fehlermeldung angezeigt.
4. Basierend auf den vorverarbeiteten Dateien werden die relevanten Informationen zur Prozessmodellgenerierung extrahiert.
    - Können die definierten Informationen aus einer vorverarbeiteten Datei vollständig extrahiert werden, stehen diese Informationen zur Prozessmodellgenerierung bereit.
    - Können die definierten Informationen nur teilweise aus einer vorverarbeiteten Datei extrahiert werden, werden die Informationen teilweise aus dieser Datei extrahiert und der teilweise Ausschluss dieser Datei zur Prozessmodellgenerierung wird im Protokoll vermerkt.
    - Können keine definierten Informationen aus einer Datei extrahiert werden, wird der Ausschluss der Datei zur Prozessmodellgenerierung im Protokoll vermerkt.
    - Können aus keiner vorverarbeiteten Datei definierte Informationen extrahiert werden oder führt eine Extraktion zu einem Abbruch des Prozessmodellgenerators, wird eine Fehlermeldung angezeigt.
5. Die extrahierten Informationen werden in einem Geschäftsprozessmodell durch die Elemente der unterstützten Modellierungssprache (hier höhere Petri-Netze gemäß Abschnitt 3.2: Stellen, Transitionen, Kanten sowie deren Inschriften) abgebildet.

   - Können die extrahierten Informationen vollständig in einem Modell abgebildet werden, erfolgt deren Transformation.
   - Können die extrahierten Informationen nur teilweise in einem Modell abgebildet werden, werden diese Informationen teilweise abgebildet und die nicht abbildbaren extrahierten Informationen im Protokoll vermerkt.
   - Können keine extrahierten Informationen in einem Modell abgebildet werden oder führt eine Abbildung zu einem Abbruch des Prozessmodellgenerators, wird eine Fehlermeldung angezeigt.
6. Es gibt drei mögliche Ergebnisse durch den Prozessmodellgenerator:
   - Ist die Generierung erfolgreich, ist das Ergebnis das *grafische Geschäftsprozessmodell* sowie eine *maschinenlesbare Beschreibung* des Geschäftsprozessmodells.
   - Ist die Generierung erfolgreich, aber es sind Probleme aufgetreten, ist das Ergebnis das *grafische Geschäftsprozessmodell*, eine *maschinenlesbare Beschreibung* des Geschäftsprozessmodells sowie ein nichtleeres *Protokoll mit Einträgen* aus den Schritten 2 - 5.
   - Ist die Generierung nicht erfolgreich ist das Ergebnis eine entsprechende *Fehlermeldung* sowie ggf. ein bis zur Fehlermeldung erstelltes *Protokoll*.
7. Nach der erfolgreichen Prozessmodellgenerierung werden dem Anwender folgende Ergebnisse (falls vorhanden) angezeigt sowie zum Export bereitgestellt:
   - das grafische Geschäftsprozessmodell
   - die *maschinenlesbare Beschreibung* des Geschäftsprozessmodells
   - das Protokoll

Anhand dieses Anwendungsfalls sollen nun die Anforderungen an die Prozessmodellgenerierung formuliert werden. Dazu werden die einzelnen Schritte durch die Anforderungen an den Input (die Input-Dateien, Abschnitt 5.1), den Output (das generierte Modell, Abschnitt 5.2) und die Prozessmodellgenerierung sowie den Anwender (Abschnitt 5.3) folgend weiter betrachtet.

5.1 Anforderungen an die Input-Daten

Wie in Abschnitt 3.1 beschrieben, soll die Prozessmodellgenerierung insbesondere auf Daten zu den Objektinstanzen eines Geschäftsprozesses basieren. Zusätzlich könnten jedoch auch noch weitere Daten als Informationsbasis für die Prozessmodellgenerierung verwendet werden, um beispielsweise den beobachtbaren Kontrollfluss zu ermitteln. An die Daten, die zur Prozessmodellgenerierung verwendet werden sollen, sind Anforderungen zu stellen, um deren Verarbeitung bei der Prozessmodellgenerierung zu garantieren.

Bei der Betrachtung der Zweckeignung der Daten und Informationen zur Geschäftsprozessmodellierung (siehe Abschnitt 3.1), werden das Datenformat, die Datenqualität und der Informationsgehalt bewertet. Daher werden bei der Spezifizierung der Anforderungen an die Input-Daten diese Aspekte betrachtet. Insgesamt ergeben sich damit die in Tabelle 5-1 definierten Anforderungen an die Input-Daten (AI).

Tabelle 5-1: Anforderungen an die Input-Daten

Nr.	Anforderung
AI01	*Informationsgehalt*: Vorhandensein der Daten und der notwendigen Informationen AI01a. Die zur Prozessmodellgenerierung vom Anwender auszuwählenden Daten müssen dem Anwender vorliegen. AI01b. Die zur Prozessmodellgenerierung notwendigen Informationen müssen in den vom Anwender bereitgestellten Daten enthalten sein. Dies betrifft die Informationen zur Generierung des Kontrollflusses und der Objekte: • Kontrollfluss: Informationen zu den Aktivitäten sowie die Reihenfolge der Aktivitäten, um Rückschlüsse auf Zusammenhänge beobachten und ziehen zu können • Objektinformationen: Informationen zu der Objektstruktur, dem Objektzustand, den Objektbeziehungen und dem Objektfluss, um diese zu modellieren und den beobachteten Kontrollfluss zu unterstützen.
AI02	*Datenqualität*: Ausreichende Qualität der Daten nach den DQ-Dimensionen Die Daten müssen eine ausreichende Qualität haben, damit diese zur Generierung verwendet werden können. Diese Datenqualität lässt sich durch die DQ-Dimensionen in Abschnitt 3.1 betrachten. Dazu können Einschränkungen an die Daten formuliert werden, die beispielsweise • das Datenformat (um die Einheitlichkeit zu verbessern), • den Umfang der hochladbaren Daten (um die Skalierbarkeit des Software-Prototyps zu erhalten) oder • den Inhalt der Daten (um Anforderungen an die Genauigkeit, den Detailierungsgrad oder die Vollständigkeit zu erreichen) betreffen. Diese Einschränkungen sollen jedoch nicht die Erweiterbarkeit des Software-Prototyps beeinträchtigen.
AI03	*Datenformat*: Daten mit vorgegebener Struktur oder umwandelbar in diese Struktur

Nr.	Anforderung
	Wie in Abschnitt 3.1 beschrieben, werden für eine automatische Verarbeitung als Informationsträger oftmals strukturierte Daten bevorzugt, da diese nach vorgegebenen Regeln verarbeitet werden können. Zum einen könnten daher das Dateiformat und die Struktur der Daten vorgegeben werden. Mit den Einschränkungen der Dateiformate und der Struktur der Daten wird jedoch auch die Verwendbarkeit des Prozessmodellgenerators eingeschränkt. Weiterhin können keine einheitlichen Dateiformate und Strukturen angenommen werden, da die Generierung auf den realen Daten, die die Anwender in einem Unternehmen zur Verfügung haben, basieren soll. Zum anderen könnte eine Datenvorverarbeitung die Daten strukturieren und in das vorgegebene Format konvertieren indem beispielsweise Benennungskonventionen und Attributnamen durch semantische Analysen betrachtet und verändert werden. So könnten sowohl die vorgegebenen Dateiformate zur Generierung eingesetzt werden als auch weitere Formate, die zur Vorverarbeitung akzeptiert werden. Dadurch kann eine Erweiterung für ähnliche Dateiformate mit geringem Aufwand möglich sein.

Um diese Anforderungen zu erfüllen, werden sie in der Methode berücksichtigt. Nach der Berücksichtigung dieser Anforderungen ist auch darauf einzugehen, welche Daten *besonders gut* geeignet sind, um basierend auf ihnen Modelle in einer vorgegebenen Qualität zu generieren.

5.2 Anforderungen an das generierte Modell

Das zu generierende Modell wird den Geschäftsprozessmodellen zugeordnet. Ein Geschäftsprozessmodell ist ein unter Verwendung einer (in dieser Arbeit: grafischen) Modellierungssprache abgebildetes Modell, das alle bezüglich eines definierten Ziels relevanten Aspekte eines Geschäftsprozesses beschreibt (siehe Definition 2-2). Dementsprechend sind die Anforderungen basierend auf den Anforderungen, die in Abschnitt 2.3 zu den Qualitätsaspekten betrachtet wurden, zu definieren. Die hier zu definierenden Anforderungen sind daher – angelehnt an den Qualitätsaspekten für Geschäftsprozessmodelle – nach den Anforderungen an das Modell, die Modellierungssprache sowie die Modellierung betrachtet worden.

In Tabelle 5-2 werden die Anforderungen an das generierte Modell (AM) dargestellt.

Tabelle 5-2: Anforderungen das generierte Modell

Nr.	Anforderung
AM01	*Modell*: Die Modellqualität soll den in Abschnitt 2.3.1 definierten Anforderungen genügen. AM01a. *Konsistenz*: Das Modell bildet keine widersprüchlichen Informationen ab. AM01b. *Vollständigkeit*: Das Modell bildet die Aspekte eines Geschäftsprozesses ab, die aus den Daten abgeleitet werden können. Die abzubildenden Aspekte sind insbesondere der Kontrollfluss und die Informationen zu den Objekten. AM01c. *Redundanzfreiheit*: Das Model bildet einen Aspekt eines Geschäftsprozesses genau einmal ab. AM01d. *Korrektheit*: Das Model bildet den Geschäftsprozess entsprechend den Daten semantisch richtig ("valide") ab. Die abgebildeten Aspekte im Modell sind rückverfolgbar zu den Daten. AM01e. *Eindeutigkeit*: Das Modell lässt keine mehrdeutige Interpretation zu. AM01f. *Genauigkeitsebene*: Die Abstraktionsebene wurde für den Zweck entsprechend gewählt. AM01g. *Eigenschaften Petri-Netz*: Das generierte Modell soll die Eigenschaften eines *Workflow-Netzes* (siehe Definition 3-2) erfüllen.
AM02	*Modellierungssprache*: Die Modellierungssprache soll den Anforderungen einer Modellierungssprache zur Geschäftsprozessmodellierung genügen (siehe Abschnitt 2.3.2) AM02a. *Visualisierbarkeit*: Das zu generierende Geschäftsprozessmodell soll mithilfe einer Modellierungssprache grafisch dargestellt werden. AM02b. *Beschreibung*: Es soll eine maschinenlesbare Beschreibung der grafischen Darstellung möglich sein (AM02b:). Dies soll die Wiederverwendbarkeit, die Austauschbarkeit zu anderen Editoren und die Verarbeitbarkeit erhöhen. Daher soll auf dem Beschreibungsformat für Petri-Netze PNML (Petri Net Markup Language)[1] aufgebaut werden, ohne die Rückkompatibilität zu verletzen. AM02c. *Ausdrucksmächtigkeit*: Durch die gewählte Modellierungssprache soll insbesondere die Abbildung des Kontrollflusses und der Informationen zu den Objekten möglich sein (siehe Abschnitt 3.2). AM02d. *Angemessenheit*: Die Aspekte des Kontrollflusses und der Objektinformationen sollen in einer angemessenen Weise abgebildet werden. Das heißt, dass beispielsweise ein einfacher Sachverhalt nicht durch ein komplexes Modell dargestellt werden soll. AM02e. *Verständlichkeit*: Die Anforderung an die Verständlichkeit wird auch durch AM02d und die Anforderungen AM03 beeinflusst. Zusätz-

[1] http://www.pnml.org/

Nr.	Anforderung
	lich kann sie durch unterschiedliche Sichten oder Abstraktion verbessert werden. Dazu ist insbesondere auch die Erwartung über die die Erfahrungen und über das Maß der Heterogenität der Anwender der Modelle zu berücksichtigen.
AM03	*Modellierung*: Das generierte Modell soll die Richtlinien aus Abschnitt 2.3.3 berücksichtigen. Die Anforderungen AM03 betreffen die Richtlinien, um die Verständlichkeit und Übersichtlichkeit des Modells zu verbessern: AM03a. *Elementanzahl:* Es sollen so wenig Elemente verwendet werden wie möglich und die Modellgröße soll eine festgelegte Elementanzahl nicht überschreiten. AM03b. *Pfadmöglichkeiten*: Ein Element sollte möglichst wenig eingehende und ausgehende Kanten haben. AM03c. *Start- und ein Endereignis*: Es sollte nur ein Start- und ein Ereignis verwendet werden. AM03d. *Strukturiertheit*: Jeder Aufspaltungs-Konnektor soll mit einem entsprechenden Zusammenführungs-Konnektor desselben Typs zusammengeführt werden und falls möglich, soll das Modell keine Kantenüberschneidungen enthalten. AM03e. *Beschriftung und Inschriften*: Die Elemente sollen die Beschriftungs- und Inschriftenkonventionen (Tabelle 5-3) einhalten.

Da in Abschnitt 3.2 als Modellierungssprache höhere Petri-Netze wie XML- oder JSON-Netze festgelegt wurden, sind die Beschriftung und Inschriften der Elemente nach deren Modellierungskonventionen definiert [KeNS92, BePV12]. Diese Beschriftungs- und Inschriftenkonventionen: (BIK) wurden in Tabelle 5-3 zusammengefasst. Durch diese Konventionen werden insbesondere auch die Anforderungen an die Verständlichkeit und Einheitlichkeit unterstützt.

Tabelle 5-3: Beschriftungs- und Inschriftenkonventionen für das generierte Modell

Nr.	Element	Konvention	Beispiel
BIK01	Beschriftung	Beschriftungen sind in der vorgegebenen Sprache	Rechnung erstellen, Rechnung erstellt
BIK02	Transition	Objekt + Infinitiv	Rechnung erstellen, Lieferschein drucken
BIK03	Stellen	Subjekt [+Partizip Perf./Adjektiv]	Rechnung erstellt, Lieferschein gedruckt
BIK04	Zusätzliche Beschreibung	Adjektive vor Substantiven sind erlaubt, diese werden auch am Anfang klein	neue Rechnung erstellt, ausgefüllter Liefer-

Nr.	Element	Konvention	Beispiel
		geschrieben	schein gedruckt
BIK05	Aufzählung	Worte können aufgezählt werden	Rechnung erstellt und versendet, Lieferschein ausgefüllt und gedruckt
BIK06	Transitions-inschriften	XML-Netz: Prädikatenlogischer Ausdruck	Siehe [LeOb03]
		JSON-Netz: beispielsweise Jssonet	Siehe Abbildung 3-4
BIK07	Stellen-inschriften	XML-Netz: beispielsweise DTD	Siehe [LeOb03]
		JSON-Netz: beispielsweise JSON-Schema	Siehe Listing 3-1
BIK08	Kanten-inschriften	XML-Netz: beispielsweise XManiLa	Siehe [LeOb03]
		JSON-Netz: beispielsweise Jsonnet, JSON-Path	Siehe Abbildung 3-4
BIK09	Marken	XML-Netz: XML-Dokumente	Siehe [LeOb03]
		JSON-Netz: JSON-Dokumente	Siehe Abbildung 3-4

Mit diesen an das generierte Modell definierten Anforderungen sollen die von [AgGL98] für generierte Modelle definierten Eigenschaften erfüllt werden:

- Das Modell bildet *das in den Daten beobachtete Verhalten* ab (dazu zählen die in den Daten vorhandenen Aktivitäten, Abhängigkeiten der Aktivitäten und Objektinformationen sowie auch die in den Daten möglichen Ausführungsreihenfolgen der Aktivitäten).
- Das Modell bildet *nur* das in den Daten beobachtete Verhalten ab und führt keine zusätzlichen Abhängigkeiten zwischen Aktivitäten ein.

5.3 Anforderungen an die Generierung

Neben den Anforderungen an die Input-Daten und das generierte Modell sind weitere Anforderungen an die Generierung zu stellen. Die Generierung ist der (teil-)automatisierte Modellierungsprozess, um aus den bereitgestellten Input-Daten das

generierte Modell zu erhalten. Diese Generierung soll den zeitaufwändigen und fehleranfälligen Prozess der manuellen Modellierung ersetzen. Für diese Generierung ist eine Methode zu entwickeln, die anschließend in einem Software-Prototyp, dem *Prozessmodellgenerator*, umgesetzt wird und somit dem Anwender zur Verfügung gestellt werden kann.

In Tabelle 5-4 werden die funktionalen Anforderungen an die Generierung (AG) im Bezug zu den einzelnen Schritten des Anwendungsfalls (siehe Abbildung 5-1) betrachtet.

Tabelle 5-4: Anforderungen an die Generierung

Nr.	Anforderung
AG01	Funktionale Eignung Schritt 1: Seitenaufruf
	Der Prozessmodellgenerator sollte plattformunabhängig aufrufbar und nutzbar sein und zu Beginn eine Seite, um Dateien hochladen zu können, anzeigen.
AG02	Funktionale Eignung Schritt 2: Dateien hochladen und prüfen
	Der Prozessmodellgenerator sollte das Hochladen von mehreren Dateien in unterschiedlichen vorgegebenen Formattypen ermöglichen (AG02a) und die verschiedenen hochgeladenen Dateien auf ihre Formate überprüfen können, um nur die festgelegten Dateiformate vorzuverarbeiten (AG02b). Dabei soll eine Nicht-Einhaltung dieser Anforderungen den Anwender in der Nutzung des Prozessmodellgenerators möglichst nicht einschränken, weswegen der Prozessmodellgenerator robust gegenüber nicht formatgetreuen Input-Daten sein soll.
AG03	Funktionale Eignung Schritt 3: Dateivorverarbeitung
	Der Prozessmodellgenerator soll in der Lage sein, die Dateien vorzuverarbeiten (AG03a) und zu überprüfen, ob die Dateien vollständig vorverarbeitet wurden (AG03b).
AG04	Funktionale Eignung Schritt 4: Informationsextraktion
	Der Prozessmodellgenerator soll in der Lage sein, die Informationen der vorverarbeiteten Dateien zu extrahieren (AG04a) und zu überprüfen, ob die definierten Informationen extrahiert werden konnten (AG04b).
AG05	Funktionale Eignung Schritt 5: Generierungsalgorithmus

Nr.	Anforderung
	Der Prozessmodellgenerator soll in der Lage sein, ein grafisches Geschäftsprozessmodell in der definierten Modellierungssprache zu generieren, das alle relevanten Elemente, Beschriftungen und Inschriften enthält (AG05a) und dem Anwender eine maschinenlesbare Beschreibung des Modells in PNML zur Verfügung stellt (AG05b). Dabei sind die Anforderungen an das generierte Modell AM01 bis AM03 einzuhalten. Hierzu ist auch zu definieren, wie mit widersprüchlichen und/oder unvollständigen Informationen umgegangen wird (AG05c). Dabei sind auch die Grenzen eines Geschäftsprozesses zu definieren, da die Daten sich auf unterschiedliche Geschäftsprozesse beziehen können (AG05d). Es müssen für die folgenden Elemente Transformationsregeln definiert werden: AG05e. Stellen (+ Marken) durch Objektspeicher inkl. der Schemata AG05f. Transitionen inkl. Beziehungen zu den Stellen, um den Kontrollfluss abzubilden AG05g. Inschriften für die Aktivierungsbedingung und Objektmanipulation (Kantenfilter und Transitionsinschrift) Die Mindestanforderung für diese Arbeit ist dabei die Darstellbarkeit der in Abschnitt 2.2.1 vorgestellten Basis-Kontrollflussmuster (AG05h). Zusätzlich ist zu definieren, welche Prozessregeln bezüglich der Objekte überprüft werden und durch Kanten- und Transitionsinschriften in dem generierten Modell abgebildet werden können (AG05i). Dazu soll der Prozessmodellgenerator auch überprüfen, ob die extrahierten Informationen *vollständig* und *korrekt* in einem Modell abgebildet werden konnten (AG05j). Des Weiteren soll bei dieser Generierung die Nachvollziehbarkeit gewährleistet werden (AG05k).
	Funktionale Eignung Schritt 6: Generierungs-Ergebnis
AG06	AG06a. ei erfolgreicher Generierung ist das Ergebnis ein *grafisches Geschäftsprozessmodell*, eine *maschinenlesbare Beschreibung* des Geschäftsprozessmodells und ein *Protokoll*. AG06b. ei nicht erfolgreicher Generierung ist das Ergebnis eine entsprechende *Fehlermeldung* sowie ein bis zur Fehlermeldung erstelltes *Protokoll*.
	Funktionale Eignung Schritt 7: Ergebnis-Anzeige
AG07	Der Prozessmodellgenerator soll nach der Generierung das Modell grafisch darstellen können (AG07a) Der Prozessmodellgenerator soll zusätzlich eine Möglichkeit bieten, das grafische Geschäftsprozessmodell (AG07b) und die maschinenlesbare Beschreibung des Modells (AG07c) zu exportieren. Dadurch wird dem Anwender Möglichkeit gegeben, das Modell in andere Anwendungen zu integrieren oder zu teilen. Zusätzlich sollte der Prozessmo-

Nr.	Anforderung
	dellgenerator eine Möglichkeit bieten, das Protokoll (siehe AG08) zu exportieren (AG07d), um dem Anwender eine detaillierte Übersicht über die Schritte der Prozessmodellgenerierung zu geben und gegebenenfalls diagnostizieren zu können, wo Probleme aufgetreten sind.
AG08	Funktionale Eignung: Protokoll Der Prozessmodellgenerator soll folgendes im Protokoll vermerken, um Probleme bei der Prozessmodellgenerierung zu dokumentieren: AG08a. den Ausschluss von Dateien wegen des Formates vor der Vorverarbeitung AG08b. den (teilweisen) Ausschluss der Datei trotz gewünschtem Format bei der Vorverarbeitung AG08c. den (teilweisen) Ausschluss der Datei trotz erfolgreicher Vorverarbeitung bei der Informationsextraktion AG08d. die nicht oder nur teilweise abbildbaren, aber extrahierten Informationen bezüglich des Prozesstyps
AG09	Funktionale Eignung: Fehlermeldung Der Prozessmodellgenerator soll Fehlermeldungen anzeigen, wenn: AG09a. keine Dateien zur Vorverarbeitung, Extraktion und/oder Transformation vorhanden sind AG09b. ein Abbruch beim Hochladen, Vorverarbeiten, Extrahierung oder Transformation auftritt

5.4 Zusammenfassung

In diesem Kapitel wurden basierend auf den vorangegangenen Kapiteln sowie der Zielsetzung der Arbeit Anforderungen entwickelt, mit denen die Methode zur Prozessmodellgenerierung ausgearbeitet werden soll. Damit dient dieses Kapitel der Spezifizierung der Anforderungen an die Prozessmodellgenerierung und den Software-Prototyp, der die Methode zur Prozessmodellgenerierung umsetzt. Zur systematischen Spezifizierung der Anforderungen wurde der Anwendungsfall der Prozessmodellgenerierung betrachtet. Dieser Anwendungsfall besteht insbesondere aus der Datenvorverarbeitung, der Informationsextraktion und der Transformation. Zur Anforderungsspezifikation wurden die einzelnen Schritte des Anwendungsfalls durch die Anforderungen an den Input, den Output und die Prozessmodellgenerierung aufgegriffen.

Die Anforderungen an den *Input* (die Daten, die zur Prozessmodellgenerierung verwendet werden sollen) wurden anhand der Betrachtung der Zweckeignung der Daten und Informationen zur Geschäftsprozessmodellierung (siehe Abschnitt 3.1) spezifiziert.

Daher wurden bei der Spezifizierung der Anforderungen an die Input-Daten die Aspekte *Datenformat*, *Datenqualität* und *Informationsgehalt* betrachtet.

Die Anforderungen an den *Output* (das zu generierende Modell) wurden anhand der Anforderungen an ein Geschäftsprozessmodell spezifiziert. Dementsprechend wurden die Anforderungen basierend auf den Anforderungen, die in Abschnitt 2.3 zu den Qualitätsaspekten für Geschäftsprozessmodelle betrachtet wurden, spezifiziert. Diese Anforderungen sind Anforderungen an das *Modell*, die *Modellierungssprache* sowie die *Modellierung*. Dazu wurden zusätzlich Beschriftungs- und Inschriftenkonventionen definiert, um durch diese Konventionen die Anforderungen an die Verständlichkeit und Einheitlichkeit zu unterstützen.

Bei den Anforderungen an die *Generierung* wurden insbesondere die *funktionale Eignung* der einzelnen Schritte des Anwendungsfalls berücksichtigt.

6 Generierung von Geschäftsprozessmodellen

Ein Geschäftsprozessmodell repräsentiert die Aktivitäten eines Geschäftsprozesses sowie die Beziehungen zwischen diesen Aktivitäten (siehe Abschnitt 2.1). Die Beziehung zwischen zwei Aktivitäten ist durch kausale Zusammenhänge zwischen diesen Aktivitäten sowie organisatorischen Vorgaben in einem Geschäftsprozess definiert. Zusätzlich können zwei Aktivitäten auch unabhängig voneinander sein, wenn weder ein naturgegebener Zusammenhang noch eine organisatorische Vorgabe zwischen diesen Aktivitäten gegeben ist. Durch die vorhandenen und nicht-vorhandenen Zusammenhänge zwischen den Aktivitäten ergeben sich Ordnungsbeziehungen für die Aktivitäten. Beispiele dafür sind:

- Durch den kausalen Zusammenhang, dass ein Brief erst versendet werden kann, nachdem dieser gedruckt wurde, liegt zwischen den Aktivitäten *Brief drucken* und *Brief versenden* eine *sequenzielle* Ordnungsbeziehung vor.
- Durch die organisatorische Vorgabe, dass ein Produkt erst versendet werden soll, wenn der *Zahlungseingang* erfolgt ist, ergibt sich eine *sequenzielle* Ordnungsbeziehung zwischen den Aktivitäten *Zahlungseingang prüfen* und *Produkt versenden*.
- Existiert keine organisatorische Vorgabe (und kein kausaler Zusammenhang), dass das Produkt erst versendet werden darf (kann), wenn der Zahlungseingang geprüft wurde, finden diese Aktivitäten *unabhängig* voneinander statt. Daher können diese Aktivitäten in einer beliebigen Reihenfolge ausgeführt werden. Zwischen diesen Aktivitäten liegt eine *nebenläufige* Ordnungsbeziehung vor.
- Wird ein bestimmtes Produkt entweder vom Kunden abgeholt oder dem Kunden geliefert, schließen sich die beiden Aktivitäten naturgegeben gegenseitig aus. Zwischen diesen Aktivitäten liegt somit eine *alternative* Ordnungsbeziehung vor.
- Durch die organisatorische Vorgabe, dass ein Kunde entweder die Rechnung per Mail oder per Post bekommen soll, ergibt sich zwischen den Aktivitäten *Rechnung per Mail versenden* und *Rechnung per Post versenden* eine *alternative* Ordnungsbeziehung.

Damit ergeben sich durch die Beziehungen zwischen den Aktivitäten *sequenzielle*, *nebenläufige* oder *alternative* Ordnungsbeziehungen. Entsprechend Abbildung 3-3 können diese in einem Geschäftsprozessmodell in Form eines Petri-Netzes abgebildet werden.

Wie in Abschnitt 2.1.2 beschrieben, wird als Geschäftsprozesstyp der allgemeine Rahmen eines Geschäftsprozesses bezeichnet, der als Beschreibung der Aktivitätstypen (siehe Definition 6-2) und deren Ordnungsbeziehungen gegeben ist, ohne jedoch auf konkrete Aktivitäten (Aktivitätsinstanzen) einzugehen. In dieser Arbeit ergibt sich Definition 6-1 für einen Geschäftsprozesstyp. Für die folgenden Definitionen gilt, dass *OT* die Menge aller Objekttypen und *AT* die Menge aller Aktivitätstypen eines Geschäftsprozesstyps (siehe Definition 6-1) sowie *AI* die Menge aller Aktivitätsinstanzen und *OI* die Menge aller Objektinstanzen einer Prozessinstanz (Definition 6-6) sind.

Definition 6-1: Geschäftsprozesstyp

Ein Geschäftsprozesstyp ist ein Tupel $gpt = (id, name, AT,$ ordnungsbeziehungen) mit:

- einem eindeutigen Identifikator *id,*
- einer Bezeichnung *name,*
- einer Menge *AT* von Aktivitätstypen,
- einer Menge von Ordnungsbeziehungen ordnungsbeziehungen $\subseteq AT \times AT \times$ *beziehungstyp*,

beziehungstyp $\in$ {SEQUENZIELL, ALTERNATIV, NEBENLÄUFIG}.

Die Ordnungsbeziehung zwischen zwei Aktivitätstypen ist eindeutig. Daher gilt für die Ordnungsbeziehungen: $\forall at_1, at_2 \in \text{AT}: |\{b \mid (at_1, at_2, b) \in ordnungsbeziehungen\}| \leq 1$

Aktivitäten erzeugen (ein physischer *Brief* wird durch die Aktivität *Brief drucken* erzeugt), verbrauchen (der Verkauf eines Produktes mindert den Lagerbestand) oder verändern (durch einen Zahlungseingang wird der Status einer Rechnung zu bezahlt geändert) Objekte. Außerdem können Objekte Informationen für Aktivitäten liefern (Informationen zu einem Kunden werden in einer Rechnung übernommen). Objekte sind somit Input und/oder Output für Aktivitäten. Daher werden in dieser Arbeit Aktivitätstypen über die betreffenden Objekttypen sowie den Regeln, denen die Aktivitäten folgen müssen, definiert. Diese Regeln können entweder angeben, wie die Objekte bearbeitet werden oder welche Bedingungen bezüglich der Objekte zutreffen müssen, um eine Aktivität auszuführen (siehe Definition 6-2). Für die Input- bzw. Output-Objekttypen können ebenfalls Kardinalitäten[1] angegeben werden. Die Kardinalität gibt an, wie viele Instanzen eines Objekttyps als Input erforderlich sind bzw. als Output erzeugt werden.

[1] Zusätzlich zu den Kardinalitäten können zu den Objekttypen auch zusätzlich Filter wie in XML-Netzen [LeOb03] angegeben werden, um nur auf Teilstrukturen einer Objektinstanz zuzugreifen. Dies wird in dieser Arbeit nicht betrachtet und daher die Kardinalitäten auf die natürlichen Zahlen eingeschränkt. Objektmanipulationen werden über die Regeln der Aktivitäten abgebildet.

Definition 6-2: Aktivitätstyp

Ein Aktivitätstyp ist ein Tupel $at = (id, name, OT\_in, OT\_out, regeln)$ mit:

- einen eindeutigen Identifikator *id,*
- einer Bezeichnung *name,*
- einer Menge von Input-Objekttypen $OT\_in = \{(ot, kardinalität) \mid ot \in OT \wedge kardinalität \in \mathbb{N}\}$,
- einer Menge von Output-Objekttypen $OT\_out = \{(ot, kardinalität) \mid ot \in OT \wedge kardinalität \in \mathbb{N}\}$,
- einem Tupel $regeln = (AB, OM)$, wobei *AB* eine Menge von Ausführungsbedingungen und *OM* eine Menge von Objektmanipulationen sind.

In einem Geschäftsprozess sind Objekttypen beispielsweise *Bestellung*, *Rechnung* oder *Lieferschein* und demnach mögliche Objektinstanzen die konkrete *Bestellung B1* mit zugehöriger *Rechnung R1* und *Lieferschein L1*. Jeder Objekttyp wird durch charakteristische Eigenschaften beschrieben. Eine Bestellung enthält beispielsweise eine Bestellnummer, ein Bestelldatum und einen Gesamtbetrag. Diese Eigenschaften werden durch Attribute definiert, wobei jedes Attribut einen Namen und einen zulässigen Wertebereich hat. Eine Objektinstanz ist eine konkrete Ausprägung eines Objekttyps, bei der diesen Attributen dann spezifische Werte zugewiesen sind [Lins02, Zimm99]. Objekte werden wie in Abschnitt 2.1.2 beschrieben danach differenziert, ob sie einer einzelnen Prozessinstanz zugeordnet werden können (Prozessinstanzobjekte) oder über Prozessinstanzen hinweg existieren (prozessinstanzunabhängige Objekte). In dieser Arbeit ergibt sich damit die Definition 6-3 für Objekttypen.

Definition 6-3: Objekttyp

Ein Objekttyp ist ein Tupel $ot = (id, name, kategorie, attribute)$ mit:

- einem eindeutigen Identifikator *id*,
- einer Bezeichnung *name,*
- einer Kategorie $kategorie \in \{\text{PROZESSINSTANZABHÄNGIG}, \text{PROZESSINSTANZUNABHÄNGIG}\}$, die angibt, ob Instanzen von *ot* abhängig bzw. unabhängig zu einer Prozessinstanz sind,
- einer Menge von Attributdefinitionen $attribute = \{(name, wertebereich)\}$.

Die Regeln eines Aktivitätstyps müssen sich auf die im Aktivitätstyp definierten Objekttypen beziehen und nur die in den Objekttypen definierten *Attribute* betreffen. Daher sind die folgenden Integritätsbedingungen definiert:

Für die Ausführungsbedingungen *AB* und Objektmanipulationen *OM* gilt:

$$AB \subseteq \text{EXPR}(OT\_in),$$

$$OM \subseteq \text{EXPR}(OT\_in \cup OT\_out),$$

wobei EXPR(M) die Menge aller gültigen Ausdrücke über den Objekttypen aus M bezeichnet.

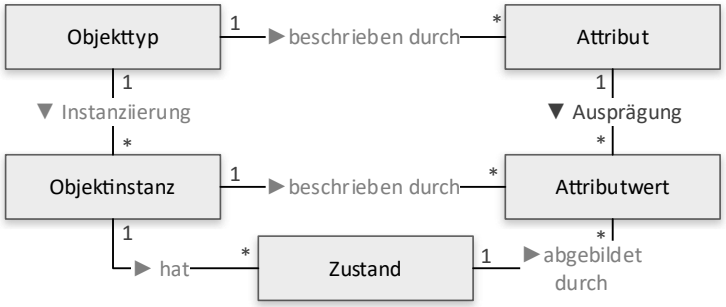
Für alle Ausdrücke $a \in \text{EXPR(M)}$ gilt:

$\forall$ *attr* $\in$ ATTR(a), wobei ATTR(a) die Menge aller in a referenzierten Attribute bezeichnet:

$\exists ot \in$ M mit $ot = (ot\_id, ot\_name, ot\_kategorie, ot\_attribute)$: $\exists(n, w) \in ot\_attribute: attr = n$

$\wedge$ *wert* $\in w$, falls a: $attr = wert$.

Wie in Abbildung 2-2 dargestellt, sind Instanzen konkrete Ausprägungen zu einem Typ. Diese Instanzen haben einen Zustand, der sich über die Zeit verändern kann. Der Zustand einer Instanz ist daher durch die Gesamtheit aller Eigenschaften zu einem bestimmten Zeitpunkt definiert (siehe Abschnitt 2.1.2). Bei Objektinstanzen ist die Gesamtheit aller Eigenschaften durch die Zuweisung von konkreten Werten zu denen im Objekttyp definierten Attributen gegeben. Dies ist in Abbildung 6-1 dargestellt.



in Anlehnung an [CoYo91, BKAP+23]

Abbildung 6-1: Objekttypen und -instanzen

Damit ergibt sich die Definition 6-4 für Objektinstanzen.

Definition 6-4: Objektinstanz[2]

Eine Objektinstanz ist ein Tupel $oi = (id, ot\_typ, zustand)$ mit:

- einem eindeutigen Identifikator *id*,
- einem Objekttyp $ot\_typ \in OT$ mit $ot\_typ = (ot\_id, ot\_name, ot\_kategorie, ot\_attribute)$,
- einem Zustand $zustand = \{(attribut\_name, wert) \mid \exists(n, w) \in ot\_attribute: attribut\_name = n \wedge wert \in w\}$.

[2] Die Objektinstanz ist durch die Gesamtheit aller Eigenschaften zu einem bestimmten Zeitpunkt t definiert. Für Prozessinstanzobjekte entspricht dieser *Zeitpunkt t* dem repräsentativen Zeitpunkt einer Objektinstanz während der Prozessausführung. Dieser Zeitpunkt ist für die zeitlichen Zusammenhänge zwischen Objektinstanzen relevant.

Objektinstanzen werden erzeugt (instanziiert), gelesen, bearbeitet und verbraucht. Wird eine Objektinstanz bearbeitet, verändert sich der Zustand. Damit gilt ein Zustand in einem konkreten Zeitintervall. Die Zustandsübergänge von Objektinstanzen identifizieren Aktivitäten. Eine Aktivitätsinstanz ist definiert über die erzeugten, gelesenen, bearbeiteten oder verbrauchten Objektinstanzen der im Aktivitätstyp definierten Objekttypen (Definition 6-5). Der Zustand einer Aktivitätsinstanz (siehe Abbildung 2-5) ist von den beteiligten Objektinstanzen abhängig. Eine Aktivitätsinstanz ist im Zustand *bereit*, wenn die notwendigen Input-Objektinstanzen vorliegen und die Ausführungsbedingungen für diese erfüllt sind, aber die Ausführung der Aktivität noch nicht begonnen wurde. Wenn die Output-Objektinstanzen entsprechend den Objektmanipulationen erzeugt wurden, ist die Aktivitätsinstanz im Zustand *regulär beendet*.

Definition 6-5: Aktivitätsinstanz

Eine Aktivitätsinstanz ist ein Tupel *ai* = (*id*, *at_typ*, *OI_in*, *OI_out*, *zustand*) mit:

- einem eindeutigen Identifikator *id*,
- einem Aktivitätstyp *at_typ* ∈ *AT,*
- einer Menge von Input-Objektinstanzen *OI_in* ⊆ *OI*, die Instanzen der für den Aktivitätstyp *at_typ* definierten Objekttypen aus *OT_in* sind,
- einer Menge von Output-Objektinstanzen *OI_out* ⊆ *OI*, die Instanzen der für den Aktivitätstyp *at_typ* definierten Objekttypen aus *OT_out* sind,
- einem Zustand *zustand* ∈ {VOR-AUSFÜHRUNG, BEREIT, AKTIV, REGULÄR BEENDET, GESTÖRT, ABGEBROCHEN}.

Zusätzlich gelten für alle Aktivitätsinstanzen *ai* = (*id*, *at_typ*, *OI_in*, *OI_out*, *zustand*) die folgenden Integritätsbedingungen: Sei der Aktivitätstyp der Aktivitätsinstanz *at_typ* = (*id*, *name*, *OT_in*, *OT_out*, *regeln*) und die Objektinstanzen der Input-Objektinstanzen bzw. Output-Objektinstanzen der Aktivitätsinstanz *oi* = (*oi_id*, *ot_typ*, *eigenschaften*):

- Für die Input-Objektinstanzen gilt:
  - ∀ *oi* ∈ *OI_in*: ∃(*ot*, *kardinalität*) ∈ *OT_in*: *ot_typ* = *ot*,
  - ∀(*ot*, *kardinalität*) ∈ *OT_in*: |{*oi* ∈ *OI_in* | *ot_typ* = *ot*}| = *kardinalität,*
- Für die Output-Objektinstanzen gilt:
  - ∀ *oi* ∈ *OI_out*: ∃(*ot*, *kardinalität*) ∈ *OT_out*: *ot_typ* = *ot,*
  - ∀(*ot*, *kardinalität*) ∈ OT_out: |{*oi* ∈ *OI_out* | *ot_typ* = *ot*}| = *kardinalität.*

Eine Prozessinstanz ist eine konkrete Ausprägung des Geschäftsprozesstyps, die über eine Menge der Aktivitätsinstanzen definiert wird. Über die Aktivitätsinstanzen kann der Zustand einer Prozessinstanz (siehe Abbildung 2-5) abgeleitet werden. Mögliche Zustände einer Prozessinstanz sind "*vor-Ausführung*", "*aktiv*", "*wartend*", "*gestört*", "*abgebro-*

chen" und "*regulär beendet*" (siehe Abschnitt 2.1.2). In dieser Arbeit ergibt sich Definition 6-6 für Prozessinstanzen.

Definition 6-6: Prozessinstanz

Eine Prozessinstanz ist ein Tupel $pi = (id, gpt\_typ, AI, zustand)$ mit:

- einen eindeutigen Identifikator *id,*
- einen Geschäftsprozesstyp $gpt\_typ \in GPT$,
- einer Menge *AI* von Aktivitätsinstanzen, die Instanzen von den im Geschäftsprozesstyp definierten *AT* sind,
- einem Zustand *zustand* ∈ {VOR-AUSFÜHRUNG, BEREIT, AKTIV, REGULÄR BEENDET, GESTÖRT, ABGEBROCHEN}.

Durch die definierten Ordnungsbeziehungen zwischen den Aktivitätstypen ergeben sich die möglichen Ausführungsreihenfolgen der Aktivitätsinstanzen in einer Prozessinstanz. Die Ausführungsreihenfolge der Aktivitätsinstanzen wird beschrieben durch $(ai_1, ..., ai_n)$ mit $ai_x \in AI$ mit $x \in \{1, ..., n\}$.

Für Prozessinstanzen mit den entsprechenden Ordnungsbeziehungen des Geschäftsprozesstyps gelten die folgenden Integritätsbedingungen:

1. Für die Ausführungsreihenfolge = $(ai_1, ..., ai_n)$ mit $ai_x \in$ AI und $ai_x = (at\_typ_x, OI\_in, OI\_out, zustand)$ mit $x \in \{1, ..., n\}$ gilt:
   - Vollständigkeit: $\{ai_1, ..., ai_n\}$ = AI,
   - Ordnungskonformität: $\forall i,j \in \{1, ..., n\}$:
     1. falls $(at\_typ_i, at\_typ_j,$ SEQUENZIELL$) \in ordnungsbeziehungen$
        $\Longrightarrow i < j,$
     2. falls $(at\_typ_i, at\_typ_j,$ UNABHÄNGIG$) \in ordnungsbeziehungen$
        $\Longrightarrow$ beliebige Reihenfolge erlaubt,
     3. falls $(at\_typ_i, at\_typ_j,$ ALTERNATIV$) \in ordnungsbeziehungen$
        $\Longrightarrow \neg(ai_i \in AI \wedge ai_j \in AI)$.
2. Falls eine Objektinstanz $oi \in$ OI ein Prozessinstanzobjekt ist, ist *oi* nur in Aktivitätsinstanzen einer Prozessinstanz Input- bzw. Output.

Basierend auf den Objektinstanzen, die in den Ausführungen des Geschäftsprozesstyps (den Prozessinstanzen) aufgetreten sind, soll der Geschäftsprozesstyp abgeleitet werden. Dazu werden Dateien, die in ihren Rohdaten (siehe Definition 7-1) diese Objektinstanzen

beschreiben, vom Anwendenden bereitgestellt. Eine Datei bildet den Zustand einer oder mehrerer Objektinstanz(en) eines Geschäftsprozesses zu einem bestimmten Zeitpunkt in den Rohdaten ab. Durch Aktivitäten wird der Zustand einer Prozessinstanz verändert. Sofern dabei auch die Eigenschaften der Objektinstanzen verändert werden, lässt sich die Durchführung der Aktivität und die Änderung des Zustandes anhand der beteiligten Objektinstanzen beobachten (siehe Abschnitt 2.1.2). Daher sollen durch die Betrachtung der Objektinstanzen Rückschlüsse auf die Aktivitätstypen und deren Ordnungsbeziehungen gezogen werden. Diese sollen anschließend durch ein Geschäftsprozessmodell repräsentiert werden. Dementsprechend ist eine Modellierungssprache zu wählen, die die Aktivitätstypen (AT) einschließlich ihrer Regeln, die Objekttypen (OT) einschließlich ihrer Attributdefinitionen sowie die Ordnungsbeziehungen zwischen den Aktivitätstypen adäquat darstellen kann. Darüber hinaus wird von einer endlichen Menge von Objektinstanzen (OI) ausgegangen, die konkrete Ausprägungen der Objekttypen repräsentieren. Dazu werden in dieser Arbeit höhere Petri-Netze gewählt, die die Grundlage für die Definition 6-7 eines objektbasierten Geschäftsprozessmodells bilden.

Definition 6-7: Objektbasiertes Geschäftsprozessmodell

Ein objektbasiertes Geschäftsprozessmodell ist ein Tupel $GPM = (S, T, F, SB, TB, TI, M_0)$ mit:

i. S, T sind endliche Mengen von Stellen und Transitionen mit $S \cap T = \emptyset$, $S \cup T \neq \emptyset$. Dabei werden Stellen $s \in S$ als Objekttypen interpretiert und Transitionen $t \in T$ als Aktivitätstypen.

ii. $F \subseteq (S \times T) \cup (T \times S)$ ist Flussrelation, wobei für jede Transition $t \in T$ gilt:
   - $\bullet t = \{s \in S \mid (s, t) \in F\}$ entspricht der Menge aller Stellen im Vorbereich der Transition t,
   - $t\bullet = \{s \in S \mid (t, s) \in F\}$ entspricht der Menge aller Stellen im Nachbereich der Transition t.

iii. Die Stellenbeschriftung SB: $S \rightarrow OT$ weist jeder Stelle $s \in S$ einen Objekttyp $ot \in OT$ zu.

iv. Die Transitionsbeschriftung TB: $T \rightarrow AT$ weist jeder Transition $t \in T$ einen Aktivitätstyp $at \in AT$ zu, wobei für jede Transition $t \in T$ gilt:
   - Die Objekttypen der Stellen in $\bullet t$ entsprechen den Input-Objekttypen von $TB(t)$,
   - Die Objekttypen der Stellen in $t\bullet$ entsprechen den Output-Objekttypen von $TB(t)$.

v. Die Transitionsinschriften TI: T → (AB × OM) weisen jeder Transition t ∈ T die Regeln TI(t) = (AB(t), OM(t)) ihres zugehörigen Aktivitätstyps TB(t) zu, wobei:
   - AB(t) ⊆ EXPR({SB(s) | s ∈ •t}),
   - OM(t) ⊆ EXPR({SB(s) | s ∈ •t ∪ t•})

vi. Die Anfangsmarkierung M_0 weist den Stellen $s \in S$ Objektinstanzen $oi \in OI$ zu.

Die Integritätsbedingungen für das Geschäftsprozessmodell sind:

- Ein Geschäftsprozess ist eine Menge von Aktivitäten, die in einem Betrieb nach bestimmten Regeln, auf ein bestimmtes Ziel hin ausgeführt werden und die Aktivitäten hängen über betroffene Personen, Maschinen, Dokumente u. ä. miteinander zusammenhängen. Daher ist in dieser Arbeit ein Geschäftsprozesstyp mit Aktivitätstypen beschrieben, die über die entsprechenden Objekttypen zusammenhängen. Somit ist das Geschäftsprozessmodell zusammenhängend. Ein Geschäftsprozessmodell GPM = (S, T, F, SI, TI, AI, M_0) ist zusammenhängend, wenn für alle Knoten x,y ∈ (S ∪ T) gilt, dass ein ungerichteter Pfad von x nach y existiert: ∀x,y ∈ (S ∪ T): ∃(k_1, ..., k_n) mit $k_1 = x \wedge k_n = y$: ∀i ∈ {1, ..., n-1}: (k_i, k_{i+1}) ∈ F ∨ (k_{i+1}, k_i) ∈ F.
- Die Flussrelationen F des Geschäftsprozessmodells GPM müssen die definierten Ordnungsbeziehungen zwischen den Aktivitätstypen des Geschäftsprozesstyps einhalten.
    - ∀(at_1, at_2, SEQUENZIELL): Für alle t_1, t_2 ∈ T mit $TB(t_1) = at_1$, $TB(t_2) = at_2$ existiert ein Pfad von t_1 nach t_2.
    - ∀(at_1, at_2, ALTERNATIV): Für alle t_1, t_2 ∈ T mit $TB(t_1) = at_1$, $TB(t_2) = at_2$ existiert eine gemeinsame Eingangsstelle.
    - ∀(at_1, at_2, UNABHÄNGIG): Für alle t_1, t_2 ∈ T mit $TB(t_1) = at_1$, $TB(t_2) = at_2$ sind die Transitionen nebenläufig aktivierbar.

Wie in Abbildung 5-1 dargestellt, sind zur Generierung von Geschäftsprozessmodellen basierend auf Objektinstanzen die Schritte Informationsextraktion und Transformation nacheinander durchzuführen. Um aus den bereitgestellten Dateien ein Geschäftsprozessmodell generieren zu können, müssen zunächst die relevanten Informationen für den Geschäftsprozesstyp aus den Rohdaten der Dateien zu den Objektinstanzen extrahiert werden (siehe Abschnitt 6.1). Anschließend sollen die Informationen in einem Geschäftsprozessmodell repräsentiert werden (siehe Abschnitt 6.2). Jeder dieser Schritte soll automatisiert durchgeführt werden.

6.1 Informationsextraktion

Das zu generierende Geschäftsprozessmodell soll sowohl die Ordnungsbeziehungen zwischen den Aktivitätstypen als auch Informationen zu den Objekttypen (siehe Abschnitt 5.3, Anforderung AG05) repräsentieren. Diese Informationen sind aus den bereitgestellten Dateien, deren Rohdaten die Objektinstanzen eines Geschäftsprozesses beschreiben, zu extrahieren. Aufbauend auf der Abbildung 2-3, in der der Zusammenhang zwischen Objekten und Aktivitäten auf Instanz-Ebene dargestellt wurde, wird in Abbildung 6-2 betrachtet, wie aus den bereitgestellten Daten zu den Objektinstanzen zunächst deren Zustände (R1) und daraus die Objekttypen (R2) innerhalb eines Geschäftsprozesses

identifiziert werden. Anschließend werden die Objektbeziehungstypen zwischen den Objekttypen über Zusammenhänge bezüglich der Zustände einzelner Objektinstanzen (R3a) sowie daraus resultierende Aktivitätstypen mit einzuhaltenden Regeln (R3b) extrahiert. Aus den Objekttypen und Aktivitätstypen sollen Objektspeicher und Transitionen abgeleitet werden, um ein Geschäftsprozessmodell nach Definition 6-7 in Form eines Petri-Netzes zu generieren. Zusätzlich könnten die Objektinstanzen als Marken in Petri-Netzen abgebildet werden. Die beschriebenen Rückschlüsse, die auf die Informationsextraktion zu Objekttypen und Aktivitätstypen zielen, werden nachfolgend weiter beschrieben. Die Transformation der extrahierten Informationen zu einem Geschäftsprozessmodell wird in Abschnitt 6.2 betrachtet.

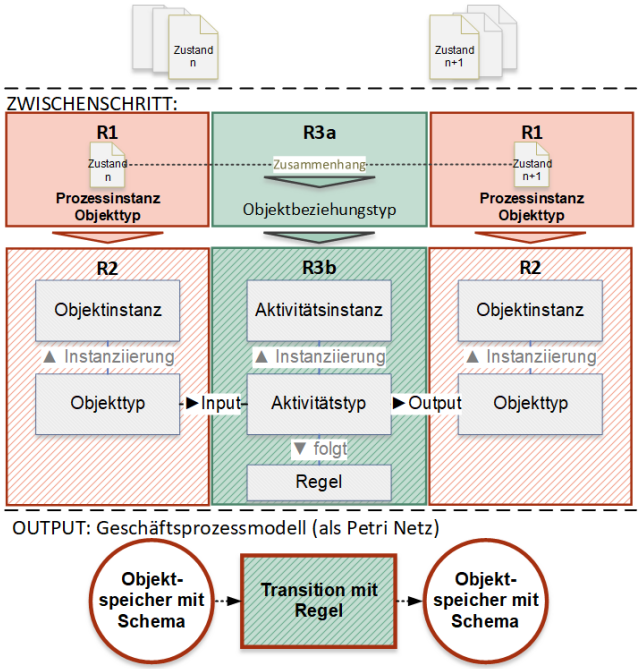


Abbildung 6-2: Mögliche Rückschlüsse aus Objektinstanzen

Zur Informationsextraktion aus den Objektinstanzen zu den Objekttypen, Objektbeziehungstypen und Aktivitätstypen sind die Objektinstanzen zunächst zu Objekttypen und zu Prozessinstanzen zuzuordnen. Die Zuordnung der Objektinstanzen zu *Objekttypen* und *Prozessinstanzen* ist in Abbildung 6-3 verdeutlicht.

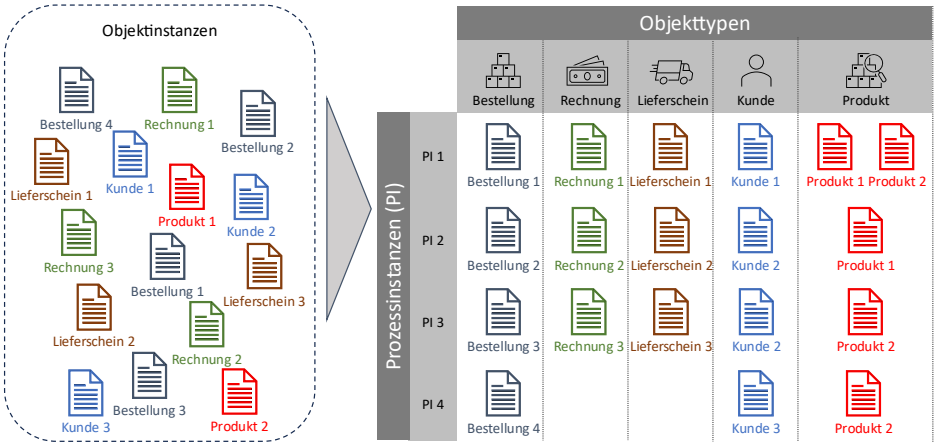


Abbildung 6-3: Zuordnung der Objektinstanzen zu Objekttypen und Prozessinstanzen

Erst nach dieser Zuordnung können aus der Betrachtung der Zustände der Objektinstanzen zunächst die Objekttypen definiert und anschließend Rückschlüsse zu Objektbeziehungstypen gezogen werden. Aus den Objektbeziehungstypen können dann die Aktivitätstypen abgeleitet werden. Die zu extrahierenden Informationen je Betrachtungsgegenstand werden in Tabelle 6-1 zusammengefasst.

Tabelle 6-1: Informationsextraktion zu Objekttypen und deren Zusammenhänge

Betrachtungsgegenstand	**Betrachtete Objektinstanzen**	**Informationsextraktion**
Objekttyp (R2)	Objektinstanzen *eines* Objekttyps aus einer Prozessinstanz	Schlüsselattribute, um Objektinstanzen eines Objekttyps eindeutig identifizieren zu können und Versionen zu erkennen.
	Objektinstanzen *eines* Objekttyps aus unterschiedlichen Prozessinstanzen	Über Prozessinstanzen hinweg gültiger Objekttyp und dessen Attribute sowie Einschränkungen für diese.
Objektbeziehungstyp (R3a)	Objektinstanzen *verschiedener* Objekttypen aus einer Prozessinstanz	Beobachtete Zusammenhänge zwischen Objektinstanzen durch Übernahme von Attributnamen mit geändertem Wert, Werte mit Änderung des Attributnamens oder Attribut-Wert-Paare sowie beobachtbare Zusammenhänge durch deren zeitliches Auftreten.
	Objektinstanzen *verschiedener* Objekttypen aus unterschiedlichen Prozessinstanzen	Für die Betrachtung von Regeln (R3b) sind die Objektinstanzen einer Prozessinstanz in einem Vergleich zu anderen Prozessinstanzen zu berücksichtigen.

Um die Informationen aus den Objektinstanzen zur Prozessmodellgenerierung zu extrahieren, lassen sich die in Tabelle 6-2 beschrieben Extraktionsschritte definieren.

Tabelle 6-2: Extraktionsschritte

Extraktionsschritt	Beschreibung
Zuordnung zu Objekttypen	Liegen zu den Objektinstanzen keine Typinformationen vor, ist sowohl unbekannt, welche Objekttypen in einem Geschäftsprozess auftreten als auch welche Objektinstanzen von welchem Objekttyp vorliegen. Daher werden die Objektinstanzen nach ihrer Struktur zu Objekttypen klassifiziert.
Zuordnung zu Prozessinstanzen	Sind die Objektinstanzen keinen Prozessinstanzen zugeordnet, ist unbekannt, welche Objektinstanzen innerhalb welcher Prozessinstanz erstellt, gelesen, bearbeitet oder verbraucht wurden und welche Objektinstanzen über welche Prozessinstanz zusammenhängen. Daher werden die Objektinstanzen Prozessinstanzen zugeordnet.
Extraktion von Objekttypen	Auch wenn die Objektinstanzen nach Objekttypen klassifiziert sind, ist noch nicht definiert, welche Objektstruktur diese haben. Daher sind aus den zugeordneten Objektinstanzen zu einem Objekttyp die Struktur und die Bezeichnung abzuleiten.
Extraktion von Objektbeziehungstypen	Basierend auf den Zusammenhängen zwischen den Objektinstanzen in einer Prozessinstanz, werden die Objektbeziehungstypen abgeleitet.
Extraktion von Aktivitätstypen	Durch eine Analyse der identifizierten Objektbeziehungstypen über die verschiedenen Prozessinstanzen hinweg werden die Aktivitätstypen definiert.

Die Informationsextraktion erfolgt daher in mehreren aufeinander aufbauenden Schritten: Zunächst werden in Abschnitt 6.1.1 die Zustände der Objektinstanzen analysiert, um die Objekttypen zu identifizieren. Anschließend erfolgt in Abschnitt 6.1.2 die Zuordnung der Objektinstanzen zu Prozessinstanzen durch die Untersuchung von Zusammenhängen zwischen den Objektinstanzen. In Abschnitt 6.1.3 werden die Objekttypen definiert, indem die gemeinsamen Eigenschaften der zugeordneten Objektinstanzen analysiert werden. Aus den Zusammenhängen zwischen den Objekttypen werden in Abschnitt 6.1.4 Objektbeziehungstypen abgeleitet, die wiederum die Grundlage für die Ableitung von Aktivitätsinstanzen und -typen in Abschnitt 6.1.5 sind.

6.1.1 Zuordnung zu Objekttypen

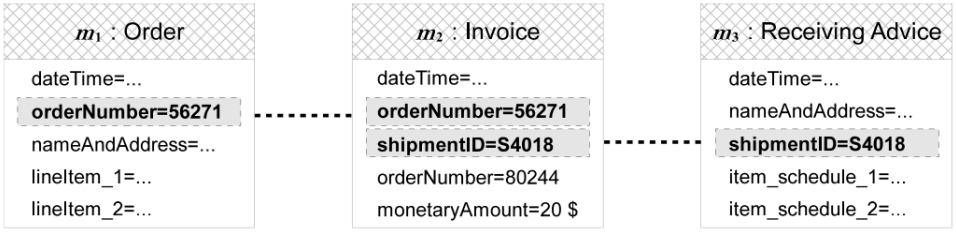
Da in dieser Arbeit Instanzen verschiedener Objekttypen aus verschiedenen Prozessinstanzen analysiert werden, sind die Objektinstanzen zunächst zu einzelnen Objekttypen zu klassifizieren. Für die Klassifizierung der Objektinstanzen zu Objekttypen kann geprüft werden, inwieweit die unterschiedlichen Zustände der Objektinstanzen Übereinstimmungen in den Datenstrukturen haben. Da der Zustand der Objektinstanzen durch die Attributwerte zu den Attributen abgebildet ist, wird die Datenstruktur durch deren auftretende Attribute und Wertebereiche der Objekte vorgegeben. Daher ist zur Klassifizierung der Objektinstanzen die Datenstruktur unabhängig von ihrer Verschachtelungstiefe zu betrachten, um alle verfügbaren Attribute einzubeziehen. Anhand maschineller Lernverfahren können Typen basierend auf den Übereinstimmungen gebildet werden. Um eine konsistente Datengrundlage zu erhalten, sind die extrahierten Attribute für die weitere Analyse zu bereinigen, indem beispielsweise Stoppwörter entfernt und die Texte normalisiert werden. Diese Vorverarbeitungsschritte sind entscheidend, um Rauschen zu reduzieren und die Qualität der nachfolgenden Analyse zu verbessern. Da die Attributstrukturen der Objektinstanzen vor der Analyse nicht bekannt sind, werden unüberwachte Lernverfahren zur Klassifizierung eingesetzt. Der K-Means-Clustering-Algorithmus [JaDu88] teilt Datenpunkte ohne vordefinierte Klassenzugehörigkeit in k disjunkte Cluster ein. Zur Bestimmung der Cluster-Anzahl kann beispielsweise der Silhouette-Score [Rous87] verwendet werden, der die Qualität der Cluster-Zuordnung für jeden Datenpunkt bewertet. Der Silhouette-Score bewertet, wie ähnlich eine Objektinstanz zu seinem eigenen Cluster im Vergleich zu anderen Clustern ist. Ein hoher Silhouette-Score deutet auf gut definierte und trennbare Cluster hin. Zusätzlich kann die Summe der quadratischen Abweichungen [TiWH01] analysiert werden. Diese Metrik hilft dabei, den Punkt zu identifizieren, an dem zusätzliche Cluster keine signifikante Verbesserung der Clusterqualität mehr bringen.

6.1.2 Zuordnung zu Prozessinstanzen

Für die Rückschlusse, die aus den Objektinstanzen abgeleitet werden sollen, werden deren Zustände analysiert. Anhand der Zustände sollen die Objektinstanzen einzelnen Prozessinstanzen zugeordnet werden. Dafür sind beispielsweise über verschiedene Objektinstanzen (die zum selben oder zu verschiedenen Objekttypen gehören können) hinweg Überschneidungen in den Werten zu finden, um diese den jeweiligen Prozessinstanzen zuordnen zu können. Während diese Zuordnung bei Instanzen von Prozessinstanzobjekten eindeutig ist, ist sie bei Instanzen von prozessinstanzunabhängigen Objekten nicht eindeutig, da diese mehreren Prozessinstanzen zugeordnet sein können. Abbildung 6-3 zeigt auch, dass die Instanzen von prozessinstanzunabhängigen Objekten

(hier: Daten zu Kunden und Produkten) mehreren Prozessinstanzen zugeordnet werden können und somit Duplikate in der Tabelle enthalten sind.

Für diese Zuordnung wird von [EAZP+12] ein Verfahren basierend auf EDIFACT-Nachrichten vorgestellt. Diese definieren in den EDIFACT-Nachrichten Korrelatoren als Schlüssel für ein bestimmtes Datenelement, das in einem oder mehreren EDI-Nachrichtentypen definiert ist und zur Korrelation von diesen Nachrichten mit Prozessinstanzen verwendet werden kann. Dazu werden die Werte, die in dem Datenelement enthalten sind, abgeglichen. Nachrichten werden bestimmten Prozessinstanzen zugeordnet, wenn jede von ihnen das durch den Korrelator bezeichnete Datenelement enthält und die Werte in den Datenelementen übereinstimmen. Dieser Korrelator kann sich über die EDIFACT-Nachrichten verändern. Ebenso kann der Korrelator auch unterschiedlich benannt sein, solange die Werte übereinstimmen. Mit diesem Ansatz kann die Zugehörigkeit zu einer Prozessinstanz aus dem gegenseitigen Vergleich von Objektinstanzen über verschiedene Prozessinstanzen hinweg ermittelt werden. Das Vorgehen wird in Abbildung 6-4 dargestellt.



[EAZP+12]

Abbildung 6-4: Zusammenhänge von EDIFACT-Nachrichten in einer Prozessinstanz

Einige Datenelemente können jedoch nicht als Korrelatoren dienen, da diese beispielsweise zufällig wiederholte Werte in mehreren Nachrichten enthalten können (z. B. einzelne Zahlenwerte, Boolean-Werte, Ortsnamen, …) oder die enthaltene Information ist möglicherweise nicht mit einzelnen Prozessinstanzen verbunden (z. B. Kunde als Instanz eines prozessinstanzunabhängigen Objekttyps) [EAZP+12]. Daher ist zum einen zu betrachten, welche Korrelatoren gewählt werden können und zum anderen, welche Objekttypen prozessinstanzunabhängig sind und in welchen (i. d. R. mehreren) Prozessinstanzen deren Objektinstanzen auftreten.

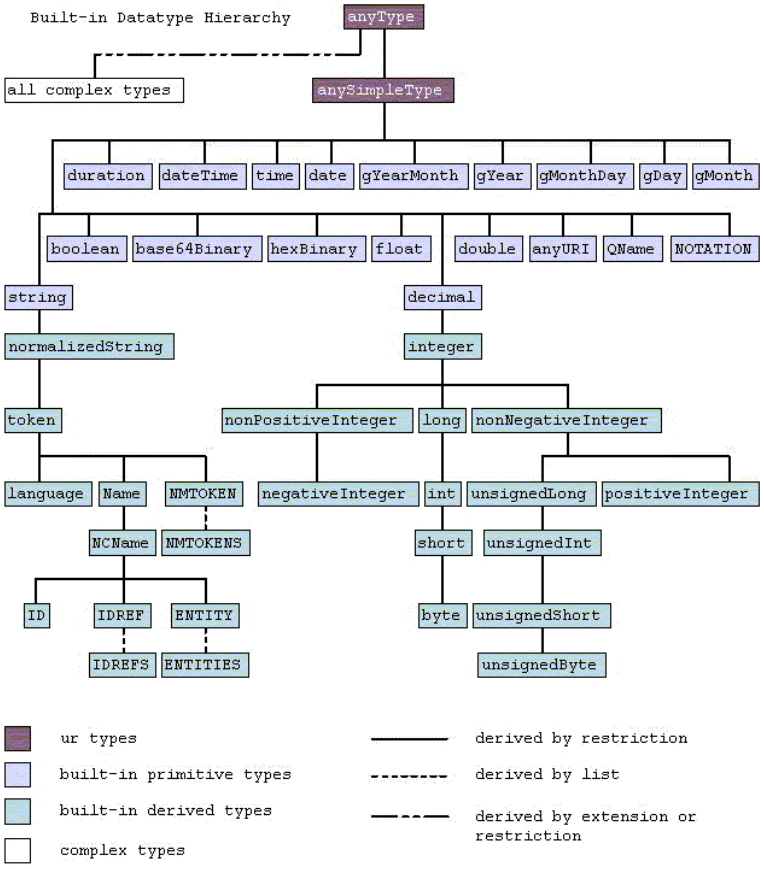
Wurden die Objektinstanzen zu den Prozessinstanzen zugeordnet und in Attribut-Strukturähnliche Cluster eingeteilt, ist über die Prozessinstanzen hinweg zu prüfen, ob mehrere Objektinstanzen innerhalb einer Prozessinstanz zu einem Cluster existieren. Ist dies der Fall, ist anhand der Inhalte der Objektinstanzen zu bewerten, ob diese unterschiedliche Objektinstanzen darstellen oder Versionen von Objektinstanzen. Versionen

liegen vor, wenn insbesondere Schlüsselattribute identisch sind. Zusätzlich können die Attribute auf Versionseigenschaften und die Titel der Objektinstanzen geprüft werden. Werden Versionen entdeckt, wird bei den Objektinstanzen ein Versionsattribut ergänzt.

6.1.3 Definition Objekttypen

Da die Objektinstanzen konkrete Ausprägungen der in einem Geschäftsprozess vorkommenden Objekttypen beschreiben, können die Strukturen der Objekttypen aus der Analyse der Zustände der verschiedenen Objektinstanzen abgeleitet werden. Zur Analyse der Zustände wird die Attribut-Struktur (das Schema) der Objektinstanzen abgeleitet, um die Objekttypen zu definieren. Werden die Objektinstanzen im XML-Format dargestellt, können die Attribut-Strukturen der Objekttypen über ein XML-Schema definiert werden[3]. Dieses legt fest, über welche Attribute mit definiertem Datentyp ein Zustand einer Instanz beschrieben werden kann. Ein Datentyp wird hinsichtlich seiner möglichen Werte (Wertebereich), wie diese Werte textuell dargestellt werden können (lexikalischer Bereich) und zusätzlicher Merkmale oder Einschränkungen für den Datentyp (Facetten) definiert [Worl04]. Die möglichen Datentypen in einem XML-Schema sind in Abbildung 6-5 gelistet.

[3] Zur Extrahierung eines Schemas aus einem XML- bzw. JSON-Dokument existieren bereits einige Umsetzungen. Beispielsweise wird in https://www.jetbrains.com/help/pycharm/generating-dtd.html#ws_xml_generate_dtd die Generierung einer Typbeschreibung basierend auf einer XML-Datei für Python beschrieben.



[Worl04]

Abbildung 6-5: Hierarchie der Datentypen in XML-Schema

Die Objekttypen und deren Attribute sowie Einschränkungen für diese können nach [HeNW06] aus den XML-basierten und nach [KKSS23] aus den JSON-basierten Dokumenten automatisch abgeleitet werden.

6.1.4 Objektbeziehungstypen

Wie in Abbildung 6-2 dargestellt, sind zusätzlich zu den Informationen zu den einzelnen Objekttypen auch die Beziehungen zwischen den Objekttypen zu betrachten, um Informationen zu den Aktivitäten extrahieren zu können. Wie in Abschnitt 2.1.2 beschrieben können aus den dynamischen Aspekten bezüglich der Objekte (d. h. aus den Manipulationen der Eigenschaften von Objektinstanzen) Aktivitäten abgeleitet werden. Zu den Manipulationen zählen insbesondere das Hinzufügen, Verändern und Löschen von Attribut-Wert-Paaren. Zusätzlich können jedoch auch ohne Manipulationen der Eigen-

schaften von Objektinstanzen Aktivitäten abhängig von Objekten sein, wenn beispielsweise ein Produkt zunächst vorhanden sein muss, bevor es versendet werden kann. Um Rückschlüsse auf diese Aktivitäten zu erhalten, sollen daher die Zusammenhänge zwischen den Objektinstanzen analysiert werden. Aus diesen Zusammenhängen können anschließend Objektbeziehungstypen zwischen den Objekttypen hergeleitet und dann die zu modellierenden Aktivitäten sowie deren Reihenfolge definiert werden.

Es gibt zwei Ansätze, um Informationen zu Objektbeziehungstypen (sowie anschließend zu den Aktivitätstypen, deren In- und Outputs sowie deren Ordnungsbeziehungen) basierend auf den Zusammenhängen zwischen den Objektinstanzen zu erhalten. Zum einen können Objektbeziehungstypen abgeleitet werden, wenn zwischen zwei (bzw. mehreren) Objektinstanzen innerhalb einer Prozessinstanz *inhaltliche Zusammenhänge* bezüglich der Zustände der Objektinstanzen beobachtbar sind, die Manipulationen der Eigenschaften aufzeigen. Aus den inhaltlichen Zusammenhängen ist jedoch die Richtung des Zusammenhangs der Objekte (und damit die Input- bzw. Output-Objekttypen für die daraus resultierenden Aktivitätstypen) nicht unmittelbar beobachtbar. Zum anderen kann ein *zeitlicher Zusammenhang* zwischen Objektinstanzen basierend auf deren zeitlichen Auftreten beobachtet werden. Abbildung 6-6 verdeutlicht die Möglichkeit, aus inhaltlichen bzw. zeitlichen Zusammenhängen, Objektbeziehungstypen abzuleiten. Bei Möglichkeit 1 sind zusätzliche Objektbeziehungstypen im Vergleich zu Möglichkeit 2 zu erwarten, da beispielsweise Instanzen von prozessinstanzunabhängigen Objekttypen kein Datum besitzen, das für eine einzelne Prozessinstanz repräsentativ ist und daher bei Möglichkeit 2 kein Zusammenhang entdeckt werden kann. Bei Möglichkeit 2 sind zusätzliche Objektbeziehungstypen im Vergleich zu Möglichkeit 1 zu erwarten, da beispielsweise auch Zusammenhänge zwischen zwei Objekten bestehen können, ohne dass Attribut-Wert-Paare verändert werden. Dies ist beispielsweise der Fall, wenn nur eine Ausführungsbedingung für ein Objekt zutreffen muss.

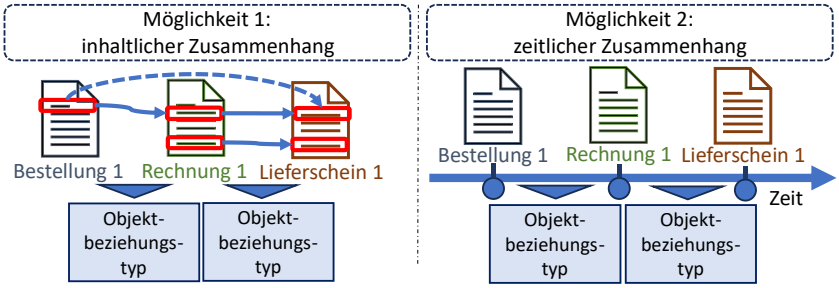


Abbildung 6-6: Informationsextraktion zu Objektbeziehungstypen

Aus diesen Zusammenhängen werden *direkte* und *indirekte* Objektbeziehungstypen gefolgert. Direkte Objektbeziehungstypen sind durch eine Übernahme von Attributen

bzw. Werten zwischen Objektinstanzen gekennzeichnet und werden daher insbesondere durch inhaltliche Zusammenhänge (Möglichkeit 1) identifiziert. Übernahmen zwischen exakt zwei Objektinstanzen werden als direkte einfache Objektbeziehungstypen bezeichnet. Es können auch mehr als zwei Objektinstanzen an einem Zusammenhang beteiligt sein. Ein direkter mehrfacher Objektbeziehungstyp, der nicht aus dem Vergleich zweier Objektinstanzen abgeleitet werden kann, ist beispielsweise vorhanden, wenn sich ein Attributwert aus der Addition mehrerer anderer Attributwerte ergibt (beispielsweise bei der Aufsummierung der Preise verschiedener Produkte, um eine Rechnungssumme zu berechnen).

Indirekte Objektbeziehungstypen bestehen, wenn lediglich das Vorhandensein oder ein bestimmtes Attribut relevant ist, ohne, dass ein Attribut oder Attributwert übernommen oder verändert wird (beispielsweise, wenn die Ware erst versendet wird, wenn ein Zahlungseingang für eine Bestellung vorhanden ist). Daher sind diese nicht unmittelbar durch Möglichkeit 1 beobachtbar, können aber durch Möglichkeit 2 identifiziert und anschließend weiter analysiert werden.

Während aus den direkten einfachen bzw. mehrfachen Objektbeziehungstypen Aktivitäten definiert werden, die Manipulationen ausführen (Regeln in der Menge von Objektmanipulationen), definieren die indirekten Objektbeziehungstypen Aktivitäten, die bestimmten Regeln folgen (Ausführungsbedingungen). Die sich damit ergebenden Objektbeziehungstypen sind:

- **Direkter einfacher Objektbeziehungstyp**: Der direkte einfache Objektbeziehungstyp ergibt sich aus inhaltlichen Zusammenhängen, bei denen Attributnamen mit geänderten Werten, Werte mit unterschiedlichen Attributnamen oder Attribut-Wert-Paare zwischen zwei Objektinstanzen einer Prozessinstanz übernommen werden.
- **Direkter mehrfacher Objektbeziehungstyp**: Der direkte mehrfache Objektbeziehungstyp bezieht sich auf Attributwerte, die sich aus mehreren Objektinstanzen ableiten lassen. Dazu sind die Werte darauf zu überprüfen, ob sie sich aus mehreren Werten anderer Objektinstanzen ableiten lassen. Dazu sind sowohl Rechenoperatoren als auch String-Funktionen sowie Boolean-Operatoren zu betrachten.
- **Indirekter einfacher Objektbeziehungstyp**: Der indirekte einfache Objektbeziehungstyp lässt sich durch das Auftreten von einer Objektinstanz bzw. deren Attribute oder Attributwerte beobachten, die zwar zu keiner Übernahme von den Attributen oder Attributwerten führen, aber zur Durchführung einer Aktivität vorhanden sein müssen.
- **Indirekter mehrfacher Objektbeziehungstyp**: Der indirekte mehrfache Objektbeziehungstyp lässt sich durch das gemeinsame Auftreten und/oder Fern-

bleiben von Objektinstanzen bzw. Attributen oder Attributwerten beobachten, die zwar zu keiner Übernahme von den Attributen oder Attributwerten führen, aber zur Durchführung einer Aktivität erfüllt sein müssen.

Möglichkeit 1: Objektbeziehungstypen durch inhaltliche Zusammenhänge

Um zwischen Objekttypen direkte Objektbeziehungstypen aus inhaltlichen Zusammenhängen zu identifizieren, sind die Zustände der Objektinstanzen relevant. Dafür sind zunächst die Übereinstimmungen zwischen den Objektinstanzen aufzudecken. Diese Übereinstimmungen sind gekennzeichnet durch:

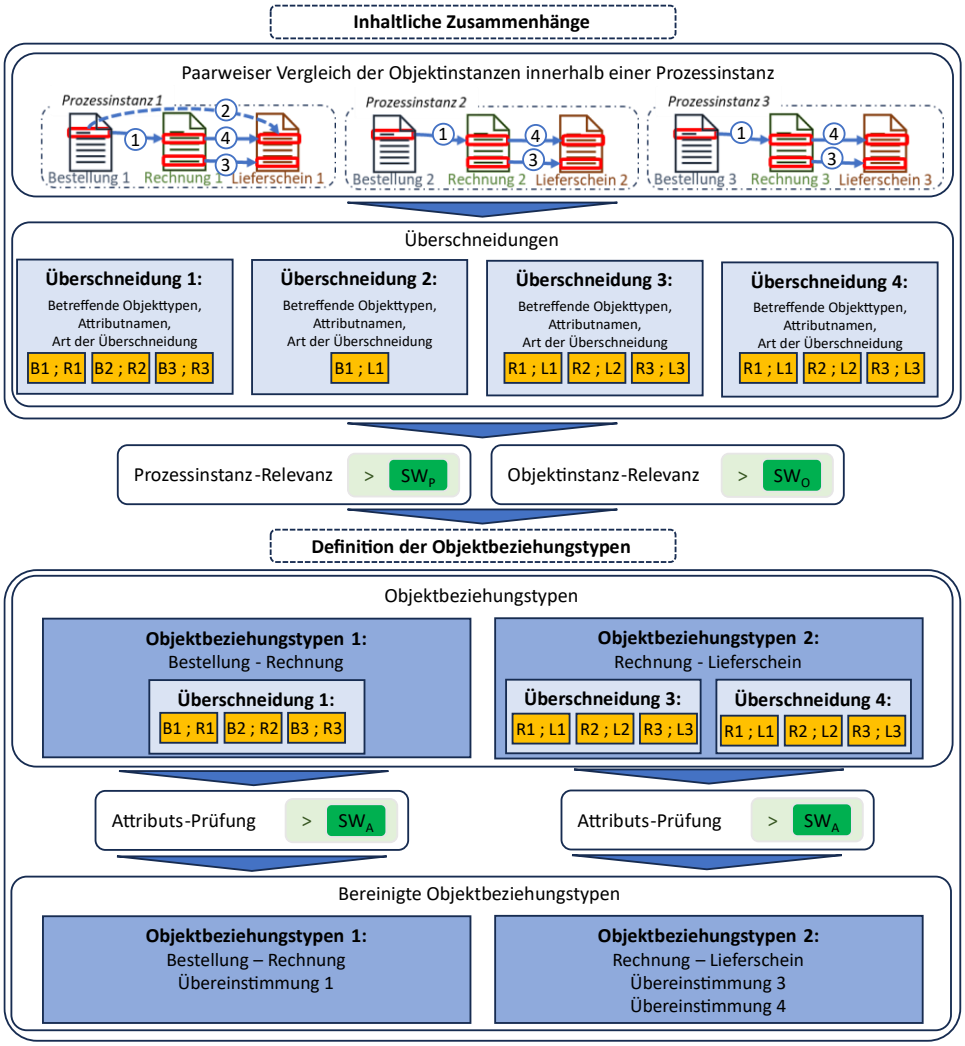
- Übernahme von Attributnamen mit geänderten Werten,
- Übernahme von Werten mit Änderung des Attributnamens,
- Übernahme von Attribut-Wert-Paaren.

Um die direkten einfachen Objektbeziehungstypen aus den inhaltlichen Zusammenhängen abzuleiten, werden die Objektinstanzen einer Prozessinstanz (über alle Prozessinstanzen hinweg) paarweise verglichen. Nicht alle gefundenen Zusammenhänge sollen zu Objektbeziehungstypen führen. Dies bezieht sich insbesondere auf häufig verwendete Attributnamen und Werte. Beispielsweise kann aus dem Vorkommen der Attributnamen wie *Name*, *Datum*, *caseID* kein Zusammenhang geschlossen werden. Ebenso sind Werte wie *true*, *false* und Ziffern in ihrer Verwendung häufig, sodass solche Übereinstimmungen zwischen Objektinstanzen nicht zu Objektbeziehungstypen führen sollten. Für diese Zusammenhänge ist zu prüfen, welche inhaltlichen Zusammenhänge tatsächliche Objektbeziehungstypen darstellen. Dazu wird analysiert, in wie vielen Prozessinstanzen ein bestimmter inhaltlicher Zusammenhang zwischen zwei Objektinstanzen auftritt. Anhand eines Schwellwerts in Bezug zur Gesamtanzahl der Prozessinstanzen (*Prozessinstanz-Relevanz*) und der Anzahl von Objektinstanzen der beteiligten Objekttypen (*Objektinstanz-Relevanz*) wird bestimmt, ob aus dem Zusammenhang ein Objektbeziehungstyp zwischen diesen Objekttypen definiert wird. Treten Zusammenhänge zwischen Objekttypen in nur wenigen Prozessinstanzen auf, kann dies auf zufällige Zusammenhänge hinweisen. Jedoch könnte es dennoch ein Objektbeziehungstyp darstellen, der beispielsweise einen alternativen Pfad in einem Geschäftsprozess aufzeigt. Daher ist ein geeigneter Schwellwert zu definieren, ab dem eine Übereinstimmung als signifikant gilt. Bei der Objektinstanz-Relevanz wird die Anzahl der Objektinstanzen analysiert, in denen die Überschneidung zwischen den Instanzen der betreffenden Objekttypen vorkommt. Beispiel: Wenn 100 Bestellungen und 100 Rechnungen existieren, müssen mindestens 90 davon diese Überschneidung aufweisen, wenn der Schwellwert 90% ist.

Die identifizierten inhaltlichen Zusammenhänge, die als Objektbeziehungstypen definiert werden, sollen über die verschiedenen Prozessinstanzen bezüglich der Übernahme-

Attribute- bzw. Werte sowie Regeln widerspruchsfrei sein (*Attributs-Prüfung*). Zunächst wird in der Attributs-Prüfung die Anzahl der unterschiedlichen Überschneidungen verglichen. Tritt beispielsweise eine Überschneidung zwischen zwei Objekttypen 80-mal auf und eine andere Überschneidung zwischen den gleichen Objekttypen nur 3-mal auf, könnte die zweite Überschneidung zufällig gefunden worden sein und widersprüchlich sein. Zusätzlich werden die Überschneidungen eines Objektbeziehungstyps auf Widersprüche geprüft, indem analysiert wird, ob die einzelnen Überschneidungen disjunkt in den Attributen sind. Sind die Überschneidungen nicht disjunkt, werden die Objektinstanzen, die diese Überschneidungen haben, geprüft. Entweder müssen es immer die gleichen Wertebelegungen sein oder unterschiedliche Objektinstanzen betreffen. Ansonsten sollte nur eine der Überschneidungen behalten werden.

Abbildung 6-7 stellt dieses zweistufige Verfahren dar.



Legende: SW_x = Schwellwert für Relevanzkriterium x;

Abbildung 6-7: Definition der Objekttypen aus inhaltlichen Zusammenhängen

Aus den direkten einfachen Objektbeziehungstypen kann zunächst ein Objektmodell abgeleitet werden. Die Schritte zur Ableitung eines Objektmodells zum Geschäftsprozesstyp durch die Definition der Objekttypen, deren Attributstrukturen sowie der Objektbeziehungstypen sind in Abbildung 6-8 dargestellt.

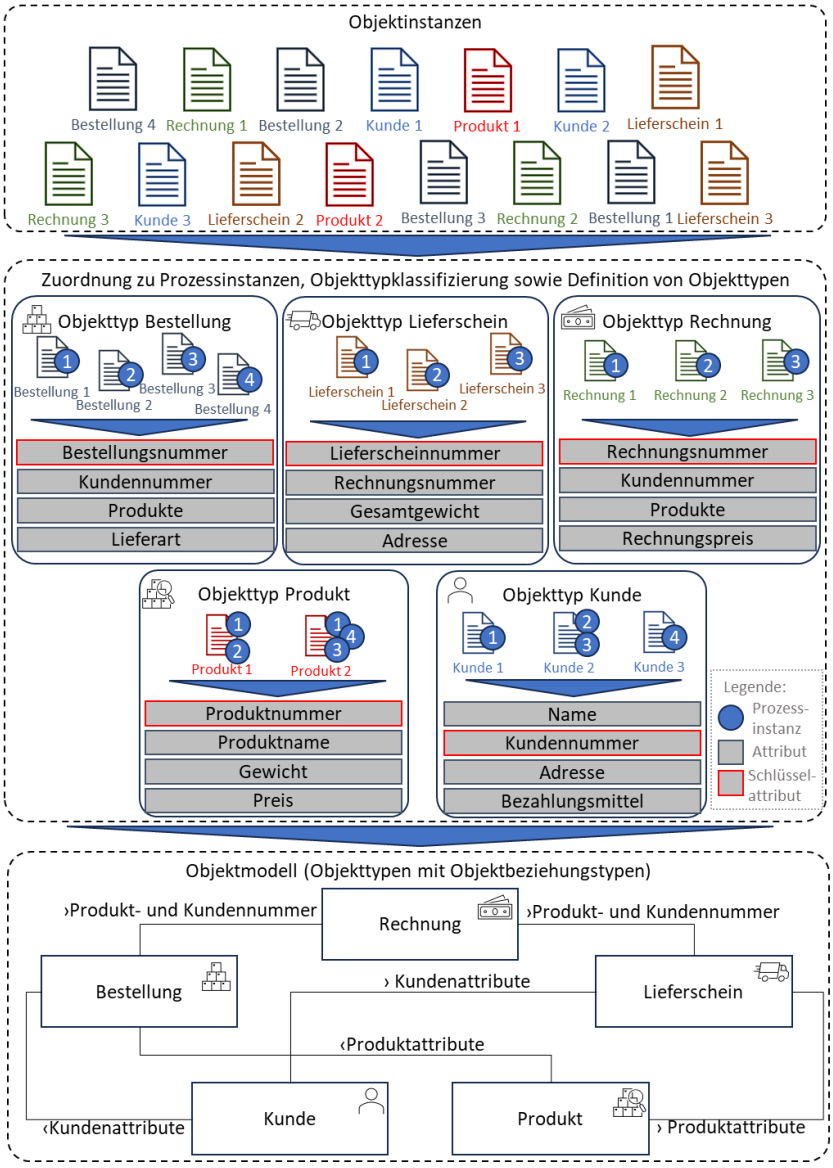


Abbildung 6-8: Generierung des Objektmodells

Zusätzlich zu dem paarweisen Vergleich können basierend auf den gesamten Objektinstanzen einer Prozessinstanz über verschiedene Prozessinstanzen hinweg Korrelationen von Werten identifiziert werden. Dadurch können direkte mehrfache Objektbeziehungstypen (und auch indirekte Objektbeziehungstypen) aufgedeckt werden.

Möglichkeit 2: Objektbeziehungstypen durch zeitliche Zusammenhänge

Basierend auf dem zeitlichen Auftreten der einzelnen Objektinstanzen kann zwischen zwei direkt aufeinander folgenden Objektinstanzen ein Objektbeziehungstyp bestehen. Aus diesem Objektbeziehungstyp kann nachfolgend eine Aktivitätsinstanz abgeleitet werden, wobei die zuerst auftretende Objektinstanz der Input und die nachfolgende Objektinstanz der Output dieser Aktivitätsinstanz darstellt. Daraus resultierend erhält diese Aktivitätsinstanz zwei Zeitpunkte, die aus den Zeitpunkten der Input-Objektinstanz und der Output-Objektinstanz bestehen.

Für die Objektinstanzen sind daher zunächst diese Zeitpunkte zu identifizieren und zu bewerten. Eine Objektinstanz ist immer zu einem bestimmten Zeitpunkt t definiert. Dieser Zeitpunkt kann nicht vor der Erzeugung der Objektinstanz liegen. Jedoch könnte der Zeitpunkt Jahre nach der letzten Veränderung oder dem Verbrauch der Objektinstanz liegen. Dadurch würde eine Reihenfolge der Objektinstanzen basierend auf diesen Zeitpunkten nicht die Änderung durch Aktivitäten aufzeigen. Daher sind die Zeitpunkte der Objektinstanzen zu beachten, die deren Erzeugung, Veränderung oder Verbrauch repräsentieren oder zwischen diesen Zustandsübergängen liegen.

Der Zeitpunkt t für die Objektinstanz ergibt sich aus den Metadaten der Datei, von der die Objektinstanz beschrieben ist. Weitere mögliche relevante Zeitpunkte können vorhandene Daten-Attribute sein. Treten mehrere Daten auf, ist basierend auf verschiedenen Aspekten zu bewerten, welches Datum als repräsentativer Zeitpunkt für die Objektinstanz verwendet werden soll. Beispiele für Daten bezüglich einer Objektinstanz sind:

- Datum aus Metadaten einer Objektinstanz:
    - Erstellungsdatum aus Metadaten: Zeitpunkt der Erzeugung der Objektinstanz.
    - Bearbeitungs-/Änderungsdatum aus Metadaten: Zeitpunkt der letzten Aktualisierung oder Modifikation der Objektinstanz.
    - Speicherdatum aus Metadaten: Zeitpunkt der letzten Speicherung der Objektinstanz.
- Daten-Attribute einer Objektinstanz:
    - Erstellungs-/Rechnungsdatum: Datum, das in der Objektinstanz die Erstellung der Objektinstanz beschreibt.
    - Unterschriftendatum: Zeitpunkt einer manuellen Aktivität (beispielsweise Bestätigung, dass ein Dokument gesichtet wurde und Genehmigung der damit verbundenen Inhalte) bzgl. einer Objektinstanz.
    - Datum im Fließtext: spezifische Zeitpunkte wie beispielsweise ein angekündigtes Lieferdatum oder einer Bezahlungsfrist.

Um bei einem Auftreten von mehreren Daten das Datum auszuwählen, welches das Auftreten in einer Prozessinstanz bezüglich der anderen Objektinstanzen repräsentativ beschreibt, werden die Daten nach den *intrinsischen*[4] und *kontextuellen*[5] Datenqualitätsdimensionen [WaSt96] in Abschnitt 3.1 bewertet.

Tabelle 6-3: Bewertungsaspekte für die Repräsentativität eines Datums

Bewertungsaspekt	Beschreibung
kontextuelle DQ bezüglich des Datentyps	Erstellungs- und/oder Unterschriftendatum: Beschreiben den Zeitpunkt des Auftretens einer Objektinstanz repräsentativ, da sie sich beispielsweise nicht auf das Datum der Erstellung einer Vorlage, der Nachbearbeitung oder Speicherung einer Objektinstanz beziehen, das ggf. außerhalb der Ausführungszeit einer Prozessinstanz liegt.
	Datum im Fließtext: Dieses kann sich auch auf andere Aktivitäten oder Objekte beziehen und ist daher nicht repräsentativ für Aktivitäten, die in Verbindung mit der Objektinstanz stehen.
	Datum aus Metadaten: Die Relevanz von Metadaten kann sehr gering sein, da beispielsweise ein Meta-Erstellungsdatum einer Datei auch dem Erstellen einer Vorlage entsprechen könnte, das somit keinen Bezug zu einem zeitlichen Auftreten in einer Prozessinstanz hat. Liegt jedoch ein Meta-Datum *innerhalb* der Zeitpunkte der anderen Objektinstanzen einer Prozessinstanz, kann es sich um ein repräsentatives Datum handeln. Auch das Meta-Bearbeitungs- oder Speicherdatum kann durch ein nachträgliches Bearbeiten oder Speichern ggf. nicht repräsentativ für eine Objektinstanz sein.
kontextuelle DQ bezüglich eines Objekttyps	Unterschiedliche Objekttypen haben unterschiedliche relevante Daten-Attribute. Beispielsweise kann in Lieferdokumenten das Lieferdatum als Zeitpunkt verwendet werden, da es die Durchführung der Lieferung widerspiegelt. Für Rechnungen ist das Unterschriftsdatum oder das Rechnungsdatum entscheidend, während für Vertragsdokumente das Datum der letzten Änderung von Bedeutung ist.
intrinsische DQ bezüglich einer Prozessinstanz	Die Glaubwürdigkeit kann zum einen anhand der festgelegten Zeitpunkte anderer Objektinstanzen in einer Prozessinstanz festgestellt werden: Liegt das Datum einer Objektinstanz deutlich außerhalb der anderen auftretenden Daten, sinkt dessen Glaubwürdigkeit. Zum anderen sinkt die Glaubwürdigkeit eines Datums, wenn es deutlich in der Vergangenheit bzw. Zukunft liegt.

[4] intrinsische DQ: Genauigkeit, Objektivität, Glaubwürdigkeit und Ansehen

[5] kontextuelle DQ: Relevanz, Mehrwert, Aktualität, Vollständigkeit und angemessene Menge an Daten

Bewertungsaspekt	Beschreibung
intrinsische DQ bezüglich eines Objekttyps	Die Glaubwürdigkeit kann anhand der festgelegten Zeitpunkte anderer Objektinstanzen des gleichen Objekttyps (in Vergleich zu den vorgehenden bzw. nachfolgenden Objektinstanzen) bewertet werden: Ist das Datum einer Objektinstanz im Vergleich zu den anderen Objektinstanzen des gleichen Objekttyps deutlich näher oder entfernter zu den zeitlich vorgehenden bzw. nachfolgenden Objektinstanzen, sinkt dessen Glaubwürdigkeit.

Nach der Festlegung der repräsentativen Zeitpunkte für alle Objektinstanzen werden die zeitlichen Zusammenhänge zwischen direkt aufeinanderfolgenden Objektinstanzen innerhalb einer Prozessinstanz analysiert. Diese Zusammenhänge werden durch die jeweiligen Input- und Output-Objektinstanzen sowie deren Zeitpunkte beschrieben. Daraus werden die Objektbeziehungstypen abgeleitet, die durch Input- und Output-Objekttypen definiert sind. Objektinstanzen, die in Abschnitt 6.1.3 als Versionen identifiziert wurden, wird eine Versionsnummer basierend auf ihrem zeitlichen Auftreten zugewiesen.

6.1.5 Aktivitätsinstanzen, -typen und Regeln

Aus den gefundenen Objektbeziehungstypen sind Aktivitätsinstanzen abzuleiten, um dadurch deren Aktivitätstypen zu definieren. Der Typ der Aktivitätsinstanzen ist über die gleichen Objekttypen, die im Zusammenhang stehen sowie ggf. den darauf bezogenen Manipulationen definiert. Aus den Objektbeziehungstypen, die aus den zeitlichen Zusammenhängen abgeleitet wurden, können unmittelbar Aktivitätstypen definiert werden, da bei diesen bereits eindeutig ist, was Input und Output ist und für diese keine Manipulationen definiert sind. Für die Objektbeziehungstypen, die aus den inhaltlichen Zusammenhängen abgeleitet wurden, können nicht unmittelbar Aktivitätstypen definiert werden, da die Richtung des Zusammenhangs nicht definiert ist.

Um die Richtung des Zusammenhangs zu bestimmten, soll zunächst für jede betreffende Objektinstanz eines Objektbeziehungstyps eine Bewertung stattfinden, wie wahrscheinlich diese Objektinstanz Input oder Output für die daraus abzuleitende Aktivitätsinstanz ist. Das heißt, es ist zu bewerten, welche Objektinstanz einer Aktivitätsinstanz die Attribute bzw. Werte bereitstellt und welche Objektinstanz von der Aktivitätsinstanz erzeugt wird und die Attribute bzw. Werte übernimmt. Diese Bewertung ist durch unterschiedliche Aspekte beeinflusst, die in Tabelle 6-4 beschrieben sind.

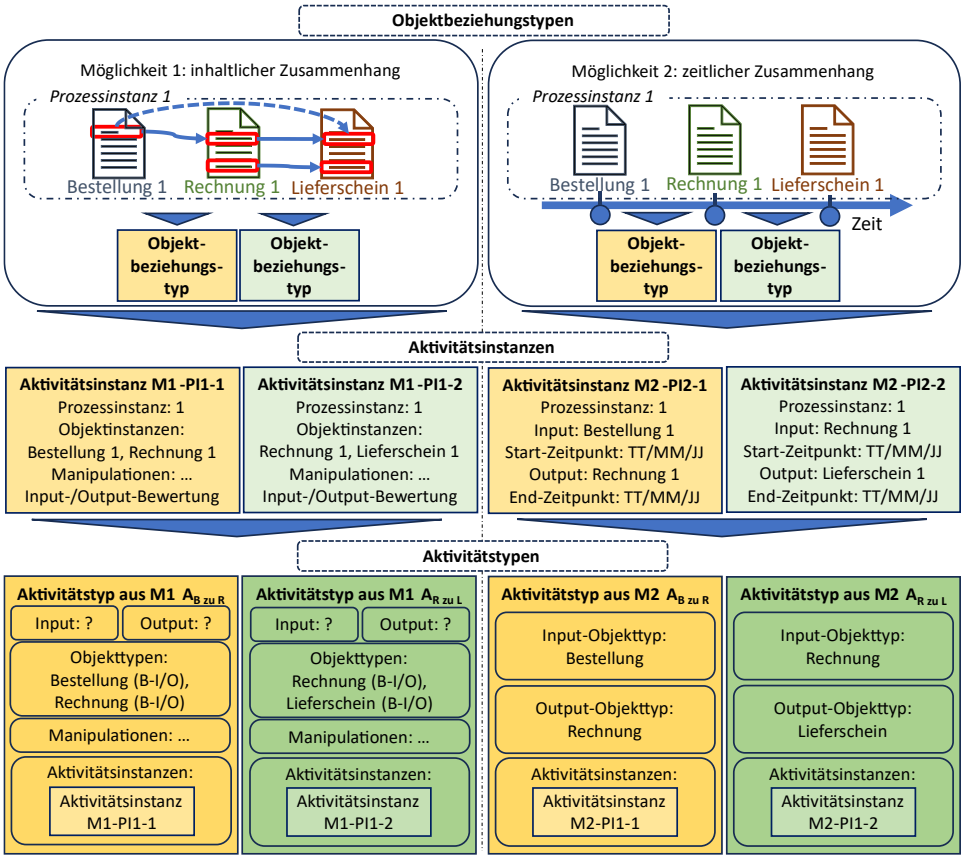
Tabelle 6-4: Bewertungsaspekte für die Richtung des Zusammenhangs

Bewertungsaspekt	Beschreibung
Ist in den Objektinstanzen ein Datum enthalten?	Tritt bei beiden Instanzen ein Datum auf, kann anhand des Datums die Reihenfolge vermutet werden.
Wurden die Objektinstanzen als prozessinstanzunabhängig klassifiziert?	Prozessinstanzunabhängige Objekte sind häufig zur Datenbereitstellung gedacht und sind in diesem Fall Input für eine Aktivität. Dies wird widerlegt, wenn sich innerhalb einer Prozessinstanz die Werte der Objektinstanzen ändern, da diese dann als Output einer Aktivität gewertet werden.
Sind die Objektinstanzen bei weiteren Zusammenhängen beteiligt?	Ist eine Objektinstanz sowohl Input als auch Output von unterschiedlichen Aktivitätsinstanz, besteht eine sequenzielle Ordnungsbeziehung zwischen diesen Aktivitäten: Die Aktivität, die die gemeinsame Objektinstanz als Output hat, muss zuerst stattfinden, damit diese Objektinstanz Input für eine andere Aktivität sein kann. Daher sind anhand der Objektinstanzen innerhalb der jeweiligen Prozessinstanz Bewertungen vorzunehmen.
Ist es ein mehrfacher Zusammenhang?	Werden mehrfache Zusammenhänge aufgedeckt, beispielsweise eine Addition von Preisen zur Berechnung des Gesamtpreises, ist die Reihenfolge dadurch bestimmt, wenn die Berechnung nur in eine Richtung eindeutig ist.

Anschließend sind die Aktivitätsinstanzen zu Aktivitätstypen zu generalisieren und für die einzelnen Aktivitätsinstanzen der jeweilige Aktivitätstyp festzuhalten. Es wird für jede Aktivitätsinstanz geprüft, ob bereits ein Aktivitätstyp definiert wurde, der dieselben Objekttypen betrifft und aus derselben Möglichkeit (zeitlicher oder inhaltlicher Zusammenhang) abgeleitet wurde. Für die Aktivitätsinstanzen der inhaltlichen Zusammenhänge wird zusätzlich geprüft, ob die Manipulationen übereinstimmen. Für die übereinstimmenden Aktivitätsinstanzen werden mit den Bewertungsaspekten aus Tabelle 6-4 deren Bewertung, ob diese Input bzw. Output sind, berechnet. Ist zwar ein Aktivitätstyp vorhanden, der dieselben Objekttypen betrifft, jedoch andere Manipulationsanweisungen enthält, wird zunächst ein neuer Aktivitätstyp erzeugt.

Nach der Betrachtung aller Aktivitätsinstanzen sind die abgeleiteten Aktivitätstypen weiter zu spezifizieren, da bisher nur für die entsprechenden Aktivitätsinstanzen die Bewertung der Objektinstanzen für Input bzw. Output berechnet wurden. Somit wurden die Input- und Output-Objekttypen für diese Aktivitätstypen noch nicht eindeutig festgelegt. Dazu werden die Bewertungen der einzelnen Aktivitätsinstanzen eines Aktivitäts-

typs analysiert, und basierend auf diesen die betreffenden Objekttypen auf Typ-Ebene der Aktivität als Input bzw. Output definiert. Das Vorgehen zur Ableitung der Aktivitätstypen wird in Abbildung 6-9 dargestellt.



Legende: PI = Prozessinstanz; M1 = Möglichkeit 1; M2 = Möglichkeit 2; B-I/O = Bewertung für Input/Output

Abbildung 6-9: Definition der Aktivitätstypen

Basierend auf diesem Schritt werden die Aktivitätstypen aus zeitlichen und inhaltlichen Zusammenhängen generiert. Wie in Abbildung 6-6 in Möglichkeit 1 dargestellt, können jedoch insbesondere bei den inhaltlichen Zusammenhängen mehrere Aktivitätstypen generiert worden sein, beispielsweise wenn Attribut-Wert-Paare wie Informationen zu Kunden oder Produkten über mehrere Objektinstanzen hinweg identisch sind. Daher sind die Aktivitätstypen auf Redundanzen und mögliche Zusammenführungen zu prüfen. Als Ergebnis soll eine Menge an Aktivitätstypen definiert werden, bei der alle Objekttypen beteiligt sind, sowie alle Attribute der Überschneidungen abdeckt werden. Zusätzlich sollen durch die Aktivitätsinstanzen dieser Aktivitätstypen auch alle gegebenen Zustände

der Objektinstanzen erreichbar sein. Zur Definition der relevanten Aktivitätstypenmenge erfolgt zunächst eine systematische Bewertung der abgeleiteten Aktivitätstypen, bei der anhand spezifischer Eigenschaften ihre Relevanz für den Geschäftsprozess bestimmt wird. Diese Bewertung basiert auf mehreren definierten Kriterien, für die zur flexiblen Anpassung an unterschiedliche Prozesscharakteristika individuelle Gewichtungsfaktoren festgelegt werden. Diese Bewertungskriterien sind:

1. Strukturelle Positionierung:
    - Aktivitätstypen die früh in einem Geschäftsprozess auftreten können, sollen eine positive Gewichtung erhalten, während späte Positionen mit einem Reduktionsfaktor versehen werden sollen. Diese Differenzierung basiert auf der Annahme, dass die beteiligten Objekttypen so früh wie möglich erzeugt, bearbeitet, gelesen oder verbraucht werden sollen.
2. Integration der prozessinstanzunabhängigen Objekttypen:
    - Die Verarbeitung von prozessinstanzunabhängigen Objekttypen wird durch zwei korrelierende Faktoren gesteuert: Ein Basisgewicht wird durch einen Positionsfaktor verstärkt. Zum einen können Informationen dieser prozessinstanzunabhängigen Objekttypen, unabhängig des Zustandes einer Prozessinstanz verwendet werden. Somit soll beispielsweise das Einbinden der Informationen aus Kundendaten so früh wie möglich in einem Geschäftsprozess stattfinden. Daher erhöht das Basisgewicht die Bewertung der Aktivitätstypen die prozessinstanzunabhängigen Objekttypen einbeziehen und zusätzlich verstärkt der Positionsfaktor die Gewichtung des frühen Einbeziehens dieser Aktivitätstypen.
3. Eigenschaften des Zusammenhangs:
    - Die Bewertung der Eigenschaften des Zusammenhangs kann durch drei Aspekte beeinflusst werden:
        - Häufigkeit der Aktivitätsinstanzen des Aktivitätstyps bezüglich der zeitlichen Zusammenhänge.
        - Anzahl und Bedeutung der übertragenen Attribute: Die Bedeutung eines Attributs ist höher, je seltener es in den Regeln aller Aktivitätstypen auftritt.
        - Zeitliche Nähe zwischen Input- und Output-Objektinstanzen: Dafür soll für alle Aktivitätsinstanzen die zeitliche Differenz der beteiligten Objektinstanzen ins Verhältnis zur maximalen Differenz der Prozessinstanz gesetzt werden.

Für die Positions-Bewertungen kann ein Graph generiert werden, dessen Knoten die Objekttypen sind und die Kanten die Aktivitätstypen abbilden. Außerdem können die Kanten die Gewichtung der Attribute und Anzahl der Aktivitätsinstanzen widerspiegeln.

Weitere Eigenschaften können in der Bewertung der Aktivitätstypen berücksichtigt werden: Beispielsweise ob zwei Aktivitätstypen, die die gleichen Objekttypen betreffen, aber verschiedene Objektinstanzen einer Prozessinstanz, höher bewertet werden sollen. Dabei sollte jedoch berücksichtigt werden, dass sowohl Input- als auch Output-Objektinstanzen der Aktivitätstypen unterschiedlich sind, wenn diese als Versionen erkannt wurden. Sind die Objektinstanzen des gleichen Objekttyps, keine Versionen (eine Objektinstanz zu unterschiedlichen Zeitpunkten), sondern unterschiedliche Objektinstanzen und deren bestimmte Zeitpunkte sowie die Regeln der Aktivitätstypen widersprechen keiner Zusammenführung, so könnte es sich um einen Aktivitätstyp handeln der mehrfache Objektinstanzen eines Objekttyps als Input bzw. Output benötigt, bzw. erzeugt. Zusätzlich könnten Aktivitätstypen höher oder niedriger bewertet werden, wenn deren Input- und Output-Objekttyp identisch sind (Schleifen im Geschäftsprozess) oder eine Unabhängigkeit der Objekttypen vermutet werden kann. Dies kann beispielsweise zu einer Zusammenführung von Aktivitätstypen oder für eine nebenläufige Ordnungsbeziehung von Aktivitätstypen führen. Ist dies beispielsweise zur Verkürzung der Durchlaufzeiten gewünscht, könnte eine höhere Gewichtung zu einer höheren Bewertung führen. Ist jedoch bekannt, dass sequenzielle Ordnungsbeziehungen überwiegend vorliegen, sollten demnach diese Eigenschaften zu einer Verringerung der Gesamtbewertung (durch einen negativen Gewichtungsfaktor) oder einer geringeren Berücksichtigung in der Gesamtbewertung (durch einen Gewichtungsfaktor zwischen 0 und 1) führen.

Anhand der Bewertungskriterien und deren Gewichtungsfaktoren wird für jeden Aktivitätstyp ein Wert berechnet. Anhand dieses Wertes wird die Relevanz des Aktivitätstyps bestimmt. Da eine Bewertung durch negative Gewichtungsfaktoren auch reduziert werden kann, sollte auch eine Mindestbewertung angegeben werden können. Dadurch können beispielsweise Aktivitätstypen, die negative Bewertungen haben, für die Weiterverwendung ausgeschlossen werden. Die Gewichtungsfaktoren beeinflussen somit die Auswahl der Aktivitätstypen. Damit diese Methode an spezifische Prozesscharakteristika und Analyseanforderungen angepasst werden kann, sollen diese konfigurierbar sein (siehe Abschnitt 7.2).

Basierend auf der Bewertung der Aktivitätstypen sollen anschließend nur so viele Aktivitätstypen wie nötig ausgewählt werden (siehe Anforderung an das generierte Modell in Abschnitt 5.2). Daher werden die Aktivitätstypen absteigend nach ihrer Bewertung sortiert.

Die Auswahl der relevanten Aktivitätstypen erfolgt in drei aufeinanderfolgenden Phasen, die verschiedene Aspekte berücksichtigen. Dabei werden in jeder Phase alle noch nicht als relevant gekennzeichneten Aktivitätstypen erneut geprüft.

1. In der ersten Phase wird geprüft, ob der Aktivitätstyp prozessinstanzunabhängiger Objekttypen berücksichtigt. Nach erstmaliger Berücksichtigung eines prozessinstanzunabhängigen Objekttyps werden weitere Aktivitätstypen mit diesem Input für spätere Phasen zurückgestellt. Da die Aktivitätstypen, bei denen prozessinstanzunabhängiger Objekttypen beteiligt sind, höher bewertet werden, verfälscht dies die Relevanz in Bezug zu den anderen Aktivitätstypen.
   Außerdem werden die Aktivitätstypen als relevant gekennzeichnet, wenn neue Objekttypen beteiligt sind oder neue Attribute für einen Objekttyp berücksichtigt werden.
2. In der zweiten Phase werden die Aktivitätstypen, bei denen prozessinstanzunabhängige Objekttypen beteiligt sind, hinzugefügt, die neue Attribute betreffen.
3. In der dritten Phase werden die Aktivitätstypen hinzugefügt, deren Aktivitätsinstanzen neue Objektinstanzen als Input beziehungsweise Output haben. Dabei wird geprüft, falls es eine Objektinstanz ist die erstmalig als Input (Output) in einer Aktivitätsinstanz auftritt, ob es sich dabei zeitlich um die letzte (erste) Objektinstanz einer Prozessinstanz handelt. In diesem Fall sollte die Objektinstanz kein Input (Output) sein und der Aktivitätstyp wird daher dennoch nicht als relevant definiert.

Durch diese Bewertung und Auswahl der Aktivitätstypen, soll gewährleistet werden, dass die Aktivitätstypen alle Objekttypen, Attribut-Überschneidungen sowie Objektinstanzen abdecken. Dieses Verfahren ist in Abbildung 6-10 dargestellt.

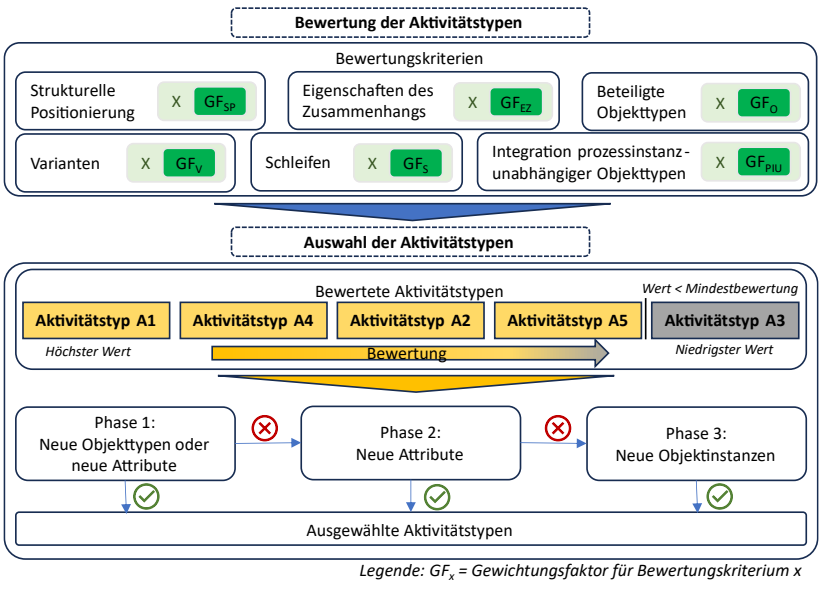


Abbildung 6-10: Bewertung und Auswahl der Aktivitätstypen

Nach dieser Auswahl sollten Aktivitätstypen, die über alle beteiligten Objekttypen hinweg identisch sind (siehe Abschnitt 6.1.5.1) sowie Aktivitätstypen, die mindestens einen gemeinsamen Objekttyp haben (siehe Abschnitt 6.1.5.2) auf eine Zusammenführung geprüft werden. Bei der Zusammenführung von Aktivitätstypen sind die Prozessinstanzen, in denen die Aktivitätsinstanzen der betrachteten Aktivitätstypen auftreten, zu analysieren. Treten diese beispielsweise nur in unterschiedlichen Prozessinstanzen auf, so stellen die Aktivitätstypen Alternativen in einem Geschäftsprozess dar. Treten diese beispielsweise nur in denselben Prozessinstanzen auf, so könnten die Aktivitätstypen gegebenenfalls nebenläufig oder sogar zusammen ausgeführt werden. Daher sind die Fälle anhand der Aktivitätsinstanzen der Aktivitätstypen bezüglich der Prozessinstanzen, in denen die Aktivitätsinstanzen auftreten, zu betrachten. Diese Fälle sind in Abbildung 6-11 dargestellt.

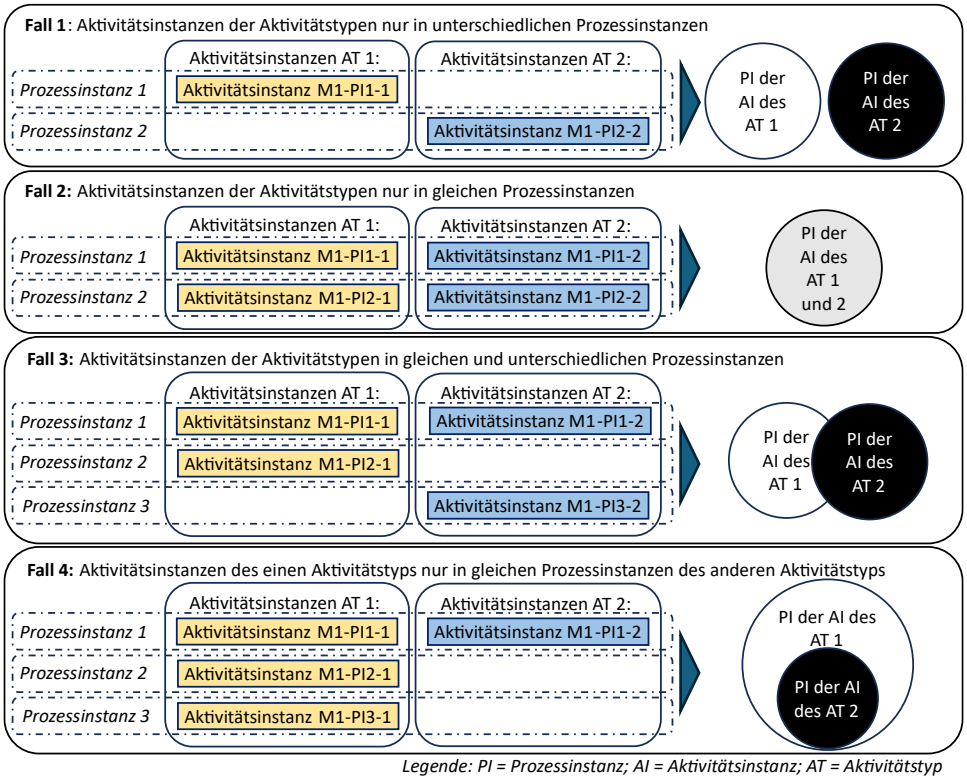


Abbildung 6-11: Betrachtungsfälle anhand der Prozessinstanzen

6.1.5.1 Aktivitätstypen mit denselben Input- und Output-Objekttypen

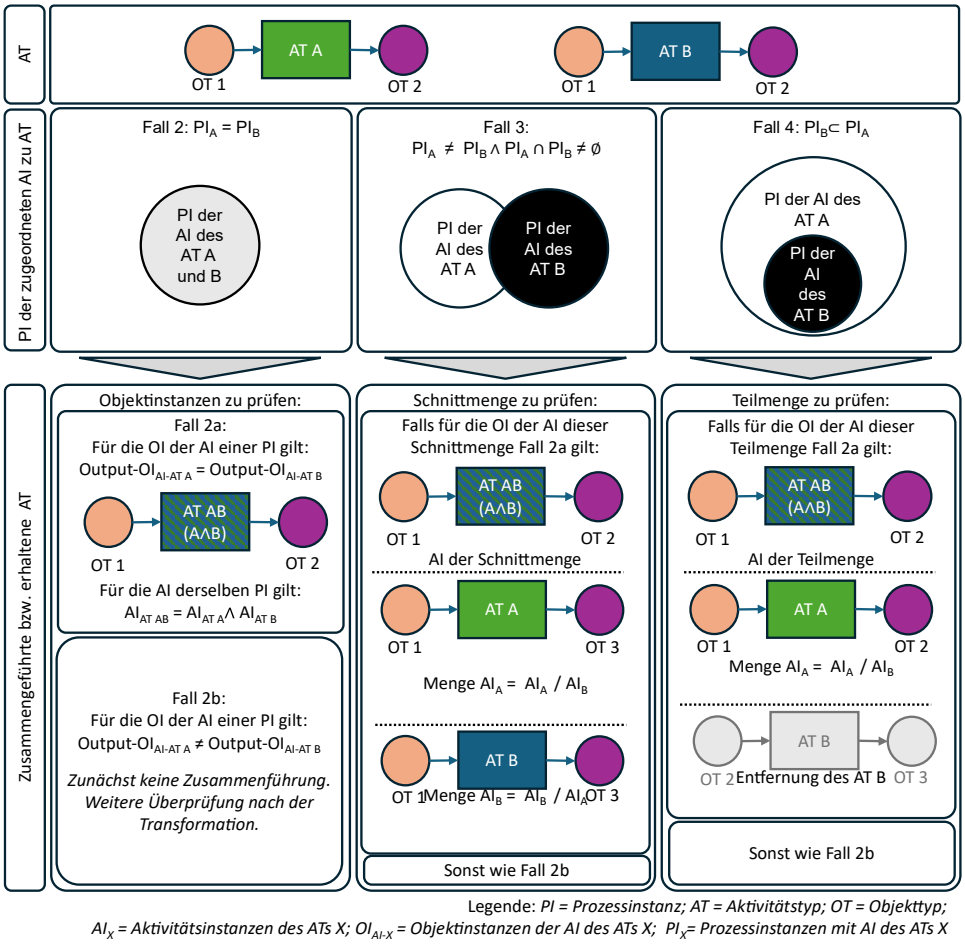
Werden Aktivitätstypen geprüft, die dieselben Input- und Output-Objekttypen betreffen, können dies nur Aktivitätstypen mit unterschiedlichen Manipulationsregeln sein, da ansonsten kein neuer Aktivitätstyp erzeugt worden wäre. Dies wird anhand der vier Fälle auf eine mögliche Zusammenführung überprüft:

1. Fall: *Aktivitätsinstanzen der Aktivitätstypen nur in unterschiedlichen Prozessinstanzen*
   Keine Zusammenführung der Aktivitätstypen, da zwischen den Aktivitätstypen eine alternative Ordnungsbeziehung vorliegt.
2. Fall: *Aktivitätsinstanzen der Aktivitätstypen nur in denselben Prozessinstanzen*
   Es werden die Objektinstanzen der entsprechenden Aktivitätsinstanzen geprüft:
   a. Wenn die Aktivitätsinstanzen einer Prozessinstanz der Aktivitätstypen dieselben Input- und Output-Objektinstanzen zum selben Zeitpunkt t betreffen, werden die Aktivitätstypen zu einem Aktivitätstyp zusammengeführt, indem deren Manipulationen zusammengeführt werden. Nach der Zusammenführung werden die ursprünglichen Aktivitätstypen entfernt und die Aktivitätsinstanzen dem neuen Aktivitätstyp zugeordnet.
   b. Wenn die Aktivitätsinstanzen einer Prozessinstanz der Aktivitätstypen nicht dieselben Input- und Output-Objektinstanzen zum selben Zeitpunkt t betreffen, werden die Aktivitätstypen dieser Aktivitätsinstanzen weiter überprüft, da diese unterschiedliche Aktivitätstypen an unterschiedlichen Zeitpunkten in einer Prozessinstanzen darstellen könnten. Beispielsweise könnten die Aktivitätstypen *Rechnungserstellung* und die *Rechnungsverbesserung* über dieselben Input- und Output-Objekttypen definiert sein, die Aktivitätsinstanzen davon jeweils Objektinstanzen zu unterschiedlichen Zeitpunkten betreffen (die *erstellte* Rechnung zum Zeitpunkt t sowie die *verbesserte* Rechnung zum Zeitpunkt $t + x$).
3. Fall: *Aktivitätsinstanzen der Aktivitätstypen teilweise in denselben Prozessinstanzen*
   Es wird die Schnittmenge der gemeinsamen Prozessinstanzen geprüft:
   a. Trifft auf die Schnittmenge die Bedingung von Fall 2a zu, können die Aktivitätstypen für diese Schnittmenge entsprechend zusammengeführt werden. Die anderen Aktivitätsinstanzen sind weiterhin Instanzen der ursprünglichen Aktivitätstypen.
   b. Trifft auf die Schnittmenge die Bedingung von Fall 2b zu, werden die Aktivitätstypen dieser Aktivitätsinstanzen weiter überprüft.

4. Fall: *Aktivitätsinstanzen des einen Aktivitätstyps nur in denselben Prozessinstanzen des anderen Aktivitätstyps*
   Es wird die Teilmenge der gemeinsamen Prozessinstanzen geprüft:
   a. Trifft auf die Teilmenge die Bedingung von Fall 2a zu, können für die Teilmenge die Aktivitätstypen zusammengeführt werden. Die Aktivitätsinstanzen außerhalb der Teilmenge sind weiterhin vom Aktivitätstyp der Obermenge.
   b. Trifft auf die Teilmenge die Bedingung von Fall 2b zu, werden die Aktivitätstypen dieser Aktivitätsinstanzen weiter überprüft.

Die weitere Überprüfung für Fall 2b (und Fall 3 und 4, falls für die Schnittmenge bzw. Teilmenge, Fall 2b zutrifft) erfolgt in der Transformation (siehe Abschnitt 6.2), da hier die Ordnungsbeziehungen zwischen den Aktivitätstypen berücksichtigt werden müssen. Ein Beispiel: Die Objekttypen *Bestellung* und *Rechnung* sind in zwei Aktivitätstypen als Input- bzw. Output-Objekttypen definiert. In einer Prozessinstanz gibt es die Bestellung *b1* und die Rechnungen *r1_1* und *r1_2*. Der Aktivitätstyp *Rechnung erstellen* erzeugt *r1_1*, der Aktivitätstyp *Rechnung verbessern* erzeugt *r1_2* (beide basierend auf *b1*). Obwohl beide Aktivitätstypen dieselben Input- und Output-Objekttypen haben, stellen sie unterschiedliche Aktivitäten dar. Daher muss abhängig vom Geschäftsprozesstyp geprüft werden, ob mit den entdeckten Ordnungsbeziehungen beide Aktivitätstypen innerhalb einer Prozessinstanz ausführbar sind und ob diese zusammengeführt werden sollten.

Abbildung 6-12 stellt die Zusammenführung der Aktivitätstypen dar. Da in Fall 1 keine Zusammenführung oder Entfernung der Aktivitätstypen erfolgt, ist Fall 1 in der Abbildung nicht dargestellt.



Legende: *PI = Prozessinstanz; AT = Aktivitätstyp; OT = Objekttyp;*
AI_X = Aktivitätsinstanzen des ATs X; $OI_{AI\text{-}X}$ = Objektinstanzen der AI des ATs X; PI_X= Prozessinstanzen mit AI des ATs X

Abbildung 6-12: Zusammenführung von Aktivitätstypen mit denselben Input- und Output-Objekttypen

6.1.5.2 Aktivitätstypen mit gemeinsamen Input- <u>oder</u> Output-Objekttypen

Neben Aktivitätstypen mit denselben Input- und Output-Objekttypen sind auch Aktivitätstypen zu analysieren, die mindestens einen Input- oder Output-Objekttyp gemeinsam haben. Zum einen werden aus den direkten Zusammenhängen, wie einfachen Attributübernahmen, Aktivitätstypen mit jeweils genau einem Input- und Output-Objekttyp generiert. Gegebenenfalls können zwei oder mehr dieser einfachen direkten Zusammenhänge (siehe Abschnitt 6.1.4) durch einen Aktivitätstyp mit mehreren Inputs oder Outputs zusammen abgebildet werden. Zum anderen können Aktivitätstypen mit mehreren Inputs und Outputs entstehen, wenn mehrfache Zusammenhänge (siehe Abschnitt 6.1.4) wie die Addition von Attributwerten verschiedener Objekttypen erkannt werden. Diese Aktivi-

tätstypen können Überschneidungen in den Objekttypen mit anderen Aktivitätstypen haben, ohne dass die Objekttypen vollständig übereinstimmen. Daher erfüllen diese nicht die Anforderung aus Abschnitt 6.1.5.1. Als Ergebnis dieser Prüfung können die Aktivitätstypen entweder zusammengeführt, beibehalten oder entfernt werden.

Da bei Aktivitätstypen mit mindestens einem gemeinsamen Input- oder Output-Objekttyp sowohl gleiche als auch unterschiedliche Manipulationen auftreten können, sind diese nach den Manipulationen differenziert zu überprüfen. Anhand der Übereinstimmungen in den Manipulationen zwischen den Aktivitätstypen, die mindestens einen Input- oder Output-Objekttypen gemeinsam haben, können vier Fälle unterschieden werden:

- Zwei Aktivitätstypen haben keine Überschneidungen in den Manipulationen (siehe Abschnitt 6.1.5.2.1): Die Aktivitätstypen haben in den Manipulationen keine Überschneidungen in den Attributen oder Attributwerten.
- Zwei Aktivitätstypen haben nur redundante Manipulationen (siehe Abschnitt 6.1.5.2.2): Redundante Manipulationen liegen vor, wenn zwei Aktivitätstypen dieselben Attribute und/oder Attributwerte auf die gleiche Weise verarbeiten. Beispielsweise tritt eine redundante Manipulation auf, wenn der Kundenname eines Lieferscheins sowohl aus einem Attribut-Wert-Paar der Bestellung als auch der Rechnung übergeben werden kann (siehe gestrichelte Linie in Abbildung 6-6) und keine weiteren Manipulationen zwischen Bestellung-Lieferschein oder Rechnung-Lieferschein entdeckt wurden.
- Zwei Aktivitätstypen haben eine gemeinsame nicht-leere Schnittmenge in den Manipulationen (siehe Abschnitt 6.1.5.2.3): Die Aktivitätstypen haben sowohl redundante Manipulationen als auch unterschiedliche Manipulationen. Beispielsweise wird die Kundennummer in Bestellung-Lieferschein und Rechnung-Lieferschein übergeben. Zusätzlich werden auch die aufgelisteten Produkte im Lieferschein aus der Bestellung und die Rechnungsadresse aus der Rechnung übergeben.
- Die Manipulationen des einen Aktivitätstyps sind eine echte Teilmenge der Manipulationen des anderen Aktivitätstyps (siehe Abschnitt 6.1.5.2.4): Die zwei Aktivitätstypen haben redundante Manipulationen. Zusätzlich hat ein Aktivitätstyp noch weitere Manipulationen, die nicht im anderen Aktivitätstyp vorkommen.

Durch diese Prüfung sollen Aktivitätstypen entfernt werden, die nur redundante Manipulationen durchführen. Das heißt, wenn beispielsweise Attribut-Wert-Paare durch verschiedene Aktivitätstypen übernommen werden können, soll nur ein Aktivitätstyp definiert werden, ohne die Vollständigkeit der übernommenen Attribute und/oder Attributwerte zu vernachlässigen. Außerdem werden Aktivitätstypen auf eine Zusammenführung geprüft. Beispielsweise sollten zwei Aktivitätstypen mit gleichem Output aber unterschiedlichen Input zusammengeführt werden, wenn diese Aktivitätstypen

dieselbe Output-Objektinstanzen erzeugen und diese Objektinstanzen somit in einer Aktivitätsinstanz mit den Objektinstanzen der zwei Input-Objekttypen erzeugt werden sollten.

Für jeden Fall der Manipulationen wird anhand der zugeordneten Aktivitätsinstanzen geprüft, ob die Prozessinstanzen übereinstimmen. Treten hierbei keine Überschneidungen auf, sind die Aktivitätstypen weder zusammenzuführen noch zu entfernen (Alternativen). Treten jedoch in Prozessinstanzen die Aktivitätsinstanzen beider Aktivitätstypen auf, könnten redundante Manipulationen vorliegen und es ist fallbezogen zu entscheiden, welcher Aktivitätstyp für einen Geschäftsprozess gewählt bzw. ob die Aktivitätstypen zusammengeführt werden sollte. In den Abschnitten 6.1.5.2.1- 6.1.5.2.4 ist dies anhand eines gemeinsamen Output-Objekttypens betrachtet. Dies ist ebenfalls auf Aktivitätstypen mit einem gemeinsamen Input-Objekttyp oder mit mehreren gemeinsamen Objekttypen übertragbar, wobei die Zusammenführung dieser Fälle in Abbildung 6-13 dargestellt ist und dementsprechend bei den Zusammenführungen von Aktivitätstypen mit gemeinsamen Output-Objekttypen in den Abbildungen zu ersetzen ist.

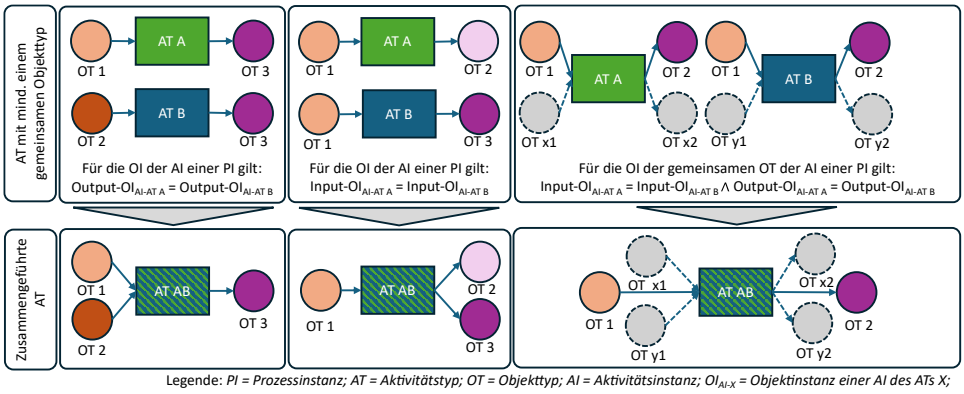


Abbildung 6-13: Zusammenführung von Aktivitätstypen

Wenn die Aktivitätsinstanzen der verschiedenen Aktivitätstypen nicht ausschließlich redundante Manipulationen haben und sich sowohl auf denselben Objekttypen als auch auf dieselben Objektinstanzen zum selben Zeitpunkt[6] beziehen, werden die Aktivitätstypen zusammengeführt. Das heißt, die Aktivitätstypen *A* und *B* werden zusammengeführt, wenn:

- die Aktivitätstypen *A* und *B* einen gemeinsamen Objekttyp haben und
- für ihre Aktivitätsinstanzen in denselben Prozessinstanzen gilt:

[6] Versionen von Objektinstanzen sind dieselbe Objektinstanz zu einem unterschiedlichen Zeitpunkt.

Objektinstanz des gemeinsamen Objekttyps einer Aktivitätsinstanz des Aktivitätstyps A ($OI_{AI\text{-}AT\ A}$)	=	*Objektinstanz des gemeinsamen Objekttyps einer Aktivitätsinstanz des Aktivitätstyps B* ($OI_{AI\text{-}AT\ B}$).

Gilt dies nicht, ist die Zusammenführung der Aktivitätstypen nicht unmittelbar möglich, auch wenn deren Aktivitätsinstanzen in denselben Prozessinstanzen auftreten. Eine Zusammenführung ist dann in Abhängigkeit der vorherigen bzw. nachfolgenden Objekte und Aktivitäten zu prüfen und wird daher erst in der Transformation (siehe Abschnitt 6.2) vorgenommen. Widerspricht beispielsweise ein zeitlicher Zusammenhang einer Zusammenführung sind die Aktivitätstypen beide zu erhalten, um diese nebenläufig bzw. sequenziell (ggf. nicht unmittelbar aufeinanderfolgend) ausführen zu können. Sonst könnte bei einer Zusammenführung beispielsweise eine Objektinstanz bereits von einem anderen Aktivitätstyp als Input verwendet werden, bevor sie erzeugt wurde. Wie in Abschnitt 6.1.5.1 sind die verschiedenen Fälle anhand der Prozessinstanz-Mengen zu prüfen. Dies wird im Folgenden beispielhaft für Aktivitätstypen mit einem gemeinsamen Output-Objekttyp gezeigt.

6.1.5.2.1 Fall 1: Aktivitätstypen ohne Überschneidungen in den Manipulationen

Haben zwei Aktivitätstypen keine Überschneidungen in den Manipulationen (**Fall 1**), werden anhand der Prozessinstanzen, in denen die Aktivitätstypen auftreten, vier Fälle unterschieden (siehe Abbildung 6-14):

Fall 1-1. *Aktivitätsinstanzen der Aktivitätstypen nur in unterschiedlichen Prozessinstanzen*

Keine Zusammenführung der Aktivitätstypen, da zwischen den Aktivitätstypen eine alternative Ordnungsbeziehung vorliegt.

Fall 1-2. *Aktivitätsinstanzen der Aktivitätstypen nur in denselben Prozessinstanzen*

Es werden die Objektinstanzen des gemeinsamen Objekttyps der entsprechenden Aktivitätsinstanzen geprüft:

a. Wenn die Aktivitätsinstanzen einer Prozessinstanz der Aktivitätstypen dieselben Objektinstanzen zum selben Zeitpunkt t des gemeinsamen Objekttyps betreffen, werden die Aktivitätstypen zu einem Aktivitätstyp zusammengeführt, indem deren Objekttypen und Manipulationen zusammengeführt werden. Zusätzlich ist der Aktivitätstyp bei allen betreffenden Aktivitätsinstanzen anzupassen.
b. Wenn die Aktivitätsinstanzen einer Prozessinstanz der Aktivitätstypen nicht dieselben Objektinstanzen zum selben Zeitpunkt t betreffen, werden die Aktivitätstypen dieser Aktivitätsinstanzen in Abhängigkeit der vorherigen bzw. nachfolgenden Objekte und Aktivitäten in der Transformation (siehe Abschnitt 6.2) weiter überprüft.

Fall 1-3. *Aktivitätsinstanzen der Aktivitätstypen teilweise in denselben Prozessinstanzen*

Es wird die nicht-leere Schnittmenge der gemeinsamen Prozessinstanzen geprüft:

a. Erzeugen die Aktivitätsinstanzen der Schnittmenge dieselben Objektinstanzen zum selben Zeitpunkt, ist ein dritter Aktivitätstyp wie in Fall 1-2a zu ergänzen. Die Aktivitätsinstanzen der Prozessinstanzen aus der Schnittmenge werden entfernt und entsprechend mit Aktivitätsinstanzen des dritten Aktivitätstyps ersetzt.
b. Erzeugen die Aktivitätsinstanzen der Schnittmenge nicht dieselben Objektinstanzen zum selben Zeitpunkt, ist dies wie in Fall 1-2b in der Transformation weiter zu betrachten

Fall 1-4. *Aktivitätsinstanzen des einen Aktivitätstyps nur in denselben Prozessinstanzen des anderen Aktivitätstyps*

Es wird die Teilmenge der gemeinsamen Prozessinstanzen geprüft:

a. Erzeugen die Aktivitätsinstanzen der Teilmenge dieselben Objektinstanzen zum selben Zeitpunkt, ist ein Aktivitätstyp wie in Fall 1-2a zu ergänzen. Die Aktivitätsinstanzen der Prozessinstanzen aus der Teilmenge werden entfernt und entsprechend mit Aktivitätsinstanzen des zusammengeführten Aktivitätstyps ersetzt. Der Aktivitätstyp der Teilmenge wird entfernt.
b. Erzeugen die Aktivitätsinstanzen der Teilmenge nicht dieselben Objektinstanzen zum selben Zeitpunkt, ist dies wie in Fall 1-2b in der Transformation weiter zu betrachten.

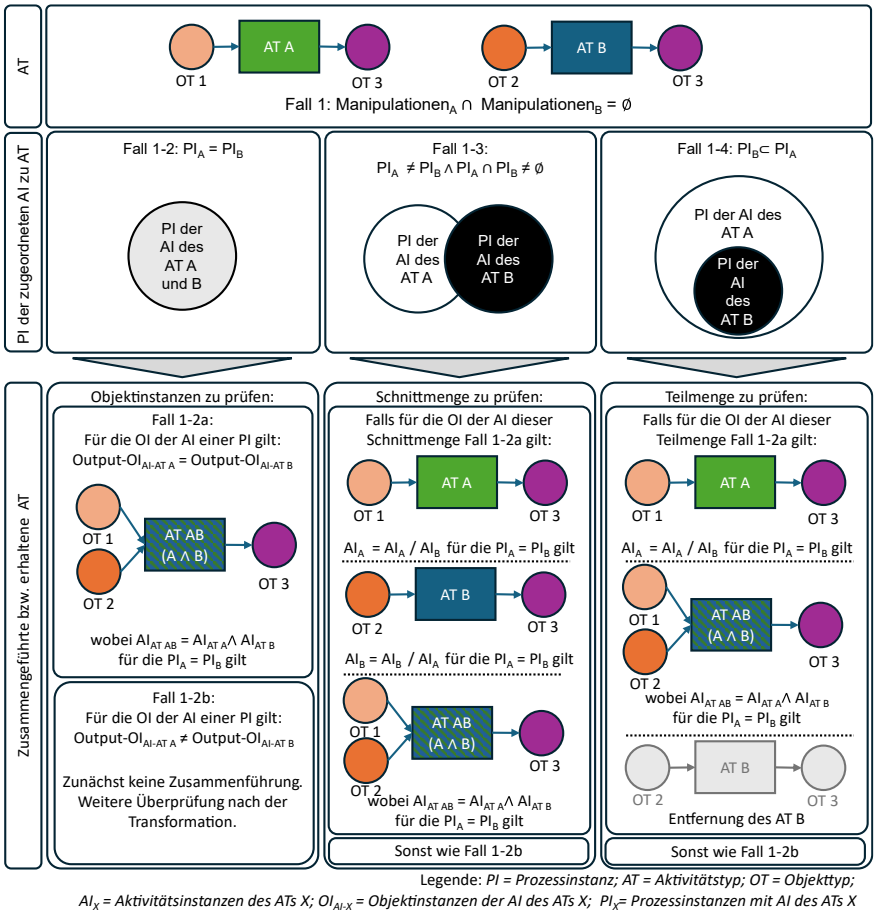


Abbildung 6-14: Zusammenführung / Entfernung der Aktivitätstypen mit unterschiedlichen Manipulationen

6.1.5.2.2 Fall 2: Aktivitätstypen mit redundanten Manipulationen

Redundante Manipulationen liegen vor, wenn zwei Aktivitätstypen dieselben Attribute auf die gleiche Weise verarbeiten. Haben zwei Aktivitätstypen nur redundante Manipulationen (**Fall 2**), werden anhand der Prozessinstanzen, in denen die Aktivitätstypen auftreten, vier Fälle unterschieden (siehe Abbildung 6-15):

Fall 2-1. *Aktivitätsinstanzen der Aktivitätstypen nur in unterschiedlichen Prozessinstanzen*

Keine Zusammenführung der Aktivitätstypen, da zwischen den Aktivitätstypen eine alternative Ordnungsbeziehung vorliegt.

Fall 2-2. *Aktivitätsinstanzen der Aktivitätstypen nur in den gleichen Prozessinstanzen*

Es werden die Objektinstanzen des gemeinsamen Objekttyps der entsprechenden Aktivitätsinstanzen geprüft:

a. Wenn die Aktivitätsinstanzen einer Prozessinstanz der Aktivitätstypen dieselben Objektinstanzen zum selben Zeitpunkt t des gemeinsamen Objekttyps betreffen, wird ein Aktivitätstyp (inklusive der Aktivitätsinstanzen des Aktivitätstyp) entfernt, da sie die Objekte redundant manipulieren. Dazu wird folgendes überprüft:
   i. Überprüfen, ob Aktivitätstypen aus zeitlichen Zusammenhängen existieren, die dieselben Input- und Output-Objekttypen betreffen.
   ii. Überprüfen, ob ein beteiligter Objekttyp sonst von keinem anderen Aktivitätstyp Input oder Output ist.

   Existiert ein Aktivitätstyp aus den zeitlichen Zusammenhängen, der dieselben Input- und Output-Objekttypen betrifft wie einer der betrachteten Aktivitätstypen, so ist der Aktivitätstyp zu behalten, der mit dem Aktivitätstyp aus den zeitlichen Zusammenhängen übereinstimmt. Zusätzlich muss der andere Aktivitätstyp keinen beteiligten Objekttyp aufweisen, der in keinem anderem Aktivitätstyp beteiligt ist. Ist ein Objekttyp beteiligt, der sonst keine weiteren Beteiligungen hat, bleibt der Aktivitätstyp erhalten.
b. Wenn die Aktivitätsinstanzen einer Prozessinstanz der Aktivitätstypen nicht dieselben Objektinstanzen zum selben Zeitpunkt t betreffen, werden die Aktivitätstypen dieser Aktivitätsinstanzen in Abhängigkeit der vorherigen bzw. nachfolgenden Objekte und Aktivitäten in der Transformation (siehe Abschnitt 6.2) weiter überprüft.

Fall 2-3. *Aktivitätsinstanzen der Aktivitätstypen teilweise in denselben Prozessinstanzen*

Es wird die nicht-leere Schnittmenge der gemeinsamen Prozessinstanzen geprüft:

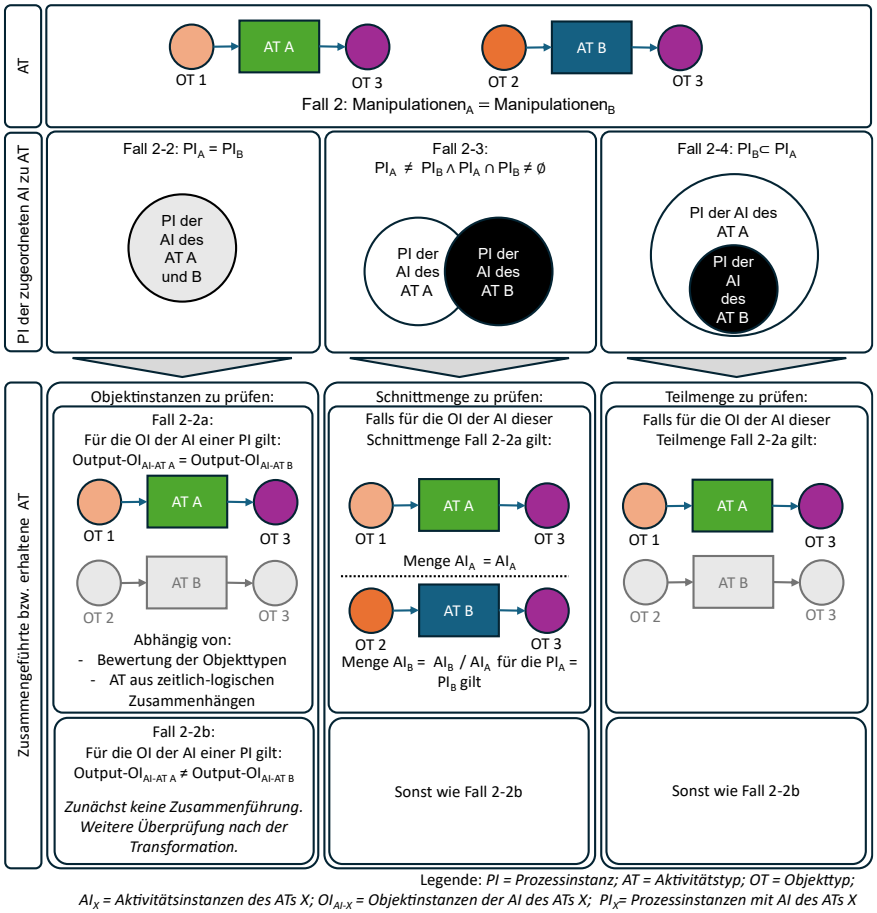
a. Erzeugen die Aktivitätsinstanzen der Schnittmenge dieselben Objektinstanzen zum selben Zeitpunkt, ist wie in Fall 2-2a zu prüfen, welche Aktivitätsinstanzen welchen Aktivitätstyps entfernt werden, da sonst redundante Objektmanipulationen durchgeführt werden würden.
b. Erzeugen die Aktivitätsinstanzen der Schnittmenge nicht dieselben Objektinstanzen zum selben Zeitpunkt, ist dies wie in Fall 2-2b in der Transformation weiter zu betrachten.

Fall 2-4. *Aktivitätsinstanzen des einen Aktivitätstyps nur in denselben Prozessinstanzen des anderen Aktivitätstyp*

Es wird die Teilmenge der gemeinsamen Prozessinstanzen geprüft:

a. Erzeugen die Aktivitätsinstanzen der Teilmenge dieselben Objektinstanzen zum selben Zeitpunkt, werden der Aktivitätstyp der Teilmenge sowie dessen Instanzen entfernt.

b. Erzeugen die Aktivitätsinstanzen der Teilmenge nicht dieselben Objektinstanzen zum selben Zeitpunkt, ist dies wie in Fall 2-2b in der Transformation weiter zu betrachten.



Legende: *PI = Prozessinstanz; AT = Aktivitätstyp; OT = Objekttyp; AI$_X$ = Aktivitätsinstanzen des ATs X; OI$_{AI\text{-}X}$ = Objektinstanzen der AI des ATs X; PI$_X$= Prozessinstanzen mit AI des ATs X*

Abbildung 6-15: Entfernung der Aktivitätstypen mit redundanten Manipulationen

6.1.5.2.3 Fall 3: Aktivitätstypen mit teilweise redundanten Manipulationen

Haben zwei Aktivitätstypen teilweise redundante Manipulationen (**Fall 3**), ist zunächst zu prüfen, ob die zusätzlichen Manipulationen der Aktivitätstypen ggf. bereits durch andere Aktivitätstypen abgedeckt sind. Ist dies der Fall, würde durch Entfernen der Manipulationen anstatt Fall 3 der Fall 4 zutreffen. Werden die Manipulationen nicht durch andere Aktivitätstypen abgedeckt, können anhand der Prozessinstanzen, in denen die Aktivitätstypen auftreten, vier Fälle unterschieden werden (siehe Abbildung 6-16):

Fall 3-1. *Aktivitätsinstanzen der Aktivitätstypen nur in unterschiedlichen Prozessinstanzen*

Keine Zusammenführung der Aktivitätstypen, da zwischen den Aktivitätstypen eine alternative Ordnungsbeziehung vorliegt.

Fall 3-2. *Aktivitätsinstanzen der Aktivitätstypen nur in den gleichen Prozessinstanzen*

Es werden die Objektinstanzen des gemeinsamen Objekttyps der entsprechenden Aktivitätsinstanzen geprüft:

a. Wenn die Aktivitätsinstanzen einer Prozessinstanz der Aktivitätstypen dieselben Objektinstanzen zum selben Zeitpunkt t betreffen, werden die Aktivitätstypen zu einem Aktivitätstyp zusammengeführt, indem deren Objekttypen und Manipulationen zusammengeführt werden. Zusätzlich ist der Aktivitätstyp bei allen betreffenden Aktivitätsinstanzen anzupassen. Da die Manipulationen eine Schnittmenge haben, ist zu entscheiden, welche der redundanten Manipulationen entfernt werden sollen.
b. Wenn die Aktivitätsinstanzen einer Prozessinstanz der Aktivitätstypen nicht dieselben Objektinstanzen zum selben Zeitpunkt t betreffen, werden die Aktivitätstypen dieser Aktivitätsinstanzen in Abhängigkeit der vorherigen bzw. nachfolgenden Objekte und Aktivitäten in der Transformation (siehe Abschnitt 6.2) weiter überprüft.

Fall 3-3. *Aktivitätsinstanzen der Aktivitätstypen teilweise in denselben Prozessinstanzen*

Es wird die nicht-leere Schnittmenge der gemeinsamen Prozessinstanzen geprüft:

a. Erzeugen die Aktivitätsinstanzen der Schnittmenge dieselben Objektinstanzen zum selben Zeitpunkt, ist ein Aktivitätstyp wie in Fall 3-2a zu ergänzen. Die Aktivitätsinstanzen der Prozessinstanzen aus der Schnittmenge werden entfernt und entsprechend mit Aktivitätsinstanzen des dritten Aktivitätstyps ersetzt. Zusätzlich sollen die redundanten Manipulationen überprüft werden, damit ausgewählt werden kann, welche dieser Manipulationen entfernt werden sollen.
b. Erzeugen die Aktivitätsinstanzen der Schnittmenge nicht dieselben Objektinstanzen zum selben Zeitpunkt, ist dies wie in Fall 3-2b in der Transformation weiter zu betrachten.

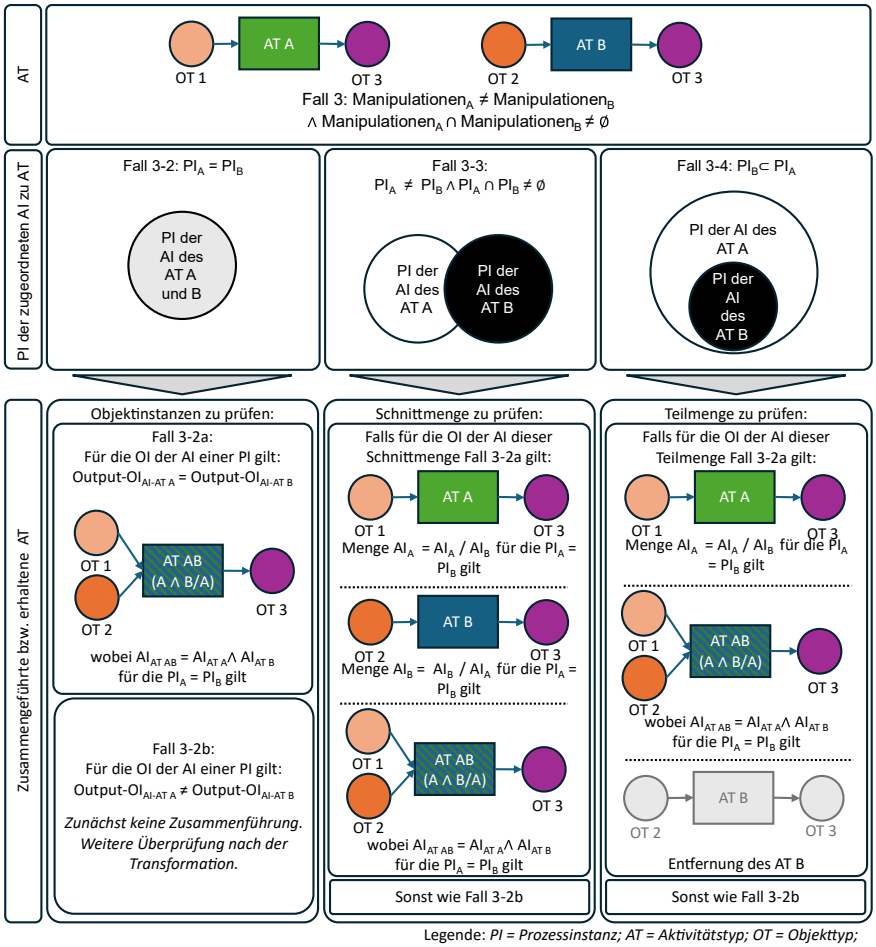
Fall 3-4. *Aktivitätsinstanzen des einen Aktivitätstyps nur in denselben Prozessinstanzen des anderen Aktivitätstyps*

Es wird die Teilmenge der gemeinsamen Prozessinstanzen geprüft:

a. Erzeugen die Aktivitätsinstanzen der Teilmenge dieselben Objektinstanzen zum selben Zeitpunkt, ist ein Aktivitätstyp wie in Fall 3-2a zu ergänzen. Die Aktivitätsinstanzen der Prozessinstanzen aus der Teilmenge werden entfernt und entsprechend mit Aktivitätsinstanzen des zusammengeführten Aktivitätstyps ersetzt. Zusätzlich sollen die redun-

danten Manipulationen überprüft werden, damit ausgewählt werden kann, welche dieser Manipulationen entfernt werden können. Der Aktivitätstyp der Teilmenge wird entfernt.

b. Erzeugen die Aktivitätsinstanzen der Teilmenge nicht dieselben Objektinstanzen zum selben Zeitpunkt, ist dies wie in Fall 3-2b in der Transformation weiter zu betrachten.



Legende: *PI = Prozessinstanz; AT = Aktivitätstyp; OT = Objekttyp;*
AI$_X$ = Aktivitätsinstanzen des ATs X; OI$_{AI\text{-}X}$ = Objektinstanzen der AI des ATs X; PI$_X$= Prozessinstanzen mit AI des ATs X

Abbildung 6-16: Zusammenführung / Entfernung der Aktivitätstypen mit teilweise redundanten Manipulationen

6.1.5.2.4 Fall 4: Nur redundante Manipulationen des einen Aktivitätstyp zu den Manipulationen des anderen Aktivitätstyps

Sind die Manipulationen des einen Aktivitätstyps eine echte Teilmenge der Manipulationen des anderen Aktivitätstyps (**Fall 4**), ist zunächst zu prüfen, ob die zusätzlichen Manipulationen der Obermenge ggf. bereits durch andere Aktivitätstypen abgedeckt sind. Ist dies der Fall, würde durch Entfernen der Manipulationen anstatt Fall 4 der Fall 2 zutreffen. Werden die Manipulationen nicht durch andere Aktivitätstypen abgedeckt, können anhand der Prozessinstanzen, in denen die Aktivitätstypen auftreten, vier Fälle unterschieden werden (siehe Abbildung 6-17):

Fall 4-1. *Aktivitätsinstanzen der Aktivitätstypen nur in unterschiedlichen Prozessinstanzen*
Keine Zusammenführung der Aktivitätstypen, da zwischen den Aktivitätstypen eine alternative Ordnungsbeziehung vorliegt.

Fall 4-2. *Aktivitätsinstanzen der Aktivitätstypen nur in denselben Prozessinstanzen*
Es werden die Objektinstanzen des gemeinsamen Objekttyps der entsprechenden Aktivitätsinstanzen geprüft:

a. Wenn die Aktivitätsinstanzen einer Prozessinstanz der Aktivitätstypen dieselben Objektinstanzen zum selben Zeitpunkt t betreffen, wird ein Aktivitätstyp (inklusive der Aktivitätsinstanzen des Aktivitätstyp) entfernt, da sie die Objekte redundant manipulieren. Es wird der Aktivitätstyp ausgewählt werden, der die Obermenge der Manipulationen enthält. Jedoch werden auch folgende Aspekte überprüft:
   i. Überprüfen, ob Aktivitätstypen aus zeitlichen Zusammenhängen existieren, die dieselben Input- und Output-Objekttypen betreffen.
   ii. Überprüfen, ob ein beteiligter Objekttyp in keinem anderen Aktivitätstyp vorkommt.

   Existiert ein Aktivitätstyp aus den zeitlichen Zusammenhängen, der dieselben Input- und Output-Objekttypen betrifft wie einer der betrachteten Aktivitätstypen, so ist der Aktivitätstyp zu behalten, der mit dem Aktivitätstyp aus den zeitlichen Zusammenhängen übereinstimmt. Zusätzlich muss der andere Aktivitätstyp keinen beteiligten Objekttyp aufweisen, der in keinem anderem Aktivitätstyp beteiligt ist. Ist ein Objekttyp beteiligt, der sonst keine weiteren Beteiligungen hat, bleibt dieser Aktivitätstyp erhalten.
b. Wenn die Aktivitätsinstanzen einer Prozessinstanz der Aktivitätstypen nicht dieselben Objektinstanzen zum selben Zeitpunkt t betreffen, werden die Aktivitätstypen dieser Aktivitätsinstanzen in Abhängigkeit der vorherigen bzw. nachfolgenden Objekte und Aktivitäten in der Transformation (siehe Abschnitt 6.2) weiter überprüft.

Fall 4-3. *Aktivitätsinstanzen der Aktivitätstypen teilweise in denselben Prozessinstanzen*
Es wird die nicht-leere Schnittmenge der gemeinsamen Prozessinstanzen geprüft:

a. Erzeugen die Aktivitätsinstanzen der Schnittmenge dieselben Objektinstanzen zum selben Zeitpunkt, sind die Aktivitätsinstanzen der Schnittmenge der Prozessinstanzen des Aktivitätstyps, der die Teilmenge der Manipulationen abbildet, zu entfernen.
b. Erzeugen die Aktivitätsinstanzen der Schnittmenge nicht dieselben Objektinstanzen zum selben Zeitpunkt, ist dies wie in Fall 4-2b in der Transformation weiter zu betrachten

Fall 4-4. *Aktivitätsinstanzen des einen Aktivitätstyps nur in denselben Prozessinstanzen des anderen Aktivitätstyp*
Es wird die Teilmenge der gemeinsamen Prozessinstanzen geprüft:

a. Erzeugen die Aktivitätsinstanzen der Teilmenge dieselben Objektinstanzen zum selben Zeitpunkt, werden der Aktivitätstyp der Teilmenge sowie dessen Instanzen entfernt, falls dieser auch die Teilmenge der Manipulationen abbildet. Andernfalls ist wie in Fall 4-3 vorzugehen
b. Erzeugen die Aktivitätsinstanzen der Teilmenge nicht dieselben Objektinstanzen zum selben Zeitpunkt, ist dies wie in Fall 4-2b in der Transformation weiter zu betrachten.

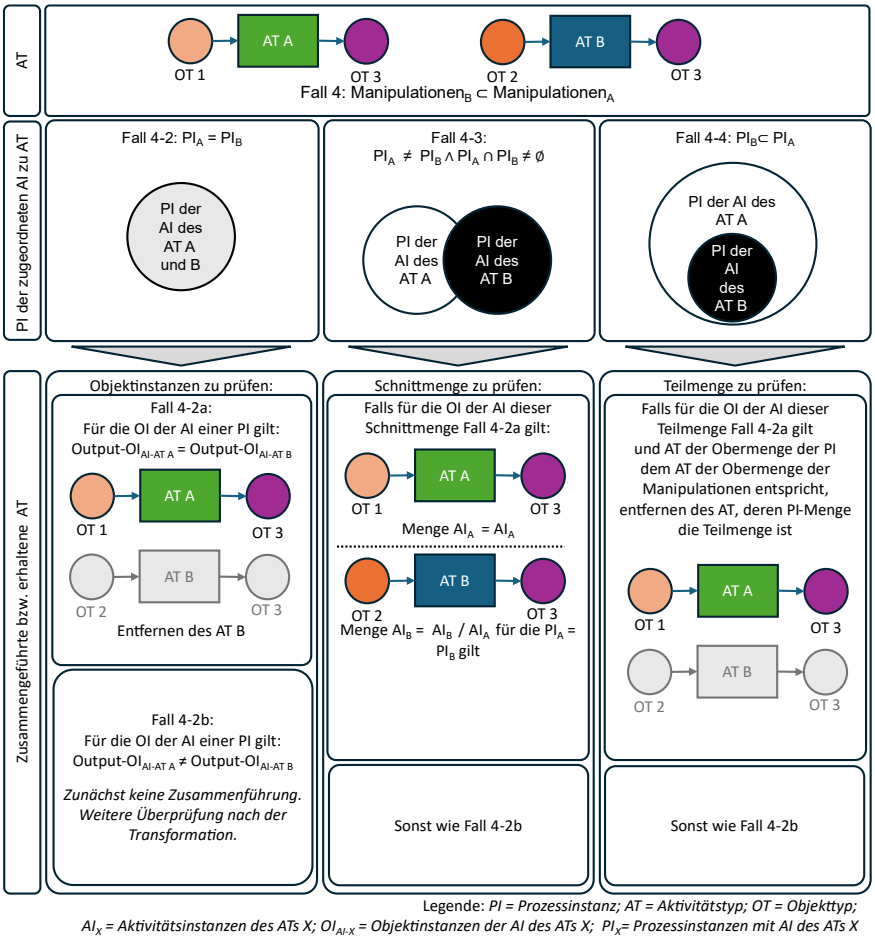


Abbildung 6-17: Entfernung der Aktivitätstypen mit nur redundanten Manipulationen

Für die Modellierung dieser Aktivitätstypen sind noch weitere Zusammenführungen bzw. Entfernungen zu prüfen, die jedoch abhängig der Anforderungsstufe der Transformation durchzuführen sind. Daher wird die weitere Betrachtung erst in der Transformation (siehe Abschnitt 6.2) durchgeführt.

6.2 Transformation

In dieser Arbeit soll ein Geschäftsprozessmodell in Form eines höheren Petri-Netzes generiert werden. Dazu sind die in Abschnitt 6.1 extrahierten Informationen zu Objekttypen und Aktivitätstypen in die Elemente eines Petri-Netzes zu transformieren. Die Ob-

jekttypen werden dabei durch die Stellen, die Aktivitätstypen durch die Transitionen und der Kontrollfluss durch die Kanten eines Petri-Netzes dargestellt. Wie in Abschnitt 3.2 beschrieben, können in höheren Petri-Netzen zusätzliche Informationen abgebildet werden. Daher können bei JSON- bzw. XML-Netzen wie in Tabelle 3-2 beschrieben zusätzlich die Attribut-Struktur der Objekttypen und die Manipulationen bzw. Regeln der Aktivitätstypen abgebildet werden (siehe Definition 6-7). Damit unterscheidet sich dieser Ansatz insbesondere zu den in Kapitel 4 vorgestellten Ansätzen. Zum einen basiert er auf Objektinstanzen und leitet aus diesen die Aktivitäten ab. Zum anderen kann nicht nur der beobachtete Kontrollfluss dargestellt werden, sondern es kann auch analysiert werden, auf welchen Informationen basierend welcher Pfad gewählt wird und wie die Zustände der Objekte durch Aktivitäten verändert werden.

Für die einzelnen Prozessinstanzen können zunächst die Kontrollflussmuster Sequenz (siehe Abschnitt 2.2.1.1), Nebenläufigkeit (siehe Abschnitt 2.2.1.2) und Iteration (siehe Abschnitt 2.2.1.4) abgeleitet werden. Das Kontrollflussmuster Alternative Ausführung (siehe Abschnitt 2.2.1.3) lässt sich durch die Analyse verschiedener Prozessinstanzen ermitteln, wenn unterschiedliche Aktivitätstypen alternativ in verschiedenen Prozessinstanzen auf einen Output folgen können. Zusätzlich soll das Geschäftsprozessmodell auch Informationen zu den Objekten abbilden. Folgend wird dies anhand der objekt(-fluss-)orientierten Perspektiven betrachtet (siehe Abbildung 2-9 *a-d*)

Basierend auf den zeitlichen Zusammenhängen wird zunächst der Kontrollfluss abgeleitet, um anschließend das resultierende generierte Petri-Netz um *statische Aspekte* der Objekte (die Objektstruktur der Objekttypen) zu erweitern (in Abbildung 2-9 *1a*). Hierzu werden die Stellen als Objektspeicher interpretiert, denen ein bestimmter Objekttyp mit beschriebener Objektstruktur zugeordnet wird.

Basierend auf den inhaltlichen Zusammenhängen werden die *Objektbeziehungstypen* zu den Objekttypen in einem Petri-Netz abgebildet (in Abbildung 2-9 *1b*) und *dynamische Aspekte* der Objekte durch deren Zustandsänderungen beschrieben (in Abbildung 2-9 *2a*). Dazu werden Transitionen um Inschriften erweitert, die die durchzuführenden Manipulationen der Objekte in einer Aktivität beschreiben.

Basierend auf der Betrachtung aller Zusammenhänge sowie der Analyse über alle Prozessinstanzen hinweg, werden sowohl direkte als auch indirekte Objektbeziehungstypen abgebildet. Für Aktivitäten, die Objekte verbrauchen, verändern, erzeugen oder lesen, können Ein- und Ausgabeparameter definiert werden. Bei der Ausführung einer Aktivität werden die Eingabeparameter gelesen und bei der Beendigung der Aktivität die Ausgabeparameter belegt. Die Ausführungsbedingung einer Aktivität kann daher in Abhängigkeit von Objekten definiert werden. Bei dieser Objektflussmodellierung überlagert die objektorientierte Perspektive die Kontrollflussperspektive. Dabei werden in den Model-

len die Attribut-Wert-Paare der Objekte zur Erfüllung von Bedingungen in Geschäftsprozessen analysiert. Damit ergeben sich die folgenden Transformationsschritte aus Tabelle 6-5.

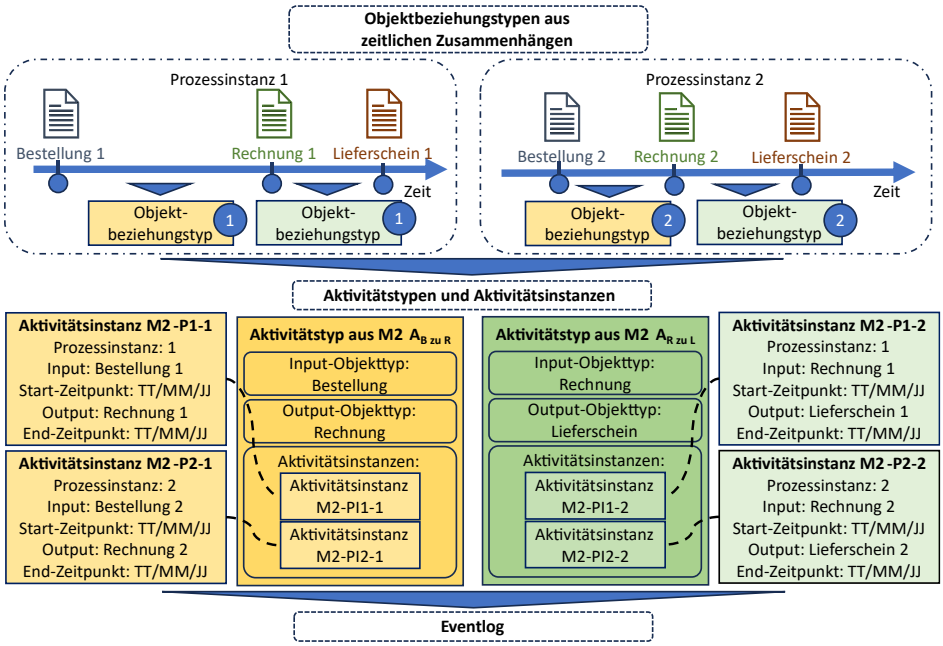
Tabelle 6-5: Transformationsschritte

Schritt	Beschreibung	Extraktionsschritt
1	Kontrollflussabbildung mit Objektstrukturen der Stellen Ordnungsbeziehungen werden aus den zeitlichen Zusammenhängen zwischen den Aktivitätstypen extrahiert und im Petri-Netz repräsentiert.	Ordnungsbeziehung aus zeitlichen Zusammenhängen
2	Kontrollflussabbildung mit Objektstrukturen und Manipulationsregeln Ordnungsbeziehungen werden aus den inhaltlichen Zusammenhängen zwischen den Aktivitätstypen extrahiert und im Petri-Netz repräsentiert. Die zeitlichen Zusammenhänge werden eingehalten. Zusätzlich sind die Aktivitätstypen über Manipulationsregeln weiter spezifiziert.	Ordnungsbeziehung aus inhaltlichen Zusammenhängen
3	Kontrollflussabbildung mit Objektstrukturen, Manipulationsregeln und Ausführungsbedingungen. Anhand der abgeleiteten Ordnungsbeziehungen und den Objektinstanzen der Aktivitätsinstanzen in den einzelnen Prozessinstanzen, werden Entscheidungsregeln abgeleitet	Ausführungsbedingungen

6.2.1 Ordnungsbeziehungen aus zeitlichen Zusammenhängen

Zur Ableitung der Ordnungsbeziehungen zwischen den Aktivitätstypen eines Geschäftsprozesses aus den zeitlichen Zusammenhängen können Process Discovery Algorithmen verwendet werden [Aals22]. Diese generieren basierend auf einem Eventlog ein Geschäftsprozessmodell. Diese Eventlogs sollen die Annahme erfüllen, dass die Ereignisse in den Eventlogs sequenziell aufgezeichnet werden und sich jedes Ereignis auf einen definierten Task (hier: Aktivitätstyp) und einen Case (hier: Prozessinstanz) bezieht [Aals22]. Daher könnte aus den Aktivitätsinstanzen (mit zugehörigem Aktivitätstyp und zugeordneter Prozessinstanz), die aus den zeitlichen Zusammenhängen abgeleitet wurden (in Abschnitt 6.1.5; siehe Abbildung 6-9), Eventlogs generiert werden, indem für jede Aktivitätsinstanz ein Ereignis in einem Eventlog erstellt und dieses durch den Namen des Aktivitätstyps, der Prozessinstanz-Identifikationsnummer sowie einem Zeitstempel

definiert wird. Der Zeitstempel besteht aus dem Startzeitpunkt und dem Endzeitpunkt. Der Startzeitpunkt wird durch den Mittelwert der Zeitpunkte der Input-Dokumente und der Endzeitpunkt durch den Mittelwert der Zeitpunkte der Output-Objektinstanzen bestimmt. Hierbei werden nur die Aktivitätsinstanzen berücksichtigt, die aus den zeitlichen Zusammenhängen abgeleitet wurden, um zum einen gewährleisten zu können, dass Zeitpunkte gegeben sind und zum anderen, dass keine redundanten Aktivitätstypen abgebildet werden. Das Vorgehen zur Eventlog-Generierung ist in Abbildung 6-18 verdeutlicht.



Prozessinstanz	Aktivitätstyp	Start-Zeitpunkt	End-Zeitpunkt
1	$A_{B\ zu\ R}$	TT/MM/JJ	TT/MM/JJ
2	$A_{B\ zu\ R}$	TT/MM/JJ	TT/MM/JJ
1	$A_{R\ zu\ L}$	TT/MM/JJ	TT/MM/JJ
2	$A_{R\ zu\ L}$	TT/MM/JJ	TT/MM/JJ

Abbildung 6-18: Eventlog-Generierung basierend auf den Aktivitätsinstanzen

Das prinzipielle Vorgehen bei Process Discovery Algorithmen wird folgend zunächst anhand des α-Algorithmus nach [AaWM04] beschrieben. Durch den Algorithmus soll die Reihenfolge der Aktivitäten anhand der Ordnungsbeziehungen definiert werden. Es wird zwischen vier Ordnungsbeziehungen zwischen Aktivitäten unterschieden, die in der Definition 6-8 beschrieben sind.

Definition 6-8: Eventlog basierte Ordnungsbeziehungen

Sei L ein Eventlog, $a, b \in A$ Aktivitäten, dann gilt:

1. $\boldsymbol{a >_L b}$ *dann und nur dann, wenn es ein Trace* $\sigma = < a_1, a_2, a_3, \dots, a_{n-1} >$ *und* $i \in \{1, 2, \dots, n-2\}$ *gibt, sodass* $\sigma \in L$ *und* $a_i = a$ *und* $a_{i+1} = b$ *gilt,*

2. $\boldsymbol{a \rightarrow_L b}$ *dann und nur dann, wenn* $\boldsymbol{a >_L b}$ *und* $\boldsymbol{b \not>_L a}$ *gilt,*

3. $\boldsymbol{a \#_L b}$ *dann und nur dann, wenn* $\boldsymbol{a \not>_L b}$ *und* $\boldsymbol{b \not>_L a}$ *gilt,*

4. $\boldsymbol{a \|_L b}$ *dann und nur dann, wenn* $\boldsymbol{a >_L b}$ *und* $\boldsymbol{b >_L a}$ *gilt.*

[AaWM04]

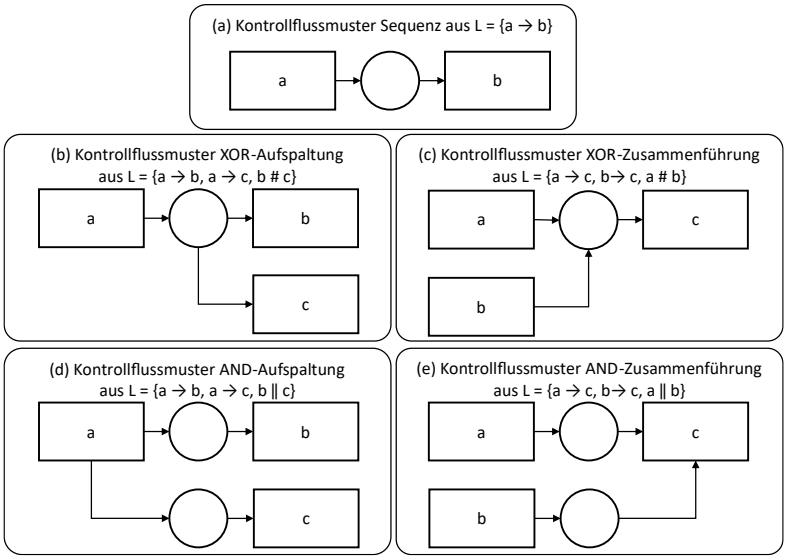
Um die Mengen nach Definition 6-8 zu bilden, wird zuerst die Menge der Ordnungsbeziehungen $>L_O$ (direkt aufeinanderfolgende Aktivitätsinstanzen) analysiert. Für jede Ordnungsbeziehung O in der Liste L_O wird die Existenz einer inversen Ordnungsbeziehung O_{-1} überprüft. Das heißt, es wird für eine beliebige Ordnungsbeziehung der Form $a > b$ geprüft, ob es eine inverse Ordnungsbeziehung $b > a$ in der Liste L_O gibt. Anhand dieser Menge werden anschließend die Mengen $\rightarrow_L$ (direkt aufeinanderfolgende Aktivitäten), $\#_L$ (nicht aufeinander folgende Aktivitäten) und $\|_L$ (nebenläufige Aktivitäten) gebildet. Zusätzlich wird die Menge $\rightarrow_L^{-1}$ aus den inversen Beziehungen zu $\rightarrow_L$ gebildet. Für jedes Eventlog L über eine Menge A von Aktivitäten und zwei beliebige Aktivitäten $a, b \in$ A gilt $a >_L b$, $b \rightarrow_L a$, $a \#_L b$ oder $a \|_L b$, d. h. für jedes Paar von Aktivitäten gilt genau eine dieser Ordnungsbeziehungen.

Beim α-Algorithmus werden zunächst alle Aktivitäten als Transitionen abgebildet sowie die Start- und Endknoten definiert. Anschließend werden aus den Aktivitäten der Menge $\rightarrow_{L_o}$ die jeweiligen Knoten als Sequenz verbunden (Kontrollflussmuster (a) in Abbildung 6-19). Das heißt, wenn es für die Ordnungsbeziehung a > b keine inverse Ordnungsbeziehung b > a gibt, gilt $a \rightarrow_L b$ und somit werden die Transitionen t_a und t_b (die die Aktivitäten *a* und *b* abbilden*)* als Sequenz mit einer zu den Transitionen adjazenten Stelle s_1 abgebildet. Es wird s_1 ergänzt, falls weder t_a eine Stelle im Nachbereich noch t_b eine Stelle im Vorbereich hat, ansonsten werden entsprechend diese Stellen verwendet. Zusätzlich gilt:

- Gelten für die Aktivität a, b und c die Ordnungsbeziehungen $a \rightarrow_L b$, $a \rightarrow_L c$ und $b \# c$, entspricht dies dem XOR-Aufspaltungs-Kontrollflussmuster.
- Gelten für die Aktivität a, b und c die Ordnungsbeziehungen $a \rightarrow_L c$, $b \rightarrow_L c$, und $a \# b$, entspricht dies dem XOR-Zusammenführungs-Kontrollflussmuster.
- Gelten für die Aktivität a, b und c die Ordnungsbeziehungen $a \rightarrow_L b$, $a \rightarrow_L c$, und $b \| c$, entspricht dies dem AND-Aufspaltungs-Kontrollflussmuster.

- Gelten für die Aktivität a, b und c die Ordnungsbeziehungen $a \rightarrow_L c$, $b \rightarrow_L c$, und $a \parallel b$, entspricht dies dem AND- Zusammenführungs-Kontrollflussmuster.

Damit werden basierend auf den Ordnungsbeziehungen die in Abschnitt 2.2.1 beschriebenen Kontrollflussmuster entdeckt (siehe Abbildung 6-19).



[AaWM04]

Abbildung 6-19: Kontrollflussmuster basierend auf den Ordnungsbeziehungen

Die Aktivitätstypen und -instanzen werden in Abschnitt 6.1 aus den Zustandsübergängen der Objektinstanzen basierend auf den zeitlichen Zusammenhängen abgeleitet. Besteht beispielsweise ein Geschäftsprozess aus zwei Aktivitäten, bei dem Aktivität 1 aus dem Input-Objekt A die Output-Objekte B_1 und B_2 erzeugt und Aktivität 2 aus den Input-Objekten B_1 und B_2 das Output-Objekt C erzeugt, tritt folgende Herausforderung auf: Aufgrund der mehrfachen Inputs und Outputs wären die möglichen zeitlichen Ordnungen der Objektinstanzen in den verschiedenen Prozessinstanzen AB_1B_2C und AB_2B_1C möglich, was zu folgenden Aktivitätstypen führt:

- Aktivitätstypen der Prozessinstanzen mit AB_1B_2C als zeitliche Ordnung der Objekttypen:
  - Aktivitätstyp 1a mit A als Input-Objekttyp, B_1 als Output-Objekttyp,
  - Aktivitätstyp 2a mit B_1 als Input-Objekttyp, B_2 als Output-Objekttyp,
  - Aktivitätstyp 3a mit B_2 als Input-Objekttyp, C als Output-Objekttyp.
- Aktivitätstypen der Prozessinstanzen mit AB_2B_1C als zeitliche Ordnung der Objekttypen:

- Aktivitätstyp 1b mit A als Input-Objekttyp, B_2 als Output-Objekttyp,
- Aktivitätstyp 2b mit B_2 als Input-Objekttyp, B_1 als Output-Objekttyp,
- Aktivitätstyp 3b mit B_1 als Input-Objekttyp, C als Output-Objekttyp.

Nach den Regeln aus Abschnitt 6.1.5 wird für diese Aktivitätstypen keine Zusammenführung vorgenommen, wodurch sich ohne weitere Anpassungen eine Darstellung von mehreren alternativen Aktivitätstypen in der Transformation ergibt. Daher ist das Vorgehen der Generierung anhand der einzelnen Kontrollflussmuster weiter zu betrachten. Zusätzlich sollen den Stellen auch Objekttypen zugeordnet werden, was ebenfalls nicht durch bisherige Process Mining Algorithmen gelöst wird (siehe Kapitel 4). Dazu ist zu überprüfen, dass die Stellen vor/nach den Transitionen, die die Aktivitätstypen abbilden, die Objekttypen dieser Aktivitätstypen abbilden.

6.2.1.1 Sequenz und Iteration

Zur Betrachtung des Kontrollflussmusters *Sequenz* wird das Beispiel in Abbildung 6-18 weitergeführt. In dem Beispiel kann nur die Ordnungsbeziehung *Rechnungserstellung* $>_L$ *Lieferscheinerstellung* ermittelt werden, da auf die Aktivität *Rechnungserstellung* immer die Aktivität *Lieferscheinerstellung* folgt, aber nie umgekehrt. Daher werden die Aktivitäten *Rechnungserstellung* und *Lieferscheinerstellung* in einem Petri-Netz als Sequenz (siehe Abbildung 6-19 (a)) abgebildet. Durch die Betrachtung der definierten In- und Output-Objekttypen der Aktivitätstypen werden den Stellen die entsprechenden Objekttypen zugeordnet und demnach wird das in Abbildung 6-20 dargestellte Petri-Netz generiert.

Abbildung 6-20: Generiertes Petri-Netz aus dem Eventlog in Abbildung 6-18

Da die Aktivitätsinstanzen über den Input und Output definiert werden, kann eine Iteration an einer Stelle abgebildet werden, sobald bei einem Aktivitätstyp der Objekttyp des Inputs und Outputs gleich ist und dieser Aktivitätstyp mehrmals in einer Prozessinstanz ausgeführt wird.

6.2.1.2 XOR- Aufspaltung und XOR- Zusammenführung

Bei der Analyse des Kontrollflussmusters XOR-Aufspaltung bzw. XOR-Zusammenführung dient ein Geschäftsprozess als Beispiel, in dem eine Bestellung aus einer Anfrage resultiert und anschließend entweder eine Zu- oder Absage erfolgt. Bei diesem Beispiel liegen die Objektinstanzen in zwei möglichen zeitlichen Zusammenhän-

gen vor (Anfrage > Bestellung > Zusage und Anfrage > Bestellung > Absage). Aus diesen zeitlichen Zusammenhängen werden drei Aktivitätstypen definiert (*Anfrage zu Bestellung (1)*; *Bestellung zu Zusage (2)*; *Bestellung zu Absage (3)*). Wird aus den Aktivitätsinstanzen ein Eventlog generiert, können zwei Ordnungsbeziehungen abgeleitet werden ((1) → (2); (1) → (3)). Da die Aktivitätstypen 2 und 3 ausschließlich in unterschiedlichen Prozessinstanzen auftreten, kann das Kontrollflussmuster der XOR-Aufspaltung entdeckt werden ((2)#(3)). Die anschließende Analyse der Stellen zur Zuordnung von Objekttypen zeigt, dass die Endstelle zwei verschiedene Objekttypen repräsentiert. Daher ist eine zusätzliche Stelle zu ergänzen.

Dieses Vorgehen ist in Abbildung 6-21 für eine XOR- Aufspaltung dargestellt. Bei einer XOR-Zusammenführung kann das gleiche Vorgehen verwendet werden. Dies ist in Abbildung 6-22 dargestellt.

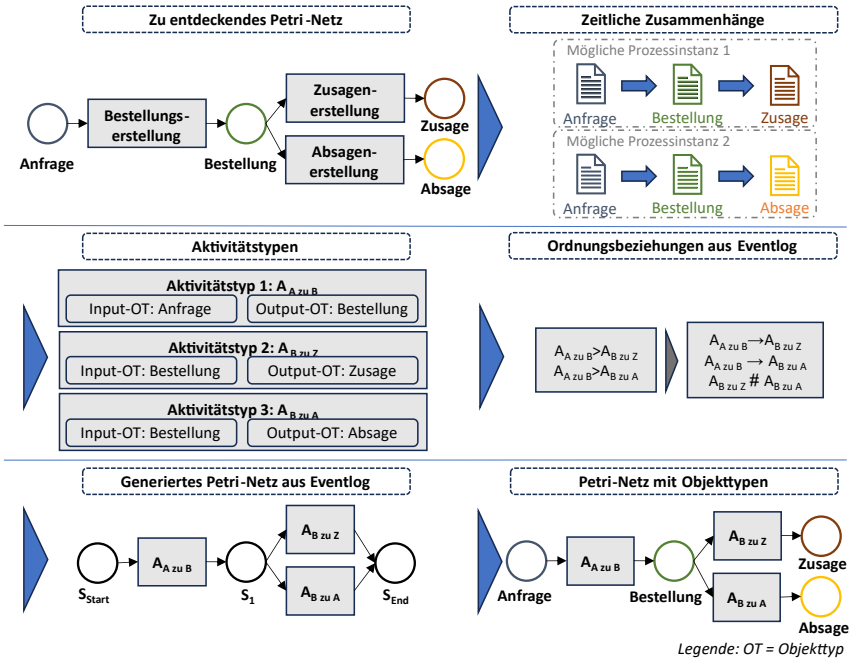


Abbildung 6-21: Generierung eines Geschäftsprozessmodells für eine XOR-Aufspaltung

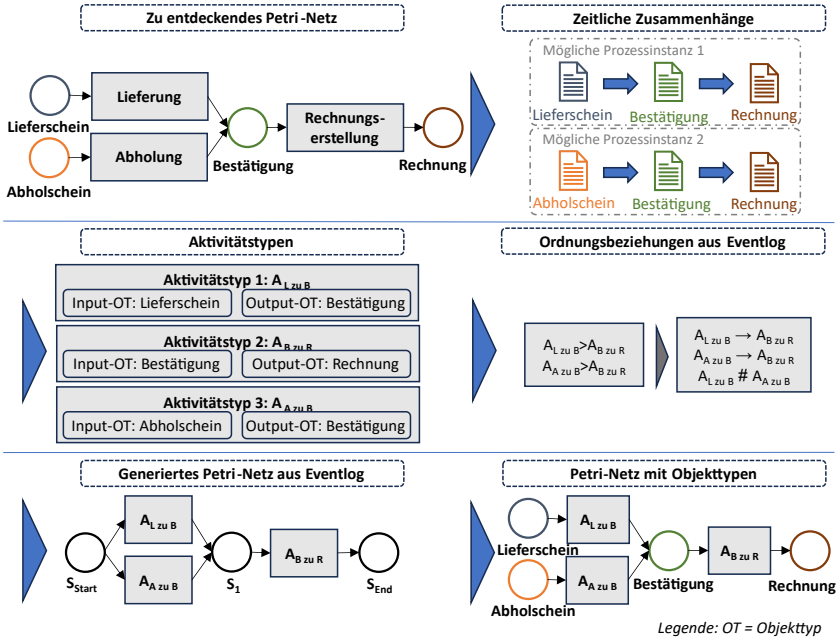


Abbildung 6-22: Generierung eines Geschäftsprozessmodells für eine XOR-Zusammenführung

6.2.1.3 AND-Aufspaltung und AND-Zusammenführung

Hat in einem Geschäftsprozess eine Aktivität mehrere Inputs oder Outputs, werden bei der objektbasierten Ableitung der Aktivitätstypen basierend auf den zeitlichen Zusammenhängen mehrere Aktivitätstypen abgeleitet. Daher werden bei AND-Aufspaltungen bzw. -Zusammenführungen zunächst auch Aktivitäten erzeugt, die nicht unmittelbar als Transitionen in einem Modell abgebildet werden sollten. Dies ist in Abbildung 6-23 für eine AND-Aufspaltung und in Abbildung 6-24 für eine AND-Zusammenführung dargestellt.

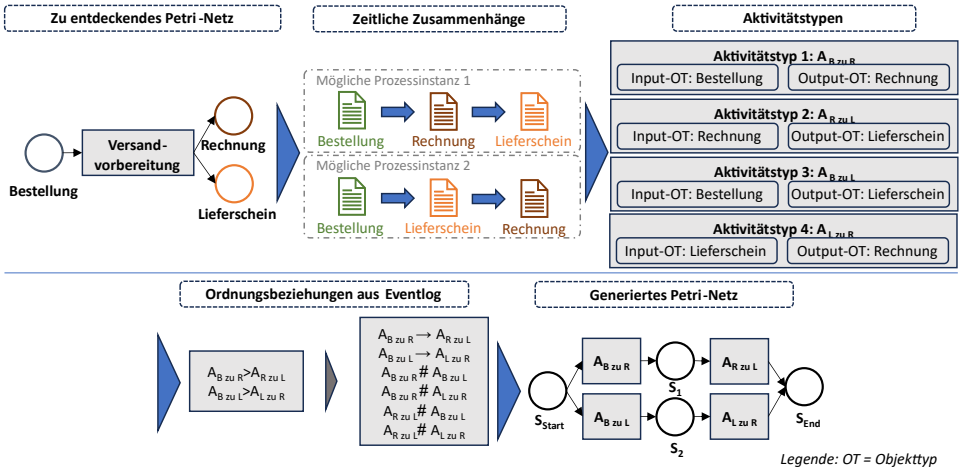


Abbildung 6-23: Generierung eines Geschäftsprozessmodells für eine AND-Aufspaltung

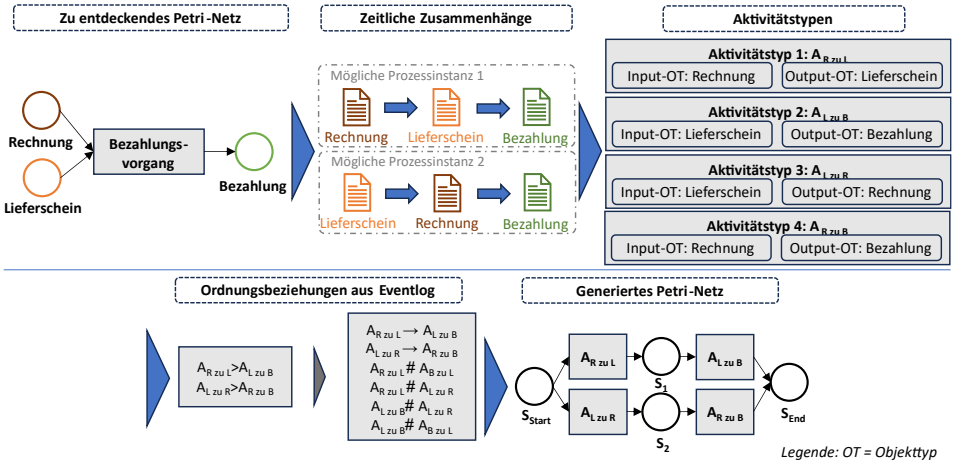


Abbildung 6-24: Generierung eines Geschäftsprozessmodells für eine AND-Zusammenführung

Das zeigt, dass ohne Anpassung nur Kausalitäten und Unabhängigkeiten abgeleitet werden, da inverse Ordnungsbeziehungen durch den Zwischenschritt der Aktivitätstypengenerierung nicht erkannt werden können. Diese inversen Ordnungsbeziehungen sind jedoch beim α-Algorithmus entscheidend, um eine Nebenläufigkeit zu entdecken. Außerdem zeigt sich, dass die Muster der generierten Geschäftsprozessmodelle bei AND-Aufspaltung und Zusammenführung identisch sind. Diese können unterschieden werden,

wenn die Objekttypen zu den Stellen ergänzt und Stellen mit gleichen Objekttypen zusammengeführt werden. Dies ist in Abbildung 6-25 dargestellt.

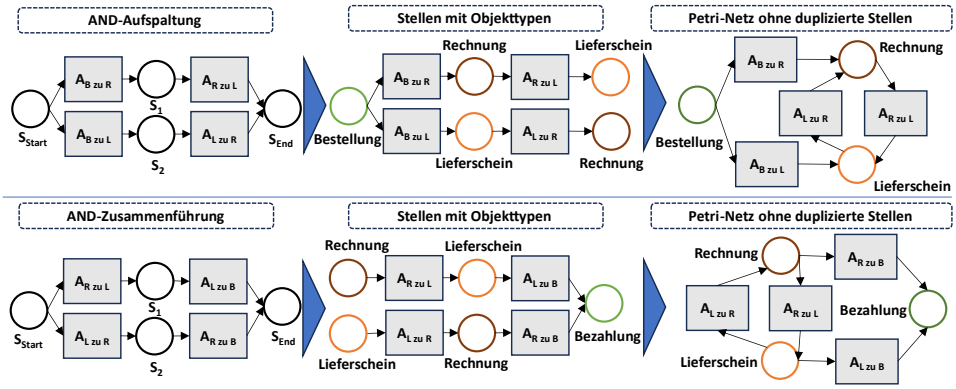


Abbildung 6-25: Zuordnung der Objekttypen zu Stellen sowie Zusammenführung von Stellen mit identischen Objekttyp

Die Analyse der Objektinstanzen nach dem Prinzip des α-Algorithmus für Nebenläufigkeit zeigt am Beispiel in Abbildung 6-24 zwei mögliche Reihenfolgen: Entweder folgt die Rechnung direkt auf die Bestellung und vor dem Lieferschein, oder der Lieferschein erscheint nach der Bestellung und vor der Rechnung. Da in beiden Reihenfolgen sowohl Rechnung als auch Lieferschein der Bestellung nachfolgen, sollte eine Transition modelliert werden, die die Stelle für Bestellung im Vorbereich und die Stellen für Rechnung und Lieferschein im Nachbereich hat.

Diese Schlussfolgerung lässt sich auch durch die Analyse der zeitlichen Zusammenhänge der Objektinstanzen statt der Aktivitätstypen ableiten. Bei der Untersuchung der Ordnungsbeziehungen zwischen Objektinstanzen indiziert das Auftreten inverser Ordnungsbeziehungen das Fehlen eines eindeutigen zeitlichen Zusammenhangs zwischen zwei Objekttypen. In solchen Fällen ist eine Zusammenführung der betreffenden Aktivitätstypen erforderlich.

Seien O_A und O_B die Objekttypen, bei denen ein inverse Ordnungsbeziehung nach Definition 6-8 erkannt wurden, werden die Aktivitätstypen nach folgenden Regeln zusammengeführt:

- Zusammenführung von Aktivitätstypen die O_A oder O_B als Input sowie denselben Output haben.
- Zusammenführung von Aktivitätstypen die O_A oder O_B als Output sowie denselben Input haben.
- Entfernung von Aktivitätstypen die nur O_A oder O_B als Input / Output haben.

Dies ist auch in Abbildung 6-26 dargestellt.

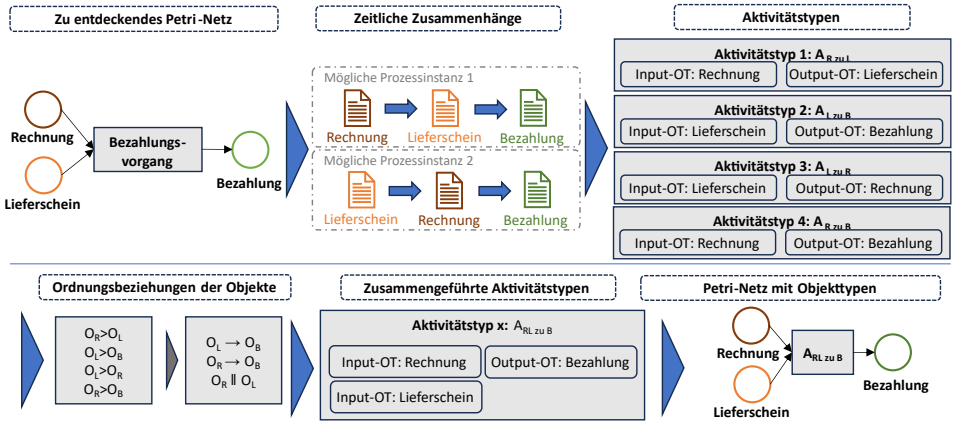


Abbildung 6-26: Zusammenführung von Aktivitätstypen basierend auf inversen Ordnungsbeziehungen zwischen Objekttypen ($O_R \parallel O_L$)

Basierend auf einer Mustererkennung und Ersetzen der Muster aus Abbildung 6-25 durch eine Transition mit entsprechenden Kanten oder durch die Analyse der zeitlichen Zusammenhänge der Objektinstanzen können AND-Aufspaltungen und AND-Zusammenführungen erkannt und erstellt werden.

6.2.2 Ordnungsbeziehungen aus inhaltlichen Zusammenhängen

Um zur Generierung auch Objektinstanzen (bzw. deren Objekttypen) ohne Zeitpunkt zu berücksichtigen, werden in diesem Abschnitt die Aktivitätstypen aus den inhaltlichen Zusammenhängen berücksichtigt. Werden die Aktivitätstypen aus verschiedenen Zusammenhängen abgeleitet, können Redundanzen auftreten. Daher sind alle identifizierbaren Aktivitätstypen darauf zu überprüfen, ob nur eine minimale Menge an Aktivitätstypen definiert wurde, die alle beobachteten Manipulationen (aufgedeckt durch Möglichkeit 1 in R2) sowie zeitlichen Zusammenhänge (aufgedeckt durch Möglichkeit 2 in R2) abdecken. Die inhaltlichen Zusammenhänge werden dabei zunächst fokussiert, wobei die zeitlichen Zusammenhänge eingehalten werden sollen. Das Ergebnis dieses Schritts sind Geschäftsprozessmodelle mit den abgebildeten Aktivitätstypen, bei denen Input und Output sowie ggf. die Manipulationen sowie der Abbildung der Ordnung dieser Aktivitätstypen definiert sind.

Anschließend werden die Aktivitätstypen, die aus den inhaltlichen Zusammenhängen abgeleitet wurden, anhand überschneidender Objekttypen zu den in Abbildung 6-27

dargestellten Kontrollflüssen zusammengesetzt. Da die Aktivitätstypen, die innerhalb einer Prozessinstanz auftreten und zusammen ausgeführt werden könnten, bereits in Abschnitt 6.1.5 zusammengeführt wurden, sind Aktivitätstypen, die gemeinsame Objekttypen als Input und/oder Output haben, zunächst als Alternativen zu modellieren.

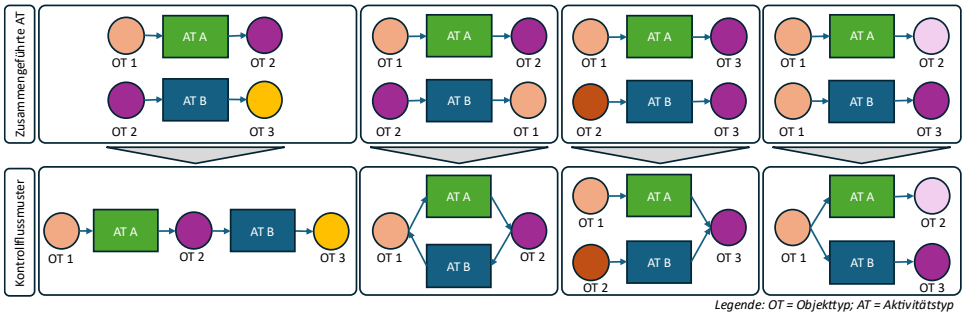


Abbildung 6-27: Zusammenführung von zusammenhängenden Aktivitätstypen zu Kontrollflussmustern

Daraus können jedoch auch Petri-Netze generiert werden, die nicht zusammenhängend sind, was in Abschnitt 5.2 eine Anforderung darstellt (Anforderung AM01g,), oder nicht die Eigenschaft der Workflow-Netze erfüllen. Existiert beispielsweise ein Teilnetz[7], das nur über einen Objekttyp (1) oder einen Aktivitätstyp (2) mit dem restlichen Petri-Netz zusammenhängt, erfüllt das Netz nicht die Eigenschaft (iii) der Definition 3-2 für Workflow-Netze, da es nur schwach zusammenhängend ist. Außerdem können auch nichtzusammenhängende Petri-Netze (ggf. mehrere Netzteile eines Netztes) generiert werden, wenn Aktivitätstypen bzw. Objekttypen keine gegenseitigen Zusammenhänge haben (3). Diese drei Fälle sind in Abbildung 6-28 dargestellt.

[7] Ein Netz N' = (S', T', F') heißt Teilnetz von einem Netz N = (S, T, F), falls gilt: (1) S' ⊆ S, (2) T' ⊆ T sowie (3) F' = F ∩ ((S'×T') ∪ (T'×S')). Gilt in (3) nur F' ⊂ F ∩ ((S'×T') ∪ (T'×S')), so heißt N' Netzteil von N [Star90, Reis86].

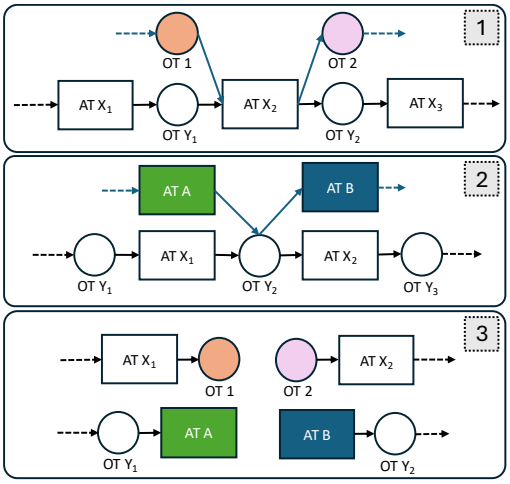


Abbildung 6-28: Generierte Petri-Netzteile, die nicht die Eigenschaft von Workflow-Netzen erfüllen[8]

Daher sollen weitere Zusammenführungen bzw. Netzerweiterungen basierend auf einer Betrachtung der zeitlichen Zusammenhänge durchgeführt werden. Kann aus den zeitlichen Zusammenhängen kein zusammenhängendes Petri-Netz generiert werden, könnten Objekt- oder Aktivitätstypen ausgeschlossen, in Abschnitt 6.1.5 ausgeschlossene Aktivitätstypen hinzugefügt, unterschiedliche Zusammenführungen vorgeschlagen oder weitere Daten miteinbezogen werden. Daher können folgende Fälle unterschieden werden:

- Verringern der minimal zu erreichenden Bewertung: Bei der Auswahl der relevanten Aktivitätstypen kann ein minimaler Wert angegeben werden, mit dem die Aktivitätstypen bewertet sein müssen, um als relevant gekennzeichnet zu werden. Wird dieser verringert, werden weitere Aktivitätstypen hinzugefügt, wenn sie neue Objekttypen als Input oder Output haben. Dadurch könnte das Petri-Netz zusammenhängend werden.
- Einbeziehen zeitlicher Zusammenhänge: Wenn nicht-zusammenhängende Petri-Netze generiert wurden, enthalten die Objekttypen der unterschiedlichen nicht-zusammenhängenden Petri-Netze keine als relevant definierten inhaltlichen Zusammenhänge. In Abschnitt 6.1.5 wurden aus den zeitlichen Zusammenhängen Aktivitätsinstanzen abgeleitet, außerdem wurden in Abschnitt 6.2.1 die Ordnungsbeziehungen der Objekttypen abgeleitet sowie ein entsprechendes Geschäftspro-

[8] Die Abbildung zeigt verschiedene Netze. Jedes ist so zu verstehen. dass es über die gestrichelten Pfeile in ein größeres Netz *N* eingebettet sein kann. Dabei hat das Netz *N* in (1) und (2) keine weiteren gemeinsamen Stellen oder Transitionen und in (3) bleiben die Netze nicht-zusammenhängend.

zessmodell generiert. Diese Aspekte können daher zusätzlich berücksichtigt werden. Werden dabei weitere Zusammenführungen aufgedeckt, sind die Netzteile zu einem Petri-Netz zusammenzuführen.

- Zusätzliche Möglichkeiten: Können keine weiteren Zusammenführungen aufgedeckt werden, bestehen verschiedene Möglichkeiten:
    - Ausschließen von Objekttypen und Aktivitätstypen: Objekttypen, die keinen Zusammenhang (weder inhaltlich noch zeitlich) zu anderen Objekttypen aufweisen, sind für keinen Aktivitätstyp Input oder Output. Aktivitätstypen, die keinen gemeinsamen Input oder Output zu anderen Aktivitätstypen haben und auch nicht zeitlich eingeordnet werden können, weisen lediglich einen inhaltlichen Zusammenhang zwischen zwei oder mehreren Objekttypen auf, für die keine Daten-Attribute vorliegen. Diese Objekttypen und Aktivitätstypen könnten ausgeschlossen werden.
    - Alternative Vorschläge: Kann der Zusammenhang zwischen einzelnen Netzteilen nicht eindeutig abgeleitet werden, könnten verschiedene Möglichkeiten vorgeschlagen werden, indem Stellen ohne eingehende bzw. ausgehende Kanten der Netzteile analysiert und mögliche Zusammenführungen vorgenommen werden. Da mit steigender Anzahl an nicht-zusammenhängenden Petri-Netzen sowie Elementen die Anzahl der Alternativen steigt, ist die Anzahl an Vorschlägen zu begrenzen.
    - Weitere Daten: Werden weitere Daten zur Verfügung gestellt, können aus diesen Daten weitere Informationen zur Zusammenführung der nicht-zusammenhängenden Petri-Netze extrahiert werden. Dies wird in Abschnitt 6.3 betrachtet.

In Option 2 sollen die Netzteile zusammengeführt werden, indem die Ordnungsbeziehungen der Ränder[9] der einzelnen Netzteile überprüft werden. Dazu werden die Ordnungsbeziehungen der Objektinstanzen analysiert, die bereits in Abschnitt 6.2.1 abgeleitet wurden. Können basierend darauf Netzteile zusammengeführt werden, sollen diese zu den in Abbildung 6-29 dargestellten Mustern führen, bei denen Stellen, Transitionen und Kanten hinzugefügt werden. Dabei ist jedoch zu beachten, dass nur Transitionen und Stellen miteinander verbunden werden dürfen. In der Abbildung sind die Stellen der zusammenzuführenden Aktivitätsmuster gestrichelt dargestellt, da diese Stellen auch fehlen können oder aus mehreren Stellen bestehen können. Demnach werden anhand gegebener Aufspaltungs- bzw. Zusammenführungsmustern Stellen und Transitionen ergänzt.

[9] Der Rand eines Teilnetz N' bzgl. des Gesamt-Netzes N sind diejenigen seiner Knoten, die über Kanten mit Knoten aus N verbunden sind: $rand(N', N) = \{x \in (S' \cup T') | (x\bullet \cup \bullet x) \setminus (S' \cup T') \neq \emptyset\}$ [Star90, Reis86]. (In dieser Definition bezieht sich der Nachbereich $x\bullet$ bzw. Vorbereich $\bullet x$ auf das Gesamt-Netz N).

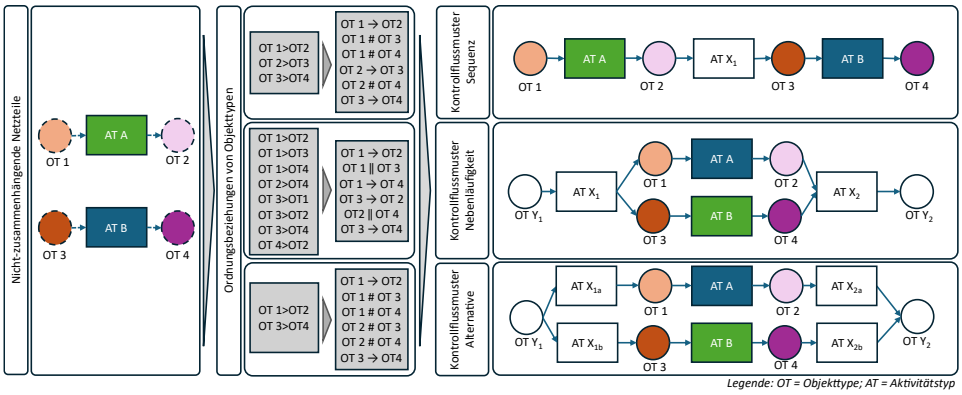


Abbildung 6-29: Zusammenführung von nicht-zusammenhängenden Netzteilen zu einem Netz

Anhand der Zusammenhänge zwischen den Objekttypen über Aktivitätstypen hinweg, kann geprüft werden, ob weitere Aktivitätstypen zusammengeführt werden sollten. Zunächst sind die Aktivitätstypen zu betrachten, die in Abschnitt 6.1.5 noch nicht zusammengeführt wurden (Fall 2b). Dort treten die Aktivitätsinstanzen der Aktivitätstypen in den gleichen Prozessinstanzen auf, ohne jedoch, dass die Objektinstanzen zu denselben Zeitpunkten sind. Daher konnte nicht differenziert werden, ob die Objektinstanzen desselben Objekttyps zusammen oder durch verschiedene Aktivitätstypen erzeugt werden. Da die Aktivitätstypen in denselben Prozessinstanzen auftreten, müssen beide Aktivitätstypen in dem Geschäftsprozessmodell in einer Prozessinstanz ausführbar sein. Daher bestehen drei Möglichkeiten (siehe Abbildung 6-30):

- Zusammenführung der Aktivitätstypen und somit Erzeugen beider Objektinstanzen bei Ausführung des zusammengeführten Aktivitätstyps.
- Durch eine AND-Aufspaltung vor den Aktivitätstypen können beide Aktivitätstypen in einer Prozessinstanz ausgeführt werden.
- Durch einen Zyklus können ebenfalls beide Aktivitätstypen ausgeführt werden. Ohne entsprechende Transitionsinschrift, ist jedoch nicht vorgegeben, welche Aktivität wann ausgeführt wird, wodurch auch zweimal dieselbe Aktivität ausgeführt werden könnte. Daher sollten zusätzliche Regeln definiert werden (Abschnitt 6.2.3).

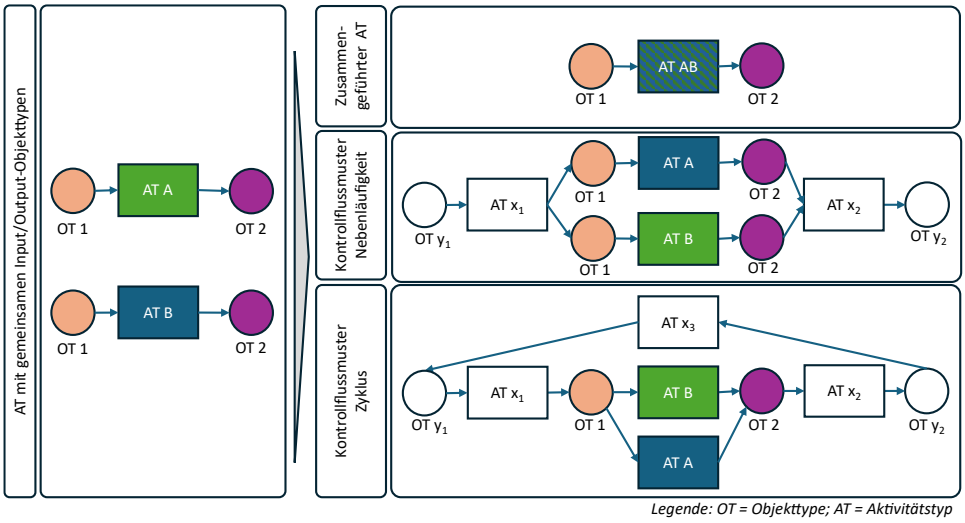


Abbildung 6-30: Überprüfung von Aktivitätstypen mit gemeinsamen Input/Output-Objekttypen

Die Aufspaltungen zu den nebenläufig ausführbaren Aktivitätstypen oder zur Iteration müssen dabei nicht unmittelbar vor/nach den entsprechenden Aktivitätstypen erfolgen. Liegt kein Zyklus in dem Petri-Netz vor, ist das Petri-Netz so anzupassen, dass entweder eine AND-Aufspaltung die Nebenläufigkeit von den Aktivitätstypen A und B ermöglicht oder dass die Aktivitätstypen zusammengeführt werden. Dies ist anhand der zeitlichen Zusammenhänge zu prüfen. Sollte beispielsweise die Objektinstanz nicht unmittelbar vor oder nach der anderen Objektinstanz liegen und die frühere Objektinstanz vor Erzeugen der späteren Objektinstanz bereits wieder verbraucht werden oder Input für das Erzeugen der späteren Objektinstanz sein, ist ein Zyklus notwendig.

Sind exakt zwei Aktivitätstypen mithilfe von zusätzlichen Aktivitätstypen nebenläufig wie in Abbildung 6-31 abgebildet worden, können diese auch als zusammengeführte Aktivitätstyp dargestellt werden. Dadurch wird insbesondere die Elementzahl in einem Geschäftsprozessmodell verringert, was die Verständlichkeit erhöhen soll (siehe Abschnitt 2.3.3).

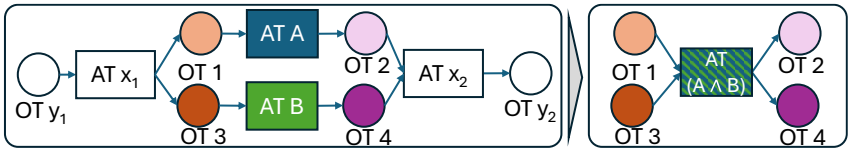


Abbildung 6-31: Vereinfachung von Nebenläufigkeit

6.2.3 Ausführungsbedingungen aus Prozessinstanzen

Die extrahierten und abgebildeten Informationen in 6.2.2 betreffen sowohl den Kontrollfluss als auch durchgeführte Manipulationen an den Objekten. Jedoch wurden noch keine Informationen abgebildet, die die Auswahl des Pfades an Entscheidungspunkten betreffen. Anhand des Kontrollflusses können diese Entscheidungspunkte identifiziert und weitere indirekte Zusammenhänge durch Entscheidungsbäume berechnet werden. Entscheidungspunkte sind Zeitpunkte, an denen eine Entscheidung getroffen wird, die sich auf die Art und Weise der Prozessausführung auswirkt. Dadurch sollen an alternativen Aktivitäten Regeln definiert werden, wann welche Aktivität ausgeführt werden soll. Dazu müssen anhand des Kontrollflusses zunächst Entscheidungspunkte innerhalb des Geschäftsprozessmodells identifiziert werden (XOR-Aufspaltungen). Darauf aufbauend werden die relevanten Informationen aus den betreffenden Prozessinstanzen extrahiert und aufbereitet. Die anschließende Analyse der Unterschiede in den Prozessinstanzen vor und nach den Entscheidungspunkten ermöglicht die Ableitung spezifischer Entscheidungsregeln. Dies wird in [RoAa06] als Decision Mining benannt und in Abbildung 6-32 dargestellt.

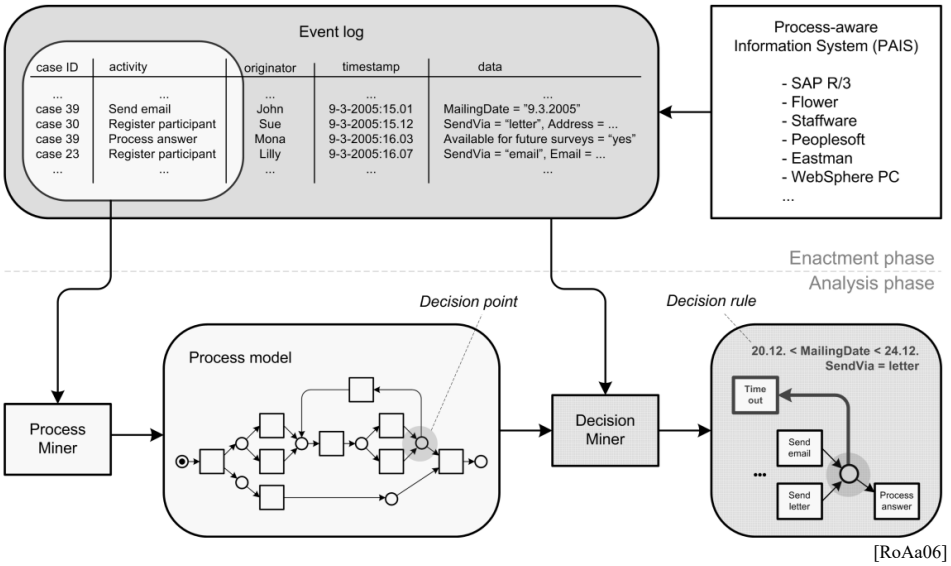


Abbildung 6-32: Decision Mining

Die Regeldefinition wird in ein Klassifizierungsproblem umgewandelt, um Vorhersagen der Gruppenzugehörigkeit von Dateninstanzen abzuleiten. Bei einem Klassifizierungsproblem können beispielsweise Verfahren wie Entscheidungsbäume [RoAa06], Naive Bayes [FrGG97], K-Nearest Neighbour [LaOj96] und Support Vector Machine [Nobl06]

angewandt werden. Bei einem *Entscheidungsbaum* wird basierend auf den Attributen versucht, eine Teilung des Datensatzes zu ermitteln. Eine Teilung wird so lange durchgeführt, bis keine weitere Teilung mehr möglich oder ein Abbruchkriterium erfüllt ist. Jeder der resultierenden Blattknoten des Baumes kann infolgedessen einer Klasse zugeordnet werden. Bei *Naive-Bayes-Klassifikatoren* wird die bedingte Wahrscheinlichkeit für jedes Attribut bestimmt, um anschließend durch Anwendung des Satzes von Bayes die Wahrscheinlichkeit einer Klasse für eine gegebene Prozessinstanz zu berechnen. Dabei sind diese jedoch insbesondere für stetige Attribute bzw. Attribute mit vielen Ausprägungen nicht geeignet. Bei der Klassifizierung nach dem *K-Nearest Neighbor* werden Trainingsinstanzen einer Klasse zugeordnet und als Punkte im Raum dargestellt. Soll anschließend eine unbekannte Instanz ebenfalls als Punkt klassifiziert werden, so werden dessen k nächste Nachbarpunkte bestimmt und die Klassenzugehörigkeit nach der Mehrheitsregel festgelegt. Durch die große Anzahl von Abstandsberechnungen ergibt sich eine große Rechenkomplexität. Bei *Support Vector Machines* werden die Trainingsinstanzen ebenfalls zunächst einer Klasse zugeordnet und als Punkte in einem hochdimensionalen Raum dargestellt. Anschließend wird eine Hyperebene gewählt, die die beiden Klassen trennt. Dabei wird die Ebene so gewählt, dass der Abstand zum nächstgelegenen Punkt der beiden Klassen maximal ist.

6.3 Unterstützung durch zusätzliche Daten

Wie in den einzelnen Schritten beschrieben, können verschiedene Herausforderungen in der Prozessmodellgenerierung auftreten. Dies kann dazu führen, dass der Kontrollfluss nicht eindeutig aus den Daten abgeleitet werden kann. Grund dafür können beispielsweise fehlende Daten oder über verschiedene Objektinstanzen widersprüchliche Informationen sein. Um die Modellqualität zu steigern, könnten daher zusätzliche Daten bereitgestellt werden und somit zusätzliche Informationen zur Prozessmodellgenerierung extrahiert werden.

Für die Modellierung des Kontrollflusses basierend auf den Objektinstanzen wird insbesondere die Reihenfolge der Objektinstanzen analysiert. Diese wird aus zeitlichen und inhaltlichen Zusammenhängen abgeleitet (siehe Abschnitt 6.2). Dabei treten jedoch Herausforderungen auf, wenn beispielsweise Objektinstanzen keinen repräsentativen Zeitpunkt haben oder redundante Manipulationen aus den inhaltlichen Zusammenhängen abgeleitet werden können. Zusätzlich kann auch nur die Existenz eines Objektes (beispielsweise ein angefordertes Produkt ist vorhanden) oder ein Wert eines Attributes (beispielsweise Abholung=*true*) für eine Aktivität relevant sein, welches nicht zu direkten Objektbeziehungstypen führen könnte, falls beispielsweise keine Korrelationen, beispielsweise aufgrund der Datenmenge, abgeleitet werden können. Eine hilfreiche Zusatzinformation wäre die Reihenfolge der Objekte bereits zusätzlich als Inputdaten

bereitzustellen. Dies könnte zum einen bereits mit den bereitgestellten Objektinstanzen erfolgen. Beispielsweise könnte für einzelne Objektinstanzen eine Reihenfolge in der Form *Bestellung* → *Rechnung* → *Bezahlung* ergänzt werden (*Erweiterungsmöglichkeit 1a*). Es könnten auch zuerst die Objekttypen abgeleitet und anschließend vom Anwender Informationen zu deren Reihenfolge erfragt werden (*Erweiterungsmöglichkeit 1b*). In Möglichkeit 1a sollte sichergestellt werden, dass die angegebene Reihenfolge nur Objekte betrifft, die der Nutzer auch bereitstellt. In Möglichkeit 1b könnte dem Anwender bereits eine Reihenfolge vorgeschlagen werden, die anschließend vom Anwender angepasst werden kann. Dafür ist zu beachten, ob der Anwender nur Sequenzen der Dokumente oder auch nebenläufige Pfade angeben kann. Außerdem ist zu bewerten, wie mit Widersprüchen umgegangen wird, wenn beispielsweise die angegebene Reihenfolge nicht mit der beobachteten Reihenfolge der Objekte übereinstimmt. Zusätzlich könnte der Anwender ggf. auch nur zu einem Teil der Objekte die Reihenfolge wissen. Daher sollte das Verfahren trotz der Ergänzung wie in Abschnitt 6.2 mit fehlenden Informationen umgehen.

Eine weitere Möglichkeit wäre, bereits vorhandene weitere Informationen zum Kontrollfluss zu verwenden. So könnten bereits vorhandene Geschäftsprozessmodelle (beispielsweise in Form eines Petri-Netzes) als zusätzliche Informationsquelle genutzt werden (*Erweiterungsmöglichkeit 2*). Dadurch wird bestehendes dokumentiertes Prozesswissen nutzbar und mit den Daten zu Objektinstanzen ergänzt. Dabei ist insbesondere die Identifikation und Beseitigung von Inkonsistenzen herausfordernd, da die Aktivitäten des bereitgestellten Modells zunächst mit den abgeleiteten Aktivitätstypen aus den Objekten verknüpft werden müssen. Liegt das Geschäftsprozessmodell nicht in Form eines Petri-Netzes vor, ist der Umgang mit unterschiedlichen Modellnotationen und deren Zusammenführung zu definieren. Die Geschäftsprozessmodelle könnten erst in Petri-Netze transformiert werden oder die Ordnungsbeziehungen der Aktivitäten extrahiert und als Informationen in der Transformation wie in Abschnitt 6.2.2 berücksichtigt werden.

Die vorgestellte Methode könnte durch die Integration bereits existierender Eventlogs erweitert werden (*Erweiterungsmöglichkeit 3*). Da aus den zeitlichen Zusammenhängen (Abschnitt 6.2.1) bereits ein Eventlog generiert wird (Abschnitt 6.2.1) und dies für die inhaltlichen Zusammenhängen ebenfalls möglich wäre, könnten zusätzliche Eventlog-Daten als weitere Informationsquelle in die Modellgenerierung einfließen. Eine erfolgreiche Integration würde jedoch voraussetzen, dass die Eventlog-Daten eine ausreichende Qualität aufweisen und auf einer vergleichbaren Abstraktionsebene wie die anderen Informationsquellen vorliegen. Dies ist entscheidend, da Diskrepanzen zwischen den Abstraktionsgraden der verschiedenen Eventlogs und der daraus abgeleiteten Modelle die Generierung eines Geschäftsprozessmodells erschweren könnten.

In dieser Arbeit werden zu Vorverarbeitung und Informationsextraktion bereits NLP-Ansätze angewandt. Daher könnten diese auch zur Extraktion von Prozessinformationen aus unstrukturierten Textdokumenten angepasst werden [FoSc24]. Dadurch werden textuelle Informationen für die Modellierung nutzbar, was diese Methode insbesondere für Anwender ohne technisches Vorwissen nutzbar macht (*Erweiterungsmöglichkeit 4*). Aufgrund der eigenen Erstellungsmöglichkeit des Textes könnte direkt Bezug zu den Daten genommen werden. Jedoch führen insbesondere die Mehrdeutigkeit und Variabilität natürlicher Sprache zu Herausforderungen.

Die verschiedenen Erweiterungsansätze sollten als ergänzende Elemente zur Analyse der Objektinstanzen eingesetzt werden, um Synergieeffekte zu nutzen und eine durchgängig objektbasierte Generierung zu gewährleisten.

6.4 Zusammenfassung

Basierend auf den Objektinstanzen, die in den Ausführungen des Geschäftsprozesstyps (den Prozessinstanzen) aufgetreten sind, soll der Geschäftsprozesstyp abgeleitet werden. Dazu werden Dateien, die Daten zu den Objektinstanzen enthalten, vom Anwendenden bereitgestellt. Eine Datei bildet den Zustand einer oder mehrerer Objektinstanz(en) eines Geschäftsprozesses zu einem bestimmten Zeitpunkt in den Rohdaten ab. Durch Aktivitäten wird der Zustand einer Prozessinstanz verändert. Sofern dabei auch die Eigenschaften der Objektinstanzen verändert werden, lässt sich die Durchführung der Aktivität und die Änderung des Zustandes anhand der beteiligten Objektinstanzen beobachten. Daher werden durch die Betrachtung der Objektinstanzen Rückschlüsse auf die Aktivitätstypen und deren Ordnungsbeziehungen gezogen und anschließend durch ein Geschäftsprozessmodell repräsentiert.

Die in diesem Kapitel beschriebene Methode zur Generierung von höheren Petri-Netzen aus Objektinstanzen folgt einem zweistufigen Vorgehen: *Informationsextraktion* und *Transformation*.

Um aus den bereitgestellten Dateien ein Geschäftsprozessmodell generieren zu können, werden zunächst die relevanten *Informationen* für den Geschäftsprozesstyp aus den Rohdaten der Dateien zu den Objektinstanzen *extrahiert*. Dazu werden die Dateien zunächst Prozessinstanzen zugeordnet, anschließend werden Objekttypen klassifiziert und schließlich aus den inhaltlichen und zeitlichen Zusammenhängen der Objektinstanzen Aktivitätstypen abgeleitet.

Die extrahierten Informationen zu Objekttypen und Aktivitätstypen werden in der *Transformation* in die Elemente eines höheren Petri-Netzes transformiert. Die Objekttypen

werden durch Stellen, die Aktivitätstypen durch Transitionen und der Kontrollfluss durch Kanten eines Petri-Netzes dargestellt. Zusätzliche werden die Objektstruktur der Objekttypen und die Manipulationen bzw. Regeln der Aktivitätstypen abgebildet. Dabei werden in Schritt 1 zunächst die Aktivitätstypen aus den zeitlichen Zusammenhängen abgebildet und in Schritt 2 zusätzlich die Aktivitätstypen aus den inhaltlichen Zusammenhängen berücksichtigt und daher auch Manipulationsregeln im Geschäftsprozessmodell definiert. Im dritten Schritt werden weitere Ausführungsbedingungen integriert.

Die einzelnen Schritte, die sich aus der Betrachtung der Informationsextraktion und der Transformation in ein Modell mit den jeweiligen Ergebnissen ergeben sind in Tabelle 6-6 zusammengefasst.

Tabelle 6-6: Schritte der Prozessmodellgenerierung inklusive des jeweiligen Inputs und Outputs

	Schritt	Input	Output
Informationsextraktion	Zuordnung von Objektinstanzen zu Prozessinstanzen sowie Finden der prozessinstanzunabhängigen Objektinstanzen.	Objektinstanzen als Attribut-Wert-Paare	Prozessinstanzen mit Referenzen zu Objektinstanzen
	Analyse von Datenstrukturen der Objektinstanzen, um diese zu Typen zu klassifizieren.	Objektinstanzen als Attribut-Wert-Paare	Typen mit Referenzen zu Objektinstanzen
	Ableitung der Objekttypen aus den klassifizierten Objektinstanzen.	Typen mit Referenzen zu Objektinstanzen	Objekttypen mit Schema und Benennung sowie Referenzen zu Objektinstanzen
	Festlegen von repräsentativen Zeitpunkten aus allen vorkommenden Daten-Attributen sowie der Metadaten jeder Objektinstanz.	Objektinstanzen	Objektinstanzen mit repräsentativem Zeitpunkt
	Ableiten von Objektbeziehungstypen aus Überschneidungen in Attribut-Wert-Paaren sowie zeitlichen Auftreten der Objektinstanzen innerhalb der Prozessinstanzen.	Objekttypen, Prozessinstanzen	Objektbeziehungstypen aus zeitlichen und inhaltlichen Zusammenhängen

	Schritt	Input	Output
	Definieren von Aktivitätsinstanzen aus den Objektbeziehungstypen.	Objektbeziehungstypen	Aktivitätsinstanzen mit Bewertung für Input- / Output-Objekttypen
	Ableiten der Aktivitätstypen aus den Aktivitätsinstanzen mit festgelegtem Input- und Output und Manipulationen.	Aktivitätsinstanzen	Aktivitätstypen inklusive zugeordneter Aktivitätsinstanzen
	Überprüfen der Aktivitätstypen auf Redundanzen und Zusammenführungsmöglichkeiten.	Aktivitätstypen	Bereinigte Aktivitätstypen
Transformation	Eventloggenerierung aus den Aktivitätsinstanzen der Aktivitätstypen aus den zeitlichen Zusammenhängen.	Aktivitätstypen aus zeitlichen Zusammenhängen	Eventlog mit Traces zu den Prozessinstanzen mit durchgeführten Aktivitätstypen inklusive Zeitstempel
	Zusammenführung von Aktivitätstypen basierend auf den Ordnungsbeziehungen zwischen Objektinstanzen.	Objektinstanzen, Eventlog	Bereinigter Eventlog
	Anwendung eines Process Discovery Algorithmus sowie Objekttyp-Zuordnung zu Stellen basierend auf den Aktivitätstypen.	Bereinigter Eventlog, Objekttypen	Petri-Netz mit zugeordneten Objekttypen zu den Stellen
	Zusammenführung von zusammenhängenden Aktivitätstypen.	Aktivitätstypen aus inhaltlichen Zusammenhängen	Petri-Netz(-teile) mit Manipulationen
	Betrachtung der Ordnungsbeziehungen zwischen Objektinstanzen, um Petri-Netzteile zusammenzuführen.	Petri-Netzteile, Objektinstanzen	Petri-Netz, das die Workflow-Eigenschaft erfüllt
	Objekttyp-Zuordnung zu Stellen basierend auf den Aktivitätstypen sowie Betrachtung der Aktivitätstypen die gemeinsame Objekttypen haben, um weitere Zusammenführungen zu überprüfen.	Objekttypen, Aktivitätstypen, Petri-Netz	Petri-Netz mit zugeordneten Objekttypen zu den Stellen

Schritt	Input	Output
Entscheidungspunkte im Petri-Netz identifizieren und Entscheidungsregeln anhand der Prozessinstanzen ableiten und als Ausführungsbedingungen in den Aktivitäten ergänzen.	Petri-Netz, Prozessinstanzen	Petri-Netz mit Ausführungsbedingungen

Die Generierung des Petri-Netzes aus Objektinstanzen kann durch fehlende oder widersprüchliche Informationen in der Qualität beeinträchtigt werden, insbesondere bei der Ableitung des Kontrollflusses. Zur Verbesserung wurden daher auch verschiedene Erweiterungsmöglichkeiten identifiziert. Diese umfassen die explizite Angabe einer Reihenfolge für Aktivitäten oder Objekte, entweder direkt bei der Dateieneingabe oder durch interaktive Validierung automatischer Vorschläge. Auch könnten bereits existierende Geschäftsprozessmodelle integriert werden. Zusätzlich ist die Ergänzung durch Eventlogs möglich, wobei hier die Datenverfügbarkeit und unterschiedliche Abstraktionsgrade zu beachten sind. Eine weitere Option bietet die Erweiterung der bereits implementierten NLP-Verarbeitung zur Extraktion von Prozessinformationen aus unstrukturierten Textdokumenten, was besonders benutzungsfreundlich für Anwender ohne technisches Vorwissen ist.

7 Software-Prototyp zur Prozessmodellgenerierung

In diesem Kapitel wird die Entwicklung eines Software-Prototyps zur automatisierten Generierung von Geschäftsprozessmodellen beschrieben. Der Software-Prototyp implementiert die im Kapitel 6 vorgestellte Methode zur Generierung von Geschäftsprozessmodellen aus Objektinstanzen. Er wird in der in Kapitel 8 beschriebenen Evaluation eingesetzt, um die Funktionsfähigkeit und Qualität der generierten Modelle zu bewerten.

In Abschnitt 7.1wird zunächst die Architektur des Prototyps vorgestellt. Anschließend werden in Abschnitt 7.2 die technischen Details der Implementierung beschrieben: die Informationsextraktion zur Ableitung der Objekttypen und Aktivitätstypen (Abschnitt 7.2.1) sowie die Transformation der extrahierten Informationen in ein Geschäftsprozessmodell in Form eines Petri-Netzes (7.2.2).

Die Implementierung fokussiert sich dabei auf die technische Umsetzung der zuvor entwickelten Methode und stellt die notwendigen Mechanismen bereit, um basierend auf Objektinstanzen Geschäftsprozessmodelle zu generieren. Der Software-Prototyp erkennt Attribut-Wert-Übereinstimmungen zwischen Objektinstanzen sowie zeitliche Zusammenhänge, um Aktivitätstypen und deren Ordnungsbeziehungen abzuleiten. Dabei wurde die Methode aus Kapitel 6 implementiert, die sowohl inhaltliche als auch zeitliche Zusammenhänge zwischen den Objektinstanzen in den Prozessausführungen berücksichtigt. Der modulare Aufbau ermöglicht eine flexible Erweiterung und Anpassung der einzelnen Komponenten. Die Implementierung basiert auf Python 3.10, da diese Programmiersprache sowohl objektorientierte Softwareentwicklungskonzepte unterstützt als auch Zugriff auf Bibliotheken für Datenanalyse und Process Mining bietet.[1]

7.1 Architektur

Die Implementierung folgt dem Model-View-Controller (MVC [GHJV95] Architekturmuster, wodurch eine klare Trennung zwischen Datenmodell (Model), Benutzungsoberfläche (View) und Steuerungslogik (Controller) erreicht wird. Die Architektur soll einen robusten und flexiblen Rahmen für die automatisierte Prozessmodellierung bieten. Die

[1] Der Software-Prototyp steht unter https://gitlab.kit.edu/kit/aifb/BIS/kit-bis/objektbasiertemodellgenerierung öffentlich und MIT-lizensiert bereit.

Trennung der Verantwortlichkeiten und die Verwendung standardisierter Schnittstellen gewährleisten Wartbarkeit und Erweiterbarkeit.

Der Prozessmodellgenerator gliedert sich in drei Schritte: die Datenvorverarbeitung, die Informationsextraktion und die Transformation in das Geschäftsprozessmodell. Abbildung 7-1 zeigt die drei MVC-Komponenten des Prozessmodellgenerators:

- Die *View-Komponente* stellt die Benutzungsoberfläche bereit und ermöglicht die Interaktion mit dem System.
- Der *Controller* übernimmt die Verarbeitungslogik und steuert den Datenfluss zwischen View und Model.
- Das *Model* repräsentiert die Datenstrukturen gemäß den Definitionen aus Kapitel 6.

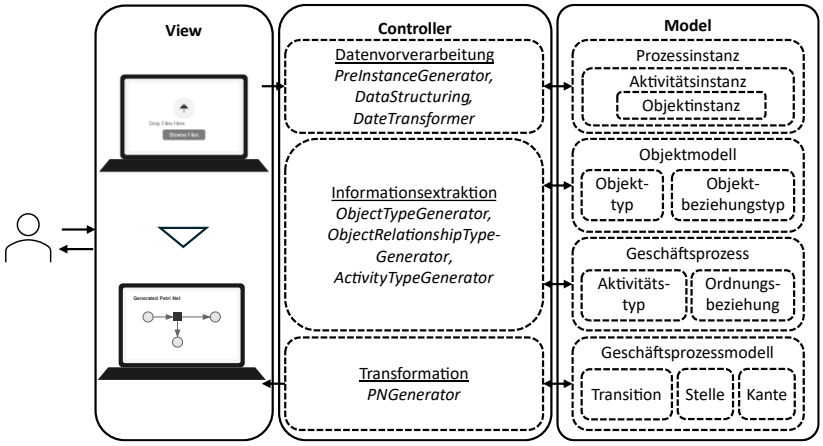


Abbildung 7-1: Model-View-Controller Architektur des Prozessmodellgenerators

Die Model-Komponente implementiert die wesentlichen Datenstrukturen. Diese umfassen die Prozessinstanzen inklusive der Objektinstanzen, die Objekttypen und Objektbeziehungstypen, die Aktivitätstypen und deren Ordnungsbeziehungen sowie das generierte Geschäftsprozessmodell in Form eines Petri-Netzes.

Die Controller-Komponenten implementieren die Verarbeitungslogik. Der *PreInstance-Generator* übernimmt die initiale Datenvorverarbeitung und greift auf den *DataStructuring* und den *DateTransformer* zu, die für die Datenerstellung und die Vereinheitli-

chung der Daten zuständig sind. Das Ergebnis ist ein Dataframe[2], das die einzelnen Objektinstanzen zu einem konkreten Zeitpunkt mit den Attribut-Wert-Paaren sowie den Metadaten der Datei, die die Objektinstanz repräsentiert, enthält. Die Informationsextraktion wird durch spezialisierte Generator-Klassen realisiert: Der *ObjectTypeGenerator* klassifiziert die verschiedenen Objekttypen und leitet deren Schema aus den entsprechenden Objektinstanzen ab. Der *ProcessInstanceClassifier* weist die einzelnen Objektinstanzen den Prozessinstanzen zu. Der *ObjectRelatioshipTypGenerator* analysiert die Beziehungen zwischen den Objekttypen und der *ActivityTypeGenerator* leitet daraus die Aktivitätstypen ab. Der Transformations-Schritt im *PNGenerator* generiert aus den extrahierten Informationen das Geschäftsprozessmodell. Dabei werden sowohl zeitliche als auch inhaltliche Zusammenhänge berücksichtigt und in einem Petri-Netz abgebildet. Der Datenfluss zwischen den Komponenten erfolgt über definierte Schnittstellen.

Die View-Komponente stellt die Benutzerinteraktion sicher und bietet verschiedene Visualisierungsmöglichkeiten für die generierten Geschäftsprozessmodelle. Dabei werden die Ergebnisse der einzelnen Schritte zur Unterstützung der Nachvollziehbarkeit bereitgestellt. Die Integration einer detaillierten Terminal-Ausgabe unterstützt zudem die Entwicklung und Wartung des Systems.

Die Implementierung stützt sich auf verschiedene Python-Bibliotheken. *Pandas* wird für die Datenverarbeitung und -analyse eingesetzt, *NetworkX* ermöglicht Graph-basierte Analysen, und *PM4Py* stellt Process Mining-Funktionalitäten bereit. *Genson* wird für die JSON-Schema-Generierung und -Validierung verwendet. Die *Natural Language Toolkit* (NLTK) Bibliothek unterstützt die Textvorverarbeitung durch Stopword-Filterung und linguistische Analysen. Für die Objekttypklassifikation wird die *scikit-learn* Bibliothek mit *TfidfVectorizer* und *KMeans-Clustering* verwendet. *Matplotlib* dient der Visualisierung von Analyseergebnissen und Statistiken.

Die Integration etablierter Bibliotheken ermöglicht eine effiziente Implementierung der Analysefunktionalitäten, während die modulare Struktur die einfache Erweiterung um neue Funktionen gewährleistet. Beispielsweise können weitere Dateiformate durch eine Erweiterung der Datenvorverarbeitung integriert werden. Ebenso können zusätzliche Analysetechniken in die Informationsextraktion eingebunden sowie die Transformation um weitere Modellierungssprachen ergänzt werden. Durch die Verwendung von Konfigurationsdateien für Schwellwerte und Gewichtungsfaktoren ist die Implementierung zudem flexibel an verschiedene Anwendungskontexte anpassbar. Die Architektur unter-

[2] Ein Dataframe ist eine zentrale Datenstruktur für die Verarbeitung tabellarischer Daten. Dataframes werden insbesondere durch die Python-Bibliothek pandas [McK10] [McKi10] unterstützt. Ein Dataframe organisiert Daten in einer zweidimensionalen Tabellenstruktur mit bezeichneten Spalten und Zeilen, ähnlich einer Excel-Tabelle oder einer SQL-Datenbank-Tabelle.

stützt auch die Protokollierung der Verarbeitungsschritte und ermöglicht so eine transparente Nachvollziehbarkeit der Prozessmodellgenerierung.

7.2 Implementierung

Die nachfolgende Beschreibung der Implementierung orientiert sich an der beschriebenen Architektur und erläutert die konkrete Umsetzung der einzelnen Controller-Komponenten sowie deren Zusammenspiel. Wie im Anwendungsfall in Kapitel 5 beschrieben, soll der Anwender zunächst die Dateien auswählen, die die Objektinstanzen des zu modellierenden Geschäftsprozesses repräsentieren.

Die Startseite des Prozessmodellgenerators (siehe Abbildung 7-2) bietet dem Anwender zunächst eine Übersicht über die Prozessmodellgenerierung. Es werden zentrale Begriffe erläutert sowie ein Überblick über die Funktionsweise der Modellgenerierung gegeben. Der Anwender kann bei der Prozessmodellgenerator zwischen zwei Vorgehensweisen wählen:

- Die *automatisierte Vorgehensweise* führt die Prozessmodellgenerierung ohne weitere Benutzerinteraktion durch. Diese Vorgehensweise wird in den folgenden Abschnitten detailliert beschrieben.
- Die *iterative Vorgehensweise* ermöglicht Interaktionen während des Generierungsprozesses. Diese Vorgehensweise wird in Abschnitt 7.3.1 erläutert.

Über die Fortschrittsanzeige erhält der Anwender Rückmeldung über den aktuellen Generierungsschritt sowie über den Fortschritt der Generierung. Im Reiter *Protokoll* werden diese Generierungsschritte zusammengefasst. Zusätzlich kann der Anwender durch Aktivieren des Detail-Modus eine umfassendere Protokollierung aller Generierungsschritte in diesem Reiter erhalten.

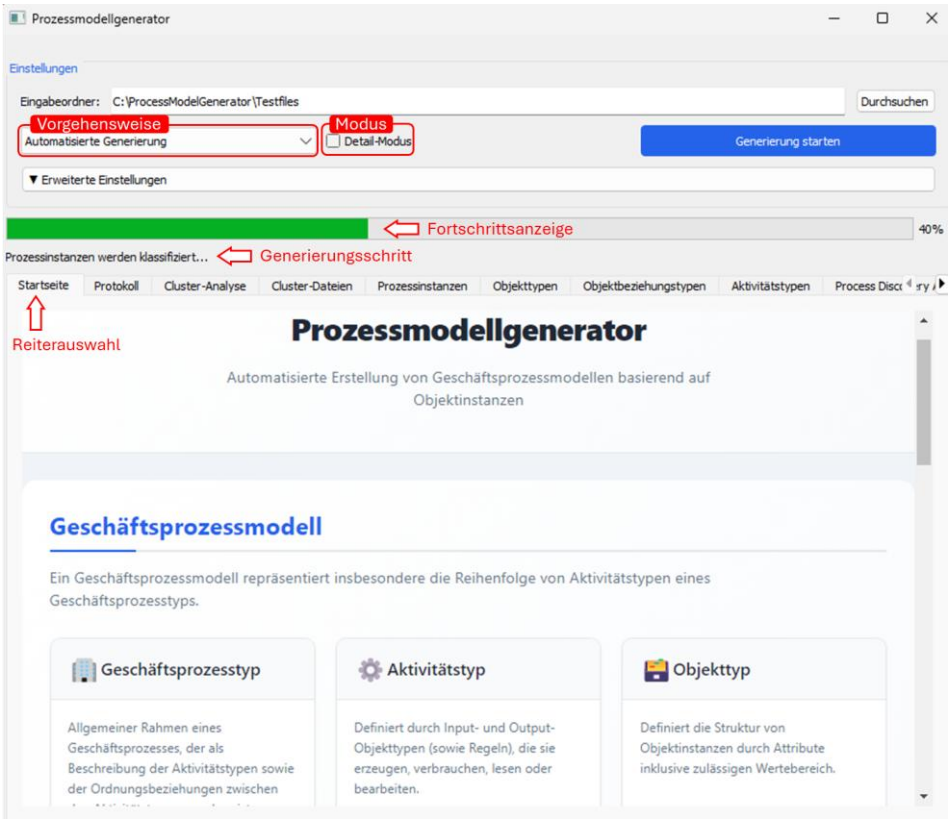


Abbildung 7-2: Ausschnitt der Startseite des Prozessmodellgenerators während einer laufenden Prozessmodellgenerierung

Bei einer Datenvorverarbeitung werden die vom Anwender bereitgestellten Dateien (Input-Dateien) so vorbereitet, dass die Durchführung der nachfolgenden Schritte (*Informationsextraktion* und *Transformation*) gewährleistet werden kann. Ist dies für einzelne Input-Datei nicht möglich, werden diese von der Weiterverarbeitung ausgeschlossen und protokolliert. Als Input-Datei werden zunächst JSON- oder XML-Dateien angenommen, die bereits in ihren Rohdaten die Objektinstanzen als Attribut-Wert-Paare repräsentieren. Jedoch werden durch die Erweiterungen in Abschnitt 7.3 weitere Dateiformate ermöglicht. In dieser Arbeit ist eine Input-Datei zur Beschreibung von Objektinstanzen definiert über:

Definition 7-1: Input-Datei zur Beschreibung von Objektinstanzen

Eine Datei ist ein Tripel D = (*name*, *metadaten*, *inhalt*), mit:

- *name*: Bezeichnung
- *metadaten* = (
  pfad: Dateipfad/Name,
  format ∈ {XML, JSON, ...},
  zeitpunkte = (erstellungsdatum: Zeitpunkt der Erstellung, änderungsdatum: Zeitpunkt der letzten Änderung, zugriffsdatum: Zeitpunkt des letzten Zugriffs),
- *inhalt* = Rohdaten der Datei, die die Objektinstanz beschreiben

Nach der Dateivorverarbeitung folgt basierend auf den in Abschnitt 6.1 beschriebenen Schritten die Informationsextraktion zur Ableitung der Objekttypen und Aktivitätstypen (Abschnitt 7.2.1). In Abschnitt 7.2.2 folgt die Transformation der extrahierten Informationen in ein Geschäftsprozessmodell in Form eines Petri-Netzes basierend auf den beschriebenen Schritten in Abschnitt 6.2.

7.2.1 Informationsextraktion

Dieser Abschnitt beschreibt die Umsetzung der einzelnen Schritte zur Informationsextraktion. Wie in Abschnitt 6.1 beschrieben, erfolgt zunächst die Einteilung zu verschiedenen Objekttypen (Abschnitt 7.2.1.1). Anschließend werden diese einzelnen Prozessinstanzen zugeordnet (Abschnitt 7.2.1.2). Basierend auf diesen Zuordnungen werden die Objekttypen definiert (Abschnitt 7.2.1.3) sowie daraus Objektbeziehungstypen extrahiert (Abschnitt 7.2.1.4). In Abschnitt 7.2.1.5 werden aus den Objektbeziehungstypen die Aktivitätstypen abgeleitet sowie auf mögliche Zusammenführungen oder Entfernungen geprüft.

7.2.1.1 Zuordnung zu Objekttypen

Die Dateien repräsentieren in ihren Rohdaten Objektinstanzen von Objekttypen, die in den Ausführungen der Aktivitätstypen eines Geschäftsprozesstyps erzeugt, verbraucht, gelesen oder bearbeitet worden sind. Zur Prozessmodellgenerierung muss bekannt sein, welche Objekttypen in dem Geschäftsprozesstyp auftreten und welche Objektinstanzen von welchem Objekttyp sind. Daher erfolgt zunächst die Zuordnung der Dateien, die die Objektinstanzen repräsentieren, zu Clustern. Die Cluster sollen dabei jeweils die Objektinstanzen eines Objekttyps repräsentieren. Eine Herausforderung stellt die strukturelle und inhaltliche Heterogenität der Dateien dar. Diese können sowohl einzelne als auch mehrere Objektinstanzen repräsentieren und die Objektinstanzen können verschachtelte Attributstrukturen mit unterschiedlicher Tiefe aufweisen.

Für diese Einordnung wird wie in Abschnitt 6.1.3 beschrieben ein Cluster-Algorithmus angewandt, der die Dateien basierend auf ihrem Titel, ihren Metadaten und den Attribut-Wert-Paaren der Objektinstanzen in verschiedene Cluster einordnet. Die Implementierung basiert auf der Python-Bibliothek *scikit-learn* [PVGM+11], die verschiedene Clustering-Verfahren bereitstellt. Im ersten Schritt werden alle Attribute der Objektinstanzen rekursiv extrahiert und in eine flache Liste überführt, um auch verschachtelte Strukturen zu berücksichtigen. Anschließend erfolgt eine linguistische Vorverarbeitung der Dateiinhalte durch Tokenisierung, Normalisierung und Elimination von Stoppwörtern. Die vorverarbeiteten Texte werden mittels Term Frequency-Inverse Document Frequency (TF-IDF) [Spar72] vektorisiert. Dabei wird die Häufigkeit eines Terms in einem Dokument (TF) mit der inversen Dokumenthäufigkeit (IDF) gewichtet. Die IDF reduziert den Einfluss häufig auftretender Terme, die wenig Diskriminierungskraft besitzen. Dies ermöglicht eine numerische Repräsentation der Texte unter Berücksichtigung der Relevanz der einzelnen Terme. Die Anzahl der Cluster (Anzahl der Objekttypen) wird durch zwei Evaluierungsmetriken bestimmt:

- Der Silhouette-Koeffizient [Rous87] misst die durchschnittliche Distanz eines Punktes zu allen anderen Punkten im selben Cluster im Verhältnis zur durchschnittlichen Distanz zu den Punkten des nächstgelegenen Clusters. Ein Silhouette-Score nahe 1 deutet auf gut definierte und trennbare Cluster hin [0,1].
- Die Sum of Squared Errors (SSE) [TiWH01] summiert die quadrierten euklidischen Distanzen zwischen den Datenpunkten und ihren Clusterzentren. Diese Metrik hilft dabei, den Punkt zu identifizieren, an dem zusätzliche Cluster keine signifikante Verbesserung der Clusterqualität mehr erreichen. Der Wert 0 würde eine perfekte Clustereinteilung bedeuten (jeder Datenpunkt ist sein eigenes Cluster), während hohe Werte eine größere Streuung innerhalb der Cluster anzeigen.

Die finale Clusterbildung erfolgt durch den K-Means-Algorithmus [JaDu88], der die Datenpunkte iterativ dem nächstgelegenen Clusterzentrum zuordnet und die Zentren neu berechnet, bis Konvergenz erreicht ist. Der Software-Prototyp stellt dazu die Ergebnisse der Schritte im Reiter *Cluster-Analyse* bereit (siehe Abbildung 7-3). Die ermittelten Cluster werden anschließend zur Ableitung der Attribute eines Objekttyps in Abschnitt 7.2.1.3 weiter analysiert.

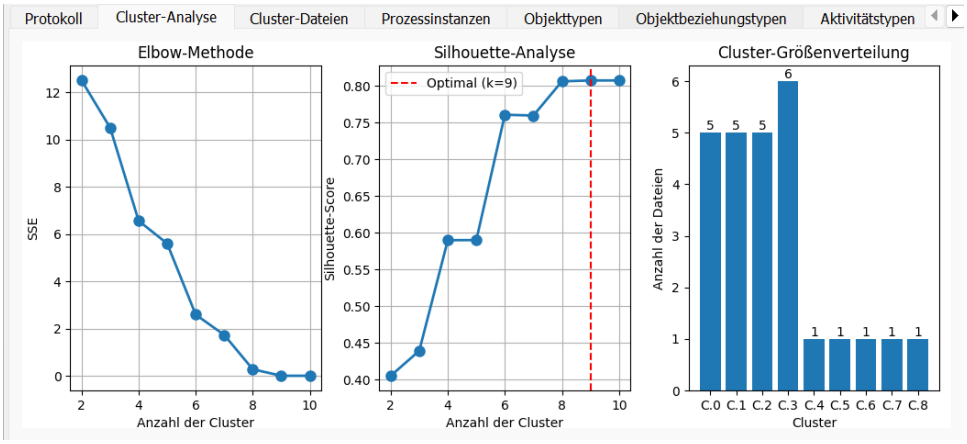


Abbildung 7-3: Ergebnis-Ansicht für die Clustering-Analyse

7.2.1.2 Zuordnung zu Prozessinstanzen

Nach der Clustering-Analyse sollen basierend auf den Zusammenhängen zwischen den Objektinstanzen die Prozessinstanzen erkannt werden. Um mehrere beschriebene Objektinstanzen innerhalb einer übergebenen Datei zu erkennen (beispielsweise bei einem Datenbankexport, der alle Kunden in einer Datei abbildet), werden die Datenstrukturen rekursiv analysiert und strukturelle Uniformität anhand der Attribut-Übereinstimmung geprüft, um zwischen Einzel- und Mehrfacheinträgen zu unterscheiden. Dazu wird auch geprüft, wie ähnlich strukturierte Objektinstanzen aufgebaut sind und bewertet, ob die einzelnen Objektinstanzen prozessinstanzunabhängig sind (*check_uniform_structure*). Anschließend werden alle Objektinstanzen paarweise verglichen, um zwischen den Objektinstanzen Übereinstimmungen in den Attribut-Wert-Paaren zu identifizieren. Diese Überschneidungen werden in der Form [Objektinstanz-Referenz 1, Objektinstanz-Referenz 2] (Attribut 1, Attribut 2, Überschneidungstyp, Wert1, Wert2, Cluster 1, Cluster 2) zwischengespeichert (*search_dependencies*). Häufig verwendete Attributnamen und Werte sollten dabei separat anaysiert werden. Beispielsweise können Attributnamen wie *Name*, *Datum*, *caseID* zufällig in unterschiedlichen Objektinstanzen auftreten, ohne dass daraus ein Zusammenhang geschlossen werden kann. Ebenso sind Werte wie *true*, *false* und Ziffern in ihrer Verwendung häufig, sodass trotz Vorkommen in verschiedenen Objektinstanzen kein Zusammenhang gefolgert werden sollte. Die gefundenen Übereinstimmungen werden nach den fünf *Überschneidungstypen* unterschieden:

- SAME_KEY_VALUE_PAIR = "gleiche Attribut-Wert-Paare"
- SAME_VALUE = "gleiche Werte bei unterschiedlichen Attributen "
- SAME_COMMON_VALUE = "gleiche Werte, die jedoch häufig vorkommen"
- SAME_KEY = "gleiche Attribute mit unterschiedlichen Werten"

- SAME_COMMON_KEY = "gleiche Attribute, die jedoch häufig vorkommen"

Anschließend sollen die signifikanten von zufälligen Zusammenhängen durch Schwellwertanalysen getrennt werden sowie Zusammenhänge zu prozessinstanzunabhängigen Objekten (*unabhängige Korrelatoren*) identifiziert werden (*define_correlators*). Die Schwellwerte der unterschiedlichen Methoden zur Prozessinstanz-Zuordnung sind konfigurierbar. Die signifikanten Attribut-Wert-Paare werden nach *direkten Korrelatoren*, die sowohl gleiche Attribute als auch gleiche Werte haben und *synonyme Korrelatoren*, die gleiche Werte zu unterschiedlichen Attributen haben, unterschieden. Diese differenzierte Betrachtung soll eine Erkennung identifizierender Zusammenhänge auch bei inkonsistenter Attributbenennung ermöglichen und fehlerhafte Zuordnungen zu Prozessinstanzen vermeiden. Dazu werden die Zusammenhänge zunächst gruppiert und *Korrelationspaare* über folgende Eigenschaften definiert:

- Attributnamen der Objektinstanzen, die eine Überschneidung aufweisen
- Überschneidungstyp: SAME_KEY_VALUE_PAIR, SAME_VALUE, SAME_COMMON_VALUE
- Clusterzuordnung der Objektinstanzen

Durch die Gruppierung wird die Häufigkeit je Korrelationspaar im Zusammenhang zur Häufigkeit der Objektinstanzen je Cluster bewertet. Tritt das Korrelationspaar zwischen mehreren unterschiedlichen Objektinstanzen der gleichen Cluster häufiger als ein definierter Schwellwert (beispielsweise 80% der zugeordneten Objektinstanzen je Cluster) auf, wird eine Signifikanz angenommen. Zusätzlich werden die korrelierenden Objektinstanzen je Korrelationspaar auf Ähnlichkeiten analysiert. Hierzu wird ein Ansatz basierend auf der Jaccard-Ähnlichkeit verwendet, um die Qualität und Zuverlässigkeit der identifizierten Korrelatoren zu bewerten. Für jede Kombination von Objektinstanz-Clustern und Korrelationspaare wird die Überlappung der zugeordneten Objektinstanzen analysiert. Die Jaccard-Ähnlichkeit wird dabei als Verhältnis zwischen der Schnittmenge und Vereinigungsmenge der Objektinstanzen berechnet. Dieser Ähnlichkeitskoeffizient liefert einen Wert zwischen 0 und 1, wobei 1 eine vollständige Übereinstimmung und 0 keine Überschneidung bedeutet. Eine hohe Überlappung der Objektinstanzen zwischen verschiedenen Korrelationspaaren deutet auf konsistente Zusammenhänge hin [NSNW13]. Daher werden Korrelatoren, die einen bestimmten Schwellwert erfüllen, als relevant für die Prozessinstanz-Zuordnung bewertet. Dieser Ansatz ermöglicht eine Filterung der initial identifizierten Zusammenhänge und reduziert das Risiko falscher Zuordnungen aufgrund zufälliger Übereinstimmungen (*compare_id_similarity*).

Mit den identifizierten Korrelatoren wird ein ungerichteter Graph erstellt, dessen Knoten die Objektinstanzen und dessen Kanten die Korrelationen repräsentieren. Die zusammenhängenden Komponenten dieses Graphen bilden die Grundlage für die Prozessinstanzen

[EAZP+12]. Da jedoch prozessinstanzunabhängige Objekte zu fehlerhaften Zusammenführungen von Prozessinstanzen führen kann, sind Korrelationspaare, die diese betreffen, separat zu behandeln. Zusätzlich sind auch Korrelationspaare zu entfernen, die durch die Transitivität der Korrelationen auftreten. Daher sind die anderen Korrelationspaare auch auf die Attribute der Korrelationspaare zu den prozessinstanzunabhängigen Objekten zu untersuchen und zu entfernen. Tritt das Attribut auch in einem Korrelationspaar mit Prozessinstanzen auf, ist zu prüfen, ob für diese auch jeweils ein Korrelationspaar zu diesem prozessinstanzunabhängigen Objekt zu diesem Attribut existiert. Ist dies der Fall, ist es aus der Menge der zu beachtenden Korrelationspaare zu entfernen (*detect_transitive_correlations; building_document_correlation_graph*).

Aus dem erstellten Graphen werden anschließend alle zusammenhängenden Knoten, die einer Referenz zu den Objektinstanzen entsprechen, analysiert und Komponenten zugeordnet. Eine Komponente entspricht den Referenzen zu den Objektinstanzen einer Prozessinstanz. Die Ergänzung der prozessinstanzunabhängigen Objekte erfolgt anschließend über die zuvor als unabhängig definierten Korrelationspaare. Basierend auf den Referenzen zu den Objektinstanzen erfolgt die Generierung der Prozessinstanzen. Bei Dateien, die mehrere Objektinstanzen in ihren Daten abbilden, wird diejenige Objektinstanz separiert extrahiert, die eine entsprechende Attribut-Wert-Überschneidung zu einer anderen Objektinstanz in der Prozessinstanz hat (*find_connected_components; create_process_instances*).

Die Prozessinstanzen werden in einem Dataframe gespeichert. Für jede Prozessinstanz wird eine Liste mit den Informationen zu den zugehörigen Objektinstanzen spezifiziert. Die Informationen zu den einzelnen Objektinstanzen bestehen dabei aus der Clusternummer, dem Objektinstanz-Inhalt, den identifizierten Zeitpunkten und Boolean-Werten, ob die Objektinstanz nur einen Teil einer Datei betrifft und ob die Objektinstanz auch zu anderen Prozessinstanzen zugeordnet wurde. Die Zeitpunkte wurden dabei sowohl aus den Metadaten der Objektinstanzen als auch den auftretenden Daten in den Inhalten der Objektinstanzen in der Vorverarbeitung extrahiert und in ein einheitliches Format gebracht. Basierend auf den Bewertungskriterien in Tabelle 6-3 wird ein repräsentativer Zeitpunkt definiert, der den *Zeitpunkt der Objektinstanz* nach Definition 6-4 angibt. Dieser Zeitpunkt gibt an, zu welchem Zeitpunkt in einer Prozessinstanz der Zustand der Objektinstanz galt. Das Datenmodell der Prozessinstanzen ist in Tabelle 7-1 zusammengefasst.

Tabelle 7-1: Prozessinstanzen

Attribut	Wertetyp	Beschreibung
Referenz-ID (inst_id)	String	Zur eindeutigen Identifikation wird für jede Prozessinstanz die Referenz zum Eintrag im Datafra-

Attribut	Wertetyp	Beschreibung
		me als ID festgelegt.
Zugeordnete Objektinstanzen zur Prozessinstanz (process_docs)	Liste [Objekt]	o Liste mit den Informationen zu den zugeordneten Objektinstanzen: o Cluster [Int], o Dateireferenz [String], o Partieller Inhalt [Liste]: Objektinstanz(en) (Inhalts-Ausschnitt einer Datei, falls Datei mehrere Objektinstanzen beschreibt. Falls mehrere Objektinstanzen einer Datei relevant sind, werden alle in der Liste gespeichert), o Zeitpunkte der Metadaten [Dict], o Zeitpunkte aus den Attribut-Wert-Paaren der Objektinstanzen [Dict], o berechneter repräsentativer Zeitpunkt der Objektinstanz [dd.mm.yyyy HH:MM:SS], o is_partial [BOOL]: wahr, falls die Datei mehrere Objektinstanzen beschreibt, o is_shared [BOOL]: wahr, falls die Objektinstanz mehreren Prozessinstanzen zugeordnet wurde.

Für die Objektinstanzen werden Versionsnummern definiert, falls mehrere Objektinstanzen eines Objekttyps in einer Prozessinstanz auftreten. Diese Versionen sind für die Analyse der Zusammenhänge in einer Prozessinstanz relevant, um daraus die Objektbeziehungstypen und Aktivitätstypen abzuleiten (Abschnitt 7.2.1.4 und 7.2.1.5). Der Software-Prototyp stellt dazu die Ergebnisse der Prozessinstanz im Reiter *Prozessinstanzen* zusammengefasst bereit (siehe Abbildung 7-4). Es werden auch die nicht zugeordneten Dateien notiert. Zusätzlich ist ersichtlich, ob eine Objektinstanz mehreren Prozessinstanzen zugeordnet wurde und ob eine Datei mehrere Objektinstanzen beschreibt. Für jede Prozessinstanz können die Objektinstanzen angezeigt werden (siehe Abbildung 7-5).

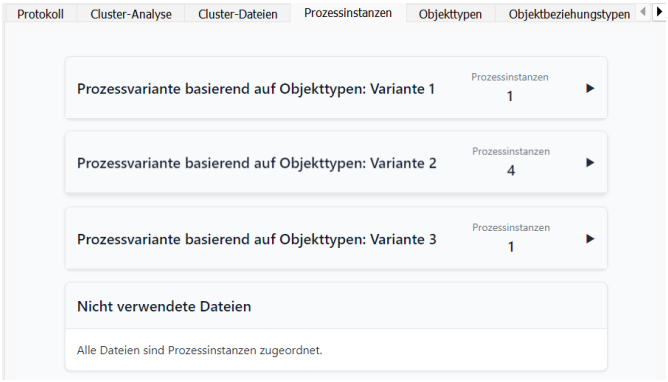


Abbildung 7-4: Ergebnisse-Ansicht für die Prozessinstanz-Varianten (mit der Möglichkeit durch ► mehr Informationen anzuzeigen)

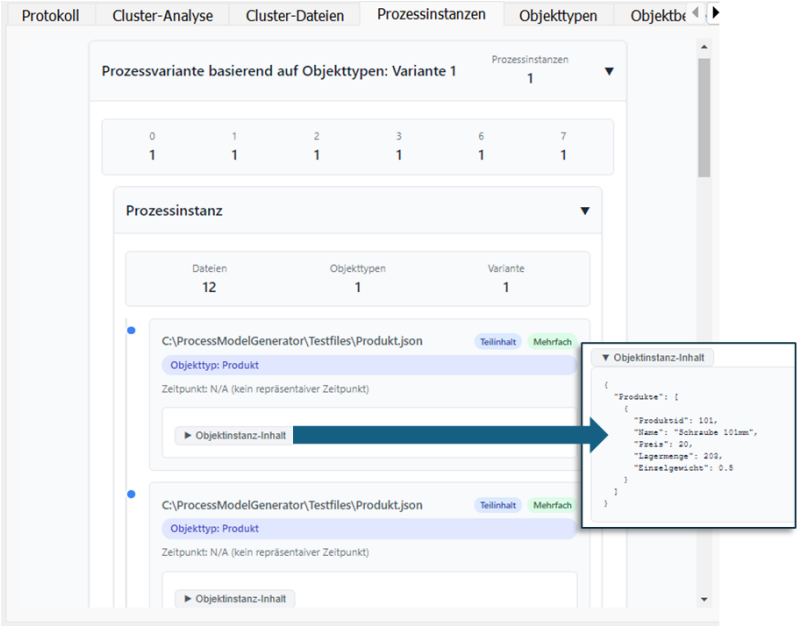


Abbildung 7-5: Aufgeklappte Prozessinstanz inklusive der Objektinstanzen (zum Scrollen)

7.2.1.3 Objekttypen

Aus den analysierten Objektinstanzen, die bereits Prozessinstanzen und Clustern zugeordnet wurden, können die in einem Prozesstyp auftretenden Objekttypen definiert werden. Für jedes durch das Clustering-Verfahren ermittelte Cluster wird ein Objekttyp erzeugt. Diese Objekttypen werden über eine Bezeichnung, die Referenzen für die

zugehörigen Objektinstanzen, die beschreibenden Eigenschaften des Attributtyps durch die möglichen Attribute und deren Wertetyp sowie die Art der Zuordnung der Objektinstanzen definiert. Die Bezeichnung des Objekttyps wird wie in Abbildung 7-6 dargestellt über ein generatives Transformermodell (OpenAI[3]) aus den Attributen eines Objekttyps abgeleitet. Die Schnittstelle zum Transformermodell wird durch die Bibliothek Langchain[4] bereitgestellt. Alternativ wird der Name aus den Dateinamen der zugeordneten Dateien eines Objekttyps bestimmt (*ObjectTypeGenerator NameGenerator*).

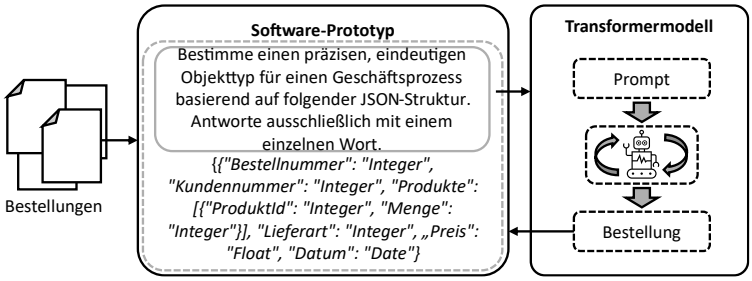


Abbildung 7-6: Bezeichnungsgenerierung durch ein Transformermodell

Für die Definition der Attribute und Wertetypen wird zunächst ein JSON-Schema erstellt, das die Struktur der Objektinstanzen eines Objekttyps beschreibt. Dazu wird die Bibliothek GenSON[5] verwendet, die aus JSON-Objekten automatisch Schemas generiert. Bei Array-Objekten werden alle Einträge einzeln zum Schema hinzugefügt, um eine vollständige Abdeckung der Datenstruktur zu gewährleisten. Das generierte Schema wird anschließend in ein Dictionary mit Attributnamen und deren Datentypen konvertiert, wobei auch verschachtelte Objekte und Arrays berücksichtigt werden. Treten die Attribute in jeder Objektinstanz eines Objekttyps auf, werden diese als *required* gekennzeichnet. Die Kategorie der Zuordnung wird basierend auf den Prozesszuordnungen der Objektinstanzen bestimmt. Wenn beispielsweise alle Objektinstanzen eines Objekttyps nur jeweils einer Prozessinstanz zugeordnet sind, wird der Objekttyp als PROCESS_INSTANCE kategorisiert. Wird mindestens eine Objektinstanz mehrfach zugeordnet, wird der Objekttyp als PROCESS_INSTANCE_INDEPENDENT kategorisiert. Ist keine Objektinstanz einer Prozessinstanz zugeordnet, ist der Objekttyp als INDEPENDENT kategorisiert. Das Datenmodell der Objekttypen ist in Tabelle 7-2 zusammengefasst.

[3] https://openai.com/
[4] https://python.langchain.com/docs/integrations/chat/openai/
[5] https://pypi.org/project/genson/

Tabelle 7-2: Objekttypen

Attribut	Wertetyp	Beschreibung
Referenz-ID (inst_id)	String	Zur eindeutigen Identifikation wird für jeden Objekttyp die Referenz zum Python-Objekt als ID festgelegt.
Bezeichnung (name)	String	Bezeichnung des Objekttyps.
Attribute (attribute)	Dictionary [Objekt]	Liste der Attribute, mit denen ein Objekttyp beschrieben wird. Ein Eintrag in dem Dictionary besteht aus dem Attributname und dem Wertetyp sowie dem Wert *required*, falls das Attribut in allen Objektinstanzen des Objekttyps auftritt.
Kategorie (category)	Enum	Einteilung der Objekttypen je nach Zuordnung der Objektinstanzen in Prozessinstanzobjekte, prozessinstanzunabhängige Objekte.

Der Software-Prototyp stellt dazu die Ergebnisse der Objekttypen im Reiter *Objekttypen* bereit (siehe Abbildung 7-7).

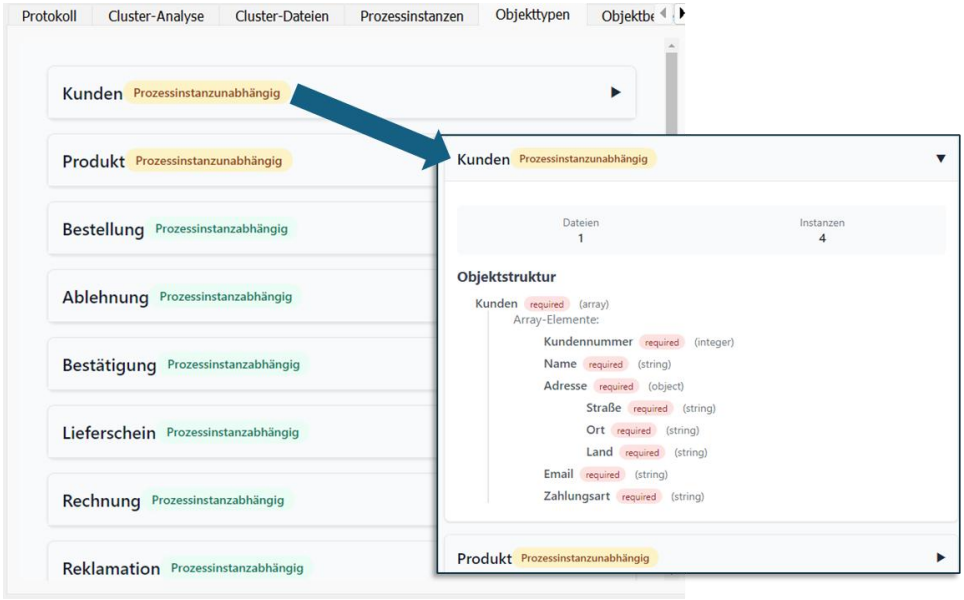


Abbildung 7-7: Ergebnis-Ansicht der Objekttypen im Software-Prototyp (zum Scrollen)

7.2.1.4 Objektbeziehungstypen

Zur Identifikation und Analyse der Objektbeziehungstypen zwischen den Objekttypen werden sowohl inhaltliche als auch zeitliche Zusammenhänge zwischen den Objektinstanzen analysiert (*ObjectRelationshipTypeGenerator*).

Für das Aufdecken der *inhaltlichen Zusammenhänge* werden die Überschneidungen zwischen den Objektinstanzen aus dem paarweisen Vergleich für die Prozessinstanz-Zuordnung extrahiert. Diese Überschneidungen werden durch folgende Aspekte definiert:

- die betreffenden Objekttypen,
- die korrelierenden Attribute,
- den Regeltyp basierend auf den Zusammenhängen in Abschnitt 7.2.1.2
    - SAME_KEY_VALUE_PAIR = "gleiche Attribut-Wert-Paare"
    - SAME_VALUE = "gleiche Werte bei unterschiedlichen Attributen "
    - SAME_COMMON_VALUE = "gleiche Werte, die häufig vorkommen"
    - SAME_KEY = "gleiche Attribute mit unterschiedlichen Werten"
    - SAME_COMMON_KEY = "gleiche Attribute, die häufig vorkommen"
- die Prozessinstanzen, in denen die Überschneidung auftritt inklusive der beteiligten IDs der Objektinstanzen.

Da auch inhaltliche Zusammenhänge entdeckt werden, die zufällig sein können, erfolgt eine Bewertung anhand definierter Schwellwerte:

- **Prozessinstanz-Relevanz (= 80%):**
    - Bewertet, wie häufig die gleichen Werte für die definierten Attributnamen und Objekttypen in verschiedenen Prozessvarianten vorkommen müssen. Beispiel: Wenn eine Überschneidung der *Bestellnummer* in einer Prozessvariante mit 10 Prozessinstanzen vorkommt, muss die Überschneidung in mindestens 8 der Prozessinstanzen dieser Prozessvariante vorkommen. Für Standardwerte wie *true*/*false* gilt ein höherer Schwellwert (= 1), da diese Werte häufig vorkommen können.
- **Objektinstanz-Relevanz (= 90%):**
    - Bewertet die Anzahl der Objektinstanzen, in denen die Überschneidung vorkommen muss. Beispiel: Wenn 100 Rechnungen existieren, müssen mindestens 90 davon diese Überschneidung aufweisen.
- **Attributs-Prüfung (= 90%):**
    - Wird verwendet, wenn mehrere Überschneidungen zwischen denselben Objekttypen gefunden wurden. Vergleicht die Häufigkeit dieser Überschneidungen miteinander. Beispiel: Überschneidung A tritt zwischen

100 Objektinstanzen auf; Überschneidung B tritt zwischen 60 Objektinstanzen auf. Dementsprechend wird Überschneidung B entfernt, da diese unter 90% der Häufigkeit von A liegt.

Die Übereinstimmungen der Attribute zwischen zwei Objekttypen, die die Schwellwerte erfüllen, werden für die jeweilige Objektbeziehungstypen gespeichert. Werden mehrere Übereinstimmungen gefunden, enthält diese Liste demnach mehrere Einträge. Aus den Zusammenhängen werden die Objektbeziehungstypen definiert. Die Zusammenhänge, die als zufällig bewertet werden, werden als NO_RELATIONSHIP gespeichert. Jeder Objektbeziehungstyp enthält Informationen über:

- die beteiligten Objekttypen
- den identifizierten Zusammenhang (korrelierende Attribute) mit Regeltyp

Die Objektbeziehungstypen bilden damit zunächst nur einfache direkte Zusammenhänge zwischen zwei Objekttypen ab. Zur weiteren Analyse werden zu jedem Objektbeziehungstyp die konkreten Ausprägungen gespeichert. Dazu werden Referenzen zu den Prozessinstanzen und den beteiligten Objektinstanzen, in denen der Objektbeziehungstyp auftritt, der identifizierte Zusammenhang und die Attributwerte der im Zusammenhang stehenden Attribute gespeichert.

Für das Aufdecken der *zeitlichen Zusammenhänge* wird das Dataframe mit den gespeicherten Objektinstanzen so erweitert, dass für jede zugewiesene Prozessinstanz, ein separater Eintrag entsteht. Anschließend werden für jede Prozessinstanz Objektinstanzen ohne Zeitpunkt entfernt und die verbliebenen Objektinstanzen nach ihrem Zeitpunkt sortiert. Für zwei direkt aufeinanderfolgende Objektinstanzen einer Prozessinstanz wird ein Objektbeziehungstyp erzeugt, wobei die Objektinstanz mit früherem Zeitpunkt als Input-Objekttyp und die mit späterem Zeitpunkt als Output-Objekttyp definiert werden.

Das Datenmodell der Objektbeziehungstypen (der inhalts- und zeit-basierten Zusammenhänge) ist in Tabelle 7-3 zusammengefasst.

Tabelle 7-3: Objektbeziehungstyp

Attribut	Wertetyp	Beschreibung
Referenz-ID (inst_id)	String	Zur eindeutigen Identifikation wird für jeden Objektbeziehungstyp die Referenz zum Python-Objekt als ID festgelegt.
Bezeichnung (name)	String	Bezeichnung des Objektbeziehungstyps
Objektbeziehungstyp (RelationshipType)	Enum	Content_based, Time_based, no_relationship

Attribut	Wertetyp	Beschreibung
Beteiligte Objekttypen (Objekttypes)	Dictionary {(Objekttyp, Version): Wert}	Ein Objektbeziehungstyp wird zwischen Objekttypen definiert, wenn Attribute, Werte oder Attribut-Wert-Paare übereinstimmen. Daher werden Objektbeziehungstypen durch die Objekttypen und deren Bewertung (siehe Tabelle 6-4) für Input bzw. Output beschrieben.
Prozessinstanz-IDs in denen der Objektbeziehungstyp auftritt (Processinstances)	Dictionary {(Ruletype, Attributnamen): ProcessId, Attributwerte, Objektinstanzen-IDs}	IDs, der Prozessinstanzen, die eine bestimmte Regel des Objektbeziehungstypen aufweisen sowie die entsprechenden übereinstimmenden Attributnamen inklusive der Attributwerte und der Objektinstanzen.
Regeln (rules)	Liste [(ENUM, String)]	Liste der Regeln für einen Objektbeziehungstyp. Dabei ist der ENUM-Wert der Regeltyp und der String die übereinstimmenden Attributnamen.

Diese Erfassung der Objektbeziehungstypen bildet die Grundlage für die spätere Generierung des Prozessmodells, da sie Aufschluss über die Abhängigkeiten und den Datenfluss zwischen den Objektinstanzen gibt, um daraus die Aktivitäten eines Geschäftsprozesses abzuleiten. Abbildung 7-8 zeigt die Darstellung der Objektbeziehungstypen, wobei auch die angezeigt werden, die basierend auf der Bewertung entfernt wurden. Abbildung 7-9 zeigt, wie ein inhaltsbasierter Objektbeziehungstyp dargestellt wird.

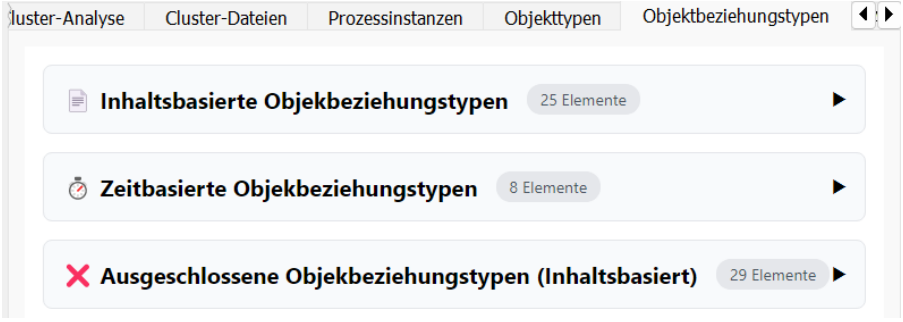


Abbildung 7-8: Ergebnis-Ansicht der extrahierten Objektbeziehungstypen je Typ (mit der Möglichkeit durch ► mehr Informationen anzuzeigen)

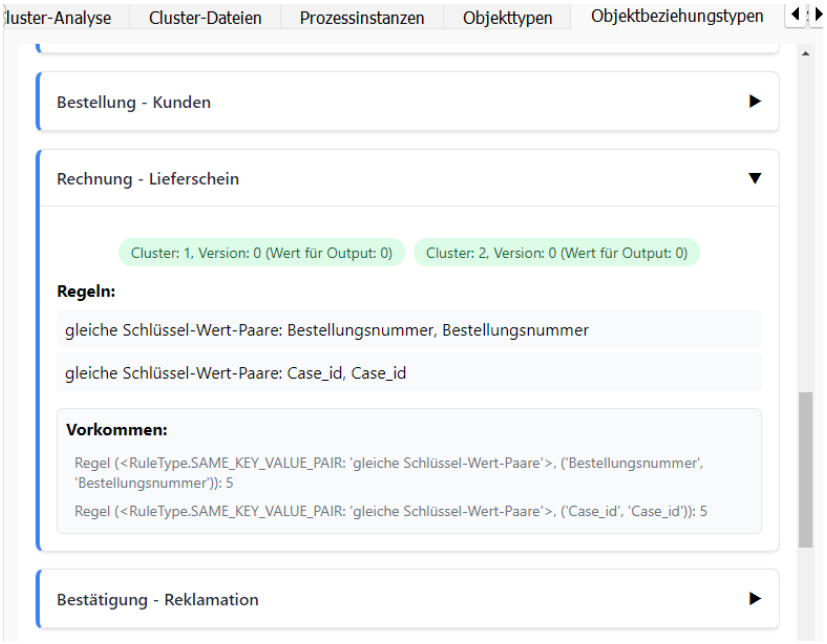


Abbildung 7-9: Exemplarische Detail-Ansicht eines inhaltsbasierten Objektbeziehungstyps (▼)

7.2.1.5 Aktivitätstypen

Die Ableitung der Aktivitätstypen erfolgt in einem mehrstufigen Prozess: Zunächst werden aus den Objektbeziehungstypen Aktivitätstypen identifiziert, die durch ihre Input- und Output-Objekttypen definiert sind. Anschließend werden die Aktivitätstypen bewertet und bei Bedarf zusammengeführt.

Für jeden Objektbeziehungstyp aus den *inhaltlichen Zusammenhängen* wird anhand der verschiedenen Kriterien für jeden aufgetretenen Zusammenhang (Aktivitätsinstanzen) in den Prozessinstanzen bewertet, ob ein Objekttyp Input bzw. Output ist. Dies wird anhand der in Tabelle 6-4 beschriebenen Bewertungsaspekte durchgeführt. Die Bewertung gibt einen Wert zwischen -1 und 1 zurück, wobei das Vorzeichen die Richtung (Input: -; Output: +) und der Wert die Konfidenz der Beziehung angibt. Die initiale Bewertung erfolgt anhand folgender Kriterien:

1. Zeitliche Analyse der Objektinstanzen: Durch den Vergleich der Zeitpunkte zwischen den Objektinstanzen wird die zeitliche Reihenfolge bestimmt. Der Objekttyp der früheren Objektinstanz erhält -1 als Wert für diese Instanz des Objektbeziehungstyps. Der Objekttyp der späteren Objektinstanz erhält 1 als Wert für diese Instanz des Objektbeziehungstyps.

2. Prozessinstanzunabhängigkeitsanalyse: Es wird untersucht, ob eine Objektinstanz prozessinstanzunabhängig ist und ob sich die Werte der Objektinstanz innerhalb von den Prozessinstanzen ändern. Ist dies nicht der Fall, wird der Wert des Objekttyps der prozessinstanzunabhängigen Objektinstanzen mit -1 als Wert festgelegt. Der andere Objekttyp erhält 1 als Wert.
3. Trifft kein Kriterium zu, erhalten beide Objektinstanzen den Wert 0.

Durch diese Bewertung erhält jede Ausprägung des Objektbeziehungstyp einen Wert für die Objektinstanzen der beteiligten Objekttypen. Anhand der einzelnen Ausprägungen der Objektbeziehungstypen je Objekttyp wird der Durchschnitt der Werte berechnet. Ein negativer Durchschnittswert im Bereich [-1; -0,2] identifiziert den Objekttyp als Input, ein positiver Wert im Bereich [0,2; 1] als Output. Liegt der Wert für die Objekttypen in (-0,2; 0,2), erfolgt eine automatische Nachanalyse unter Einbeziehung bereits definierter Aktivitätstypen. Dabei werden die Aktivitätstypen genutzt, um die Werte anzupassen und eine eindeutigere Zuordnung zu ermöglichen. Dabei wird für jeden unklaren Objekttyp geprüft, ob dieser in anderen Aktivitätstypen als Input oder Output auftritt. Bei einem Auftreten als Input wird das Gewicht um 0,15 erhöht, bei einem Auftreten als Output entsprechend um 0,15 verringert. Die erstellten Aktivitätstypen sind anschließend noch weiter zu überprüfen, da noch redundante Übernahmen oder zusammenführbare Übernahmen von Attributen vorhanden sein können. Tritt beispielsweise eine Kundennummer in mehreren Objekten auf, wird zwischen allen diesen Objekten eine Übereinstimmung erkannt. Jede dieser Überstimmungen führt zu einem Objektbeziehungstyp, sofern sie den definierten Schwellwert erreicht und wird anschließend als Aktivitätstyp definiert.

Für die Auswahl der relevanten Aktivitätstypen wurde mit dem in Abschnitt 6.1.5 definierten Aspekten ein Bewertungssystem entwickelt, das verschiedene Aspekte der Aktivitätstypen berücksichtigt und durch konfigurierbare Gewichtungsfaktoren gesteuert wird:

- Die *positionsbasierten Faktoren* bewerten die relative Position der Aktivitäten im Geschäftsprozess. Aktivitäten, die früh im Prozess auftreten, erhalten einen Bonus (Gewichtungsfaktor 1,5), während späte Aktivitäten mit einem Abschlag (Gewichtungsfaktor 0,5) versehen werden. Dies erfolgt durch die Analyse der durchschnittlichen Knotenentfernung vom Prozessstart.
- Bei der *Bewertung von prozessinstanzunabhängigen Objekttypen* werden zwei Faktoren kombiniert: Der *Referenzgewichtungsfaktor* (2,0) erhöht die Bewertung für Aktivitäten mit prozessinstanzunabhängigen Objekten, wobei nähere Beziehungen durch Division mit der Knotenentfernung stärker gewichtet werden. Der *Referenz-Positions-Faktor* (2,0) verstärkt diesen Effekt zusätzlich durch Multiplikation. Ausgehende Kanten von prozessinstanzunabhängigen Objekten, die keine eingehende Kante haben, bekommen eine höhere Gewichtung je näher sie am ersten Objekt einer Prozessinstanz sind. Diese Objekte dienen demnach nur zur In-

formationsbereitstellung und liegen im Unternehmen bereits vor den einzelnen Ausführungen des Geschäftsprozesses vor (wie Kunden- oder Produktdaten).

- Die *eigenschaftsbasierten Faktoren* umfassen die Bewertung *sequenzieller* Eigenschaften (1,5), *attributbasierter* Eigenschaften (1,2) und der *zeitlichen Nähe* zwischen Objektinstanzen (2,0). Die Gewichtung sequenzieller Eigenschaften steigt dabei mit der Häufigkeit ihres Auftretens. Die durchschnittliche Attribut-Bewertung wird mit dem attributbasierten Eigenschaftenfaktor multipliziert.
- *Zusätzliche Faktore*n beeinflussen die Bewertung: Der *inverse-Ordnungsbeziehungsfaktor* (3,0) für Objekttypen reduziert die Bewertung bei auftretenden inversen Ordnungsbeziehungen zwischen den beteiligten Objekttypen eines Aktivitätstyp. Der *Versions-Faktor* (0,5) berücksichtigt verschiedene Versionen von Objekttypen. Die Bewertung wird für Aktivitätstypen erhöht, die disjunkte Versionen von Objektinstanzen betreffen. Dabei wird die Bewertung dann erhöht, wenn ein Aktivitätstyp nur die niedrigeren Versionen der Objekttypen betrifft und der andere nur die höheren. Betreffen zwei Aktivitätstypen im Input oder Output gleiche Versionen, wird die Bewertung des Aktivitätstyps mit der höheren Version reduziert. Der *Schleifen-Gewichtungsfaktor* (0,5) reduziert die Bewertung potenzieller Schleifenstrukturen. Es wird die Bewertung für Aktivitätstypen, die die gleichen Input- und Output-Objekttypen betreffen, verringert.

Zur Berechnung der *eigenschaftsbasierten Faktoren* wird mittels der *NetworkX*[6]-Bibliothek ein gewichteter gerichteter Graph erstellt. Die Knoten repräsentieren die Objekttypen und die Kanten die Aktivitätstypen. Das Kantengewicht setzt sich dabei aus den drei Komponenten zusammen:

- sequenzielle Eigenschaft (Prozessinstanzbasierte Gewichtung):
    - Das Kantengewicht wird zusätzlich durch die Häufigkeit des Auftretens des Aktivitätstyps in den Prozessinstanzen beeinflusst.
    - Je häufiger ein Zusammenhang in den Prozessinstanzen beobachtet wird, desto höher das resultierende Kantengewicht.
- Attributbasierte Eigenschaft:
    - Für jedes Attribut wird zunächst die Verwendungshäufigkeit über alle Aktivitätstypen ermittelt.
    - Seltener verwendete Attribute erhalten ein höheres Gewicht (1 - Häufigkeit/Maximale Häufigkeit eines Attributs).
    - Aktivitätstypen, die solche selten verwendeten Attribute übertragen, werden als wichtiger eingestuft.
- zeitlichen Nähe zwischen Objektinstanzen:

[6] https://networkx.org/

- Für jeden Aktivitätstyp wird ein normalisierter Score basierend auf der zeitlichen Nähe der beteiligten Objektinstanzen berechnet. Dazu wird die Zeitdifferenz zwischen den Objektinstanzen einer Aktivität in Relation zur Gesamtdauer der jeweiligen Prozessinstanz gesetzt. Die Normalisierung der zeitlichen Abstände erfolgt nach der Formel: *normalisierterAbstand* = (*Zeitdifferenz*) / (*Prozessinstanzdauer*). Dies ermöglicht einen prozessinstanzübergreifenden Vergleich der zeitlichen Nähe. Existiert zu einem Aktivitätstyp auch ein Objektbeziehungstyp aus den zeitlichen Zusammenhängen wird dies auch höher gewichtet.

Die Gesamtgewichtung *w* einer Kante (eines Aktivitätstyps) berechnet sich als gewichtete Summe der Einzelkomponenten: $w = \alpha * AttributGewicht + \beta * sequenzGewicht + \gamma * normalisierterAbstand$ wobei α, β und γ konfigurierbare Gewichtungsfaktoren sind.

Diese Gewichtungsfaktoren erlauben eine Anpassung des Aktivitätserkennungsprozesses. Durch Anpassung der Werte können beispielsweise:

- Frühe und häufige Aktivitätstypen bevorzugt werden.
- Der Einfluss von objektinstanzunabhängigen Objekttypen verstärkt oder verringert werden.
- Die Bedeutung zeitlicher Nähe adjustiert werden.
- Die Berücksichtigung von Versionen und Schleifen gesteuert werden.

Durch dieses Vorgehen erhält jeder Aktivitätstyp eine Bewertung. Diese Aktivitätstypen werden anschließend absteigend nach dieser Bewertung sortiert und analysiert. Wenn sie einen neuen Objekttyp enthalten oder eine Regel, die neue Attribute betrifft, werden sie als relevant gekennzeichnet. Zusätzlich wird ein Aktivitätstyp als relevant gekennzeichnet, wenn dieser zwar bereits verwendete Objekttypen verwendet, jedoch noch nicht abgedeckte Objektinstanzen dieser Objekttypen betrifft. Mit diesem schrittweisen Vorgehen soll eine vollständige Abdeckung der Prozessinstanzen gewährleistet werden. Durch die Angabe eines minimalen Scores kann jedoch beeinflusst werden, ob tatsächlich alle Objektinstanzen berücksichtigt werden müssen und dadurch beispielsweise verhindert werden, dass einzelne Ausreißer modelliert werden.

Wird ein Aktivitätstyp als relevant gewertet, so werden alle Attribute, die darüber abgedeckt werden, gespeichert. Diese Auswahl erfolgt in drei Runden. In der ersten Runde werden Aktivitätstypen hinzugefügt, die die ersten zwei Aspekte erfüllen. Aktivitätstypen die prozessinstanzunabhängige Objekttypen betreffen, werden dabei bei erstmaligem Auftreten ausgewählt, jedoch bei erneutem Auftreten zunächst nicht mehr hinzugefügt, auch wenn ein neuer Objekttyp beteiligt ist. Zwar sollen die Informationen der prozessinstanzunabhängigen Objekte so früh wie möglich berücksichtigt werden, jedoch sollen

insbesondere die Zusammenhänge der Prozessinstanzobjekte in einem Geschäftsprozess berücksichtigt werden.

Zusätzlich ermöglicht die Implementierung die Auswahl, wie die Attribute berücksichtigt werden sollen: Die Attributanalyse kann entweder pro Objekttyp (*Attribute-je-Objekttyp*: True) oder für den gesamten Geschäftsprozesstyp (False) erfolgen. Bei der objekttypspezifischen Analyse wird für jeden Objekttyp geprüft, ob ein Attributwert bereits durch einen bereits ausgewählten Aktivitätstyp abgedeckt ist. Bei der Analyse bezüglich des Geschäftsprozesstyps wird geprüft, ob ein Attributwert in einem Aktivitätstyp berücksichtigt wird, unabhängig der beteiligten Objekttypen. Die Auswahl relevanter Aktivitätstypen kann zusätzlich durch einen *minimalen Schwellwert* (0,0) gesteuert werden.

Anschließend werden anhand der Analyse aus Abschnitt 6.1.5 die Aktivitätstypen auf mögliche Zusammenführungen oder auf Entfernung geprüft. Zunächst werden Aktivitätstypen mit denselben Input- und Output-Objekttypen untersucht. Dazu werden zunächst die Aktivitätstypen nach den Input- und Output-Objekttypen gruppiert. Für jede identifizierte Gruppe wird eine detaillierte Analyse der Objektverwendung durchgeführt. Werden dieselben Objektinstanzen als Input- und Output verwendet, kann eine Zusammenführung erfolgen. Werden die Aktivitätstypen nur in unterschiedlichen Prozessinstanzen durchgeführt, haben die Aktivitätstypen eine alternative Ordnungsbeziehung und werden daher nicht zusammengeführt. Betreffen die Aktivitätstypen weniger als 90% derselben Objektinstanzen, wird die Zusammenführung weiter geprüft. In diesem Fall kann es auch vorkommen, dass eine Zusammenführung stattfindet und dennoch die ursprünglichen Aktivitätstypen behalten werden. Dazu werden die Prozessinstanzen, in denen die Aktivitätsinstanzen der betreffenden Aktivitätstypen auftreten, analysiert, welche Objektinstanzen beteiligt sind. Dabei wird die Übereinstimmung geprüft, bei der das Verhältnis zwischen gemeinsamen und individuell verarbeiteten Objektinstanzen berechnet wird. Als entscheidendes Kriterium gilt dabei standardmäßig ein Schwellwert von 90%. Liegt die tatsächliche Übereinstimmung unter diesem Wert, initiiert die Methode einen Separierungsprozess. Dazu wird für die Aktivitätsinstanzen, die in den gleichen Prozessinstanzen auftreten und die gleichen Objektinstanzen betreffen ein gemeinsamer Aktivitätstyp erzeugt und bei den ursprünglichen Aktivitätstypen entfernt. Für die nicht übereinstimmenden Prozessinstanzen wird geprüft, ob die Objektinstanzen bereits bei anderen Aktivitätstypen beteiligt sind. Ist dies nicht der Fall, werden die Aktivitätstypen behalten. Werden von einem Aktivitätstyp alle Aktivitätsinstanzen in dem zusammengeführten Aktivitätstyp abgedeckt, wird der Aktivitätstyp entsprechend entfernt.

Anschließend werden Aktivitätstypen untersucht, die gemeinsame Input- oder Output-Objekttypen aufweisen. Dabei werden zunächst die Aktivitätstypen geprüft, die inverse Ordnungsbeziehungen zwischen den Objekttypen aufweisen. Anschließend werden Aktivitätstypen geprüft, die einen gemeinsamen Output haben und deren Input prozessin-

stanzunabhängige Objekttypen betrifft. Zuletzt werden die restlichen Aktivitätstypen geprüft, die Überschneidungen in den Input- oder Output-Objekttypen haben.

Die gefundenen möglichen Zusammenführungen werden auf widersprüchliche Beziehungen zwischen Aktivitäten geprüft. Dabei wird die zeitliche Reihenfolge der Objektinstanzen aller Prozessinstanzen berücksichtigt, um sicherzustellen, dass kein Aktivitätstyp einen Input-Objekttyp (bezüglich derselben Prozessinstanz) zeitlich nach dessen Erzeugung durch eine anderen Aktivitätstyp verwendet. Zusätzlich werden die Regeln, die in den Aktivitätstypen definiert sind, auf Kompatibilität geprüft, da inkompatible Regeln darauf hinweisen, dass eine Zusammenführung nicht sinnvoll sein könnte. Bei der Zusammenführung werden die Input- und Output-Objekttypen zusammengeführt, dabei werden auch die Versionen der Objekttypen berücksichtigt. Zusätzlich werden die Aktivitätsinstanzen der betreffenden Aktivitätstypen vereinigt. Die Regeln werden zusammengeführt, wobei Redundanzen eliminiert werden. Anschließend wird ein neuer, repräsentativer Name generiert, der die kombinierte Funktionalität widerspiegelt.

Der Software-Prototyp stellt dazu die Ergebnisse der Aktivitätstypen im Reiter *Aktivitätstypen* bereit (siehe Abbildung 7-10).

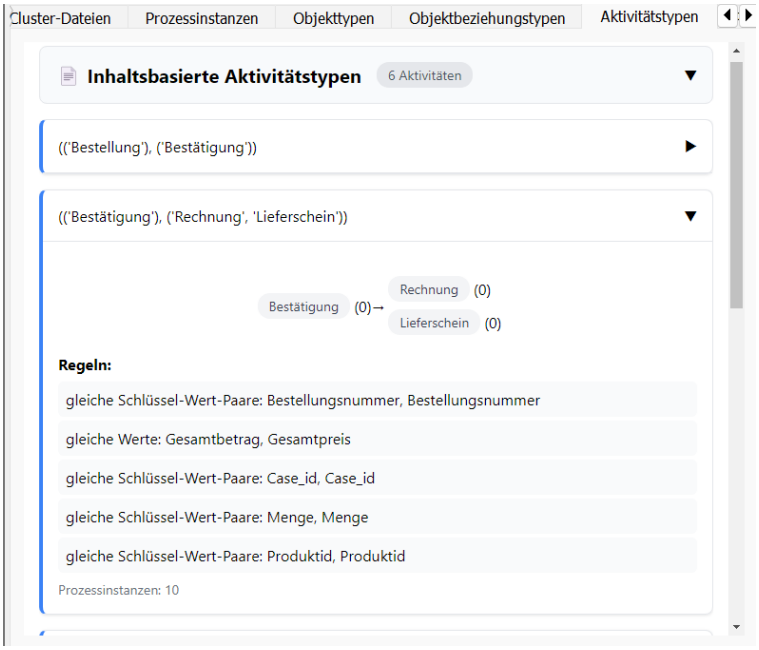


Abbildung 7-10: Darstellung der inhaltsbasierten Aktivitätstypen im Software-Prototyp (zum Scrollen)

Für die Objektbeziehungstypen aus den *zeitlichen Zusammenhängen* zwischen den Objekttypen ist bereits festgelegt, welche Objekttypen Input bzw. Output sind. Jedoch werden bei den zeitlichen Zusammenhängen wie in Abschnitt 6.2.1 beschrieben bei AND-Aufspaltungen und -Zusammenführungen mehrfache Objektbeziehungstypen erzeugt, da die Objekte in unterschiedlichen Reihenfolgen auftreten können und jeder unterschiedliche Übergang zu einem Objektbeziehungstyp führt. Daher wird mittels des Footprint-Discovery-Algorithmus von PM4Py für die Objektinstanzen analysiert, ob diese inverse Ordnungsbeziehungen. Dazu wird aus den Objektinstanzen jeder Prozessinstanz ein Trace erstellt, wobei die Events den Objekttypen und die Zeitstempel dem Mittelwert der Zeitpunkte der Input- bzw. Output-Objektinstanzen entsprechen. Aus dem Eventlog, der aus den einzelnen Traces besteht, werden anschließen die Ordnungsbeziehungen für die Objekttypen abgeleitet:

- Direkte Folgebeziehungen
- Inverse Ordnungsbeziehungen
- Start-Objekttyp
- End-Objekttyp

Aus den gefundenen inversen Ordnungsbeziehungen zwischen den Objekttypen sollen die Objektbeziehungstypen zusammengeführt werden, die diese betreffen. In Abbildung 7-11 sind dies beispielsweise Lieferschein und Rechnung. Anschließend werden die Inputs und Outputs der Aktivitätstypen auf folgende Fälle hin analysiert:

- Objektbeziehungstypen, die nur Objekttypen betreffen, die in einer inversen Ordnungsbeziehung stehen.
- Eingangsbasierte Objektbeziehungstypen, die einen Objekttyp, der in der inversen Ordnungsbeziehung auftritt, als Input haben und dieselben Output-Objekttypen.
- Ausgangsbasierte Objektbeziehungstypen, die einen Objekttyp, der in der inversen Ordnungsbeziehung auftritt, als Output haben und dieselben Input-Objekttypen.

	Bestellung	Bestätigung	Lieferschein	Rechnung	Reklamation	Zahlung
Bestellung	#	>	#	#	#	#
Bestätigung	<	#	>	>	#	#
Lieferschein	#	<	#	||	#	#
Rechnung	#	<	||	#	#	>
Reklamation	#	#	#	#	#	<
Zahlung	#	#	#	<	>	#

Abbildung 7-11: Footprint-Matrix für die Objekttypen

Daher wird geprüft, ob in den Mengen der Objektbeziehungstypen, die eingangsbasiert bzw. ausgangsbasiert sind, jeweils beide Objekttypen mit demselben Input- bzw. demselben Output auftreten. Ist dies der Fall oder treten Objektbeziehungstypen in inversen Ordnungsbeziehungen auf, werden sie in einer zusammenzuführenden Liste gespeichert und als USED_TIME_BASED_RELATIONSHIP markiert. Anschließend werden aus den unverändert markierten Objektbeziehungstypen (TIME_BASED_ RELATIONSHIP) Aktivitätstypen (TIME_BASED) mit entsprechenden Input- und Output-Objekttypen erstellt. Zusätzlich werden aus den zusammenzuführenden Objektbeziehungstypen Aktivitätstypen mit mehrfachen Input- bzw. Output-Objekttypen erstellt (AGGREGATED_TIME_BASED).

Der Software-Prototyp stellt dazu die Ergebnisse dieser Aktivitätstypen im Reiter *Aktivitätstypen* bereit (siehe Abbildung 7-12).

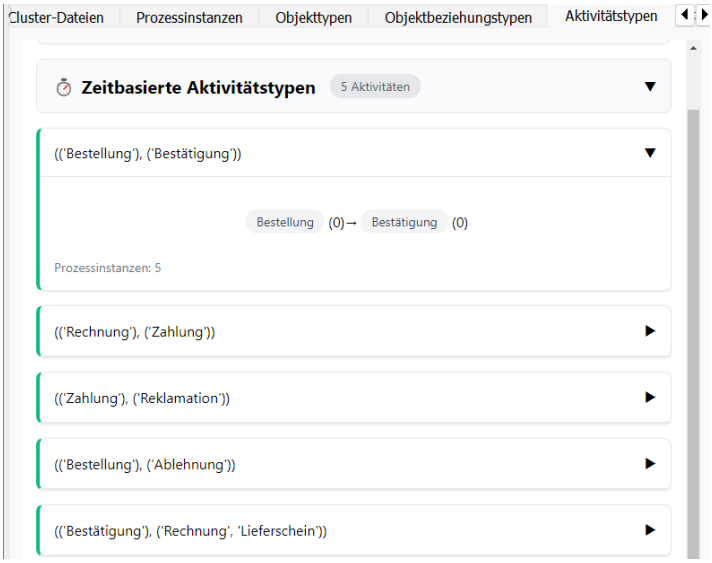


Abbildung 7-12: Darstellung der zeitbasierten Aktivitätstypen im Software-Prototyp (zum Scrollen)

Zur anschließenden Transformation werden für alle Aktivitätstypen die Prozessinstanzen, in denen diese auftreten, gespeichert. Das Datenmodell der Aktivitätstypen ist in Tabelle 7-4 zusammengefasst.

Tabelle 7-4: Aktivitätstypen

Attribut	Wertetyp	Beschreibung
Referenz-ID (inst_id)	String	Zur eindeutigen Identifikation wird für jeden Aktivitätstyp eine ID festgelegt.
Bezeichnung (name)	String	Bezeichnung eines Aktivitätstyp.
Aktivitätstyp (Aktvitytype)	Enum	CONTENT_BASED_AT, TIME_BASED_AT, NO_AT.
Input-Objekttypen (input_objekttypes)	Liste[Objekttyp]	Ein Aktivitätstyp wird über die Input-Objekttypen definiert.
Output-Objekttypen (output_objekttypes)	Liste[Objekttyp]	Ein Aktivitätstyp wird über die Output-Objekttypen definiert.
Regeln (rules)	Liste [String]	Liste der Regeln für die Aktivitätstypen. Dabei besteht ein String immer aus den übereinstimmenden Attributen, Werten bzw. Attribut-Wert-Paaren (beziehungsweise *timebased* bei den zeitlich-basierten Aktivitätstypen).

7.2.2 Transformation

Da die Aktivitätstypen sowohl aus zeitlichen als auch inhaltlichen Zusammenhängen abgeleitet wurden, werden verschiedene Prozessmodellgenerierungsansätze implementiert:

1. Generierung eines Eventlogs, basierend auf den zeitlich-basierten Aktivitätstypen.
2. Zusammensetzen der zeitlich-basierten Aktivitätstypen, basierend auf den Input- und Output-Objekttypen.
3. Zusammensetzen der inhaltich-basierten Aktivitätstypen, basierend auf den Input- und Output-Objekttypen.

In dem ersten Generierungsansatz werden zunächst nur die Aktivitätstypen verarbeitet, die aus den zeitlichen Zusammenhängen stammen. Um aus den zeitlichen Zusammenhängen ein Geschäftsprozessmodell zu generieren, wird zunächst aus den Ausführungen der Aktivitätstypen (Aktivitätsinstanzen) ein Eventlog erstellt. Diese Ausführungen ergeben sich aus den Objektinstanzen der Prozessinstanzen. Für jede Ausführung wird ein Ereignis generiert, das durch die Bezeichnung des Aktivitätstyps, die Prozessinstanz-ID sowie Zeitstempel definiert ist. Dabei werden für jede Ausführung der Aktivitätstypen

die beteiligten Objektinstanzen analysiert, um Start- und Endzeitpunkte zu bestimmen. Der Startzeitpunkt ist der früheste repräsentative Zeitpunkt der Input-Objektinstanzen und der Endzeitpunkt ist der späteste repräsentative Zeitpunkt der Output-Objektinstanzen einer Aktivitätsinstanz. Die Implementierung unterstützt sowohl die Generierung der noch nicht zusammengeführten Aktivitätstypen sowie der aus den Ordnungsbeziehungen der Objektinstanzen abgeleiteten zusammengeführten Aktivitätstypen. Für die Generierung des Geschäftsprozessmodells werden anschließend verschiedene Discovery-Algorithmen von PM4py angewandt. Die extrahierten Informationen wurden mit verschiedenen Process-Discovery-Algorithmen (Alpha Miner, Alpha Miner Plus, Heuristic Miner und Inductive Miner) in Petri-Netze transformiert. Die Verwendung mehrerer Algorithmen dient der Validierung der Ergebnisse, da die Algorithmen unterschiedliche Stärken bei der Erkennung von Prozessstrukturen aufweisen [ACDL+19, WBVB12]:

- Der Alpha Miner erkennt grundlegende Kontrollflussstrukturen
- Der Alpha Plus Miner verbessert die Erkennung von Schleifen und nicht-freien Entscheidungen
- Der Heuristic Miner ist robuster gegenüber Rauschen und seltenen Pfaden
- Der Inductive Miner garantiert ein strukturell korrektes Modell

Abbildung 7-13 zeigt das generierte Geschäftsprozessmodell mit dem Alpha-Miner, bevor die Aktivitätstypen zusammengeführt wurden. In Abbildung 7-14 wird das Petri-Netz mit zusammengeführten Aktivitäten gezeigt.

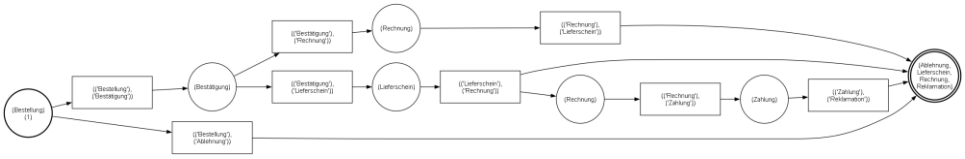


Abbildung 7-13: Aus einem Eventlog abgeleitetes Petri-Netz (Alpha Miner, Export)

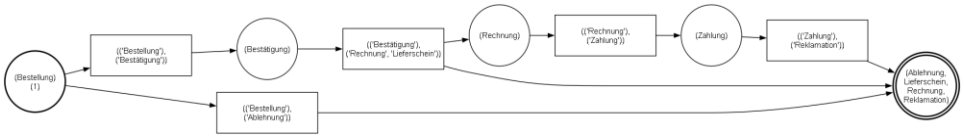


Abbildung 7-14: Aus einem Eventlog abgeleitetes Petri-Netz (Alpha Miner, Export) nach Zusammenführung

Abbildung 7-15 zeigt das Geschäftsprozessmodell aus der Zusammensetzung der zeitlich-basierten Aktivitätstypen basierend auf den Input- und Output-Objekttypen. Bei einem generierten Petri-Netz mit einem Process Discovery Algorithmus sind zusätzlich

die Objekttypen den Stellen zuzuordnen. Die Objekttypen werden anhand der beteiligten Objekttypen der Aktivitätstypen, die im Vor- und Nachbereich einer betrachteten Stelle liegen, zugewiesen.

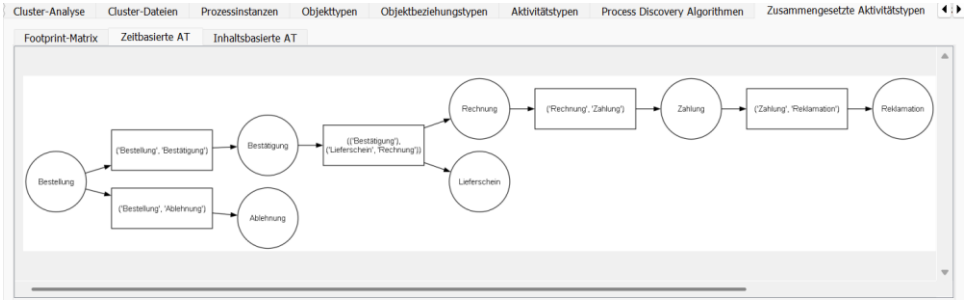


Abbildung 7-15: Aus den zusammengeführten zeitlich-basierten Aktivitätstypen abgeleitetes Petri-Netz

Abbildung 7-16 zeigt das Geschäftsprozessmodell aus der Zusammensetzung der inhaltlich-basierten Aktivitätstypen, basierend auf den Input- und Output-Objekttypen.

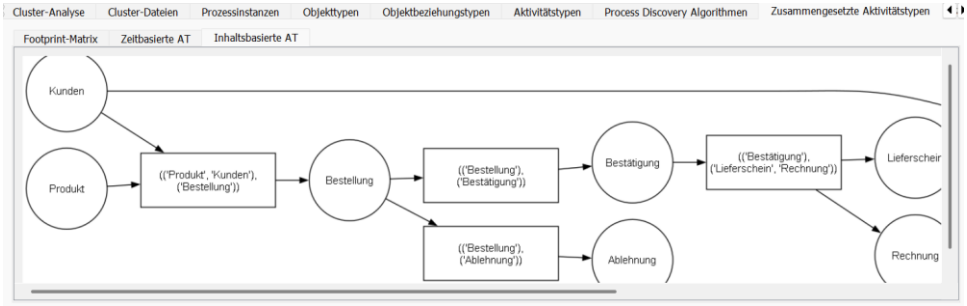


Abbildung 7-16: Aus den inhaltlich-basierten Aktivitätstypen abgeleitetes Petri-Netz

Die Integration von Entscheidungsregeln stellt eine Erweiterung für das generierten Geschäftsprozessmodell dar. Die Regeln ermöglichen es, Entscheidungen im Zusammenhang mit Verzweigungen im Prozessablauf explizit zu machen und für eine automatisierte Ausführung zu nutzen. Die Implementierung folgt dabei einem dreistufigen Ansatz:

1. Identifikation der Entscheidungspunkte
    - Erkennung von XOR-Aufspaltungen im Petri-Netz
2. Informationsextraktion
    - Sammlung der Objektinstanzen der Prozessinstanzen, die den Entscheidungspunkt betreffen
    - Analyse der Zustände der Objektinstanzen vor und nach Entscheidungspunkten

- Identifikation entscheidungsrelevanter Attribute

3. Ableitung von Entscheidungsregeln
   - Transformation in ein Klassifizierungsproblem
   - Anwendung von Klassifizierungsverfahren
   - Validierung der abgeleiteten Regeln

Die erste Phase umfasst die Erkennung von Entscheidungspunkten im Geschäftsprozessmodell. Die Implementierung nutzt hierfür eine strukturelle Analyse des Petri-Netzes, indem die Stellen mit mehreren ausgehenden Kanten als Entscheidungspunkte identifiziert werden.

Für jeden identifizierten Entscheidungspunkt erfolgt anschließend eine detaillierte Analyse der verfügbaren Daten. Diese umfasst die Sammlung aller Objektinstanzen und ihrer Attribute, die zum Entscheidungspunkt verfügbar sind. Anschließend werden die Attribute weiter aufbereitet. Die Implementierung unterscheidet zwischen numerischen Attributen, die direkt verwendet werden können, ordinalen Attributen, deren Ordnungsrelation erhalten bleiben muss und nominalen Attributen, die mittels One-Hot-Encoding [PoPP17] in binäre Attribute transformiert werden. Das One-Hot-Encoding erzeugt für jede mögliche Ausprägung eines nominalen Attributs ein separates binäres Attribut. Dies ist in Abbildung 7-17 dargestellt.

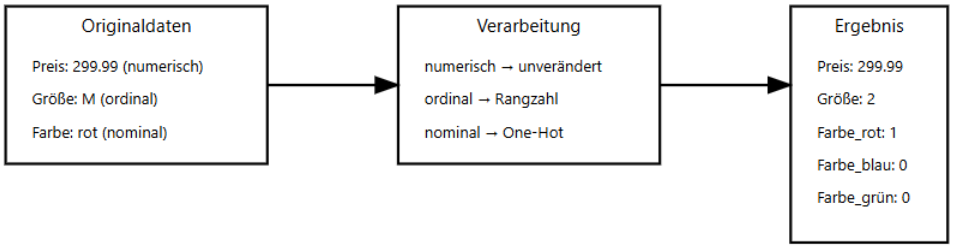


Abbildung 7-17: Aufbereitung der Attributwerte

Die Regelableitung erfolgt durch Überführung in ein Klassifizierungsproblem, bei dem die verschiedenen Entscheidungsalternativen die Zielklassen bilden, während die verfügbaren Attribute als Features dienen. Die Implementierung unterstützt durch die scikit-Bibliothek[7] verschiedene Klassifizierungsverfahren wie Entscheidungsbäume für Regeln, Support Vector Machines für komplexe Entscheidungsgrenzen, Random Forests für robuste Vorhersagen und Naive Bayes für probabilistische Entscheidungen. Die generierten Regeln durchlaufen einen mehrstufigen Validierungsprozess, der die statistische

[7] https://scikit-learn.org/stable/

Güte, praktische Anwendbarkeit, logische Konsistenz und mögliche Nebeneffekte prüft. Abbildung 7-18 stellt einen beispielhaften Entscheidungsbaum dar.

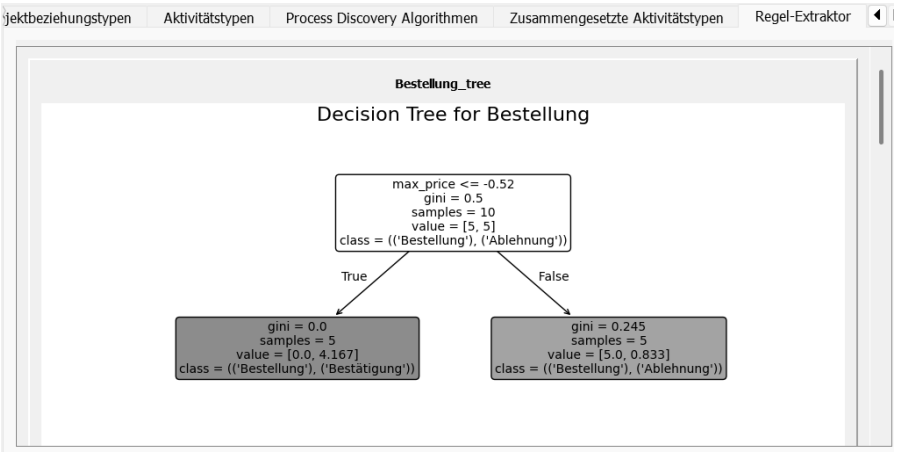


Abbildung 7-18: Entscheidungsbaum eines Entscheidungspunktes

7.3 Erweiterungen des Software-Prototyps

Zur Erweiterung der Methode wurden bereits in Abschnitt 6.3 Möglichkeiten vorgeschlagen, indem die Prozessmodellgenerierung durch weitere Daten unterstützt wird. Um die Anwendbarkeit der Methode zu steigern können auch technische Erweiterungen betrachtet (und ggf. umgesetzt) werden. Dafür wurde zum einen eine iterative Vorgehensweise umgesetzt und die Datenvorverarbeitung weiter betrachtet und implementiert. Diese Implementierungen wurden bisher nur teilweise implementiert, um die Möglichkeit der Erweiterung zu zeigen.

7.3.1 Iterative interaktionsbasierte Vorgehensweise

Basierend auf der Methode lässt sich die Implementierung durch eine iterative Vorgehensweise erweitern, um die Qualität der generierten Geschäftsprozessmodelle weiter zu verbessern. Während die automatisierte Vorgehensweise lediglich die Bereitstellung der Objektinstanz-Dateien erfordert, ermöglicht die iterative Vorgehensweise eine schrittweise Überprüfung und Verfeinerung der Zwischenergebnisse. Die iterative Vorgehensweise folgt dabei dem grundlegenden Ablauf der Informationsextraktion, erlaubt jedoch Eingriffe und Anpassungen an verschiedenen Punkten.

Im ersten Schritt der Objekttyp-Klassifikation kann die gewünschte *Anzahl der Cluster* vorgegeben werden, was direkten Einfluss auf die Granularität der Objekttyp-Erkennung hat. Qualitätsmetriken wie der Silhouette-Koeffizient und die Sum of Squared Errors werden visualisiert, um die Clusterzahl-Entscheidung zu unterstützen.

Die resultierenden Cluster können manuell angepasst werden, indem Dateien anderen oder neuen Clustern zugeordnet werden oder die ermittelte Kategorie (prozessinstanzabhängig/prozessinstanz*un*abhängig) angepasst wird (Abbildung 7-19).

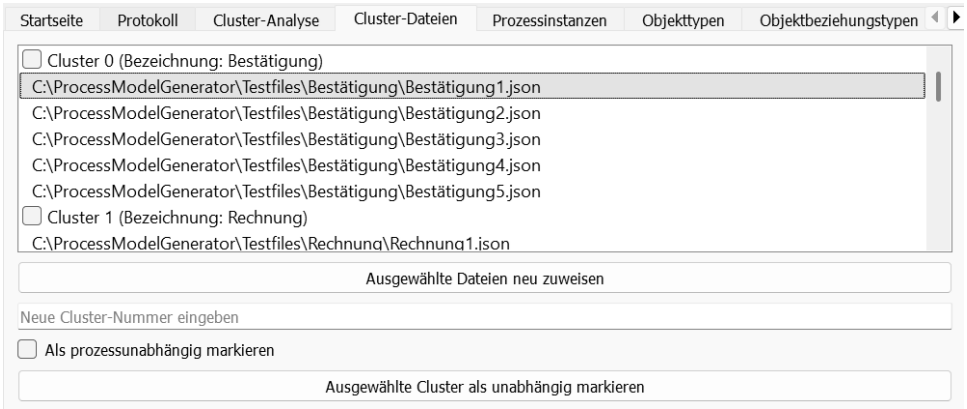


Abbildung 7-19: Anpassung der Cluster-Ergebnisse und der Kategorie[8]

Bei der anschließenden Prozessinstanz-Zuordnung kann der Schwellwert für die *Korrelationserkennung* zwischen Objektinstanzen justiert werden. In der Objekttyp-Definition können die generierten Bezeichnungen angepasst werden.

Für die Ableitung der Objektbeziehungstypen können die Schwellwerte für die Bewertung der Überschneidungen (siehe Abschnitt 7.2.1.4) angepasst werden: *Prozessinstanz-Relevanz*, *Objektinstanz-Relevanz* und *Attributs-Relevanz*.

Anschließend können Objektbeziehungstypen als relevant markiert werden und die Gewichtungsfaktoren (siehe Abschnitt 7.2.1.5) für die Aktivitätstyp-Bewertung konfiguriert werden. Dies beeinflusst die anschließende Auswahl der Aktivitätstypen. Die Bezeichnungen der resultierenden Aktivitätstypen können zusätzlich vor der Transformation angepasst werden.

Diese iterative Vorgehensweise erlaubt es, Domänenwissen systematisch in den Generierungsprozess einzubringen und die Qualität der Zwischenergebnisse zu verbessern. Da

[8] Weitere Screenshots für die einzelnen Interaktionsmöglichkeiten sind in Anhang D dargestellt.

die einzelnen Schritte aufeinander aufbauen, können gezielte Anpassungen früher Zwischenergebnisse die Qualität der nachfolgenden Transformation systematisch verbessern. Die Methode selbst bleibt dabei unverändert - sie wird lediglich um Interaktionsmöglichkeiten erweitert, die eine schrittweise Validierung und Verfeinerung ermöglichen. Dies erhöht die Flexibilität und Anpassbarkeit des Ansatzes an spezifische Anwendungskontexte.

7.3.2 Datenvorverarbeitung

Die Erweiterung der Methode und des Software-Prototyps um die Datenvorverarbeitung soll gewährleisten, dass den weiteren Schritten die bereitgestellten Daten zu den Objektinstanzen in der vorgegebenen Qualität zur Verfügung gestellt werden und diese somit zuverlässig weiterverarbeitet werden können. Durch die Verwendbarkeit von verschiedenen Dateiformaten wird die Anwendbarkeit der Methode gesteigert. Daher wird zunächst betrachtet, wie die Dateien vorliegen können und wie diese vorverarbeitet werden sollten. Durch die Datenvorverarbeitung werden damit insbesondere die Anforderungen an die Datenqualität (AI02), die Anforderungen an das Datenformat (AI03), die Anforderungen an die Funktionale Eignung bezüglich Schritt 2 zum Dateien hochladen und prüfen (AG02) sowie die Anforderungen an den Schritt 3 des Prozessmodellgenerators, die Datenvorverarbeitung selbst (AG03), gewährleistet. In Tabelle 7-5 werden diese Formate nach den Herausforderungen und dadurch notwendigen Vorverarbeitungsschritten betrachtet.

Tabelle 7-5: Herausforderungen und Verarbeitungsschritte für die unterschiedlichen Formate

Format	Herausforderungen	Vorverarbeitungsschritt
PDF	○ Erhalt der Dokumentstruktur ○ Erkennung von Tabellen und Layouts ○ Eingebettete Bilder/Grafiken ○ Verschiedene PDF-Versionen	○ Datenerstellung für Grafiken ○ Textextraktion ○ Strukturextraktion ○ Tabellenextraktion ○ Layout-Analyse
DOCX	○ Formatierungen ○ Eingebettete Objekte ○ Dokumentversionen ○ Metadaten	○ Textextraktion ○ Formatierungserhalt ○ Datenerstellung für Grafiken

Format	Herausforderungen	Vorverarbeitungsschritt
PPTX	○ Komplexe Layouts ○ Grafische Elemente ○ Animationen ○ Notizen	○ Layout-Analyse ○ Textextraktion ○ Bildverarbeitung
XLSX	○ Formeln ○ Mehrere Tabellenblätter ○ Verknüpfungen ○ Formatierungen	○ Tabellenstrukturanalyse ○ Formelextraktion ○ Datentyperkennung
JSON	○ Verschachtelte Strukturen ○ Datentypen ○ Kodierung	○ Strukturvalidierung ○ Typkonvertierung ○ Normalisierung
XML	○ Schema-Konformität ○ Namespaces ○ Referenzen	○ Schema-Validierung ○ Namespace-Auflösung ○ Referenzprüfung
JPG	○ Bildqualität ○ Textelemente im Bild ○ Metadaten	○ Bildvorverarbeitung ○ Datenerstellung ○ Metadatenextraktion

Diese betrachteten Dateiformate und notwendigen Vorverarbeitungsschritte unterscheiden sich darin, ob die Dateien unmittelbar maschinenlesbar und strukturiert sind. Ziel der Datenvorverarbeitung ist es, aus den bereitgestellten Dateien, die Objektinstanzen eines Geschäftsprozesses in einheitlicher Qualität zu erhalten. Die erforderlichen Vorverarbeitungsschritte sind in Tabelle 7-6 für unterschiedlichen Qualitätsstufen der bereitgestellten Dateien beschrieben.

Tabelle 7-6: Schritte der Datenvorverarbeitung

Vorverarbeitungs-schritt	Beschreibung
1. Datenerstellung	Nicht unmittelbar maschinenlesbare unstrukturierte Dateien.
	Dateien, deren Daten weder strukturiert noch sofort maschinenlesbar sind, erfordern eine zusätzliche Verarbeitung, z.B. OCR für gescannte Dokumente oder Texterkennung in Bildern. Für diese Dateien ist eine Datenerstellung durchzuführen, um unmittelbar maschinenlesbare unstrukturierte Daten zu erhalten.
2.	Unmittelbar maschinenlesbare unstrukturierte Dateien.

Vorverarbeitungs-schritt	Beschreibung
Datenstrukturierung	Dateien wie Textdateien sind unstrukturiert, aber dennoch maschinenlesbar. Diese Dateien stellen die relevanten Informationen in keinem einheitlichen Format bereit, sodass keine Weiterverarbeitung gewährleistet werden kann. Für diese Dateien ist daher eine Datenstrukturierung durchzuführen, um unmittelbar maschinenlesbare (semi-)strukturierte Daten zu erhalten.
	Unmittelbar maschinenlesbare (semi-)strukturierte Dateien.
3. Datenbereinigung	Dateien, die bereits in einem strukturierten oder semi-strukturierten Format vorliegen und direkt von Maschinen verarbeitet werden können, wie z.B. CSV-Dateien oder XML-Daten, sollen auf eine Datenbereinigung geprüft werden, um eine bessere Qualität der Ergebnisse zu ermöglichen. Dazu werden die Dateien so aufbereitet, dass einzelne Objektinstanzen nach der Definition 6-4 vorliegen.

Daher werden folgend Ansätze zur Datenerstellung (siehe Abschnitt 7.3.2.1) und Datenstrukturierung (siehe Abschnitt 7.3.2.2) sowie Schritte zur Datenbereinigung (siehe Abschnitt 7.3.2.3) betrachtet.

7.3.2.1 Datenerstellung

Wie im Anwendungsfall in Kapitel 5 beschrieben, können zur Prozessmodellgenerierung mehrere Dateien in unterschiedlichen vorgegebenen Formattypen hochgeladen werden. Zur Weiterverarbeitung sollten diese jedoch so vorliegen, dass auf die enthaltenen Informationen zu den Objektinstanzen zugegriffen werden kann. Daher wird in der Datenerstellung zunächst betrachtet, wie aus noch nicht unmittelbar maschinenlesbaren Dateien maschinenlesbare Daten erstellt werden können. Damit diese anschließend auch in einheitlicher Qualität als Daten zu Objektinstanzen vorliegen, werden die Daten nach der Datenerstellung einer Datenstrukturierung und -bereinigung unterzogen, die in Abschnitt 7.3.2.2 und 7.3.2.3 betrachtet wird.

Werden nicht unmittelbar maschinenlesbare Daten zur Verfügung gestellt, können Ansätze zur automatischen Erkennung von Zeichen mittels optischer Mechanismen wie beispielsweise Optical Character Recognition (OCR) [HaKa16] oder Visual Document Understanding-Modelle (VDU-Modelle) [FBZN+24] verwendet werden. Durch die Datenerstellung mittels optischer Mechanismen kann sowohl maschinell gedruckter Text als auch handschriftlicher Text auf einem Bild identifiziert und in maschinenlesbaren Text umgewandelt werden. Durch die Texterkennung und die anschließende Speicherung in maschinenlesbare Form, können die erstellten Texte weiterverarbeitet werden. Die

Schritte der OCR, um nach einer Bildaufnahme Texte in maschinenlesbare Form zu erhalten, sind [HaKa16]:

1. Pre-Processing: Das Eingabebild wird durch Anpassungen wie der Helligkeit, des Kontrastes oder des Bildausschnitts vorverarbeitet, um die Ergebnisse der Texterkennung zu verbessern ohne relevante Informationen zu verlieren.
2. Segmentierung: Bei der Segmentierung wird das Bild in einzelne Teilbereiche aufgeteilt, die anschließend separat weiterverarbeitet werden. Dazu wird die Segmentierung auf Zeilenebene, Wortebene und Zeichenebene vorgenommen.
3. Merkmalsextraktion: Bei der Merkmalsextraktion werden aus jeder, im vorherigen Schritt segmentierten Komponente spezifische Merkmale abgeleitet, anhand derer im nächsten Schritt die Klassifizierung stattfindet. Merkmalsarten sind das Bild selbst, geometrische Merkmale (z. B. Schleifen, Striche) und statistische Merkmale (z. B. Schriftgröße, Zeichenabstand).
4. Klassifizierung: Anhand der identifizierten Merkmale erfolgt die Einordnung der segmentierten Zeichen zu einer bestimmten Klasse. Hierfür existieren verschiedene Klassifizierungsverfahren. Strukturelle Klassifizierungsverfahren basieren auf den Merkmalen, die aus der Struktur des Bildes extrahiert werden. Statistische Klassifizierungsverfahren beruhen auf probabilistischen Modellen und Verwendung von Unterscheidungsfunktionen. Weiter werden in der Klassifizierung erkannte Zeichen anhand sprachspezifischer Modelle und Wörterbücher zu Wörtern zusammengefügt.
5. Post-Processing: Einige Fehler bei der OCR sind durch eine falsche Klassifikation der Zeichen begründet, die wiederum durch eine falsche Merkmalsextraktion oder Rauschen im Bild zurückzuführen sind. Fehler in der OCR zeigen sich z.B. durch Grammatikfehler. Grammatikfehler können im Rahmen des Post-Processing mithilfe von Sprachmodellen, Wörterbüchern oder Grammatikhilfen verbessert werden.

Da die Qualität der Ergebnisse bei OCR-basierten Ansätze insbesondere bei der Verarbeitung verschiedener Sprachen oder Dokumentformaten nicht gewährleistet werden kann, schlagen [FBZN+24] VDU vor. Diese Modelle nutzen künstliche Intelligenz wie Convolutional Neural Networks (CNNs) und generative Transformer-Modelle, um Merkmale aus visuellen Inhalten zu extrahieren.

Durch die Anwendung von OCR bzw. VDU werden aus der Eingabe eines Dokumentes, das nicht unmittelbar maschinenlesbar ist (wie beispielsweise einige Dateien in PDF oder JPG-Format) maschinenlesbare Daten erstellt. Diese Datenerstellung ist in Abbildung 7-20 anhand der Verarbeitung einer gescannten Rechnung beispielhaft dargestellt.

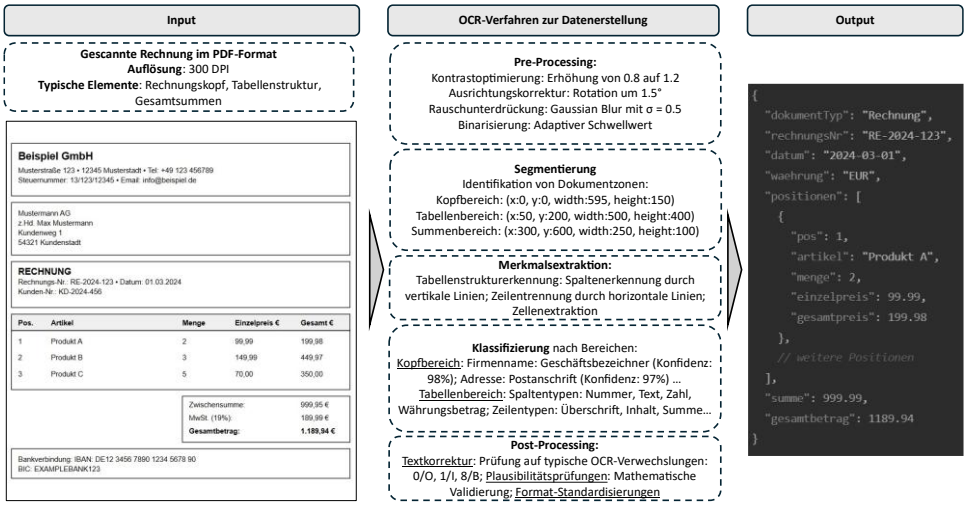


Abbildung 7-20: Rechnungsverarbeitung

Die Strukturierung des Dokuments entspricht dem Layout einer Geschäftsrechnung und wird daher zur Demonstration der OCR-Verarbeitungsschritte verwendet. Die verschiedenen Bereiche sind klar voneinander abgegrenzt, was die Segmentierung erleichtert.

7.3.2.2 Datenstrukturierung

Für die Informationsextraktion aus den Daten sollen die Daten in ein geeignetes Format oder eine geeignete Struktur gebracht werden. Oftmals liegen Daten in verschiedenen Formaten, Strukturen und Qualitäten vor und müssen zu einem einheitlichen Format zusammengeführt werden. Die Herausforderung besteht darin, diese heterogenen Datenquellen in ein einheitliches, maschinenverarbeitbares Format zu überführen, ohne dabei wesentliche Informationen zu verlieren. Dies kann beispielsweise das Zusammenführen von Tabellen, die Umwandlung von Datentypen oder das Skalieren von Werten umfassen. Für die Informationsextraktion (siehe Abschnitt 6.1) sollen in dieser Arbeit die relevanten Informationen der Objektinstanzen als Attribut-Wert-Paare gespeichert sein. Attribut-Wert-Paare ermöglichen eine einheitliche Repräsentation von Informationen aus unterschiedlichen Dateien. Durch eine Strukturierung lassen sich die Informationen sowohl aus strukturierten Datenbanken, semi-strukturierten Dokumenten oder auch unstrukturierten Texten in Attribut-Wert-Paaren darstellen. Jedes Attribut-Wert-Paar bildet eine in sich geschlossene Informationseinheit zu einer Objektinstanz. Dies erleichtert die Verarbeitung und gewährleistet die Eindeutigkeit der Information. Zusätzlich können Objektinstanzen durch die Ergänzung um neue Attribut-Wert-Paare erweitert werden, ohne bestehende Strukturen ändern zu müssen. Die klare Struktur ermöglicht eine effiziente maschinelle Verarbeitung, sei es durch regelbasierte Systeme oder maschinelle Lernverfahren. Dadurch können auch für die Generierung relevante Über-

schneidungen in den Objekten festgestellt sowie Redundanzen erkannt und eliminiert werden. Durch die eindeutige Zuordnung von Werten zu Attributen wird eine kontextunabhängige Interpretierbarkeit und eine standardisierte Verarbeitung ermöglicht. Diese Strukturierung kann außerdem eine Konvertierung zwischen verschiedenen Formaten gewährleisten.

Damit die Strukturierung die maschinenlesbaren Daten so vorverarbeitet, dass diese für die Weiterverarbeitung verwendet werden können, ohne Informationen zu verändern oder zu entfernen, sind die in Tabelle 7-7 genannten Prinzipien zu beachten [SmSm77, AbHV95]:

Tabelle 7-7: Prinzipien der Datenstrukturierung und deren Begründung

Prinzip	Beschreibung	Begründung
1	Hierarchische Organisation • Logische Gruppierung zusammengehöriger Informationen • Zuordnung von Attributen zu Objektinstanzen • Verschachtelung von Attributen für komplexe Strukturen • Eindeutige Zuordnung von Werten zu Attributen	• Verbesserte Übersichtlichkeit • Einfachere Wartbarkeit • Natürliche Abbildung von Beziehungen
2	Atomare Informationseinheiten • Jedes Attribut-Wert-Paar enthält eine einzelne Information • Keine Vermischung verschiedener Informationstypen • Eindeutige Interpretierbarkeit	• Eindeutige Verarbeitung möglich • Vereinfachte Datenpflege • Bessere Suchfunktionalität
3	Semantische Strukturierung • Beibehaltung der inhaltlichen Bedeutung • Kontexterhaltung durch geeignete Attributnamen • Logische Gruppierung zusammengehöriger Informationen	• Verbesserte Verständlichkeit • Fachliche Nachvollziehbarkeit • Einfachere Weiterverarbeitung

Abbildung 7-21 zeigt die Strukturierung von maschinenlesbaren Daten in Attribut-Wert-Paare. Dabei werden die Informationen hierarchisch in Metadaten, Artikelinformationen und Lieferadresse aufgebaut (Prinzip 1). Jedes Attribut enthält dabei einen Wert (Prinzip

2) und die Informationen der maschinenlesbaren Daten bleiben semantisch erhalten (Prinzip 3).

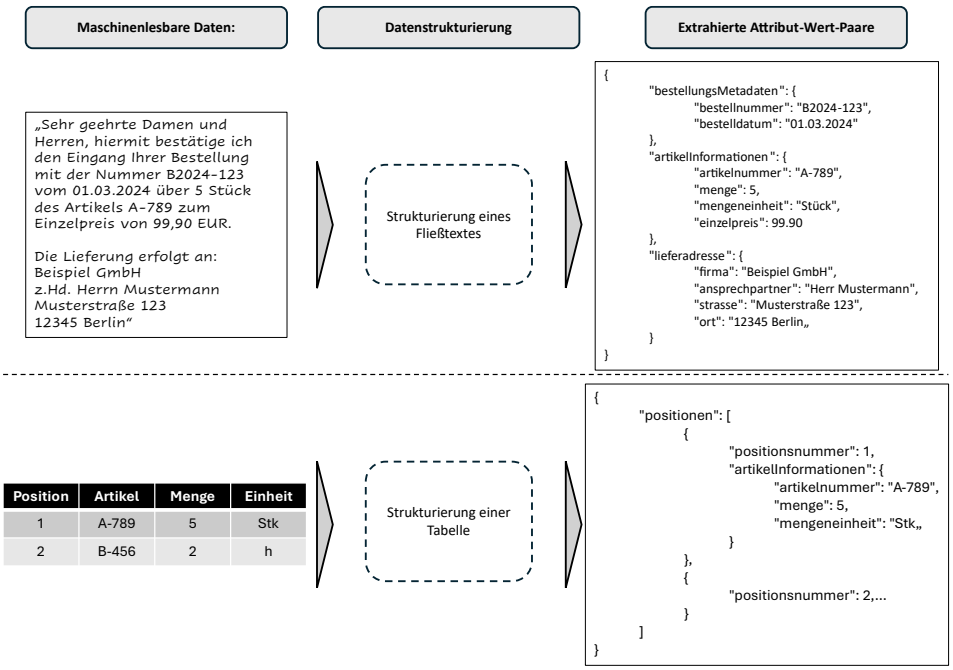


Abbildung 7-21: (Um-)Strukturierung von Daten

Um maschinenlesbare Daten zu strukturieren, können regelbasierte Verfahren angewandt werden, die nach definierten Mustern in Dokumenten suchen. Dazu werden beispielsweise bestimmte Attributnamen, Datentypen und Textmuster definiert. Beispielsweise kann nach Wörtern wie *ID* und/oder *Nummer* gesucht werden, um eine Identifikationsnummer eines Objektes zu ermitteln. Bei Datentypen können unterschiedliche Datumsformate definiert und auf Existenz geprüft werden. Außerdem können beispielsweise Muster für E-Mail-Adressen und Telefonnummern definiert werden, deren Vorhandensein in den Daten geprüft werden soll. Ebenso könnte bei Vorliegen einer zweispaltigen Tabelle beispielsweise deren Inhalt in Attribut-Wert-Paare umgewandelt werden, indem der linke Eintrag dem Attributnamen und der rechte Eintrag dem Wert entspricht. Zur Erkennung solcher Attribut-Wert-Paaren mit maschinellen Lernverfahren wird es als Sequenzkennzeichnungsproblem betrachtet. Dazu gehören die Zuordnung von Wortarten (Part-of-Speech-Tagging), Chunking und die Erkennung benannter Entitäten. Es wird jedem Wert ein Attribut zugewiesen, welches auch von anderen Werten und Attributen in einer Sequenz abhängen kann [Jian12]. Außerdem existieren auch Verfahren, die mit generativen Transformer-Modellen beispielsweise unvorhersehbare Strukturen oder auch Fließ-

text verarbeiten, um Attribute und Werte aus Fließtexten zu extrahieren [EHSR+22, IMDD+23].

7.3.2.3 Datenbereinigung

Um die Qualität der Ergebnisse einer Weiterverarbeitung der Daten zu verbessern, die in einem maschinenlesbaren Datenformat mit vorgegebener Struktur vorliegen, sollen die Daten bereinigt werden. Durch diese Bereinigung sollen mögliche Fehler, Ausreißer oder unvollständige Werte der bereitgestellten Daten, die die Generierung negativ beeinflussen können, entfernt, verbessert oder ergänzt werden. Durch Bereinigungsschritte wie das Entfernen von Duplikaten, das Auffüllen fehlender Werte oder das Behandeln von Ausreißern können zuverlässigere Ergebnisse erzielt werden. Es gibt verschiedene Möglichkeiten der Datenvorverarbeitung, die je nach den spezifischen Anforderungen und dem Kontext variieren können. Einige gängige Vorverarbeitungsschritte umfassen [AgZh12, BlAt03, WRPM22]:

- Datenanpassung: Entfernung von Duplikaten, Behandlung von fehlenden Werten oder Ausreißern.
- Datenintegration: Zusammenführung von Daten aus verschiedenen Quellen oder Tabellen.
- Datentransformation: Umwandlung von Daten in ein geeignetes Format oder eine geeignete Skala.
- Merkmalsextraktion: Schaffung neuer Merkmale aus vorhandenen Daten oder Merkmalen.
- Dimensionalitätsreduktion: Verringerung der Anzahl der Merkmale durch Techniken wie Hauptkomponentananalyse (PCA) oder Dimensionsreduktion.
- Normalisierung und Skalierung: Anpassung der Werte, um sicherzustellen, dass sie auf einer vergleichbaren Skala liegen.
- Datenaggregation: Zusammenfassen von Daten auf einer höheren Aggregationsebene, um die Datenmenge zu reduzieren und wichtige Informationen zu bewahren.
- Datensegmentierung: Unterteilung der Daten in bestimmte Segmente basierend auf vordefinierten Kriterien, um spezifische Analysen auf Teilmengen der Daten durchzuführen.
- Datenauswahl: Auswahl relevanter Datenattribute oder Merkmale, die für die Analyse oder das Modell von Interesse sind, um die Datenmenge zu reduzieren oder den Fokus zu schärfen.
- Datenbalancierung: Ausgleich von Klassenungleichgewichten in den Daten, insbesondere bei Aufgaben wie Klassifikation, um sicherzustellen, dass das Modell alle Klassen angemessen berücksichtigt.

Dies auszuwählenden Vorverarbeitungsschritte hängen von den spezifischen Daten und Zielen ab. Insgesamt dient die Datenvorverarbeitung dazu, die Qualität, Konsistenz und Relevanz der Daten zu verbessern. Durch eine sorgfältige Datenvorverarbeitung können potenzielle Probleme frühzeitig erkannt und behoben werden, um genaue, aussagekräftige und verlässliche Ergebnisse zu erzielen. Ohne eine angemessene Datenvorverarbeitung können die Ergebnisse von Analysen oder Modellen unzuverlässig oder sogar irreführend sein. Daher ist die Datenvorverarbeitung ein essenzieller Schritt, um den vollen Nutzen aus den vorhandenen Daten zu ziehen und zu einer erfolgreichen Prozessmodellgenerierung beizutragen.

Die regelbasierte Bereinigung von Daten beinhaltet die Erarbeitung und Definition spezifischer Regeln, die auf die Daten angewandt werden, um sie zu bereinigen und zu transformieren. Diese Regeln sind basierend auf dem vorhandenen Wissen über die Daten und die spezifischen Anforderungen der Analyse oder Modellierung zu entwickeln. Ein Beispiel für regelbasierte Bereinigung ist die Behandlung von fehlenden Werten. Hier kann eine Regel festgelegt werden, dass fehlende Werte durch den Durchschnitt oder den Median der vorhandenen Werte ersetzt werden sollen. Ebenso könnten Wertbereiche definiert werden, um Werte, die außerhalb des definierten Bereichs liegen, zu entfernen.

Der Vorteil der regelbasierten Bereinigung liegt in der Kontrolle und dem Verständnis über die angewandten Transformationen. Durch die explizite Definition der Regeln kann sichergestellt werden, dass die erforderlichen Änderungen an den Daten vorgenommen werden. Allerdings erfordert dieser Ansatz oft eine gründliche Analyse der Daten und eine manuelle Erstellung der Regeln, was zeitaufwändig sein kann und Wissen über Daten erfordert. Daher werden auch KI-basierte Ansätze zur Bereinigung angewandt.

- Clustering: Durch Clustering-Techniken wie k-means oder hierarchisches Clustering können ähnliche Datenpunkte gruppiert werden. Dies kann hilfreich sein, um Ausreißer zu identifizieren oder Datenpunkte mit ähnlichen Merkmalen zu aggregieren [PVGM+11].
- Textanalyse und NLP-Techniken: Bei der Vorverarbeitung von Textdaten können spezifische Techniken des Natural Language Processing (NLP) angewandt werden, um Text in eine geeignete Form zu bringen. Dazu gehören Tokenisierung, Part-of-Speech-Tagging, Named Entity Recognition, Sentimentanalyse und weitere Methoden zur Textklassifikation oder Textverarbeitung [MLTA19].
- Sprachmodelle: Bei einer Bereinigung der Daten durch generative Transformer-Modelle werden die einzelnen anzuwendenden Regeln nicht definiert, sondern es kann festgelegt werden, wie das Ergebnis sein soll. Auf diese Weise kann das Modell beispielsweise automatisch fehlende Werte ergänzen, Ausreißer erkennen und behandeln oder Daten in das gewünschte Format umwandeln. Ein Beispiel für die Vorverarbeitung durch Sprachmodelle ist die Textnormalisierung. Hierbei

> kann das Modell darauf trainiert werden, Abkürzungen zu erkennen und in ihre vollständigen Formen umzuwandeln, Schreibfehler zu korrigieren oder bestimmte sprachliche Nuancen anzupassen. Das Modell kann auch lernen, Stoppwörter oder irrelevante Informationen zu entfernen und die Textdaten in einen standardisierten und konsistenten Zustand zu bringen [FBZN+24].

Sowohl die regelbasierte Vorverarbeitung als auch die Vorverarbeitung durch Sprachmodelle ermöglichen die Bereinigung, Konvertierung und Strukturierung von Daten, um sie für weiterführende Analysen oder Modellierungsaufgaben zu nutzen. Diese Bereinigung ist dabei sowohl auf die einzelnen Daten ansich, als auch im gesamten zu betrachten. Auf Instanzebene müssen fehlerhafte oder doppelte Werte betrachtet werden. Für die gesamten Daten sind dabei weitere Bereinigungen vorzunehmen. Dabei ist insbesondere die einheitliche Bezeichnung von Daten ein fundamentaler Aspekt der Datenbereinigung. Zur Standardisierung der Bezeichnung zählt eine konsistente Schreibweise (beispielsweise für Identifikatoren: *ID* vs. *Id* vs. *id* vs. *Identifier*) sowie einheitliche Datumsformate bis hin zur Normalisierung von Maßeinheiten. Bei den Namenskonventionen sollen Standards verwendet sowie eine einheitliche Behandlung von Sonderzeichen und Leerzeichen berücksichtigt werden. Ebenso sollte eine konsistente Verwendung von Präfixen und Suffixen eingehalten werden. Dazu können dokumentierte Benennungskonventionen mit einheitlichen Abkürzungen und Akronyme und die Erstellung eines umfassenden Glossars (ergänzt durch Mapping-Tabellen für verschiedene Bezeichnungen) dienen. Dabei sollen Änderungen für alle Objekte einheitlich vorgenommen werden, um anschließend in der Informationsextraktion noch basierend auf Unterschieden und Übereinstimmungen Zusammenhänge ableiten zu können.

Abbildung 7-22 beschreibt eine beispielhafte Datenbereinigung mit ausgewählten Regeln zur Bereinigung. Die Regeln könnten dabei in einer Konfigurationsdatei definiert werden und können bei Bedarf erweitert oder angepasst werden. Reguläre Ausdrücke und Pattern Matching werden verwendet, um die verschiedenen Eingabeformate zu erkennen und zu bereinigen.

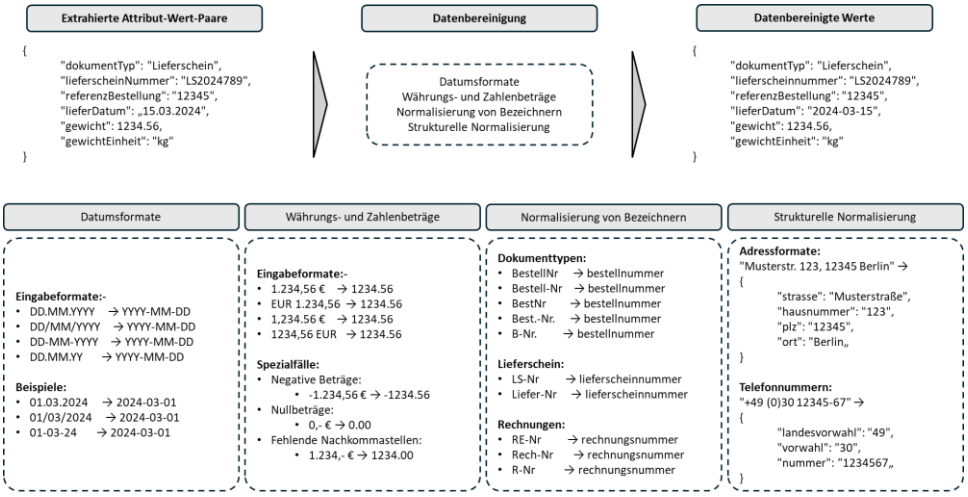


Abbildung 7-22: Exemplarische Datenbereinigung

7.4 Zusammenfassung

In diesem Kapitel wurde die Implementierung eines Software-Prototyps zur automatisierten Generierung von Geschäftsprozessmodellen aus Objektinstanzen vorgestellt. Der Software-Prototyp setzt die Schritte der Methode zur Prozessmodellgenerierung aus Kapitel 6 um:

In der *Informationsextraktion* werden die Objekttypen, Objektbeziehungstypen und Aktivitätstypen aus den Objektinstanzen abgeleitet.
In der *Transformation* werden die extrahierten Informationen in einem Geschäftsprozessmodell in Form eines Petri-Netz repräsentiert.

Der *PreInstanceGenerator* übernimmt die initiale Datenvorverarbeitung. Die Informationsextraktion wird durch den *ObjectTypeGenerator* zur Klassifikation der Objekttypen, den *ProcessInstanceClassifier* zur Zuordnung von Objektinstanzen zu Prozessinstanzen, den *ObjectRelationshipTypeGenerator* zur Analyse der Objektbeziehungstypen und den *ActivityTypeGenerator* zur Ableitung der Aktivitätstypen realisiert. Der *PNGenerator* generiert aus den extrahierten Informationen ein Geschäftsprozessmodell in Form eines Petri-Netzes. Die Darstellung der Ergebnisse aus den einzelnen Schritten sowie die Integration von detailierten Terminal-Ausgaben und eine umfangreiche Protokollierung unterstützen die Nachvollziehbarkeit der Prozessmodellgenerierung. Die Implementierung nutzt Python-Bibliotheken wie Pandas für die Datenverarbeitung, NetworkX für Graph-Analysen, PM4Py für Process Mining und scikit-learn für maschinelles Lernen.

Die modulare Architektur ermöglicht Erweiterungen um weitere Funktionalitäten. Dies wurde gezeigt, indem der Prototyp bereits um Funktionalitäten zur iterativen und interaktionsbasierten Vorgehensweise erweitert wurde. Zusätzlich wurden prototypische Ansätze mit OCR zur Verarbeitung nicht-maschinenlesbare Formate und verschiedene Mechanismen zur Strukturierung und Bereinigung der extrahierten Daten integriert. Diese Erweiterungen erhöhen die Anwendbarkeit des Software-Prototyps durch Unterstützung unterschiedlicher Eingabeformate bei gleichzeitiger Sicherstellung einer einheitlichen Datenqualität für die Prozessmodellgenerierung. In Tabelle 7-8 wurden die Schritte der Datenvorverarbeitung zusammengefasst.

Tabelle 7-8: Schritte der Datenvorverarbeitung inklusive des jeweiligen Inputs und Outputs

	Schritt	Input	Output
Datenvorverarbeitung	Datenerstellung (OCR/VDU).	Nicht-unmittelbar maschinenlesbare Daten (beispielsweise JPG)	Maschinenlesbare Daten
	Datenstrukturierung (Umwandlung in Attribut-Wert-Paare).	Maschinenlesbare Daten (beispielsweise DOCX, PDF)	Strukturierte Attribut-Wert-Paare
	Datenbereinigung (Einheitliche Namensgebung und Zeitformate, Qualitätsbetrachtung).	Strukturierte Attribut-Wert-Paare (beispielsweise XML, JSON)	Bereinigte, qualitätsgesicherte Daten als Attribut-Wert-Paare

Damit stellt der implementierte Prototyp einen erweiterbaren Rahmen für die automatisierte Prozessmodellierung bereit, der sowohl die in Kapitel 6 beschrieben Methode umsetzt als auch praktische Anforderungen wie Erweiterbarkeit und Wartbarkeit berücksichtigt.

8 Evaluation

Die Evaluation der entwickelten Methode erfolgt sowohl konzeptionell als auch anhand des Software-Prototyps. Dabei werden die zentralen Schritte der Prozessmodellgenerierung - die *Informationsextraktion* und die *Transformation* - systematisch untersucht. Zusätzlich wird der Erfüllungsgrad der in Kapitel 5 definierten Anforderungen bewertet. Diese Evaluation verfolgt die Überprüfung der aufgestellten Anforderungen und validiert damit die Fähigkeit der Methode, aus Objektinstanzen Geschäftsprozessmodelle zu generieren, die den definierten Qualitätsanforderungen genügen. Gleichzeitig dient sie der Identifikation von Weiterentwicklungsmöglichkeiten und Verbesserungspotenzialen der Methode und des Software-Prototyps [SHPR+11].

Für die systematische Bewertung wird eine dreistufige Evaluationsskala verwendet. Soweit es für die einzelnen Aspekte möglich ist, werden die Methode (M) und der Software-Prototyp (P) differenziert betrachtet:

- Vollständig erfüllt:
  - $\checkmark_M$: Die Anforderung wird von der Methode erfüllt.
  - $\checkmark_P$: Der Software-Prototyp setzt die Anforderung vollständig um
- Teilweise erfüllt ($\sim_M$):
  - $\sim_M$: Die Methode erfüllt wesentliche Aspekte der Anforderung, weist aber noch Lücken auf.
  - $\sim_P$: Der Software-Prototyp implementiert die Anforderung teilweise, einzelne Aspekte fehlen.
- Potenziell erfüllbar (χ_M):
  - χ_M : Die Methode erfüllt die Anforderung aktuell nicht, könnte aber konzeptionell erweitert werden.
  - χ_P : Der Software-Prototyp implementiert die Anforderung nicht, eine Umsetzung wäre aber technisch möglich.

Die Evaluierung erfolgt strukturiert entlang der drei in Kapitel 5 definierten Anforderungskategorien: Anforderungen an die Input-Daten (Abschnitt 8.1), Anforderungen an das generierte Modell (Abschnitt 8.2) sowie Anforderungen an den Generierungsprozess (Abschnitt 8.3).

Ergänzend wird in Abschnitt 8.4 die Methode hinsichtlich ihrer Fähigkeit zur Abbildung der grundlegenden Kontrollflussmuster evaluiert. Es wird geprüft, ob und wie die Metho-

de grundlegende Kontrollflussmuster wie Sequenz, Alternative und Nebenläufigkeit korrekt erkennt und im generierten Modell abbildet.

Die durchgeführte Evaluation unterliegt verschiedenen Limitationen, die bei der Interpretation der Ergebnisse berücksichtigt werden müssen. Eine zentrale Einschränkung ergibt sich aus dem prototypischen Charakter der Implementierung. Der entwickelte Software-Prototyp dient primär dem Nachweis der technischen Machbarkeit und implementiert die Kernfunktionalitäten der Methode, ist jedoch nicht als produktionsreifes System konzipiert.

Eine weitere Limitation betrifft die externe Validität der Ergebnisse. Die Evaluation basiert auf ausgewählten Testszenarien, die zwar typische Geschäftsprozessstrukturen abdecken, aber auch nur ausgewählte Anwendungsfälle repräsentieren. Um die Generalisierbarkeit der Methode auf beliebige Geschäftsprozesse zu validieren, sind weitere empirische Studien mit unterschiedlichen Geschäftsprozessen und realen Anwendungskontexten erforderlich.

Die Evaluation konzentriert sich zudem hauptsächlich auf funktionale Anforderungen, während nicht-funktionale Aspekte weitgehend unberücksichtigt bleiben. So wurden Performance und Skalierbarkeit der Implementierung nicht systematisch untersucht. Auch die Benutzerfreundlichkeit des Systems wurden nicht evaluiert. Weitere wichtige nicht-funktionale Anforderungen wie die Integration in bestehende Systemlandschaften sowie die langfristige Wartbarkeit und Erweiterbarkeit der Lösung wurden im Rahmen dieser Arbeit weiter nicht betrachtet.

Diese Limitationen bieten wichtige Ansatzpunkte für zukünftige Forschungsarbeiten. Eine umfassendere Evaluation unter Einbeziehung nicht-funktionaler Anforderungen sowie Studien in realen Anwendungsszenarien würden das Verständnis der praktischen Einsetzbarkeit der entwickelten Methode weiter vertiefen.

8.1 Evaluierung der Anforderungen an die Input-Daten

In Abschnitt 5.1 werden die Anforderungen an die Input-Daten definiert, die auf den in Kapitel 3 betrachteten Daten zur Geschäftsprozessmodellierung aufbauen. Es wurden die drei Anforderungen *Informationsgehalt* (AI01), *Datenqualität* (AI02) und *Datenformat* (AI03) definiert.

Informationsgehalt (AI01): Für diese Anforderung ist das Vorhandensein der Daten sowie der notwendigen Informationen zu betrachten. Zur Erfüllung dieser Anforderung

muss sichergestellt werden, dass die erforderlichen Informationen zur Ableitung von Aktivitätstypen und deren Ordnungsbeziehungen aus Objektinstanzen extrahiert werden können.

Evaluation Methode: Die Methode basiert auf Daten zu Objektinstanzen. Der Zustand einer Objektinstanz zu einem bestimmten Zeitpunkt t ist über Attribut-Wert-Paare beschrieben. Basierend auf den Zuständen der Objektinstanzen werden anhand der Methode Objekttypen erkannt und durch verschiedene Schritte Aktivitätstypen und deren Ordnungsbeziehungen abgeleitet. Diese Ableitung basiert auf der Annahme, dass Objektinstanzen desselben Objekttyps *ähnliche Objektstrukturen* haben. Anhand von *übereinstimmenden Attribut-Wert-Paaren* zwischen Objektinstanzen unterschiedlicher Objekttypen wird die Prozessinstanz-Zuordnung durchgeführt. Zusätzlich wird für die Ableitung ein *repräsentativer Zeitpunkt* für die Prozessinstanzobjekte ermittelt. Nach den Definitionen in Kapitel 6, die anhand der Betrachtung von Geschäftsprozessen und Objekten abgeleitet wurden, erfüllen die Objekte, die in Geschäftsprozessen, durch Aktivitäten erzeugt, bearbeitet, gelesen oder verbraucht werden, diese Annahmen. Somit erfüllt die Methode diese Anforderung vollständig (AI01: $\checkmark_M$).

Evaluation Software-Prototyp: Der Software-Prototyp stellt eine Benutzungsoberfläche bereit, über die Dateien hochgeladen werden können. Die hochgeladenen Dateien werden zunächst auf Verwendbarkeit geprüft und gegebenenfalls vorverarbeitet. Nicht verwendbare Dateien werden dabei protokolliert und von der weiteren Verarbeitung ausgeschlossen. Bei der Informationsextraktion werden die bereitgestellten Dateien automatisiert analysiert, um die notwendigen Informationen zur Prozessmodellgenerierung abzuleiten. Somit setzt die Implementierung die entwickelte Methode um. Durch die Erweiterung des Software-Prototyps in Abschnitt 7.3.1 können bei Problemen in den Daten Verbesserungen vorgenommen werden. Die identifizierten Probleme umfassen zu ähnliche Objektstrukturen verschiedener Objekttypen, zu unterschiedliche Objektstrukturen der Objektinstanzen eines Objekttyps, falsche Erkennung von prozessinstanzunabhängigen Objekten, falsche Erkennung eines repräsentativen Zeitpunktes für eine Objektinstanz oder fehlerhafte Zuordnung zu Prozessinstanzen. Diese Verbesserungsmaßnahmen betreffen somit exakt die notwendigen Annahmen, die in der Methode für die Daten zu Objektinstanzen erfüllt sein müssen. Zusätzlich verarbeitet der Software-Prototyp die Dateien schrittweise und stellt durch die mehrstufige Analyse sicher, dass auch bei teilweise fehlenden Informationen eine Prozessmodellgenerierung erfolgt. Fehlende oder unzureichende Informationen werden dabei protokolliert. Durch diese umfassende Implementierung der Datenverarbeitung und der Validierungsmöglichkeiten von Zwischenergebnissen erfüllt der Software-Prototyp die Anforderung AI01 vollständig (AI01: $\checkmark_P$).

Datenqualität (AI02): Diese Anforderung betrifft die Qualität der bereitzustellenden Daten gemäß den definierten DQ-Dimensionen. Es muss sichergestellt werden, dass die

Daten eine ausreichende Qualität aufweisen, um daraus Geschäftsprozessmodelle generieren zu können.

Evaluation Methode: Die Methode definiert Einschränkungen an die Datenqualität der Objektinstanzen:

- *Einheitlichkeit*: Objektinstanzen eines Objekttyps müssen ähnliche Strukturen aufweisen.
- *Genauigkeit*: Die Objektzustände müssen durch Attribut-Wert-Paare beschrieben sein.
- *Vollständigkeit*: Relevante Prozessinformationen müssen in den Objektdaten enthalten sein.
- *Aktualität*: Für die Prozessinstanzobjekte müssen repräsentative Zeitpunkte ermittelbar sein.

Zusätzlich berücksichtigt die Methode unterschiedliche Qualitätsstufen der bereitgestellten Daten durch:

- Gewichtungsfaktoren für die Bewertung von Zusammenhängen,
- Schwellwerte für die Erkennung signifikanter Beziehungen,
- Mechanismen zum Umgang mit unvollständigen oder inkonsistenten Daten.

Die Methode ermöglicht so eine Verarbeitung von Daten, die die definierten Anforderungen an die Daten nicht vollständig erfüllen. Sind die Abweichungen von diesen Anforderungen zu stark (beispielsweise unstrukturierte Objekte, ohne Übereinstimmungen in den Attributwerten) ist eine Erfüllung nicht mehr gewährleistet. Basierend auf den Einschränkungen wird die Anforderung erfüllt (AI02: $\checkmark_M$).

Evaluation Software-Prototyp: Der Software-Prototyp implementiert verschiedene Mechanismen zur Sicherstellung und Verbesserung der Datenqualität. In Abschnitt 7.3.2 wird die Datenvorverarbeitung als Erweiterung des Software-Prototyps vorgestellt. Im Rahmen der Datenvorverarbeitung werden Attributnamen vereinheitlicht, Datumsformate standardisiert und potenzielle Duplikate erkannt. Dies gewährleistet eine konsistente Datenbasis für die weitere Verarbeitung. Nicht-verarbeitbare Dateien werden protokolliert sowie Dateien ohne erkennbare Zusammenhänge ausgeschlossen und dokumentiert. Zusätzlich werden die Dateien darauf geprüft, ob sie mehrere Objektinstanzen beschreiben. Zusätzlich werden im Software-Prototyp konfigurierbare Schwellwerte verwendet. Die schrittweise Verarbeitung unterstützt weitere Validierungsmöglichkeiten. Aufgrund der Methode ist auch bei teilweise eingeschränkter Datenqualität eine Prozessmodellgenerierung möglich, da:

- Zusammenhänge nur in 80% der Fälle auftreten müssen.
- Einzelne fehlerhafte Objektinstanzen ausgeschlossen werden können.
- Die Gewichtungsfaktoren anpassbar sind.

Für eine verbesserte Datenqualität könnten noch weitere Vorverarbeitungsschritte ergänzt werden. Daher wird die Anforderung nur als weitgehend erfüllt bewertet (AI02: $\sim_P$).

Datenformat (AI03): Diese Anforderung adressiert die Verarbeitung unterschiedlicher Dateiformate und Datenstrukturen. Dabei soll ein Kompromiss zwischen standardisierter Verarbeitung und flexibler Anwendbarkeit erreicht werden.

Evaluation Methode: Die Methode basiert auf der Verarbeitung von Objektinstanzen, die durch Attribut-Wert-Paare beschrieben sind. Dieser Ansatz ermöglicht eine einheitliche Weiterverarbeitung, ist aber format-unabhängig konzipiert. Die Methode definiert:

- Eigenschaften von Objektinstanzen, zu denen die Daten bereitgestellt werden.
- Anforderungen an die Struktur der Attribut-Wert-Paare, mit denen der Zustand einer Objektinstanz beschrieben wird.
- Ansätze zur Integration unterschiedlicher Datenquellen.
- Schritte zur Datenvorverarbeitung.

Die Methode sieht Erweiterungsmöglichkeiten für verschiedene Eingabeformate vor, ist aber primär für semi-strukturierte Daten konzipiert. Die Anforderung wird erfüllt (AI03: $\checkmark_M$).

Evaluation Software-Prototyp: Der Software-Prototyp implementiert die Verarbeitung von JSON- und XML-Dateien. Die Erweiterungen zur Verarbeitung weiterer Formate in Abschnitt 7.3.2 umfassen die Transformation unstrukturierter Daten durch OCR und die Strukturierung und Bereinigung von Daten. Dadurch soll die Extraktion von natürlichsprachlichen Text, die Strukturierung in Attribut-Wert-Paaren, die Vereinheitlichung von Benennungen ermöglicht werden. Obwohl der Prototyp erweiterbar konzipiert ist und erste zusätzliche Formate unterstützt, beschränkt sich die vollständige Implementierung auf JSON und XML. Weitere Formate wurden konzeptionell betrachtet, aber noch nicht vollständig integriert. Daher wird die Anforderung nur teilweise erfüllt (AI03: $\sim_P$).

Ergebnis der Evaluation: Diese Betrachtung der Anforderungen an die Input-Daten zeigt, dass die Methode die Anforderungen erfüllt. Der Software-Prototyp erfüllt diese weitgehend. Die Bewertung *weitgehend erfüllt* wurde gewählt, da grundlegende Mechanismen zur Qualitätssicherung und Datenverarbeitung implementiert sind, jedoch in spezifischen Bereichen noch Erweiterungen vorgenommen werden sollten. Die identifizierten Limitationen betreffen dabei die Datenqualität und die Datenformate:

- Im Bereich der Datenqualität (AI02) implementiert der Prototyp zwar grundlegende Qualitätssicherungsmechanismen wie die Vereinheitlichung von Attributnamen, Datumsformaten und die Erkennung von Duplikaten. Einschränkungen bestehen jedoch bei der automatischen Qualitätsverbesserung komplexerer Datenstrukturen und der Integration fortgeschrittener Bereinigungs- und Standardisierungsmechanismen.
- Bezüglich der unterstützten Datenformate (AI03) beschränkt sich die vollständige Implementierung auf JSON und XML. Während weitere Formate nur prototypisch unterstützt werden. Daher fehlt noch die Integration der Vorverarbeitungen, um die Verarbeitung von nicht unmittelbar maschinenlesbarem, unstrukturiertem Daten zu ermöglichen

Die Funktionalität für semi-strukturierte Daten ist implementiert und grundlegende Qualitätssicherungsmechanismen sind vorhanden. Die modulare Architektur ermöglicht zudem künftige Erweiterungen. Alternative Verarbeitungswege existieren für nicht direkt unterstützte Formate und die identifizierten Limitationen können durch manuelle Vorverarbeitung oder entsprechende Konfiguration ausgeglichen werden. Für eine bessere Anwendbarkeit sind dies kritische Faktoren. Um jedoch die Methode zur objektbasierte Prozessmodellierung zu evaluieren, stellt dies keine Einschränkung dar.

In Tabelle 8-1 wird die Bewertung des Erfüllungsgrads der Anforderungen zusammengefasst.

Tabelle 8-1: Erfüllungsgrad der Anforderungen an die Input-Daten

Nr.	Anforderung	Bewertung Methode	Bewertung Software-Prototyp
AI01	Informationsgehalt	✓	✓
AI02	Datenqualität	✓	~
AI03	Datenformat	✓	~

8.2 Evaluierung der Anforderungen an das generierte Modell

In Abschnitt 2.3 werden Qualitätsaspekte für ein Geschäftsprozessmodell beschrieben. Anhand dieser wurden in Abschnitt 5.2 Anforderungen an das generierte Modell gestellt, anhand derer die Methode und der Software-Prototyp evaluiert werden. Die Erfüllung dieser Anforderungen wird durch die Evaluierung der Generierung der Kontrollflussmuster in Abschnitt 8.4 nochmals verdeutlicht.

Bei der Betrachtung der Qualität von Geschäftsprozessmodellen lassen sich zwei Betrachtungsweisen unterscheiden:

- *Betrachtung der Qualität des Geschäftsprozesses*: Das Geschäftsprozessmodell wird in Vertretung des abgebildeten Modelloriginals (dem Geschäftsprozess) auf definierte Qualitätsaspekte analysiert.
- *Betrachtung der Modellqualität*: Das Geschäftsprozessmodell wird als eigenständiges Objekt auf definierte Qualitätsaspekte analysiert.

In dieser Evaluation soll insbesondere die Modellqualität betrachtet werden, da ein bestehender Geschäftsprozess basierend auf dessen Objektinstanzen modelliert werden soll. Es wurden die drei Anforderungskategorien *Modell* (AM01), *Modellierungssprache* (AM02) und *Modellierung* (AM03) definiert.

Modell (AM01): Diese Anforderung umfasst verschiedene Qualitätsaspekte des generierten Geschäftsprozessmodells, die für die Zweckerfüllung wesentlich sind.

Evaluation Methode: Die Methode definiert systematische Konzepte zur Sicherstellung der Modellqualität:

- *Konsistenz und Redundanzfreiheit*: Die Methode basiert auf Daten zu Objektinstanzen, die bei der Ausführung eines Geschäftsprozesses erzeugt, bearbeitet, gelesen oder verbraucht wurden. Daher folgen diese den kausalen Zusammenhängen und den organisatorischen Vorgaben. Zusätzlich werden die Objekte und deren Zusammenhänge mehrstufig analysiert und bewertet. Basierend auf dem paarweisen Vergleich aller vorliegenden Objektinstanzen wird anschließend der Zusammenhänge zwischen Objektinstanzen innerhalb der Prozessinstanzen die Konsistenz überprüft. Zusätzlich werden anhand von definierbaren Schwellwerten nur Informationen extrahiert, die in der definierten Häufigkeit auftreten. Auffälligkeiten, wie beispielsweise Ausreißer in den ermittelten repräsentativen Zeitpunkten, werden erkannt und protokolliert. Diese werden anschließend nicht in der Prozessmodellgenerierung beachtet. Die Aktivitätstypen werden explizit auf Widersprüche in den Objektinstanzen und in Bezug zu anderen Aktivitätstypen geprüft. Zusätzlich werden die übereinstimmenden Attribute in den unterschiedlichen Aktivitätstypen betrachtet und auch Redundanzen entfernt bzw. durch Zusammenführungen aufgelöst.
- *Vollständigkeit*: Die mehrstufige Analyse der Objektinstanzen soll sicherstellen, dass alle relevanten Aspekte erfasst werden. Für die Extraktion der Aktivitätstypen werden bei dem zeitlich-basierten Ansatz nur die Prozessinstanzobjekte verwendet, da diese ein für den Geschäftsprozess repräsentativen Zeitpunkt für das Auftreten der Objektinstanzen innerhalb einer Ausführung haben. Bei dem in-

haltsbasierten Ansatz werden auch die prozessinstanzunabhängigen Objekte berücksichtigt. Zusätzlich wird bei der Aktivitätsgenerierung geprüft, ob alle gefundenen Überschneidungen in den Aktivitätstypen abgedeckt werden. Es wird auch geprüft, dass jeder Objekttyp und jede Objektinstanz in den Aktivitätstypen berücksichtigt wird. Dabei wird dies differenziert nach Input und Output betrachtet. Durch die Definition der übernommenen Attribute in den Aktivitätstypen sowie der Definition der Attribute, durch die ein Objekttyp beschrieben wird, ist zusätzlich ersichtlich, welche nicht beachteten Attribute die Objekttypen haben. Dadurch kann aufgedeckt werden, welche weiteren Attribute benötigt werden, um bei der Erzeugung einer Objektinstanz alle erforderlichen Informationen vorliegen zu haben. Die Methode definiert Ansätze, um einfache und mehrfache direkte Objektbeziehungstypen sowie einfache und mehrfache indirekte Objektbeziehungstypen zu identifizieren.

- *Korrektheit*: Die semantische Korrektheit wird unterstützt durch die Rückverfolgbarkeit zu den bereitgestellten Daten und der mehrstufigen Analyse, der sowohl die zeitlichen Zusammenhänge als auch inhaltlichen Zusammenhänge berücksichtigt.
- *Eindeutigkeit und Abstraktionsebene*: Die Methode unterstützt die eindeutige Interpretation durch das schrittweise Vorgehen, ein klares Benennungskonzepte, Möglichkeiten zur Zusammenfassung und Entfernung von Objekttypen und Aktivitätstypen sowie der Verwendung von Petri-Netzen als Modellierungssprache.

Die Methode erfüllt damit die Qualitätsanforderungen an das Modell (AM01: $\checkmark_M$).

Evaluation Software-Prototyp: Da der Software-Prototyp die Methode implementiert, sind ebenfalls einige der Anforderungen erfüllt. In der Generierung werden alle vom Anwender bereitgestellten Objektinstanzen eines Geschäftsprozesses berücksichtigt, solange diese einer Prozessinstanz zugeordnet werden können. Andernfalls wird die entsprechende Datei als *nicht-berücksichtigt* protokolliert (AM01a - Konsistenz). Durch den interaktiven Modus kann eine Verbesserung vorgenommen werden.

Der Software-Prototyp implementiert die Aufdeckung und Analyse der zeitlichen und inhaltlichen Zusammenhänge (AM01b - Vollständigkeit). Jedoch sind zur Ableitung von mehrfachen Objektbeziehungstypen noch weitere Vergleichsansätze, beispielsweise basierend auf Korrelationsprüfungen oder maschinellen Lernverfahren, zu implementieren. Redundante Elemente werden durch systematische Prüfungen erkannt und entfernt (AM01c - Redundanzfreiheit).

Durch die schrittweisen Darstellungen der Ergebnisse sowie der Unterstützung zur Benennung von Objekt- und Aktivitätstypen durch generative Transformermodelle wird insbesondere die Eindeutigkeit verbessert (AM01e - Eindeutigkeit). Die separate Darstel-

lung von den Objekt- und Aktivitätstypen ermöglicht eine reduzierte Darstellung im Modell (AM01f - Abstraktionsebene).

Das Modell bildet den Geschäftsprozess entsprechend den Input-Dateien ab und erweitert dieses nicht willkürlich. Daher sind die abgebildeten Aspekte im Modell rückverfolgbar zu den Daten (AM01d - Korrektheit). Bei der Anwendung von Process Discovery Algorithmen genügen die generierten Modelle den gewährleisteten Eigenschaften der Algorithmen (AM01g - Eigenschaften Petri-Netz). Bei der inhaltsbasierten Generierung genügen die Modelle den Eigenschaften der Objektinstanzen. Das bedeutet, dass beispielsweise eine Objektinstanz nicht mehrfach als Input verwendet werden kann, ohne dass diese ebenfalls Output einer Aktivität ist. Hierbei werden ebenfalls zeitliche Zusammenhänge eingehalten. So kann eine Objektinstanz nicht als Input eines Aktivitätstyp definiert sein, bevor diese erzeugt wurde. Davon ausgenommen sind prozessinstanzunabhängige Objekte und Objektinstanzen, durch die ein Geschäftsprozess startet.

Der Prototyp implementiert die Methode, jedoch wird beispielsweise die Erkennung von mehrfachen Objektbeziehungstypen noch nicht unterstützt. Somit erfüllt dieser weitgehend die Qualitätsanforderungen (AM01: $\sim_P$).

Modellierungssprache (AM02): Diese Anforderung definiert die notwendigen Eigenschaften der verwendeten Modellierungssprache für die Geschäftsprozessmodellierung.

Evaluation Methode: Die Methode basiert auf höheren Petri-Netzen als Modellierungssprache. Höhere Petri-Netze bieten die Möglichkeit, sowohl die Ordnungsbeziehungen der Aktivitäten als auch Objektinformationen zu repräsentieren. Die Ordnungsbeziehungen werden dabei durch die Petri-Netz-Elemente (Stellen, Transitionen, Kanten) modelliert, während die Objektinformationen durch JSON-Schema-basierte Stelleninschriften und Ausdruckslogik-basierte Transitionsinschriften repräsentiert werden. Dies ermöglicht die Spezifikation der Objektstrukturen durch JSON-Schemata und dazu valide JSON-Objekte. Zusätzlich können Objektmanipulationen und Ausführungsbedingungen durch Jsonnet-Ausdrücke formuliert werden. Die Angemessenheit der Modellierung wird durch verschiedene Verbesserungsmechanismen unterstützt, indem beispielsweise Aktivitätstypen zusammengeführt und redundante Aktivitätstypen und Attribute entfernt werden. Die Verständlichkeit der generierten Modelle wird durch eine einheitliche Benennung und durch die Darstellung als Petri-Netze unterstützt. Die Methode erfüllt damit die Anforderungen an die Modellierungssprache vollständig (AM02: $\checkmark_M$).

Evaluation Software-Prototyp: Die grafische Darstellung der generierten Modelle wird durch die Integration der *PM4Py*-Bibliothek und der *Graphviz*-Bibliothek realisiert, die verschiedene Visualisierungsvarianten mit Zoomfunktionen unterstützt. Zusätzlich können die generierten Modelle in verschiedene Bildformate exportiert werden. Da die

Objekttypen, Aktivitätstypen und Kanten zusätzlich im Modell des Software-Prototyps erstellt werden, können diese Elemente durch eine Erweiterung in weitere Austauschformate transformiert werden. Die maschinenlesbare Beschreibung der Modelle wird durch die Implementierung des PNML-Standards gewährleistet. Die Ergebnisse der einzelnen Schritte werden bereitgestellt, wodurch die Nachvollziehbarkeit und Verständlichkeit verbessert werden sollen. Der Software-Prototyp repräsentiert die Objektstrukturen und die Regeln der Aktivitätstypen separat vom Petri-Netz. In JSON- oder XML-Netzen können die Objektstrukturen und die Regeln der Aktivitätstypen integriert dargestellt werden. Eine vollständige Integration aller Aspekte in XML- oder JSON-Netzen ist konzeptionell möglich, aber noch nicht implementiert. Der Software-Prototyp erfüllt daher die Anforderungen weitgehend (AM02: $\sim_P$).

Modellierung (AM03): Diese Anforderung umfasst konkrete Richtlinien zur Verbesserung der Verständlichkeit und Übersichtlichkeit des generierten Modells.

Evaluation Methode: Die Methode definiert systematische Konzepte zur Erfüllung der Modellierungsrichtlinien. Für die Reduktion der Elementanzahl (AM03a) gibt es verschiedene Ansätze. Durch eine Angabe der maximalen Cluster-Anzahl wird die Anzahl der Stellen beeinflusst. Außerdem werden die Aktivitätstypen auf Zusammenführungen und Entfernungen geprüft, ohne die Vollständigkeit der Übernahmen und Manipulationen zu beeinträchtigen. Die Pfadreduktion (AM03b) erfolgt durch die Gewichtung und Auswahl von Aktivitätstypen. Eine Aktivität wird nur ausgewählt, wenn diese einen neuen Objekttyp oder neue Attribute betrifft. Dadurch werden redundante Pfade eliminiert. Zusätzlich werden Redundanzen im Modell geprüft und entfernt. Die Anforderung nach einem eindeutigen Start- und Endereignis (AM03c) ist entsprechend der verwendeten Process Discovery Algorithmen für den zeitlich-basierten Ansatz gewährleistet. Bei der inhaltsbasierten Generierung und dem Zusammensetzen der Aktivitätstypen bezüglich der beteiligten Objekttypen kann die Anforderung nach einem eindeutigen Start- und Endereignis durch Ergänzen von weiteren Elementen erreicht werden. Zur Strukturierung definiert die Methode Regeln für Split-/Join-Strukturen sowie umfassende Benennungskonventionen. Diese Richtlinien gewährleisten eine einheitliche und nachvollziehbare Modellstrukturierung.

Evaluation Software-Prototyp: Die Elementreduktion wird durch eine konfigurierbare maximale Cluster-Anzahl sowie Algorithmen zur Aktivitätszusammenführung realisiert. Bei den Aktivitäten wird auf redundante Elemente geprüft. Die Anforderung nach einem eindeutigen Start- und Endereignis (AM03c) wurde bewusst nicht implementiert, da prozessinstanzunabhängige Objekte wie Kunden- oder Produktdaten ohne vorherige Prozessbeteiligung existieren können. Zusätzlich kann ein Geschäftsprozess auch mit verschiedenen erzeugten Objekten enden. Bei einem Bestellungsprozess könnte eine Stornierung oder eine Sendung das Ende darstellen. Dies ermöglicht eine realitätsnahe

Modellierung, bei der ein Geschäftsprozess vom Vorhandensein verschiedener Objekttypen abhängen kann – beispielsweise startet der Geschäftsprozess erst bei Vorliegen einer Bestellung und der entsprechenden Produktdaten, oder endet mit der parallelen Erzeugung mehrerer Objekttypen wie Rechnung und Lieferschein. Stellen ohne eingehende Kanten repräsentieren dabei Schnittstellen über die Prozessgrenze hinweg.

Es werden so lange Aktivitätstypen hinzugefügt, bis die Prozessinstanzobjekte sowohl als Input als auch als Output in einem Aktivitätstyp beteiligt sind (mit Ausnahme der zeitlich ersten und letzten Objekte). Dies gewährleistet ein zusammenhängendes Petri-Netz. Der minimale Score (minScore) kann dies beeinflussen, da Aktivitätstypen unter diesem Score nicht mehr hinzugefügt werden.

Die Strukturiertheit (AM03d) der generierten Modelle wird zum einen durch die integrierten Bibliotheken realisiert, die verschiedene Visualisierungsvarianten bereitstellt. Die Einhaltung der Beschriftungs- und Inschriftenkonventionen (AM03e) wird durch übergebene Benennungsregeln als Prompt für ein generatives Transformermodell unterstützt. Dies umfasst Vorgaben für die Bezeichnungen von:

- Transitionen (einheitliche Verbformen im Infinitiv, abgeleitet aus den Input- und Output-Objekttypen oder Verwendung der beteiligten Objekttypen als Bezeichnung in der Form (*Bezeichnungen der Input-Objekttypen*, *Bezeichnungen der Output-Objekttypen*),
- Stellen (konsistente Objektbezeichnungen, abgeleitet aus den Schemata).

Der Prototyp erfüllt die Modellierungsanforderungen ebenfalls weitgehend. Künftig könnten durch nachgelagerte manuelle oder automatisierte Anpassungen dies weiter verbessert werden. Zusätzlich wird geprüft, ob Aufspaltungen und Zusammenführungen in gewünschter Weise auftreten (AM03: $\sim_P$).

Ergebnis der Evaluation: Diese Betrachtung der Qualitätsaspekte für das generierte Modell zeigt, dass die Methode die Anforderungen erfüllt. Die Anforderungen an die Modellierung werden vom Software-Prototyp weitgehend erfüllt: Die Bewertung *weitgehend erfüllt* wurde gewählt, da zwar alle grundlegenden Anforderungen umgesetzt sind, aber in den genannten Bereichen noch Verbesserungen vorgenommen werden sollten. Diese Einschränkungen beeinträchtigen jedoch nicht die Hauptfunktionalität der objektbasierten Prozessmodellgenerierung. Die identifizierten Limitationen betreffen dabei verschiedene Aspekte der Modellqualität, Modellierungssprache und konkreten Modellierung:

- Im Bereich der Modellqualität (AM01) zeigt die Evaluation des Software-Prototyps Einschränkungen durch fehlende Unterstützung für mehrfache Objekt-

beziehungstypen. Zudem ist die Anpassung der Abstraktionsebenen nur manuell möglich.

- Bezüglich der Modellierungssprache (AM02) besteht die zentrale Limitation in der noch ausstehenden Integration von Objektstrukturen, Aktivitätstypen und deren Regeln in XML- oder JSON-Netzen. Während die separate Darstellung dieser Aspekte funktional ist, würde eine integrierte Repräsentation die Verwendung des Modells zur weiteren Analyse verbessern. Zusätzlich ist die Erweiterbarkeit um weitere Austauschformate zwar vorgesehen, aber noch nicht implementiert.
- Bei der konkreten Modellierung (AM03) wurde bewusst auf standardisierte Start- und Endereignisse verzichtet, um die gefundenen Zusammenhänge entsprechende den bereitgestellten Daten im Modell abzubilden. Eine automatische Strukturierung komplexer Modelle könnte die Modelle verbessern.

Diese Einschränkungen sind für den praktischen Einsatz jedoch nicht kritisch. Die Kernfunktionalität zur Prozessmodellgenerierung ist implementiert. Die modulare Architektur ermöglicht zudem künftige Erweiterungen. Einige Einschränkungen, wie der Verzicht auf standardisierte Start-/Endereignisse, stellen dabei bewusste Designentscheidungen zur Erhaltung der Modellierungsflexibilität dar. Identifizierte Limitationen können durch manuelle Nachbearbeitung oder entsprechende Konfiguration ausgeglichen werden.

Der Erfüllungsgrad der Anforderungen an das generierte Modell wird in Tabelle 8-2 zusammengefasst.

Tabelle 8-2: Erfüllungsgrad der Anforderungen an das generierte Modell

Nr.	Anforderung	Bewertung Methode	Bewertung Software-Prototyp
AM01	Modell	✓	~
AM02	Modellierungssprache	✓	~
AM03	Modellierung	✓	~

8.3 Evaluierung der Anforderungen an die Generierung

Die funktionalen Anforderungen an den Generierungsprozess sind in Abschnitt 5.3 definiert. Da diese Anforderungen primär die Implementierung betreffen, wird folgend nur der Software-Prototyp betrachtet. Die methodischen Aspekte sind bereits durch die vorherigen Evaluationen (Input-Daten, Modell) abgedeckt.

Plattformunabhängigkeit (AG01) – *Evaluation Software-Prototyp*: Die Plattformunabhängigkeit wird durch die Implementierung einer PyQt5[1]-basierten Anwendung gewährleistet. Die Benutzungsoberfläche ermöglicht einen Zugriff auf alle Funktionalitäten und ist auf gängigen Betriebssystemen lauffähig.

Dateiverarbeitung (AG02) – *Evaluation Software-Prototyp*: Die Anforderungen werden durch die implementierte Upload-Funktionalität erfüllt. Die Anwendung unterstützt das gleichzeitige Hochladen mehrerer JSON- oder XML-Dateien und validiert diese auf Konformität. Nicht-konforme Dateien werden mit entsprechender Dokumentation im Protokoll ausgeschlossen. Liegen keine Dateien zur Verarbeitung vor, wird eine entsprechende Meldung gegeben.

Dateivorverarbeitung (AG03) und **Informationsextraktion** (AG04) – *Evaluation Software-Prototyp*: Die Erfüllung der Anforderungen wurde in Kapitel 7 gezeigt. Zusätzlich wurde darauf bereits in den vorangegangenen Evaluationen eingegangen.

Generierungsalgorithmus (AG05) – *Evaluation Software-Prototyp*: Die Beschriftungen und Inschriften (AG05a) erfolgen dabei durch ein generatives Transformermodell, das die definierten Konventionen sicherstellen soll. Die maschinenlesbare Repräsentation (AG05b) wird PNML-konform erzeugt, wodurch die Interoperabilität mit anderen Werkzeugen gewährleistet ist. Bei widersprüchlichen Informationen werden diese protokolliert und ausgeschlossen, um eine konsistente Modellierung sicherzustellen (AG05c). Die Abgrenzung der Geschäftsprozesse (AG05d) erfolgt durch eine kombinierte Analyse von zeitlichen Zusammenhängen und Überschneidungen in den Objektinstanzen, indem diese Prozessinstanzen zugeordnet werden. Zusätzlich werden die Varianten der Prozessinstanzen bezüglich der auftretenden Objekttypen angezeigt, um eine Nachvollziehbarkeit zu ermöglichen. Die implementierte Lösung stellt dem Anwender verschiedene Ausgabeformate zur Verfügung. Die grafische Visualisierung ermöglicht eine direkte Inspektion der generierten Modelle, während der PNML-Export die Weiterverarbeitung in anderen Systemen ermöglicht. Ein detaillierter Terminal-Modus dokumentiert den gesamten Generierungsprozess und macht Entscheidungen und eventuelle Probleme transparent. Dazu werden die Objekttypen und Aktivitätstypen definiert sowie diese in einem Petri-Netz als Stellen und Transitionen abgebildet. Die Ordnungsbeziehungen der Aktivitätstypen werden durch die Kanten zwischen den Stellen und Transitionen abgebildet (AG05e-g). Die Evaluation der Basis-Kontrollflussmuster (AG05h) wird in Abschnitt 8.4 durchgeführt. Diese zeigt, dass alle Basis-Kontrollflussmuster basierend auf den entsprechenden Testdaten erfolgreich umgesetzt werden können.

[1] https://pypi.org/project/PyQt5/

Generierungsergebnis (AG06) und **Ergebnisanzeige** (AG07) – *Evaluation Software-Prototyp*: Die Implementierung ermöglicht eine Visualisierung der Modelle, einen Export der generierten Modelle sowie eine Protokollierung und Fehlerbehandlung (AG06 und AG07).

Protokollierung (AG08) und **Fehlerbehandlung** (AG09) – *Evaluation Software-Prototyp:* Das System dokumentiert systematisch alle Verarbeitungsschritte und Probleme. Dies umfasst sowohl ausgeschlossene Dateien aufgrund von Formatproblemen als auch Teilverarbeitungen bei unvollständigen oder inkonsistenten Daten. Nicht abbildbare Informationen werden ebenso protokolliert wie eventuelle Abbrüche. Diese Dokumentation ermöglicht es dem Anwender, den Generierungsprozess nachzuvollziehen und bei Bedarf die bereitgestellten Dateien zu verändern.

Ergebnis der Evaluation: Die Evaluation der Anforderungen an den Generierungsprozess zeigt, dass der Software-Prototyp diese weitgehend erfüllt. Die Bewertung *weitgehend erfüllt* wurde gewählt, da in spezifischen Bereichen noch Verbesserungspotenzial besteht. Die identifizierten Limitationen betreffen dabei drei Hauptaspekte:

- Die Dateivorverarbeitung (AG03) unterstützt zwar die grundlegenden Vorverarbeitungsschritte, die Integration weiterer Dateiformate und fortgeschrittener Vorverarbeitungsmechanismen steht jedoch noch aus.
- Die Ergebnisanzeige (AG07) bietet bereits flexible Visualisierungs- und Exportoptionen, könnte aber durch erweiterte Darstellungsmöglichkeiten und zusätzliche Exportformate verbessert werden.
- Bei Protokollierung (AG08) und Fehlerbehandlung (AG09) fehlen noch detailliertere Diagnosemöglichkeiten und eine umfassendere Dokumentation von Verarbeitungsproblemen.

Diese Einschränkungen sind für den praktischen Einsatz jedoch nicht kritisch. Die Kernfunktionalität der Prozessmodellgenerierung ist implementiert und die modulare Architektur ermöglicht künftige Erweiterungen. Die identifizierten Limitationen betreffen primär Optimierungspotenziale in der Benutzerinteraktion und Fehlerdiagnose, welche die grundlegende Funktionalität nicht beeinträchtigen. Diese Ergebnisse sind in Tabelle 8-3 zusammengefasst.

Tabelle 8-3: Erfüllungsgrad der Anforderungen an das generierte Modell

Nr.	Anforderung	Bewertung
AG01	Seitenaufruf	✓
AG02	Dateien hochladen und prüfen	✓

Nr.	Anforderung	Bewertung
AG03	Dateivorverarbeitung	~
AG04	Informationsextraktion	✓
AG05	Generierungsalgorithmus	✓
AG06	Generierungsergebnis	✓
AG07	Ergebnisanzeige	~
AG08	Protokoll	~
AG09	Fehlerbehandlung	~

8.4 Evaluation der Kontrollflussmuster-Generierung

Die in dieser Arbeit entwickelte Methode generiert Geschäftsprozessmodelle basierend auf Objektinstanzen, die in den Aktivitätsinstanzen eines Geschäftsprozesses erzeugt, bearbeitet, gelesen oder verbraucht worden sind. Aus diesen Objektinstanzen werden Rückschlüsse auf die Aktivitätsinstanzen gezogen. Daher wurden die fünf Basis-Kontrollflussmuster anhand der Zustände der Aktivitätsinstanzen beschrieben (siehe Abschnitt 2.2.1). Folgend wird die Evaluation der Methode anhand des Software-Prototyps bezüglich dieser Kontrollflussmuster beschrieben. Zusätzlich wurden auch die Kontrollflussmuster Schleife und Iteration betrachtet.

Für diese Evaluation wurde das Erkennen der Ordnungsbeziehungen zwischen den Aktivitäten aus Definition 6-1 sowie die korrekte Abbildung dieser Ordnungsbeziehungen in einem Geschäftsprozessmodell (siehe Definition 6-7) evaluiert. Die Evaluation erfolgte anhand von künstlich erzeugten Testdaten, die in Tabelle 8-4 zusammengefasst sind.

Tabelle 8-4: Testdaten zur Evaluation

Kontrollflussmuster	Dateien	Prozessinstanzen	Objekttypen
Sequenz	16	5	4
Nebenläufigkeit	30	4	9
Alternative	50	9	11
Schleife	16 / 18	4	6
Iteration	33	4	8

Für jeden dieser Fälle wurden zunächst die Ergebnisse für die Zuordnung zu den Objekttypen und zu den Prozessinstanzen evaluiert. Anschließend wurde das Erkennen der Ordnungsbeziehungen basierend auf den zeitlichen und inhaltlichen Zusammenhängen zwischen Objektinstanzen sowie die Generierung mittels Process Discovery-Algorithmen und mittels der Zusammensetzung der Aktivitätstypen evaluiert.

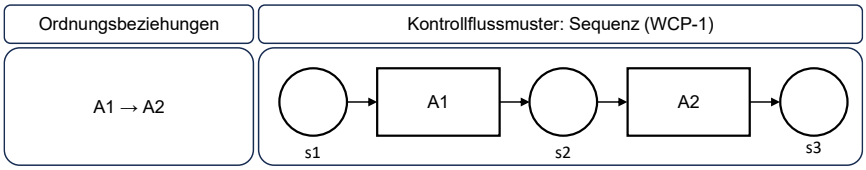
Für die Generierung wurden die Gewichtungsfaktoren mit den Werten in Tabelle 8-5 belegt. Dies sind die voreingestellten Werte aus Abschnitt 7.2.1.5, die jedoch anpassbar sind.

Tabelle 8-5: Werte der Gewichtungsfaktoren

Faktor	Beschreibung	Wert
early_position_bonus	Bonus für Aktivitätstypen, die früh im Geschäftsprozess auftreten.	1.5
late_position_penalty	Gewichtung für Aktivitäten, die spät im Geschäftsprozess auftreten.	0.5
reference_weight	Gewichtung für Aktivitätstypen die prozessinstanzunabhängige Objekttypen berücksichtigen.	2.0
reference_position_factor	Gewichtung von Aktivitätstypen mit prozessinstanzunabhängigen Objekttypen, die früh im Geschäftsprozess auftreten.	2.0
sequence_weight	Gewichtung für die Häufigkeit des Aktivitätstyps.	1.5
attribute_weight	Gewichtung für die übereinstimmenden Attribute zwischen den beteiligten Objekttypen des Aktivitätstyp.	1.2
temporal_weight	Gewichtung für die zeitliche Bewertung der beteiligten Objektinstanzen des Aktivitätstyps.	4.0
variant_weight	Gewichtung für Aktivitätstypen, die die gleichen Objekttypen mit verschiedenen Versionen von Objektinstanzen betreffen.	0.5,
loop_weight	Gewichtung für Schleifenstrukturen	0.5
parallel_object_penalty	Strafwert für Aktivitätstypen, die nur Objekttypen mit nebenläufiger Ordnungsbeziehung betreffen.	3.0
Min_AT_score	Minimaler Score für die Aufnahme eines Aktivitätstyps.	0
each_attribute_by_object	Boolean-Wert, der angibt, ob Attribute für jeden Objekttyp betrachtet werden oder für den gesamten Geschäftsprozess.	False

8.4.1 Sequenzielle Ordnungsbeziehungen von Aktivitäten (Sequenz)

Die sequenzielle Ordnungsbeziehung von Aktivitäten wird mit dem Kontrollflussmuster Sequenz repräsentiert. Das in Abbildung 8-1 dargestellte Kontrollflussmuster Sequenz repräsentiert, dass Aktivität A2 nach dem Abschluss der Aktivität A1 ausgeführt werden kann. Dies entspricht einer sequenziellen Ordnungsbeziehung zwischen den Aktivitäten A1 und A2 (siehe Abschnitt 2.2.1.1).



in Anlehnung an [AHKB03]

Abbildung 8-1: Ordnungsbeziehung und Kontrollflussmuster zu Sequenz

Um zu evaluieren, ob der Software-Prototyp diese Ordnungsbeziehungen anhand der Objektinstanzen korrekt erkennt, wird ein Geschäftsprozess angenommen, bei dem zunächst eine Bestellung eingeht. Basierend auf dieser Bestellung wird zuerst der Lieferschein erstellt und anschließend die Rechnung versendet.

Nach dem beschriebenen Geschäftsprozess haben die Aktivitäten *Lieferschein erstellen* und *Rechnung versenden* eine sequenzielle Ordnungsbeziehung. Außerdem erzeugen, bearbeiten, lesen oder verbrauchen die Aktivitätsinstanzen die Objektinstanzen der Objekttypen *Bestellung*, *Lieferschein* und *Rechnung*.

In den *Sequenz-Testdaten*[2] werden die Objektinstanzen zu fünf Ausführungen des Geschäftsprozesses bereitgestellt. Zu den Produkten, die in den Bestellungen angegeben wurden, wird zusätzlich eine Datei bereitgestellt. Abbildung 8-2 zeigt die Rohdaten der Dateien *Bestellung1.json*, *Lieferschein1.json* und *Rechnung1.json* zu den Objektinstanzen, die in einer Prozessinstanz des Geschäftsprozesses erzeugt, bearbeitet, gelesen oder verbraucht wurden. Für die einzelnen Objektinstanzen (und somit für den ermittelten repräsentativen Zeitpunkt der Objektinstanz) gilt, dass sie je Prozessinstanz die Reihenfolge einhalten, die durch die sequenzielle Ordnungsbeziehung der Aktivitäten vorgegeben ist. Da die Produktdaten prozessinstanzunabhängig sind, haben diese keinen repräsentativen Zeitpunkt für eine einzelne Prozessinstanz.

[2] Siehe https://gitlab.kit.edu/kit/aifb/BIS/kit-bis/objektbasiertemodellgenerierung/testszenarios/Sequenz-Testdaten für das Beispielszenario.

Abbildung 8-2: Objektinstanzen einer Prozessinstanz als JSON-Objekte

Nachfolgend werden in Abschnitt 8.4.1.1 zunächst die Ergebnisse zu den Zuordnungen zu Objekttypen und Prozessinstanzen vorgestellt. In den Abschnitten 8.4.1.2 und 8.4.1.3 wird die Prozessmodellgenerierung basierend auf den zeitlichen Zusammenhängen und basierend auf den inhaltlichen Zusammenhängen beschrieben sowie das Ergebnis evaluiert.

8.4.1.1 Informationsextraktion zu Objekttypen und Prozessinstanzen

Mittels des Clustering-Algorithmus wurden vier Objekttypen sowie fünf Prozessinstanzen erkannt. Da in jeder Prozessinstanz genau eine Objektinstanz der Objekttypen *Bestellung*, *Lieferschein* und *Rechnung* erzeugt, bearbeitet, gelesen oder verbraucht wurden, wurde nur eine Prozessvariante erkannt. Alle Dateien konnten Prozessinstanzen zugeordnet werden (siehe Abbildung 8-3).

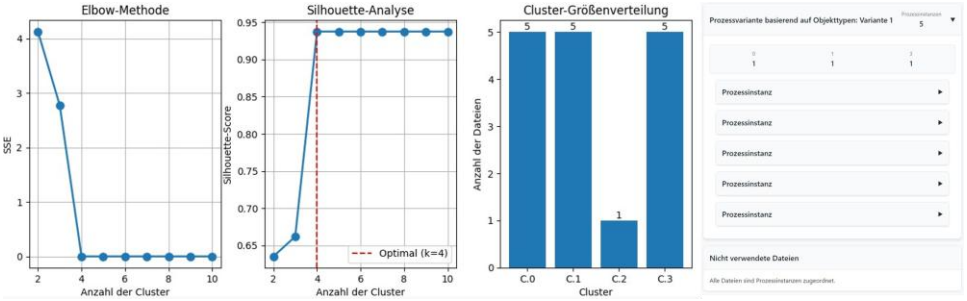


Abbildung 8-3: Cluster-Ergebnis und erkannte Prozessinstanzen

Ergebnis der Evaluation: Die Ergebnisse der Evaluation zeigen, dass die Objekttyp-Anzahl korrekt erkannt wurde und die Dateien entsprechend zugeordnet wurden. Ebenso wurden die Prozessinstanzen korrekt erkannt und die Objektinstanzen korrekt zugeordnet. Die Objekttypen wurden korrekt als prozessinstanzabhängig oder prozessinstanzunabhängig eingeordnet.

8.4.1.2 Prozessmodellgenerierung basierend auf den zeitlichen Zusammenhängen

Für die Generierung basierend auf zeitlichen Zusammenhängen werden die Objektinstanzen vom Software-Prototyp je Prozessinstanz nach den repräsentativen Zeitpunkten sortiert. Für zwei unmittelbar aufeinanderfolgende Objektinstanzen wird jeweils ein Event in einem Eventlog ergänzt sowie entsprechend ein Aktivitätstyp definiert. Somit ergaben sich für eine Prozessinstanz folgende Zusammenhänge:

1. *Bestellung → Lieferschein*:
   - Bestellung.json: ermittelter repräsentativer Zeitpunkt für die Objektinstanz *12.11.2023*
   - Lieferschein1.json: ermittelter repräsentativer Zeitpunkt für die Objektinstanz *29.11.2023*
2. *Lieferschein → Rechnung*:
   - Lieferschein1.json: ermittelter repräsentativer Zeitpunkt für die Objektinstanz *29.11.2023*
   - Rechnung1.json: ermittelter repräsentativer Zeitpunkt für die Objektinstanz *30.11.2023*

Da diese Zusammenhänge durch die sequenzielle Ordnungsbeziehung zwischen den Aktivitäten *Lieferschein erstellen* und *Rechnung versenden* für alle Objektinstanzen ermittelt wurden, werden wie in Abbildung 8-4 dargestellt die zwei Aktivitäten *Bestellung → Lieferschein* und *Lieferschein → Rechnung* aus den repräsentativen Zeitpunkten der Objektinstanzen erkannt.

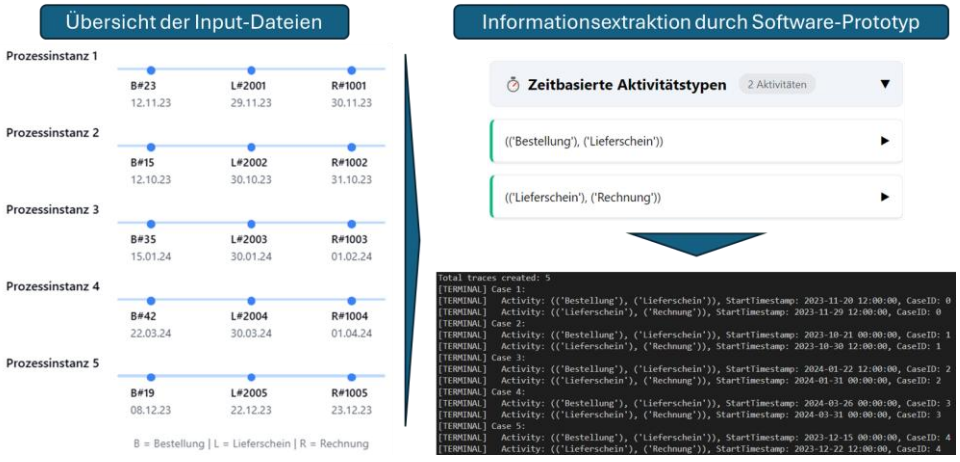


Abbildung 8-4: Zeitpunkte der Objektinstanzen aus den Input-Dateien je Prozessinstanz sowie daraus ermittelte Aktivitätstypen und generierter Eventlog

Das Ergebnis ist für jeden im Software-Prototyp zur Verfügung gestellten Process Discovery-Algorithmus (*Alpha Miner*, *Alpha Miner Plus*, *Heuristic Miner* und *Inductive Miner)* jeweils das Petri-Netz in Abbildung 8-5. Da für prozessinstanzunabhängige Objekte kein repräsentativer Zeitpunkt zu einer Prozessinstanz angenommen werden kann, werden diese bei der Generierung basierend auf den zeitlichen Zusammenhängen ausgeschlossen.

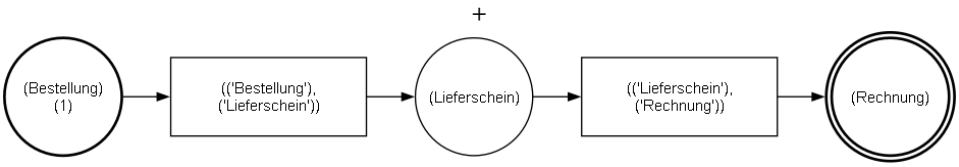


Abbildung 8-5: Aus den zeitlich-basierten Aktivitätstypen abgeleitetes Petri-Netz (Alpha Miner)

Ergebnis der Evaluation: Die zeitlichen Zusammenhänge zwischen den Objekttypen wurden entsprechend den Objektinstanzen korrekt erkannt und als Aktivitätstypen definiert. Mittels verschiedener Process Discovery-Algorithmen (Alpha Miner, Alpha Miner Plus, Heuristic Miner, Inductive Miner) wurden die Objekttypen (außer der prozessinstanzunabhängigen Objekte) und Aktivitätstypen in ein Petri-Netz transformiert, das die sequenziellen Ordnungsbeziehungen der Aktivitätstypen entsprechend den bereitgestellten Dateien repräsentiert.

8.4.1.3 Prozessmodellgenerierung basierend auf den inhaltlichen Zusammenhängen

Bei diesem Generierungs-Ansatz analysiert der Software-Prototyp die inhaltlichen Überschneidungen zwischen den Objektinstanzen einer Prozessinstanz sowie zu den prozessinstanzunabhägigen Objekten. Dabei werden zunächst auch zufällige Überschneidungen gefunden. Beispielsweise entspricht die Menge des Produktes mit der ID *101* im Lieferschein *2001* ebenfalls der Lieferart in der zugehörigen Bestellung (=*3*; siehe Abbildung 8-2). Um diese entdeckten zufälligen Überschneidungen auszuschließen, werden basierend auf den in Abschnitt 6.1.4 definierten und in Abschnitt 7.2.1.4 implementierten Aspekten die Überschneidungen bewertet und anschließend Objektbeziehungstypen definiert.

In Abbildung 8-6 sind auf der linken Seite die inhaltlichen Überschneidungen der Objektinstanzen dargestellt, die erkannt werden sollten. Auf der rechten Seite sind die vom Software-Prototyp erkannten Überschneidungen dargestellt. Da diese mit denen auf der linken Seite übereinstimmen, wurden zufällig erkannte Überschneidungen korrekt entfernt.

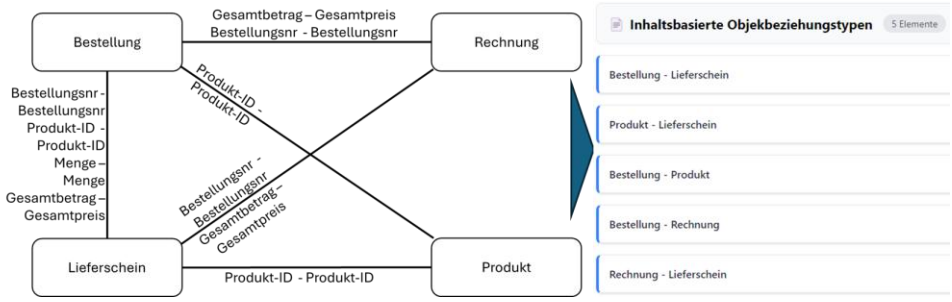


Abbildung 8-6: Datenmodell zu den Objekttypen (links) sowie extrahierte Objektbeziehungstypen (rechts)

Aus diesen ungerichteten Objektbeziehungstypen wird anhand der Objektinstanzen der betreffenden Objekttypen bewertet, welcher Objekttyp Input oder Output eines Aktivitätstyps ist. Anschließend werden die Aktivitätstypen bewertet und absteigend sortiert. Das Vorgehen zur Bewertung ist in Abschnitt 7.2.1.5 beschrieben. Mit den in Tabelle 8-5 definierten Gewichtungsfaktoren ergab sich für die einzelnen Aktivitätstypen die Bewertung in Abbildung 8-7.

```
[TERMINAL] Aktivitätstyp: (('Lieferschein'), ('Rechnung')) Bewertung: 13.212160741211266
[TERMINAL] Aktivitätstyp: (('Bestellung'), ('Lieferschein')) Bewertung: 11.240232196778987
[TERMINAL] Aktivitätstyp: (('Produkt'), ('Bestellung')) Bewertung: 6.0
[TERMINAL] Aktivitätstyp: (('Produkt'), ('Lieferschein')) Bewertung: 4.0
[TERMINAL] Aktivitätstyp: (('Bestellung'), ('Rechnung')) Bewertung: 3.4956413328657634
```

Abbildung 8-7: Bewertung der Aktivitätstypen (Detail-Modus)

Die bewerteten Aktivitätstypen werden anschließend absteigend nach ihrer Bewertung sortiert. Für jeden Aktivitätstyp wird geprüft, ob dieser im Geschäftsprozessmodell abgebildet werden sollte. Dazu wird geprüft, ob neue Objekttypen in dem betrachteten Aktivitätstyp verwendet werden und diese keine bereits verwendeten Objektinstanzen verbrauchen bzw. erzeugen. Zusätzlich wird geprüft, ob neue Attribute übernommen werden, und ob Zusammenführungen erfolgen sollten. Der Software-Prototyp bewertete für die bereitgestellten Dateien nur die ersten drei Aktivitätstypen als relevant, da diese alle Objekttypen, Objektinstanzen und übernommene Attribute abdecken. Zusätzlich wurden keine Zusammenführungen erkannt, da diese drei Aktivitäten keine gemeinsamen Objekttypen aufweisen und die zeitliche Reihenfolge in jeder Prozessinstanz gleich ist. Somit ergaben sich die in Abbildung 8-8 dargestellten Aktivitätstypen. Exemplarisch werden die übereinstimmenden Attribute zwischen den beteiligten Objekttypen für den Aktivitätstyp Bestellung-Lieferschein gezeigt.

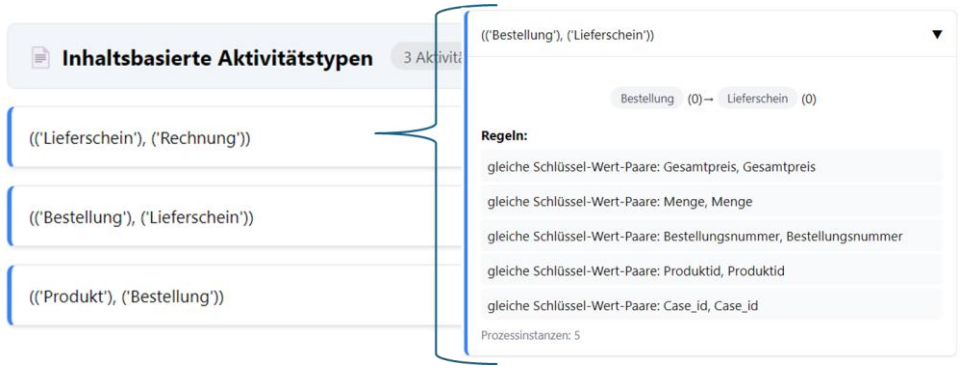


Abbildung 8-8: Inhaltsbasierte Aktivitätstypen (links) sowie Detail-Darstellung des Aktivitätstyp Bestellung-Lieferschein (rechts)

Durch das Zusammensetzen der Aktivitätstypen anhand der definierten Input- und Output-Objekttypen ergab sich das Petri-Netz in Abbildung 8-9. Da bei der Generierung basierend auf den inhaltlichen Zusammenhängen auch Objektinstanzen einbezogen werden, für die kein repräsentativer Zeitpunkt gefunden wurde (hier: *Produktdaten*), unterscheidet sich das generierte Geschäftsprozessmodell von dem in Abbildung 8-5.

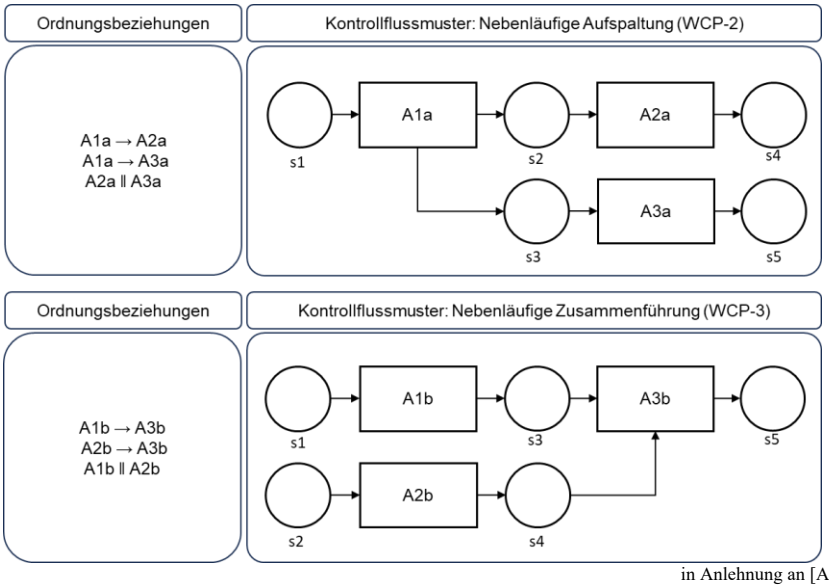
Abbildung 8-9: Aus den inhaltlich-basierten Aktivitätstypen abgeleitetes Petri-Netz

Ergebnis der Evaluation: Es wurden die inhaltlichen Zusammenhänge erkannt und zufällige Übereinstimmungen entfernt. Basierend auf den Eigenschaften der betreffenden Objektinstanzen eines Objektbeziehungstyps wurde die Richtung der Objektbeziehungen (Input/Output) korrekt bestimmt. Anschließend wurden entsprechend des Beispielprozesses so viele Aktivitätstypen als relevant erkannt, dass alle gefundenen Überschneidungen, Objekttypen und Objektinstanzen berücksichtig werden. Das resultierende Petri-Netz bildet die sequenziellen Ordnungsbeziehung der Aktivitätstypen sowie alle Objekttypen korrekt ab.

8.4.2 Nebenläufige Ordnungsbeziehungen (AND-Aufspaltung; AND-Zusammenführung)

Die nebenläufige Ordnungsbeziehung zwischen Aktivitäten wird mit dem Kontrollflussmuster Nebenläufige Aufspaltung (AND-Aufspaltung) oder Nebenläufige Zusammenführung (AND-Zusammenführung) repräsentiert. Das in Abbildung 8-10 oben dargestellte Kontrollflussmuster *Nebenläufige Aufspaltung* repräsentiert, dass ab Aktivität A1a der

Kontrollfluss in zwei (oder mehr) nebenläufige Pfade verzweigt. Dadurch können die Aktivitäten A2a und A3a nebenläufig ausgeführt werden. Das in Abbildung 8-10 unten dargestellte Kontrollflussmuster *Nebenläufige Aufspaltung* repräsentiert, dass zwei (oder mehr) nebenläufige Pfade nach Beenden der Aktivität A1b und Aktivitäten A2b wieder zu einem Pfad zusammengeführt werden. Aktivität A3b kann erst ausgeführt werden, wenn beide Aktivitäten beendet wurden (siehe Abschnitt 2.2.1.2).



in Anlehnung an [AHKB03]

Abbildung 8-10: Ordnungsbeziehungen und Kontrollflussmuster zur Nebenläufigkeit[3]

Um zu überprüfen, ob der Software-Prototyp diese Ordnungsbeziehungen korrekt extrahiert, wird ein Geschäftsprozess angenommen, bei dem zunächst eine *Bestellung* eingeht. Nach der *Bestätigung* der Bestellung wird der *Lieferschein* erstellt sowie die *Rechnung* versendet. Die Logistik-Abteilung übernimmt anschließend den *Versand*, während die Buchhaltung für die Prüfung des *Zahlungseingangs* anhand der Rechnung zuständig ist. Da der Versand und die Abwicklung der Zahlung von unterschiedlichen Abteilungen übernommen werden und keine gemeinsamen Objekte betreffen, sind diese Aktivitäten unabhängig voneinander. Liegt sowohl eine Versandbestätigung als auch die Zahlung vor, kann der Geschäftsprozess *beendet* werden.

Für diesen Geschäftsprozess wurden 30 Dateien[4] bereitgestellt, die vier Prozessinstanzen, betreffen, die aus je einer Objektinstanz der Objekttypen *Bestellung*, *Bestätigung*, *Liefer-*

[3] In [AHKB03] mit parallel split / join bezeichnet.

schein, *Rechnung*, *Zahlung* bestehen. Zusätzlich wurden *Kunden-* und *Produktdaten* bereitgestellt.

Aufgrund der Unabhängigkeit der Aktivitäten *Versand* und *Zahlungseingang prüfen*, werden die entsprechend zugehörigen Objekte (*Lieferschein-Versandbestätigung* sowie *Rechnung-Zahlung*) ebenfalls unabhängig zueinander verbraucht bzw. erzeugt. Während auf die *Bestellung* in den betrachteten Prozessinstanzen immer die *Bestätigung* folgte und der *Abschlussvermerk* immer zuletzt in den Prozessinstanzen auftrat, liegen die Objektinstanzen der Objekttypen *Lieferschein*, *Versandbestätigung*, *Rechnung* und *Zahlung* in den Prozessinstanzen in keiner eindeutigen Reihenfolge vor (siehe Abbildung 8-11).

Abbildung 8-11: Zeitpunkte der Objektinstanzen aus den Input-Dateien je Prozessinstanz

Nachfolgend werden in Abschnitt 8.4.2.1 zunächst die Ergebnisse zu den Zuordnungen zu Objekttypen und Prozessinstanzen vorgestellt. In den Abschnitten 8.4.2.2 und 8.4.2.3 wird die Prozessmodellgenerierung basierend auf den zeitlichen Zusammenhängen und basierend auf den inhaltlichen Zusammenhängen beschrieben sowie das Ergebnis evaluiert.

8.4.2.1 Informationsextraktion zu Objekttypen und Prozessinstanzen

Wie Abbildung 8-12 zeigt, wurden durch den Software-Prototyp die Dateien zu neun Clustern zugeordnet, vier Prozessinstanzen erkannt und die Objekttypen definiert sowie als prozessinstanzabhängig oder prozessinstanzunabhängig eingeordnet.

[4] Siehe https://gitlab.kit.edu/kit/aifb/BIS/kit-bis/objektbasiertemodellgenerierung/testszenarios/nebenlaeufigkeit-testdaten für das Beispielszenario.

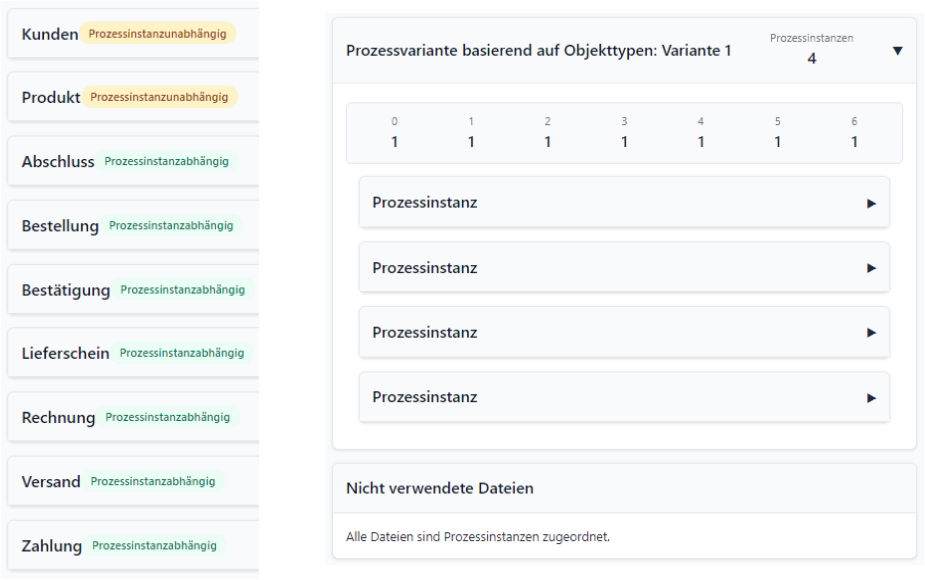


Abbildung 8-12: Ergebnis zu den Objekttypen und der Prozessinstanz-Zuordnung (eine Prozessvariante)

Ergebnis der Evaluation: Die Ergebnisse der Evaluation zeigen, dass die Objekttyp-Anzahl korrekt erkannt wurde und die Dateien entsprechend zugeordnet wurden. Ebenso wurden die Prozessinstanzen korrekt erkannt und die Objektinstanzen korrekt zugeordnet. Die Objekttypen wurden korrekt als prozessinstanzabhängig oder prozessinstanzunabhängig eingeordnet. Es wurde korrekt nur eine Prozessvariante erkannt.

8.4.2.2 Prozessmodellgenerierung basierend auf den zeitlichen Zusammenhängen

Da für die Objektinstanzen durch die nebenläufigen Aktivitäten keine einheitliche Reihenfolge anhand ihrer repräsentativen Zeitpunkte in den Prozessinstanzen erkannt wird, wurden zwölf zeitliche Objektbeziehungstypen für die bereitgestellten Dateien definiert. Ohne weitere Zusammenführungen wird aus diesen zunächst das Geschäftsprozessmodell in Abbildung 8-13 generiert.

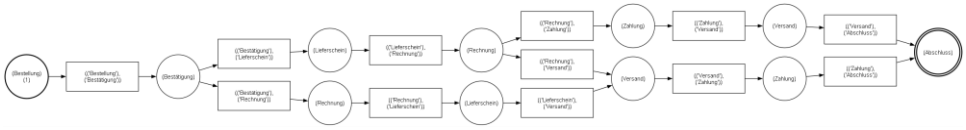


Abbildung 8-13: Aus den zeitlich-basierten Aktivitätstypen abgeleitetes Petri-Netz (vor Zusammenführung der Aktivitätstypen)

Dieses Geschäftsprozessmodell zeigt, dass die Objekttypen mehrfach im Geschäftsprozessmodell auftreten und in unterschiedlichen Aktivitätstypen erzeugt, gelesen, bearbeitet oder verbraucht werden. Daher werden die Objektbeziehungstypen basierend auf der Betrachtung der Ordnungsbeziehungen zwischen den beteiligten Objektinstanzen auf mögliche Zusammenführungen geprüft (siehe Abschnitt 6.2.1.3). Für die Objekttypen wurde zwischen der Rechnung und dem Lieferschein sowie dem Versand und der Zahlung inverse Ordnungsbeziehungen erkannt (siehe Abbildung 8-14).

	Abschluss_0	Bestellung_0	Bestätigung_0	Lieferschein_0	Rechnung_0	Versand_0	Zahlung_0
Abschluss_0	#	#	#	#	#	<	<
Bestellung_0	#	#	>	#	#	#	#
Bestätigung_0	#	<	#	>	>	#	#
Lieferschein_0	#	#	<	#	‖	>	#
Rechnung_0	#	#	<	‖	#	>	>
Versand_0	>	#	#	<	<	#	‖
Zahlung_0	>	#	#	#	<	‖	#

Abbildung 8-14: Footprint-Matrix zu den Objekttypen

Die Objektbeziehungstypen, die diese Objekttypen betreffen, wurden daher nach dem in Abschnitt 6.2.1.3 beschrieben Vorgehen zusammengeführt. Bei dieser Zusammenführung ist auch zu beachten, in welchen Prozessinstanzen die zuvor ermittelten Objektbeziehungstypen aufgetreten sind. Ausgehend von den zwölf Objektbeziehungstypen wurden fünf Aktivitätstypen definiert.

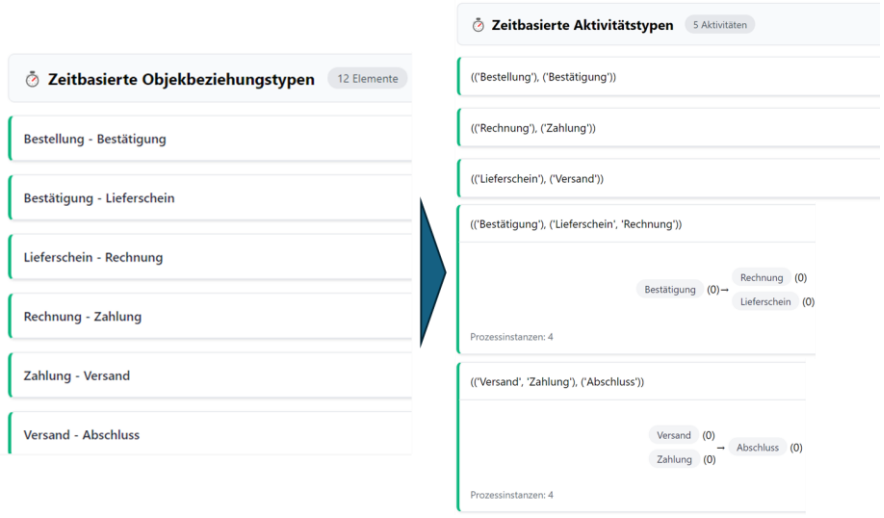


Abbildung 8-15: Erkannte Objektbeziehungstypen sowie die daraus definierten Aktivitätstypen

Durch die Zusammenführung der Aktivitätstypen sowie anschließender Generierung eines Eventlogs (siehe Abbildung 8-16) und Anwenden eines Process Discovery-Algorithmus wurde das Petri-Netz in Abbildung 8-17 generiert.

```
Total traces created: 4
[TERMINAL] Case 1:
[TERMINAL]   Activity: (('Bestellung'), ('Bestätigung')), StartTimestamp: 2023-11-14 12:00:00, CaseID: 0
[TERMINAL]   Activity: (('Bestätigung'), ('Lieferschein', 'Rechnung')), StartTimestamp: 2023-11-21 00:00:00, CaseID: 0
[TERMINAL]   Activity: (('Rechnung'), ('Zahlung')), StartTimestamp: 2023-12-01 00:00:00, CaseID: 0
[TERMINAL]   Activity: (('Versand', 'Zahlung'), ('Abschluss')), StartTimestamp: 2024-12-03 22:30:00, CaseID: 0
[TERMINAL] Case 2:
[TERMINAL]   Activity: (('Bestellung'), ('Bestätigung')), StartTimestamp: 2023-10-16 18:00:00, CaseID: 1
[TERMINAL]   Activity: (('Bestätigung'), ('Lieferschein', 'Rechnung')), StartTimestamp: 2023-10-27 04:30:00, CaseID: 1
[TERMINAL]   Activity: (('Lieferschein'), ('Versand')), StartTimestamp: 2023-11-03 12:00:00, CaseID: 1
[TERMINAL]   Activity: (('Versand', 'Zahlung'), ('Abschluss')), StartTimestamp: 2024-11-03 22:30:00, CaseID: 1
[TERMINAL] Case 3:
[TERMINAL]   Activity: (('Bestellung'), ('Bestätigung')), StartTimestamp: 2024-01-18 06:00:00, CaseID: 2
[TERMINAL]   Activity: (('Bestätigung'), ('Lieferschein', 'Rechnung')), StartTimestamp: 2024-01-24 21:00:00, CaseID: 2
[TERMINAL]   Activity: (('Lieferschein'), ('Versand')), StartTimestamp: 2024-02-11 00:00:00, CaseID: 2
[TERMINAL]   Activity: (('Rechnung'), ('Versand')), StartTimestamp: 2024-02-14 00:00:00, CaseID: 2
[TERMINAL]   Activity: (('Rechnung'), ('Zahlung')), StartTimestamp: 2024-02-17 12:00:00, CaseID: 2
[TERMINAL]   Activity: (('Versand', 'Zahlung'), ('Abschluss')), StartTimestamp: 2024-08-15 13:30:00, CaseID: 2
[TERMINAL] Case 4:
[TERMINAL]   Activity: (('Bestellung'), ('Bestätigung')), StartTimestamp: 2024-03-24 00:00:00, CaseID: 3
[TERMINAL]   Activity: (('Bestätigung'), ('Lieferschein', 'Rechnung')), StartTimestamp: 2024-03-31 18:00:00, CaseID: 3
[TERMINAL]   Activity: (('Lieferschein'), ('Versand')), StartTimestamp: 2024-04-14 00:00:00, CaseID: 3
[TERMINAL]   Activity: (('Versand', 'Zahlung'), ('Abschluss')), StartTimestamp: 2024-10-16 04:30:00, CaseID: 3
```

Abbildung 8-16: Generierter Eventlog basierend auf den zusammengeführten Aktivitätstypen (Terminal)

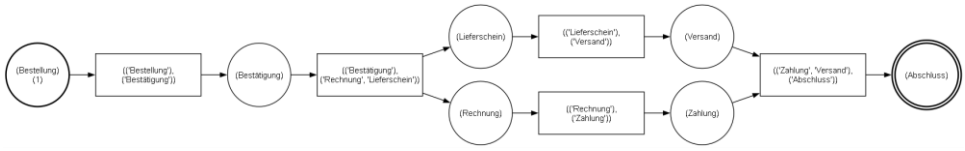


Abbildung 8-17: Aus den zeitlich-basierten Aktivitätstypen abgeleitetes Petri-Netz (Alpha Miner)

Ergebnis der Evaluation: Die zeitlichen Zusammenhänge zwischen den Objekttypen wurden entsprechend der Objektinstanzen korrekt erkannt und die Ordnungsbeziehungen zwischen den Aktivitätstypen entsprechend als sequenziell bzw. nebenläufig definiert. Die Footprint-Matrix identifizierte erfolgreich die nebenläufigen Ordnungsbeziehungen zwischen Lieferschein/Rechnung und Versandbestätigung/Zahlung. Mit jedem der Process Discovery-Algorithmen wurden die extrahierten Informationen in ein Petri-Netz transformiert, das die nebenläufigen Ordnungsbeziehungen der Aktivitätstypen durch entsprechende AND-Aufspaltungen und -Zusammenführungen korrekt repräsentiert.

8.4.2.3 Prozessmodellgenerierung basierend auf den inhaltlichen Zusammenhängen

Basierend auf den inhaltlichen Zusammenhängen zwischen den Objektinstanzen wurden zunächst 28 Objektbeziehungstypen vom Software-Prototyp erkannt (Abbildung 8-18).

Bei den inhaltlichen Zusammenhängen werden auch die prozessinstanzunabhängigen Objekte berücksichtigt.

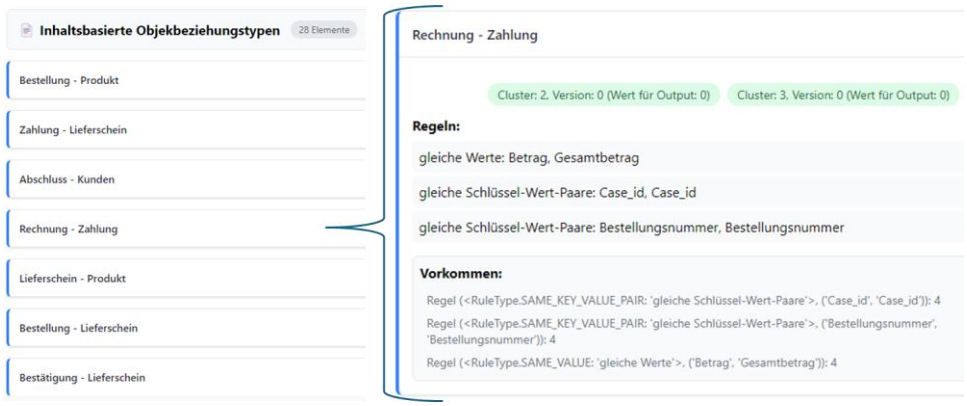


Abbildung 8-18: Inhaltsbasierte Aktivitätstypen (links) sowie Detail-Darstellung des Aktivitätstyp Rechnung-Zahlung (rechts)

Aus den ungerichteten Objektbeziehungstypen wurde anhand der Objektinstanzen der betreffenden Objekttypen bewertet, welcher Objekttyp Input oder Output eines Aktivitätstyps ist. Diese wurden mit denen in Tabelle 8-5 angegebenen Gewichtungsfaktoren bewertet und nach ihrer Bewertung absteigend sortiert. Die Aktivitätstypen mit ihrer Bewertung sind in Abbildung 8-19 gegeben. Basierend auf der Betrachtung der verwendeten Input- und Output-Objekttypen, der übereinstimmenden Attribute sowie der Objektinstanzen werden die Aktivitätstypen schrittweise als relevant für das Geschäftsprozessmodell ausgewählt.

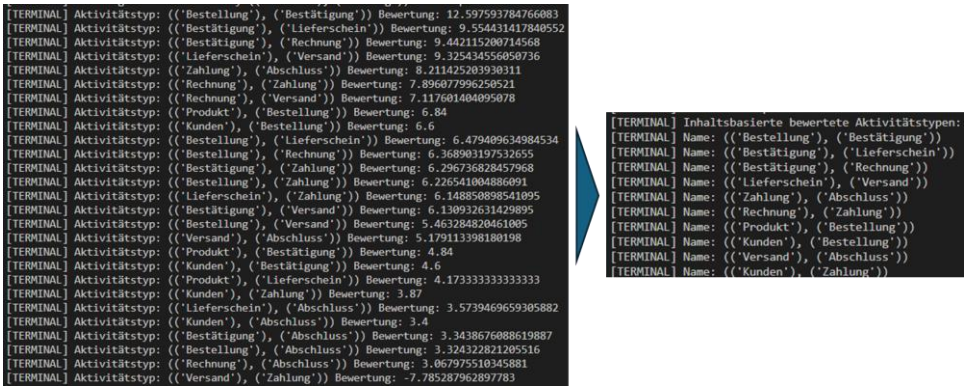


Abbildung 8-19 Bewertung der Aktivitätstypen und Auswahl (Terminal)

Die Aktivitätstypen, die gleiche Objekttypen betreffen, werden anschließend nach dem definierten Vorgehen in Abschnitt 6.1.5.2 auf Zusammenführungen geprüft. Die daraus ergebenen Aktivitätstypen sind in Abbildung 8-20 dargestellt. Bei einer Zusammenführung werden ebenfalls die Prozessinstanzen, in denen die Aktivitätsinstanzen der zusammenzuführenden Aktivitätstypen auftreten, sowie die Regeln der Aktivitätstypen berücksichtigt und aktualisiert.

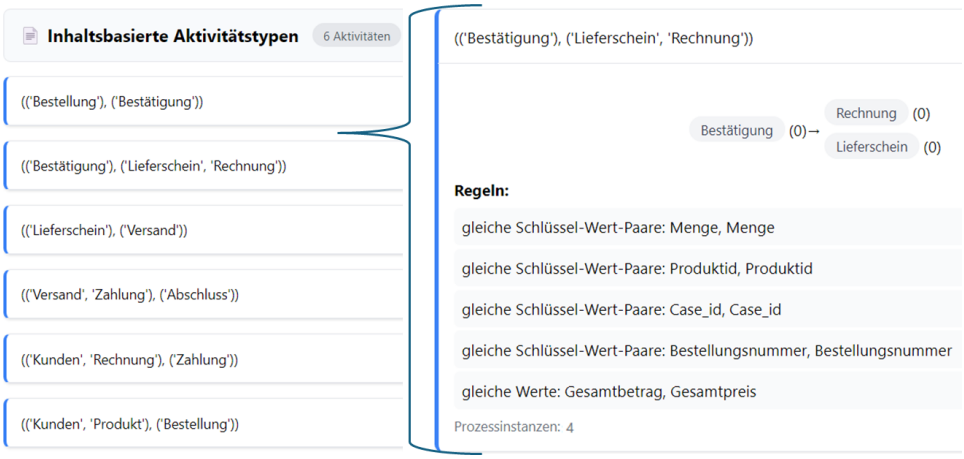


Abbildung 8-20: Abgeleitete und zusammengeführte Aktivitätstypen

Das resultierende Petri-Netz aus der Generierung basierend auf den zeitlichen und inhaltlichen Zusammenhängen ist in Abbildung 8-21 gegeben.

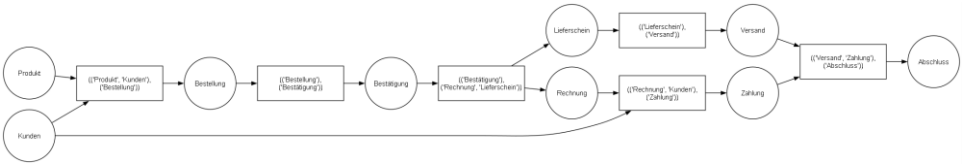


Abbildung 8-21: Aus den inhaltlich-basierten Aktivitätstypen abgeleitetes Petri-Netz

Da die Kunden- und Produktdaten prozessinstanzunabhängig sind und deren Zustände nicht in den Prozessinstanzen verändert wurden, sind die Kanten zu diesen als lesende Kanten (siehe Abschnitt 3.2.2) zu interpretieren. Der Software-Prototyp bietet daher zusätzlich die Möglichkeit, dies grafisch mit ungerichteten Kanten darzustellen (siehe Abbildung 8-22).

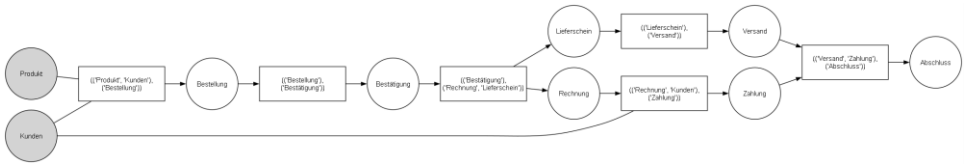
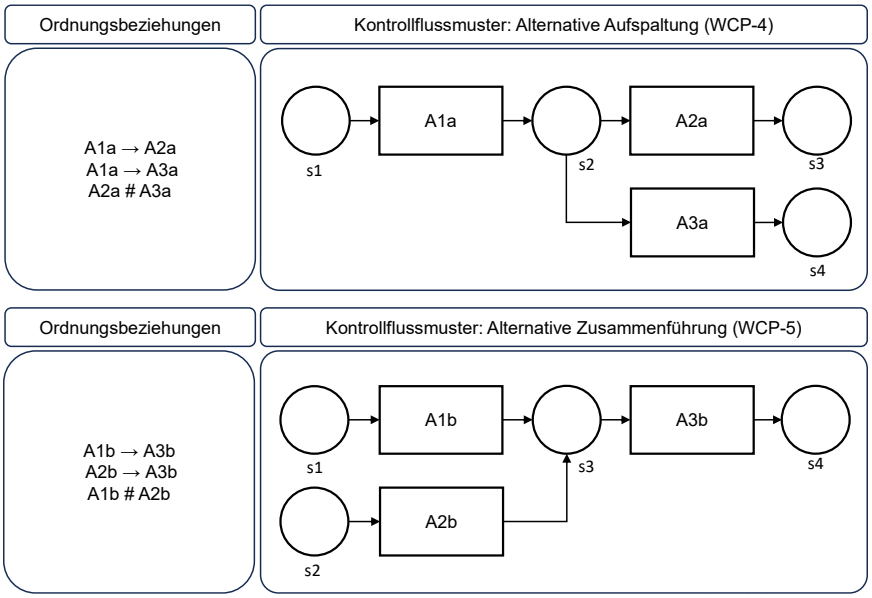


Abbildung 8-22: Aus den inhaltlich-basierten Aktivitätstypen abgeleitetes Petri-Netz mit Lesekanten

Ergebnis der Evaluation: Die inhaltlichen Zusammenhänge wurden korrekt erkannt und zufällige Überschneidungen entfernt. Ebenso wurden für die Aktivitätstypen die Richtung der Objektbeziehungen (Input/Output) korrekt bestimmt und Aktivitätstypen, die keine neuen Informationen berücksichtigen entfernt. Die Evaluation bestätigt, dass der Prototyp auch bei der vorliegenden nicht eindeutigen Reihenfolge der Objektinstanzen die korrekten Ordnungsbeziehungen erkennt und bei der Zusammenführung der Aktivitätstypen berücksichtigt. Das resultierende Petri-Netz bildet sowohl die sequenzielle bzw. nebenläufige Ordnungsbeziehung der Aktivitätstypen korrekt ab.

8.4.3 Alternative Ordnungsbeziehung (XOR-Aufspaltung; XOR-Zusammenführung)

Die alternative Ordnungsbeziehung zwischen Aktivitäten wird mit dem Kontrollflussmuster Alternative Aufspaltung (XOR-Aufspaltung) oder Alternative Zusammenführung (XOR-Zusammenführung) repräsentiert. Das in Abbildung 8-23 oben dargestellte Kontrollflussmuster *Alternative Verzweigung* repräsentiert, dass ab Aktivität A1a der Kontrollfluss in zwei (oder mehr) alternative Pfade verzweigt. Von diesem kann nur genau einer ausgewählt werden, sodass entweder Aktivitäten A2a oder A3a ausgeführt werden kann. Das in Abbildung 8-10 unten dargestellte Kontrollflussmuster *Alternative Aufspaltung* repräsentiert, dass zwei (oder mehr) alternative Pfade nach Beenden der Aktivität A1b oder Aktivitäten A2b wieder zu einem Pfad zusammengeführt werden. Aktivität A3b kann erst ausgeführt werden, wenn entweder Aktivität A1b oder Aktivitäten A2b beendet wurde (siehe Abschnitt 2.2.1.3).



in Anlehnung an [AHKB03]

Abbildung 8-23: Ordnungsbeziehungen und Kontrollflussmuster zur Alternative

Um zu überprüfen, ob der Software-Prototyp alternative Ordnungsbeziehungen korrekt extrahiert, wird ein Geschäftsprozess angenommen, bei dem zunächst eine *Bestellung* eingeht. Eine Bestellung kann entweder *bestätigt* oder *abgelehnt* werden. Bei einer Ablehnung endet der Geschäftsprozess. Bei einer Bestellung kann der Kunde zwischen Versand und Abholung wählen. Wird die Bestellung bestätigt und der Kunde hat einen Versand gewünscht, wird ein *Lieferschein erstellt*. Sowohl bei Versand als auch bei Abholung wird eine *Rechnung erstellt*. Nachdem der Kunde bezahlt hat, wird entweder die *Bestellung versendet* oder ein *Abholschein erstellt*. Anschließend wird der Geschäftsprozess *beendet*.

Für diesen Geschäftsprozess wurden 50 Dateien[5] bereitgestellt, die neun Prozessinstanzen betreffen. Bei vier Prozessinstanzen erfolgte auf die Bestellung eine Absage. Bei vier Prozessinstanzen wurde die Bestellung versendet und bei zwei Prozessinstanzen abgeholt. Es gibt dabei die Objekttypen *Bestellung*, *Bestätigung*, *Ablehnung*, *Lieferschein*, *Rechnung*, *Zahlung*, *Abholschein*, *Versandbestätigung* und *Abschluss*. Zusätzlich wurden *Kunden*- und *Produktdaten* bereitgestellt.

[5] Siehe https://gitlab.kit.edu/kit/aifb/BIS/kit-bis/objektbasiertemodellgenerierung/testszenarios/alternative-testdaten für das Beispielszenario.

Da die Aktivitätstypen *Bestellung annehmen* und *Bestellung ablehnen* sowie *Abholung durch Kunden* und *Versand* sich gegenseitig ausschließen, treten die entsprechend zugehörigen Objekte *Bestätigung* und *Ablehnung* sowie *Lieferschein* und *Abholschein* niemals in derselben Prozessinstanz auf (siehe Abbildung 8-24)

Abbildung 8-24: Objektinstanzen je Prozessinstanz-Variante

Nachfolgend werden in Abschnitt 8.4.3.1 zunächst die Ergebnisse zu den Zuordnungen zu Objekttypen und Prozessinstanzen vorgestellt. In den Abschnitten 8.4.3.2 und 8.4.3.3 wird die Prozessmodellgenerierung basierend auf den zeitlichen Zusammenhängen und basierend auf den inhaltlichen Zusammenhängen beschrieben sowie das Ergebnis evaluiert.

8.4.3.1 Informationsextraktion zu Objekttypen und Prozessinstanzen

Wie Abbildung 8-25 zeigt, wurden durch den Software-Prototyp die Dateien elf Clustern zugeordnet und zehn Prozessinstanzen in drei verschiedenen Varianten erkannt. Außerdem wurden die Objekttypen definiert sowie als prozessinstanzabhängig oder prozessinstanzunabhängig eingeordnet.

Abbildung 8-25: Ergebnis zu den Objekttypen und der Prozessinstanz-Zuordnung (3 Prozessvarianten)

Ergebnis der Evaluation: Die Ergebnisse der Evaluation zeigen, dass die Objekttyp-Anzahl korrekt erkannt wurde und die Dateien entsprechend zugeordnet wurden. Ebenso wurden die Prozessinstanzen und Varianten korrekt erkannt und die Objektinstanzen zu den Prozessinstanzen korrekt zugeordnet. Die Objekttypen wurden korrekt als prozessinstanzabhängig oder prozessinstanzunabhängig eingeordnet.

8.4.3.2 Prozessmodellgenerierung basierend auf den zeitlichen Zusammenhängen

Wenn die Aktivitäten eines Geschäftsprozesses außer sequenziellen Ordnungsbeziehungen nur alternative Ordnungsbeziehungen aufweisen, weisen die Objektinstanzen wie in Abschnitt 8.4.1 eine eindeutige Reihenfolge auf. Basierend auf den zeitlichen Zusammenhängen wurden elf Objektbeziehungstypen abgeleitet (siehe Abbildung 8-26), die nicht weiter zusammengeführt werden konnten. Daher resultierten die Objektbeziehungstypen in elf Aktivitätstypen.

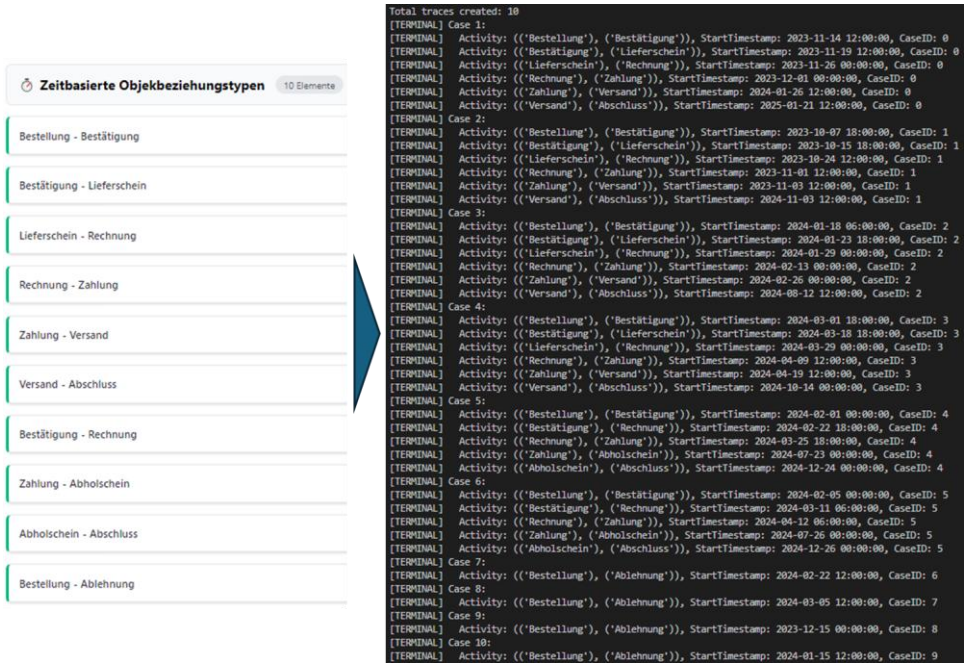


Abbildung 8-26: Erkannte Objektbeziehungstypen aus den zeitlichen Zusammenhängen (links) sowie generierter Eventlog (rechts, Terminal)

Das Ergebnis ist für jeden im Software-Prototyp zur Verfügung gestellten Process Discovery-Algorithmus (*Alpha Miner*, *Alpha Miner Plus*, *Heuristic Miner* und *Inductive Miner)* jeweils das Petri-Netz in Abbildung 8-27. Da für prozessinstanzunabhängige Objekte kein repräsentativer Zeitpunkt zu einer Prozessinstanz angenommen werden kann, werden diese bei der Generierung basierend auf den zeitlichen Zusammenhängen ausgeschlossen.

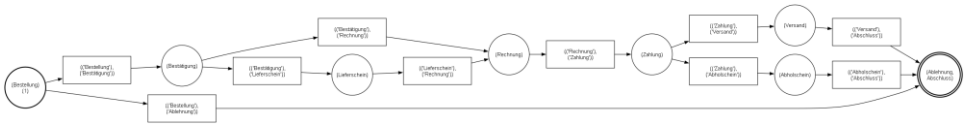


Abbildung 8-27: Aus den zeitlich-basierten Aktivitätstypen abgeleitetes Petri-Netz (Alpha Miner)

Ergebnis der Evaluation: Die zeitlichen Zusammenhänge zwischen den Objekttypen wurden entsprechend der Objektinstanzen korrekt erkannt und die Ordnungsbeziehungen zwischen den Aktivitätstypen entsprechend als sequenziell bzw. alternativ definiert. Der Algorithmus zur Erstellung der Footprint-Matrix identifizierte erfolgreich die sich ausschließenden Beziehungen zwischen Bestätigung/Ablehnung sowie Lieferschein/Abholschein. Die zur Verfügung gestellten Process Discovery-Algorithmen

generierten aus den extrahierten Informationen jeweils ein Petri-Netz, das die alternativen Ordnungsbeziehungen der Aktivitätstypen durch entsprechende XOR-Aufspaltungen und -Zusammenführungen korrekt repräsentiert.

8.4.3.3 Prozessmodellgenerierung basierend auf den inhaltlichen Zusammenhängen

Basierend auf den inhaltlichen Zusammenhängen zwischen den Objektinstanzen wurden zunächst 38 Objektbeziehungstypen vom Software-Prototyp erkannt. Dabei werden nun auch zusätzlich die prozessin-stanzunabhängigen Objekte berücksichtigt. Aus diesen ungerichteten Objektbeziehungstypen wurde anhand der Objektinstanzen der betreffenden Objekttypen bewertet, welcher Objekttyp Input oder Output eines Aktivitätstyps ist und diese mit denen in Tabelle 8-5 angegebenen Gewichtungsfaktoren bewertet und absteigend sortiert. Die Aktivitätstypen sowie deren Bewertung sind in Abbildung 8-28 gegeben. Basierend auf der Betrachtung der verwendeten Input- und Output-Objekttypen, der übergebenen Attribute sowie der Objektinstanzen werden die Aktivitätstypen schrittweise als relevant für das Geschäftsprozessmodell ausgewählt. Die Aktivitätstypen, die gleiche Objekttypen betreffen, werden anschließend nach dem definierten Vorgehen in Abschnitt 6.1.5.2 auf Zusammenführungen geprüft. Dabei wurde erkannt, dass alternative Ordnungsbeziehungen vorhanden sind. Die daraus ergebenen Aktivitätstypen sind in Abbildung 8-28 dargestellt.

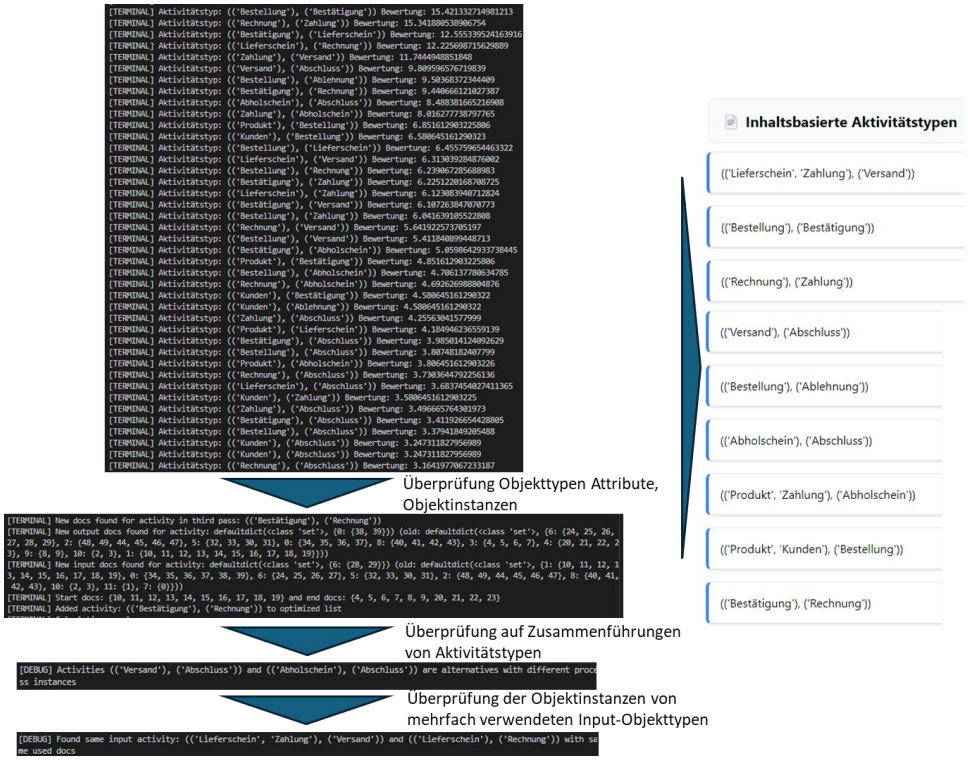


Abbildung 8-28 Bewertung der Aktivitätstypen (links, Terminal) sowie resultierende Aktivitätstypen (rechts)

Das resultierende Petri-Netz aus der Generierung basierend auf den zeitlichen und inhaltlichen Zusammenhängen ist in Abbildung 8-29 gegeben.

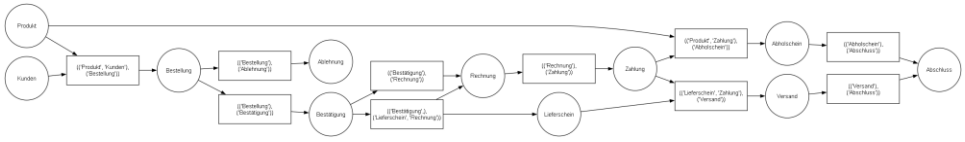


Abbildung 8-29: Aus den inhaltlich-basierten Aktivitätstypen abgeleitetes Petri-Netz

Ergebnis der Evaluation: Die inhaltlichen Zusammenhänge wurden ebenfalls korrekt erkannt und zufällige Überschneidungen entfernt. Ebenso wurden für die Aktivitätstypen die Richtung der Objektbeziehungen (Input/Output) korrekt bestimmt. Aktivitätstypen, die keine neuen Informationen berücksichtigen, wurden entfernt. Die Evaluation zeigt, dass der Software-Prototyp die sich gegenseitig ausschließenden Aktivitätstypen erkennt und bei der Zusammenführung der Aktivitätstypen berücksichtigt. Das resultierende Petri-Netz bildet sowohl die sequenziellen als auch die alternativen Ordnungsbeziehun-

gen der Aktivitätstypen korrekt ab. Im Vergleich zur Generierung basierend auf den zeitlichen Zusammengängen, wird der Zusammenhang zwischen Lieferschein und Versand erkannt und somit explizit im Geschäftsprozessmodell abgebildet (durch die Aktivität ((Lieferschein, Zahlung), (Versand))). Dieser Zusammenhang ist in den Petri-Netzen aus den zeitlichen Zusammenhängen nicht erkennbar.

8.4.4 Ordnungsbeziehungen zwischen gleichen Aktivitätstypen (Schleife)

Das Kontrollflussmuster *Schleife* repräsentiert, dass die Aktivitäten A1 wiederholt nacheinander ausgeführt werden kann (siehe Abbildung 8-30). Somit hat die Aktivität eine sequenzielle Ordnungsbeziehung zu sich selbst.

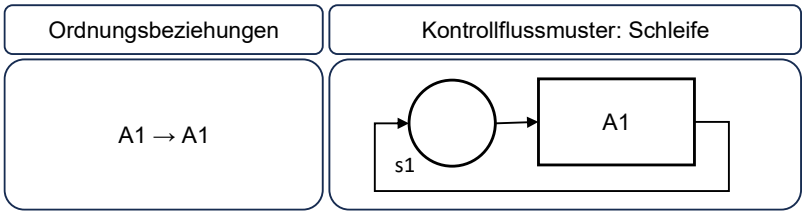


Abbildung 8-30: Ordnungsbeziehung und Kontrollflussmuster zur Schleife

Um zu überprüfen, ob der Software-Prototyp Schleifen erkennt, wird ein Geschäftsprozess angenommen, bei dem zunächst eine *Bestellung* eingeht. Auf diese Bestellung folgt anschließend eine *Bestätigung*. In einzelnen Fällen ruft jedoch nochmal der Kunde an und die erstellte Bestätigung wird angepasst. Wenn die Bestätigung den gewünschten Inhalt hat, wird die *Rechnung erstellt* und mit dem *Zahlungseingang* endet der Geschäftsprozess.

Für diesen Geschäftsprozess wurden 18[6] (bzw. 20[7]) Dateien bereitgestellt, die vier Prozessinstanzen betreffen, die Objektinstanzen der Objekttypen *Bestellung*, *Bestätigung*, *Rechnung*, *Zahlung* beinhalten. Zusätzlich wurden *Kunden*- und *Produktdaten* bereitgestellt.

[6] Siehe https://gitlab.kit.edu/kit/aifb/BIS/kit-bis/objektbasiertemodellgenerierung/testszenarios/schleife2-testdaten für das Beispielszenario.

[7] Siehe https://gitlab.kit.edu/kit/aifb/BIS/kit-bis/objektbasiertemodellgenerierung/testszenarios/schleife-testdaten für das Beispielszenario.

Aufgrund der Möglichkeit zur Überarbeitung der *Bestätigung* können mehrfache Objektinstanzen des Objekttyps in einer Prozessinstanz auftreten. Daher werden auch Objektbeziehungstypen von Objekttypen zu sich selbst extrahiert.

Nachfolgend werden in Abschnitt 8.4.4.1 zunächst die Ergebnisse zu den Zuordnungen zu Objekttypen und Prozessinstanzen vorgestellt. In den Abschnitten 8.4.4.2 und 8.4.4.3 wird die Prozessmodellgenerierung basierend auf den zeitlichen Zusammenhängen und basierend auf den inhaltlichen Zusammenhängen beschrieben sowie das Ergebnis evaluiert.

8.4.4.1 Informationsextraktion zu Objekttypen und Prozessinstanzen

Wie Abbildung 8-31 zeigt, wurden durch den Software-Prototyp die Dateien zu sechs Clustern zugeordnet und zwei Prozessinstanzen in zwei verschiedenen Varianten erkannt. Außerdem wurden die Objekttypen definiert sowie als prozessinstanzabhängig oder prozessinstanzunabhängig eingeordnet. In der Prozessvariante 1 ist ersichtlich, dass der Objekttyp *Bestätigung* (Cluster 0) zweimal auftritt. In Prozessvariante 2 existiert nur eine Bestätigung. Außerdem liegt für diese Variante kein Zahlungseingang vor. Zusätzlich wurde dieser-Beispielprozess auch mit drei Objektinstanzen der Bestätigung in den Prozessinstanzen getestet.

Abbildung 8-31: Ergebnis zu den Objekttypen und der Prozessinstanz-Zuordnung (Prozessvarianten)

Ergebnis der Evaluation: Die Ergebnisse der Evaluation zeigen, dass die Objekttyp-Anzahl korrekt erkannt wurde und die Dateien entsprechend zugeordnet wurden. Ebenso wurden die Prozessinstanzen und Varianten korrekt erkannt und die Objektinstanzen zu den Prozessinstanzen korrekt zugeordnet. Die Objekttypen wurden korrekt als prozessinstanzabhängig oder prozessinstanzunabhängig eingeordnet.

8.4.4.2 Prozessmodellgenerierung basierend auf den zeitlichen Zusammenhängen

Die Ordnungsbeziehungen zwischen den Aktivitätstypen des Geschäftsprozesses sind sequenziell. Daher weisen die Objektinstanzen wie in Abschnitt 8.4.1 eine eindeutige Reihenfolge auf. Basierend auf den zeitlichen Zusammenhängen ergaben sich somit für die Dokumente vier Objektbeziehungstypen (siehe Abbildung 8-32 links), die nicht weiter zusammengeführt werden konnten und daher in vier Aktivitätstypen resultierten.

Abbildung 8-32: Erkannte Objektbeziehungstypen aus den zeitlichen Zusammenhängen (links) sowie generierter Eventlog (rechts, Terminal)

Die Process Discovery-Algorithmen *Alpha Miner*, *Alpha Miner Plus*, *Heuristic Miner* und *Inductive Miner* generieren basierend auf dem aus den Aktivitätstypen und zugehörigen Aktivitätsinstanzen erzeugten Eventlog (siehe Abbildung 8-32 rechts) die Petri-Netze in Abbildung 8-33 bis Abbildung 8-35. Die grauen Transitionen sind *Hilfstransitionen*, die keinen Aktivitätstyp des Eventlogs abbilden.

Abbildung 8-33: Aus den zeitlich-basierten Aktivitätstypen abgeleitetes Petri-Netz (Alpha Miner)

Abbildung 8-34: Aus den zeitlich-basierten Aktivitätstypen abgeleitetes Petri-Netz (Heuristic und Inductive Miner)

In den generierten Petri-Netzen werden keine Schleifen abgebildet. Es wurde erst eine Schleife von den Process Discovery-Algorithmen erkannt, wenn der Aktivitätstyp Bestätigung-Bestätigung mehrmals auftritt (Beispiel 2, mit je 3 Bestellungen in den Prozessinstanzen). Dadurch ergab sich das Petri-Netz in Abbildung 8-35.

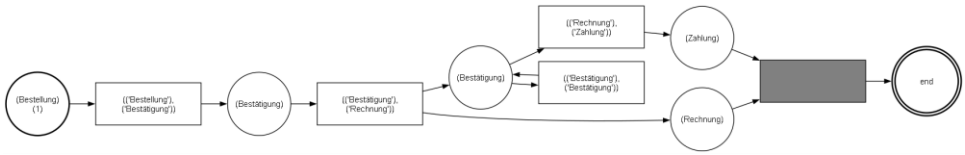


Abbildung 8-35: Aus den zeitlich-basierten Aktivitätstypen abgeleitetes Petri-Netz (Alpha Plus)

Bei der Generierung durch den Heuristic-Miner wurde erkannt, dass eine mehrfache Ausführung eines Aktivitätstyp stattgefunden hat (siehe Abbildung 8-36). Dies wird nicht mit einer Schleife dargestellt, sondern durch eine zusätzliche Transition, die eine erneute Markierung der Stelle im Vorbereich der Transition Bestätigung-Betätigung ermöglicht.

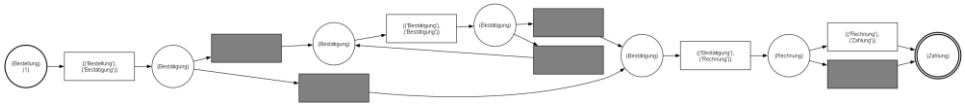


Abbildung 8-36: Aus den zeitlich-basierten Aktivitätstypen abgeleitetes Petri-Netz (Heuristic Miner)

Die Aktivitätstypen, die sich aus den zeitlichen Zusammenhängen ergeben, ergeben basierend auf dem Zusammensetzen nach ihren Input- und Output-Objekttypen das Petri-Netz in Abbildung 8-37.

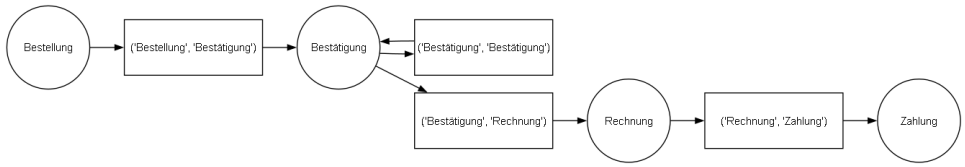


Abbildung 8-37: Aus den zeitlich-basierten Aktivitätstypen abgeleitetes Petri-Netz (zusammengesetzt)

Ergebnis der Evaluation: Die zeitlichen Zusammenhänge zwischen den Objekttypen wurden entsprechend den Objektinstanzen korrekt erkannt und die Ordnungsbeziehungen zwischen den Aktivitätstypen entsprechend als sequenziell definiert. Die verschiedenen Process Discovery-Algorithmen (Alpha Miner, Alpha Miner Plus, Heuristic Miner und Inductive Miner) erzeugten dabei unterschiedliche Repräsentationen der Aktivitätstypen im Petri-Netz. Dabei werden jedoch keine Schleifen oder Rückführungen modelliert. Stattdessen wird eine Transition *Bestätigung-Bestätigung* mit unterschiedlichen Stellen im Vor- und Nachbereich abgebildet. Wird der Testdatensatz mit je 3 Objektinstanzen der Bestätigung verwendet, identifizierte der Alpha Plus Miner korrekt die Schleife. Die anderen Algorithmen erkannten zwar die mehrfache Ausführung, repräsentierten die extrahierten Informationen jedoch als zyklische Ordnungsbeziehungen der Aktivitätsty-

pen durch eine Rückführung mittels zusätzlicher Transition und entsprechender Kanten. Das Petri-Netz (zusammengesetzt) bildet die Schlaufe korrekt ab.

8.4.4.3 Prozessmodellgenerierung basierend auf den inhaltlichen Zusammenhängen

Treten unterschiedliche Objektinstanzen des gleichen Objekttyps mehrmals in einer Prozessinstanz auf, wird bei den Zusammenhängen durch eine Nummerierung der Objektinstanzen unterschieden, welche Objektinstanz es betrifft. Für diese wird anhand ihres Zeitpunktes in der Prozessinstanz sowie der Ähnlichkeit zu den Objektinstanzen des gleichen Objekttyps innerhalb einer Prozessinstanz eine Versionsnummer zugeteilt. Sind die Attributwerte der Objektinstanzen unterschiedlich, werden die Objektinstanzen nicht als Versionen einer Objektinstanz zu verschiedenen Zeitpunkten erkannt, sondern als unterschiedliche Objektinstanzen, die außer dem gleichen Objekttyp zunächst keinen Zusammenhang aufweisen. Für die Dateien wurden somit zwischen den Objektinstanzen zunächst 17 Objektbeziehungstypen vom Software-Prototyp erkannt. Abbildung 8-38 zeigt die erkannten Objektbeziehungstypen zwischen *Bestätigung* und *Rechnung* mit unterschiedlichen Objektinstanzen.

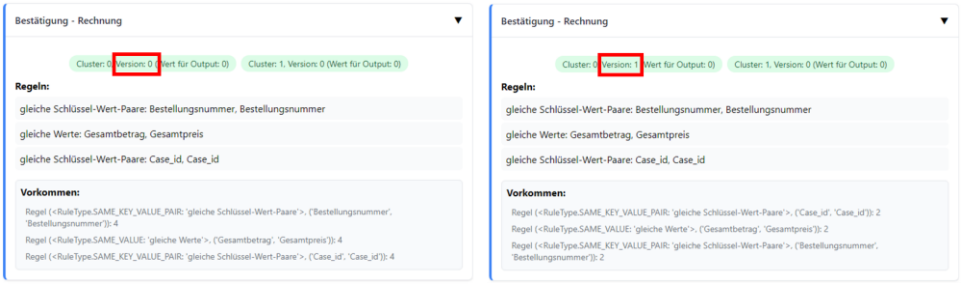


Abbildung 8-38: Objektbeziehungstyp Bestätigung – Rechnung (links Bestätigung der Version 0, rechts Bestätigung der Version 1)

Aus diesen ungerichteten Objektbeziehungstypen wurde anhand der Objektinstanzen der betreffenden Objekttypen bewertet, welcher Objekttyp Input oder Output eines Aktivitätstyps ist. Die Aktivitäten wurden anschließend mit denen in Tabelle 8-5 angegebenen Gewichtungsfaktoren bewertet und absteigend sortiert.

Die Aktivitätstypen sowie deren Bewertung werden in Abbildung 8-39 gezeigt. Basierend auf der Betrachtung der verwendeten Input- und Output-Objekttypen, der übergebenen Attribute sowie der Objektinstanzen werden die Aktivitätstypen schrittweise als relevant für das Geschäftsprozessmodell bewertet. Die Aktivitätstypen, die gleiche Objekttypen betreffen, werden anschließend nach dem definierten Vorgehen in Abschnitt

6.1.5.2 auf mögliche Zusammenführungen hin geprüft. Die daraus ergebenen Aktivitätstypen sind in Abbildung 8-39 dargestellt.

Abbildung 8-39 Bewertung der Aktivitätstypen (Terminal) sowie resultierende Aktivitätstypen

Das resultierende Petri-Netz aus der Generierung basierend auf den zeitlichen und inhaltlichen Zusammenhängen ist in Abbildung 8-40 gegeben.

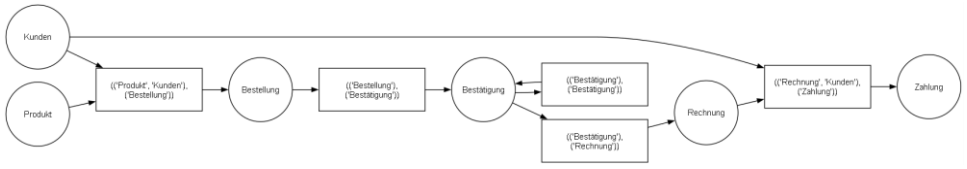


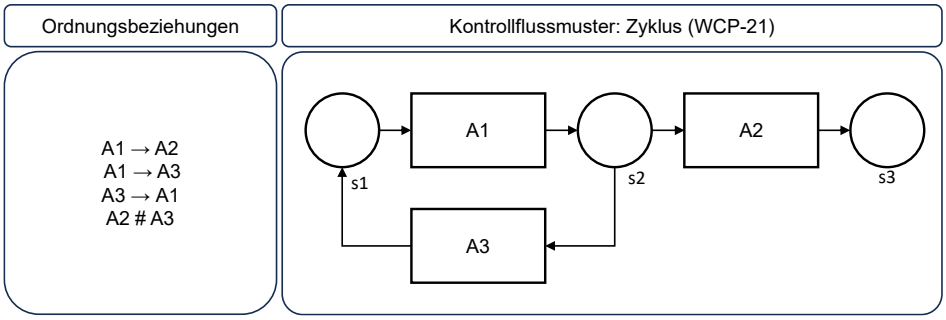
Abbildung 8-40: Aus den inhaltlich-basierten Aktivitätstypen abgeleitetes Petri-Netz

Ergebnis der Evaluation: Die inhaltlichen Zusammenhänge wurden ebenfalls korrekt erkannt, wobei mehrfach auftretende Objektinstanzen des gleichen Typs korrekt durch Versionsnummern unterschieden wurden. Ebenso wurden für die Aktivitätstypen die Richtung der Objektbeziehungen (Input/Output) korrekt bestimmt und die Objekttypbeziehungen zwischen verschiedenen Versionen des gleichen Objekttyps erkannt. Die Evaluation zeigt, dass der Prototyp die mehrfache Ausführung von Aktivitätstypen erkennt und bei der Zusammenführung der Aktivitätstypen berücksichtigt. Das resultierende Petri-Netz bildet die sequenziellen Ordnungsbeziehungen der Aktivitätstypen korrekt ab. Die Schleife wird im Unterschied zu der Generierung mit Process Discovery-Algorithmen bereits bei zwei Objektinstanzen eines Objekttyps erkannt.

8.4.5 Zyklus

Eine Rückführung des Kontrollflusses wird mit dem Kontrollflussmuster *Zyklus* repräsentiert (siehe Abbildung 8-23). Die Aktivitäten A1 und A3 können wiederholt nachei-

nander ausgeführt werden. Nach einer Beendigung der Aktivität A kann Aktivität 2 ausgeführt werden.



in Anlehnung an [AHKB03]

Abbildung 8-41: Ordnungsbeziehungen und Kontrollflussmuster zum Zyklus

Um zu überprüfen, ob der Software-Prototyp Zyklen erkennt, wird ein Geschäftsprozess angenommen, bei dem zunächst eine *Bestellung* eingeht. Auf diese Bestellung folgt entweder eine *Ablehnung* oder eine *Bestätigung*. Nach der *Bestätigung* der Bestellung werden der *Lieferschein* sowie die *Rechnung* erstellt. In einzelnen Fällen ruft jedoch nochmal der Kunde an und möchte die Bestellung ändern. Daher werden dann die bereits erstellte Rechnung und der erstellte Lieferschein archiviert und es wird eine neue Bestellung erstellt. Anschließend erfolgt die erneute Erstellung der Rechnung und des Lieferscheins. Mit dem *Zahlungseingang* endet der Geschäftsprozess.

Für diesen Geschäftsprozess wurden 33 Dateien[8] bereitgestellt, die sechs Prozessinstanzen betreffen und die Objektinstanzen der Objekttypen *Bestellung*, *Bestätigung*, *Lieferschein*, *Rechnung*, *Zahlung* beinhalten. Zusätzlich wurden *Kunden*- und *Produktdaten* bereitgestellt.

Aufgrund der Möglichkeit zur nachträglichen Überarbeitung der *Bestätigung* werden mehrfache Dateien zu unterschiedlichen Zeitpunkten einer Objektinstanz bereitgestellt. Daher gibt es mehrere Dateien zu den Objekttypen *Bestellung*, *Bestätigung*, *Lieferschein* und *Rechnung* in einer Prozessinstanz und Aktivitätstypen werden mehrfach in einer Prozessinstanz ausgeführt.

Nachfolgend werden in Abschnitt 8.4.5.1 zunächst die Ergebnisse zu den Zuordnungen zu Objekttypen und Prozessinstanzen vorgestellt. In den Abschnitten 8.4.5.2 und 8.4.5.3 wird die Prozessmodellgenerierung basierend auf den zeitlichen Zusammenhängen und

[8] Siehe https://gitlab.kit.edu/kit/aifb/BIS/kit-bis/objektbasiertemodellgenerierung/testszenarios/zyklus-testdaten für das Beispielszenario.

basierend auf den inhaltlichen Zusammenhängen beschrieben sowie das Ergebnis evaluiert.

8.4.5.1 Informationsextraktion zu Objekttypen und Prozessinstanzen

Wie Abbildung 8-42 zeigt, wurden durch den Software-Prototyp die Dateien zu acht Clustern zugeordnet und sechs Prozessinstanzen in vier verschiedenen Varianten erkannt. Außerdem wurden die Objekttypen definiert und als Prozessinstanzobjekt oder prozessinstanzunabhängiges Objekt eingeordnet. In der Prozessvariante 2 ist ersichtlich, dass die Objekttypen *Bestellung* (Cluster 0), *Bestätigung* (Cluster 1), *Lieferschein* (Cluster 2) und *Rechnung* (Cluster 3) jeweils zweimal auftreten.

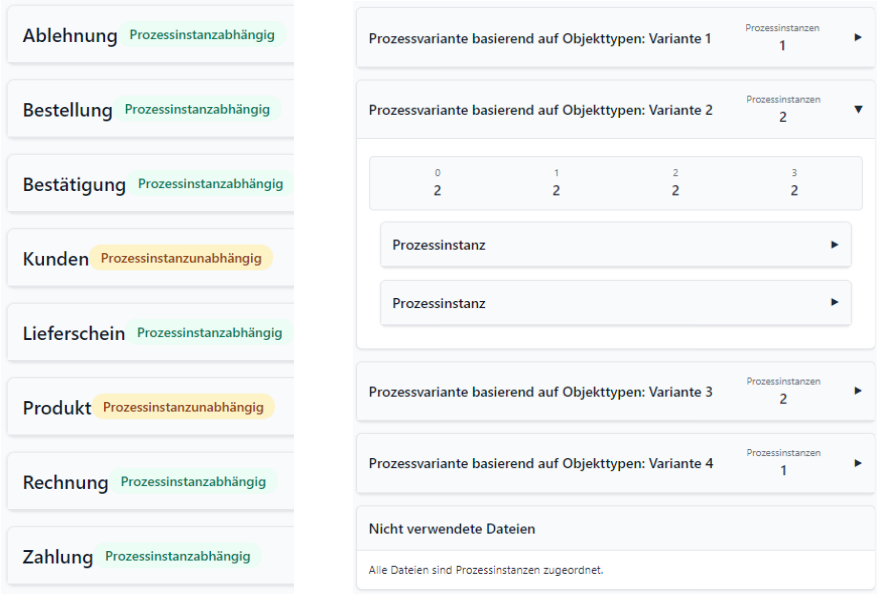


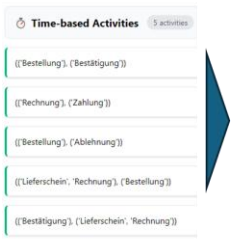
Abbildung 8-42: Ergebnis zu den Objekttypen und der Prozessinstanz-Zuordnung

Ergebnis der Evaluation: Die Ergebnisse der Evaluation zeigen, dass die Objekttyp-Anzahl korrekt erkannt wurde und die Dateien entsprechend zugeordnet wurden. Ebenso wurden die Prozessinstanzen, ihre Varianten und die mehrfach auftretenden Objektinstanzen korrekt erkannt und zugeordnet. Die Objekttypen wurden korrekt als prozessinstanzabhängig oder prozessinstanzunabhängig eingeordnet. Es wurde korrekt vier Prozessvariante erkannt.

8.4.5.2 Prozessmodellgenerierung basierend auf den zeitlichen Zusammenhängen

Die Aktivitätstypen des Geschäftsprozesses können in sequenzieller Beziehung zueinander stehen oder alternativ zueinander stattfinden. Daher weisen die Aktivitätstypen wie in Abschnitten 8.4.1 und 8.4.3 beschrieben eine eindeutige Reihenfolge auf. Somit zeigen auch die Objektinstanzen der Aktivitätstypen eine eindeutige Reihenfolge auf, wenn die Aktivitätstypen nur ein Objekt erzeugen. Da in der Bestätigungsverarbeitung sowohl ein Lieferschein als auch die Rechnung erzeugt werden, treten die Objektinstanzen in unterschiedlichen Reihenfolgen auf.

Daher ergaben sich für die Objektinstanzen basierend auf den zeitlichen Zusammenhängen neun Objektbeziehungstypen, die in fünf Aktivitätstypen resultieren (siehe Abbildung 8-43 links). Die Aktivitätsinstanzen der ermittelten Aktivitätstypen ergeben den Eventlog in Abbildung 8-43 (rechts).




```
Total traces created: 6
[DEBUG] Case 1:
[DEBUG]   Activity: (('Bestellung'), ('Bestätigung')), StartTimestamp: 2023-11-14 18:00:00, CaseID: 0
[DEBUG]   Activity: (('Bestätigung'), ('Rechnung', 'Lieferschein')), StartTimestamp: 2023-11-21 12:00:00, CaseID: 0
[DEBUG]   Activity: (('Rechnung'), ('Zahlung')), StartTimestamp: 2024-01-25 12:00:00, CaseID: 0
[DEBUG] Case 2:
[DEBUG]   Activity: (('Bestellung'), ('Bestätigung')), StartTimestamp: 2023-10-16 18:00:00, CaseID: 1
[DEBUG]   Activity: (('Bestätigung'), ('Rechnung', 'Lieferschein')), StartTimestamp: 2023-10-27 02:15:00, CaseID: 1
[DEBUG]   Activity: (('Rechnung', 'Lieferschein'), ('Bestellung')), StartTimestamp: 2023-11-06 18:00:00, CaseID: 1
[DEBUG]   Activity: (('Bestellung'), ('Bestätigung')), StartTimestamp: 2023-11-13 12:00:00, CaseID: 1
[DEBUG]   Activity: (('Bestätigung'), ('Rechnung', 'Lieferschein')), StartTimestamp: 2023-12-08 12:00:00, CaseID: 1
[DEBUG] Case 3:
[DEBUG]   Activity: (('Bestellung'), ('Bestätigung')), StartTimestamp: 2024-01-19 00:00:00, CaseID: 2
[DEBUG]   Activity: (('Bestätigung'), ('Rechnung', 'Lieferschein')), StartTimestamp: 2024-01-25 19:30:00, CaseID: 2
[DEBUG]   Activity: (('Rechnung', 'Lieferschein'), ('Bestellung')), StartTimestamp: 2024-07-23 00:00:00, CaseID: 2
[DEBUG]   Activity: (('Bestellung'), ('Bestätigung')), StartTimestamp: 2025-01-19 00:00:00, CaseID: 2
[DEBUG]   Activity: (('Bestätigung'), ('Rechnung', 'Lieferschein')), StartTimestamp: 2025-01-27 12:00:00, CaseID: 2
[DEBUG] Case 4:
[DEBUG]   Activity: (('Bestellung'), ('Bestätigung')), StartTimestamp: 2024-03-25 12:00:00, CaseID: 3
[DEBUG]   Activity: (('Bestätigung'), ('Rechnung', 'Lieferschein')), StartTimestamp: 2024-03-31 21:45:00, CaseID: 3
[DEBUG] Case 5:
[DEBUG]   Activity: (('Bestellung'), ('Bestätigung')), StartTimestamp: 2023-12-12 06:00:00, CaseID: 4
[DEBUG]   Activity: (('Bestätigung'), ('Rechnung', 'Lieferschein')), StartTimestamp: 2023-12-17 14:15:00, CaseID: 4
[DEBUG] Case 6:
[DEBUG]   Activity: (('Bestellung'), ('Ablehnung')), StartTimestamp: 2023-12-15 00:00:00, CaseID: 5
```

Abbildung 8-43: Erkannte Objektbeziehungstypen aus den zeitlichen Zusammenhängen sowie generierter Eventlog (Terminal)

Die Process Discovery-Algorithmen *Alpha Miner*, *Alpha Miner Plus*, *Heuristic Miner* und *Inductive Miner* generieren basierend auf dem aus den Aktivitätstypen und zugehörigen Aktivitätsinstanzen erzeugten Eventlog (siehe Abbildung 8-43 rechts) die Petri-Netze in Abbildung 8-44 bis Abbildung 8-46. Die grauen Transitionen sind *Hilfstransitionen*, die keinen Aktivitätstyp des Eventlogs abbilden.

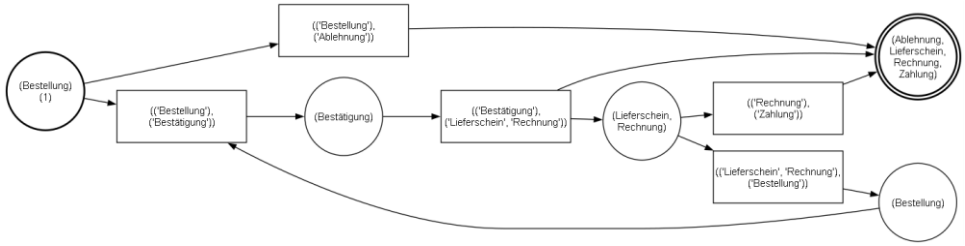


Abbildung 8-44: Aus den zeitlich-basierten Aktivitätstypen abgeleitetes Petri-Netz (Alpha Miner)

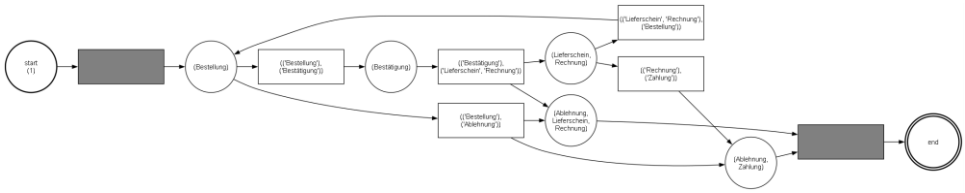


Abbildung 8-45: Aus den zeitlich-basierten Aktivitätstypen abgeleitetes Petri-Netz (Alpha Miner Plus)

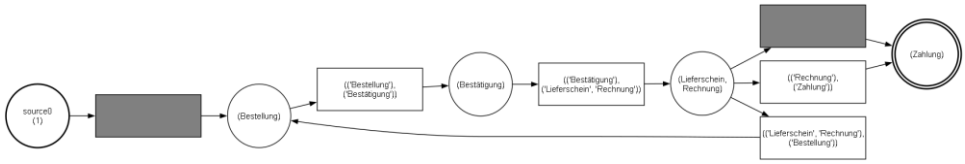


Abbildung 8-46: Aus den zeitlich-basierten Aktivitätstypen abgeleitetes Petri-Netz (Heuristic Miner)

Werden die Aktivitätstypen, die sich aus den zeitlichen Zusammenhängen ableiten lassen, basierend auf ihren Input- und Output-Objekttypen zusammengesetzt, ergibt sich das Petri-Netz in Abbildung 8-47.

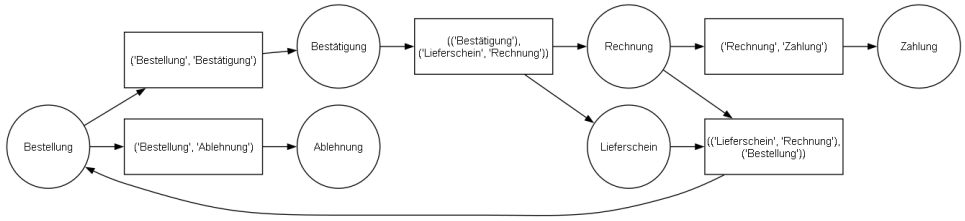


Abbildung 8-47: Aus den zeitlich-basierten Aktivitätstypen abgeleitetes Petri-Netz (zusammengesetzt)

Ergebnis der Evaluation: Die zeitlichen Zusammenhänge zwischen den Objekttypen wurden entsprechend der Objektinstanzen korrekt erkannt und die Ordnungsbeziehungen

zwischen den Aktivitätstypen als sequenziell und alternativ abgeleitet. Die verschiedenen Process Discovery-Algorithmen transformierten die extrahierten Informationen in unterschiedliche Petri-Netz-Darstellungen. Der Alpha Plus Miner repräsentierte den Zyklus korrekt und erkannte zusätzlich die alternativen Pfade zwischen Ablehnung und Bestätigung. Der Heuristic Miner erzeugte eine kompaktere Darstellung der zyklischen Struktur, bildete jedoch nicht die Ablehnung ab. Das Zusammengesetzte Petri-Netz bildet den Geschäftsprozess korrekt ab.

8.4.5.3 Prozessmodellgenerierung basierend auf den inhaltlichen Zusammenhängen

Basierend auf den inhaltlichen Zusammenhängen zwischen den Objektinstanzen wurden zunächst 45 Objektbeziehungstypen vom Software-Prototyp erkannt. Aus diesen ungerichteten Objektbeziehungstypen wurde anhand der Objektinstanzen der betreffenden Objekttypen bewertet, welcher Objekttyp Input oder Output eines Aktivitätstyps ist. Die Aktivitätstypen wurden mit den in Tabelle 8-5 angegebenen Gewichtungsfaktoren bewertet und absteigend sortiert. Die bewerteten Aktivitätstypen sowie die zusammengeführten Aktivitätstypen werden in Abbildung 8-48 gezeigt. Die Aktivitätstypen, die unterschiedliche Versionen der Objektinstanzen berücksichtigen, werden dabei zunächst separat analysiert und erst nach einer Überprüfung auf Widersprüche zu einem Aktivitätstyp zusammengeführt. Diese Überprüfung berücksichtigt die Reihenfolge der Objektinstanzen der Objekttypen, die Reihenfolge zu anderen Aktivitätstypen, die Objektinstanzen der Aktivitätsinstanzen der Aktivitätstypen und die Regeln der Aktivitätstypen.

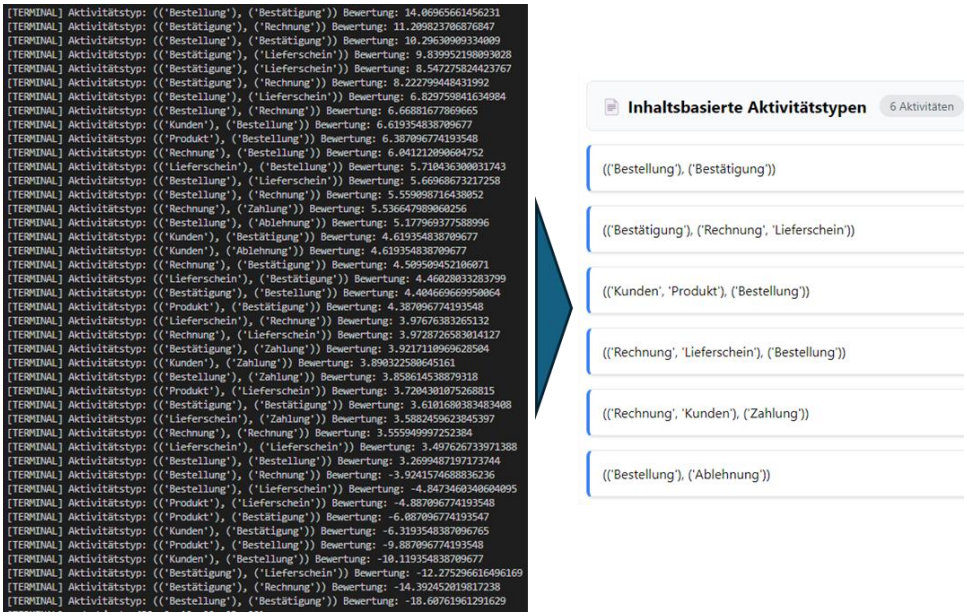


Abbildung 8-48 Bewertung der Aktivitätstypen (links, Terminal) sowie resultierende Aktivitätstypen (rechts)

Das resultierende Petri-Netz aus der Generierung, basierend auf den inhaltlichen Zusammenhängen, wird in Abbildung 8-49 dargestellt.

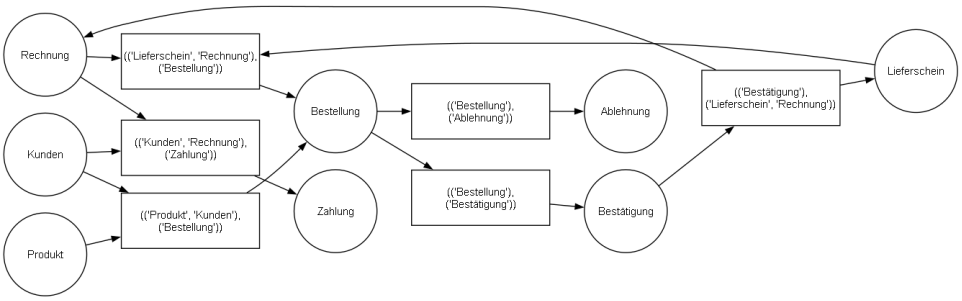


Abbildung 8-49: Aus den inhaltlich-basierten Aktivitätstypen abgeleitetes Petri-Netz

Ergebnis der Evaluation: Die inhaltlichen Zusammenhänge wurden korrekt erkannt. Die Evaluation zeigt, dass der Prototyp die mehrfache Ausführung von Aktivitätstypen und deren Zusammenhänge erkennt. Das resultierende Petri-Netz bildet sowohl die sequenziellen und alternativen Ordnungsbeziehungen der Aktivitätstypen korrekt ab und repräsentiert auch die mögliche Mehrfachausführung der ausgewählten Aktivitätstypen.

8.5 Zusammenfassung

Die Evaluation der Methode und des Software-Prototyps zeigt, ob und wie die Anforderungen aus Kapitel 5 umgesetzt wurden. Zusätzlich werden Einschätzungen über die Erreichung der Anforderungen gegeben, falls diese als nicht vollständig erfüllt bewertet wurden. Es wurde auch evaluiert, ob die gewünschten Kontrollflussmuster aus Abschnitt 2.2.1 extrahiert und entsprechend in einem Geschäftsprozessmodell repräsentiert werden. Die Evaluation zeigt, dass sequenzielle, alternative und nebenläufige Ordnungsbeziehungen für die gegebenen Testfälle erkannt wurden. Diese Testfälle erfüllen jedoch gewisse Bedingungen:

- Die Objekttypen weisen unterschiedliche Objektstrukturen auf, sodass deren Objektinstanzen korrekt geclustert werden können.
- Für die Objektinstanzen konnten die korrekten repräsentativen Zeitpunkte ermittelt werden. Durch die begrenzte Anzahl an Dateien (zwischen 16-50) treten die für das Erkennen der jeweiligen Kontrollflussmuster notwendigen Aspekte eindeutig auf. Das bedeutet beispielsweise, dass die Objektinstanzen bei der Überprüfung der sequenziellen Ordnungsbeziehung die daraus resultierende Reihenfolge konsequent einhalten.
- Die Prozessinstanzen weisen ähnliche zeitliche Abstände auf. Dadurch ermöglicht der normalisierte Abstand aussagekräftige Vergleiche.
- Die Objektinstanzen haben Attributüberschneidungen zu anderen Objektinstanzen. Nur durch diese können die Objektinstanzen einzelnen Prozessinstanzen zugeordnet und inhaltliche Zusammenhänge aufgedeckt werden.
- Die prozessinstanzunabhängigen Objekttypen (wie Kundendaten) werden früh zur Informationsbereitstellung genutzt und korrekt als prozessinstanzunabhängig erkannt.

Diese kontrollierten Bedingungen können zu Limitationen und Herausforderungen in realen Szenarien führen, insbesondere bei stark variierenden Prozesslaufzeiten, bei zu wenigen oder zu vielen Attributüberschneidungen sowie bei vielen Objekttypen mit ähnlicher Objektstruktur. Diesen Limitationen begegnet die Methode bereits mit implementierten Verbesserungen, wie beispielsweise eine iterative Vorgehensweise (siehe Abschnitt 7.3.1) und eine verbesserte Datenvorverarbeitung (siehe Abschnitt 7.3.2).

Die Evaluation zeigt für die bereitgestellten Testfälle außerdem, dass auch mehrfache Ausführungen von Aktivitätstypen erkannt wurden. Es wurde gezeigt, dass die Berücksichtigung inhaltlicher Zusammenhänge das Erkennen dieser Strukturen verbessern kann. Durch die Generierung basierend auf den Objektinstanzen und die Modellierung der Objekttypen als Stellen werden diese Kontrollflussmuster in der Generierung basierend auf den inhaltlichen Zusammenhängen durch weniger Elemente dargestellt als bei der

Generierung durch traditionelle Process Discovery-Algorithmen. Durch die inhaltlichen Zusammenhänge werden auch Zusammenhänge zwischen Aktivitätstypen erkannt, die in der Generierung durch Eventlogs vernachlässigt werden. Beispielsweise ist dadurch ein Zusammenhang zwischen dem Versenden und dem zuvor erstellten Lieferschein erkannt worden. Dadurch kann die Transition, die das Versenden repräsentiert, nur schalten, wenn zuvor der Lieferschein erstellt wurde. Im Vergleich hierzu ist die zweite vorwärtsverzweigte Stelle bei der Generierung durch Process Discovery-Algorithmen nicht abhängig von der ersten vorwärtsverzweigten Stelle. Die Ableitung der Aktivitätstypen mit korrekten Input- und Output-Objekttypen wird bei den zeitlichen Zusammenhängen nur auf die Reihenfolge der beteiligten Objekttypen bezogen. Bei der Berücksichtigung der inhaltlichen Zusammenhänge wird diese Ableitung basierend auf einem mehrstufigen Verfahren durchgeführt, das auch die zeitlichen Zusammenhänge einbezieht. Bei der Auswahl und Zusammenführung von Aktivitätstypen werden die übereinstimmenden Attribute und die Objektkategorien berücksichtigt. Außerdem werden auch die weiteren Objekttypen und -instanzen beachtet. Zusätzlich wird mit uneindeutigen Reihenfolgen von Objektinstanzen und Versionen von Objekten umgegangen.

Die konfigurierbaren Gewichtungsfaktoren bieten zusätzlich eine flexible Grundlage für die Anpassung an weitere Geschäftsprozesse. Zusätzlich wurde zu der evaluierten automatisierten Generierung wie in Abschnitt 7.3 auch eine iterative interaktionsbasierte Vorgehensweise implementiert, wodurch die Zwischenschritte der Generierung vom Anwender zunächst betrachtet und bei Bedarf verbessert werden können (siehe Abschnitt 7.3.1).

9 Fazit und Ausblick

Die *Modellierung von Geschäftsprozessen* dient in Unternehmen als Grundlage zur Dokumentation, Analyse, Verbesserung, Überwachung und Steuerung der Geschäftsprozesse [Wesk19]. Durch die Modellierung können die Geschäftsprozesse in Unternehmen kontinuierlich betrachtet sowie (bei Bedarf) angepasst und verbessert werden. Die Erstellung und Aktualisierung von Geschäftsprozessmodellen ist jedoch eine zeit- und ressourcenintensive Aufgabe, die sowohl Expertise in Modellierungssprachen wie BPMN oder Petri-Netzen als auch domänenspezifisches Wissen erfordert. Geschäftsprozessmodelle, die manuell oder (teil-)automatisch erstellt werden, können fehlerhaft sein. Bei der manuellen Modellierung sind Modellierer und Prozessexperte oftmals unterschiedliche Personen. Dies kann dazu führen, dass das Prozesswissen des Experten nicht korrekt in das Modell übertragen wird, etwa durch Missverständnisse in der Kommunikation oder Verzerrungen bei der Interpretation. Zusätzlich können mangelnde Expertise in der Prozessmodellierung oder unzureichendes Domänenwissen eine unzureichende Qualität der Geschäftsprozessmodelle bewirken.

Fehlerhafte Geschäftsprozessmodelle können bei der Verwendung verschiedene Probleme verursachen. Bei der Verwendung als Kommunikationsgrundlage können Missverständnisse zwischen verschiedenen Stakeholdern entstehen. Bei der Entscheidungsfindung besteht die Gefahr, dass auf Basis fehlerhafter Modelle falsche Schlussfolgerungen gezogen und Entscheidungen getroffen werden. In der Prozessanalyse werden durch fehlerhafte Modelle falsche Verbesserungspotenziale identifiziert. Bei der automatisierten Prozessausführung können die fehlerhaften Ausführungen beispielsweise zu Ressourcenverschwendung oder Compliance-Verstößen führen.

Existierende Ansätze zur automatisierten Prozessmodellgenerierung basieren primär auf Ereignisprotokollen (Eventlogs) aus IT-Systemen. Diese Beschränkung auf eine einzelne Datenquelle führt zu Informationslücken, da reale Geschäftsprozesse durch das komplexe Zusammenspiel zahlreicher Objekte und oft auch mehrerer Systeme charakterisiert sind. Prozessrelevante Informationen finden sich entsprechend nicht nur in Eventlogs, sondern auch in Dokumenten, E-Mails und anderen Prozessartefakten. Diese Artefakte enthalten wichtige Informationen über Objektstrukturen, deren Veränderungen und Beziehungen. Die systematische Nutzung dieser Informationen für die Prozessmodellierung wurde bisher jedoch kaum untersucht. Insbesondere fehlte ein Ansatz zur automatisierten Generierung von Geschäftsprozessmodellen basierend auf Objekten, der sowohl inhaltliche als auch zeitliche Zusammenhänge berücksichtigt. Ziel dieser Arbeit war es, diese Lücke durch die Entwicklung einer Methode zur objektbasierten, automatisierten Pro-

zessmodellierung in Form von Petri-Netzen zu schließen. Dazu werden die Objektinstanzen, die in den Prozessinstanzen des zu modellierenden Geschäftsprozesses erzeugt, gelesen, bearbeitet oder verbraucht wurden, analysiert. Diese Methode ermöglicht nicht nur die Abbildung des Kontrollflusses eines Geschäftsprozesses, sondern auch die Abbildung von Objektstrukturen und Operationen auf diese Objektstrukturen.

9.1 Zusammenfassung

Ausgehend von den Herausforderungen der Geschäftsprozessmodellierung wurde in **Kapitel 1** die automatisierte Prozessmodellierung als Möglichkeit identifiziert, das Problem zu lösen, dass viele Geschäftsprozesse in der Praxis nicht oder nur unzureichend modelliert werden [DLMR18, BBMB+11]. Zusätzlich zeigt eine Betrachtung der Daten zu Geschäftsprozessen, dass für die automatisierte Modellierung insbesondere die in den Geschäftsprozessen auftretenden Objekte (wie beispielsweise Bestellungen, Rechnungen, Lieferscheine) geeignet sein können, da diese sowohl Aktivitäts-, als auch Objektinformationen enthalten [ScAl24]. Die Nutzung dieser Objektinformationen für die Prozessmodellierung wurde bisher jedoch kaum untersucht. Bestehende Methoden zur automatisierten Prozessmodellgenerierung basieren primär auf Eventlogs und natürlich sprachlichen Texten. Daher war das Ziel der Arbeit eine Methode zur automatisierten Prozessmodellgenerierung im Kontext von Geschäftsprozessen zu entwickeln, die auf der Analyse von den Objektinstanzen, die in den Prozessinstanzen des Geschäftsprozesses erzeugt, gelesen, bearbeitet oder verbraucht wurden, basiert.

In **Kapitel 2** wurden für die vorliegende Arbeit die zentralen Begriffe *Modell*, *Geschäftsprozess* und *Geschäftsprozessmodellierung* definiert. In der allgemeinen Modelltheorie wird der *Modellbegriff* unter Verwendung von den drei Hauptmerkmalen Abbildungsmerkmal, Verkürzungsmerkmal und pragmatisches Merkmal beschrieben. Zusammenfassend ist ein Modell eine im Hinblick auf einen bestimmten Zweck reduzierte Abbildung eines Modelloriginals. Bei einem *Geschäftsprozess* werden eine Menge von Aktivitäten nach bestimmten Regeln auf ein bestimmtes Ziel hin ausgeführt. Wird die Modellierung im Kontext von Geschäftsprozessen betrachtet, bedarf es verschiedener Anforderungen an eine Modellierungssprache, um die relevanten Eigenschaften oder Verhaltensmuster des Modelloriginals, den *Geschäftsprozess*, entsprechend abbilden zu können. Daher wurden die Eigenschaften von Geschäftsprozessen betrachtet. Die bezüglich eines definierten Ziels relevanten Aspekte eines Geschäftsprozesses können unter Verwendung einer grafischen Modellierungssprache in einem Geschäftsprozessmodell abgebildet werden. Um in Geschäftsprozessmodellen unterschiedliche Aspekte zu fokussieren, können bei der Modellierung unterschiedliche Modellierungsperspektiven eingenommen werden. In der Geschäftsprozessmodellierung ist dabei insbesondere die kontrollfluss- und objekt(-fluss-)orientierte Modellierungsperspektive wichtig. Bei der kontrollflussori-

entierten Modellierungsperspektive steht die Reihenfolge der Aktivitäten im Vordergrund. Die Reihenfolge der Aktivitäten ergibt sich durch die naturgegebenen sowie von einer Organisation vorgegebenen Zusammenhänge zwischen den Aktivitäten. Bei der Fokussierung auf die Objekte eines Geschäftsprozesses können in einem Modell die Funktionen und/oder Daten zu den Objekten integriert werden. Ein wichtiger Aspekt bei der Modellierung und späteren Nutzung von Geschäftsprozessmodellen ist die Modellqualität. Daher wurden verschiedene Qualitätsaspekte für Geschäftsprozessmodelle analysiert und diskutiert. Die Qualität eines Geschäftsprozessmodells bezieht sich auf den Erfüllungsgrad von definierten Anforderungen bezüglich der Gesamtheit aller Eigenschaften des Geschäftsprozessmodells. Bei der Betrachtung der Qualität eines Geschäftsprozessmodells wird zwischen der *Betrachtung der Modellqualität* und *Betrachtung der Qualität des modellierten Geschäftsprozesses* unterschieden. Zur Betrachtung der Modellqualität wurden die Qualitätsaspekte nach den Anforderungen an ein Modell, an eine Modellierungssprache sowie an die Modellierung betrachtet.

Bei der Prozessmodellgenerierung wird aus bereitgestellten Daten ein Geschäftsprozessmodell erzeugt. In **Kapitel 3** wurden zunächst die Daten und das zu generierende Geschäftsprozessmodell betrachtet. Dafür ist zum einen relevant, wie diese Daten gespeichert werden (*Informationsträger)* und welche Informationen zu dem Geschäftsprozess durch diese bereitgestellt werden *(getragene Information)*. Informationsträger sind Speichermedien wie Datenbanken oder Dateien, die sich insbesondere in den unterstützten Datenformaten unterscheiden. Die Daten, die für die Geschäftsprozessmodellierung relevant sein können, wurden insbesondere anhand der Betrachtung von Geschäftsprozessen hergeleitet. Die Daten wurden bezüglich ihrer Datenstruktur, ihrer Datenqualität und ihres Informationsgehalts bewertet. Der Informationsgehalt wurde differenziert zwischen dem Informationsgehalt für die Geschäftsprozessmodellierung und dem für die Objektmodellierung, da ein Geschäftsprozessmodell generiert werden soll, das die Kontrollfluss- und Objektfluss-Modellierungsperspektive miteinander kombiniert. Durch die Bewertung der möglichen Daten wurde herausgestellt, dass insbesondere die Dateien, deren Rohdaten Objektinstanzen beschreiben, relevante Informationen für die Prozessmodellgenerierung bereitstellen und zur Abbildung weiterer relevanter Objekt-Aspekte in einem Geschäftsprozessmodell geeignet sind. Um die Informationen, die solche Dateien liefern, ebenfalls in den Geschäftsprozessmodellen abbilden zu können, wurden einige Modellierungssprachen im Hinblick auf ihre Fähigkeit zur Modellierung von Objektstrukturen, Objektzuständen, Objektbeziehungen und Objektflüssen betrachtet. Es gibt bereits einige Modellierungssprachen, die Objekte darstellen können, und es gibt dazu auch Ansätze, welche die kontrollflussorientierte Modellierungsperspektive mit der objektorientierten Modellierungsperspektive kombinieren. Jedoch gibt es insbesondere in der Ausdrucksmächtigkeit Unterschiede. Petri-Netze bieten eine formale Definition an. Des Weiteren können in höheren Petri-Netzen durch die unterscheidbaren Marken detaillierte Informationen zu Objekten beschrieben und durch die Transitionen auch verändert

werden. Dies ist bei anderen Modellierungssprachen nicht in vergleichbarer Granularität möglich. Somit können sowohl Objektstrukturen, Objektbeziehungen, Objektzustandsänderungen und Objektflüsse repräsentiert werden. Durch die unterscheidbaren Marken und die Beschreibung von Kontroll- und Objektfluss kann ein Petri-Netz sowohl einen Geschäftsprozesstyp als auch die Instanzen des Geschäftsprozesstyps abbilden. Daher werden in dieser Arbeit die Geschäftsprozessmodelle in Form eines Petri-Netzes generiert. Es werden Varianten höherer Petri-Netze betrachtet, die sich besonders zur Abbildung von Objekten eignen. Zu diesen Varianten zählen insbesondere die XML- und JSON-Netze, deren Marken XML-basierte bzw. in JSON-beschriebene Objekte abbilden können.

Nach der Untersuchung von Input und Output der Prozessmodellgenerierung, wurde in **Kapitel 4** die automatisierte Prozessmodellierung betrachtet. Dazu wurden 125 verwandte Arbeiten mittels einer Literaturrecherche identifiziert. Die Resultate unterscheiden sich in den verwendeten Input-Daten, den verwendeten Methoden zur Generierung und den unterstützten Modellierungssprachen. Auf der Grundlage der Input-Daten konnten sieben Gruppen identifiziert werden: Software-Code, Tabellen, Regeln, Modelle, Wissensdatenbanken, Eventlogs und natürlichsprachlicher Text. Für diese Gruppen wurden die unterschiedlichen Methoden zur Prozessmodellgenerierung beschrieben. Die Literaturanalyse deckt eine bedeutende Forschungslücke auf: Während Prozessinstanzobjekte wie Bestellungen, Lieferscheine oder Rechnungen wesentliche Prozessinformationen enthalten (siehe Kapitel 3), existieren bisher keine Ansätze, die diese für die Prozessmodellgenerierung nutzen. Bestehende Methoden basieren stattdessen primär auf Eventlogs, Prozessbeschreibungen oder Regelwerken, was zu einer Informationslücke führt, da die inhaltlichen Zusammenhänge zwischen Prozessobjekten nicht berücksichtigt werden. Basierend auf den bestehenden Ansätzen wurden die Möglichkeiten zur Prozessmodellgenerierung in Bezug zu dieser Arbeit erarbeitet und die Herausforderungen der Prozessmodellgenerierung durch die gefundenen Ansätze aufgezeigt. Die identifizierten Herausforderungen betreffen insbesondere die Qualität der Input-Daten, die Komplexität der Generierungsalgorithmen und die Qualität der resultierenden Modelle. Aus der Analyse bestehender Lösungsansätze wurden mögliche Ansätze für die neu zu entwickelnde Methode abgeleitet, etwa zur Vorverarbeitung von Objektinstanzen, zur Erkennung von Objektbeziehungen und zur Qualitätssicherung der generierten Modelle. Diese Erkenntnisse bilden die Grundlage für die Entwicklung einer neuen Methode zur automatisierten, objektbasierten Prozessmodellierung. Durch die Berücksichtigung der Möglichkeiten und Herausforderungen der betrachteten vergleichbaren Ansätze zur Prozessmodellgenerierung soll sichergestellt werden, dass die in dieser Arbeit entwickelte Methode diesbezüglich einen Mehrwert für die Prozessmodellgenerierung mit sich bringt.

Basierend auf den vorangegangenen Kapiteln sowie der Zielsetzung der Arbeit wurden in **Kapitel 5** Anforderungen entwickelt, mit denen die Methode zur Prozessmodellgenerie-

rung ausgearbeitet wurde. Zur systematischen Spezifizierung der Anforderungen wurde der Anwendungsfall der Prozessmodellgenerierung betrachtet. Dieser Anwendungsfall besteht insbesondere aus der Datenvorverarbeitung, der Informationsextraktion und der Transformation. Zur Anforderungsspezifikation wurden die einzelnen Schritte des Anwendungsfalls durch die Anforderungen an den Input, den Output und die Prozessmodellgenerierung aufgegriffen. Die Anforderungen an den *Input* (die Daten, die zur Prozessmodellgenerierung verwendet werden sollen) wurden anhand der Betrachtung der Zweckeignung der Daten und Informationen zur Geschäftsprozessmodellierung (siehe Abschnitt 3.1) spezifiziert. Daher wurden bei der Spezifizierung der Anforderungen an die Input-Daten die Aspekte *Datenformat*, *Datenqualität* und *Informationsgehalt* betrachtet. Die Anforderungen an den *Output* (das zu generierende Modell) wurden anhand der Anforderungen an ein Geschäftsprozessmodell spezifiziert. Diese Anforderungen sind Anforderungen an das *Modell*, die *Modellierungssprache* sowie die *Modellierung*. Dazu wurden zusätzlich Beschriftungs- und Inschriftenkonventionen definiert, um durch diese Konventionen die Anforderungen an die Verständlichkeit und Einheitlichkeit zu unterstützten. Bei den Anforderungen an die *Generierung* wurde insbesondere die *funktionale Eignung* der einzelnen Schritte des Anwendungsfalls betrachtet.

Die entwickelte Methode adressiert insbesondere die Generierung des Geschäftsprozessmodells, da die Erfüllung der Anforderungen an den Input durch entsprechende Vorverarbeitungen und die Erfüllung der Anforderungen an den Output durch entsprechende nachgelagerte Überarbeitungen weiter verbessert werden können. Der Input für die Generierung sind Dateien, die den Zustand einer oder mehrerer Objektinstanz(en) eines Geschäftsprozesses zu einem bestimmten Zeitpunkt beschreiben. Diese beschriebenen Objektinstanzen sind demnach in den Ausführungen des Geschäftsprozesstyps (den Prozessinstanzen) aufgetreten. Basierend auf diesen Objektinstanzen der Prozessinstanzen wird der Geschäftsprozesstyp abgeleitet und in einem Geschäftsprozessmodell in Form eines Petri-Netzes repräsentiert. Diese Ableitung basiert darauf, dass durch die Aktivitäten eines Geschäftsprozesses der Zustand einer Prozessinstanz verändert wird. Sofern auch die Eigenschaften der Objektinstanzen verändert werden, lässt sich die Durchführung der Aktivität und die Änderung des Zustandes anhand der beteiligten Objektinstanzen beobachten. Daher können durch die Betrachtung der Objektinstanzen Rückschlüsse auf die Aktivitätstypen und deren Ordnungsbeziehungen gezogen und anschließend in einem Geschäftsprozessmodell repräsentiert werden.

Die in **Kapitel 6** beschriebene Methode zur Generierung von höheren Petri-Netzen aus Objektinstanzen folgt einem zweistufigen Vorgehen: *Informationsextraktion* und *Transformation*. Um aus den bereitgestellten Dateien ein Geschäftsprozessmodell generieren zu können, werden zunächst die relevanten *Informationen* für den Geschäftsprozesstyp aus den Rohdaten der Dateien zu den Objektinstanzen *extrahiert*. Dazu werden die Dateien zunächst Prozessinstanzen zugeordnet, anschließend werden Objekttypen klassi-

fiziert und schließlich aus den inhaltlichen und zeitlichen Zusammenhängen der Objektinstanzen Aktivitätstypen abgeleitet. Die extrahierten Informationen zu Objekttypen und Aktivitätstypen werden in der *Transformation* in die Elemente eines höheren Petri-Netzes transformiert. Die Objekttypen werden dabei durch die Stellen, die Aktivitätstypen durch die Transitionen und der Kontrollfluss durch die Kanten eines Petri-Netzes dargestellt. Zusätzlich werden die Datenstrukturen der Objekttypen und die Manipulationen bzw. Regeln der Aktivitätstypen in den generierten Petri-Netzen abgebildet. Dabei werden zuerst die Aktivitätstypen aus den zeitlichen Zusammenhängen abgeleitet. Anschließend werden zusätzlich die Aktivitätstypen aus den inhaltlichen Zusammenhängen berücksichtigt und dadurch auch Manipulationsregeln im Geschäftsprozessmodell definiert. Danach werden weitere Ausführungsbedingungen integriert.

Die Methode zur automatisierten Generierung von Geschäftsprozessmodellen aus Objektinstanzen wurde in einem Software-Prototyp implementiert (siehe **Kapitel 7**). Der Software-Prototyp setzt die Schritte der Methode zur Prozessmodellgenerierung aus Kapitel 6 automatisiert um:

- In der *Informationsextraktion* werden die Objekttypen, Objektbeziehungstypen und Aktivitätstypen aus den Objektinstanzen abgeleitet.
- In der *Transformation* werden die extrahierten Informationen in einem Geschäftsprozessmodell in Form eines Petri-Netz repräsentiert.

Die modulare Architektur des Software-Prototyps ermöglicht Erweiterungen für zusätzliche Funktionalitäten. Dies wurde gezeigt, indem der Prototyp bereits um Funktionalitäten zur Verarbeitung verschiedener Dateiformate erweitert wurde. Die Erweiterungen umfassen die Implementierung von OCR und VDU für nicht-maschinenlesbare Formate sowie Mechanismen zur Strukturierung und Bereinigung der extrahierten Daten. Diese Erweiterungen erhöhen die Anwendbarkeit des Software-Prototyps durch Unterstützung unterschiedlicher Eingabeformate bei gleichzeitiger Sicherstellung einer einheitlichen Datenqualität für die Prozessmodellgenerierung.

Die Evaluation der Methode anhand des Software-Prototyps in **Kapitel 8** zeigt, ob und wie die Anforderungen aus Kapitel 5 umgesetzt wurden. Zusätzlich wurde eine Einschätzung über den Erfüllungsgrad der Anforderungen gegeben, falls diese als nicht vollständig erfüllt bewertet wurden. Dazu wird auch evaluiert, ob die gewünschten Kontrollflussmuster extrahiert und adäquat in einem Geschäftsprozessmodell repräsentiert werden. Die Evaluation zeigt, dass sequenzielle, alternative und nebenläufige Ordnungsbeziehungen erkannt werden. Zusätzlich werden auch mehrfache Ausführungen von Aktivitätstypen erkannt. Insbesondere hier wird gezeigt, dass die Berücksichtigung inhaltlicher Zusammenhänge das Erkennen dieser Strukturen verbessern kann. Durch die Generierung basierend auf den Objekten und die Modellierung der Objekttypen als

Stellen werden diese Kontrollflussmuster in der Generierung basierend auf den inhaltlichen und zeitlichen Zusammenhängen durch weniger Elemente dargestellt als bei der Generierung des Modells durch Process Discovery-Algorithmen. Durch die Analyse inhaltlicher Zusammenhänge werden Abhängigkeiten zwischen Aktivitätstypen erkannt, die bei einer reinen Eventlog-basierten Generierung vernachlässigt werden. Beispielsweise wird in einem Versandprozess durch die Analyse inhaltlicher Zusammenhänge erkannt, dass das Versenden eines Pakets vom zuvor erstellten Lieferschein abhängt. Diese Abhängigkeit wird im generierten Modell dadurch abgebildet, dass die Transition, die das Versenden repräsentier, nur schalten kann, wenn zuvor der Lieferschein erstellt wurde. Durch die Process Discovery-Algorithmen, die nur auf den generierten Eventlogs basieren, wird lediglich die beobachtete zeitliche Folge der abgeleiteten Aktivitätstypen repräsentiert, wodurch wichtige Zusammenhänge nicht repräsentiert werden.

Die Zuordnung von Input- und Output-Objekttypen zu Aktivitätstypen erfolgt bei den zeitlichen Zusammenhängen basierend auf der beobachteten Reihenfolge. Bei den inhaltlichen Zusammenhängen wird ein mehrstufiges Analyseverfahren angewandt. Dieses berücksichtigt neben den zeitlichen Zusammenhängen der Objekte auch übereinstimmende Attribute zwischen Objekten, deren Kategorisierung und die Objektinstanzen der beteiligten Objekttypen. Zusätzlich werden auch Objekttypen und -instanzen anderer Aktivitätstypen berücksichtigt. Das Verfahren kann dabei auch mit uneindeutigen Reihenfolgen und verschiedenen Objektversionen umgehen.

Die implementierten konfigurierbaren Gewichtungsfaktoren bieten zusätzlich eine Grundlage für die flexible Anpassung an weitere Geschäftsprozesse. Neben der evaluierten automatisierten Generierung wurde auch eine iterative interaktionsbasierte Vorgehensweise implementiert, wodurch die Zwischenschritte der Generierung zunächst betrachtet und bei Bedarf verbessert werden können.

9.2 Beitrag und Grenzen der Arbeit

Bei der Betrachtung von Geschäftsprozessen und der umfangreichen Analyse der automatisierten Prozessmodellgenerierung wurde gezeigt, dass aus den Objektinstanzen, die in den Prozessinstanzen eines Geschäftsprozesses erzeugt, gelesen, bearbeitet oder verbraucht wurden, relevantes Prozesswissen abgeleitet werden kann. Durch die Betrachtung der Objektinstanzen können Objekttypen, die an einem Geschäftsprozess beteiligt sind, definiert werden. Zusätzlich können durch die Zustandsänderungen der Objektinstanzen Aktivitätstypen identifiziert werden. Dennoch werden Objektinstanzen bisher nur wenig als Informationsgrundlage zur Modellierung genutzt. An dieser Stelle leistet diese Arbeit einen Beitrag für die Verwendung von Objektinstanzen zur Prozessmodellgenerierung.

In diesem Zusammenhang beschäftigt sich die Arbeit auch mit der Herausforderung, dass die Objektinstanzen möglicherweise nicht explizit einzelnen Prozessinstanzen zugeordnet sind. Das heißt, dass nicht angegeben ist, welche Objektinstanz in welcher Prozessinstanz erzeugt, gelesen, bearbeitet oder verbraucht wurde: beispielsweise ist nicht angegeben, dass die *Bestellung 1* zur *Rechnung 1* geführt hat. Es wurde in dieser Arbeit dazu ein Vorgehen entwickelt, wie Objektinstanzen Prozessinstanzen zugeordnet werden können. Dieses Vorgehen berücksichtigt, dass einzelne Objektinstanzen auch mehreren Prozessinstanzen zugeordnet werden können. Beispielsweise kann ein Kunde mehrere Bestellungen durchführen, die in unterschiedlichen Prozessinstanzen bearbeitet werden. Dafür wird zunächst analysiert, welche Objekttypen prozessinstanzunabhängig sind, um diese bei der Prozessinstanz-Zuordnung auszuschließen. Dies ist notwendig, da prozessinstanzunabhängige Objekte zu fehlerhaften Zusammenführungen von Prozessinstanzen führen würden. Beispielsweise würden dadurch verschiedene Bestellungen eines Kunden zu einer Prozessinstanz zugeordnet werden. Bei vergleichbaren Ansätzen werden diese vorab manuell ausgeschlossen [EAZP+12]. Vergleichbare Arbeiten zu Eventlogs verwenden zur Prozessinstanz-Zuordnung zeitliche Eigenschaften [FBMM+24, DKŽA20]. Dies führt bei prozessinstanzunabhängigen Objekten zu fehlerhafter und nur einmaliger Zuordnung, da prozessinstanzunabhängige Objekte keinen eindeutigen repräsentativen Zeitpunkt für eine Prozessinstanz haben. Diese Einschränkungen führen zu Informationslücken und zu einer verzerrten Prozesssicht. Diese Probleme bei der Zuordnung von Objekten zu Prozessinstanzen werden durch die vorliegende Arbeit adressiert.

Die entwickelte Generierungsmethode ermöglicht die Extraktion von Prozesswissen aus den Objektzuständen und deren Veränderungen sowie die Transformation von diesem Prozesswissen in ein Geschäftsprozessmodell. In der Informationsextraktion werden sowohl inhaltliche als auch zeitliche Zusammenhänge zwischen Objektinstanzen analysiert. Im Gegensatz zum Process Mining, das die Ereignisfolgen betrachtet, wird nicht nur die Reihenfolge der Objekte (bzw. der daraus abgeleiteten Aktivitäten) in die Generierung einbezogen, sondern es werden auch Objektstrukturen und deren Veränderungen analysiert. Dies ermöglicht die Erkennung von Objekttypen, deren Objektstrukturen sowie direkten und indirekten Objektbeziehungstypen. Darauf aufbauend werden Aktivitätstypen mit Input/Output-Beziehungen und Ausführungsbedingungen basierend auf Objektzuständen abgeleitet. Dadurch erhalten die Aktivitätstypen nicht nur Informationen zu den Ordnungsbeziehungen zu anderen Aktivitätstypen, sondern auch Informationen dazu, wie die Aktivitäten Objekte erzeugen oder verändern. Die zweistufige Generierungsmethode kombiniert dafür, soweit nach [ScAl24] bekannt ist, erstmals die Analyse inhaltlicher und zeitlicher Zusammenhänge zwischen Objektinstanzen. Die Evaluation zeigt, dass diese Methode Prozessstrukturen wie Schleifen und Nebenläufigkeit erkennen kann, bei denen Process Discovery-Algorithmen oftmals Schwierigkeiten haben [ACDL+19, WBVB12]. Insbesondere die Ableitung von Aktivitätstypen aus Objektinformationen wurde bisher in dieser Form nicht ermöglicht. Somit können auch Aktivitä-

ten erkannt werden, die nicht in Systemen aufgezeichnet wurden. Auch hier zeigt die Arbeit, dass im Vergleich zu gängigen Process Discovery-Algorithmen durch den Einbezug der inhaltlichen Überschneidungen nicht jede Ausführungsmöglichkeit in den Daten repräsentiert sein muss [AaWe04], da Zusammenhänge nicht nur zeitlich sondern auch inhaltlich aufgedeckt werden.

Die Bewertung zur Auswahl der relevanten Aktivitätstypen berücksichtigt verschiedene Aspekte zur strukturellen Position, der beteiligten Objekttypen und den Eigenschaften des inhaltlichen Zusammenhangs. Dies ermöglicht eine nachvollziehbare Ableitung der Aktivitätstypen. Durch die konfigurierbaren Gewichtungsfaktoren ist es möglich, sowohl ein Ist-Modell als auch mögliche Soll-Modelle zu erstellen:

- Ein Ist-Modell wird durch die stärkere Gewichtung der zeitlichen Eigenschaften erreicht. Die repräsentativen Zeitpunkte der Objektinstanzen zeigen die Reihenfolge der Objektinstanzen wie diese in den vergangenen Prozessinstanzen erzeugt, gelesen, bearbeitet oder verbraucht wurden.
- Mögliche Soll-Modelle könnten durch die stärkere Gewichtung inhaltlicher sowie schwächere Gewichtung der zeitlichen Eigenschaften erreicht werden. Dadurch könnte beispielsweise eine sinnvollere Reihenfolge erreicht werden, wenn es um die Übergabe von Attribut-Wert-Paaren geht.

Zusätzlich wird betrachtet, ob die abgeleiteten Aktivitätstypen zusammengeführt werden können. Diese Überprüfung basiert sowohl auf den beteiligten Objekttypen als auch auf den übereinstimmenden Attribut-Wert-Paaren. Somit basiert das Erkennen von Aktivitäts-Duplikaten im Vergleich zu Process Discovery-Algorithmen nicht nur auf der Bezeichnung der Aktivitäten. Neben der Erkennung und Zusammenführung von Duplikaten werden auch Aktivitäten analysiert, die ähnliche Input- oder Output-Objekte verwenden. Eine Zusammenführung erfolgt jedoch nur dann, wenn sie mit den beobachteten Objektinstanzen der jeweiligen Aktivitäten vereinbar ist. Würde eine Zusammenführung zu Widersprüchen bei den verarbeiteten Objektinstanzen führen, wird sie nicht durchgeführt. Darüber hinaus ermöglicht der Software-Prototyp auch eine Anpassung der initial erkannten Objekttypen. Das ursprüngliche Clustering-Ergebnis, das die Dateien verschiedenen Objekttypen zuordnet, kann dabei manuell angepasst werden, um beispielsweise Objekttypen *zusammenzuführen*.

Die anschließende Transformation integriert die extrahierten Informationen in einem formalen Modell durch die Verwendung höherer Petri-Netze. Die formale Semantik der Petri-Netze ermöglicht nicht nur eine Repräsentation der Ordnungsbeziehungen zwischen Aktivitätstypen sowie der Operationen auf Objektstrukturen, sondern auch die Integration von Ausführungsregeln und die Verifikation von Modelleigenschaften. Die unterscheid-

baren Marken erlauben die explizite Repräsentation von Objektinstanzen und von deren Zuständen und Zustandsübergängen.

Durch den Fokus auf Objekte als zentrale Prozessartefakte zur Prozessmodellgenerierung und durch die Integration von Kontrollfluss- und Objektaspekten eignen sich die generierten Modelle für die Prozessanalyse und -dokumentation und bilden auch eine Grundlage für Systemspezifikationen und Prozessautomatisierungen. Durch die Generierung der Modelle basierend auf realen Daten zu den Objektinstanzen der Prozessinstanzen, werden auch die Objektstrukturen und -manipulationen formal spezifiziert. Dies bestärkt die Nutzung für die Konfiguration von IT-Systemen basierend auf den extrahierten Objektstrukturen und -beziehungstypen [LBWM+20]. Die formale Integration von Objekt- und Verhaltensaspekten ermöglicht zudem die Verifikation von Prozesseigenschaften.

Die prototypische Implementierung demonstriert die Umsetzbarkeit und die praktische Anwendbarkeit der Methode. Durch die modulare, erweiterbare Architektur können zusätzliche Dateiformate, Analysemethoden und Transformationsregeln flexibel integriert werden. Die konfigurierbaren Gewichtungsfaktoren ermöglichen eine Anpassung an spezifische Anwendungskontexte. Die Integration von Process Mining-Algorithmen ermöglicht einen Vergleich zu bestehenden Ansätzen. Zusätzlich zeigt die Evaluation Vorteile gegenüber traditionellen Process Discovery-Algorithmen auf. Die entwickelte Methode stellt Prozessstrukturen wie Mehrfachausführungen einer/mehrerer Aktivität(en) durch die objektorientierte Darstellung kompakter dar. Beispielsweise werden Objekttypen nicht mehrfach repräsentiert und weniger zusätzliche Transitionen zur Abbildung des gewünschten Verhaltens verwendet. Zusätzlich werden Zusammenhänge zwischen nicht unmittelbar aufeinanderfolgende Aktivitäten erkannt.

Den originellen Beiträgen der vorliegenden Arbeit stehen verschiedene Grenzen gegenüber. Eine Einschränkung betrifft die Analyse der Objektbeziehungstypen. Die entwickelte Methode fokussiert primär direkte Attribut-Wert-Übereinstimmungen zwischen Objektinstanzen. Komplexere semantische Beziehungen oder indirekte Abhängigkeiten zwischen Objekten werden nur teilweise erfasst (beispielsweise durch die Betrachtung der zeitlichen Zusammenhänge). Bisher werden auch nur einfache Beziehungen aufgedeckt. Dadurch wird zum Beispiel nicht erkannt, wenn ein Wert sich aus der Summe mehrere Werte ergibt. Durch die Ergänzung weiterer Analysen könnten weitere Zusammenhänge und Regeln aufgedeckt werden.

Zudem basiert die Bewertung der Beziehungsrelevanz auf konfigurierbaren Gewichtungswerten, für deren Bestimmung bislang keine empirisch fundierten Richtlinien existieren. Die voreingestellten Werte sind daher subjektiv ausgewählt und wurden anhand der Testdaten aus der Evaluation bestimmt. Jedoch könnten für andere Anwen-

dungsbeispiele andere Werte sinnvoll sein. Die Konfigurationsmöglichkeit dieser Gewichtungswerte ermöglicht eine Anpassung, jedoch fehlen beispielsweise noch Best Practice Erfahrungen zur Einstellung dieser Werte.

In technischer Hinsicht steht für den entwickelten Prototyp noch die systematische Evaluation verschiedener Eigenschaften aus. Dies betrifft insbesondere die Leistungsfähigkeit bei der Verarbeitung großer Datenmengen sowie die Handhabung komplexer Objektstrukturen. Durch die rekursiven Vergleiche und die Verschachtelung von Attribut-Wert-Paaren skaliert der Software-Prototyp nicht linear mit der Anzahl der Objektinstanzen und deren Attributen. Auch existieren Einschränkungen in der iterativen Vorgehensweise sowie der Unterstützung der Benutzenden. Die Integration zusätzlicher Informationen in das Petri-Netz ist noch unvollständig implementiert. Während der Kontrollfluss und grundlegende Objektstrukturen bereits abgebildet werden, fehlt noch die vollständige Integration der Objektmanipulationen durch Transitionen sowie die Abbildung komplexer Geschäftsregeln und Ausführungsbedingungen in den Transitionsinschriften. Diese werden bisher nur separat dargestellt. Die aktuelle Implementierung verarbeitet hauptsächlich semi-strukturierte Datenformate wie JSON oder XML. Eine Erweiterung um weitere Datenquellen wie unstrukturierte Textdokumente wäre wünschenswert, um die praktische Anwendbarkeit zu erhöhen. Dies würde zusätzliche Vorverarbeitungsschritte und Analysemethoden erfordern, um die Informationen zu den Objektinstanzen aus diesen Datenformaten als Attribut-Wert-Paare zu extrahieren.

Die Evaluation beschränkt sich auf die Überprüfung der Basis-Kontrollflussmuster wie Sequenz, Alternative, Nebenläufigkeit, Schleife und Zyklus. Weitere Muster nach [AHKB03] wie Multi-Choice oder Abbruchsmuster wurden nicht evaluiert. Zusätzlich werden bei der Generierung Integritätsbedingungen zwischen Objekttypen nicht berücksichtigt. Die Gewichtung von Kanten, die die Anzahl der in einer Aktivität beteiligten Objekte repräsentiert, wird bisher nur teilweise berücksichtigt. Dies lässt sich am Beispiel einer Bestellungsabwicklung verdeutlichen: Eine Bestellung steht in einer mengenmäßigen Beziehung zu ihren Bestellpositionen, wobei eine Bestellung mindestens eine oder mehrere Bestellpositionen umfassen muss. Solche 1:n oder n:m Beziehungstypen zwischen den Objekttypen werden in der aktuellen Implementierung bei der Analyse und Prozessmodellgenerierung noch nicht vollständig berücksichtigt. Die Evaluation der generierten Modelle erfolgte anhand künstlich erzeugter Testdaten. Eine umfassende Validierung in verschiedenen Anwendungsdomänen mit unterschiedlichen Prozesscharakteristiken wurde nicht durchgeführt. Auch fehlen detailliertere Vergleichsstudien mit anderen Modellierungsansätzen mit realen Daten.

9.3 Ausblick

Mit dieser Arbeit wurde eine Methode zur objektbasierten Geschäftsprozessmodellierung bereitgestellt. Der Software-Prototyp demonstriert die software-technische Umsetzbarkeit der Methode und ermöglicht eine Evaluation der Anforderungen. Sowohl die Methode als auch der Software-Prototyp stehen für weiterführende Forschungsarbeiten und für die Anwendung in der Praxis zur Verfügung stehen.

Die aufgezeigten Grenzen der Arbeit bieten wichtige Ansatzpunkte für die Weiterentwicklung der objektbasierten Prozessmodellierung. Diese lassen sich in künftige Forschungsarbeiten zur Methode und in Erweiterungen des Software-Prototyps unterteilen. Die Betrachtung künftiger Forschungsarbeiten erfolgt für die Methode und den Software-Prototyp nach den einzelnen Schritten der Generierung sowie der anschließenden Weiterverwendung der generierten Modelle.

Die Methode zur Prozessmodellgenerierung basiert unter anderem auf dem inhaltlichen Vergleich der Objektinstanzen. Dieser Vergleich kann um weitere semantische Analysetechniken erweitert werden. Aktuelle Entwicklungen im Bereich der generativen Transformermodelle und des maschinellen Lernens bieten vielversprechende Ansätze zur Erkennung komplexer Objektbeziehungen [EHSR+22, IMDD+23]. Dazu könnten auch quantifizierte Objektbeziehungstypen sowie die Prüfung von Integritätsbedingungen für die Transitionen bezüglich der Objekte ergänzt werden [HSVW09, PGEM+20]. Die Kombination mit Domänenontologien oder Referenzmodellen könnte zudem die semantische Fundierung der Objektklassifikation sowie die Ableitung von Objektbeziehungstypen beziehungsweise Aktivitäten verbessern [SiAC13, KüKR06].

Die konfigurierbaren Gewichtungsfaktoren für die Aktivitätstyp-Bewertung beeinflussen die Auswahl der relevanten Aktivitätstypen und bestimmen somit, welche Aktivitätstypen im Geschäftsprozessmodell repräsentiert werden. Für die empirische Validierung der Gewichtungsfaktoren müssen systematische Vergleichsstudien in verschiedenen Anwendungskontexten durchgeführt werden [Mend08]. Zusätzlich könnte die Integration von weiteren Feedback-Mechanismen und die automatische Anpassung der Gewichtungsfaktoren die Methode weiter verbessern. Der bestehende Prototyp bietet durch seine modulare Architektur und die konfigurierbaren Gewichtungsfaktoren eine solide Basis für diese Weiterentwicklungen. Die Methode bietet weiterhin das Potential, weitere Kontrollflussmuster, Objektstrukturen und Ausführungsbedingungen zu generieren. Die Überprüfung und Evaluation dieser Generierung steht jedoch noch aus.

Um die Qualität der generierten Geschäftsprozessmodelle zusätzlich zu verbessern, bieten sich weitere Ansätze an. Zum einen könnte ein Human-in-the-Loop-Ansatz (HITL-Ansatz) domänenspezifische Fachwissens der Prozessexperten systematisch in die

Prozessmodellgenerierung einbinden [MHAB+23]. Dies würde interaktives Feedback während der Objektklassifikation, der Aktivitätstyp-Ableitung und der Transformation ermöglichen. Der HITL-Ansatz wäre besonders wertvoll für die Validierung komplexer Objektbeziehungen und die Feinjustierung der Konfigurationsparameter [MHAB+23]. Die Herausforderung liegt dabei in der Gestaltung effektiver Interaktionspunkte und der nahtlosen Integration des menschlichen Feedbacks in den automatisierten Generierungsprozess. Zum anderen könnten Selbstreflexionsmechanismen [MTGH+23] implementiert werden, die eine iterative Verbesserung der generierten Modelle ermöglichen. Durch Überprüfung der generierten Modelle mit weiteren formalen Qualitätskriterien könnten potenzielle Verbesserungen automatisch identifiziert werden. Die resultierenden Anpassungsvorschläge könnten anschließend entweder automatisch umgesetzt oder dem Benutzer zur Validierung vorgelegt werden [MTGH+23].

Das auswählbare iterative Vorgehen des Software-Prototyps erlaubt bereits unmittelbare Anpassungen und Verbesserungen während der Prozessmodellgenerierung. Es können die Gewichtungsfaktoren, die Clusterzuordnung sowie die Auswahl der prozessinstanzunabhängigen Objekte und die Benennungen der Objekttypen und Aktivitätstypen angepasst werden. Dies sollte jedoch noch weitergehend unterstützt werden. Dazu könnte beispielsweise die Anpassbarkeit der generierten Modelle ermöglicht werden, indem die Integration in einem Modellierungswerkzeug bereitgestellt wird. Zusätzlich sollte auch die manuelle Auswahl der relevanten Objektbeziehungstypen unterstützt und gegebenenfalls auch die Auswahl zusammenzuführender Aktivitätstypen ermöglicht werden. Dadurch könnte der Software-Prototyp nicht nur zur automatisierten Modellierung genutzt werden, sondern auch zur Modellierungsunterstützung. Die Standardisierung von Schnittstellen und Austauschformaten könnte zudem die Integration in bestehende BPM-Landschaften erleichtern oder unmittelbare Datenübertragungen ermöglichen. Auch die umfassendere Behandlung unstrukturierter Daten durch Natural Language Processing würde den Anwendungsbereich erweitern [HaKa16, EHSR+22, FBZN+24].

Eine technische Weiterentwicklung sollte sich auch auf die Skalierbarkeit und Benutzungsfreundlichkeit des Systems konzentrieren. Verteilte Algorithmen und inkrementelle Analysetechniken könnten die Performance bei großen Datenmengen verbessern. Die Entwicklung oder Einbindung spezialisierter Visualisierungs- und Konfigurationswerkzeuge würde die praktische Anwendbarkeit erhöhen. Dazu sollte ebenfalls die Darstellung der zusätzlichen Informationen zu Aktivitäts- und Objekttypen in das Petri-Netz betrachtet werden. Ansätze zur benutzungsfreundlicheren Darstellung von JSON-Netzen existieren bereits in [Frit25].

Die Überprüfung von Prozesseigenschaften anhand eines Modells wurde in dieser Arbeit nicht betrachtet. Jedoch könnte dies insbesondere unter der Berücksichtigung der Objektperspektive wertvolle zusätzliche Informationen liefern [RiMa21]. Die formale Fundie-

rung erfordert zunächst die präzise Definition des Schaltverhaltens höherer Petri-Netze, um Eigenschaften wie Deadlock-Freiheit oder die Erreichbarkeit bestimmter Markierungen [Reis13] untersuchen zu können. [BaHM15] untersucht formale Eigenschaften von Petri-Netzen mit semistrukturierten Daten. Darauf aufbauend könnte betrachtet werden, ob eine zuverlässige Ausführung eines Prozessmodells gewährleistet werden kann, oder gezeigt werden, dass unerwünschte Zustände niemals auftreten können [Kind94]. Für die Anwendung sind Simulationsalgorithmen erforderlich, die die spezifischen Herausforderungen objektbasierter Geschäftsprozessmodelle adressieren. Diese Simulationsalgorithmen könnten auch den nebenläufigen Zugriff auf Objektstrukturen [Ober96] und die effiziente Ermittlung konfliktfreier Markierungen berücksichtigen. Die Integration der vorhandenen Objekte in Geschäftsprozessen als Grundlage für die Generierung von Simulationsszenarien würde dabei realitätsnahe Analysen ermöglichen.

Durch die Analyse der zeitlichen und inhaltlichen Zusammenhänge sowie die konfigurierbaren Gewichtungsfaktoren können sowohl bestehende Geschäftsprozesse (Ist-Zustand) als auch mögliche Varianten (Soll-Zustand) generiert werden (siehe auch Abschnitt 9.2). Durch den Vergleich der Soll-Modelle mit Referenzmodellen könnten Vorschläge zur Prozessverbesserung abgeleitet werden. Die Simulation der Ist- und Soll-Modelle sowie der Vergleich der Simulationsergebnisse könnten nicht nur die Prozessverbesserung unterstützten, sondern auch die systematische Ableitung von Schritten für eine digitale Transformation von Unternehmen ermöglichen. Die Simulation verschiedener Prozessvarianten kann Schwachstellen wie Konflikte in den Operationen für Objekte oder unerwünschte Zustände aufdecken [Gehl19]. Die Integration von Techniken aus dem Object-Centric Process Mining, etwa für Conformance Checking und Predictive Monitoring [AaWe22] könnten diese Analysemöglichkeiten weiter ergänzen.

Ein vielversprechender Anwendungsbereich der generierten Geschäftsprozessmodelle ist die Prozessautomatisierung. Die generierten Modelle können als Grundlage für Automatisierungsbestrebungen dienen, da sie sowohl die Prozesslogik als auch die Datenmanipulationen formal spezifizieren [LeOb03]. Dabei könnten die extrahierten Objektstrukturen und -beziehungstypen direkt für die Konfiguration von IT-Systemen genutzt werden [LBWM+20]. Da regelbasierte Operationen für Dateien derzeit häufig mit Robotic Process Automation (RPA) automatisiert werden, könnte dies ein erfolgsversprechendes Anwendungsgebiet sein [RySc22, GZLA19]. Würden die generierten Modelle zur Automatisierung und Steuerung verwendet werden, könnten Veränderungen der Reihenfolge der Aktivitäten sowie Änderungen der Struktur der Objekte unmittelbar zu Änderungen der Automatisierung selbst führen. Durch die modellbasierte Umsetzung kann der Geschäftsprozess außerdem nicht nur automatisiert, sondern vorab auch analysiert und verbessert werden.

Zusammenfassend lässt sich feststellen, dass die Arbeit einen innovativen Beitrag zur automatisierten Geschäftsprozessmodellierung leistet. Die entwickelte Methode nutzt erstmals die Objekte eines Geschäftsprozesses als Datengrundlage für eine automatisierte Prozessmodellierung und ermöglicht so die Integration von Kontrollfluss- und Objektaspekten in den generierten Geschäftsprozessmodellen. Die prototypische Implementierung demonstriert die praktische Anwendbarkeit der Methode. Die vielfältigen Forschungsperspektiven unterstreichen das Potential der objektbasierten Prozessmodellgenerierung. Die Weiterentwicklung der Methode und des Software-Prototyps kann dazu beitragen, die Lücke zwischen der prozess- und objektorientierten Sichtweise bei der Prozessmodellierung zu schließen.

Literaturverzeichnis

[AaBe20] Aalst, W. M. P. van der; Berti, A.: Discovering Object-centric Petri Nets. Fundamenta Informaticae 1-4/175, S. 1–40, 2020.

[AaCa22] Aalst, W.M.P. van der; Carmona, J. Hrsg.: Process mining handbook. Springer, Cham, 2022.

[Aals11] Aalst, W. M. P. van der: Process Mining. Springer, Berlin, Heidelberg, 2011.

[Aals19] Aalst, W. M. P. van der: A practitioner's guide to process mining: Limitations of the directly-follows graph. Procedia Computer Science 164, S. 321–328, 2019.

[Aals22] Aalst, W. M. P. van der: Process Mining: A 360 Degree Overview. In (Aalst, W. M. P. van der; Carmona, J. Hrsg.): Process mining handbook. Springer, Cham, S. 3–35, 2022.

[Aals98] Aalst, W. M. P. van der: The Application of Petri Nets to Workflow Management. Journal of Circuits, Systems and Computers 01/08, S. 21–66, 1998.

[AaWe04] Aalst, W. M. P. van der; Weijters, A. J. M. M.: Process mining: a research agenda. Computers in Industry 3/53, S. 231–244, 2004.

[AaWM04] Aalst, W. M. P. van der; Weijters, T.; Maruster, L.: Workflow mining: discovering process models from event logs. IEEE Transactions on Knowledge and Data Engineering 9/16, S. 1128–1142, 2004.

[AbHV95] Abiteboul, S.; Hull, R.; Vianu, V.: Foundations of databases. Addison-Wesley, Reading, Mass., 1995.

[ABMN20] Aa, H. van der; Balder, K. J.; Maggi, F. M.; Nolte, A.: Say It in Your Own Words: Defining Declarative Process Models Using Speech Recognition. In (Fahland, D. et al. Hrsg.): Business Process Management Forum. Springer International Publishing, Cham, S. 51–67, 2020.

[ACDL+19] Augusto, A. et al.: Automated Discovery of Process Models from Event Logs: Review and Benchmark. IEEE Transactions on Knowledge and Data Engineering 4/31, S. 686–705, 2019.

[ACLR19] Aa, H. van der; Ciccio, C. Di; Leopold, H.; Reijers, H. A.: Extracting Declarative Process Models from Natural Language. In (Giorgini, P.; We-

ber, B. Hrsg.): Advanced Information Systems Engineering. Springer International Publishing, Cham, S. 365–382, 2019.

[AcVo13] Ackermann, L.; Volz, B.: model[NL]generation. In (Gray, J.; Kelly, S.; Sprinkle, J. Hrsg.): Proceedings of the 2013 ACM workshop on Domain-specific modeling - DSM '13. ACM Press, New York, USA, S. 45–50, 2013.

[AdGD21] Adamo, G.; Ghidini, C.; Di Francescomarino, C.: What is a process model composed of? A systematic literature review of meta-models in BPM. Software & Systems Modeling 4/20, S. 1215–1243, 2021.

[AgGL98] Agrawal, R.; Gunopulos, D.; Leymann, F.: Mining process models from workflow logs. In (Goos, G. et al. Hrsg.): Advances in Database Technology — EDBT'98. Springer, Berlin, Heidelberg, S. 467–483, 1998.

[AgHE20] Aghabaghery, R.; Hashemi Golpayegani, A.; Esmaeili, L.: A new method for organizational process model discovery through the analysis of workflows and data exchange networks. Social Network Analysis and Mining 1/10, 2020.

[AGTW11] Awad, A.; Goré, R.; Thomson, J.; Weidlich, M.: An Iterative Approach for Business Process Template Synthesis from Compliance Rules. In (Mouratidis, H.; Rolland, C. Hrsg.): Advanced Information Systems Engineering. CAiSE. Springer, Berlin, Heidelberg, S. 406–421, 2011.

[Agui04] Aguilar-Savén, R. S.: Business process modelling: Review and framework. International Journal of Production Economics 2/90, S. 129–149, 2004.

[AgZh12] Aggarwal, C.C.; Zhai, C. Hrsg.: Mining Text Data. Springer US, Boston, MA, 2012.

[AHKB03] Aalst, W. M. P. van der; Hofstede, A. H. ter; Kiepuszewski, B.; Barros, A. P.: Workflow Patterns. Distributed and Parallel Databases 1/14, S. 5–51, 2003.

[Alaw15] Alawairdhi, M.: Static analysis based business logic modelling from legacy system code: Business process model notation (BPMN) extraction using abstract syntax tree (AST): 2015 International Symposium on Networks, Computers and Communications (ISNCC). IEEE, S. 1–6, 2015.

[AMDG17] Aalst, W. M. P. van der; Masellis, R. de; Di Francescomarino, C.; Ghidini, C.: Learning Hybrid Process Models from Events. In (Carmona, J.; Engels, G.; Kumar, A. Hrsg.): Business Process Management. Springer International Publishing, Cham, S. 59–76, 2017.

AMFG17 Aalst, W. M. P. van der; Masellis, R. de; Francescomarino, C. Di; Ghidini, C.: Learning Hybrid Process Models From Events: Process Discovery Without Faking Confidence. arXiv, 2017.

[AnDQ22] Analytics for Information Systems (IS@CS); Department of Industrial Engineering & Innovation Sciences (IS@IEIS); Queensland University of Technology (QUT): BPM Center. a collaborative virtual research center in the area of BPM. http://bpmcenter.org/, Stand: 03.01.2022.

[ATSD+18] Almeida Bordignon, A. C. de; Thom, L. H.; Silva, T. S.; Dani, V. S.; Fantinato, M.; Ferreira, R. C. B.: Natural Language Processing in Business Process Identification and Modeling: Proceedings of the XIV Brazilian Symposium on Information Systems. ACM, New York, USA, S. 1–8, 2018.

[BaBC18] Banziger, R. B.; Basukoski, A.; Chaussalet, T.: Discovering Business Processes in CRM Systems by Leveraging Unstructured Text Data: International Conference on High Performance Computing and Communications; International Conference on Smart City; International Conference on Data Science and Systems (HPCC/SmartCity/DSS). IEEE, S. 1571–1577, 2018.

[BaEK20] Baïna, K.; El Hamlaoui, M.; Kabbaj, H.: Business Process Modelling Augmented. Model Driven transformation of User Stories to Processes: Proceedings of the 13th International Conference on Intelligent Systems: Theories and Applications. ACM, New York, USA, S. 1–6, 2020.

[BaHM15] Badouel, E.; Hélouët, L.; Morvan, C.: Petri Nets with Structured Data. In (Devillers, R.; Valmari, A. Hrsg.): Application and Theory of Petri Nets and Concurrency. Springer International Publishing, Cham, S. 212–233, 2015.

[Balz11] Balzert, H.: Lehrbuch der Softwaretechnik: Entwurf, Implementierung, Installation und Betrieb. Spektrum Akademischer Verlag, Heidelberg, 2011.

[BBMB+11] Brocke, J. vom et al.: Current and Future Issues in BPM Research: A European Perspective from the ERCIS Meeting 2010. Communications of the Association for Information Systems 28, 2011.

[BBSG+16] Beheshti, S.-M.-R. et al.: Process Analytics. Concepts and Techniques for Querying and Analyzing Process Data. Springer International Publishing, Cham, 2016.

[BeAa20] Berti, A.; Aalst, W. M. P. van der: Extracting Multiple Viewpoint Models from Relational Databases. In (Ceravolo, P.; van Keulen, M.; Gómez-López, M. T. Hrsg.): Data-Driven Process Discovery and Analysis. Springer International Publishing, Cham, S. 24–51, 2020.

[BeDG22] Bellan, P.; Dragoni, M.; Ghidini, C.: Leveraging pre-trained language models for conversational information seeking from text. http://arxiv.org/pdf/2204.03542v1.

[BeHO16] Betz, S.; Hickl, S.; Oberweis, A.: AAISP: An Approach for Aligning Information Systems Perspectives: 42th Euromicro Conference on Software Engineering and Advanced Applications (SEAA). IEEE, S. 339–346, 2016.

[BeMW09] Becker, J.; Mathas, C.; Winkelmann, A.: Geschäftsprozessmanagement. Springer, Berlin, Heidelberg, 2009.

[BePV12] Becker, J.; Probandt, W.; Vering, O.: Grundsätze ordnungsmäßiger Modellierung. Konzeption und Praxisbeispiel für ein effizientes Prozessmanagement. Springer, Berlin, 2012.

[BGB24] Bundesministerium der Finanzen: Gesetz zur Stärkung von Wachstumschancen, Investitionen und Innovation sowie Steuervereinfachung und Steuerfairness (Wachstumschancengesetz). BGBl. 2024 I Nr. 108 vom 27.03.2024, 2024.

[BKAP+23] Berti, A., et al.: OCEL (Object-Centric Event Log) 2.0 Specification. https://www.ocel-standard.org/2.0/ocel20_specification.pdf, Stand: 22.02.2024.

[BKBT+07] Brereton, P.; Kitchenham, B. A.; Budgen, D.; Turner, M.; Khalil, M.: Lessons from applying the systematic literature review process within the software engineering domain. Journal of Systems and Software 4/80, S. 571–583, 2007.

[BlAt03] Blumberg, R.; Atre, S.: The problem with unstructured data. Dm Review 42-49/13, S. 62, 2003.

[BMMS17] Bergmann, R.; Minor, M.; Müller, G.; Schumacher, P.: Project EVER: Extraction and Processing of Procedural Experience Knowledge in Workflows: ICCBR (Workshops), S. 137–146, 2017.

[Booc00] Booch, G.: Objektorientierte Analyse und Design. Mit praktischen Anwendungsbeispielen. Addison-Wesley, Bonn, 2000.

[Brin96] Brinkkemper, S.: Method engineering: engineering of information systems development methods and tools. Information and Software Technology 4/38, S. 275–280, 1996.

[BuDA12] Buijs, J. van; Dongen, B. F.; Aalst, W. van der: A genetic algorithm for discovering process trees: Congress on Evolutionary Computation. IEEE, S. 1–8, 2012.

[Capo16] Caporale, T.: Geschäftsprozessmodellierung mit kontrollierter natürlicher Sprache. In (Mayr, H. C.; Pinzger, M. Hrsg.): Informatik 2016. Gesellschaft für Informatik e.V, Bonn, S. 1957–1962, 2016.

[CDLW04] Cook, J. E.; Du, Z.; Liu, C.; Wolf, A. L.: Discovering models of behavior for concurrent workflows. Computers in Industry 3/53, S. 297–319, 2004.

[CeFB14] Cesar, I.; Fertalj, K.; Batos, V.: Towards a method to retrieving business process model from source code: Iberian Conference on Information Systems and Technologies (CISTI). IEEE, S. 1–6, 2014.

[CFFG+18] Corradini, F. et al.: A Guidelines framework for understandable BPMN models. Data & Knowledge Engineering 113, S. 129–154, 2018.

[Chen76] Chen, P. P.-S.: The entity-relationship model—toward a unified view of data. In (Hsiao, D. K. Hrsg.): Transactions on Database Systems. Association for Computing Machinery, New York, S. 9–36, 1976.

[ChKB18] Chadli, N.; Kabbaj, M. I.; Bakkoury, Z.: An Improved Ad Hoc Approach based on Active Help Method to Detect Data Flow Anomalies in a Loop of Business Process Modeling: Proceedings of the 1st International Conference of Computer Science and Renewable Energies. SCITEPRESS - Science and Technology Publications, S. 284–290, 2018.

[ChOJ15] Cheng, H.-J.; Ou-Yang, C.; Juan, Y.-C.: A hybrid approach to extract business process models with high fitness and precision. Journal of Industrial and Production Engineering 6/32, S. 351–359, 2015.

[CKSB11] Chanda, J.; Kanjilal, A.; Sengupta, S.; Bhattacharya, S.: FAM2BP: Transformation Framework of UML Behavioral Elements into BPMN Design Element. In (Meghanathan, N.; Kaushik, B. K.; Nagamalai, D. Hrsg.): Advances in Computer Science and Information Technology. Springer, Berlin, Heidelberg, S. 70–79, 2011.

[COWZ21] Combi, C.; Oliboni, B.; Weske, M.; Zerbato, F.: Seamless conceptual modeling of processes with transactional and analytical data. Data & Knowledge Engineering 134, S. 101895, 2021.

[CoYo91] Coad, P.; Yourdon, E.: Object-oriented analysis (2nd ed.). Yourdon Press, USA, 1991.

[CWFL+19] Cheng, N.; Wang, L.; Fei, R.; Li, W.; Wang, B.: Workflow Model Mining Based On Educational Management Data Logs: 2019 Chinese Control And Decision Conference (CCDC). IEEE, S. 5450–5455, 2019.

[DaSB19] Danenas, P.; Skersys, T.; Butleris, R.: Enhancing the extraction of SBVR business vocabularies and business rules from UML use case diagrams with natural language processing. In (Manolopoulos, Y. et al. Hrsg.): Proceedings of the 23rd Pan-Hellenic Conference on Informatics. ACM, New York, USA, S. 1–8, 2019.

[Dave93] Davenport, T. H.: Process innovation. Reengineering work through information technology. Harvard Business School Press, Boston, Mass., 1993.

[DeCh16] Deb, D.; Chaki, N.: A Framework towards Generating Effective Business Process Model by Goal based Pruning. In (Akila, V. et al. Hrsg.): Proceedings of the International Conference on Informatics and Analytics. ACM, New York, USA, S. 1–4, 2016.

[DePo14] Decreus, K.; Poels, G.: A Goal-Oriented Requirements Engineering Method for Business Processes. In (Bayro-Corrochano, E.; Hancock, E. Hrsg.): Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications. Springer International Publishing, Cham, S. 29–43, 2014.

[DhRa10] Dhaussy, P.; Raji, A.: Automatic Formal Model Derivation from Use Cases: Journées sur l'Ingénierie Dirigée par les Modèles, 2010.

[DiJZ16] Ding, Z.; Jiang, M.; Zhou, M.: Generating Petri Net-Based Behavioral Models From Textual Use Cases and Application in Railway Networks. IEEE Transactions on Intelligent Transportation Systems 12/17, S. 3330–3343, 2016.

[DiMe13] Di Ciccio, C.; Mecella, M.: A two-step fast algorithm for the automated discovery of declarative workflows: IEEE Symposium on Computational Intelligence and Data Mining (CIDM). IEEE, S. 135–142, 2013.

[DiTC21] Dinter, R. van; Tekinerdogan, B.; Catal, C.: Automation of systematic literature reviews: A systematic literature review. Information and Software Technology 136, S. 106589, 2021.

[DKŽA20] Djedović, A.; Karabegović, A.; Žunić, E.; Alić, D.: A Rule Based Events Correlation Algorithm for Process Mining. In (Avdaković, S. et al. Hrsg.): Advanced Technologies, Systems, and Applications IV -

Proceedings of the International Symposium on Innovative and Interdisciplinary Applications of Advanced Technologies (IAT 2019). Springer International Publishing, Cham, S. 587–605, 2020.

[DLMR18] Dumas, M.; La Rosa, M.; Mendling, J.; Reijers, H. A.: Fundamentals of Business Process Management. Springer, Berlin, Heidelberg, 2018.

[DMSZ+12] Di Ciccio, C.; Mecella, M.; Scannapieco, M.; Zardetto, D.; Catarci, T.: MailOfMine – Analyzing Mail Messages for Mining Artful Collaborative Processes. In (Aberer, K.; Damiani, E.; Dillon, T. Hrsg.): Data-Driven Process Discovery and Analysis. Springer, Berlin, Heidelberg, S. 55–81, 2012.

[DOJC+93] Davis, A. et al.: Identifying and measuring quality in a software requirements specification: Proceedings First International Software Metrics Symposium, S. 141–152, 1993.

[EAZP+12] Engel, R.; Aalst, W. M. P. van der; Zapletal, M.; Pichler, C.; Werthner, H.; van der Aalst, W. M. P.: Mining Inter-organizational Business Process Models from EDI Messages: A Case Study from the Automotive Sector. In (Ralyté, J. et al. Hrsg.): Advanced Information Systems Engineering. CAiSE. Springer, Berlin, Heidelberg, S. 222–237, 2012.

[EfSa20] Effendi, Y. A.; Sarno, R.: Time-based α+ miner for modelling business processes using temporal pattern. TELKOMNIKA (Telecommunication Computing Electronics and Control) 1/18, S. 114, 2020.

[EHSR+22] Elwany, E.; Hegel, A.; Shah, M.; Roof, B.; Peaslee, G.; Rivet, Q.: DeeperDive: The Unreasonable Effectiveness of Weak Supervision in Document Understanding A Case Study in Collaboration with UiPath Inc. http://arxiv.org/pdf/2208.08000v1.

[EMHD+15] Epure, E. V.; Martin-Rodilla, P.; Hug, C.; Deneckere, R.; Salinesi, C.: Automatic process model discovery from textual methodologies: International Conference on Research Challenges in Information Science (RCIS). IEEE, S. 19–30, 2015.

[EnGr11] Engelmann, F.; Großmann, C.: Was wissen wir über Information? In (Hildebrand, K. et al. Hrsg.): Daten- und Informationsqualität. Vieweg+Teubner, Wiesbaden, S. 3–24, 2011.

[Esch85] Eschenröder, G.: Planungsaspekte einer ressourcenorientierten Informationswirtschaft. Zugl.: Köln, Univ., Diss., 1984. Eul, Bergisch Gladbach, 1985.

[FBMM+24] Fazio, R. de et al.: CaseID Detection for Process Mining: A Heuristic-Based Methodology. In (Smedt, J. de; Soffer, P. Hrsg.): Process Mining Workshops. Springer Nature Switzerland, Cham, S. 45–57, 2024.

[FBZN+24] Feyisa, D. W.; Berihun, H.; Zewdu, A.; Najimoghadam, M.; Zare, M.: The future of document indexing: GPT and Donut revolutionize table of content processing. http://arxiv.org/pdf/2403.07553v1.

[FHSB+22] Fritsch, A.; Hammerstein, J. von; Schreiber, C.; Betz, S.; Oberweis, A.: Pathways to Greener Pastures: Research Opportunities to Integrate Life Cycle Assessment and Sustainable Business Process Management Based on a Systematic Tertiary Literature Review. Sustainability 18/14, S. 11164, 2022.

[FiAv14] Figueira, C.; Aveiro, D.: A New Action Rule Syntax for DEmo MOdels Based Automatic worKflow procEss geneRation (DEMOBAKER). In (Aalst, W. M. P. van der et al. Hrsg.): Advances in Enterprise Engineering VIII. Springer International Publishing, Cham, S. 46–60, 2014.

[FiFK23] Fill, H.-G.; Fettke, P.; Köpke, J.: Conceptual Modeling and Large Language Models: Impressions From First Experiments With ChatGPT. 3:1-15 Pages / Enterprise Modelling and Information Systems Architectures (EMISAJ), Vol. 18 (2023). Enterprise Modelling and Information Systems Architectures (EMISAJ) 18, 1-15, 2023.

[FiHR08] Fieber, F.; Huhn, M.; Rumpe, B.: Modellqualität als Indikator für Softwarequalität: eine Taxonomie. Informatik-Spektrum 5/31, S. 408–424, 2008.

[FlKM05] Fliedl, G.; Kop, C.; Mayr, H. C.: From textual scenarios to a conceptual schema. Data & Knowledge Engineering 1/55, S. 20–37, 2005.

[FoGP15] Folino, F.; Guarascio, M.; Pontieri, L.: On the Discovery of Explainable and Accurate Behavioral Models for Complex Lowly-structured Business Processes: Proceedings of the 17th International Conference on Enterprise Information Systems. SCITEPRESS - Science and and Technology Publications, S. 206–217, 2015.

[FoSc24] Forell, M.; Schüler, S.: Modeling meets Large Language Models. Modellierung Satellite Events. Potsdam, 2024.

[Foth10] Foth, E.: Exzellente Geschäftsprozesse mit SAP. Praxis des Einsatzes in Unternehmensgruppen. Springer, Berlin, Heidelberg, 2010.

[Fran10] Frank, U.: The MEMO Meta Modelling Language (MML) and Language Architecture. DuEPublico: Duisburg-Essen Publications Online, University

of Duisburg-Essen, Germany, ICB Research Reports - Forschungsberichte des ICB, 2010.

[FrGG97] Friedman, N.; Geiger, D.; Goldszmidt, M.: Bayesian network classifiers. Machine Learning 2/3/29, S. 131–163, 1997.

[Frit25] Fritsch, A.: Nachhaltiges Geschäftsprozessmanagement mit JSON-Netzen. Dissertation. Karlsruher Institut für Technologie (KIT), Karlsruhe, 2025.

[FrLa03] Frank, U.; Laak, B. L. van: Anforderungen an Sprachen zur Modellierung von Geschäftsprozessen. Arbeitsberichte des Instituts für Wirtschafts- und Verwaltungsinformatik. Universität Koblenz-Landau, Koblenz-Landau, 2003.

[FrMP11] Friedrich, F.; Mendling, J.; Puhlmann, F.: Process Model Generation from Natural Language Text. In (Mouratidis, H.; Rolland, C. Hrsg.): Advanced Information Systems Engineering. CAiSE. Springer, Berlin, Heidelberg, S. 482–496, 2011.

[FrRü17] Freund, J.; Rücker, B.: Praxishandbuch BPMN. Mit Einführung in CMMN und DMN. Hanser, München, 2017.

[FSFO23] Fritsch, A.; Schüler, S.; Forell, M.; Oberweis, A.: Modelling and Execution of Data-Driven Processes with JSON-Nets. In (Aa, H. van der et al. Hrsg.): Enterprise, Business-Process and Information Systems Modeling. 29–43, Cham, S. 29–43, 2023.

[FSSO+14] Fleischmann, A.; Schmidt, W.; Stary, C.; Obermeier, S.; Börger, E.: Subject-Oriented Business Process Management. Springer, Berlin, 2014.

[FTOA+17] Ferreira, R. C. B.; Thom, L. H.; Oliveira, J. P. M. de; Avila, D. T.; dos Santos, R. I.; Fantinato, M.: Assisting Process Modeling by Identifying Business Process Elements in Natural Language Texts. In (Cesare, S. de; Frank, U. Hrsg.): Advances in Conceptual Modeling. Springer International Publishing, Cham, S. 154–163, 2017.

[FZDL12] Fan, S.; Zhao, J. L.; Dou, W.; Liu, M.: A framework for transformation from conceptual to logical workflow models. Decision Support Systems 1/54, S. 781–794, 2012.

[Gada17] Gadatsch, A.: Grundkurs Geschäftsprozess-Management. Analyse, Modellierung, Optimierung und Controlling von Prozessen. Springer Vieweg, Wiesbaden, 2017.

[GBGO21] Ghanbari Ghooshchi, N.; Beest, N. van; Governatori, G.; Olivieri, F.: Synthesis of Regulation Compliant Business Processes. IEEE Transactions on Services Computing 4/14, S. 1179–1193, 2021.

[GCPV22] Goossens, A.; Claessens, M.; Parthoens, C.; Vanthienen, J.: Extracting Decision Dependencies and Decision Logic from Text Using Deep Learning Techniques. In (Marrella, A.; Weber, B. Hrsg.): Business Process Management Workshops. Springer International Publishing, Cham, S. 349–361, 2022.

[GDLW+14] García-Bañuelos, L.; Dumas, M.; La Rosa, M.; Weerdt, J. de; Ekanayake, C. C.: Controlled automated discovery of collections of business process models. Information Systems 46, S. 85–101, 2014.

[Gehl19] Gehlot, V.: From Petri NETS to Colored Petri NETS: A Tutorial Introduction to NETS Based Formalism For Modeling And Simulation: Winter Simulation Conference (WSC). IEEE, S. 1519–1533, 2019.

[GeLa79] Genrich, H. J.; Lautenbach, K.: The analysis of distributed systems by means of predicate/transition-nets. In (Kahn, G. Hrsg.): Semantics of Concurrent Computation. Springer-Verlag, Berlin/Heidelberg, S. 123–146, 1979.

[GGPS06] Greco, G.; Guzzo, A.; Pontieri, L.; Sacca, D.: Discovering expressive process models by clustering log traces. IEEE Transactions on Knowledge and Data Engineering 8/18, S. 1010–1027, 2006.

[GHJV95] Gamma, E.; Helm, R.; Johnson, R.; Vlissides, J. M.: Design patterns. Elements of reusable object-oriented software. Addison-Wesley, Reading, Mass., 1995.

[GhKC07] Ghose, A.; Koliadis, G.; Chueng, A.: Process Discovery from Model and Text Artefacts: IEEE Congress on Services (Services 2007). IEEE, S. 167–174, 2007.

[GhSP10] Ghattas, J.; Soffer, P.; Peleg, M.: A Formal Model for Process Context Learning. In (Aalst, W. M. P. van der et al. Hrsg.): Business Process Management Workshops. Springer, Berlin, Heidelberg, S. 140–157, 2010.

[GHvS+98] Goos, G. et al. Hrsg.: Advances in Database Technology — EDBT'98. Springer, Berlin, Heidelberg, 1998.

[GLMH11] Gambini, M.; La Rosa, M.; Migliorini, S.; Hofstede, A. H. M. ter: Automated error correction of business process models: International Conference on Business Process Management. Springer, S. 148–165, 2011.

[GNEM+08] Gutiérrez, J. J.; Nebut, C.; Escalona, M. J.; Mejías, M.; Ramos, I. M.: Visualization of Use Cases through Automatically Generated Activity Diagrams. In (Czarnecki, K. et al. Hrsg.): Model Driven Engineering Languages and Systems. Springer, Berlin, Heidelberg, S. 83–96, 2008.

[GNPP+11] Ghose, A. K.; Narendra, N. C.; Ponnalagu, K.; Panda, A.; Gohad, A.: Goal-Driven Business Process Derivation. In (Kappel, G.; Maamar, Z.; Motahari-Nezhad, H. R. Hrsg.): Service-Oriented Computing. Springer, Berlin, Heidelberg, S. 467–476, 2011.

[Goog14] Google: Jsonnet. https://jsonnet.org/, Stand: 25.04.2023.

[GoPi03] Golani, M.; Pinter, S. S.: Generating a Process Model from a Process Audit Log. In (Goos, G. et al. Hrsg.): Business Process Management. Springer, Berlin, Heidelberg, S. 136–151, 2003.

[GoRA19] González López de Murillas, E.; Reijers, H. A.; Aalst, W. M. P. van der: Connecting databases with process mining: a meta model and toolset. Software & Systems Modeling 2/18, S. 1209–1247, 2019.

[GoSB11] Gonçalves, J. C. d. A. R.; Santoro, F. M.; Baião, F. A.: Let Me Tell You a Story - On How to Build Process Models. Journal of Universal Computer Science 17 (2), S. 276–295, 2011.

[Göss07] Gössner, S.: JSONPath. XPath for JSON. https://goessner.net/articles/JsonPath/, Stand: 25.04.2023.

[GrMM18] Grigorova, K.; Mironov, K.; Malysheva, E.: Applying process mining techniques and neural networks to creating and assessment of business process models: International Conference on "Information Technology and Nanotechnology"Technology and Nanotechnology. IP Zaitsev V.D, S. 278–286, 2018.

[GuDe17] Gurbuz, O.; Demirors, O.: From Organizational Guidelines to Business Process Models: Exploratory Case for an Ontology Based Methodology: 2017 IEEE 19th Conference on Business Informatics (CBI). IEEE, S. 320–329, 2017.

[GZLA19] Gao, J.; Zelst, S. J. van; Lu, X.; Aalst, W. M. P. van der: Automated Robotic Process Automation: A Self-Learning Approach. In (Panetto, H. et al. Hrsg.): On the Move to Meaningful Internet Systems: OTM 2019 Conferences. Springer International Publishing, Cham, S. 95–112, 2019.

[HaCh93] Hammer, M.; Champy, J.: Reengineering the corporation: A manifesto for business revolution. Business Horizons 5/36, S. 90–91, 1993.

[HaKa16] Hamad, K.; Kaya, M.: A Detailed Analysis of Optical Character Recognition Technology. International Journal of Applied Mathematics, Electronics and Computers Special Issue-1/4, S. 244, 2016.

[HaKP11] Han, J.; Kamber, M.; Pei, J.: Data mining. Concepts and techniques. Elsevier Morgan Kaufmann, Amsterdam, Boston, Heidelberg, 2011.

[HALD11] Huang, Z.; Aalst, W. van der; Lu, X.; Duan, H.: Reinforcement learning based resource allocation in business process management. Data & Knowledge Engineering 1/70, S. 127–145, 2011.

[HeNW06] Hegewald, J.; Naumann, F.; Weis, M.: XStruct: Efficient Schema Extraction from Multiple and Large XML Documents: 22nd International Conference on Data Engineering Workshops (ICDEW'06). IEEE, S. 81, 2006.

[HePv11] Hermans, F.; Pinzger, M.; van Deursen, A.: Supporting professional spreadsheet users by generating leveled dataflow diagrams. In (Taylor, R. N.; Gall, H.; Medvidović, N. Hrsg.): Proceedings of the 33rd International Conference on Software Engineering. ACM, New York, USA, S. 451–460, 2011.

[HGAE+22] Hübscher, G. et al.: Graph-based managing and mining of processes and data in the domain of intellectual property. Information Systems 106, S. 101844, 2022.

[HHMD+20] Han, X. et al.: A-BPS: Automatic Business Process Discovery Service using Ordered Neurons LSTM: IEEE International Conference on Web Services (ICWS). IEEE, S. 428–432, 2020.

[HJBG21] Haj, A.; Jarrar, A.; Balouki, Y.; Gadir, T.: The Semantic of Business Vocabulary and Business Rules: An Automatic Generation From Textual Statements. IEEE Access 9, S. 56506–56522, 2021.

[HnCe21] Hnatkowska, B.; Cebinka, M.: Activity Diagram Generation Based on Use-Case Textual Specification. Computing and Informatics 4/40, S. 772–795, 2021.

[HoKW18] Honkisz, K.; Kluza, K.; Wiśniewski, P.: A Concept for Generating Business Process Models from Natural Language Description. In (Liu, W.; Giunchiglia, F.; Yang, B. Hrsg.): Knowledge Science, Engineering and Management. Springer International Publishing, Cham, S. 91–103, 2018.

[HSSS20] Heinrich, B.; Schiller, A.; Schön, D.; Szubartowicz, M.: Adapting process models via an automated planning approach. Journal of Decision Systems 4/29, S. 223–259, 2020.

[HSVW09] Hee, K. M. van; Sidorova, N.; Voorhoeve, M.; Werf, J. M. van der: Generation of Database Transactions with Petri Nets. Fundamenta Informaticae 1-3/93, S. 171–184, 2009.

[HUCM19] HU, W.; Chen, T.; Meyer, G.: Constructing workflow models of alarm responses via trace labeling and dependency analysis: 2019 58th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE). IEEE, S. 1615–1620, 2019.

[HuXH11] Hu, H.; Xie, J.; Hu, H.: A novel approach for mining stochastic process model from workflow logs. Journal of Computational Information Systems 9/7, S. 3113–3126, 2011.

[HZFD12] Hernandes, E.; Zamboni, A.; Fabbri, S.; Di Thommazo, A.: Using GQM and TAM to evaluate StArt – a tool that supports Systematic Review. CLEI Electronic Journal 1/15, 2012.

[IMDD+23] Ignaczak, L.; Martins, M. G.; Da Costa, C. A.; Donida, B.; Da Silva, M. C. P.: An evaluation of NERC learning-based approaches to discover personal data in Brazilian Portuguese documents. Discover Data 1/1, 2023.

[IqBa16] Iqbal, U.; Bajwa, I. S.: Generating UML activity diagram from SBVR rules: 2016 Sixth International Conference on Innovative Computing Technology (INTECH). IEEE, S. 216–219, 2016.

[ISO11] ISO/IEC 25010:2011(E):2011, Systems and software engineering - Systems and software quality requirements and evaluation (SQuaRE) - System and software quality models.

[ISO15] ISO/IEC 9000-2015:2015, Quality management systems: fundamentals and vocabulary.

[JaDu88] Jain, A. K.; Dubes, R. C.: Algorithms for clustering data. Prentice-Hall, Englewood Cliffs, NJ, 1988.

[JaGo08] Jablonski, S.; Goetz, M.: Perspective oriented business process visualization. In (Hofstede, A. ter; Benatallah, B.; Paik, H.-Y. Hrsg.): Business process management workshops. Springer, Berlin, Heidelberg, S. 144–155, 2008.

[JaHu11] Janjua, N. K.; Hussain, F. K.: Rule-Based Business Policies Specification, Reasoning and Integration for Business Process Model Extraction: 2011 International Conference on Broadband and Wireless Computing, Communication and Applications. IEEE, S. 51–56, 2011.

[JaLi21] Javed, M.; Lin, Y.: iMER: Iterative process of entity relationship and business process model extraction from the requirements. Information and Software Technology 135, S. 106558, 2021.

[JaLi96] Jarzabek, S.; Ling, T. W.: Model-based support for business re-engineering. Information and Software Technology 5/38, S. 355–374, 1996.

[Jens81] Jensen, K.: Coloured petri nets and the invariant-method. Theoretical Computer Science 3/14, S. 317–336, 1981.

[Jian12] Jiang, J.: Information Extraction from Text. In (Aggarwal, C. C.; Zhai, C. Hrsg.): Mining Text Data. Springer US, Boston, MA, S. 11–41, 2012.

[JlGB17] Jlailaty, D.; Grigori, D.; Belhajjame, K.: Mining Business Process Activities from Email Logs: 2017 IEEE International Conference on Cognitive Computing (ICCC). IEEE, S. 112–119, 2017.

[JLKR07] Jansen-Vullers, M. H. M.; Loosschilder, M. W. N. C.; Kleingeld, P. A. M.; Reijers, H. A.: Performance measures to evaluate the impact of best practices: Proceedings of the 19th International Conference on Advanced Information Systems Engineering (CAiSE'07). Tapir Academic Press, Trondheim, 2007.

[Joch09] Jochem, R.: Quality governance. Total Quality Management & Business Excellence 7/20, S. 777–785, 2009.

[JRBD19] Jimenez-Ramirez, A.; Reijers, H. A.; Barba, I.; Del Valle, C.: A Method to Improve the Early Stages of the Robotic Process Automation Lifecycle. In (Giorgini, P.; Weber, B. Hrsg.): Advanced Information Systems Engineering. Springer International Publishing, Cham, S. 446–461, 2019.

[JWBD15] Jiménez-Ramírez, A.; Weber, B.; Barba, I.; Del Valle, C.: Generating optimized configurable business process models in scenarios subject to uncertainty. Information and Software Technology 57, S. 571–594, 2015.

[KaMA15] Kamarudin, N. J.; Mohd Sani, N. F.; Atan, R.: Automated transformation approach from user requirement to behavior design. Journal of Theoretical and Applied Information Technology 81 (1), S. 73–83, 2015.

[Kasc99] Kaschek, R.: Was sind eigentlich Modelle? In (Gesellschaft für Informatik (GI) Hrsg.): EMISA Forum, S. 31–35, 1999.

[KEKR21] Kecht, C.; Egger, A.; Kratsch, W.; Roglinger, M.: Event Log Construction from Customer Service Conversations Using Natural Language Infer-

ence: International Conference on Process Mining (ICPM). IEEE, S. 144–151, 2021.

[KeNS92] Keller, G.; Nüttgens, M.; Scheer, A.-W.: Semantische Prozeßmodellierung auf der Grundlage "Ereignisgesteuerter Prozeßketten (EPK)". Institut für Wirtschaftsinformatik Universität, Saarbrücken, 1992.

[KiCh07] Kitchenham, B.; Charters, S.: Guidelines for performing Systematic Literature Reviews in Software Engineering 2, 2007.

[Kind94] Kindler, E.: Safety and liveness properties: A survey. Bulletin of the European Association for Theoretical Computer Science 268-272/53, S. 30, 1994.

[KITd14] Kalsing, A. C.; Iochpe, C.; Thom, L. H.; do Nascimento, G. S.: Re-learning of Business Process Models from Legacy System Using Incremental Process Mining. In (Hammoudi, S. et al. Hrsg.): Enterprise Information Systems. Springer International Publishing, Cham, S. 314–330, 2014.

[KKSS23] Klessinger, S.; Klettke, M.; Störl, U.; Scherzinger, S.: Extracting JSON Schemas with Tagged Unions. http://arxiv.org/pdf/2306.07085v1.

[KlHo16] Kluza, K.; Honkisz, K.: From SBVR to BPMN and DMN Models. Proposal of Translation from Rules to Process and Decision Models. In (Rutkowski, L. et al. Hrsg.): Artificial Intelligence and Soft Computing. Springer International Publishing, Cham, S. 453–462, 2016.

[KlNa15] Kluza, K.; Nalepa, G. J.: Generation of Hierarchical Business Process Models from Attribute Relationship Diagrams. In (Mach-Król, M.; M. Olszak, C.; Pełech-Pilichowski, T. Hrsg.): Advances in ICT for Business, Industry and Public Sector. Springer International Publishing, Cham, S. 57–76, 2015.

[Kneu15] Kneuper, R.: Messung und Bewertung von Prozessqualität – Ein Baustein der Governance. HMD Praxis der Wirtschaftsinformatik 2/52, S. 301–311, 2015.

[Kobl10] Kobler, M.: Qualität von Prozessmodellen. Kennzahlen zur analytischen Qualitätssicherung bei der Prozessmodellierung. Dissertation, Berlin, 2010.

[KoOr20] Kopp, A.; Orlovskyi, D.: Towards the Generalized Criterion for Evaluation of Business Process Model Quality. ICT in Education, Research and Industrial Applications ; PhD Symposium 3, 2020.

[KovW18] Koninck, P. de; vanden Broucke, S.; Weerdt, J. de: act2vec, trace2vec, log2vec, and model2vec: Representation Learning for Business Processes.

In (Weske, M. et al. Hrsg.): Business Process Management. Springer International Publishing, Cham, S. 305–321, 2018.

[KrDe10] Krumnow, S.; Decker, G.: A Concept for Spreadsheet-Based Process Modeling. In (Mendling, J.; Weidlich, M.; Weske, M. Hrsg.): Business Process Modeling Notation. Springer, Berlin, Heidelberg, S. 63–77, 2010.

[KsKo21] Kshenin, A.; Kovalchuk, S.: Data-Driven Modeling of Complex Business Process in Heterogeneous Environment of Healthcare Organization with Health Information Systems. Studies in health technology and informatics 285, S. 118–123, 2021.

[KuBV10] Kumar, R.; Bhattacharyya, C.; Varshneya, V.: Discovering Business Process Model from Unstructured Activity Logs: 2010 IEEE International Conference on Services Computing. IEEE, S. 250–257, 2010.

[KüKR06] Küster, J. M.; Koehler, J.; Ryndina, K.: Improving Business Process Models with Reference Models in Business-Driven Development. In (Hutchison, D. et al. Hrsg.): Business Process Management Workshops. Springer, Berlin, Heidelberg, S. 35–44, 2006.

[KüRG07] Küster, J. M.; Ryndina, K.; Gall, H.: Generation of Business Process Models for Object Life Cycle Compliance. In (Alonso, G.; Dadam, P.; Rosemann, M. Hrsg.): Business Process Management. Springer, Berlin, Heidelberg, S. 165–181, 2007.

[KWZA+22] Kluza, K. et al.: Proposal of a Method for Creating a BPMN Model Based on the Data Extracted from a DMN Model. In (Memmi, G. et al. Hrsg.): Knowledge Science, Engineering and Management. Springer International Publishing, Cham, S. 349–358, 2022.

[KYYW20] Kunchala, J.; Yu, J.; Yongchareon, S.; Wang, G.: Trace-Based Approach for Consistent Construction of Activity-Centric Process Models from Data-Centric Process Models. In (Borovica-Gajic, R.; Qi, J.; Wang, W. Hrsg.): Databases Theory and Applications. Springer International Publishing, Cham, S. 42–54, 2020.

[Lamg21] Lamghari, Z.: An Integrated Approach for Discovering Process Models According to Business Process Types. ASM Science Journal 16, S. 1–14, 2021.

[LaOj96] Laaksonen, J.; Oja, E.: Classification with learning k-nearest neighbors: Proceedings of International Conference on Neural Networks (ICNN'96). IEEE, S. 1480–1483, 1996.

[LBWM+20] Leroy, D.; Bousse, E.; Wimmer, M.; Mayerhofer, T.; Combemale, B.; Schwinger, W.: Behavioral interfaces for executable DSLs. Software & Systems Modeling 4/19, S. 1015–1043, 2020.

[LCZW22] Liu, C.; Cheng, L.; Zeng, Q.; Wen, L.: Formal Modeling and Discovery of Hierarchical Business Processes: A Petri Net-Based Approach. IEEE Transactions on Systems, Man, and Cybernetics: Systems, S. 1–12, 2022.

[LeAa15] Leemans, M.; Aalst, W. M. P. van der: Process mining in software systems: Discovering real-life business transactions and process models from distributed systems: 2015 ACM/IEEE 18th International Conference on Model Driven Engineering Languages and Systems (MODELS). IEEE, S. 44–53, 2015.

[LeMS21] Lekić, J.; Milićev, D.; Stanković, D.: Generating Block-Structured Parallel Process Models by Demonstration. Applied Sciences 4/11, S. 1876, 2021.

[LeOb03] Lenz, K.; Oberweis, A.: Inter-organizational Business Process Management with XML Nets. In (Goos, G. et al. Hrsg.): Petri Net Technology for Communication-Based Systems. Springer, Berlin, Heidelberg, S. 243–263, 2003.

[LGHH+16] Li, C. et al.: Process mining with token carried data. Information Sciences 328, S. 558–576, 2016.

[LGXX+14] Liu, C.; Ge, Y.; Xiong, H.; Xiao, K.; Geng, W.; Perkins, M.: Proactive workflow modeling by stochastic processes with application to healthcare operation and management. In (Macskassy, S. et al. Hrsg.): Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, New York, USA, S. 1593–1602, 2014.

[LiDi13] Liu, X.; Ding, C.: Learning workflow models from event logs using co-clustering. International Journal of Web Services Research (IJWSR) 3/10, S. 42–59, 2013.

[LiFe06a] Li, Y.; Feng, Y.: An Automatic Business Process Modeling Method Based on Markov Matrix: 2006 International Conference on Machine Learning and Cybernetics. IEEE, S. 1302–1307, 2006.

[LiFe06b] Li, Y.; Feng, Y.: Design of an Automatic Workflow Modeling Method in Cooperative WFMS: 2006 10th International Conference on Computer Supported Cooperative Work in Design. IEEE, S. 1–6, 2006.

[Lins02] Linsen, O.: Die objektorientierte Modellierung von Geschäftsprozessen. ie erweiterte Object Behavior Analysis (OBA++) als Ergänzung der Unified

Modeling Language (UML). Dissertation. VDM Verlag Dr. Müller, Wuppertal, 2002.

[LiSS94] Lindland, O. I.; Sindre, G.; Sølvberg, A.: Understanding Quality in Conceptual Modeling: IEEE Software, S. 42–49, 1994.

[LoKR09] Lodhi, A.; Kassem, G.; Rautenstrauch, C.: Modeling and analysis of business processes using business objects: 2009 2nd International Conference on Computer, Control and Communication. IEEE, S. 1–6, 2009.

[Lübk06] Lübke, D.: Transformation of Use Cases to EPC models, Conference: 5. Workshop der Gesellschaft für Informatik e.V. (GI) und Treffen ihres Arbeitskreises "Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten (WI-EPK)", 2006.

[LüSc08] Lübke, D.; Schneider, K.: Visualizing Use Case Sets as BPMN Processes: 2008 Requirements Engineering Visualization. IEEE, S. 21–25, 2008.

[LWZZ10] Li, J.; Wang, H. J.; Zhang, Z.; Zhao, J. L.: A policy-based process mining framework: mining business policy texts for discovering process models. Information Systems and e-Business Management 2/8, S. 169–188, 2010.

[LYCT+18] Li, J.; Yang, S.; Chen, S.; Tao, F.; Marsic, I.; Burd, R. S.: Discovering interpretable medical workflow models: 2018 IEEE International Conference on Healthcare Informatics (ICHI), S. 437–439, 2018.

[LZLJ12] Liu, Y.; Zhang, H.; Li, C.; Jiao, R. J.: Workflow simulation for operational decision support using event graph through process mining. Decision Support Systems 3/52, S. 685–697, 2012.

[MAAB+19] Maqbool, B. et al.: A Comprehensive Investigation of BPMN Models Generation from Textual Requirements—Techniques, Tools and Trends. In (Kim, K. J.; Baek, N. Hrsg.): Information Science and Applications 2018. Springer, Singapore, S. 543–557, 2019.

[MaAb21] Maatuk, A. M.; Abdelnabi, E. A.: Generating UML Use Case and Activity Diagrams Using NLP Techniques and Heuristics Rules. In (Lara Torralbo, J. A. et al. Hrsg.): International Conference on Data Science, E-learning and Information Systems 2021. ACM, New York, USA, S. 271–277, 2021.

[McKi10] McKinney, W.: Data Structures for Statistical Computing in Python: Proceedings of the 9th Python in Science Conference. SciPy, S. 56–61, 2010.

[Mend08] Mendling, J.: Metrics for process models. Springer, 2008.

[Mend09] Mendling, J.: Empirical Studies in Process Model Verification. In (Jensen, K.; Aalst, W. M. . P. van der Hrsg.): Transactions on Petri Nets and Other Models of Concurrency II. Springer, Berlin, Heidelberg, S. 208–224, 2009.

[MeRv10] Mendling, J.; Reijers, H. A.; van der Aalst, W.: Seven process modeling guidelines (7PMG). Information and Software Technology 2/52, S. 127–136, 2010.

[MeSW11] Meyer, A.; Smirnov, S.; Weske, M.: Data in business processes. Univ.-Verl. Potsdam, Potsdam, 2011.

[MHAB+23] Mosqueira-Rey, E.; Hernández-Pereira, E.; Alonso-Ríos, D.; Bobes-Bascarán, J.; Fernández-Leal, Á.: Human-in-the-loop machine learning: a state of the art. Artificial Intelligence Review 4/56, S. 3005–3054, 2023.

[MiLA17] Mitsyuk, A. A.; Lomazova, I. A.; Aalst, W. M. P. van der: Using Event Logs for Local Correction of Process Models. Automatic Control and Computer Sciences 7/51, S. 709–723, 2017.

[Mirb17] Mirbel, I.: From user goals to process-based service compositions: A flexible semantic-based approach: 2017 11th International Conference on Research Challenges in Information Science (RCIS). IEEE, S. 95–102, 2017.

[MLTA19] Mendling, J.; Leopold, H.; Thom, L. H.; Aa, H. van der: Natural Language Processing with Process Models (NLP4RE Report Paper): REFSQ Workshops, 2019.

[Mood05] Moody, D. L.: Theoretical and practical issues in evaluating the quality of conceptual models: current state and future directions: Data & Knowledge Engineering, S. 243–276, 2005.

[MrMB16] Mrasek, R.; Mülle, J.; Böhm, K.: Process Synthesis with Sequential and Parallel Constraints. In (Debruyne, C. et al. Hrsg.): On the Move to Meaningful Internet Systems: OTM 2016 Conferences. Springer International Publishing, Cham, S. 43–60, 2016.

[MSRR15] Moreno-Montes de Oca, I.; Snoeck, M.; Reijers, H. A.; Rodríguez-Morffi, A.: A systematic literature review of studies on business process modeling quality. Information and Software Technology 58, S. 187–205, 2015.

[MTGH+23] Madaan, A. et al.: Self-Refine: Iterative Refinement with Self-Feedback. In (A. Oh et al. Hrsg.): Advances in Neural Information Processing Systems. Curran Associates, Inc, S. 46534–46594, 2023.

[MüBe17] Müller, G.; Bergmann, R.: Complexity-Aware Generation of Workflows by Process-Oriented Case-Based Reasoning. In (Kern-Isberner, G.; Fürnkranz, J.; Thimm, M. Hrsg.): KI 2017: Advances in Artificial Intelligence. Springer International Publishing, Cham, S. 207–221, 2017.

[MüRH08] Müller, D.; Reichert, M.; Herbst, J.: A New Paradigm for the Enactment and Dynamic Adaptation of Data-Driven Process Structures. In (Hutchison, D. et al. Hrsg.): Advanced information systems engineering. 20th international conference, CAiSE 2008, Montpellier, France, June 16 - 20, 2008 ; proceedings. Springer, Berlin, S. 48–63, 2008.

[NaAK15] Nagm Aldeen, Y.; Abdel-Fattah, M.; Khedr, A.: A Literature Review of Business Process Modeling Techniques. International Journal of Advanced Research in Computer Science and Software Engineering 3/5, S. 43–47, 2015.

[NaGE14] Nagel, B.; Gerth, C.; Engels, G.: Goal-Driven Composition of Business Process Models. In (Hutchison, D. et al. Hrsg.): Service-Oriented Computing – ICSOC 2013 Workshops. Springer International Publishing, Cham, S. 16–27, 2014.

[NaSS19] Naderifar, V.; Shukur, Z.; Sahran, S.: Distributed Learning Automata Approach for Workflow Mining: Discovering Process Model Using Condensate Drops Method. Journal of Computer Science 11/15, S. 1694–1709, 2019.

[Nobl06] Noble, W. S.: What is a support vector machine? Nature biotechnology 12/24, S. 1565–1567, 2006.

[NSNW13] Niwattanakul, S.; Singthongchai, J.; Naenudorn, E.; Wanapu, S.: Using of Jaccard Coefficient for Keywords Similarity: International Conference on Internet Computing and Web Services, 2013.

[NüRu02] Nüttgens, M.; Rump, F. J.: Syntax und Semantik Ereignisgesteuerter Prozessketten (EPK). In (Desel, J.; Weske, M. Hrsg.): Promise 2002 – Prozessorientierte Methoden und Werkzeuge für die Entwicklung von Informationssystemen. Gesellschaft für Informatik e.V, Bonn, S. 64–77, 2002.

[Ober90] Oberweis, A.: Zeitstrukturen für Informationssysteme. Dissertation. Fakultät für Mathematik und Informatik, Universität Mannheim, Mannheim, 1990.

[Ober96] Oberweis, A.: Modellierung und Ausführung von Workflows mit Petri-Netzen. Vieweg+Teubner Verlag, Wiesbaden, 1996.

[Obje13a] Object Management Group (OMG): Semantics of Business Vocabulary and Business Rules (SBVR). https://www.omg.org/spec/SBVR/1.2/PDF, Stand: 27.12.2022.

[Obje13b] Object Management Group (OMG): Business Process Model and Notation (BPMN) (Version 2.0.2). https://www.omg.org/spec/BPMN/2.0.2/PDF, Stand: 24.01.2022.

[Obje17] Object Management Group (OMG): OMG - Unified Modeling Language (OMG UML) (Version 2.5.1). https://www.omg.org/spec/UML/2.5.1/PDF, Stand: 24.01.2022.

[ObSa96] Oberweis, A.; Sander, P.: Information system behavior specification by high level Petri nets. ACM Transactions on Information Systems 4/14, S. 380–420, 1996.

[OFFD14] Obermeier, S.; Fischer, H.; Fleischmann, A.; Dirndorfer, M.: Geschäftsprozesse realisieren. Ein praxisorientierter Leitfaden von der Strategie bis zur Implementierung. Springer Vieweg, Wiesbaden, 2014.

[OuCJ15] Ou-Yang, C.; Cheng, H.-J.; Juan, Y.-C.: An Integrated mining approach to discover business process models with parallel structures: towards fitness improvement. International Journal of Production Research 13/53, S. 3888–3916, 2015.

[OvBS12] Overhage, S.; Birkmeier, D. Q.; Schlauderer, S.: Quality Marks, Metrics, and Measurement Procedures for Business Process Models. The 3QM-Framework. Business & Information Systems Engineering 4, S. 229–246, 2012.

[ÖzDr17] Özcan, G.; Drescher, A.: Geschäftsprozessmanagement in global verteilten Produktentwicklungsprojekten. In (Eibl, M.; Gaedke, M. Hrsg.): INFORMATIK 2017. Gesellschaft für Informatik, Bonn, S. 849–855, 2017.

[PaSi10] Paradkar, A.; Sinha, A.: Deriving process models from natural language use case models. G06F8/10 Requirements analysis; Specification techniques, 2010.

[Pati06] Patig, S.: Die Evolution von Modellierungssprachen. Frank und Timme, Berlin, 2006.

[PDSP15] Pacini Rabelo, L. A.; Do Prado, A. F.; Souza, W. L. de; Pires, L. F.: An Approach to Business Process Recovery from Source Code: 2015 12th International Conference on Information Technology - New Generations. IEEE, S. 361–366, 2015.

[PéGP11] Pérez-Castillo, R.; Guzmán, I. G.-R. de; Piattini, M.: Business process archeology using MARBLE. Information and Software Technology 10/53, S. 1023–1044, 2011.

[Pele21] Pelekies, A.: EDI/eCommerce: Einführung in den elektronischen Datenaustausch. GS1 Germany GmbH, 2021.

[PeSA07] Pesic, M.; Schonenberg, H.; Aalst, W. M. van der: DECLARE: Full Support for Loosely-Structured Processes: IEEE International Enterprise Distributed Object Computing Conference (EDOC 2007). IEEE, S. 287, 2007.

[Petr62] Petri, C. A.: Kommunikation mit Automaten. Dissertation. Mathematisches Institut der Universität Bonn, Bonn, 1962.

[PFMM08] Petersen, K.; Feldt, R.; Mujtaba, S.; Mattsson, M.: Systematic Mapping Studies in Software Engineering: International Conference on Evaluation and Assessment in Software Engineering (EASE). BCS Learning & Development, 2008.

[PGEM+20] Pérez-Álvarez, J. M.; Gómez-López, M. T.; Eshuis, R.; Montali, M.; Gasca, R. M.: Verifying the manipulation of data objects according to business process and data models. Knowledge and Information Systems 7/62, S. 2653–2683, 2020.

[PiGo04] Pinter, S. S.; Golani, M.: Discovering workflow models from activities' lifespans. Computers in Industry 3/53, S. 283–296, 2004.

[PoDG17] Pourmirza, S.; Dijkman, R.; Grefen, P.: Correlation Miner: Mining Business Process Models and Event Correlations Without Case Identifiers. International Journal of Cooperative Information Systems 02/26, S. 1742002, 2017.

[PoPP17] Potdar, K.; Pardawala, T. S.; Pai, C. D.: A comparative study of categorical variable encoding techniques for neural network classifiers. International Journal of Computer Applications 4/175, S. 7–9, 2017.

[PVGM+11] Pedregosa, F. et al.: Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research 12, S. 2825–2830, 2011.

[PWOB19] Polyvyanyy, A.; Werf, J. M. van der; Overbeek, S.; Brouwers, R.: Information Systems Modeling: Language, Verification, and Tool Support. In (Giorgini, P.; Weber, B. Hrsg.): Advanced Information Systems Engineering. Springer International Publishing, Cham, S. 194–212, 2019.

[QWKL+20] Qian, C. et al.: An Approach for Process Model Extraction By Multi-Grained Text Classification: Advanced Information Systems Engineering. Springer, Cham, S. 269–282, 2020.

[RaPH08] Raj, A.; Prabhakar, T. V.; Hendryx, S.: Transformation of SBVR business design to UML models. In (Shroff, G.; Jalote, P.; Rajamani, S. Hrsg.): Proceedings of the 1st conference on India software engineering conference - ISEC '08. ACM Press, New York, USA, S. 29, 2008.

[Razv09] Razvan, P.: MINING AND MODELING DECISION WORKFLOWS FROM DSS USER ACTIVITY LOGS: Proceedings of the 11th International Conference on Enterprise Information. SCITEPRESS - Science and Technology Publications, S. 144–149, 2009.

[RBFZ20] Rashnavadi, Y.; Behzadifard, S.; Farzadnia, R.; Zamani, S.: Discovering Business Processes from Email Logs using fastText and Process Mining. SSRN Electronic Journal, 2020.

[RDHI08] Redding, G.; Dumas, M.; Hofstede, A. H. ter; Iordachescu, A.: Generating Business Process Models from Object Behavior Models. Information Systems Management 4/25, S. 319–331, 2008.

[RDHM11] Rosa, M. La; Dumas, M.; Hofstede, A. H. ter; Mendling, J.: Configurable multi-perspective business process models. Information Systems 2/36, S. 313–340, 2011.

[ReEl07] Rembert, A. J.; Ellis, C.: Learning the Control-Flow of a Business Process Using ICN-Based Process Models. In (Hutchison, D. et al. Hrsg.): Service-Oriented Computing – ICSOC 2007. Springer, Berlin, Heidelberg, S. 346–351, 2007.

[Reis13] Reisig, W.: Understanding Petri Nets. Modeling Techniques Analysis Methods Case Studies. Springer, Berlin, Heidelberg, 2013.

[Reis86] Reisig, W.: Petrinetze. Eine Einführung. Springer, Berlin, Heidelberg, 1986.

[ReLR11] Reggio, G.; Leotta, M.; Ricca, F.: "Precise is better than light"; a document analysis study about quality of business process models. Workshop on Empirical Requirements Engineering, S. 61–68, 2011.

[ReMR15] Reijers, H. A.; Mendling, J.; Recker, J.: Business Process Quality Management. In (Brocke, J. vom; Rosemann, M. Hrsg.): Handbook on Business Process Management 1. Springer, Berlin, Heidelberg, S. 167–185, 2015.

[RHAM06] Russell, N.; Hofstede, A. H. M. ter; Aalst, W. M. P. van der; Mulyar, N.: Workflow Control-Flow Patterns: A Revised View.: BPM Center Report, 2006.

[RiMa21] Rinderle-Ma, S.; Mangler, J.: Process Automation and Process Mining in Manufacturing. In (Polyvyanyy, A. et al. Hrsg.): Business Process Management. Springer International Publishing, Cham, S. 3–14, 2021.

[Ritt10] Rittgen, P.: Quality and perceived usefulness of process models. In (Shin, S. Y. et al. Hrsg.): Proceedings of the 2010 ACM Symposium on Applied Computing - SAC '10. ACM Press, New York, USA, S. 65, 2010.

[RKMP+11] Rohweder, J. P.; Kasten, G.; Malzahn, D.; Piro, A.; Schmid, J.: Informationsqualität – Definitionen, Dimensionen und Begriffe. In (Hildebrand, K. et al. Hrsg.): Daten- und Informationsqualität. Vieweg+Teubner, Wiesbaden, 23-43, 2011.

[RoAa06] Rozinat, A.; Aalst, W. M. P. van der: Decision mining in ProM: Business Process Management. Springer International Publishing, Cham, S. 420–425, 2006.

[Rose96] Rosemann, M.: Komplexitätsmanagement in Prozeßmodellen. Methodenspezifische Gestaltungsempfehlungen Für Die Informationsmodellierung. Springer Gabler, Wiesbaden, 1996.

[Rous87] Rousseeuw, P. J.: Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. Journal of Computational and Applied Mathematics 20, S. 53–65, 1987.

[RSBR14] Roy, S.; Sajeev, A. S. M.; Bihary, S.; Ranjan, A.: An Empirical Study of Error Patterns in Industrial Business Process Models. IEEE Transactions on Services Computing 2/7, S. 140–153, 2014.

[RVPv+17] Reijers, H. A. et al.: Evaluating data-centric process approaches: Does the human factor factor in? Software & Systems Modeling 3/16, S. 649–662, 2017.

[RySc22] Rybinski, F.; Schüler, S.: Process Discovery Analysis for Generating RPA Flowcharts. In (Marrella, A. et al. Hrsg.): Business Process Management: Blockchain, Robotic Process Automation, and Central and Eastern Europe Forum. Springer International Publishing, Cham, S. 231–245, 2022.

[SACS15] Sarmiento, E.; Almentero, E.; C. S. P. Leite, J.; Sotomayor, G.: Mapping Textual Scenarios to Analyzable Petri-Net Models: Proceedings of the 17th International Conference on Enterprise Information Systems. SCITEPRESS - Science and and Technology Publications, S. 494–501, 2015.

[SaSu16] Sarno, R.; Sungkono, K. R.: Hidden Markov Model for Process Mining of Parallel Business Processes. International Review on Computers and Software (IRECOS) 4/11, S. 290, 2016.

[SBCM+21] Sànchez-Ferreres, J.; Burattin, A.; Carmona, J.; Montali, M.; Padró, L.; Quishpi, L.: Unleashing textual descriptions of business processes. Software & Systems Modeling 6/20, S. 2131–2153, 2021.

[ScAl24] Schüler, S.; Alpers, S.: State of the Art: Automatic Generation of Business Process Models. In (Weerdt, J. de; Pufahl, L. Hrsg.): Business Process Management Workshops. Springer Nature Switzerland, Cham, S. 161–173, 2024.

[Scha10] Schauerte, B.: Excellence in Research in Australia (ERA). http://www.conferenceranks.com/#.

[Schi04] Schimm, G.: Mining exact models of concurrent workflows. Computers in Industry 3/53, S. 265–281, 2004.

[Schü23] Schüler, S.: Results SLR Business Process Model Generation. https://publikationen.bibliothek.kit.edu/1000158547.

[ScSe13] Schmelzer, H. J.; Sesselmann, W.: Geschäftsprozessmanagement in der Praxis. Kunden zufrieden stellen - Produktivität steigern - Wert erhöhen. Hanser, München, 2013.

[SeWG17] Senderovich, A.; Weidlich, M.; Gal, A.: Temporal Network Representation of Event Logs for Improved Performance Modelling in Business Processes. In (Carmona, J.; Engels, G.; Kumar, A. Hrsg.): Business Process Management. Springer International Publishing, Cham, S. 3–21, 2017.

[SFHP+21] Striewe, M. et al.: Kompetenzorientiertes E-Assessment für die grafische, konzeptuelle Modellierung. HMD Praxis der Wirtschaftsinformatik 6/58, S. 1350–1363, 2021.

[SHPR+11] Sein; Henfridsson; Purao; Rossi; Lindgren: Action Design Research. MIS Quarterly 1/35, S. 37, 2011.

[ShSA22] Sholiq, S.; Sarno, R.; Astuti, E. S.: Generating BPMN diagram from textual requirements. Journal of King Saud University - Computer and Information Sciences, 2022.

[ShYa21] Sharma, N.; Yalla, P.: Visualizing UML's Sequence and Class Diagrams Using Graph-Based Clusters. IOP Conference Series: Materials Science and Engineering 1/1074, S. 12010, 2021.

[SiAC13] Si-Said Cherfi, S.; Ayad, S.; Comyn-Wattiau, I.: Improving Business Process Model Quality Using Domain Ontologies. Journal on Data Semantics 2-3/2, S. 75–87, 2013.

[SiGa16] Silva, A. R.; García-Díaz, V.: Integrating Activity- and Goal-Based Workflows: A Data Model Based Design Method. In (Reichert, M.; Reijers, H. A. Hrsg.): Business Process Management Workshops. Springer International Publishing, Cham, S. 352–363, 2016.

[SiPa10] Sinha, A.; Paradkar, A.: Use Cases to Process Specifications in Business Process Modeling Notation: IEEE International Conference on Web Services. IEEE, S. 473–480, 2010.

[ŠkTr13] Škrinjar, R.; Trkman, P.: Increasing process orientation with business process management: Critical practices'. International Journal of Information Management 1/33, S. 48–60, 2013.

[SMAL+19] Steinau, S.; Marrella, A.; Andrews, K.; Leotta, F.; Mecella, M.; Reichert, M.: DALEC: a framework for the systematic evaluation of data-centric approaches to process management software. Software & Systems Modeling 4/18, S. 2679–2716, 2019.

[SmSm77] Smith, J. M.; Smith, D. C. P.: Database abstractions. Aggregation and Generalization. ACM Transactions on Database Systems 2/2, S. 105–133, 1977.

[Somé10] Somé, S. S.: Formalization of textual use case based on Petri Nets. International Journal of Software Engineering and Knowledge Engineering 05/20, S. 695–737, 2010.

[SoMF21] Sommers, D.; Menkovski, V.; Fahland, D.: Process Discovery Using Graph Neural Networks: International Conference on Process Mining (ICPM). IEEE, S. 40–47, 2021.

[SoRG22] Sonbol, R.; Rebdawi, G.; Ghneim, N.: A Machine Translation Like Approach to Generate Business Process Model From Textual Description. SN Computer Science 3/4, 2022.

[SoTS15] Sokolov, K.; Timofeev, D.; Samochadin, A.: Process Extraction from Texts using Semantic Unification: Proceedings of the 7th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management. SCITEPRESS - Science and and Technology Publications, S. 254–259, 2015.

[SpAc80] Speck, J.; Acham, K.: Handbuch wissenschaftstheoretischer Begriffe. Vandenhoeck & Ruprecht, Göttingen, 1980.

[Spar72] Sparck Jones, K.: A Statistical Interpretation of Term Specificity and Its Application in Retrieval. Journal of Documentation 1/28, S. 11–21, 1972.

[SRSP+14] Sawant, K. P.; Roy, S.; Sripathi, S.; Plesse, F.; Sajeev, A. S. M.: Deriving requirements model from textual use cases. In (Jalote, P.; Briand, L.; Hoek, A. van der Hrsg.): Companion Proceedings of the 36th International Conference on Software Engineering. ACM, New York, USA, S. 235–244, 2014.

[SSFM08] Stoitsev, T.; Scheidl, S.; Flentge, F.; Mühlhäuser, M.: From Personal Task Management to End-User Driven Business Process Modeling. In (Dumas, M.; Reichert, M.; Shan, M.-C. Hrsg.): Business Process Management. Springer, Berlin, Heidelberg, S. 84–99, 2008.

[Stac73] Stachowiak, H.: Allgemeine Modelltheorie. Springer, Wien, 1973.

[STAM+19] Soares Silva, T.; Toralles Avila, D.; Ampos Flesch, J.; Marques Peres, S.; Mendling, J.; Thom, L. H.: A Service-Oriented Architecture for Generating Sound Process Descriptions: IEEE 23rd International Enterprise Distributed Object Computing Conference (EDOC). IEEE, S. 1–10, 2019.

[Star90] Starke, P. H.: Analyse von Petri-Netz-Modellen. Vieweg+Teubner Verlag, Wiesbaden, 1990.

[StDT19] Stein Dani, V.; Dal Sasso Freitas, C. M.; Thom, L. H.: Ten years of visualization of business process models: A systematic literature review. Computer Standards & Interfaces 66, S. 103347, 2019.

[StFL19] Stark, R.; Fresemann, C.; Lindow, K.: Development and operation of Digital Twins for technical systems and services. CIRP Annals 1/68, S. 129–132, 2019.

[SuBa16a] Sun, Y.; Bauer, B.: A Novel Fitness Improvement Method for Mined Business Process Models. In (Espana, S.; Ivanovic, M.; Savic, M. Hrsg.): Advanced Information Systems Engineering. CAiSE. Springer, Berlin, Heidelberg, S. 25–32, 2016.

[SuBa16b] Sun, Y.; Bauer, B.: A Graph and Trace Clustering-based Approach for Abstracting Mined Business Process Models: Proceedings of the 18th International Conference on Enterprise Information Systems. SCITEPRESS - Science and and Technology Publications, S. 63–74, 2016.

[SuRS19] Sungkono, K.; Rochmah, U.; Sarno, R.: Heuristic Linear Temporal Logic Pattern Algorithm in Business Process Model. International Journal of Intelligent Engineering and Systems 4/12, S. 31–40, 2019.

[SWMW12] Smirnov, S.; Weidlich, M.; Mendling, J.; Weske, M.: Action patterns in business process model repositories. Computers in Industry 2/63, S. 98–111, 2012.

[SZNS06] Sun, S. X.; Zhao, J. L.; Nunamaker, J. F.; Sheng, O. R. L.: Formulating the Data-Flow Perspective for Business Process Management. Information Systems Research 4/17, S. 374–391, 2006.

[TaAk14] Tantan, O. C.; Akoka, J.: Automated transformation of business rules specification to business process model. Proceedings of the Twenty-Sixth International Conference on Software Engineering and Knowledge Engineering, S. 684–687, 2014.

[TaHo21] Takei, T.; Horita, H.: Towards Goal-Oriented Business Process Model Repair: 2021 10th International Congress on Advanced Applied Informatics (IIAI-AAI). IEEE, S. 691–696, 2021.

[TaSS22] Tangkawarow, I.; Sarno, R.; Siahaan, D.: ID2SBVR: A Method for Extracting Business Vocabulary and Rules from an Informal Document. Big Data and Cognitive Computing 4/6, S. 119, 2022.

[TGPV21] Tsakalidis, G.; Georgoulakos, K.; Paganias, D.; Vergidis, K.: An Elaborate Preprocessing Phase (p3) in Composition and Optimization of Business Process Models. Computation 2/9, S. 16, 2021.

[Thal22] Thalheim, B.: Models: the fourth dimension of computer science. Software & Systems Modeling 21, S. 9–18, 2022.

[TiMi19] Tikhonov, S. E.; Mitsyuk, A. A.: A Method to Improve Workflow Net Decomposition for Process Model Repair. In (Aalst, W. M. P. et al. Hrsg.): Analysis of Images, Social Networks and Texts. Springer International Publishing, Cham, S. 411–423, 2019.

[TiWH01] Tibshirani, R.; Walther, G.; Hastie, T.: Estimating the Number of Clusters in a Data Set Via the Gap Statistic. Journal of the Royal Statistical Society Series B: Statistical Methodology 2/63, S. 411–423, 2001.

[TrDS03] Tranfield, D.; Denyer, D.; Smart, P.: Towards a Methodology for Developing Evidence-Informed Management Knowledge by Means of Systematic Review. British Journal of Management 3/14, S. 207–222, 2003.

[UFHP+21] Ullrich, M. et al.: Platform Architecture for the Diagram Assessment Domain. In (Götz, S. et al. Hrsg.): Proceedings of the Software Engineering. Gesellschaft für Informatik e.V., S. 22–26, 2021.

[Vett98] Vetter, M.: Objektmodellierung. Eine Einführung in die objektorientierte Analyse und das objektorientierte Design. Teubner, Stuttgart, 1998.

[ViLa22] Vinogradova, M. V.; Larionov, A. S.: Analyzing of User Actions for the Business Process Models Mining and Automated Building of the Knowledge Base of a Company. In (Kravets, A. G.; Bolshakov, A. A.; Shcherbakov, M. Hrsg.): Society 5.0: Human-Centered Society Challenges and Solutions. Springer International Publishing, Cham, S. 395–406, 2022.

[VoHa18] Vonhof, C.; Haas-Betzwieser, E.: Praxishandbuch Prozessmanagement in Bibliotheken und Informationseinrichtungen. De Gruyter, Berlin, Boston, 2018.

[VöSB13] Völter, M.; Stahl, T.; Bettin, J.: Model-Driven Software Development. Technology, Engineering, Management. Wiley, s.l., 2013.

[VoSY98] Vogler, W.; Semenov, A.; Yakovlev, A.: Unfolding and finite prefix for nets with read arcs. In (Goos, G. et al. Hrsg.): CONCUR'98 Concurrency Theory. Springer, Berlin, Heidelberg, S. 501–516, 1998.

[VRAV10] Vanderfeesten, I.; Reijers, H. A.; Aalst, W. M. P.; Vogelaar, J.: Automatic Support for Product Based Workflow Design: Generation of Process Models from a Product Data Model. In (Meersman, R.; Dillon, T.; Herrero, P. Hrsg.): On the Move to Meaningful Internet Systems: OTM 2010 Workshops. Springer, Berlin, Heidelberg, S. 665–674, 2010.

[WAHD22] Wright, A.; Andrews, H.; Hutton, B.; Dennis, G.: JSON Schema. XPath for JSONA Media Type for Describing JSON Documents. Internet Engineering Task Force, 2022.

[WaSt96] Wang, R. Y.; Strong, D. M.: Beyond Accuracy: What Data Quality Means to Data Consumers. Journal of Management Information Systems 4/12, S. 5–33, 1996.

[WBVB12] Weerdt, J. de; Backer, M. de; Vanthienen, J.; Baesens, B.: A multi-dimensional quality assessment of state-of-the-art process discovery algorithms using real-life event logs. Information Systems 7/37, S. 654–676, 2012.

[WeBC15] Weerdt, J. de; Broucke, S. K. L. M. vanden; Caron, F.: Bidimensional Process Discovery for Mining BPMN Models. In (Fournier, F.; Mendling, J. Hrsg.): Business Process Management Workshops. Springer International Publishing, Cham, S. 529–540, 2015.

[Weit98] Weitz, W.: SGML nets: integrating document and workflow modeling: Proceedings of the Thirty-First Hawaii International Conference on System Sciences. IEEE Comput. Soc, S. 185–194, 1998.

[Wesk19] Weske, M.: Business Process Management. Concepts, Languages, Architectures. Springer, Berlin, 2019.

[WeWa02] Webster, J.; Watson, R. T.: Analyzing the Past to Prepare for the Future: Writing a Literature Review. MIS Quarterly 2/26, S. xiii–xxiii, 2002.

[WiKL18] Wiśniewski, P.; Kluza, K.; Ligęza, A.: An Approach to Participatory Business Process Modeling: BPMN Model Generation Using Constraint Programming and Graph Composition. Applied Sciences 9/8, S. 1428, 2018.

[WKKL19] Wiśniewski, P.; Kluza, K.; Kucharska, E.; Ligęza, A.: Spreadsheets as Interoperability Solution for Business Process Representation. Applied Sciences 2/9, S. 345, 2019.

[WLXZ+] Wang, S.; Liu, Y.; Xu, Y.; Zhu, C.; Zeng, M.: Want To Reduce Labeling Cost? GPT-3 Can Help. Findings of the Association for Computational Linguistics, S. 4195–4205.

[Work96] Workflow Management Coalition (WMC): Workflow Management Coalition Terminology and Glossary. WFMC-TC-1011, 1996.

[Worl04] World Wide Web Consortium (W3C): XML Schema Part 2: Datatypes. https://www.w3.org/TR/xmlschema-2/, Stand: 15.12.2023.

[WRPM22] Werf, J. M. E. M. van der; Rivkin, A.; Polyvyanyy, A.; Montali, M.: Data and Process Resonance. In (Bernardinello, L.; Petrucci, L. Hrsg.): Application and Theory of Petri Nets and Concurrency. Springer International Publishing, Cham, S. 369–392, 2022.

[WSLZ19] Wang, N.; Sun, S.; Liu, Y.; Zhang, S.: Business Process Model Abstraction Based on Fuzzy Clustering Analysis. International Journal of Cooperative Information Systems 03/28, S. 1950007, 2019.

[WYHX11] Wu, Z.; Yao, S.; He, G.; Xue, G.: Rules Oriented Business Process Modeling: 2011 International Conference on Internet Technology and Applications. IEEE, S. 1–4, 2011.

[YaCC19] Yanuarifiani, A. P.; Chua, F.-F.; Chan, G.-Y.: Automating Business Process Model Generation from Ontology-based Requirements: Proceedings of the 2019 8th International Conference on Software and Computer Applications. ACM, New York, USA, S. 205–209, 2019.

[YuBL10] Yue, T.; Briand, L. C.; Labiche, Y.: An Automated Approach to Transform Use Cases into Activity Diagrams. In (Hutchison, D. et al. Hrsg.): Modelling Foundations and Applications. Springer, Berlin, Heidelberg, S. 337–353, 2010.

[YZCD+17] Yang, S. et al.: Medical Workflow Modeling Using Alignment-Guided State-Splitting HMM. IEEE International Conference on Healthcare Informatics. IEEE International Conference on Healthcare Informatics 2017, S. 144–153, 2017.

[Zimm99] Zimmermann, V.: Objektorientiertes Geschäftsprozessmanagement. Deutscher Universitätsverlag, Wiesbaden, 1999.

[ZMSS18] Zhang, Y.; Martikainen, O.; Saikkonen, R.; Soisalon-Soininen, E.: Extracting Service Process Models from Location Data. In (Ceravolo, P.; Guetl, C.; Rinderle-Ma, S. Hrsg.): Data-Driven Process Discovery and Analysis. Springer International Publishing, Cham, S. 78–96, 2018.

A. Anhang: Betreute Abschlussarbeiten

Im Zusammenhang mit dem Forschungsvorhaben wurden verschiedene verwandte Themen auch im Rahmen betreuter Abschlussarbeiten aufgegriffen und vertieft. Sofern Ergebnisse aus diesen Abschlussarbeiten direkt in die Dissertation eingeflossen sind, wird dies an der entsprechenden Stelle explizit vermerkt.

[Dien23]	Diener, David Philipp: Data model extraction through adaption of business process models, Masterarbeit, Karlsruhe: Karlsruher Institut für Technologie, 2023
[Dobl21]	Dobler, Rafael: Automatisierte Prozessmodellerstellung mit Methoden des maschinellen Lernens, Bachelorarbeit, Karlsruhe: Karlsruher Institut für Technologie, 2021
[Emme22]	Emmert, Niklas Jörg: Literaturstudie zu Automatisierungskonzepten in der Modellierung, Bachelorarbeit, Karlsruhe: Karlsruher Institut für Technologie, 2022
[Imho23]	Imhof, Pascal: Automatische Generierung von Prozessmodellen basierend auf Objektinstanzen, Bachelorarbeit, Karlsruhe: Karlsruher Institut für Technologie, 2023
[Kara20]	Karatay, Barkin: Entwicklung eines formalen Modells zur Umsetzung eines automatisierten Ablaufs in einem Smart Home, Bachelorarbeit, Karlsruhe: Karlsruher Institut für Technologie, 2020
[Kuhn22]	Kuhn, Julia Anna-Maria: Prozessmodellgenerierung im Kontext der Nutzung mobiler Endgeräte, Masterarbeit, Karlsruhe: Karlsruher Institut für Technologie, 2024
[Ralp24]	Ralph, Jacob: Informationsextraktion zur Geschäftsprozessmodellierung, Bachelorarbeit, Karlsruhe: Karlsruher Institut für Technologie, 2024
[Reim23]	Reimann de la Cruz, Nico: Datenanalyse für die Geschäftsprozessmodellierung Bachelorarbeit, Karlsruhe: Karlsruher Institut für Technologie, 2023
[Rose20]	Rosenberg, Alexey: Automated generation of XML nets for modelling business processes, Bachelorarbeit, Karlsruhe: Karlsruher Institut für Technologie, 2020
[Rybi21]	Rybinsky, Fabian: Automatisierte Process Discovery für die Geschäftsprozessmodellerstellung im Rahmen von RPA, Masterarbeit, Karlsruhe: Karlsruher Institut für Technologie, 2021
[Yara23]	Yarashuk, Uladzislau: Process Mining Methoden zur XML-Netz-Generierung, Bachelorarbeit, Karlsruhe: Karlsruher Institut für Technologie, 2023

B. Anhang: Suchstring-Anfragen

Die systematische Literaturrecherche wurde mit folgender finaler Suchstrategie durchgeführt:

Titel-Kriterien:	○ "business process*" OR "workflow*" ○ "model*" OR "diagram*" OR "flowchart*" OR "graph*" OR "declarat*"
Abstract-Kriterien:	○ discovery OR learning OR generat* ○ (automat* OR algorithm* OR Script* OR program*) AND (construct* OR support OR adapt* OR correct*)

Diese Suchstrategie wurde in den folgenden wissenschaftlichen Datenbanken angewandt:

- Scopus: 1.969 Ergebnisse
- IEEE Xplore: 877 Ergebnisse
- ACM Digital Library: 111 Ergebnisse
- ScienceDirect: 501 Ergebnisse
- Web of Science: 444 Ergebnisse
- DBLP: 259 Ergebnisse
- Google Scholar: 71 Ergebnisse

Die deutsche Suche verwendete entsprechende Übersetzungen der Suchbegriffe:

Titel-Kriterien:	○ "Geschäftsprozess*" OR "Workflow*" ○ "Modell*" OR "Diagramm*" OR "Flowchart*" OR "Graph*" OR "Deklarat*"
Abstract-Kriterien:	○ Discovery OR Learning OR Generier* ○ (Automat* OR Algorithmus* OR Skript* OR Program*) AND (Erstellen* OR Unterstützen OR Anpassen* OR Verbessern*)

Die Recherche wurde auf deutsch- und englischsprachige Publikationen beschränkt. Nach Entfernung von Duplikaten und initialer Sichtung verblieben 3.617 potenziell relevante Arbeiten für die detaillierte Analyse.

C. Anhang: Resultate der LR

Tabelle 9-1: Ansätze basierend auf Eventlogs

Kürzel	Titel	Modellierungs-sprache
[ChOJ15]	A hybrid approach to extract business process models with high fitness and precision	Petri-Netz
[AgHE20]	A new method for organizational process model discovery through the analysis of workflows and data exchange networks	Petri-Netz
[HuXH11]	A novel approach for mining stochastic process model from workflow logs	Petri-Netz
[SuBa16a]	A Novel Fitness Improvement Method for Mined Business Process Models	Direkt-Folge-Graph
[LiFe06a]	An automatic Business Process Modeling Method Based on Markov Transition Matrix	Petri-Netz
[Lamg21]	An Integrated Approach for Discovering Process Models According to Business Process Types	Petri-Netz (PM)
[OuCJ15]	An integrated mining approach to discover business process models with parallel structures: towards fitness improvement	Petri-Netz
[GrMM18]	Applying process mining techniques and neural networks to creating and assessment of business process models	BPMN
[WeBC15]	Bidimensional Process Discovery for Mining BPMN Models	BPMN
[GoRA19]	Connecting databases with process mining: a meta model and toolset	Petri-Netz (PM)
[HUCM19]	Constructing workflow models of alarm responses via trace labeling and dependency analysis	Kausales Netz
[GDLW+14]	Controlled automated discovery of collections of business process models	BPMN
[PoDG17]	Correlation Miner: Mining Business Process Models and Event Correlations Without Case Identifiers	Direkt-Folge-Graph
[KsKo21]	Data-Driven Modeling of Complex Business Process in Heterogeneous Environment of Healthcare Organization with Health Information Systems	Direkt-Folge-Graph
[LiFe06b]	Design of an Automatic Workflow Modeling Method in Cooperative WFMS	Petri-Netz
[KuBV10]	Discovering Business Process Model from	Direkt-Folge-

Kürzel	Titel	Modellierungs-sprache
	Unstructured Activity Logs	Graph
[GGPS06]	Discovering expressive process models by clustering log traces	Eigenes Graph-basiertes Modell
[LYCT+18]	Discovering interpretable medical workflow models	Direkt-Folge-Graph
[CDLW04]	Discovering models of behavior for concurrent workflows	Petri-Netze
[AaBe20]	Discovering Object-centric Petri Nets	Petri-Netze
[PiGo04]	Discovering workflow models from activities' lifespans	Direkt-Folge-Graph
[NaSS19]	Distributed Learning Automata Approach for Workflow Mining: Discovering Process Model Using Condensate Drops Method	Direkt-Folge-Graph
[BeAa20]	Extracting Multiple Viewpoint Models from Relational Databases	Multiple Viewpoint (MVP)-Modell
[LCZW22]	Formal Modeling and Discovery of Hierarchical Business Processes: A Petri Net-Based Approach	Petri-Netze
[GoPi03]	Generating a Process Model from a Process Audit Log	Direkt-Folge-Graph
[LeMS21]	Generating block-structured parallel process models by demonstration	Direkt-Folge-Graph
[HGAE+22]	Graph-based managing and mining of processes and data in the domain of intellectual property	Direkt-Folge-Graph
[SuRS19]	Heuristic Linear Temporal Logic Pattern Algo-rithm in Business Process Model	Eigenes Graph-basiertes Modell
[SaSu16]	Hidden Markov Model for Process Mining of Parallel Business Processes	Kausales Netz
AMFG17	Learning hybrid process models from events process discovery without faking confidence	Petri-Netze
[ReEl07]	Learning the Control-Flow of a Business Process Using ICN-Based Process Models	Eigenes Graph-basiertes Modell
[LiDi13]	Learning workflow models from event logs using co-clustering	Petri-Netze
[DMSZ+12]	MailOfMine – Analyzing Mail Messages for Mining Artful Collaborative Processes	Eigenes Graph-basiertes Modell
[YZCD+17]	Medical Workflow Modeling Using Alignment-Guided State-Splitting HMM	Petri-Netze
[Razv09]	Mining and modeling decision workflows from DSS user activity logs	Direkt-Folge-Graph
[Schi04]	Mining exact models of concurrent workflows	directed acyclic graph
[AgGL98]	Mining process models from workflow logs	directed acyclic graph

Kürzel	Titel	Modellierungs-sprache
[FoGP15]	On the Discovery of Explainable and Accurate Behavioral Models for Complex Lowly-structured Business Processes	Multi-variant process model
[LGXX+14]	Proactive workflow modeling by stochastic processes with application to healthcare operation and management	Direkt-Folge-Graph
[SoMF21]	Process Discovery Using Graph Neural Networks	Petri-Netze
[LGHH+16]	Process mining with token carried data	Petri-Netze
[KITd14]	Re-learning of Business Process Models from Legacy System Using Incremental Process Mining	Direkt-Folge-Graph
[SeWG17]	Temporal Network Representation of Event Logs for Improved Performance Modelling in Business Processes	Temporal Network Representation
[EfSa20]	Time-based α+ miner for modelling business processes using temporal pattern	Petri-Netze
[CWFL+19]	Workflow Model Mining Based On Educational Management Data Logs	Petri-Netze
[LZLJ12]	Workflow simulation for operational decision support using event graph through process mining	Direkt-Folge-Graph

Tabelle 9-2: Ansätze basierend auf natürlichsprachlichen Text

Kürzel	Titel	Input	Output
[FlKM05]	From textual scenarios to a conceptual schema	Semi-strukturierter Text (Geschäftsregeln)	Klagenfurt Conceptual Predesign model (KCPM), UML-Aktivitätsdiagramm
[SACS15]	Mapping Textual Scenarios to Analyzable Petri-Net Models	semi-strukturierter Text (Szenario-Anforderungen)	Petri-Netz
[Somé10]	Formalization of textual use case based on Petri Nets	Semi-strukturierter Text (Use Case)	Petri-Netz
[SRSP+14]	Deriving requirements model from textual use cases	Semi-Strukturierter Text (Use Case)	BPMN-Modell
[GNEM+08]	Visualization of Use Cases through Automatically Generated Activity Diagrams	Semi-strukturierter Text (Use-Case-Beschreibung)	UML-Aktivitätsdiagrammen
[BaEK20]	Business Process Modelling Augmented: Model Driven transformation of User Stories to Processes	Semi-strukturierter Text (Use-Case-Beschreibung)	BPMN-Modell
[Capo16]	Geschäftsprozessmodellierung mit kontrollierter natürlicher Sprache	Semi-strukturierter Text aus Satzschablonen (Prozessbeschreibung)	BPMN-Modell
[HoKW18]	A concept for generating business process models from natural language description	Text	Tabelle, BPMN-Modell
[SoRG22]	A Machine Translation Like Approach to Generate Business Process Model From Textual Description	Text	TextGraph, BPMN-Modell

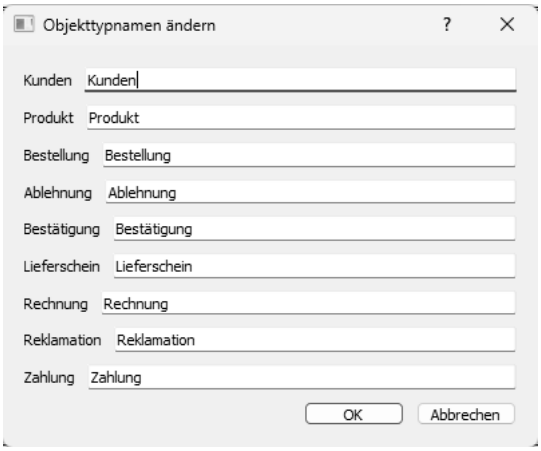
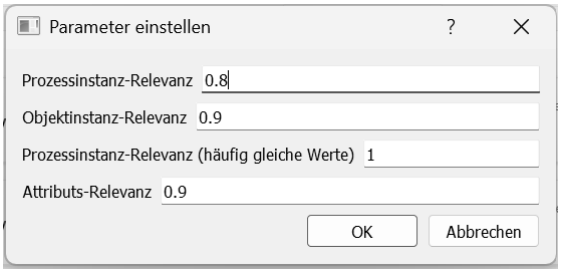
Kürzel	Titel	Input	Output
[JaLi21]	iMER: Iterative process of entity relationship and business process model extraction from the requirements	Text (Anforderungen)	ER-Diagrammen, Business Process Modell (eigen)
[KaMA15]	Automated transformation approach from user requirement to behavior design	Text (Anforderungen)	UML-Aktivitätsdiagramm
[MaAb21]	Generating UML Use Case and Activity Diagrams Using NLP techniques and Heuristics Rules	Text (Anforderungen)	UML-Aktivitätsdiagramm
[ShSA22]	Generating BPMN diagram from textual requirements	Text (Anforderungen)	Tabelle, BPMN-Modell
[SiPa10] Patent: [PaSi10]	Use Cases to Process Specifications in Business Process Modeling Notation	Text (Anforderungen)	BPMN-Modell
[QWKL+20]	An Approach for Process Model Extraction by Multi-grained Text Classification	Text (Kochrezepte)	Erweiterter Prozessstrukturbaum
[ABMN20]	Say It in Your Own Words: Defining Declarative Process Models Using Speech Recognition	Text (Prozessbeschreibung)	Declare
[ACLR19]	Extracting Declarative Process Models from Natural Language	Text (Prozessbeschreibung)	Declare
[AcVo13]	mo-del[NL]generation: natural language model extraction	Text (Prozessbeschreibung)	Beliebiges Modell durch Ergänzung eines Metamodels
[BMMS17]	Project EVER:	Text (Prozessbe-	Workflow

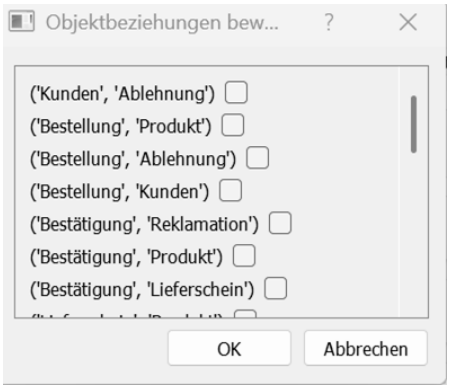
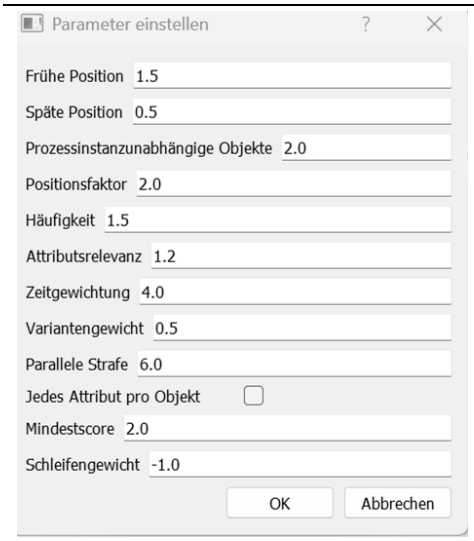
Kürzel	Titel	Input	Output
	Extraction and Processing of Procedural Experience Knowledge in Workflows	schreibung)	
[EMHD+15]	Automatic process model discovery from textual methodologies	Text (Prozessbeschreibung)	BPMN-Modell
[FrMP11]	Process model generation from natural language text	Text (Prozessbeschreibung, sequenziell)	BPMN-Modell
[STAM+19]	A Service-Oriented Architecture for Generating Sound Process Descriptions	Text (Prozessbeschreibungen, sequenziell)	BPMN-Modell, Text (Prozessbeschreibung)
[GoSB11]	Let Me Tell You a Story - On How to Build Process Models	Text (Prozessbeschreibungen als User Stories)	BPMN-Modellfragmenten
[HHMD+20]	A-BPS: Automatic Business Process Discovery Service using Ordered Neurons LSTM	Text (Prozessbeschreibungen)	BPMN-Modell
[LWZZ10]	A policy-based process mining framework: Mining business policy texts for discov-ering process models	Text (Richtlinien-Dokumente)	BPMN- Prozesskomponenten
[GuDe17]	From organizational guidelines to business process models: Exploratory case for an ontology based methodology	Text (Unternehmensrichtlinien)	Ontologie, EPK-Modell
[YuBL10]	An Automated Approach to Transform Use Cases into Activity Diagrams	Text (Use Case)	UML-Aktivitätsdiagramm
[DiJZ16]	Generating Petri	Text (Use-Case-	Petri-Netz

Kürzel	Titel	Input	Output
	Net-Based Behavioral Models From Textual Use Cases and Application in Railway Networks	Beschreibung)	
[FTOA+17]	Assisting Process Modeling by Identifying Business Process Elements in Natural Language Texts	Texte (Formulare, Normen, Ereignisdaten von Informationssystemen, E-Mail-Nachrichten (in einzelnen Sätzen))	BPMN-Modell
[GhKC07]	Process Discovery from Model and Text Artefacts	Texte (Memos, Handbücher, Anforderungsdokumente, Entwurfsdokumente, Auftrags-/Visionserklärungen, Sitzungsprotokolle) und Modelle	Proto-Modelle
[DiMe13]	A two-step fast algorithm for the automated discovery of declarative workflows		

D. Anhang: Iterative Vorgehensweise

Tabelle 9-3: Iterative Vorgehensweise für die Generierungsschritte

Generierungsschritt	Screenshot
Festlegen des Schwellwerts für die Prozessinstanzzuordnung	
Umbenennung der Objekttypen umbenennen	
Festlegen der Schwellwerte für die Objektbeziehungstypen-Generierung	

Generierungsschritt	Screenshot
Erhöhung der Gewichtung relevanter Objektbeziehungstypen (für die Aktivitätstyp-Bewertung)	
Festlegen der Gewichtungsfaktoren für die Aktivitätstyp-Bewertung	
Umbenennung der Aktivitätstypen