ANDREAS DRESCHER

Musterbasierte Kontrollflusssemantik für Geschäftsprozess-modellierungssprachen

KIT Scientific Publishing

Andreas Drescher

Musterbasierte Kontrollflusssemantik für Geschäftsprozessmodellierungssprachen

Musterbasierte Kontrollflusssemantik für Geschäftsprozessmodellierungssprachen

von
Andreas Drescher

KIT Scientific Publishing

Dissertation, Karlsruher Institut für Technologie
KIT-Fakultät für Wirtschaftswissenschaften

Tag der mündlichen Prüfung: 12. Februar 2019
Erster Gutachter: Prof. Dr. Andreas Oberweis
Zweiter Gutachter: Prof. em. Dr. Dr.h.c. Wolffried Stucky

Impressum

Karlsruher Institut für Technologie (KIT)
KIT Scientific Publishing
Straße am Forum 2
D-76131 Karlsruhe

www.ksp.kit.edu

Print on Demand 2019 – Gedruckt auf FSC-zertifiziertem Papier

ISBN 978-3-7315-0913-4
DOI 10.5445/KSP/1000091876

Musterbasierte Kontrollflusssemantik für Geschäftsprozessmodellierungssprachen

zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften

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

genehmigte

Dissertation

von

M. Sc. Andreas Drescher

aus Schwalmstadt

Tag der mündlichen Prüfung: 12.02.2019
Erster Gutachter: Prof. Dr. Andreas Oberweis
Zweiter Gutachter: Prof. em. Dr. Dr.h.c. Wolffried Stucky

Kurzfassung

Geschäftsprozessmodelle bilden eine wichtige Grundlage für das Geschäftsprozessmanagement. Damit können die Geschäftsprozesse dokumentiert, analysiert, überwacht und gesteuert werden. In der Forschung und Praxis werden zahlreiche Modellierungssprachen, wie z. B. Ereignisgesteuerte Prozessketten (EPK), Business Process Model and Notation (BPMN) oder UML-Aktivitätsdiagramme, vorgeschlagen und eingesetzt. Die meisten in der Praxis eingesetzten Geschäftsprozessmodellierungssprachen besitzen aber keine präzise Kontrollflusssemantik. Durch den Kontrollfluss wird die Ablaufreihenfolge der Elemente im Geschäftsprozessmodell festgelegt. Unerwünschte Interpretationsspielräume ergeben sich immer dann, wenn die Kontrollflusssemantik unpräzise definiert ist. Dies ist beispielsweise der Fall, wenn die Bedeutung einzelner Symbole nicht eindeutig festgelegt ist. Missverständnisse zwischen Domänenexperten und Modellierern werden durch unpräzise Beschreibungen gefördert. Mehrdeutigkeiten erschweren bzw. verhindern die Simulation, Analyse, Prozessüberwachung und Verbesserung der Geschäftsprozesse. Kontrollflussmuster sind Schablonen zur Beschreibung der Ablaufreihenfolge der Elemente im Geschäftsprozessmodell und können als Hilfsmittel für die Modellierung eingesetzt werden, um so komplexe Abläufe aus einfachen Grundstrukturen zusammenzusetzen. In dieser Arbeit wird eine auf Kontrollflussmustern basierende Methode zur Beschreibung der Kontrollflusssemantik von graphischen Geschäftsprozessmodellierungssprachen vorgestellt. Auf dieser Grundlage kann die Ablaufreihenfolge der Elemente in den Geschäftsprozessmodellen präzise beschrieben werden. Darüber hinaus werden die effiziente Anwendbarkeit und grundsätzliche Nützlichkeit der Methode sowie Möglichkeiten der Analyse der Geschäftsprozessmodelle mit einem Softwarewerkzeug aufgezeigt.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

7PMG	Seven Process Modeling Guidelines
AAL	Ambient Assisted Living
AIFB	Angewandte Informatik und Formale Beschreibungsverfahren
AdaWR	Adapt While Running
ADOxx	ActiveXX Data Objects Extension
ANSI	American National Standards Institute
AO4BPMN	Aspect-Oriented Business Process Modeling
APEL	Abstract Process Engine Language
ARIS	Architektur integrierter Informationssysteme
BPM	Business Process Management
BPML	Business Process Modeling Language
BPMN	Business Process Model and Notation
BPMN4CP	BPMN for Clinical Pathways
BPSS	Business Process Specification Schema
bspw.	beispielsweise
bzw.	beziehungsweise
CIMOSA	Computer Integrated Manufacturing Open System Architecture
CMMN	Case Management Model and Notation
CompM	Compose Model

ConCM	Configure Configurable Model
ConfED	Check Conformance using Event Data
CPN	Colored Petri Nets
CSV	Comma Separated Values
DesCM	Design Configurable Model
DesM	Design Model
DiscM	Discover Model from Event Data
DMN	Decision Model and Notation
d. h.	das heißt
EBNF	Erweiterte Backus-Naur-Form
ebXML	electronic business using XML
EDOC	Enterprise Distributed Object Computing
eEPK	erweiterte Ereignisgesteuerte Prozesskette
EnM	Enact Model
EPC	Event driven Process Chain
EPC+C	Event driven Process Chain + Controls
EPK	Ereignisgesteuerte Prozesskette
EPML	Event Driven Process Chain Markup Language
ER	Entity Relationship
etc.	et cetera
EWF-Netz	Erweitertes Workflow-Netz
ExtM	Extend Model

e. V.	eingetragener Verein
FIFO	First In First Out
GERT	Graphical Evaluation and Review Technique
ggf.	gegebenenfalls
GLIF	Guideline Interchange Format
GoM	Grundsätze ordnungsgemäßer Modellierung
GPN	Generalised Process Networks
G-CTL	Graphical Computation Tree Logic
HCM-L	Human Cognitive Model Language
IDEF	Integrated Definition
IM	Interaktionsmodell
ImpM	Improve Model
ISO	International Organization for Standardization
ISOMetrics	International Organization for Standardization Metrics
ISONorm	International Organization for Standardization Norm
KIT	Karlsruher Institut für Technologie
LIFO	Last In First Out
LogED	Log Event Data
LOVeM	Line of Visibility Engineering Methodology
meCUE	modular evaluation of key Components of User Experience
MerCM	Merge Models into Configurable Model
MerM	Merge Model

MOF	Meta Object Facility
Mon	Monitor
MS	Microsoft
NA	no answer
newYAWL	new Yet Another Workflow Language
n/a	not available
OCL	Object Constraint Language
OEP	Dienstleistungseinheit Organisationsentwicklung und Prozesse
oEPK	objektorientierte Ereignisgesteuerte Prozesskette
OMT	Object Modeling Technique
OMG	Object Management Group
OOSE	Object-Oriented Software Engineering
PN	Petri-Netz
RAD	Role Activity Diagram
RefM	Refine Model
RepM	Repair Model
PDF	Portable Document Format
PerfED	Analyze Performance using Event Data
PerfM	Analyze Performance based on Model
PMG	Process Modeling Guidelines
PML	Process Modeling Language
PMNL	Petri-Netz Markup Language

PN	Petri-Netz
ROI	Return on Investment
SAGE	Standards-based Active Guideline Environment
SelM	Select Model from Collection
SLANG	Spade Language
UEQ	User Experience Questionnaire
UID	Unique Identifier
UML	Unified Modeling Language
UML-AD	Unified Modeling Language - Aktivitätsdiagramm
USE	Usefulness, Satisfaction and Ease of use
u. a.	und anderes / unter anderem
VerM	Verify Model
vgl.	vergleiche
YAWL	Yet Another Workflow Language
wcp	Workflow Control Pattern
WebML	Web Modeling Language
WfMC	Workflow Management Coalition
WS-BPEL	Web Service - Business Process Execution Language
XLS	Microsoft Excel Spreadsheet
XPDL	XML Process Definition Language
XMI	XML Metadata Interchange
XML	Extensible Markup Language
z. B.	zum Beispiel

1 Einleitung

Die Wertschöpfung eines Unternehmens ist von der effizienten und effektiven Ausführung der Geschäftsprozesse abhängig [ŠkTr13, HVSS+15]. Dementsprechend sollten zur Wahrung der Wettbewerbsfähigkeit die Geschäftsprozesse kontinuierlich analysiert und gegebenenfalls angepasst und somit verbessert werden. Verbesserungen können beispielsweise Kostensenkungen, Verbesserungen der Durchlaufzeiten oder auch die Steigerung der Produktqualität und Kundenzufriedenheit sein [Harm16, BBSG+16]. Die notwendigen Anforderungen und die daraus resultierende Problemstellung für die Analyse der Geschäftsprozesse werden im Folgenden erörtert, um darauf aufbauend die Zielsetzung und die damit einhergehende Struktur der Arbeit zu motivieren.

1.1 Ausgangssituation

Für die Analyse der Geschäftsprozesse und einer möglicherweise damit einhergehenden Anpassung dieser Prozesse kann zur strukturierten Vorgehensweise das Geschäftsprozessmanagement eingesetzt werden. Hierbei werden alle relevanten Aspekte von der Erhebung und Dokumentation der Geschäftsprozesse, über die Analyse, Konzeption und Umsetzung bis hin zur Überwachung betrachtet [FrRü17, Wesk12]. Während der Dokumentation werden die Geschäftsprozesse durch Modelle erfasst. Die Modellierungssprachen dienen dazu, die möglichen Ausführungsvarianten der Geschäftsprozesse als Blaupause zu dokumentieren [Wesk12]. Allerdings werden in der Praxis oftmals Geschäftsprozessmodellierungssprachen eingesetzt, die keine präzise Kontrollflusssemantik besitzen [Harm16, KJHP15]. Durch den Kontrollfluss wird die Ablaufreihenfolge der Aktivitäten im Geschäftsprozessmodell festgelegt. Ungewollte Mehrdeutigkeiten ergeben sich immer dann, wenn die Kontrollflusssemantik unpräzise definiert ist. Dies ist beispielsweise der Fall, wenn die Bedeutung einzelner Symbole nicht eindeutig festgelegt ist. Missverständnisse zwischen Domänenexperten und Modellierern werden durch eine unpräzise Kontrollflusssemantik gefördert. Mehrdeutigkeiten erschweren bzw. verhindern die Analyse (z. B. in Form einer Simulation), Überwachung (z. B. mittels einer Workflow Engine) und Verbesserung (z. B. im Kontext von Kosten, Zeit und Ressourcen) der Geschäftsprozesse. Dementsprechend ist es das Ziel dieser Arbeit, eine leichtgewichtige Methode zur Beschreibung der Kontrollflusssemantik von graphischen Geschäftsprozessmodellierungssprachen zu entwickeln, um die Ablaufreihenfolge der Elemente eines Geschäftsprozessmodells präzise

beschreiben zu können. Darauf aufbauend werden zum einen erste Anhaltspunkte für die Vermutung gesammelt, dass die ungewollten Mehrdeutigkeiten in der Kontrollflusssemantik der Geschäftsprozessmodelle durch die Anwendung der Methode reduziert werden können und die Verständlichkeit der Kontrollflusssemantik der Geschäftsprozessmodelle somit verbessert werden kann. Zum anderen werden erste Erkenntnisse über die Methode gesammelt, ob diese einfach benutzbar, leicht erlernbar, leicht verständlich ist, eine einfache Beschreibung der Kontrollflusssemantik ermöglicht und für die Verständlichkeit der Kontrollflusssemantik der Geschäftsprozessmodelle nützlich ist. Die Möglichkeit der Analyse der Geschäftsprozessmodelle wird in einem Software-Werkzeug nachgewiesen. Die daher zu betrachtenden drei zentralen Aspekte sind:

(1) die *Modellierungssprachen* zur Dokumentation der Geschäftsprozesse und die daraus entstehenden *Geschäftsprozessmodelle.*

(2) die *Analyseverfahren* zur strukturierten Untersuchung der Geschäftsprozessmodelle.

(3) das *Softwaresystem* zur effizienten Unterstützung der Dokumentation, Analyse und Verbesserung der Geschäftsprozesse.

Modellierungssprachen: Im Geschäftsprozessmanagement werden die logisch-kausalen Zusammenhänge mit (graphischen) Modellierungssprachen dokumentiert [Wesk12], die durch eine Notation, Syntax und Semantik definiert sind [FiKa13]. In der Literatur gibt es zahlreiche Geschäftsprozessmodellierungssprachen, wie aus der nicht abschließenden Tabelle 3 hervorgeht. Eine geeignete Modellierungssprache kann unter anderem mit dem 4-Schichten-Modell nach [Ober96] (vgl. Abbildung 7) oder der Klassifikation nach [Aals13] (formale Sprachen, konzeptuelle Sprachen und Ausführungssprachen) ausgewählt werden. Somit können die Modellierungssprachen abhängig von den Eigenschaften und den notwendigen Anforderungen ausgewählt werden. Aufgrund der Vielfalt der Geschäftsprozessmodellierungssprachen sollen nachfolgend anhand von ausgewählten Anforderungen einige Beispielsprachen aufgezählt werden, gruppiert nach der verwendeten Notation, Syntax und Semantik.

Auf der *Notationsebene* ist die gute Lesbarkeit der Notationselemente ausschlaggebend, die z. B. durch die farb- und formorientierten Elemente bestimmt werden. Die Geschäftsprozesse werden im einfachsten Fall in umgangssprachlicher oder tabellarischer Form dokumentiert [WiSa15]. Für die Darstellung als Ablaufdiagramm können graphische Notationselemente verwendet werden [LaCB96], welche aber nicht zwangsläufig für jeden Beteiligten intuitiv verständlich sind [LaHo13]. Zur Vermeidung von Verständnisproblemen

können standardisierte Notationselemente eingesetzt werden. Jedoch können diese Elemente für konkrete Anwendungen zu generisch sein oder abhängig von der Anwendungsdomäne unterschiedliche Interpretationen besitzen. In der Regel hat jede Anwendungsdomäne spezifische Notationselemente, um kontextbezogenes Wissen darzustellen oder auch auszutauschen. Durch die standardisierten und domänenspezifischen Notationselemente können drei Kategorien von Modellierungssprachen gebildet werden, nämlich domänenspezifische Modellierungssprachen, Standardmodellierungssprachen sowie Standardmodellierungssprachen mit domänenspezifischen Erweiterungen. Die Business Process Model and Notation (BPMN) [BPMN2.0.2], das UML-Aktivitätsdiagramm (UML-AD) [UML2.5.1], die Ereignisgesteuerte Prozesskette (EPK) [KeNS92], die Yet Another Workflow Language (YAWL) bzw. new Yet Another Workflow Language (newYAWL) [AaHo05, RuHA07] sowie die Petri-Netze (PN) [Petr62, Pete81] sind standardisierte Modellierungssprachen im Geschäftsprozesskontext. Durch die Integration von domänenspezifischen Notationselementen, wie Risiko- [MaKu12], Sicherheits- [TBCB12], Verwaltungs- [AlSc17], Gesundheits- [BSBE14, BSBE16] oder internen Kontrollaspekten [ScRa14] entstehen Standardmodellierungssprachen mit domänenspezifischen Erweiterungen. Es existieren aber auch reine domänenspezifische Sprachen, wie z. B. die PICTURE-Methode, welche speziell für die öffentliche Verwaltung entwickelt worden ist [BPRF+15]. Weitere domänenspezifische Modellierungssprachen im Gesundheitsweisen sind beispielsweise der klinische Algorithmus [KoLS04], die Standards-based Active Guideline Environment (SAGE) [TCGN+07], das Guideline Interchange Format (GLIF) [BPTO+04], PROforma [SuFo03] sowie die Human Cognitive Model Language (HCM-L) [MiMa13].

Auf der *syntaktischen Ebene* beziehen sich die Anforderungen an eine Modellierungssprache auf deren Grammatik, d. h. auf die einsetzbaren Elemente sowie die Regeln für deren Verwendung [FiKa13, KaKü02]. Grammatiken können durch Graph-Grammatiken [Roze97] oder Metamodelle [GeKP98] beschrieben werden. Als Metamodellierungssprache können UML-Klassendiagramme [UML2.5.1] verwendet werden (z. B. BPMN-Spezifikation). Für komplexere Regeln können auch zusätzliche Sprachen eingesetzt werden, wie beispielsweise die Object Constraint Language (OCL) [OCL2.4] für UML.

Die *Semantik* beschreibt die Bedeutung der Ausdrücke einer Sprache. Eine mögliche Anforderung eines Modellierers kann die Existenz einer präzisen Kontrollflusssemantik der Modellierungssprache sein. In der Praxis werden häufig Modellierungssprachen eingesetzt, die zwar eine präzise Notation und Syntax besitzen, aber über keine präzise Kontrollflusssemantik verfügen. Ein Grund ist oftmals eine fehlende oder schwammige Formulierung von einzelnen Konzepten der Modellierungssprache [Aals13]. Deutlich wird dies unter anderem bei der Verwendung der natürlichen Sprache zur Beschreibung der Kontrollflusssemantik, da hierdurch gewollte oder ungewollte Mehrdeutigkeiten entstehen. Derzeit wird

typischerweise die Business Process Model and Notation (BPMN) in der Praxis zur Modellierung der Geschäftsprozesse eingesetzt [KJHP15]. Deren Spezifikation enthält zahlreiche Mehrdeutigkeiten in der Syntax und Kontrollflusssemantik [KIGK+14]. Die Kontrollflusssemantik wird häufig nur natürlich-sprachlich und auch nur ausschnittsweise beschrieben. Zusätzlich können ungewollte Mehrdeutigkeiten entstehen, wenn die Kontrollflusssemantik eines Elementes im Geschäftsprozessmodell zwar eindeutig ist, wie beispielsweise bei einer Oder-Zusammenführung, aber aufgrund der Einbettung in ein Geschäftsprozessmodell die Kontrollflusssemantik ungewollte Mehrdeutigkeiten zulässt. Deutlich wird dies an dem Nicht-Lokalitäts-Problem einer Oder-Zusammenführung der Ereignisgesteuerten Prozesskette [Kind06]. Ausgehend von einer vorherigen Entscheidung wurde der Kontrollfluss aufgespalten, so dass einzelne Elemente im Geschäftsprozessmodell nebenläufig aufgeführt werden können. Bei einer nachgelagerten Oder-Zusammenführung des Kontrollflusses kann demnach nicht lokal entschieden werden, ob noch auf die Ausführung einzelner Elemente gewartet werden muss, bevor der Kontrollfluss fortgesetzt werden kann oder nicht. In [Ritt00] werden mögliche Interpretationen der Kontrollflusssemantik für die Oder-Zusammenführung beschrieben. Dieses Problem resultiert auch aus einer bewusst ungenauen Modellierung im Geschäftsprozessmodell, da eine konkrete Modellierung aufgrund eines speziellen Modellierungszwecks [Ober96] nicht erforderlich ist oder zu aufwendig erscheint. Es kann auch sein, dass die Kontrollflusssemantik eines Elementes gar nicht bekannt bzw. definiert ist. Insbesondere bei nicht standardisierten Modellierungssprachen treten solche Aspekte auf, sodass bereits bei einfachen Sachverhalten Mehrdeutigkeiten entsteht, wie beispielsweise bei einer Verzweigung des Kontrollflusses. Aus diesem Grund existieren standardisierte Modellierungssprachen, die durch ein Spezifikationsdokument definiert sind, in der die Syntax und Notation eindeutig beschrieben sind. Wie oben bereits erwähnt, wird die Kontrollflusssemantik oftmals nur unpräzise definiert. Die Standardmodellierungssprachen mit und ohne domänenspezifischen Erweiterungen sowie die domänenspezifischen Modellierungssprachen besitzen häufig keine präzise Kontrollflusssemantik [KBBR+16]. Ausnahmen sind beispielsweise das Petri-Netz [Petr62, Pete81] und die darauf basierende Modellierungssprache Yet Another Workflow Language (YAWL) bzw. new Yet Another Workflow Language (newYAWL) [AaHo05, RuHA07], die eine präzise Kontrollflusssemantik besitzen.

Die bestehenden Methoden zur Beschreibung der Kontrollflusssemantik einer Modellierungssprache basieren auf der Entwicklung einer individuellen operationellen [Ollo74], denotationellen [Gord79], axiomatischen Kontrollflusssemantik [Hoar69] oder einer Übersetzungssemantik [RiSS93]. Insbesondere werden diese Methoden für die domänenspezifischen Modellierungssprachen und Standardmodellierungssprachen mit domänenspezifischen Erweiterungen eingesetzt, da die zugrundeliegende Kontrollfluss-

semantik typischerweise Mehrdeutigkeiten zulässt. Die Kontrollflusssemantik der Standardmodellierungssprache wird im Spezifikationsdokument beschrieben. Diese Beschreibung ist jedoch oftmals nur natürlich-sprachlich und enthält daher gewollte oder auch ungewollte Mehrdeutigkeiten. Für die Beschreibung der Kontrollflusssemantik von Geschäftsprozessmodellen verwenden die meisten Publikationen überwiegend die Methode der Übersetzungssemantik. Als Zielmodellierungssprache werden typischerweise Petri-Netze [RaEH10, Aals99, DiDO08, CoPR15] aufgrund ihrer mathematischen Fundierung eingesetzt. Eine weitere häufig eingesetzte Methode ist die Entwicklung einer individuellen operationellen Kontrollflusssemantik für die jeweilige Modellierungssprache, wie z. B. für EPK in [MeNü03] oder für BPMN mit abstrakten Zustandsmaschinen in [KIGK+14] aufgezeigt wird.

Analyse: Für die Analyse und Verbesserung der Geschäftsprozesse ist eine präzise Kontrollflusssemantik notwendig, damit keine ungewollten Mehrdeutigkeiten in den Geschäftsprozessmodellen auftreten können [BPRF+15, Aals13]. Für Geschäftsprozessmodelle, deren Ablaufreihenfolge der Elemente nicht präzise beschrieben ist, können nur strukturanalysierende Verfahren eingesetzt werden und keine Analyseverfahren, die die Ablaufreihenfolge der Elemente berücksichtigen. Die Überprüfung der inhaltlichen Korrektheit, Widerspruchs- und Redundanzfreiheit wird durch eine präzise Kontrollflusssemantik möglich, wie in [WFHS+15, SWFP15] durch eine Graphical Computation Tree Logic (G-CTL) aufgezeigt wird. Zudem können Aussagen über die Mindestdauer, den Ressourcenverbrauch oder die Kapazitätsauslastung getroffen werden, aber auch die Simulation der Geschäftsprozessmodelle und eine damit einhergehende Auswirkungsanalyse wird mit einer präzisen Kontrollflusssemantik möglich [Wesk12, Ober96]. Typischerweise wird zur Behebung des Mangels der fehlenden präzisen Kontrollflusssemantik eine Übersetzungssemantik oder operationelle Kontrollflusssemantik für die Modellierungssprache entwickelt. In diesem Zusammenhang wird in der Regel auch die Modellierungssprache in ihren Möglichkeiten zur Modellierung von Sachverhalten eingeschränkt. Deutlich wird dies beispielsweise an der Entwicklung der Übersetzungssemantik von [DiDO08], die die Kontrollflusssemantik von BPMN-Elementen mit Petri-Netzen beschreiben, aber aufgrund von Mehrdeutigkeiten unter anderem die Fehlerbehandlung von Mehrfachinstanzen in Schleifen, das inklusive Gateway sowie weitere Elemente nicht berücksichtigen. Mit einer präzisen Kontrollflusssemantik können neue Analyseverfahren für die Modellierungssprache konzipiert oder die Analyseverfahren der Zielmodellierungssprache bei einer Übersetzungssemantik angewendet werden. Nachdem die Geschäftsprozessmodelle modelliert, analysiert und gegebenenfalls angepasst bzw. verbessert wurden, können die Geschäftsprozessmodelle mit oder ohne ein Softwaresystem implementiert werden [Wesk12].

Softwaresystem: Für die effiziente Unterstützung der Geschäftsprozessmodellierung und -analyse können Softwaresysteme eingesetzt werden. In [MüLa17, MüBö15] werden ausgewählte Softwaresysteme zur Modellierung und Simulation von Geschäftsprozessmodellen verglichen. Dabei kann festgestellt werden, dass sich die Softwaresysteme zur Simulation in drei Gruppen untergliedern lassen, d. h. in

(1) die klassischen Simulationswerkzeuge,

(2) Modellierungswerkzeuge mit Simulationsfähigkeiten und

(3) Modellierungswerkzeuge mit Geschäftsprozessanimationen zur Verdeutlichung der Zusammenhänge.

Die Softwaresysteme aus (1) werden für die Bearbeitung von allgemeinen Simulationsaufgaben verwendet, wie beispielsweise AnyLogic (www.anylogic.de). AnyLogic unterstützt alle Evaluationskriterien aus [MüLa17, MüBö15], wie die Möglichkeit mehrere Simulationsexperimente parallelisiert auszuführen. Die geschäftsprozessspezifischen Eigenschaften werden von AnyLogic jedoch nur indirekt unterstützt, wie z. B. die Unterscheidung von aktiven und passiven Entitäten sowie die Unterstützung von Und-, Oder- bzw. Entweder-Oder-Entscheidungen und -Zusammenführungen. Zu (2) zählen Softwaresysteme zur Geschäftsprozessmodellierung, -analyse und -automatisierung, die auch Simulationsfähigkeiten unterstützen, wie Prozesszeiten und -kosten. Beispielsweise beinhaltet Bizagi (www.bizagi.com) einen separaten Zeiteditor, um Zwischenankunftszeiten, Dauer der Aktivitäten und anfallende Wartezeiten zu definieren. Es kann zwischen 12 verschiedenen stochastischen Verteilungen und konstanten Zeiteingaben gewählt werden. Die Simulation von BPMN-Geschäftsprozessmodellen, die ein inklusives Gateway besitzen, ist dennoch in Bizagi nicht möglich. Analog gilt dies für ähnliche Softwaresysteme aus (2). Die Softwaresysteme aus (3) bieten im Vergleich zu (2) nicht die Möglichkeit der Parametrisierung, wie z. B. für Kosten oder auch Ausführungszeiten. Für Standardmodellierungssprachen existieren in der Regel Softwaresysteme aus (2) und (3). Dahingegen sind am Markt oft nur Individuallösungen aus (1), (2) und (3) für die domänenspezifischen Modellierungssprachen und Standardmodellierungssprachen mit domänenspezifischen Erweiterungen verfügbar. Häufig wird aufgrund des geringen Verbreitungsgrades der verwendeten Modellierungssprache und hoher Kosten einer Individuallösung in der Praxis auch nur Microsoft Visio eingesetzt [KJHP15]. Im wissenschaftlichen Umfeld wird beispielsweise die ADOxx Metamodellierungsplattform zur Entwicklung von domänenspezifischen Modellierungswerkzeugen eingesetzt [KaMM16].

1.2 Zielsetzung

Motiviert durch die gegebene Ausgangssituation, ist das Ziel dieser Arbeit, wie bereits oben beschrieben:

eine Methode zur Beschreibung der Kontrollflusssemantik von graphischen Geschäftsprozessmodellierungssprachen zu entwickeln, um die Ablaufreihenfolge der Elemente eines Geschäftsprozessmodells präzise beschreiben zu können.

Die daraus abgeleiteten Teilziele, die im Rahmen dieser Arbeit adressiert werden sollen, sind in Abbildung 1 visualisiert und werden nachfolgend beschrieben:

- **Leichtgewichtige Methode:** Durch die Verknüpfung der Syntax einer Modellierungssprache mit Schablonen bzw. Kontrollflussmustern wird die Beschreibung der Kontrollflusssemantik einer Geschäftsprozessmodellierungssprache ermöglicht. Hierdurch muss keine von Grund auf neue operationelle, denotationelle oder axiomatische Semantik für die Beschreibung der Kontrollflusssemantik der Geschäftsprozessmodellierungssprache entwickelt werden.

- **Kontrollflusssemantik für Geschäftsprozessmodellierungssprachen:** Durch die Beschreibung der Kontrollflusssemantik einer Geschäftsprozessmodellierungssprache wird die mögliche Kontrollflusssemantik eines Elementes im Geschäftsprozessmodell festgelegt. Die Methoden zur Verbesserung der Abläufe und Strukturen werden erweitert, so dass die tatsächlichen Gegebenheiten auch mit den Modellierungssprachen dokumentiert werden. Auf dieser Grundlage wird die Analyse (z. B. in Form einer Simulation), Überwachung (z. B. mittels einer Workflow Engine) und Verbesserung (z. B. im Kontext von Kosten, Zeit und Ressourcen) der dokumentieren Geschäftsprozesse möglich.

- **Kontrollflusssemantik für Geschäftsprozessmodelle:** Durch eine präzise Beschreibung der Ablaufreihenfolge der Elemente im Geschäftsprozessmodell wird das Verständnis der beteiligten Akteure über die Kontrollflusssemantik der Geschäftsprozessmodelle gefördert. Die realen Geschäftsprozesse können mit den dokumentierten Geschäftsprozessmodellen verglichen werden und die logisch-kausalen Zusammenhänge auf Vollständigkeit und Korrektheit überprüft werden. Die damit einhergehende Qualitätssicherung kann Fehler vor der Implementierung

erkennen, um Folgekosten durch erzwungene Eingriffe während der Laufzeit zu vermeiden.

– **Simulation von Geschäftsprozessmodellen:** Durch die präzise Kontrollflusssemantik können Simulationsexperimente für die Geschäftsprozessmodelle durchgeführt werden, um die modellierten Sachverhalte mit der Intention des Modellierers abzugleichen. Es werden Auswirkungsanalysen (z. B. Kennzahlen, Mindestdauer, Ressourcenverbrauch oder Kapazitätsauslastung) möglich, deren Ergebnisse zur Verbesserung der Geschäftsprozesse (z. B. im Kontext der Produktqualität, Durchlaufzeiten und Kundenbedürfnisse) eingesetzt werden können.

– **Evaluation der Methode:** Durch die Beschreibung der möglichen Ablaufreihenfolge der Elemente eines Geschäftsprozessmodells ist die Kontrollflusssemantik präzise beschrieben. Entgegen der Empfehlungen der sieben Geschäftsprozessmodellierungsrichtlinien (7PMG) von [MeRv10] kann dadurch eine Oder-Verzweigung sowie -Zusammenführung in Geschäftsprozessmodellen eingesetzt werden.

Abbildung 1: Überblick - Ziele der Arbeit[1]

[1] Layout von https://slidemodel.com adaptiert.

1.3 Gliederung der Arbeit

Die Arbeit umfasst acht Kapitel (vgl. Abbildung 2) und enthält neben dem Kernbereich und einer Evaluation ein einleitendes und abschließendes Kapitel sowie einen Teil für die wesentlichen Grundlagen dieser Arbeit.

Die Grundlagen werden durch das Geschäftsprozessmanagement (Kapitel 2), die Modellierungssprachen zur Dokumentation der Geschäftsprozesse (Kapitel 3) und die Kontrollflussmuster zur Beschreibung der Kontrollflusssemantik von Geschäftsprozessmodellierungssprachen (Kapitel 4) gebildet. Im *zweiten Kapitel* werden einführend die Begriffe Geschäftsmodell, Geschäftsprozess und Geschäftsprozessmodell definiert. Darauf aufbauend wird das Geschäftsprozessmanagement betrachtet und in Form des Lebenszyklus von Geschäftsprozessen vorgestellt, welches die strukturierte Herangehensweise von der Erhebung und Dokumentation der Geschäftsprozesse, über die Analyse, Konzeption und Umsetzung bis hin zur Überwachung beschreibt. Während der Dokumentationsphase werden die Geschäftsprozesse mit Modellierungssprachen beschrieben, die im *dritten Kapitel* behandelt werden. Einleitend wird hierbei der Begriff der Modellierungssprache definiert. Nachfolgend werden die Modellierungssprachen Business Process Model and Notation (BPMN) [BPMN2.0.2], Ereignisgesteuerte Prozesskette (EPK) [KeNS92], UML-Aktivitätsdiagramm (UML-AD) [UML2.5.1], Petri-Netz (PN) [Petr62], Yet Another Workflow Language (YAWL) bzw. new Yet Another Workflow Language (newYAWL) [Russ07] vorgestellt. Das Kapitel wird mit dem Hinweis auf weitere Geschäftsprozessmodellierungssprachen abgeschlossen. Die Kontrollflussmuster zur Beschreibung der Kontrollflusssemantik von Geschäftsprozessmodellierungssprachen werden im *vierten Kapitel* betrachtet. Zu Beginn des Kapitels werden die Begriffe Kontrollfluss und Kontrollflussmuster definiert sowie die Kontrollflussmuster der Workflow Pattern Initiative vorgestellt. Für die präzise Beschreibung der Kontrollflusssemantik von Geschäftsprozessmodellierungssprachen, Geschäftsprozessmodellen und Kontrollflussmustern werden die Beschreibungsarten der Übersetzungssemantik, der operationellen, denotationellen und axiomatischen Semantik aus der Literatur betrachtet. Abgerundet werden die Grundlagen jeweils mit einer Zusammenfassung, in der die relevanten Aspekte sowie deren Implikationen für die Arbeit aufgezeigt werden.

Aufbauend auf den Grundlagen des Geschäftsprozessmanagements, der Geschäftsprozessmodellierungssprachen sowie den Kontrollflussmustern wird die Methode zur Beschreibung der Kontrollflusssemantik von Geschäftsprozessmodellierungssprachen im *fünften Kapitel* vorgestellt. In diesem Zusammenhang werden zunächst die notwendigen Anforderungen an die Geschäftsprozessmodellierungssprachen sowie das daraus resultierende Modell zur Anwendung der Methode vorgestellt. Daran anknüpfend wird das Vorgehen zur

Anwendung der Methode zur Beschreibung der Kontrollflusssemantik einer Geschäftsprozessmodellierungssprache vorgestellt. Abgerundet wird das Kapitel mit der Anwendung der Methode auf die vorgestellten Modellierungssprachen aus *Kapitel drei*. Auf der Grundlage der Beschreibung der Kontrollflusssemantik einer Geschäftsprozessmodellierungssprache wird nachfolgend im *sechsten Kapitel* eine Methode zur präzisen Beschreibung der Ablaufreihenfolge der Elemente in einem Geschäftsprozessmodell definiert. Dabei werden die Anforderungen an das Geschäftsprozessmodell erörtert, das Vorgehen zur Spezifikation der möglichen Ablaufreihenfolge der Elemente im Geschäftsprozessmodell vorgestellt und anhand von Beispielen die Anwendung demonstriert.

Das *siebte Kapitel* dokumentiert die Durchführung einer empirischen Untersuchung, bei der erste Erkenntnisse gesammelt wurden, ob durch die Anwendung der entwickelten Methode die ungewollten Mehrdeutigkeiten in den Geschäftsprozessmodellen reduziert werden können und somit die Verständlichkeit der Kontrollflusssemantik der Geschäftsprozessmodelle verbessert werden kann. Darüber hinaus wurden erste Anhaltspunkte gesammelt, ob die entwickelte Methode zur präzisen Beschreibung der Ablaufreihenfolge der Elemente in Geschäftsprozessmodellen einfach benutzbar, leicht erlernbar, leicht verständlich ist, eine einfache Beschreibung der Kontrollflusssemantik ermöglicht und nützlich für die Verständlichkeit der Kontrollflusssemantik der Geschäftsprozessmodelle ist. Die Möglichkeit zur Analyse der Geschäftsprozessmodelle wird mit einer interaktiven Simulation in einem Software-Werkzeug nachgewiesen. Zusätzlich wird die Nützlichkeit und Anwendbarkeit des entwickelten Software-Werkzeuges sowie die Zufriedenheit des Anwenders mit diesem betrachtet.

Abgerundet wird die Arbeit im *achten Kapitel* mit einer Zusammenfassung der einzelnen Ergebnisse. Es wird der originäre Beitrag der Arbeit betrachtet. Darüber hinaus werden die Grenzen der entwickelten Methode kritisch betrachtet, um daraus neue und weiterführende Fragestellungen für den Ausblick abzuleiten.

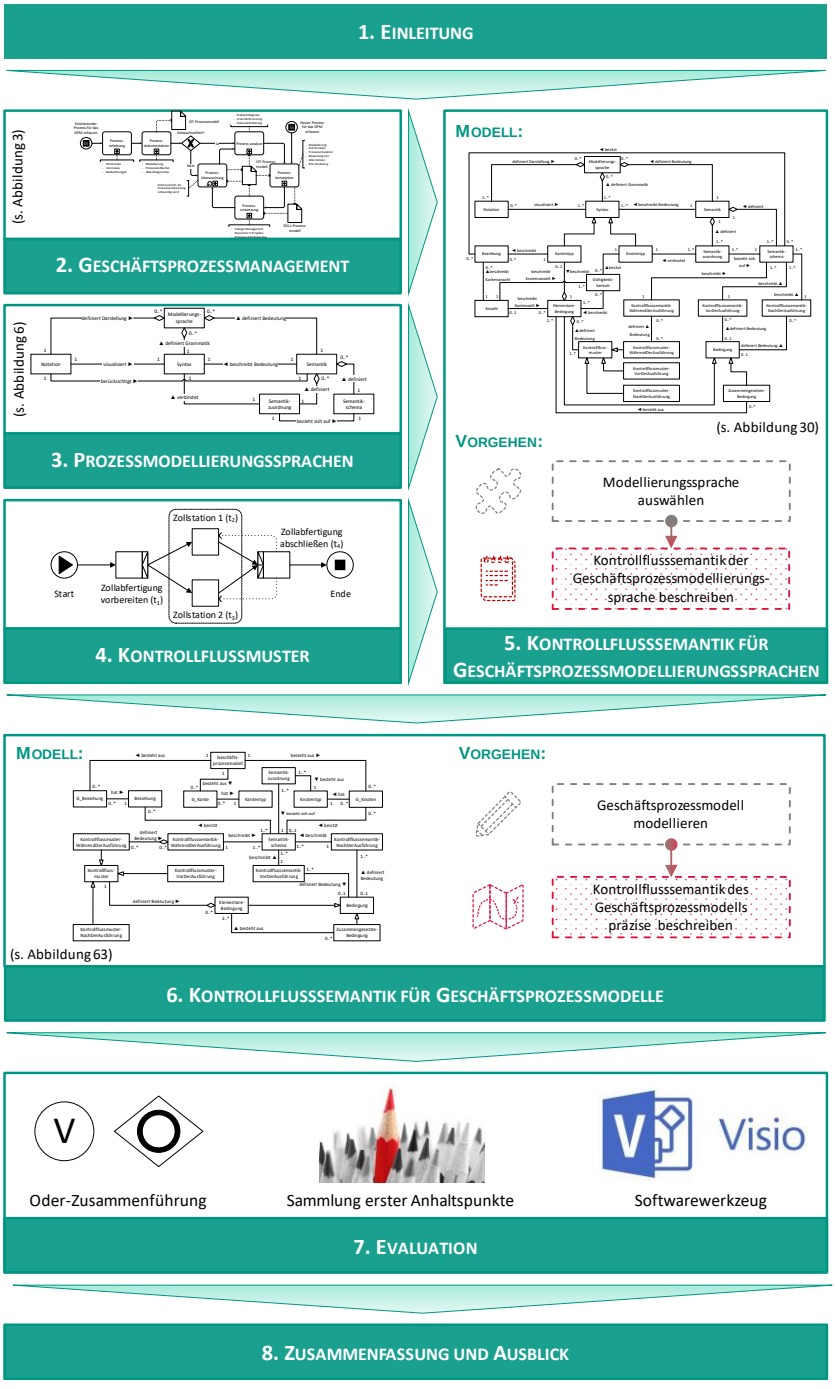


Abbildung 2: Überblick - Struktur der Arbeit

2 Grundlagen des Geschäftsprozessmanagements

Das Geschäftsprozessmanagement kann für die systematische Betrachtung der Geschäftsprozesse eingesetzt werden. Hierbei werden alle relevanten Bereiche eines Geschäftsprozesslebenszyklus, d. h. von der Erhebung und Dokumentation über die Analyse, Konzeption und Umsetzung bis hin zur Überwachung der Geschäftsprozesse für die kontinuierliche Verbesserung und Weiterentwicklung betrachtet. Die Detailinformationen der Geschäftsprozesse können wiederum in aggregierter Form in das Geschäftsmodell integriert werden, so dass sich die Reichweite des Geschäftsprozessmanagements vom übergeordneten Geschäftsmodell bis zu den implementierten Geschäftsprozessen erstreckt [Wesk12]. Im Folgenden werden zunächst die Begriffe Geschäftsmodell, Geschäftsprozess und Geschäftsprozessmodell eingeführt sowie die Teilbereiche des Geschäftsprozessmanagements, d. h. die Erhebung, Dokumentation, Analyse, Konzeption, Umsetzung und Überwachung vorgestellt. Abgerundet wird das Kapitel mit einer Zusammenfassung, in der die relevanten Aspekte sowie deren Implikationen für die Arbeit noch einmal beschrieben werden.

2.1 Geschäftsmodell, -prozess und -prozessmodell

Die durch Kunden geforderte Diversifikation und Individualisierung von Produkten und Leistungen erhöhen für Unternehmen die Komplexität und den Aufwand für das Management von Varianten betrieblicher Abläufe. Daher muss zur Erreichung der Geschäftsziele und -strategien ein geeignetes Geschäftsmodell vorliegen, um die Wettbewerbsfähigkeit sicherzustellen und Wettbewerbsvorteile zu generieren [Gait12]. Die Geschäftsziele beschreiben die langfristigen Ziele eines Unternehmens. Die Strategien sind dabei die Verhaltensweisen der Unternehmung gegenüber der Umwelt zur Erreichung dieser Ziele [Wesk12]. Das Geschäftsmodell definiert im Kern die Funktionsweise eines Unternehmens und wird in der Literatur vielfach definiert. [BiBK02] beschreiben ein Geschäftsmodell als vereinfachte Darstellung oder als Abbilder der Mechanismen sowie die Art und Weise, wie ein Unternehmen oder eine Branche am Markt Werte schafft. Eine etymologische Herleitung durch die beiden konstruierenden Wortbestandteile Modell und Geschäft liefert unter anderem [Dole14], der ein Geschäftsmodell als vereinfachte und modellhafte Abbildung definiert. Es werden mittels Ressourcentransformation und unter Einsatz besonderer Austauschbeziehungen mit anderen Wirtschaftssubjekten Werte geschaffen. Auf der

Grundlage einer Literaturrecherche betrachtet [Özsu15] den Begriff mit dem Ergebnis, dass ein Geschäftsmodell im Wesentlichen durch die drei Elemente:

(1) Produkt-/Marktkombination (Welche Produkte auf welchen Märkten?),

(2) Konfiguration und Durchführung der Wertschöpfungsaktivitäten (Wie werden die Wertschöpfungsstufen angeordnet und miteinander verknüpft?) sowie

(3) Ertragsmechanik (Wie werden die Erlöse generiert?)

und deren Beziehungen zueinander bestimmt wird, um einen Kundennutzen zu generieren und Wettbewerbsvorteile zu halten.

Definition Geschäftsmodell:

„Ein Geschäftsmodell umfasst ein Konstrukt mit drei Elementen sowie die Beziehungen zwischen diesen Bestandteilen. Die Elemente sind: «Produkt-/Markt-Kombination», «Konfiguration und Durchführung der Wertschöpfungsaktivitäten» sowie «Ertragsmechanik». Die Beziehungen zwischen den Elementen bestimmen den generierten Kundennutzen sowie entstehende Wettbewerbsvorteile als Ziele des Geschäftsmodells."

[Özsu15]

Durch das Geschäftsmodell wird eine Analyse des langfristigen Erfolgs von Unternehmen ermöglicht, da die wertschöpfenden Abläufe, Funktionen und Interaktionen der Stakeholder abgebildet werden [Dole14]. Eine zielführende Analyse wird jedoch erst durch die detaillierte Betrachtung der Ablaufreihenfolge der einzelnen Aktivitäten möglich, was unter dem Begriff Geschäftsprozess zusammengefasst wird. Die Detailinformationen der Geschäftsprozesse können wiederum in aggregierter Form in das Geschäftsmodell integriert werden. [HaCh93] definieren einen Geschäftsprozess als

„a collection of activities that takes one or more kinds of input and creates an output that is of value to the customer."

Der Schwerpunkt dieser Definition liegt auf dem Input-Output-Verhalten, wobei die Reihenfolge der Aktivitäten nicht betrachtet wird. [Dave93] integriert die Ausführungsreihenfolge:

„A process is simply a structured, measured set of activities designed to produce a specific output for a particular customer or market […]. A process is thus a specific ordering of work activities across time and space with a beginning, an end, and clearly identified inputs and outputs: a structure for action".

[AaSt11, Wesk12] legen den Schwerpunkt auf die Unternehmensperspektive. Dementsprechend besteht ein Geschäftsprozess aus einer Menge von Aktivitäten, die auf ein bestimmtes Unternehmensziel hin ausgeführt werden. Der Geschäftsprozess wird innerhalb eines Unternehmens ausgeführt und kann mit Geschäftsprozessen anderer Unternehmen interagieren. Für diese Arbeit wird hinsichtlich der nachfolgenden Ausführungen der Begriff Geschäftsprozess nach [Ober96] definiert. Weitere alternative Definitionen werden in [Dres16a] aufgezeigt. Die Begriffe betrieblicher Ablauf, Prozess und Geschäftsprozess werden synonym zueinander verwendet.

Definition Geschäftsprozess:

„Ein betrieblicher Ablauf ist eine Menge von manuellen, teil-automatisierten und automatisierten Aktivitäten, die in einem Unternehmen nach bestimmten Regeln, auf ein bestimmtes Ziel hin ausgeführt werden. Die Aktivitäten hängen über betroffene Personen, Maschinen, Dokumente u. ä. miteinander zusammen."

[Ober96]

Ein Beispiel für einen Geschäftsprozess ist die Überprüfung eines Schadensfalls bei einer Versicherungsgesellschaft. Der Prozessauslöser ist eine Schadensmeldung durch den Versicherungsnehmer. Die zu erledigenden Aktivitäten sind die Schadensaufnahme, Schadensbearbeitung sowie Schadensbegutachtung und die Kontaktaufnahme mit dem Versicherungsnehmer. Sofern bei der Schadensaufnahme festgestellt wird, dass die Schadenssumme 1.000 Euro übersteigt, muss beispielsweise zusätzlich bei der Schadensbegutachtung ein externer Gutachter beauftragt werden. Die Beauftragung des externen Gutachters ist bei einer Schadensumme von weniger als 1.000 Euro dahingegen nicht erforderlich. Darüber hinaus müssen die einzelnen Aktivitäten bei einer Schadenssumme von mehr als 1.000 Euro sequentiell abgearbeitet werden. Bei einer Schadenssumme von weniger als 1.000 Euro werden die Aktivitäten parallel abgearbeitet. Dementsprechend werden abhängig von der Schadenssumme unterschiedliche Aktivitäten und/oder auch Aktivitätsreihenfolgen ausgeführt. Die erstellte Dokumentation für einen Geschäftsprozess, welche in der Regel mit einer Visualisierung einhergeht, entspricht einem Geschäftsprozessmodell und beschreibt als Blaupause die möglichen Ausführungsvarianten [Wesk12]. Es wird somit ein Abbild einer realen Gegebenheit geschaffen, mit den Eigenschaften, die aus der Sicht des Modellierers für den gegebenen Anwendungszweck relevant sind [Stac73]. Zur Dokumentation der Geschäftsprozessmodelle werden Modellierungssprachen bzw. Beschreibungsmodelle (vgl. Kapitel 3) eingesetzt. Die Geschäftsprozessmodelle können unter anderem als Grundlage zur Spezifikation, Konfiguration, Implementierung und Analyse von Softwaresystemen verwendet werden [Aals07]. Die konkreten ausgeführten Geschäftsprozesse werden auch als Geschäftsprozessinstanzen bezeichnet und sind

mögliche Ausprägungen des Geschäftsprozessmodells [Ober96]. Die Begriffe Prozessmodell und Geschäftsprozessmodell werden synonym zueinander verwendet.

Definition Geschäftsprozessmodell:

„Ein Geschäftsprozessmodell ist eine Beschreibung aller relevanten Aspekte eines Geschäftsprozesses unter Verwendung eines Beschreibungsmodells."

[Ober96]

2.2 Geschäftsprozessmanagement

Geschäftsprozessmodelle können nicht nur zur Dokumentation verwendet werden, sondern bilden auch die Grundlage des Geschäftsprozessmanagements, wie aus den 20 Anwendungsfällen in [Aals12] (vgl. Kapitel 2.2.1) hervorgeht. Daraus wird deutlich, dass das Geschäftsprozessmanagement neben Dokumentationszwecken, wie etwa für die Einarbeitung neuer Akteure und der Dokumentation von Spezialfällen oder Ausnahmen, auch zur Analyse, Verbesserung und Überwachung eingesetzt werden kann [Wesk12]. Dies benötigt ein weitreichendes Spektrum an Methoden, Konzepten und Techniken, damit eine Modellierung, Administration, Konfiguration, Ausführung und Analyse von Geschäftsprozessen ermöglicht wird [Wesk12, Aals12]. Dies wird unter dem Begriff Geschäftsprozessmanagement zusammengefasst.

Definition Geschäftsprozessmanagement:

„Das Geschäftsprozessmanagement beschreibt Konzepte, Methoden und Techniken zur Modellierung, Administration, Konfiguration, Ausführung und Analyse der Geschäftsprozesse."

[Wesk12]

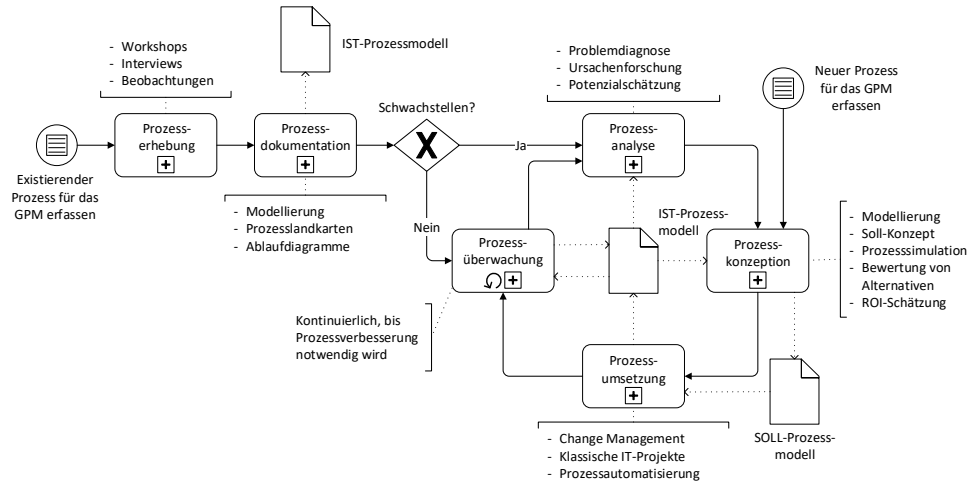


Abbildung 3: Überblick - Geschäftsprozessmanagement - Lebenszyklus [FrRü17]

Das Management dieser Geschäftsprozesse wird in der Literatur überwiegend als Kreislauf dargestellt und als Geschäftsprozesslebenszyklus bezeichnet [AHFL12]. Ein Vergleich verschiedener Kreisläufe ist beispielsweise aus [MKPC14] oder [HoFL10] zu entnehmen. Im Rahmen dieser Arbeit wird der Geschäftsprozesslebenszyklus von [FrRü17] zugrunde gelegt, welcher in Abbildung 3 dargestellt ist. Der ausgewählte Lebenszyklus enthält keine Modellierungsphase im Vergleich zu anderen Zyklen [MKPC14], da im Rahmen des Geschäftsprozesslebenszyklus von [FrRü17] die Modellierung als Querschnittsfunktion angesehen wird. Im Besonderen wird diese Methode bei der Geschäftsprozessdokumentation sowie -konzeption eingesetzt, aber auch in allen anderen Phasen des Geschäftsprozessmanagements [FrRü17]. Der Lebenszyklus wird als Geschäftsprozessmodell mit der Modellierungssprache Business Process Model and Notation (BPMN) dargestellt, um aufzuzeigen, dass der Lebenszyklus ein Modell ist und jedes Geschäftsprozessmodell einer Instanz des Lebenszyklus zugeordnet werden kann. Dementsprechend ist jedes einzelne Geschäftsprozessmodell einer Phase zugeordnet. Eine neue Instanz des Lebenszyklus wird für ein Geschäftsprozessmodell erzeugt, sobald ein existierender Geschäftsprozess dokumentiert, analysiert und/oder verbessert werden soll, d. h. das Management des Geschäftsprozesses angestrebt oder ein neuer Geschäftsprozess zum Beispiel in einem Unternehmen eingeführt wird und durch das Geschäftsprozessmanagement erfasst werden soll.

2.2.1 20 Anwendungsfälle des Geschäftsprozessmanagements

Das Geschäftsprozessmanagement umfasst zum einen die Erhebung und Dokumentation der Geschäftsprozessmodelle, welche manuell (*design model*, kurz: DesM) aus Ereignislogdaten, z. B. mit Process Mining Techniken (*discover model from event data*, kurz: DiscM), durch eine Auswahl (*select model from collection*, kurz: SelM), Vereinigung (*merge model*, kurz: MerM) oder durch das Zusammenfügen (*compose model*, kurz: CompM) von bestehenden Geschäftsprozessmodellen erfolgen kann. Darüber hinaus können Referenzmodelle bzw. konfigurierbare Geschäftsprozessmodelle entwickelt werden, die ebenfalls manuell (*design configurable model*, kurz: DesCM) oder durch das Zusammenfügen bestehender Geschäftsprozessmodelle (*merge models into configurable model*, kurz: MerCM) konstruiert werden können. Aus der Konfiguration eines Referenzmodells entsteht wiederum ein spezifisches Geschäftsprozessmodell (*configure configurable model*, kurz: ConCM) [Aals12].

Nachdem die Geschäftsprozessmodelle erhoben und dokumentiert wurden, können diese analysiert werden. In diesem Zusammenhang müssen zunächst die deskriptiven oder normativen Modelle in ausführbare Modelle transformiert werden (*refine model*, kurz: RefM), damit diese in entsprechenden Informationssystemen ausgeführt (*enact model*, kurz: EnM) werden können. Durch die Ausführung der Geschäftsprozessmodelle in einem Softwaresystem können Ereignisdaten gewonnen werden (*log event data*, kurz: LogED), so dass ereignislogdatengestützte Leistungsmessungen (*analyze performance using event data*, kurz: PerfED) oder auch ereignislogdatengestützte Konformitätsüberprüfungen (*check conformance using event data*, kurz: ConfED) möglich sind. Zu den ereignislogdatengestützten Konformitätsüberprüfungen zählt beispielsweise die Überprüfung von Compliance Regeln [FeZa16, FeZa14]. Die Geschäftsprozessmodelle können auch ohne die Betrachtung der Ereignislogdaten analysiert werden, z. B. mittels einer modellbasierten Simulation, um Durchlaufzeiten oder Auslastungen zu betrachten sowie Engpässe zu identifizieren (*analyze performance based on model*, kurz: PerfM). Die Überwachung der Geschäftsprozesse kann darüber hinaus in aggregierter Form erfolgen, indem beispielsweise nur die Durchlaufzeit des gesamten Prozesses betrachtet wird (*monitor*, kurz: Mon). Geschäftsprozessmodelländerungen (*repair model*, kurz: RepM) können durch konformitäts- und leistungsbasierte Diagnosen (*extend model*, kurz: ExtM) angestoßen werden sowie ebenfalls auf der Basis von gewonnenen Informationen aus den Ereignisdaten (*improve model*, kurz: ImpM). Neben der Geschäftsprozessmodellanalyse kann eine Verifizierung [ISO9000] (*verify model*, kurz: VerM) mit den Geschäftsprozessmodellen angestrebt werden. Die Verbesserung der identifizierten Schwachstellen kann beispielsweise während der Laufzeit erfolgen (*adapt while running*, kurz: AdaWR), um eine Flexibilitätssteigerung zu ermöglichen. Hierzu

müssen im Folgenden die Änderungen auf die einzelnen Geschäftsprozessinstanzen angewendet werden [Aals12].

In den folgenden Abschnitten werden die einzelnen Phasen des Geschäftsprozessmanagements, d. h. die Prozesserhebung, -dokumentation, -analyse, -umsetzung, -konzeption und -überwachung beschrieben. Zudem werden die 20 Anwendungsfälle aus [Aals12] den einzelnen Phasen zugeordnet.

2.2.2 Erhebung

Bei einem existierenden Geschäftsprozess eines Unternehmens, der durch das Geschäftsprozessmanagement erfasst werden soll, wird im ersten Schritt eine Prozesserhebung durchgeführt. Hierbei werden die Rahmenbedingungen, d. h. das Ziel sowie die Input- und Output-Faktoren des Geschäftsprozesses abgebildet [FrRü17]. Die auszuführenden Aktivitäten sowie die beteiligten Akteure können durch etablierte Erhebungstechniken aufgenommen werden. In der Literatur existieren zahlreiche Techniken zur Geschäftsprozesserhebung, wie aus der Literaturanalyse von [BKFL11] deutlich wird. In [BKFL11] wurden 26 Quellen identifiziert, die Erhebungstechniken betrachten, welche in sieben übergeordnete Kategorien eingeteilt sind. Die Erhebung kann demnach durch die Sichtung von Dokumenten (13 Quellen), Beobachtungen (9 Quellen), Schätzungen bzw. Messungen (5 Quellen), Fragebogen (6 Quellen), Interviews (18 Quellen), Workshops (19 Quellen) und Auswertungen von Softwaresystemen (6 Quellen) erfolgen. In den Literaturstellen wurden zwar verschiedene Erhebungstechniken genannt, jedoch sind ausführlichere Vorgehensmodelle kaum zu finden, die beschreiben, an welchem Punkt, welche Technik am besten angewendet werden kann. In diesem Zusammenhang liefern [BKFL11] Anforderungen an ein operationalisiertes Vorgehensmodell zur Prozesserhebung. Detaillierte Ausführungen zu den einzelnen Techniken sind aus [HeÖs04, Rose06, Wesk12, KrBL13] zu entnehmen.

2.2.3 Dokumentation

Nachdem die Erkenntnisse über den zu dokumentierenden Geschäftsprozess gesammelt wurden, kann das Geschäftsprozessmodell entwickelt werden, indem alle relevanten Aspekte des Geschäftsprozesses festgehalten werden. In diese Dokumentationsphase können sieben der 20 definierten Anwendungsfälle von [Aals12] eingeordnet werden, d. h. DesM, DiscM, SelM, MerM, CompM, DesCM, MerCM und ConCM. Die diversen zugrundeliegenden Informationsquellen sind auf die verschiedenen Erhebungsmaßnahmen

zurückzuführen, sodass die obigen Anwendungsfälle im weiteren Sinne den beiden Phasen der Erhebung sowie auch der Dokumentation zugeordnet werden können.

In der Dokumentationsphase werden die aktuellen Geschäftsprozesse (Ist-Zustand) modelliert, um unter anderem die übergeordneten Ziele zur Analyse und Reorganisation, wie z. B den Entwurf von Anwendungssystemen, die Ressourcenplanung sowie die Überwachung und Steuerung der Geschäftsprozesse zu ermöglichen [Ober96]. Beispielsweise kann die Kommunikation über die Geschäftsprozesse mit den diversen Stakeholdern durch die Dokumentation vereinfacht und verbessert werden [Wesk12]. Bei der Dokumentation der Geschäftsprozesse sollten unter anderem die Grundsätze ordnungsgemäßer Modellierung (GoM) [BePV12], die sieben Prozessmodellierungsrichtlinien (7PMG) [MeRv10], die 10 Tipps für eine effektive Prozessmodellierung [Silv08] und die Best Practices in BPMN [WhMi08] berücksichtigt werden. Die Geschäftsprozessmodelle, die nach diesen Grundsätzen entwickelt wurden, werden mit Modellierungssprachen dokumentiert, die in Kapitel 3 betrachtet werden [MKPC14].

Die Grundsätze *ordnungsgemäßer Modellierung (GoM)* bestehen aus den sechs Grundsätzen der (GoM 1) Korrektheit, (GoM 2) Relevanz, (GoM 3) ökonomischen Effizienz, (GoM 4) Klarheit, (GoM 5) Vergleichbarkeit und (GoM 6) des systematischen Designs:

- Korrektheit (GoM 1): Der abzubildende Sachverhalt sollte korrekt wiedergegeben werden.
- Relevanz (GoM 2): Das Modell sollte alle relevanten Informationen bzw. Daten enthalten, die für die entsprechenden Stakeholder relevant sind.
- Ökonomischen Effizienz (GoM 3): Das Kosten-Nutzen-Verhältnis sollte berücksichtigt werden, um das Dokumentationsziel mit möglichst geringem Aufwand zu erreichen.
- Klarheit (GoM 4): Mit einer adressatengerechten Hierarchisierung, Layoutgestaltung und Filterung entsprechender Teilaspekte kann die Lesbarkeit, Anschaulichkeit und Verständlichkeit sichergestellt werden.
- Vergleichbarkeit (GoM 5): Gleiche Abläufe im Modell sollten auch gleich dargestellt werden, so dass die Vergleichbarkeit gewährleistet werden kann.
- Systematisches Design (GoM 6): Der systematische Aufbau der Geschäftsprozessmodelle ist erstrebenswert, damit Sachverhalte, die für unterschiedliche Perspektiven beschrieben werden und gleich sind, auch gleich bezeichnet werden.

Beispielsweise sollten Organisationseinheiten, die im Geschäftsprozessmodell verwendet werden auch in der Aufbauorganisation entsprechend vorhanden sein [BePV12].

Die *sieben Prozessmodellierungsrichtlinien (7 PMG)* nach [MeRv10] betrachten vor allem die Übersichtlichkeit und Verständlichkeit eines Geschäftsprozessmodells, um eine hohe Qualität der Geschäftsprozessmodelle zu erreichen.

- PMG 1: Die erste Richtlinie fordert, dass so wenig Notationselemente wie möglich in einem Geschäftsprozessmodell verwendet werden sollten [MeRC07, MVDA+08, MeNA07]. [MeRv10] haben festgestellt, dass eine höhere Anzahl von Notationselementen in einem Geschäftsprozessmodell einen negativen Einfluss auf die Verständlichkeit für den Leser und die Fehlerwahrscheinlichkeit bei der Modellierung hat.

- PMG 2: Des Weiteren sollte ein Notationselement im Geschäftsprozessmodell so wenig wie möglich ein- und ausgehende Kanten besitzen. Eine zunehmende ein- und ausgehende Kantenanzahl wirkt sich negativ auf die Verständlichkeit des Notationselements im Geschäftsprozessmodell aus [MeRC07, MeNA07].

- PMG 3: Die dritte Richtlinie fordert, dass das Geschäftsprozessmodell über einen eindeutigen Start- und Endknoten verfügen sollte, da die Fehlerwahrscheinlichkeit mit der Anzahl der Start- und Endknoten positiv korreliert [MeNA07].

- PMG 4: Zusätzlich sollte das Geschäftsprozessmodell nur über strukturierte Verzweigungen und Zusammenführungen verfügen. Strukturiert heißt, dass eine Verzweigung auch wieder zusammengeführt werden muss [MeNA07, GrLa07, LaMe08, MeRC07]. [MeRv10] haben festgestellt, dass unstrukturierte Geschäftsprozessmodelle für den Leser schwerer zu verstehen sind und auch mehr Fehler enthalten.

- PMG 5: Die fünfte Richtlinie besagt, dass Oder-Verzweigungen und -Zusammenführungen nicht verwendet werden sollten, da Geschäftsprozessmodelle, die nur Und- und Entweder-Oder-(XOR)-Verzweigungen und -Zusammenführungen besitzen, weniger Fehler enthalten [MeNA07].

- PMG 6: Die sechste Richtlinie fordert, dass die Beschriftung einer Aktivität im Verb-Objekt-Stil erfolgen sollte, da diese Beschriftungsform im Vergleich zu anderen Beschriftungsformen weniger Mehrdeutigkeiten enthält [LeSM10].

– PMG 7: In der siebten Richtlinie wird gefordert, dass ein Geschäftsprozessmodell weniger als 50 Notationselemente besitzen sollte [MeRe08].

Nachdem das Ist-Geschäftsprozessmodell dokumentiert wurde, welches im Geschäftsprozesslebenszyklus in Abbildung 3 als Datenobjekt (vgl. Abbildung 12a) dargestellt wird, werden mögliche erkannte Schwachstellen während der Erhebungs- und Dokumentationsphase in der Analysephase weiter betrachtet und entsprechend eingegrenzt. Andernfalls, sofern kein Verbesserungsbedarf identifiziert wurde, kann das Geschäftsprozessmodell bzw. können die einzelnen Geschäftsprozessinstanzen kontinuierlich überwacht werden, bis Schwachstellen identifiziert werden. Beispielsweise können Prozesskennzahlen zur Identifizierung von Schwachstellen eingesetzt werden.

2.2.4 Analyse

In der Analysephase werden die dokumentierten Ist-Geschäftsprozessmodelle untersucht, um Problemdiagnosen aufzustellen, Ursachen zu ermitteln sowie Potenziale zu identifizieren, welches auf eine Vielzahl von Analyseansätzen schließen lässt. Dementsprechend können die sieben Anwendungsfälle RefM, EnM, PerfM, VerM, ConfED, PerfED und Mon nach [Aals12] in die Analysephase eingeordnet werden. Zur besseren Strukturierung der verschiedenen Analyseansätze werden in Abbildung 4 die Verfahren in Entwurfszeit und Laufzeit untergliedert sowie der notwendigen Datengrundlage zugeordnet, d. h. Realwelt, Softwaresystem, Ereignislogdaten und Geschäftsprozessmodell. Das Softwaresystem unterstützt und kontrolliert zum einen die verschiedenen Geschäftsprozesse, Ressourcen, Maschinen etc. der Realwelt und dokumentiert zum anderen die entsprechenden Ereignisse, welche als Ereignislogdaten bezeichnet werden. Beispielsweise zählen Nachrichten oder auch Transaktionen zu diesen Ereignissen, die in der Regel durch die Softwaresysteme in einem Ereignislog aufgezeichnet werden [Aals07].

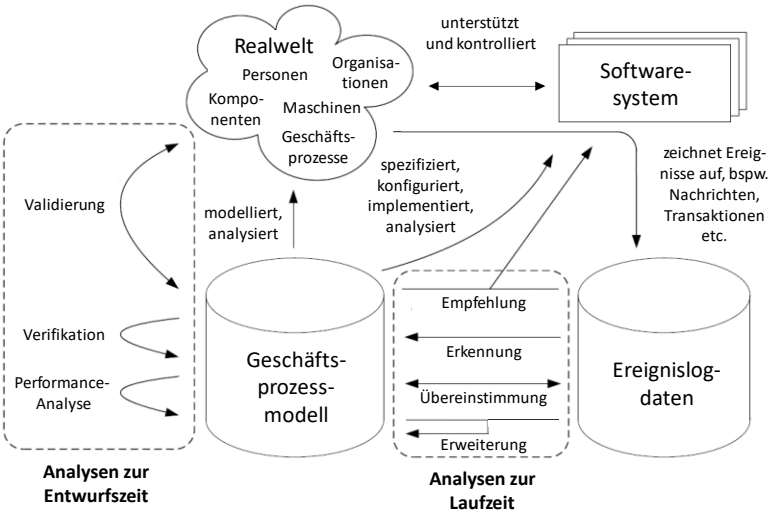


Abbildung 4: Überblick - Beziehungen zwischen der realen Welt, Softwaresystem, Ereignislogdaten und den verschiedenen Arten von Analysen [Aals07, AAMA+12]

Die modellbasierte Analyse zur Entwurfszeit ist für Unternehmen von essentieller Bedeutung, da Fehler oder Schwachstellen in den Geschäftsprozessen zu unzufriedenen Kunden, Nacharbeiten, Schadensersatzansprüchen und zum Werteverlust führen kann. Außerdem verursachen Fehler in den Geschäftsprozessen nach der Implementierung hohe Folgekosten durch erzwungene Eingriffe zur Behebung dieser Fehler während der Laufzeit [Boeh79, Boeh81]. Ebenso können hohe Durchlaufzeiten, niedrige Servicelevel oder auch Überkapazitäten durch fehlerhaftes Prozessdesign entstehen. In Abbildung 4 werden die drei Analysetypen Validierung, Verifikation und Performance-Analyse für die Entwurfszeit definiert, um Schwachstellen im Geschäftsprozessmodell zu identifizieren [Aals07].

Validierung bedeutet nach [ISO9000], einen objektiven Nachweis zu erbringen, dass die Anforderungen für einen spezifisch, beabsichtigten Gebrauch oder eine spezifisch, beabsichtigte Anwendung erfüllt wurden. Übertragen auf die Geschäftsprozessmodelle bedeutet die Validierung die Überprüfung der Gültigkeit, ob das Geschäftsprozessmodell mit dem angestrebten Ablauf bzw. der Realität übereinstimmt. Beispielsweise kann die Gültigkeit mit Tests oder auch mit einer interaktiven Simulation durch die Einbeziehung der Anwender überprüft werden [Ober96, Aals07]. Verifikation bedeutet nach [ISO9000], die Erbringung eines objektiven Nachweises, dass festgelegte Anforderungen erfüllt worden sind. Im Kontext der Geschäftsprozessmodelle ist die Verifikation die Überprüfung der syntaktischen und inhaltlichen Korrektheit eines Modells. Mittels der syntaktischen Korrektheit wird die Spezifikationsbeschreibung der Modellierungssprache mit dem dokumentierten

Geschäftsprozessmodell abgeglichen und überprüft, ob Elemente entgegen der Spezifikation kombiniert oder verknüpft wurden. Beispielsweise dürfen bei Petri-Netzen keine zwei Transitionen miteinander verbunden werden. In BPMN darf ein Nachrichtenfluss nicht für die Verbindung zweier Aktivitäten innerhalb eines Pools verwendet werden. Die logischen Fehler können mittels der inhaltlichen Verifikation entdeckt werden, sodass z. B. ein Geschäftsprozess keine Verklemmung aufweisen kann und eine Beendigung jederzeit möglich sein sollte [ReMR15]. Das Soundness-Kriterium formuliert diesen Aspekt, sodass (1) der Geschäftsprozess aus jedem Zustand beendet werden kann, (2) nach der Beendigung keine Aufgaben mehr aktiv sind und (3) keine Aufgaben im Geschäftsprozessmodell existieren, die nicht ausgeführt werden können [Aals97]. Die Einhaltung diverser Kriterien kann unter anderem mit einer Simulation überprüft werden. Hierdurch können die Geschäftsprozesse von Unternehmen verbessert werden. Die Simulation kann unter anderem auch für Performance-Analysen eingesetzt werden. Beispielsweise kann die Durchlaufzeit betrachtet werden, um die Leistungsfähigkeit des Geschäftsprozesses festzustellen [Aals07].

Eine weitere Variante, um Geschäftsprozessmodelle zu analysieren, ist die Analyse der Geschäftsprozessmodelle auf der Grundlage von Ereignislogdaten, die Laufzeitinformationen beinhalten und von den zugrundeliegenden Softwaresystemen geliefert werden. In diesem Zusammenhang haben sich die Techniken des Process Minings als Möglichkeit etabliert, sodass durch die aufgezeichneten Ereignislogdaten reale Prozesse erkannt, überwacht und verbessert werden können (vgl. Abbildung 4 - Einordung in den Gesamtkontext und Abbildung 5 - Input- und Outputparameter). Zu den drei Arten des Process Minings zählen das Erkennen (engl. Discovery), die Konformitätsprüfung (engl. Conformance) sowie die Erweiterung (engl. Extension). Beim Erkennen werden aus Ereignislogdaten entsprechende Geschäftsprozessmodelle generiert (vgl. Abbildung 4 - Einordung in den Gesamtkontext und Abbildung 5a - Input- und Outputparameter), wie beispielsweise mit der Anwendung des α-Algorithmus [AaWM04, MDAW04, Aals16, DrKO17]. Die Übereinstimmungsprüfung wird für den Abgleich der Ereignislogdaten mit dem Geschäftsprozessmodell verwendet, um Abweichungen zu identifizieren [RoAa08]. Mit dieser Methode soll analysiert werden, ob die Realität mit den entwickelten Geschäftsprozessmodellen übereinstimmt. Als Ergebnis liegt eine Diagnose vor, die die Unterschiede und Gemeinsamkeiten aufzeigt (vgl. Abbildung 4 - Einordnung in den Gesamtkontext und Abbildung 5b - Input- und Outputparameter). Mit der Technik der Erweiterung werden die Geschäftsprozessmodelle auf der Grundlage der Ereignislogdaten und des zugrundeliegenden Modells erweitert und verbessert, so dass ein neues Modell entsteht. Beispielsweise können Engpässe aufgrund der Durchlaufzeiten oder durch den Zeitstempel erkannt werden (vgl. Abbildung 4 - Einordung in den Gesamtkontext und Abbildung 5c - Input- und Outputparameter) [Aals12, AAMA+12, Aals16]. Die Process Mining Methoden beschränken sich nicht nur auf die Analyse der

Geschäftsprozessmodelle, sondern können auch für Empfehlungen verwendet werden, sodass unter anderem eine Ablaufreihenfolge vorgeschlagen werden kann. Die ausgeführten Instanzen können zum Beispiel auch nach Durchlaufzeiten, Kosten- und Qualitätsaspekten bewertet werden [Aals12, AAMA+12, Aals16]. Nachfolgend können die identifizierten Probleme, Ursachen und Potenziale der Analyseverfahren in ein Soll-Konzept integriert werden.

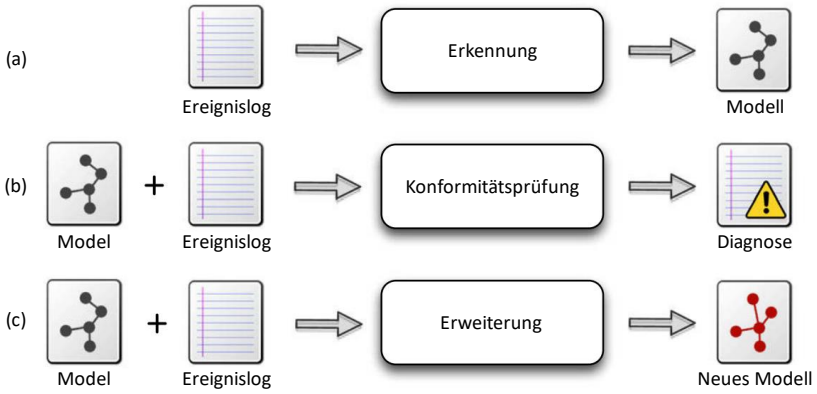


Abbildung 5: Überblick - Arten des Process Minings mit Input- und Outputparameter: (a) Erkennung, (b) Übereinstimmungsprüfung und (c) Erweiterung [AAMA+12, AcUA12, Aals16]

2.2.5 Konzeption

In der Konzeptionsphase wird das Soll-Geschäftsprozessmodell entwickelt, um die identifizierten Schwachstellen oder auch Probleme aus der Analysephase zu beheben. Ebenfalls kann eine neue Instanz des Geschäftsprozesslebenszyklus durch ein neu zu entwickelndes Geschäftsprozessmodell instanziiert werden (vgl. Abbildung 3). Eine Neuentwicklung wird vorgenommen, wenn zum Beispiel ein neuer Geschäftsprozess in das Unternehmen eingebettet werden soll [FrRü17]. Im Kontext der Anwendungsfälle für das Geschäftsprozessmanagement von [Aals12] können RepM, ExtM und ImpM in die Phase eingeordnet werden, um das Geschäftsprozessmodell zu reparieren, zu erweitern und zu verbessern.

Für die Konzeption und entsprechende Dokumentation eines geeigneten Geschäftsprozessmodells können Modellierungssprachen eingesetzt werden, die detailliert in Kapitel 3 betrachtet werden. Im Anschluss daran können die entwickelten Geschäftsprozessmodellalternativen unter anderem mittels einer Simulation oder auch Return-on-Investment (ROI)-Schätzung bewertet werden. Für die Bewertung können, je nach Anwendungsfall,

beliebige Kriterien festgelegt werden. Nachdem das Soll-Geschäftsprozessmodell entwickelt wurde, kann dieses in der nächsten Phase umgesetzt werden.

2.2.6 Umsetzung

Nachdem ein geeignetes Soll-Geschäftsprozessmodell entwickelt wurde, kann das entwickelte Modell umgesetzt werden, wobei in der Regel organisatorische sowie auch IT-basierte Änderungen damit einhergehen [FrRü17]. Dementsprechend spielt die Kommunikation mit den partizipierenden Prozessteilnehmern bezüglich der Änderungen eine zentrale Rolle, da diese Akteure für die Ausführung der einzelnen Aktivitäten verantwortlich sind. Die IT-basierten Änderungsmaßnahmen können abhängig von der IT-Infrastruktur durch eine Prozessautomatisierung, Neuentwicklung oder Anpassung von Softwaresystemen vorgenommen werden. Alternativ kann auch ein Softwaresystem beschafft werden, wie beispielsweise ein Geschäftsprozessmanagementsystem, das die strikte Einhaltung der definierten Ablaufvarianten des Geschäftsprozessmodells erfordert. Hierdurch können die einzelnen Geschäftsprozessinstanzen überwacht und kontrolliert werden. Dabei wird zu jeder Geschäftsprozessinstanz der aktuelle Zustand gespeichert, sodass das System auf Fehler sowie Ausnahmebehandlungen durch Überwachungskomponenten reagieren kann. Durch die IT-basierte Ausführung können ebenfalls eine Vielzahl an Informationen gewonnen werden. Zum Beispiel können durch das Softwaresystem die auftretenden Aktionen in einem Ereignislog gespeichert werden, das wiederum als Eingabeparameter für das Process Mining verwendet werden kann. Ein Ereignislog besteht in der Regel aus einer Menge zeitlich geordneter Einträge, insbesondere Start- und Endzeitpunkte von Aktionen. Jeder Eintrag repräsentiert eine Aktion während der Ausführung des Geschäftsprozesses. Ein ausführliches Beispiel zur Prozesskonfiguration wird exemplarisch in [GoLa10] aufgezeigt [FrRü17, Wesk12]. In die Umsetzungsphase können die Anwendungsfälle LogED und AdaWR eingeordnet werden. Der Anwendungsfall LogED erzeugt durch die Anwendung eines Geschäftsprozessmanagementsystems geeignete Ereignislogdaten. AdaWR betrachtet Änderungen am System und Modell während der Laufzeit zur Flexibilitätssteigerung und der nachgelagerten Migration der Prozessinstanzen [Aals12].

Nachdem die Änderungen im Softwaresystem abgebildet wurden, müssen diese getestet werden, bevor sie ihre Anwendung in der Zielumgebung finden. Diese Aufgabe ist dem Software Engineering zuzuordnen und wird zum Beispiel durch Testtechniken überprüft, um zu sehen, ob das System das gewünschte Verhalten abbildet. Auf der Prozessebene sind Integrations- und Performance-Tests erforderlich, um potenzielle Laufzeitprobleme frühzeitig bzw. während der Umsetzungsphase zu identifizieren. Nach Abschluss der Testphase kann das System in die Zielumgebung integriert werden, wobei abhängig von den

getätigten Änderungen unter anderem Schulungen erforderlich sind, für die Personen, die den Prozess ausführen [Wesk12].

2.2.7 Überwachung

Die Prozessüberwachung wird entgegen der anderen Phasen solange ausgeführt, bis eine definierte Abbruchbedingung erfüllt ist, wie dies im Geschäftsprozesslebenszyklus in Abbildung 3 visualisiert wird. Die dauerhafte Ausführung der Prozessüberwachung ist im Geschäftsprozessmodell als kreisförmiger Pfeil im Notationselement (vgl. Abbildung 8b) dargestellt und die Abbruchbedingung der kontinuierlichen Ausführung, bis eine Prozessverbesserung notwendig wird, ist mit einer Textannotation (vgl. Abbildung 15) modelliert. Die zentrale Aufgabe der Geschäftsprozessüberwachung ist die Evaluation und Bewertung der einzelnen Geschäftsprozessinstanzen. Die Phase der Überwachung wird nach der Umsetzung durchgeführt oder kann auch direkt auf die Prozessdokumentation folgen, sofern keine offenkundigen Schwachstellen während der Prozesserhebung und -dokumentation entdeckt wurden. Durch den Einsatz eines Geschäftsprozessmanagementsystems können mittels der integrierten Überwachungskomponenten die einzelnen Instanzen analysiert werden. In diesem Zusammenhang können Metriken zur Bewertung eingesetzt werden. In [LeMO05, MeOb05] werden die Metriken als Performance-Indikatoren bezeichnet. Weiterführend können mit Hilfe der Geschäftsprozessmanagementsysteme die Ereignislogdaten mit den Techniken des Process Minings [AAMA+12, Aals16] oder auch der Business Intelligence [GBFN16, HäFe15] überwacht werden. Dementsprechend kann die Qualität der Geschäftsprozesse ermittelt, beobachtet, analysiert und bewertet werden. Zur einfachen Darstellung der Prozesskennzahlen können neben den klassischen Visualisierungstechniken, respektive Linien-, Balken- oder Kreisdiagrammen [Schu12], auch die Signalfarben einer Ampel: grün, gelb, rot [Mevi06] oder ähnliche Techniken verwendet werden. Im Kontext der definierten Anwendungsfälle von [Aals12] können für das Geschäftsprozessmanagement die gleichen Anwendungsfälle, wie aus der Analysephase eingesetzt werden.

Sofern Schwachstellen oder Probleme bei einzelnen Geschäftsprozessinstanzen festgestellt wurden, muss nicht unmittelbar die Instanz des Lebenszyklus des Geschäftsprozessmodells in die Analysephase übergehen. Diese Einzelfälle können auch in der Überwachungsphase behoben werden, sofern die Ursachen bekannt sind, um entsprechende Gegenmaßnahmen einleiten zu können. Bei komplexeren Änderungen, Problemen oder Schwachstellen im Geschäftsprozessmodell, bei denen die Ursache unklar ist, muss eine systematische Prozessanalyse durchgeführt werden. Die Entscheidung für eine

systematische Prozessanalyse obliegt dem Verantwortlichen des Geschäftsprozesses in Abstimmung mit den entsprechenden Stakeholdern.

2.3 Zusammenfassung

In den vorherigen Abschnitten wurde neben den grundlegenden Begriffen: Geschäftsmodell, Geschäftsprozess und Geschäftsprozessmodell, der Begriff Geschäftsprozessmanagement eingeführt. Das Management der Geschäftsprozesse wurde durch einen Geschäftsprozesslebenszyklus (vgl. Abbildung 3) vorgestellt und beschreibt die strukturierte Herangehensweise von der Erhebung und Dokumentation der Geschäftsprozesse, über die Analyse, Konzeption und Umsetzung bis hin zur Überwachung. Mit dem Geschäftsprozesslebenszyklus können die im Rahmen dieser Arbeit entwickelten Methoden in das Themengebiet eingeordnet werden. Es wird zum einen in Kapitel 5 eine auf Kontrollflussmustern basierende Methode zur Beschreibung der Kontrollflusssemantik von graphischen Geschäftsprozessmodellierungssprachen vorgestellt. Zum anderen wird in Kapitel 6 eine Methode zur präzisen Beschreibung der Ablaufreihenfolge von Elementen in Geschäftsprozessmodellen beschrieben. Die Modellierungssprachen zur Dokumentation der Geschäftsprozesse werden in Kapitel 3 betrachtet sowie die Kontrollflussmuster zur Beschreibung der Kontrollflusssemantik in Kapitel 4 vorgestellt.

Häufig werden in der Praxis Geschäftsprozessmodellierungssprachen eingesetzt, deren Kontrollflusssemantik nicht präzise beschrieben ist, so dass typischerweise ungewollte Mehrdeutigkeiten entstehen. Die Missverständnisse zwischen Domänenexperten und Modellierern werden durch unpräzise Beschreibungen gefördert. Mehrdeutigkeiten erschweren bzw. verhindern die Anwendung der Methoden des Geschäftsprozessmanagements zur Analyse, Konzeption, Umsetzung und Überwachung. Unklarheiten in der Kontrollflusssemantik führen aber auch in der Dokumentationsphase zu Problemen, da der abzubildende Sachverhalt nach den Grundsätzen ordnungsgemäßer Modellierung (GoM 1) korrekt wiedergegeben werden sollte. Mit der Anwendung der entwickelten Methode kann die Ablaufreihenfolge der Elemente in den Geschäftsprozessmodellen präzise beschrieben werden. Aus diesem Grund ist beispielsweise die Berücksichtigung der fünften Regel der sieben Prozessmodellierungsrichtlinien (7PMG) nicht mehr erforderlich (vgl. Kapitel 7). Die fünfte Regel besagt, dass Oder-Verzweigungen und -Zusammenführungen nicht verwendet werden sollten.

Der Mehrwert der entwickelten Methode wird unter anderem an der Analysephase deutlich, da die realen Geschäftsprozesse mit den dokumentierten Geschäftsprozessmodellen abgeglichen werden können. Darüber hinaus können die logisch-kausalen Zusammen-

hänge auf Vollständigkeit und Korrektheit überprüft werden. Die damit einhergehende Qualitätssicherung kann Fehler vor der Implementierung erkennen, um Folgekosten durch erzwungene Eingriffe während der Laufzeit zu vermeiden. Simulationsexperimente können durchgeführt werden, um die modellierten Sachverhalte mit der Intention des Modellierers abzugleichen (vgl. Kapitel 7). Aber auch Auswirkungsanalysen (z. B. für Kennzahlen, Mindestdauer, Ressourcenverbrauch und Kapazitätsauslastung) werden möglich, deren Ergebnisse zur Verbesserung der Geschäftsprozesse (z. B. im Kontext der Produktqualität, Durchlaufzeiten und Kundenbedürfnisse) eingesetzt werden können. Die Analyseergebnisse bilden den Ausgangspunkt der Prozesskonzeption. Nachdem ein geeignetes Soll-Geschäftsprozessmodell erarbeitet wurde, kann das entwickelte Geschäftsprozessmodell umgesetzt werden, wobei in der Regel organisatorische sowie auch IT-basierte Änderungen damit einhergehen. Dementsprechend spielt die Kommunikation mit den partizipierenden Prozessteilnehmern bezüglich der Änderungen eine zentrale Rolle, da diese Akteure für die Ausführung der einzelnen Aktivitäten verantwortlich sind. Aus diesem Grund sollte die Kontrollflusssemantik in den Geschäftsprozessmodellen keine ungewollten Mehrdeutigkeiten zulassen, um Fehler bei der Implementierung zu vermeiden. Der Einsatz einer IT-basierten Geschäftsprozessüberwachung, die auch die logisch-kausalen Zusammenhänge berücksichtigt, ist nur mit einer präzisen Kontrollflusssemantik möglich. In diesem Zusammenhang wird deutlich, dass sich die entwickelte Methode aus Kapitel 5 und 6 auf alle Phasen des Geschäftsprozessmanagements auswirkt, mit Ausnahme der Erhebungsphase.

3 Modellierungssprachen für Geschäftsprozessmodelle

Im vorigen Kapitel wurde das Geschäftsprozessmanagement vorgestellt, im Rahmen dessen Geschäftsprozessmodelle entwickelt, analysiert und verbessert werden. Die entwickelten Ist- und Soll-Geschäftsprozessmodelle (vgl. Abbildung 3) werden mit Modellierungssprachen, die im Folgenden betrachtet werden, dokumentiert. Einführend werden der Begriff und die Bestandteile einer Modellierungssprache (Notation, Syntax und Semantik) definiert sowie mögliche Gründe für die Vielzahl von Modellierungssprachen in der Literatur aufgezeigt [FiKa13, KaKü02, Dres16c]. Darauf aufbauend werden exemplarisch fünf Standardmodellierungssprachen im Geschäftsprozesskontext vorgestellt. Zu diesen Sprachen zählen die Business Process Model and Notation (BPMN) [BPMN2.0.2], die Ereignisgesteuerte Prozesskette (EPK) [KeNS92], das Petri-Netz (PN) [Petr62], das UML-Aktivitätsdiagramm (UML-AD) [UML2.5.1] und die Yet Another Workflow Language (YAWL) [AaHo05] bzw. new Yet Another Workflow Language (newYAWL) [RuHE07]. Abgerundet wird das Kapitel mit dem Verweis auf weitere Modellierungssprachen für Geschäftsprozessmodelle sowie einer Zusammenfassung.

3.1 Modellierungssprachen

Eine (graphische) Modellierungssprache besteht nach [FiKa13, KaKü02] aus einer Notation, Syntax und Semantik, wie mit einem UML-Klassendiagramm in Abbildung 6 dargestellt wird. Die *Syntax* beschreibt die Elemente der Modellierungssprache sowie die Regeln für deren Verwendung und definiert somit die Grammatik der Modellierungssprache [FiKa13, KaKü02]. Grammatiken können zum einem durch Graph-Grammatiken [Roze97] und zum anderen durch Metamodelle [GeKP98] beschrieben werden. Als Metamodellierungssprache kann unter anderem das UML-Klassendiagramm [UML2.5.1] verwendet werden. Für komplexere Regeln können auch zusätzliche Sprachen eingesetzt werden, wie beispielsweise die Object Constraint Language (OCL) [OCL2.4]. Die *Notation* visualisiert die Syntax, z. B. durch Icons [KaKü02]. Die *Semantik* beschreibt die Bedeutung der Syntax und besteht aus einem Semantikschema und einer Semantikzuordnung. Das Semantikschema beinhaltet beispielsweise die Kontrollflusssemantik der Syntax. Diese kann entweder mit einer Übersetzungssemantik, operationellen, denotationellen oder axiomatischen Semantik beschrieben werden (vgl. Kapitel 4.2) oder auch durch eine einfache textuelle (informelle)

Beschreibung, wie dies unter anderem anhand des UML-Aktivitätsdiagramms [UML2.5.1], der Ereignisgesteuerten Prozesskette [KeNS92] oder der Business Process Model and Notation [BPMN2.0.2] deutlich wird. Die Semantikzuordnung verbindet das Semantikschema mit der Syntax.

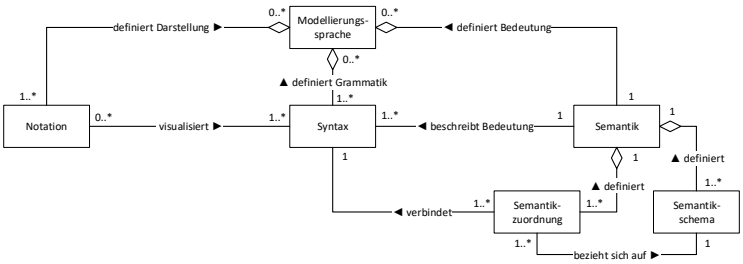


Abbildung 6: Modell - Elemente einer Geschäftsprozessmodellierungssprache (In Anlehnung an [FiKa13])

In der Literatur existieren eine Vielzahl von Modellierungssprachen, wie aus der nicht abschließenden Tabelle 3 hervorgeht. Bevor ein Geschäftsprozessmodell erstellt werden kann, muss eine geeignete Modellierungssprache ausgewählt werden. Die Auswahl ist unter anderem von den Beteiligten sowie von den darzustellenden Informationen abhängig. Hierbei ist es beispielsweise wichtig, dass der Modellierer die Modellierungssprache beherrscht. Darüber hinaus ist zu beachten, dass für einen Fachanwender die Darstellung der Aktivitäten ausreichend sein kann. Ein Entwickler benötigt dahingegen typischerweise noch weitere Informationen über die Datenstruktur [Schu12], welche unter anderem mit einem UML-Klassendiagramm [OeSc13, Stör05, UML2.5.1] oder einem Entity-Relationship-Diagramm (ER-Diagramm) [Chen76, Chen02] modelliert werden kann. Die Aufbauorganisation könnte ebenso relevant sein und mit einem Organigramm modelliert werden [Hask22, Fied14]. Eine ganzheitliche Methode zur Verknüpfung der verschiedenen Beschreibungsebenen ist das Konzept der Architektur integrierter Informationssysteme (ARIS), welches als Modellierungssprache für Geschäftsprozesse die Ereignisgesteuerte Prozesskette vorschlägt [Sche99, Sche01, Sche02]. Die Horus-Methode ist ein weiterer ganzheitlicher Ansatz zur Verknüpfung der einzelnen Beschreibungsebenen. Für die Modellierung der Geschäftsprozessmodelle werden Petri-Netze vorgeschlagen [SVOK12]. Aufgrund der Vielzahl von Methoden, Techniken und Tools ist es schwierig, allgemein eine geeignete Auswahl zu treffen [Agui04], sodass Modellierungssprachen mit ähnlichen Eigenschaften gruppiert werden können, um die Auswahl der potenziell in Frage kommenden Modellierungssprachen zu reduzieren.

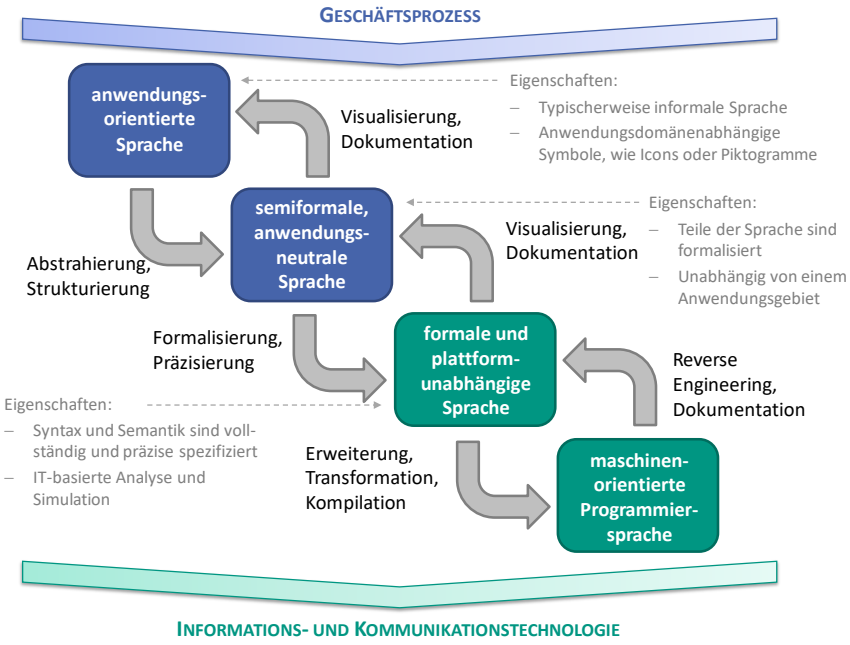


Abbildung 7: Überblick - Klassifikation für Geschäftsprozessmodellierungssprachen [Ober96]

In diesem Zusammenhang schlägt [Stra98] beispielsweise eine Differenzierung nach Funktionen oder Leistungen der Modellierungssprache vor. Eine weitere Unterscheidungsmöglichkeit ist die Differenzierung zwischen natürlicher und künstlicher Sprache [Stra98]. Alternativ können Modellierungssprachen auch in domänenspezifische Modellierungssprachen, Standardmodellierungssprachen und Standardmodellierungssprachen mit domänenspezifischen Erweiterungen gruppiert werden. Das 4-Schichten-Modell in Abbildung 7 von [Ober96] verbindet verschiedene Unterscheidungskriterien. Die Existenz der unterschiedlichen Modellierungssprachen resultiert neben der Abhängigkeit von der Anwendungsdomäne, aus einer fehlenden oder absichtlich unpräzisen Formulierung diverser Aspekte [Aals13]. Eine konkretere Modellierung kann aufgrund des Modellierungszwecks [Ober96] nicht erforderlich sein bzw. erscheint zu aufwendig. Deutlich wird dies unter anderem an der Verwendung einer Oder-Zusammenführung und der damit einhergehenden Problematik der Nicht-Lokalität einer EPK [Kind06], wodurch die Ausführung und Analyse der Geschäftsprozessmodelle problematisch ist [Aals13]. Weitere mögliche Auswahlkriterien für Modellierungssprachen werden aus den zahlreichen Vergleichsstudien in der Literatur deutlich [Aals13, DLMM+12, FiLa11, KaBe02, KhSe12, Krog12, MMRS10, MeSt08, ReDr07, MaPo06, SWMR09].

3.2 Business Process Model and Notation (BPMN)

Die Business Process Model and Notation (BPMN) ist eine graphische Modellierungssprache, die nicht nur die Geschäftsprozessmodellierung, sondern auch die Prozessimplementierung unterstützen soll. Dementsprechend wurde die Modellierungssprache mit der Intention konzipiert, dass diese für alle Anwendergruppen leicht verständlich ist. Das gilt für die Analysten, die erste Entwürfe erstellen, für die technischen Entwickler, die für die Implementierung verantwortlich sind, bis hin zu den Managern, die die Prozesse verwalten und überwachen. Im Rahmen der Spezifikation wird eine Anlehnung an das Token-Konzept der Petri-Netze (vgl. Kapitel 3.5) angestrebt, um die Kontrollflusssemantik der Syntaxelemente besser beschreiben zu können. Darüber hinaus werden teilweise die Kontrollflussmuster der Workflow Pattern Initiative [RHAM06] zur Beschreibung der Kontrollflusssemantik verwendet. Dennoch besitzt die Modellierungssprache keine präzise Kontrollflusssemantik, so dass die Modellierungssprache den semi-formalen, anwendungsneutralen Modellierungssprachen nach [Ober96] (vgl. Abbildung 7) zuzuordnen ist. Bekannte, vorhandene Modellierungssprachen sollten zudem in die Entwicklung der neuen Modellierungssprache einfließen, mit der Intention, die Vorteile von unterschiedlichen Modellierungssprachen in einer Modellierungssprache zu integrieren. Hierzu wurden die Modellierungssprachen UML-Aktivitätsdiagramm [UML2.5.1], UML-Enterprise Distributed Object Computing (EDOC) Business Processes [EDOC04], Integrated Definition (IDEF) [IDEF0, IDEF1a, IDEF1b, IDEF1X, IDEF3, IDEF4, IDEF5], electronic business eXtensible Markup Language (ebXML) [ISO15000-1, ISO15000-2, ISO15000-3, ISO15000-4, ISO15000-5], ebXML Business Process Specification Schema (BPSS) [ebXML2.0.4], RosettaNet [RosettaNet], Line of Visibility Engineering Methodology (LOVeM) [LOVeM95] und die Ereignisgesteuerte Prozesskette (EPK) [KeNS92] betrachtet [BPMI04, BPMN2.0.2, Whit04]. Im Rahmen dieser Arbeit wird die Business Process Model and Notation Version 2.0.2 [BPMN2.0.2] als Referenz zugrunde gelegt, die als MOF-konformes Schichtenmodell[1] spezifiziert ist.

Die Spezifikation beschreibt die BPMN-Diagrammtypen Prozess, Choreographie und Kollaboration. Mit einem Prozess-Diagramm können öffentliche sowie private interne nicht-ausführbare und private interne ausführbare Geschäftsprozesse modelliert werden. Die privaten internen Geschäftsprozesse beschreiben die Aktivitäten, Daten und Personen

[1] Die Meta Object Facility (MOF) ist eine Metamodellierungssprache, d. h. eine Sprache, um Metamodelle zu definieren. MOF besteht aus vier Ebenen. Dabei sind die Objekte der Realität (d. h., z. B. die Geschäftsprozessinstanz eines BPMN-Geschäftsprozessmodells) auf der MOF-Ebene 0, die Geschäftsprozessmodelle (d. h., z. B. BPMN- Geschäftsprozessmodell) auf MOF-Ebene 1, deren Meta-Modelle (d. h., z. B. BPMN-Metamodell) auf MOF-Ebene 2 sowie wiederum deren Metamodelle auf MOF-Ebene 3 (d. h., z. B. MOF). Die niedrigere Ebene ist jeweils eine Instanz der höheren Ebene [MOF2.5.1].

innerhalb eines Unternehmens. Im Kontext von Web Services wird hierbei auch von Orchestrierung der Services gesprochen. Ein interner Geschäftsprozess ist IT-basiert ausführbar, wenn die Kontrollflusssemantik mittels der Web Service - Business Process Execution Language (WS-BPEL) [WS-BPEL2.0] beschrieben wurde [BPMN2.0.2]. Andernfalls ist der Geschäftsprozess nicht IT-basiert ausführbar und wird dementsprechend zur nur Dokumentation verwendet. Öffentliche Geschäftsprozesse stellen die Interaktion von privaten Geschäftsprozessen mit anderen Geschäftsprozessen oder Stakeholdern dar. Es werden nur diejenigen Aktivitäten, Daten und Personen dargestellt, die für die Kommunikation mit den anderen Geschäftsprozessen oder Stakeholdern benötigt werden. Das Choreographie-Diagramm kann für die Modellierung des Nachrichtenaustausches verschiedener Partner verwendet werden. Die einzelnen Geschäftsprozesse der Partner werden dabei nicht modelliert und ist somit eine andere Darstellung der öffentlichen Geschäftsprozesse. Das Kollaborations-Diagramm zeigt das Zusammenspiel verschiedener Prozess-Diagramme mit Nachrichtenflüssen. Für die informelle Modellierung und Darstellung der Beziehungen des Nachrichtenaustausches der diversen Partner einer Kollaboration können Konversationen verwendet werden. Es visualisiert die Kommunikationsstrukturen in einer Vogelperspektive [BPMN2.0.2, DrKO17]. Im Folgenden werden nur die BPMN-Diagrammtypen Prozess und Kollaboration betrachtet.

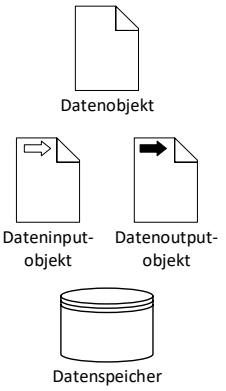
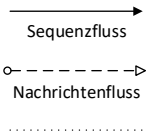
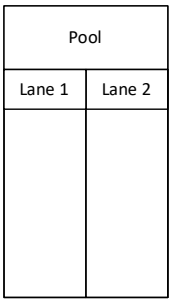
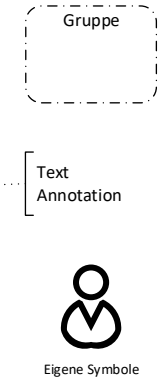
Fluss-elemente	Datenelemente	Verbindungs-elemente	Swimlane	Artefakte
				

Tabelle 1: Notationselemente - BPMN

Aufgrund der zahlreichen Notationselemente ist die Spezifikation in die fünf Basiskategorien untergliedert (vgl. Tabelle 1), die in den folgenden Abschnitten betrachtet werden. Hierzu zählen die Flusselemente (Aktivität, Ereignis und Gateway), Datenelemente

(Datenobjekt, Dateninputobjekt, Datenoutputobjekt und Datenspeicher) und Verbindungselemente (Sequenzfluss, Nachrichtenfluss und Assoziation) sowie Swimlanes (Pool und Lane) und Artefakte (Gruppe, Text Annotation sowie eigens definierte Notationselemente) [BPMN2.0.2].

3.2.1 Flusselemente

Die Flusselemente werden in Aktivität, Ereignis und Gateway untergliedert. Eine Aktivität wird für die Modellierung von Tätigkeiten verwendet. Ereignisse repräsentieren einen Zeitpunkt, an dem etwas passiert ist. Diese Elemente werden nur unter bestimmten Bedingungen (Gateways) ausgeführt [BPMN2.0.2]. Die Elemente Aktivität, Ereignis und Gateway werden in den folgenden Teilabschnitten betrachtet.

3.2.1.1 Aktivität

Aktivitäten können atomar oder nicht-atomar vorliegen und werden in *Aufgabe* (engl. task), *Teilprozess* (engl. sub-process) und *Aufruf-Aktivität* (engl. call activity) untergliedert. Eine Aufgabe (vgl. Abbildung 8a) ist eine Tätigkeit innerhalb eines Prozesses, deren Kontrollflusssemantik durch Aktivitätsmarker beeinflusst wird oder deren inhaltliche Bedeutung durch Aufgaben-Typen verfeinert werden kann. Die *Schleife* (engl. loop), *parallele* und *sequenzielle Mehrfachinstanz* (engl. multi-instance) sowie die *Kompensation* (engl. compensation) sind mögliche Aktivitätsmarker.

Eine Aufgabe, die mit dem Aktivitätsmarker *Schleife* annotiert wurde (vgl. Abbildung 8b), wird solange wiederholt, bis die definierte Abbruchbedingung (`loopCondition: Expression`) erfüllt ist. Bei der *Mehrfachinstanz* wird eine definierte Anzahl von Instanzen erzeugt (`numberOfInstances: integer`), die *parallel* (vgl. Abbildung 8c, `isSequential: boolean = false`) oder *sequenziell* (vgl. Abbildung 8d, `isSequential: boolean = true`) ausgeführt werden können. Aufgaben, die mit dem *Kompensationsmarker* (`isForCompensation: boolean = true`) gekennzeichnet sind, werden für die Rücksetzung einer bereits beendeten Aktivität verwendet [BPMN2.0.2].

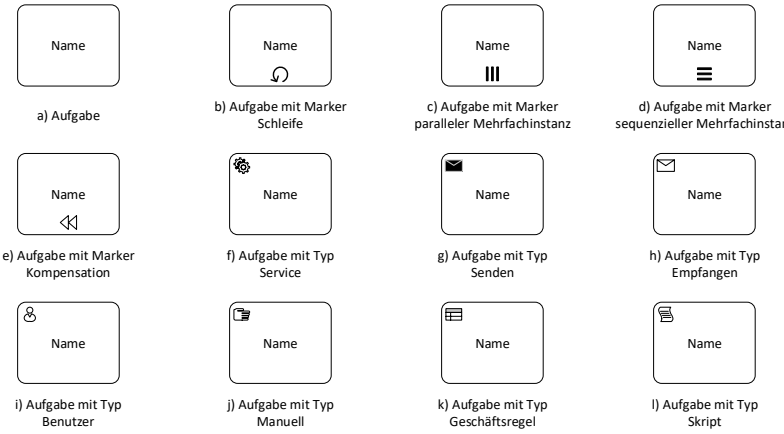


Abbildung 8: Notationselemente - BPMN-Aufgaben

Die inhaltliche Bedeutung einer Aufgabe kann mit Aufgaben-Typen konkretisiert werden. Es existieren die Aufgaben-Typen *Service* (engl. service), *Senden* (engl. send), *Empfangen* (engl. receive), *Benutzer* (engl. user), *Manuell* (engl. manual), *Geschäftsregel* (engl. business rule) und *Skript* (engl. script). Eine Aufgabe mit dem Aufgaben-Typ *Service* (vgl. Abbildung 8f) kennzeichnet die Verwendung eines Dienstes, wie beispielsweise in Form eines Web-Services oder repräsentiert die automatisierte Ausführung einer Anwendung. Eine Aufgabe mit dem Aufgaben-Typ *Senden* (vgl. Abbildung 8g) beschreibt das Versenden einer Nachricht an einen externen Prozessteilnehmer (Pool, vgl. Kapitel 3.2.4). Nach dem Versenden der Nachricht ist die Ausführung der Aufgabe beendet. Das Pendant bildet eine Aufgabe mit dem Aufgaben-Typ *Empfangen* (vgl. Abbildung 8h), welche auf das Empfangen einer Nachricht von einem anderen Prozessteilnehmer wartet. Sobald die Nachricht eingetroffen ist, ist die Aufgabe beendet. Eine Aufgabe mit dem Aufgaben-Typ *Benutzer* (vgl. Abbildung 8i) beschreibt eine Tätigkeit, die von einer menschlichen Ressource ausgeführt und durch eine Business Process Engine[2] zugewiesen wird. Mit dem Aufgaben-Typ *Manuell* (vgl. Abbildung 8j) wird eine manuelle Ausführung einer Aufgabe durch eine menschliche Ressource beschrieben. Im Unterschied zu dem Aufgaben-Typ *Benutzer* wird die Aufgabe nicht durch die Business Process Engine zugewiesen. Aufgaben können mit dem Aufgaben-Typ *Geschäftsregel* (vgl. Abbildung 8k) typisiert werden. Falls eine Geschäftsregel

[2] Eine Business Process Engine (kurz: Process Engine) ist für die Zustandsverwaltung der Geschäftsprozessinstanzen und Ausführungsüberwachung aller Interaktionen verantwortlich. Im Rahmen des Geschäftsprozessmodells müssen hierfür alle notwendigen Details beschrieben werden [Chan06].

implementiert ist, kann diese durch eine Business Rules Engine[3] ausgewertet werden. Hierbei werden Parameter übergeben (Inputs), um ein oder mehrere Ergebnisse (Output) zurückzuliefern. Ein weiterer Aufgaben-Typ ist das *Skript* (vgl. Abbildung 8l), welches die Ausführung eines Skriptes durch eine Business Process Engine ermöglicht. Dazu muss das Skript in einer Sprache vorliegen, die von der Business Process Engine verstanden wird [BPMN2.0.2].

Nicht nur Aufgaben, sondern auch Teilprozesse können verschiedene Ausprägungen aufweisen. Hierfür bietet BPMN einen *eingebetteten Teilprozess* (engl. embedded sub-process, sub-process), einen *ereignisbasierten Teilprozess* (engl. event sub-process), eine *Transaktion* (engl. transaction), einen *Ad-Hoc Teilprozess* (engl. ad-hoc sub-process) sowie eine *Aufruf-Aktivität* (engl. reusable sub-process) an. Die Teilprozesse können in aufgeklappter (vgl. Abbildung 9b, d, f, h, j) sowie in zugeklappter Form (vgl. Abbildung 9a, c, e, g, i) vorliegen, die durch Aktivitäten, Gateways und Ereignisse konkretisiert werden können (vgl. bspw. Abbildung 9b). Ein *eingebetteter Teilprozess* bzw. *Teilprozess* (vgl. bspw. Abbildung 9a, b) gruppiert Prozessausschnitte. Ein *ereignisbasierter Teilprozess* (vgl. bspw. Abbildung 9c, d) ist eine spezielle Form eines Teilprozesses und muss durch genau ein typisiertes Startereignis ausgelöst werden. Bei der Instanziierung wird abhängig vom Startereignis des Ereignis-Teilprozesses der übergeordnete Teilprozess unterbrochen oder nebenläufig ausgeführt. Im Gegensatz dazu darf ein eingebetteter Teilprozess nicht durch ein typisiertes Ereignis ausgelöst werden und ist in den Prozess durch den Sequenzfluss eingebettet. Das Startereignis wird bei einem zugeklappten ereignisbasierten Teilprozess in der oberen linken Ecke dargestellt, wie in Abbildung 9c durch das Nachrichtenereignis deutlich wird. In aufgeklappter Form ist das Startereignis als Notationselement im Geschäftsprozessmodell sichtbar (vgl. Abbildung 9d). Ein weiterer spezialisierter Teilprozess ist die *Transaktion* (vgl. Abbildung 9e, f), die eine logisch zusammengehörige Gruppe bildet, deren Ausführung erfolgreich (engl. successful completion), fehlerhaft (engl. hazard) oder fehlgeschlagen (engl. failed completion) sein kann. Endet der Kontrollfluss an einem Abbruchereignis (vgl. Kapitel 3.2.1.2), dann ist die Ausführung fehlgeschlagen, woraufhin alle Kompensationsereignisse innerhalb der Transaktion ausgelöst werden. Die Transaktion ist

[3] Eine Business Rule Engine ist eine Softwarekomponente, die für die effiziente und flexible Ausführung von Geschäftsregeln verantwortlich ist [ScGr06]. Eine Geschäftsregel ist eine Anweisung, die Aspekte in einem Unternehmen definiert oder einschränkt. Hierdurch soll ein definiertes Verhalten innerhalb des Unternehmens sichergestellt werden [CeFB00]. Die Geschäftsregeln werden als Bedingungen (engl. constraints) oder in Form von Wenn-Dann-Anweisungen (if constraint then action) formuliert [Hall02]. Die Modellierungssprache Decision Model and Notation (DMN) bietet die Möglichkeit der Formulierung solcher Anweisungen [DMN1.1]. Für die Auswertung der Geschäftsregeln werden der Business Rule Engine alle relevanten Unternehmensinformationen zur Verfügung gestellt, um auf deren Basis die Geschäftsregel auszuwerten [ScGr06].

fehlerhaft, wenn der Kontrollfluss an einem Fehlerereignis endet. Im Unterschied zu einer fehlgeschlagenen Transaktion werden die Kompensationsereignisse nicht ausgelöst. Der *Ad-Hoc Teilprozess* (vgl. Abbildung 9g, h) ist ein spezifischer Teilprozess, bei dem die Reihenfolge und die Ausführungshäufigkeit der einzelnen Aktivitäten nicht vorgeschrieben wird. Nach der Ausführung einer Aktivität wird jeweils die Beendigungsbedingung (`completionCondition: Expression`) des Ad-Hoc Teilprozess ausgewertet. Es können weitere Aktivitäten in dem Ad-Hoc Teilprozess ausgeführt werden, solange die Beendigungsbedingung nicht erfüllt ist. Die *Aufruf-Aktivität* ist ein Verweis auf einen global definierten Prozess (vgl. Abbildung 9i, j) oder eine global definierte Aufgabe (vgl. Abbildung 9k), sodass beispielsweise eine Mehrfachverwendung der Aufgabe bzw. des Teilprozesses möglich ist [BPMN2.0.2].

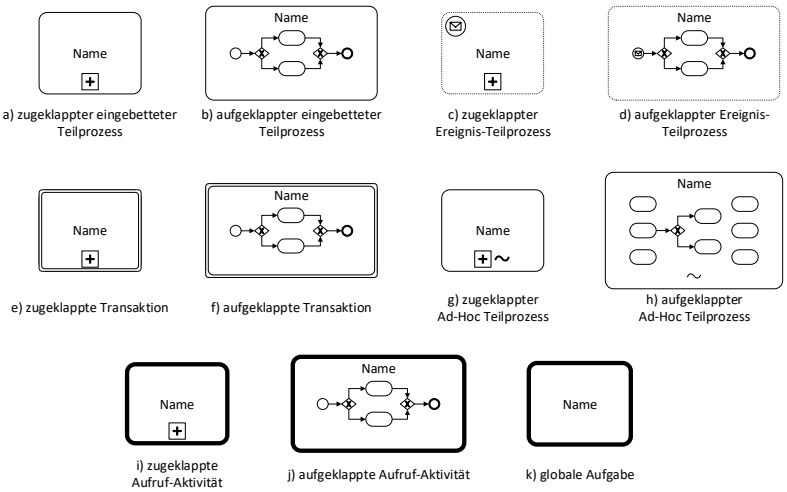


Abbildung 9: Notationselemente - BPMN-Teilprozesse

3.2.1.2 Ereignis

Ein Ereignis ist ein Zeitpunkt, an dem etwas passiert ist. Es wird zwischen Start-, End- und Zwischenereignissen differenziert. Die Startereignisse lösen einen Prozess aus, Endereignisse beenden einen Prozess oder Prozesspfad und Zwischenereignisse beeinflussen den Prozessablauf. Zusätzlich wird zwischen eingetretenen (engl. catch) und ausgelösten (engl. throw) Ereignissen unterschieden. Die eingetretenen Ereignisse wurden von einem definierten, in der Regel externen Trigger ausgelöst (Startereignisse und teilweise Zwischenereignisse). Die ausgelösten Ereignisse (teilweise Zwischenereignisse und Endereignisse) sind selbst der Auslöser [BPMN2.0.2].

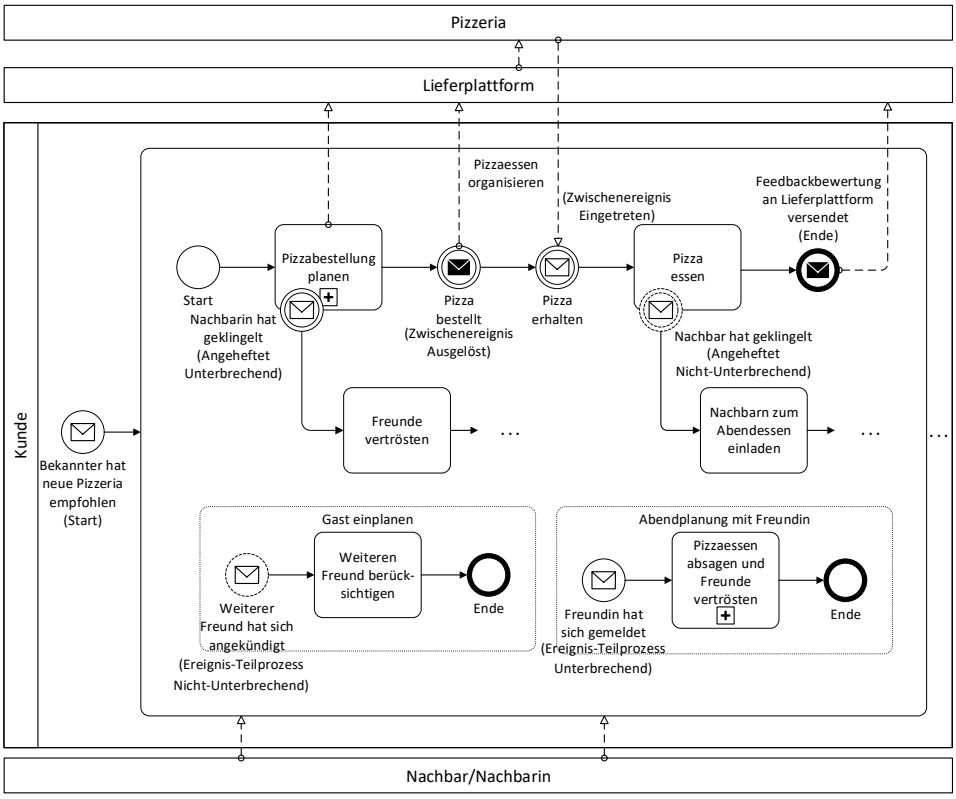


Abbildung 10: Notationselemente - BPMN-Nachrichtenereignisse

Exemplarisch sollen die verschiedenen Ausprägungen von Ereignissen anhand des *Nachrichtenereignisses* erläutert werden (vgl. Abbildung 10). Unter dem Begriff *Nachricht* wird im Rahmen der BPMN-Spezifikation jede Form der Interaktion verstanden, die einen spezifischen Adressaten hat. Der Prozess aus Abbildung 10 wird instanziiert, wenn beispielsweise ein Bekannter eine neue Pizzeria empfohlen hat (Nachrichtenereignis - Start). Im Anschluss wird die Pizzabestellung geplant, welche als zugeklappter Teilprozess modelliert wurde. Sofern während der Planung der Pizzabestellung die Nachbarin klingelt (Nachrichtenereignis - Angeheftet Unterbrechend), wird der Teilprozess Pizzabestellung planen abgebrochen und die Freunde vertröstet. Der Kontrollfluss wird an dem angehefteten Nachrichtenereignis *Nachbarin hat geklingelt* fortgeführt. Andernfalls wird nach der Planung die Pizza bei einer Lieferplattform bestellt (Nachrichtenereignis - Ausgelöst) und gewartet, bis die Bestellung an eine Pizzeria weitergegeben und die Pizza von der Pizzeria geliefert (Nachrichtenereignis - Eingetreten) wurde. Nach der Lieferung der Pizza wird die Pizza verzehrt. Während des Essens könnte es vorkommen, dass der Nachbar klingelt (Nachrichten-

ereignis - Angeheftet Nicht-Unterbrechend). Im Unterschied zur Nachbarin wird nebenläufig die Pizza gegessen und der Nachbar zum Abendessen eingeladen. Der Prozess endet, wenn eine Feedbackbewertung an die Lieferplattform versendet (Nachrichtenereignis - Ende) wurde. Während des Teilprozesses *Pizzaessen organisieren* besteht die Möglichkeit, dass sich ein weiterer Freund ankündigt (Nachrichtenereignis - Ereignis-Teilprozess Nicht-Unterbrechend). In diesem Zusammenhang wird, nebenläufig zum Teilprozess *Pizzaessen organisieren*, der weitere Freund bei der Essensplanung berücksichtigt. Wenn sich die Freundin meldet (Nachrichtenereignis - Ereignis-Teilprozess Unterbrechend), wird der Teilprozess *Pizzaessen organisieren* unterbrochen. Nachdem anschließend der Teilprozess *Pizzaessen absagen und Freunde vertrösten* abgearbeitet wurde, wird der Teilprozess *Pizzaessen organisieren* beendet. In Tabelle 2 werden die Ereignisse, die im Rahmen der Spezifikation definiert werden, in ihrer möglichen Ausprägung illustriert. Dies sind die Ereignisse *Blanko* (engl. none), *Nachricht* (engl. message), *Timer* (engl. timer), *Eskalation* (engl. escalation), *Bedingung* (engl. conditional), *Link* (engl. link), *Fehler* (engl. error), *Abbruch* (engl. cancel), *Kompensation* (engl. compensation), *Signal* (engl. signal), *Mehrfach* (engl. multiple), *Mehrfach/Parallel* (engl. parallel multiple) und *Terminierung* (engl. terminate) [BPMN2.0.2].

	Start			Zwischen				Ende
	Standard	Ereignis-Teilprozess Unterbrechend	Ereignis-Teilprozess Nicht-Unterbrechend	Eingetreten	Angeheftet Unterbrechend	Angeheftet Nicht-Unterbrechend	Ausgelöst	Standard
Blanko: Untypisierte Ereignisse,in der Regel am Start oder am Ende eines Prozesses.								
Nachricht: Empfang und Versand von Nachrichten.								
Timer: Periodische, zeitliche Ereignisse, Zeitpunkte oder Zeitspannen.								
Eskalation: Meldung an den nächsthöheren Verantwortlichen.								
Bedingung: Reaktion auf veränderte Bedingungen und Bezug auf Geschäftsregeln.								
Link: Zwei zusammengehörige Link-Ereignisse repräsentieren einen Sequenzfluss.								
Fehler: Auslösen und behandeln von definierten Fehlern.								
Abbruch: Reaktion auf abgebrochene Transaktionen oder Auslösen von Abbrüchen.								
Kompensation: Behandeln oder Auslösen einer Kompensation.								
Signal: Signal über mehrere Prozesse. Auf ein Signal kann mehrfach reagiert werden.								
Mehrfach: Eintreten eines Ereignisses von mehreren Ereignissen. Auslösen aller Ereignisse.								
Mehrfach / Parallel: Eintreten aller Ereignisse.								
Terminierung: Löst die sofortige Beendigung des Prozesses aus.								

Tabelle 2: Notationselemente - BPMN-Ereignisse [BPMN2.0.2]

3.2.1.3 Gateway

Bei der Ausführung einer Geschäftsprozessinstanz kann es erforderlich sein, dass Entscheidungen getroffen oder mehrere Flusselemente nebenläufig ausgeführt werden müssen. Hierzu liefert BPMN das Konzept der Gateways.

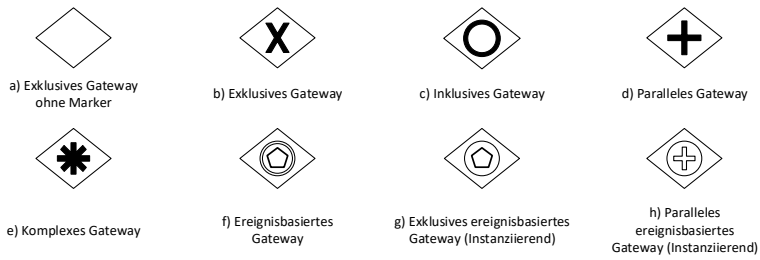


Abbildung 11: Notationselemente - BPMN-Gateway

Es wird zwischen dem *exklusiven* (engl. exclusive Gateway), *inklusiven* (engl. inclusive Gateway), *parallelen* (engl. parallel Gateway), *komplexen* (engl. complex Gateway), *exklusiven ereignisbasierten* (engl. event-based exclusive Gateway), *exklusiven ereignisbasierten instanziierenden* (engl. event-based exclusive Gateway to start a Process) und dem *parallelen ereignisbasierten instanziierenden Gateway* (engl. event-based exclusive Gateway to start a Process) differenziert. Das *exklusive Gateway* wird für eine Entweder-/Oder-Entscheidung verwendet und kann mit (vgl. Abbildung 11a) oder ohne Marker (vgl. Abbildung 11b) verwendet werden. Das *inklusive Oder* (vgl. Abbildung 11c) bildet ein logisches Oder ab. Das *parallele Gateway* (vgl. Abbildung 11d) wird für die Synchronisation oder Erzeugung von nebenläufig ausführbaren Abschnitten verwendet und entspricht einem logischen Und. Sofern sich für die Modellierung einer Verzweigung oder Zusammenführung weder das exklusive, inklusive oder parallele Gateway eignet, kann das *komplexe Gateway* (vgl. Abbildung 11e) verwendet werden. Bei einem komplexen Gateway wird im Rahmen einer Aktivierungsbedingung (`activationCondition: Expression`) das Ausführungsverhalten beschrieben. Die ereignisbasierten Gateways spielen eine Sonderrolle, sodass die Entscheidung nicht auf der Basis der vorliegenden Prozessdaten getroffen wird, sondern auf der Basis der nachfolgenden eingetretenen Ereignisse. Hierbei wird zwischen einem *exklusiven ereignisbasierten Gateway* innerhalb eines Prozesses (vgl. Abbildung 11f) oder der Instanziierung eines Prozesses (vgl. Abbildung 11g), d. h. eines *exklusiven instanziierenden ereignisbasierten Gateways*, differenziert. Sofern die Instanziierung des Prozesses erfordert, dass alle nachfolgenden Ereignisse eingetreten sind, kann das *ereignisbasierte instanziierende parallele Gateway* (vgl. Abbildung 11h) verwendet werden [BPMN2.0.2].

3.2.2 Datenelemente

Die Kategorie der Datenelemente wird für die Modellierung von Prozessinformationen verwendet, die im Rahmen der Ausführung des Prozesses typischerweise von Aktivitäten erzeugt, verarbeitet oder gespeichert wird.

Für die Modellierung der Datenaspekte werden Datenobjekte (engl. Data Object) zur Verfügung gestellt. Diese Datenobjekte werden über Assoziationen mit den einzelnen Aktivitäten oder Prozessen verbunden. Für die Datenmodellierung können *Datenobjekte* (engl. data object), *Listendatenobjekte* (engl. data object with list items), *Dateninputobjekte* (engl. data inputs), *Datenoutputobjekte* (engl. data outputs) und *Datenspeicher* (engl. data store) verwendet werden. Ein *Datenobjekt* (vgl. Abbildung 12a) kennzeichnet erforderliche Informationen in einem Geschäftsprozess. Sofern gekennzeichnet werden soll, dass mehrere Elemente im Datenobjekt (`isCollection: Boolean = true`) vorliegen, wird das *Listendatenobjekt* (vgl. Abbildung 12b) verwendet. Für die Ausführung von Geschäftsprozessen können externe Daten erforderlich sein sowie Datenoutputs generiert werden. Für diese spezielle Darstellung werden *Dateninput*- (vgl. Abbildung 12c) und *Datenoutputobjekte* (vgl. Abbildung 12d) verwendet. Die Modellierung einer persistenten Datenspeicherung kann mit einem *Datenspeicher* (vgl. Abbildung 12e), unabhängig vom Status des Prozesses, modelliert werden [BPMN2.0.2].

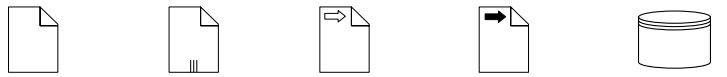


Abbildung 12: Notationselemente - BPMN-Datenelemente

3.2.3 Verbindungselemente

Die Flusselemente innerhalb eines Pools (vgl. Kapitel 3.2.4) werden über einen Sequenzfluss miteinander verbunden. Andernfalls muss der Nachrichtenfluss verwendet werden. Die Assoziation verbindet Artefakte mit anderen Notationselementen, um z. B. zusätzliche Prozessinformationen bereitzustellen. Artefakte haben jedoch keinen Einfluss auf den Kontrollfluss. Die Assoziation verknüpft auch die Datenelemente mit den Aktivitäten.

Verbindungselemente beschreiben die Zusammenhänge im Geschäftsprozessmodell. Die *Sequenzflüsse* (vgl. Abbildung 13a) werden für die logisch-kausalen Zusammenhänge der Flusselemente innerhalb eines Geschäftsprozesses verwendet. Zusätzlich besteht die

Möglichkeit, dass der Sequenzfluss an eine *Bedingung* (vgl. Abbildung 13b, `conditionExpression: Expression`) gekoppelt wird. Dementsprechend kann der Kontrollfluss erst fortgeführt werden, wenn die Bedingung erfüllt ist. Sofern alle Bedingungen an einem Gateway oder einer Aktivität als falsch ausgewertet werden, der Kontrollfluss aber trotzdem fortgeführt werden soll, kann der *default Sequenzfluss* (vgl. Abbildung 13c) verwendet werden. Der *Nachrichtenfluss* (vgl. Abbildung 13d) wird bei Kollaborations-Diagrammen verwendet, um den Nachrichtenaustausch zwischen verschiedenen Teilnehmern (Pools) zu modellieren. Für die konkrete Modellierung der Nachricht eines Nachrichtenflusses können die Nachrichtenflüsse mit (symbolisierten) Briefumschlägen annotiert werden. Dabei wird zwischen *instanziierenden* (vgl. Abbildung 13e) und *nicht-instanziierenden* (vgl. Abbildung 13f) Nachrichten differenziert. Eine Assoziation kann zum einen zur Verknüpfung von Artefakten mit anderen Notationselementen (vgl. Abbildung 13g) verwendet werden und zum anderen als gerichtete Assoziation zur Verknüpfung von Flusselementen mit Datenelementen (vgl. Abbildung 13h) sowie zur Verbindung von Kompensationsereignissen mit Kompensationsaktivitäten. Eine Assoziation kann auch *ungerichtet* oder *bidirektional* (vgl. Abbildung 13i) vorliegen [BPMN2.0.2].

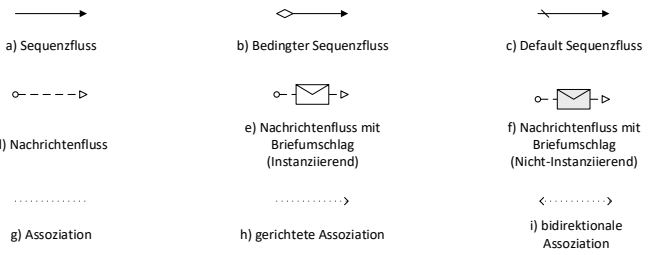


Abbildung 13: Notationselemente - BPMN-Verbindungselemente

3.2.4 Swimlane

Im Rahmen der BPMN besteht die Möglichkeit, dass die Ausführung von Flusselementen an eine bestimmte Lane gekoppelt ist. Mit einer Lane können beispielsweise die Aktivitäten einem Stakeholder zugeordnet werden. Hierfür wird das Konzept der Swimlanes (vgl. Abbildung 15) zur Verfügung gestellt. Während ein Prozess im Sinne der BPMN-Spezifikation nur aus einem Pool und beliebig vielen Lanes besteht, müssen zur Modellierung einer Kollaboration mindestens zwei Pools zugrunde liegen. Ein Pool kann auch mit dem Aktivitätsmarker *parallele Mehrfachinstanz* markiert werden, wodurch der Pool mehrfach instanziiert wird [BPMN2.0.2].

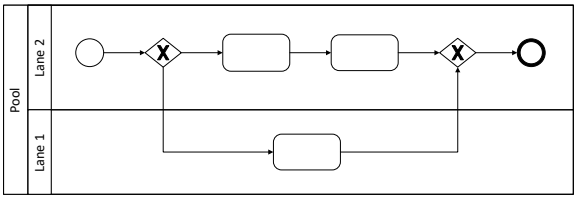


Abbildung 14: Notationselemente - BPMN-Pool

3.2.5 Artefakte

Die fünfte Basiskategorie sind die Artefakte, welche zur Beschreibung von zusätzlichen Prozessinformationen verwendet werden können, die den Kontrollfluss nicht beeinflussen. In diesem Zusammenhang können Gruppen (vgl. Abbildung 15a) zur Gruppierung von Elementen oder Text Annotationen (vgl. Abbildung 15b) für ergänzende Hinweise verwendet werden [BPMN2.0.2].

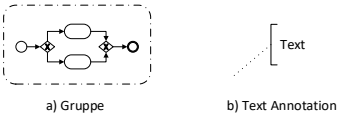


Abbildung 15: Notationselemente - BPMN-Artefakte

3.3 Ereignisgesteuerte Prozesskette (EPK)

Die Ereignisgesteuerte Prozesskette (EPK) ist eine graphische, semi-formale anwendungsneutrale Modellierungssprache. Das Ziel bei der Entwicklung war, ein einfaches und intuitives Verfahren zur Dokumentation von Geschäftsprozessen zu schaffen [KeNS92]. Diese Modellierungssprache wurde in Anlehnung an die Petri-Netze konzipiert [Sche94]. Durch die Integration in das SAP-Referenzmodell [Kell99] fand die Ereignisgesteuerte Prozesskette in der Praxis weite Verbreitung [NüRu02]. Jedoch wird die Ereignisgesteuerte Prozesskette zunehmend durch die BPMN abgelöst, was unter anderem auf einen fehlenden, einheitlichen Standard der EPKs zurückzuführen ist [Allw15]. Das Forschungsprojekt SPEAK[4] (Projektlaufzeit 2015-2017), gefördert durch das Bundesministerium für Wirtschaft und

[4] https://www.rlp-forschung.de/public/projects/20686 und
https://www.epk-norm.uni-osnabrueck.de/wiki/Towards_EPC_standardization

Technologie (BMWi), sollte durch den Entwurf und die Realisierung einer Spezifikation zur Normung der Ereignisgesteuerten Prozesskette beitragen [JKRT+16].

Die Modellierungssprache EPK besteht in ihrer ursprünglichen Form aus dem Ereignis (engl. event), der Funktion (engl. function), Verknüpfungsoperatoren (engl. logical connector) sowie dem Kontrollfluss (engl. control flow). Es existieren auch eine Reihe von Erweiterungen, wie die erweiterte Ereignisgesteuerte Prozesskette (eEPK) [KeTe97] oder die objektorientierte Ereignisgesteuerte Prozesskette (oEPK) [ScNZ97]. Andere Erweiterungen sind z. B. in [Rose96, LoAl98, NüFZ98] zu finden, eine Übersicht wird in [JKRT+16] aufgezeigt. Die Elemente Ereignis, Funktion und Kontrollfluss sowie die Verknüpfungsoperatoren [KeNS92] und die Prozessschnittstelle (engl. Process Interface) der eEPK [KeTe97] werden im Folgenden vorgestellt. Die Notationselemente werden in Abbildung 16 illustriert.

Abbildung 16: Notationselemente - EPK

Eine *Funktion* modelliert eine auszuführende Aktivität, um ein definiertes Ziel innerhalb einer vordefinierten Zeitspanne zu erreichen. Die Funktion ist eine aktive Komponente, die einen definierten Eingangszustand benötigt und in einen vorgegebenen Zielzustand transformiert (vgl. Abbildung 16a). Das Pendant zu einer Funktion ist das *Ereignis*, welches einen Zustand darstellt und die passive Komponente einer EPK ist. Dementsprechend kann ein Ereignis einen Geschäftsprozess auslösen (z. B. Buch geliefert) oder eine Veränderung (z. B. ISBN wurde geändert) darstellen (vgl. Abbildung 16b). Die Verknüpfungsoperatoren verzweigen oder führen den Kontrollfluss wieder zusammen (vgl. Abbildung 16c). Es wird zwischen einem *Entweder-Oder-* (XOR), *Und-* (AND) sowie einem *Oder-Verknüpfungsoperator* (OR) differenziert. Der Kontrollfluss (vgl. Abbildung 16d) modelliert die logisch-kausalen Zusammenhänge [KeNS92]. Eine *Prozessschnittstelle* (vgl. Abbildung 16e) kann verwendet werden, um auf eine andere EPKs zu verweisen.

3.4 UML-Aktivitätsdiagramm (UML-AD)

Die Unified Modeling Language (UML) ist eine Modellierungssprache zur Spezifikation, Konstruktion und Dokumentation von Eigenschaften eines Systems. Ihren Ursprung hat die UML 1994 durch die Zusammenführung der Konzepte der Object Modeling Technique

(OMT)[5] [RBPE+91] und der Booch-Methode[6] [Booc91]. Ein Jahr später wurde die Object-Oriented Software Engineering-Methode (OOSE) [JaCJ92] integriert, welche zur Modellierung von Telekommunikationssystemen entwickelt wurde [HKKR05]. Das Besondere an dem Ansatz von [JaCJ92] war, dass der Prozess im Vordergrund stand und nicht die Klasse oder die statische Struktur. Demzufolge wurde im Kontext der UML erstmals die Verbindung zwischen der Softwareentwicklung und der Geschäftsprozessmodellierung hergestellt [Burk99]. Die aktuellste UML Version ist die Version 2.5.1 [UML2.5.1] aus dem Jahr 2017. Im Rahmen dieser Arbeit wird die Spezifikationsversion 2.5.1 [UML2.5.1] verwendet, welche 14 Diagrammtypen definiert, die in Struktur- und Verhaltensdiagramme untergliedert werden können. Die Strukturdiagramme beschreiben die statische Struktur von Objekten, d. h. unabhängig von der Zeit. Die Verhaltensdiagramme bilden das dynamische Verhalten von Objekten in einem System ab [UML2.5.1]. In Abbildung 17 wird ein Überblick sowie eine kurze Beschreibung zu den diversen UML-Diagrammtypen aufgezeigt.

Das UML-Aktivitätsdiagramm zählt zu den Verhaltensdiagrammen (vgl. Abbildung 18) und ist eine graphische Modellierungssprache zur Modellierung von Geschäftsprozessen [UML2.5.1]. In der Spezifikation wird keine präzise Kontrollflusssemantik definiert, sondern nur ein Token-Konzept (vgl. Kapitel 3.5) eingeführt, das punktuell zur einfacheren Beschreibung der Kontrollflusssemantik verwendet wird [UML2.5.1]. Dementsprechend ist die Modellierungssprache den semi-formalen, anwendungsneutralen Modellierungssprachen nach [Ober96] zuzuordnen (vgl. Abbildung 7). Im Folgenden werden die Elemente des Aktivitätsdiagramms kurz vorgestellt. Ein Aktivitätsdiagramm besteht aus einer *Aktivität* (engl. activity), die ein bestimmtes Verhalten beschreibt, wie beispielsweise *Rechteckfläche berechnen*. Eine Aktivität kann Ein- (z. B. Höhe und Breite) und Ausgangsparameter (z. B. Fläche), aber auch Vor- (z. B. Breite ≥ 0 und Höhe ≥ 0) und Nachbedingungen (z. B. Fläche ≥ 0) besitzen (vgl. Abbildung 18) [RuQu12, UML2.5.1].

[5] Das Konzept von [RBPE+91] ermöglicht die Modellierung von komplexen Objekten in Verbindung zur Datenmodellierung auf der Grundlage der objektorientierten Erweiterung des Entity-Relationship-Modells von [Chen76].

[6] Das Konzept von [Booc91] wurde für die Modellierung von Echtzeitsystemen und Programmierkonzepten entwickelt.

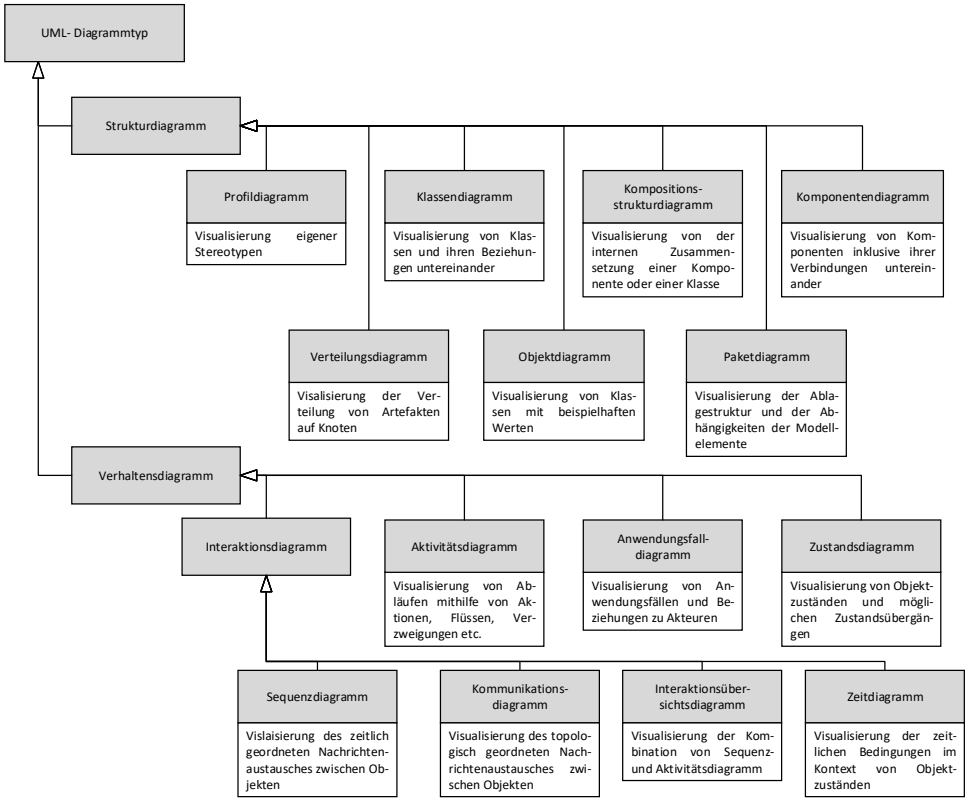


Abbildung 17: Überblick - UML-Diagrammtypen [DrKO17]

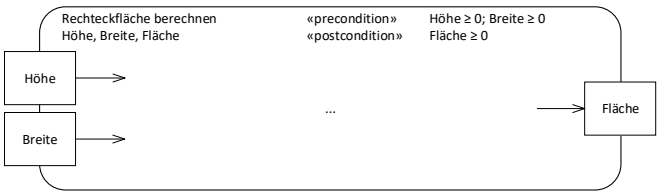


Abbildung 18: Notationselement - UML-Aktivitätsdiagramm - Aktivität (In Anlehnung an [RuQu12])

Die Aktivität kann durch Aktivitätsknoten (engl. activity nodes) spezifiziert und mithilfe von Aktivitätsgruppen (engl. activity groups) strukturiert werden. *Aktivitätsknoten* werden in *ausführbare Knoten* (engl. executable nodes), *Kontrollknoten* (engl. control nodes) und *Objektknoten* (engl. object nodes) untergliedert. Bei der *Aktivitätsgruppe* wird zwischen einem *Aktivitätsbereich* (engl. activity partitions) und *Unterbrechungsbereich* (engl. interruptible activity regions) differenziert. Die Notationselemente werden in der Spezifikation

nicht explizit vorgegeben, entgegen der Notationselemente von BPMN. Dadurch soll die Möglichkeit gegeben werden, bereits etablierte Notationselemente in der entsprechenden Domäne zu verwenden. Dennoch werden Symbole für die Notationselemente eines UML-Aktivitätsdiagramms vorgeschlagen. Für die Beschreibung der Ablaufreihenfolge der einzelnen Aktivitätsknoten innerhalb einer Aktivität werden *Kanten* (engl. edges) verwendet. Es wird zwischen *Kontroll-* (engl. control flow) und *Objektflüssen* (engl. object flow) unterschieden [UML2.5.1]. Die Vorschläge für die Notationselemente aus der Spezifikation werden in Abbildung 19 bis Abbildung 22 illustriert, auf deren Bedeutung nachfolgend eingegangen wird.

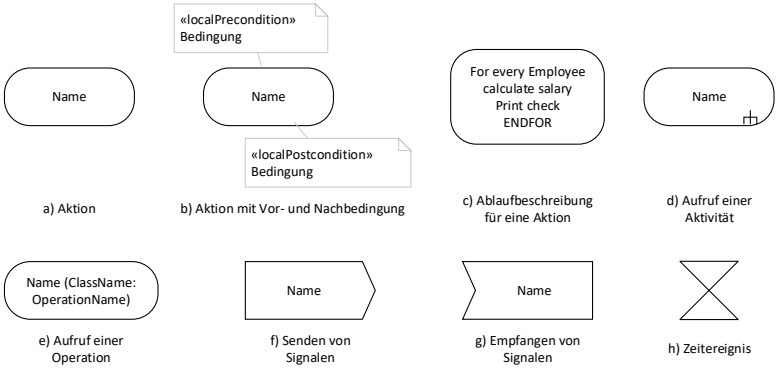


Abbildung 19: Notationselemente - UML-Aktivitätsdiagramm - Ausführbare Knoten

Die *ausführbaren Knoten* bestehen aus: *Aktion* (engl. action), *Aufruf-Aktion* (engl. invocation action), *Objekt-Aktion* (engl. object action), *Link-End Daten* (engl. link end data), *Link-Aktion* (engl. link action), *Linkobjekt-Aktion* (engl. link object action), *Strukturmerkmal-Aktion* (engl. structural feature action), *Variablen-Aktion* (engl. variable actions), *Empfangsaktion* (engl. accept event action), *strukturierte Aktion* (engl. structured action), *Mengenverarbeitung* (engl. expansion region), *reduzierte Aktion* (engl. reduce action) und *Fehlerbehandlungsaktion* (engl. raise exception action) [UML2.5.1]. Im Folgenden werden nur ausgewählte ausführbare Knoten detaillierter betrachtet, wobei alle Elemente in der UML-Spezifikation [UML2.5.1] oder auch in [OeSc13] vorgestellt werden.

Eine *Aktion* (vgl. Abbildung 19a) beschreibt eine atomare Einheit innerhalb einer Aktivität, die zur Realisierung des beschriebenen Verhaltens der Aktivität beiträgt. Dementsprechend ist eine Aktion für die Bearbeitung eines Schrittes oder für die Transformation von Daten verantwortlich. Bevor eine Aktion ausgeführt werden kann, besteht die Möglichkeit Vor- bzw. Nachbedingungen zu definieren, welche in Kommentarfeldern mit dem

Schlüsselwort *localPrecondition* bzw. *localPostcondition* gekennzeichnet sind und mit der Aktion verbunden werden können (vgl. Abbildung 19b). Aktionen können zudem Ablaufbeschreibungen in Form von Pseudocode enthalten, der auch mit der Object Constraint Language (OCL) beschrieben werden kann [OCL2.4] (vgl. Abbildung 19c) [UML2.5.1].

Die *Aufruf-Aktion* kann weiter untergliedert werden in *Aufruf einer Aktivität* (engl. call behavior action), *Aufruf einer Operation* (engl. call operation action), *Senden von Signalen* (engl. send signal) und *Senden eines Objekts* (engl. send object action). Der *Aufruf einer Aktivität* wird zur Hierarchisierung von Prozessen sowie zur Gewährleistung einer besseren Übersicht der Teilprozesse verwendet (vgl. Abbildung 19d). Der *Aufruf einer Operation* kann z. B. eine Methode aus einem Klassendiagramm aufrufen (Abbildung 19e). Beim *Senden von Signalen* bzw. *Senden eines Objekts* (vgl. Abbildung 19f) wird aus den Eingabedaten des Signalereignisses ein Signal erzeugt, welches als Broadcast versendet wird. Nachfolgend werden alle empfangenen Signale (vgl. Abbildung 19g) aktiv, für die das Signal bestimmt ist. Eine Empfangsaktion wird weiter in *Empfangsereignis* (engl. accept event action) und *Zeitereignis* (engl. accept time event action) untergliedert. Das *Empfangsereignis* bildet das Pendant zum Senden von Signalen, da es Signale empfängt und aktiv wird, sobald ein entsprechendes Signal empfangen wurde. Es besteht aber auch die Möglichkeit, dass ein Ereignis nicht durch das Empfangen eines Signals, sondern durch ein *zeitliches Ereignis* ausgelöst wird (vgl. Abbildung 19h) [UML2.5.1].

Die zweite Elementgruppe der Aktivitätsknoten sind die *Kontrollknoten*, die den Kontroll- bzw. Objektfluss zwischen den ausführbaren Knoten steuern. Kontrollknoten sind *Startknoten* (engl. initial node), *Aktivitätsendknoten* (engl. activity final node), *Ablaufende* (engl. flow final node), *Aufspaltung* (engl. fork node), *Synchronisation* (engl. join node), *Entscheidungs-* (engl. decision node) und *Zusammenführungsknoten* (engl. merge Node) [UML2.5.1].

Der *Startknoten* (vgl. Abbildung 20j) modelliert den Anfang einer Aktivität. Das Ende einer Aktivität wird durch einen *Aktivitätsendknoten* (vgl. Abbildung 20k) dargestellt, wodurch alle Aktionen beendet werden. Sofern nicht die gesamte Aktivität beendet werden soll, muss das Element *Ablaufende* (vgl. Abbildung 20l) verwendet werden. Die *Aufspaltung* (vgl. Abbildung 20f) und *Synchronisation* (vgl. Abbildung 20g) kann mit einer Und-Verzweigung bzw. -Zusammenführung verglichen werden. Durch eine Join-Spezifikation (engl. joinSpec) besteht zudem die Möglichkeit, die Synchronisation zu modifizieren, wie dies beispielsweise in Abbildung 20i dargestellt wird. Des Weiteren existiert ein *Entscheidungsknoten* (vgl. Abbildung 20a), der eine eingehende und mehrere ausgehende Kanten besitzt. Durch die Bedingungen wird überprüft, an welchem Element im Nachbereich der Kontrollfluss fortgesetzt werden soll. Die Entscheidungsrelevanten Informationen werden durch eine

vorgelagerte Aktion geliefert. Am Entscheidungsknoten wird dann eine definierte Bedingung evaluiert. Die Bedingung wird durch einen Kommentar mit dem Schlüsselwort *decisionInput* modelliert und mit dem Entscheidungsknoten verknüpft (vgl. Abbildung 20b). Darüber hinaus kann eine Entscheidung einen Input benötigen, wie z. B. die Angabe der Anzahl der Teilnehmer einer Vorlesung, welches an der eingehenden Kante mit *decisionInputFlow* gekennzeichnet wird (vgl. Abbildung 20c). Logisch entspricht diese Entscheidung einem exklusiven Oder (XOR). Der *Zusammenführungsknoten* (vgl. Abbildung 20d) führt Objekt- bzw. Kontrollflüsse als exklusives Oder (XOR) zusammen. Die Elemente Entscheidung und Zusammenführung (vgl. Abbildung 20e) sowie Synchronisation und Aufspaltung (vgl. Abbildung 20h) können auch in einem Element kombiniert werden [UML2.5.1].

Die *Objektknoten* (engl. object nodes) bilden die dritte Elementklasse der Aktivitätsknoten und beschreiben die Objektknoten generell sowie die spezialisierten Objektknoten als *Aktivitätsparameterknoten* (engl. activity parameter nodes), *Pufferknoten* (engl. central buffer nodes), *Datenspeicher* (engl. data store node) und *Pin* (engl. pin). Die Objektknoten beinhalten Token während der Ausführung einer Aktivität, um Daten oder Werte innerhalb einer Aktivität zu transportieren. Die *Aktivitätsparameterknoten* sind ausschließlich Eingabe- bzw. Ausgabeparameter und bilden daher eine Ausnahme. Ein Objektknoten besitzt keine Instanzen, sondern ist ein logischer Stellvertreter für die entsprechenden Ausprägungen eines bestimmten Wertes. Darüber hinaus können Objektknoten (vgl. Abbildung 21a) verschiedene Eigenschaften aufweisen, wie *Objektknoten mit spezifiziertem Zustand* (engl. object node for tokens containing objects in specific states, vgl. Abbildung 21b), *Objektknoten mit einer Obergrenze für Tokens* (engl. object node with a limited upper bound, vgl. Abbildung 21c), *Objektknoten mit einer Sortierung* (engl. object node with ordering, Arten: unordered, first in first out (FIFO), last in first out (LIFO), ordered, vgl. Abbildung 21d) und *Objektknoten mit einer Auswahlspezifikation* (engl. object node with selection specification, vgl. Abbildung 21e). Der *Zentralspeicherknoten* (vgl. Abbildung 21f) hat die Eigenschaft eine beliebige Anzahl an Token temporär zu speichern, bis diese nachgelagert benötigt werden. Der Zentralspeicherknoten wird mit dem Schlüsselwort *CentralBuffer* gekennzeichnet. Im Gegensatz dazu persistiert der *Datenspeicherknoten* (vgl. Abbildung 21g) alle Objekttoken und leitet lediglich Kopien an die ausgehenden Kanten weiter [UML2.5.1].

Die *Aktivitätsgruppe* besteht aus dem *Aktivitätsbereich* (engl. activity partitions) und dem *Unterbrechungsbereich* (engl. interruptible activity regions). Der *Aktivitätsbereich* gruppiert einzelne Elemente einer Aktivität mit gemeinsamen Eigenschaften. Der Aktivitätsbereich hat keinen Einfluss auf die Kontrollflusssemantik. Beispielsweise können alle Aufgaben, die in einer Reisekostenabteilung bearbeitet werden, dem Aktivitätsbereich Reisekostenabteilung zugeordnet werden. Eine einzelne Partition (vgl. Abbildung 22a), die

multidimensionale hierarchische Anordnung (Abbildung 22b) oder die hierarchische Anordnung (Abbildung 22c) sind mögliche Notationselemente zur Modellierung einer Partition. Analog besteht die Möglichkeit, die Aktivitätsbereiche in die Aktionsknoten zu integrieren (vgl. Abbildung 22d). Partitionen, die eigentlich nicht zum Betrachtungsbereich gehören, aber dennoch modelliert werden sollen, werden mit dem Schlüsselwort *external* gekennzeichnet (vgl. Abbildung 22e). Der *Unterbrechungsbereich* dahingegen hat Einfluss auf die Kontrollflusssemantik und kann zur Modellierung von Ausnahmen verwendet werden. Sobald ein Token den Unterbrechungsbereich (vgl. Abbildung 22f und Abbildung 22g) über die Unterbrechungskante (vgl. Abbildung 23d) verlässt, werden alle Token innerhalb des Unterbrechungsbereichs verworfen und die entsprechenden Aktionen abgebrochen [UML2.5.1].

Abschließend werden die diversen Arten von Kanten beschrieben. Hierbei wird zwischen ungewichteten (vgl. Abbildung 23a) und gewichteten (vgl. Abbildung 23b und Abbildung 23c) Kanten differenziert. Das Gewicht beschreibt die benötigte Anzahl der Token für die Aktivierung der nachfolgenden Aktion. Weiterführend besteht die Möglichkeit, dass eine Kante durch die Konnektornotation verknüpft wird, wie in Abbildung 23e illustriert wird. Zusätzlich wird zwischen einer Kontroll- (vgl. Abbildung 23f) und Objektflusskante mit und ohne Pin (vgl. Abbildung 23g und Abbildung 23h) differenziert.

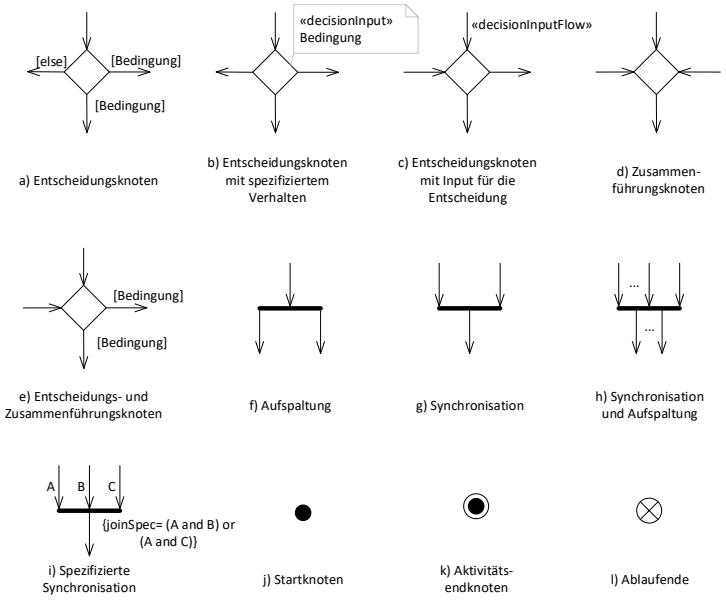


Abbildung 20: Notationselemente - UML-Aktivitätsdiagramm - Kontrollknoten

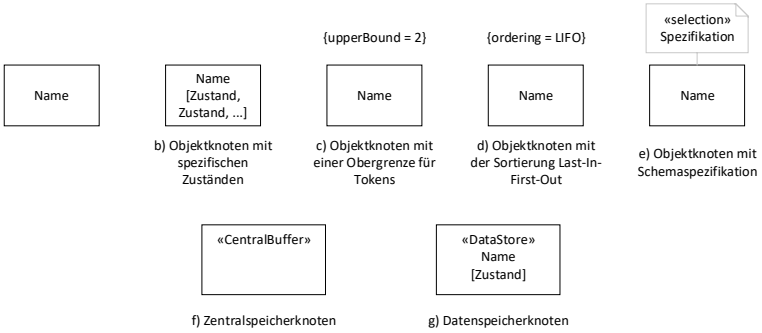


Abbildung 21: Notationselemente - UML-Aktivitätsdiagramm - Objektknoten

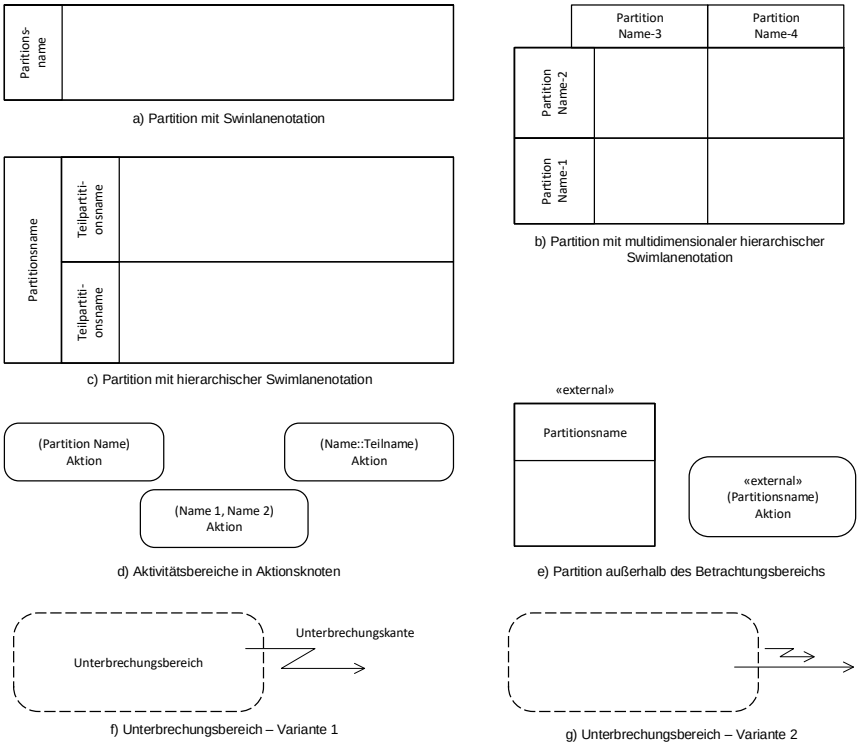


Abbildung 22: Notationselemente - UML-Aktivitätsdiagramm - Aktivitätsgruppe

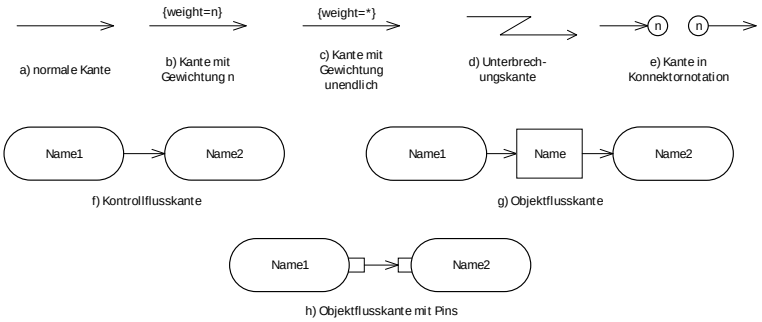


Abbildung 23: Notationselemente - UML-Aktivitätsdiagramm - Kanten

3.5 Petri-Netz (PN)

Das Petri-Netz (engl. petri net) ist ein graphischer Formalismus, um z. B. Geschäftsprozesse zu dokumentieren und darauf aufbauend eine Ablaufsimulation oder auch Simulationsexperimente zu ermöglichen [Pete81, Reis90, Ober96]. Die Modellierungssprache ist den plattformunabhängigen und formalen Modellierungssprachen nach [Ober96] zuzuordnen (vgl. Abbildung 7). Die ersten Ansätze bzw. das Aktionsnetz (vgl. Abbildung 24a) gehen auf die Dissertation von [Petr62] zurück. Mögliche Elemente eines Petri-Netzes sind die *Transition* (engl. transition, events), *Stelle* (engl. state, place), *Token* und *Flussrelation* (engl. flow relation).

Eine *Transition* ist eine atomare Aktion zur Modellierung von Ereignissen oder Aktivitäten, die den Zustand einer Geschäftsprozessinstanz verändern kann. Bei einem Zustandsübergang wird das Token in der Stelle geändert, was auch als Schalten bezeichnet wird. Die *Stelle* ist der Wirkungsbereich der Transitionen. Nur durch das Schalten einer Transition kann das Token verändert werden. Eine Stelle ist ein statisches Objekt und kann zur Darstellung von Dokumenten, Ressourcen, Daten o. ä. verwendet werden. Die aktuelle Belegung der Token in den Stellen wird als Zustand bezeichnet. Sofern das Token in der Stelle vorliegt, dann trifft der Zustand zu (z. B. Dokument liegt vor). Falls das Token nicht in der Stelle ist, dann trifft der Zustand nicht zu (z. B. Dokument liegt nicht vor). Die *Flussrelation* beschreibt den logisch-kausalen Zusammenhang zwischen den Stellen und Transitionen. Die Notationselemente in [Petr62] sind Kreise, Kanten sowie schwarze Punkte. Ein Kreis ist das Notationselement einer Stelle. Das Token wird durch einen schwarzen Punkt in der Stelle modelliert. Die Transitionen sind in [Petr62] durch die Kanteninschriften modelliert. Mit einem Petri-Netz können sequentiell, sich gegenseitig ausschließende und nebenläufige Aktivitäten modelliert werden [Pete81, Ober96].

In der Literatur existieren verschiedene Varianten von Petri-Netzen. Das Netz von [Petr62] entspricht in seinen Grundelementen [Petr76] einem Bedingungs/Ereignis-System. Eine Transition wird heute jedoch nicht mehr als Kanteninschrift (vgl. Abbildung 24a) modelliert, sondern durch ein Viereck [Petr73] (vgl. Abbildung 24b) oder auch durch einen Balken [Holt68] (vgl. Abbildung 24c). In der Literatur wird zwischen einem Netz und einem System differenziert [Reis90]. Ein System ergänzt ein Netz mit Marken, um die Systemdynamik zu modellieren [Thia87, DeRe98, RoEn98, Reis90]. Das Token entspricht dem Konzept der Marken in einem Bedingungs/Ereignis-System, da eine Stelle nach [Petr62] höchsten eine Marke aufweisen darf. Die Bedingungen entsprechen den Stellen und die Ereignisse den Transitionen. Eine Verallgemeinerung ist das Stellen/Transitions-System, das die Beschränkung der Marken in Stellen aufhebt [Pete81]. Dementsprechend müssen für die Stellen zusätzlich Kapazitäten sowie Kantengewichte eingeführt werden [Pete81, Reis90, Ober96].

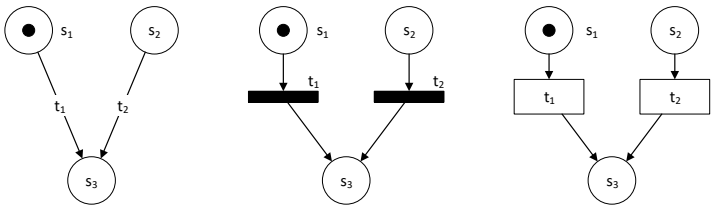


Abbildung 24: Notationselemente - Petri-Netz [Pati06]

Definition Petri-Netz:

Ein Petri-Netz (synonym: Netz) ist ein Tripel $N = (S, T, F)$, mit S als Menge der Stellen, T als Menge der Transitionen und F als Flussrelation, mit den Eigenschaften:

i) $S \cap T = \emptyset$, d. h. Stellen und Transitionen sind zwei disjunkte Mengen.

ii) $S \cup T \neq \emptyset$, d. h. es muss mindestens eine Stelle oder Transition existieren.

iii) $F \subseteq (S \times T) \cup (T \times S)$ ist die Flussrelation, sie verbindet ausschließlich Stellen mit Transitionen und Transitionen mit Stellen.

[Pete81, Reis90, Ober96]

Ein Stellen/Transitions-System ist ein markiertes Netz. Nach [Petr76] wird das Netz um die Markierung $M: S \rightarrow \mathbb{N}$ mit $\forall s \in S: M(s) \leq K(s)$ erweitert. Hierbei wird jeder Stelle eine nicht-negative Markenanzahl zugeordnet. Die Markenanzahl einer Stelle muss kleiner sein als die maximal zulässige Markenanzahl der Stelle. Die maximal zulässige Markenanzahl einer Stelle wird durch die Kapazitätsfunktion $K: S \rightarrow \mathbb{N} \cup \{\infty\}$ festgelegt. Die

Kantengewichte in einem Stellen/Transitions-System $W: F \rightarrow \mathbb{N}$ müssen für die Kanten angegeben werden [Petr76, DrKO17].

Definition Stellen/Transitions-System:

Ein Stellen/Transitions-System ist ein 6-Tupel $STS = (S, T, F, K, W, M)$, mit den Eigenschaften:

i) (S, T, F) ist ein Petri-Netz.

ii) $K: S \rightarrow \mathbb{N} \cup \{\infty\}$, d. h. jeder Stelle wird eine Kapazität zugewiesen.

iii) $W: F \rightarrow \mathbb{N}$, d. h. jeder Kante wird ein Kantengewicht zugeordnet.

iv) $M: S \rightarrow \mathbb{N}_0$ mit $\forall s \in S: M(s) \leq K(s)$, d. h. jeder Stelle wird eine Anzahl an Marken zugeordnet, die kleiner sein muss, als die Kapazität der Stelle. M_0 ist die sogenannte Startmarkierung.

[Pete81, Reis90, Ober96]

Eine Zustandsänderung für ein System erfolgt durch das Schalten von Transitionen [Pete81, Reis90]. Bei einem Zustandsübergang in einem Bedingungs/Ereignis-System werden in einem unteilbaren Schritt aus allen Eingangsbedingungen die Marken entnommen und bei allen Ausgangsbedingungen wird jeweils eine Marke hinzugefügt. Die Eingangsbedingungen werden als Vorbereich und die Ausgangsbedingungen als Nachbereich der jeweiligen Transition bezeichnet.

Definition Vor- und Nachbereich:

Für eine Stelle oder Transition $x \in S \cup T$ eines Petri-Netzes N ist:

i) $\bullet x := \{y \mid (y, x) \in F\}$ der Vorbreich von x

ii) $x\bullet := \{y \mid (x, y) \in F\}$ der Nachbereich von x

[Pete81, Reis90]

Beim Schalten einer Transition in einem Stellen/Transitions-System werden die Marken entsprechend der jeweiligen Kantengewichte aus dem Vorbereich der Transition entnommen und in den Nachbereich gelegt [Pete81, Reis90]. Eine Zustandsänderung für ein Stellen/Transitions-System kann erfolgen, wenn eine Transition aktiviert ist. Das bedeutet zum einen, dass die Anzahl der Marken jeder Stelle im Vorbereich der Transition größer ist als das Gewicht der Kante, die von der jeweiligen Stelle zur Transition geht. Zum anderen muss in jeder Stelle im Nachbereich der Transition eine noch ausreichende Kapazität zur Aufnahme der erzeugten Marken vorhanden sein.

Definition Schaltregel für S/T-Systeme:

Sei ein S/T-System $STS = (S, T, F, K, W, M)$ gegeben. Eine Transition $t \in T$ ist aktiviert unter der Markierung M, wenn gilt:

i) $\forall s \in {\bullet}t: M(s) \geq W(s,t)$ und

ii) $\forall s \in t{\bullet}: M(s) \leq K(s) - W(t,s)$

Falls eine Transition t unter der Markierung M aktiviert ist, kann t schalten. Das Schalten einer Transition t unter M überführt M in die Folgemarkierung M', wie folgt:

$$M'(s) = \begin{cases} M(s) - W(s,t) & \text{falls } s \in {\cdot}t \setminus t{\cdot} \\ M(s) + W(t,s) & \text{falls } s \in t{\cdot} \setminus {\cdot}t \\ M(s) - W(s,t) + W(t,s) & \text{falls } s \in t{\cdot} \cap {\cdot}t \\ M(s) & \text{sonst} \end{cases}$$

[Reis90]

Bislang wurden zwei Petri-Netz-Varianten, das Bedingungs/Ereignis-System sowie das Stellen/Transitions-System, vorgestellt. In der Literatur existieren zahlreiche Varianten von Petri-Netzen, wie beispielsweise an der Publikationsliste der Universität Hamburg deutlich wird, welche mehr als 7000 Einträge umfasst [UHH18]. Im Folgenden soll ausschließlich auf einige weitere Varianten hingewiesen werden. Grundsätzlich wird zwischen einfachen und höheren Petri-Netzen differenziert [DeRe98]. Höhere Petri-Netze werden im Standard ISO/IEC 15909-1:2004 [ISO15909-1] beschrieben und ermöglichen weitere Realitätsannäherungen [Neus06]. Die ISO/IEC 15909 ist dreiteilig aufgebaut und beschreibt im ersten Teil die notwendigen Begriffe, Syntax, Semantik und die Notation, welche seit 2004 ein offizieller Standard ist [ISO15909-1]. Der zweite Teil ist seit 2011 offizieller Standard und definiert die Petri Netz Markup Language (PMNL) [HKPT10] für Petri-Netze, welches als Austauschformat verwendet werden kann [ISO15909-2]. Die Konzeption des dritten Teils befindet sich in der Entwicklungsphase und soll ein Framework enthalten, um Petri-Netz-Erweiterungen zu definieren [HKPT10], wie beispielsweise gefärbte Petri-Netze (Coloured Petri Nets CPN [Jens91, Jens92, Jens95, Jens97]) oder zeitbasierte Erweiterungen (Time Petri Nets [VaFC91, CeMa99, Ober90] sowie auch Stochastic Petri Nets [BaKr96]). Ein wesentlicher Unterschied zwischen einfachen und höheren Petri-Netzen ist die Unterscheidbarkeit der Marken [ISO15909-1].

3.6 Yet Another Workflow Language (YAWL) und New Yet Another Workflow Language (newYAWL)

Die Yet Another Workflow Language (YAWL) ist eine Sprache zur Modellierung von Workflows. Ein Workflow ist die teilweise oder vollständige Automatisierung eines Geschäftsprozesses, welcher durch ein Workflow-Management-System ausgeführt wird [AaHo05]. Die Anforderungen an eine Workflow-Modellierungssprache können aus den Mustern der Workflow Pattern Initiative abgeleitet werden [Russ07]. Die Kontrollflussmuster der Workflow Pattern Initiative werden detailliert in Kapitel 4 betrachtet. YAWL basiert auf den 20 Kontrollflussmustern der Workflow Pattern Initiative [AHKB03]. 23 zusätzliche Kontrollflussmuster wurden definiert, um die Fähigkeiten eines Workflow-Management-Systems besser bewerten zu können [RHAM06]. Es wurden insbesondere Kontrollflussmuster für erweiterte Schleifen sowie die Thread-Betrachtung integriert. YAWL unterstützt 32 Kontrollflussmuster [Dres16b, HZLH12]. Um auch die anderen Kontrollflussmuster in YAWL zu unterstützen, wurden *Thread Aufgaben* (vgl. Abbildung 25o und Abbildung 25p), *Repetitive Aufgaben* (vgl. Abbildung 25q), *Teilzusammenführungsaufgaben* (vgl. Abbildung 25r), die *Blocking-Region* (vgl. Abbildung 25s), *Persistent* und *Transient Trigger Aufgaben* (vgl. Abbildung 25t und Abbildung 25u), die *Beendigungsregion* (vgl. Abbildung 25v) und die *Deaktivierungskante* (vgl. Abbildung 25w) integriert. Die Erweiterung wird als newYAWL bezeichnet [RuHA07, RuHE07] und unterstützt 42 der 43 Kontrollflussmuster (vgl. Kapitel 4.1.2). Im Kern basiert YAWL auf Petri-Netzen, sodass eine YAWL-Spezifikation auch als erweitertes Workflow-Netz interpretiert werden kann, mit zusätzlichen Notationselementen, die eine Kurzschreibweise ermöglichen. Die Petri-Netze wurden aufgrund der formalen Kontrollflusssemantik, des zustandsbasierten Charakters sowie der Vielzahl an möglichen Analyseverfahren ausgewählt. Insbesondere fehlte eine Kurzschreibweise nach [AaHo05] zur Modellierung von Mehrfachinstanzen, sodass ein Modellierer die Mehrfachinstanzen explizit durch eine Aufspaltung und Zusammenführung des Kontrollflusses modellieren muss. Darüber hinaus werden manchmal Formulierungen benötigt, bei denen eine Synchronisation oder Zusammenführung erforderlich ist. Im Rahmen der Petri-Netze muss sich der Modellierer a priori für eine Synchronisation oder die Zusammenführung entscheiden. Des Weiteren wird der Zustandsübergang durch Transitionen nur lokal betrachtet, sodass die Fehlerbehandlung oder die Kompensation von Prozessabschnitten erschwert wird bzw. die explizite Modellierung dieser Aspekte die Lesbarkeit des Workflows negativ beeinflusst [AaHo05]. Nachfolgend werden die Notationselemente von YAWL und newYAWL sowie deren inhaltliche Bedeutung vorgestellt (vgl. Abbildung 25).

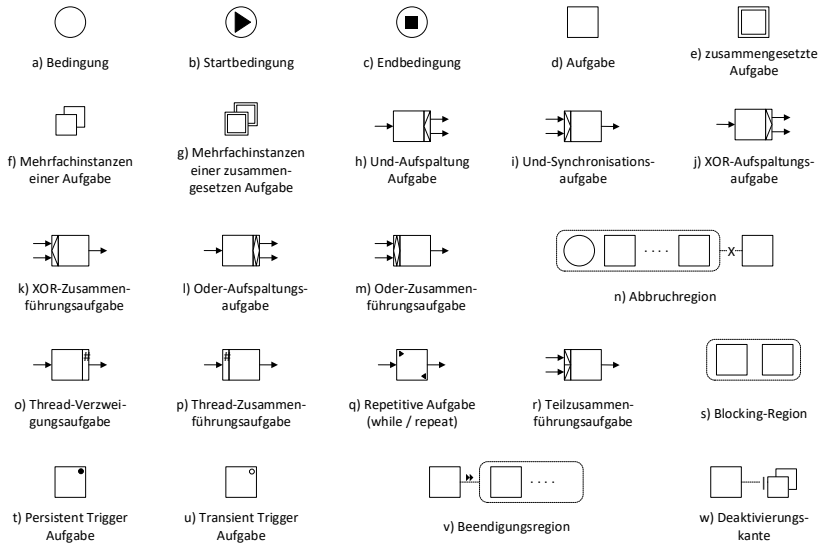


Abbildung 25: Notationselemente - YAWL und newYAWL [Russ07]

Ein erweitertes Workflow-Netz besteht aus einer Menge von Aufgaben (engl. tasks) und Bedingungen (engl. conditions), vergleichbar mit den Stellen und Transitionen eines Petri-Netzes. Die *atomaren Aufgaben* (engl. atomic tasks, vgl. Abbildung 25d) haben inhaltlich die gleiche Bedeutung, wie die Transitionen bei Petri-Netzen. Die *zusammengesetzte Aufgabe* (engl. composite tasks, Abbildung 25e) verweist auf ein anderes erweitertes Workflow-Netz. Eine Aufgabe (engl. multiple instances of an atomic task, vgl. Abbildung 25f) sowie auch eine zusammengesetzte Aufgabe kann als *Mehrfachinstanzaufgabe* spezifiziert werden (engl. multiple instances of a composite task, vgl. Abbildung 25g). In diesem Zusammenhang werden für die Ausführung der Aufgabe die einzelnen Instanzen, basierend auf einer festgelegten minimalen und maximalen Anzahl von Instanzen, erzeugt. Die Aufgabe gilt als beendet, sobald ein definierter Schwellwert von beendeten Instanzen erreicht ist. Bei newYAWL muss zusätzlich das Attribut cancelling oder nicht-cancelling spezifiziert werden, mit der Folge, dass bei der Erreichung eines Schwellwertes entweder alle Instanzen abgebrochen oder nicht-abgebrochen werden. Bei nicht-cancelling werden alle noch nicht beendeten Instanzen weiterhin ausgeführt. Die Aufgabe gilt erst als ausgeführt, wenn alle Instanzen beendet sind. Bei cancelling werden alle noch nicht erfolgreich ausgeführten Instanzen abgebrochen. Darüber hinaus besteht die Möglichkeit, dass die Anzahl der Instanzen zu Beginn genau festgelegt wird (statisch), sodass keine weiteren Instanzen zur Laufzeit hinzugefügt werden dürfen. Sofern der Wert auf dynamisch gesetzt wird, können weitere Instanzen zur Laufzeit erzeugt werden [Russ07].

Die Bedingungen werden, wie bei Petri-Netzen, durch Kreise dargestellt. Jedes erweiterte Workflow-Netz wird durch eine eindeutige *Start-* (engl. input condition, vgl. Abbildung 25b) und *Endbedingung* (engl. output condition, vgl. Abbildung 25c) gekennzeichnet. Darüber hinaus können *Bedingungen* (engl. condition, vgl. Abbildung 25a) innerhalb des Workflows modelliert werden. Ebenfalls wie bei Petri-Netzen sind Bedingungen (Stellen) mit Aufgaben (Transitionen) alternierend verknüpft. Bei erweiterten Workflow-Netzen besteht außerdem die Möglichkeit, die Aufgaben direkt miteinander zu verbinden, jedoch wird implizit angenommen, dass eine unsichtbare Bedingung zwischen den Aufgaben existiert [Russ07].

Entscheidungssituationen oder Nebenläufigkeiten werden durch spezielle Aufspaltungs- und Zusammenführungsaufgaben modelliert. Es wird in erweiterten Workflow-Netzen zwischen *Und-* (AND), *Entweder-Oder-* (XOR) und *Oder-Aufspaltungsaufgaben* (OR) bzw. *-Zusammenführungsaufgaben* unterschieden (vgl. Abbildung 25i bis Abbildung 25m). Durch newYAWL wurden *Thread-Verzweigungs-* (engl. thread split task, vgl. Abbildung 25o) und *-Zusammenführungsaufgaben* (engl. thread merge task, vgl. Abbildung 25p) sowie eine *Teilzusammenführungsaufgabe* (engl. partial-join task, vgl. Abbildung 25r) hinzugefügt. Ein Thread ist eine Ausführungsreihenfolge für die Abarbeitung eines Programms. Übertragen auf Geschäftsprozessmodelle ist ein Thread ein Bereich eines Geschäftsprozessmodells, der mit einer Thread-Verzweigungs- und -Zusammenführungsaufgabe mehrfach ausgeführt werden kann, wodurch ein weiterer Mechanismus für die Mehrfachinstanzaufgabe bereitgestellt wird. Es wird eine Anzahl von Threads entlang des ausgehenden Zweiges initiiert, die nur wieder durch eine Thread-Zusammenführungsaufgabe verschmolzen werden können. Die Thread-Verzweigungs- bzw. -Zusammenführungsaufgaben haben jeweils nur eine ein- bzw. ausgehende Kante, d. h. alle Threads werden über diese eine Kante abgewickelt. Die *Teilzusammenführungsaufgabe* bzw. n-out-of-m-Join-Aufgabe ist eine spezielle Variante einer Und-Zusammenführung und kann ausgeführt werden, wenn von den m Aufgaben im Vorbereich n Aufgaben ausgeführt wurden. Das Konstrukt wird zurückgesetzt, wenn alle m Aufgaben im Vorbereich ausgeführt wurden. Daher bietet sich die Kombination mit einer *Blocking Region* (vgl. Abbildung 25s) an, sodass diese Aufgaben bis zur Zurücksetzung nicht wieder ausgeführt werden können.

Die *Aufgabenwiederholung* (engl. repetitive task (while/repeat), vgl. Abbildung 25q) ist ebenfalls eine Erweiterung, bei der mithilfe von Vor- oder Nach-Tests überprüft wird, wie lange die Aufgabe wiederholt werden soll. *Persistent-* (engl. persistent trigger task, vgl. Abbildung 25t) und *transient Trigger-Aufgaben* (engl. transient trigger task, vgl. Abbildung 25u) sind Trigger, die einen externen Auslöser besitzen. In diesem Zusammenhang kann der Trigger dauerhaft (persistent) oder vorübergehend (transient) aktiviert sein. Die *Beendigungsregion* (engl. completion region, vgl. Abbildung 25v) bietet die Möglichkeit, dass Aufgaben zu einer Beendigung gezwungen werden. Im Gegensatz dazu werden bei einer

Abbruchregion (engl. cancellation region, vgl. Abbildung 25u) die Aufgaben abgebrochen. Die *Deaktivierungskante* (engl. disablement arc, vgl. Abbildung 25w) verhindert die weitere Instanziierung von Instanzen einer Mehrfachinstanzaufgabe, wenn die assoziierte Aufgabe ausgeführt wurde [AaHo05, RuHA07]. Abschließend werden die Definitionen eines YAWL-Netzes und newYAWL-Netzes aufgezeigt [Russ07]. Die Split-, Join-, Rem- und Nofi-Elemente aus dem erweiterten Workflow-Netz sind partielle Funktionen, da nicht jede Aufgabe einen Funktionswert besitzt. Eine partielle Funktion ($\rightharpoonup$) ist eine rechtseindeutige, aber keine linkstotale Relation, d. h. eine Relation, bei der kein Element des Definitionsbereichs auf mehr als ein Element des Wertebereichs abgebildet wird [Hoff18].

Definition YAWL-Netz:

Ein erweitertes Workflow-Netz (YAWL-Netz) ist ein Tupel (C, i, o, T, F, Split, Join, Rem, Nofi), mit den Eigenschaften:

- C ist eine Menge von Bedingungen.
- $i \in C$ ist die Input-Bedingung (Startbedingung).
- $o \in C$ ist die Output-Bedingung (Endbedingung).
- T ist die Menge der Aufgaben.
- $F \subseteq (C \backslash \{o\} \times T) \cup (T \times C \backslash \{i\}) \cup (T \times T)$ ist die Flussrelation.
- Jeder Knoten im Netz $(C \cup T, F)$ liegt auf einem direkten Pfad von i nach o.
- $\text{Split}: T \rightharpoonup \{\text{AND}, \text{XOR}, \text{OR}\}$ spezifiziert das Split-Verhalten einer Aufgabe.
- $\text{Join}: T \rightharpoonup \{\text{AND}, \text{XOR}, \text{OR}\}$ spezifiziert das Join-Verhalten einer Aufgabe.
- $\text{Rem}: T \rightharpoonup \mathbb{P}^+(T \cup C \backslash \{i, o\})$ spezifiziert Aufgaben und Bedingungen, bei denen Token entfernt werden, aufgrund der Ausführung einer anderen Aufgabe.
- $\text{Nofi}: T \rightharpoonup \mathbb{N} \times \mathbb{N}^{\inf} \times \mathbb{N}^{\inf} \times \{\text{dynamic}, \text{static}\}$ spezifiziert die Multiplizität von Mehrfachinstanzaufgaben (Minimum, Maximum, Schwelle für die Fortsetzung und dynamische oder statische Generierung der Instanzen).

[AaHo05]

Definition newYAWL-Netz:

Ein erweitertes Workflow-Netz (newYAWL-Netz) ist ein Tupel (nid, C, i, o, T, T_A, T_C, M, F, Split, Join, Default, $<_{XOR}$, Rem, Comp, Block, Nofi, Disable, Thresh, ThreadIn, ThreadOut, ArcCond, Pre, Post, PreTest, PostTest, WPre, WPost, Trig, Persist), mit den Eigenschaften:

- C ist eine Menge von Bedingungen.
- $i \in C$ ist die Input-Bedingung (Startbedingung).
- $o \in C$ ist die Output-Bedingung (Endbedingung).
- T ist die Menge der Aufgaben.
- $T_A \subseteq T$ ist die Menge von atomaren Aufgaben.
- $T_C \subseteq T$ ist die Menge von zusammengesetzten Aufgaben.
- T_A und T_C bilden die Partition, d. h. nichtleere paarweise disjunkte Teilmenge, über T.
- $M \subseteq T$ ist die Menge der Mehrfachinstanzaufgaben.
- $F \subseteq (C \backslash \{o\} \times T) \cup (T \times C \backslash \{i\}) \cup (T \times T)$ ist die Flussrelation und jeder Knoten im Netz $(C \cup T, F)$ liegt auf einem direkten Pfad von i nach o.
- $\mathrm{Split}: T \rightharpoonup \{\mathrm{AND}, \mathrm{XOR}, \mathrm{OR}, \mathrm{THREAD}\}$ spezifiziert das Verzweigungsverhalten einer Aufgabe, wobei $\forall_{t \in \mathrm{dom(Split)}}[\mathrm{Split}(t) = \mathrm{THREAD} \Rightarrow |t \cdot| = 1]$ gilt, d. h. Thread-Verzweigungen haben ausschließlich eine ausgehende Kante.
- $\mathrm{Join}: T \rightharpoonup \{\mathrm{AND}, \mathrm{XOR}, \mathrm{OR}, \mathrm{PJOIN}, \mathrm{THREAD}\}$ spezifiziert das Zusammenführungsverhalten einer Aufgabe, wobei $\forall_{t \in \mathrm{dom(Join)}}[\mathrm{Join}(t) = \mathrm{THREAD} \Rightarrow |\cdot t| = 1]$ gilt, d. h. Thread-Zusammenführungen haben ausschließlich eine eingehende Kante.
- $\mathrm{Default} \subseteq F$, $\mathrm{Default}: \mathrm{dom}(\mathrm{Split} \rhd \{\mathrm{OR}, \mathrm{XOR}\}) \rightarrow T \cup C$ definiert die Default-Kante für einen OR-Split oder XOR-Split.
- $<_{\mathrm{XOR}} \subseteq \{t \in T | \mathrm{Split}(T) = \mathrm{XOR}\} \times \mathbb{P}((T \cup C) \times (T \cup C))$ beschreibt die Auswertungssequenz von den ausgehenden Kanten einer XOR-Split-Aufgabe, sodass für jede $(t, V) \in <_{\mathrm{XOR}}$ geschrieben wird $<^t_{\mathrm{XOR}} = V$ und V streng geordnet ist über $t \cdot := \{x \in T \cup C | (t, x) \in F\}$. Die Bedingungen werden zu jeder Kante evaluiert und in der Reihenfolge der Sequenz abgeprüft, bis die erste Bedingung zutrifft. Andernfalls wird die Default-Kante ausgelöst.
- $\mathrm{Rem}: T \rightharpoonup \mathbb{P}^+(T \cup C \backslash \{i, o\})$ spezifiziert Aufgaben und deren Token werden entfernt, aufgrund der Ausführung einer anderen Aufgabe.
- $\mathrm{Comp}: T \rightharpoonup \mathbb{P}^+(T)$ spezifiziert Aufgaben, die beendet werden, aufgrund der Beendigung einer anderen Aufgabe.
- $\mathrm{Block}: T \rightharpoonup \mathbb{P}^+(T)$ mit $t \in \mathrm{dom(Block)} \Leftrightarrow \mathrm{Join}(t) = \mathrm{PJOIN}$ spezifiziert Aufgaben, die nach dem Schalten einer Teilzusammenführungsaufgabe geblockt werden, bevor diese zurückgesetzt werden $\forall_{t \in \mathrm{dom(Block)}} t \notin \mathrm{Block}(t)$.

- $\text{Nofi}: M \to \mathbb{N} \times \mathbb{N}^{\inf} \times \mathbb{N}^{\inf} \times \{\text{dynamic}, \text{static}\} \times \{\text{cancelling}, \text{non} - \text{cancelling}\}$ spezifiziert die Multiplizität jeder Mehrfachinstanzaufgabe, wie die untere und obere Schranke von Instanzen für die Initiierung. Ebenfalls wird die Schwelle für die Fortführung bestimmt und festgelegt, ob zusätzliche Instanzen «on the fly» erzeugt werden können, während die Aufgabe ausgeführt wird (dynamisch oder statisch). Zusätzlich kann definiert werden, ob das partielle Synchronisationsergebnis die anderen Instanzen beendet oder ob diese weiter ausgeführt werden sollen.
- $\text{Disable}: T \rightharpoonup \mathbb{P}^{+}(M)$ spezifiziert die Mehrfachinstanzaufgaben, die für die Erzeugung weiterer Instanzen deaktiviert werden, aufgrund einer beendeten Aufgabe.
- $\text{Thresh}: T \rightharpoonup \mathbb{N}$ mit $t \in \text{dom}(\text{Thresh}) \Leftrightarrow \text{Join} = \text{PJOIN}$ und $\forall_{t \in \text{dom}(\text{Thresh})}[1 \leq \text{Thresh}(t) < |\cdot t|]$ spezifiziert die erforderlichen Token für die Fortführung des Kontrollflusses (z. B. n-out-of-m Join).
- $\text{ThreadIn}: T \rightharpoonup \text{NatExpr}$, mit $\text{dom}(\text{ThreadIn}) = \text{dom}(\text{Join} \rhd \{\text{THREAD}\})$ die Anzahl eingehender Threads.
- $\text{ThreadOut}: T \rightharpoonup \text{NatExpr}$, mit $\text{dom}(\text{ThreadOut}) = \text{dom}(\text{Split} \rhd \{\text{THREAD}\})$ die Anzahl ausgehender Threads.
- $\text{ArcCond}: F \cap (\text{Split} \rhd \{\text{XOR}, \text{OR}\}) \times (T \cup C)) \to \text{BoolExpr}$ kennzeichnet spezifische Bedingungen für jede ausgehende Kante aus einer OR- und XOR-Verzweigung.
- $\text{Pre}: T \rightharpoonup \text{BoolExpr}$ ist eine Funktion zur Beschreibung von Vorbedingungen für Aufgaben.
- $\text{Post}: T \rightharpoonup \text{BoolExpr}$ ist eine Funktion zur Beschreibung von Nachbedingungen für Aufgaben.
- $\text{PreTest}: T \rightharpoonup \text{BoolExpr}$ ist eine Funktion zur Beschreibung von Schleifenvorbedingungen für Aufgaben. Wenn die Bedingung erfüllt ist, wird die Aufgabe ausgeführt, andernfalls wird diese übersprungen.
- $\text{PostTest}: T \rightharpoonup \text{BoolExpr}$ ist eine Funktion zur Beschreibung von Schleifennachbedingungen einer Aufgabe. Wenn die Bedingung nicht erfüllt ist, wird die Aufgabe wiederholt, andernfalls wird die Ausführung beendet.
- $\text{WPre} \in \text{BoolExpr}$ definiert die Voraussetzung für den Beginn einer Prozessinstanz.
- $\text{WPost} \in \text{BoolExpr}$ definiert die Nachbedingung für die Beendigung einer Prozessinstanz.
- $\text{Trig}: T \rightharpoonup \text{TriggerID}$ kennzeichnet Aufgaben, die mit einem Trigger assoziiert sind.
- $\text{Persist} \subseteq \text{dom}(\text{Trig})$ beschreibt die Teilmenge von Triggern, die persistent sind.

[RuHA07]

3.7 Weitere Modellierungssprachen für Geschäftsprozessmodelle

Es existieren nicht nur die vorgestellten Modellierungssprachen BPMN, EPK, Petri-Netze, UML-Aktivitätsdiagramm und YAWL bzw. newYAWL, sondern auch zahlreiche weitere Modellierungssprachen, auf die in diesem Abschnitt lediglich verwiesen wird. Gegliedert wird die Betrachtung anhand des 4-Schichtenmodells von [Ober96] (vgl. Abbildung 7).

Die anwendungsorientierten bzw. domänenspezifischen Modellierungssprachen sind von einer oder mehreren Domänen abhängig. Beispielsweise wurde die PICTURE-Methode [BePR07a, BePR07b] speziell für die öffentliche Verwaltung entwickelt. Für das Gesundheitsweisen wurde z. B. der klinische Algorithmus [KoLS04], Standards-Based Active Guideline Environment (SAGE) [TCGN+07], Guideline Interchange Format (GLIF) [BPTO+04], PROforma [SuFo03] sowie die Human Cognitive Model Language (HCM-L) [MiMa13] entwickelt. HCM-L besteht beispielsweise aus 12 Modellierungselementen und wird für die Dokumentation des Ambient Assisted Living (AAL) eingesetzt. Ein Beispielszenario einer Abendaktivität einer Person wird in Abbildung 26 aufgezeigt [KaMM16].

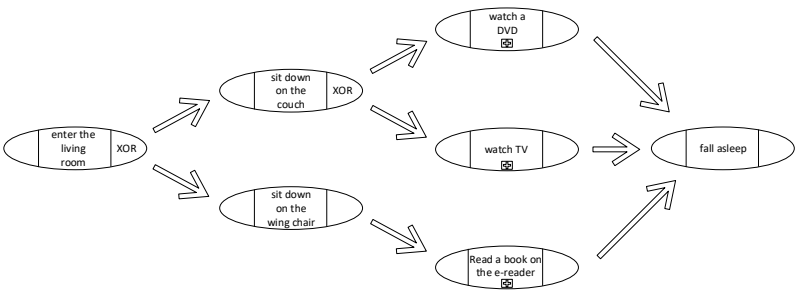


Abbildung 26: Beispiel - HCM-L - Verhaltensmodellierung einer Abendaktivität [KaMM16]

Die PICTURE-Methode ist ebenfalls eine domänenspezifische Modellierungssprache, die für die Domäne der öffentlichen Verwaltung konzipiert wurde. Die PICTURE-Methode besteht aus 24 Notationselementen, die in fünf Handlungsfelder gruppiert werden, d. h. in Verschriftlichung & Dokumentation, inhaltliche Verwaltungstätigkeit, Medienwechsel, Informationsflüsse sowie Informationsbeschaffung & Koordination [BePR07a, BePR07b]. Ein Prozess in der öffentlichen Verwaltung ist zum Beispiel die steuerliche Anmeldung eines Hundes. Zudem kann die PICTURE-BPMN angewendet werden, welche die Business Process Model and Notation um die domänenspezifischen Notationselemente der öffentlichen Verwaltung erweitert. Dementsprechend wurden die Notationselemente der PICTURE-

Methode in BPMN integriert. An diesem Beispiel lässt sich erkennen, dass anwendungsneutrale Modellierungssprachen auch um domänenspezifische Modellierungselemente erweitert werden können. Speziell BPMN bietet im Rahmen der Spezifikation explizit die Möglichkeit zur Erweiterung [BPMN2.0.2, StCV11] an. Unter anderem wird in [ScRa14] die Modellierungssprache BPMN für die Anwendungsdomäne Prozess-Monitoring modifiziert oder auch in [StCV11] um die Ressourcenperspektive erweitert, um damit eine bessere Aufgabenverteilung zu ermöglichen. Eine weitere Notationserweiterung der BPMN-Spezifikation wird beispielsweise in [BSBE15] beschrieben. Danach werden die Aspekte des Gesundheitswesens (BPMN for Clinical Pathways, kurz: BPMN4CP) integriert und unter anderem der Aufgaben-Typ Diagnose hinzugefügt. Weitere Aspekte, um die die BPMN erweitert wurde, sind z. B. die Risiko- [MaKu12], Sicherheits- [TBCB12], Verwaltungs- [AlSc17] oder auch interne Kontrollaspekte (BPMN+C) [ScRa14]. Nicht nur BPMN, sondern auch andere Modellierungssprachen, wie z. B. die Ereignisgesteuerte Prozesskette, können um domänenspezifische Modellierungselemente erweitert werden. In [RaSN15] wird die Ereignisgesteuerte Prozesskette beispielsweise um interne Kontrollaspekte (EPC for internal controls, kurz EPC+C) erweitert, wie Risiko (engl. risk) oder auch Zielsetzung (engl. control objective).

Im Rahmen der semi-formalen, anwendungsneutralen Modellierungssprachen wurden bereits die Modellierungssprachen BPMN, EPK sowie UML-Aktivitätsdiagramme vorgestellt, die graphischer Natur sind. Für diese Sprachen existieren ebenfalls textuelle Repräsentationen, die in der Regel XML-basiert sind. Somit ermöglichen sie eine maschinelle Verarbeitung und den Austausch zwischen Softwaresystemen [Wesk12]. Für die Ereignisgesteuerte Prozesskette wird die EPC Markup Language (EPML) vorgeschlagen [MeNü05]. BPMN bietet ebenfalls ein XML-basiertes Format an, mit dessen Hilfe BPMN-Diagramme gespeichert und ausgetauscht werden können [BPMN2.0.2]. Darüber hinaus ist der Austausch von BPMN Modellen auch mit dem OMG-Standard XML Metadata Interchange (XMI) möglich [UML2.5.1]. Für UML-Aktivitätsdiagramme wird auch der XMI-Standard empfohlen. Es existieren auch rein textuelle Modellierungssprachen zur Modellierung von Geschäftsprozessen [BePZ11, FaBF12]. Diese Sprachen werden in der Regel als Ausführungssprachen verwendet [Wesk12], wie z. B. die Web Services-Business Process Execution Language (WS-BPEL) [WS-BPEL2.0], die electronic business using XML (ebXML) [ebXML2.0.4] oder auch die XML Process Definition Language (XPDL) [Work12]. Die rein textuellen Modellierungssprachen werden im Rahmen dieser Arbeit nicht weiter betrachtet.

Weitere formale plattformunabhängige Modellierungssprachen und maschinenorientierten Programmiersprachen werden nicht betrachtet. Für diese Modellierungssprachen liegt bereits eine präzise Kontrollflusssemantik zugrunde. Aufgrund der formalen Eigenschaft der Kontrollflusssemantik kann bei Bedarf eine Übersetzungssemantik (siehe Kapitel 4.2) entwickelt werden. Zu den bekanntesten Modellierungssprachen der formalen

plattformunabhängigen Modellierungssprachen [Ober96] (vgl. Abbildung 7) zählen im Kontext der Geschäftsprozessmodellierung die Petri-Netze sowie die erweiterten Workflow-Netze (YAWL bzw. newYAWL), welche bereits vorgestellt wurden. Abschließend wird eine Liste von weiteren Prozessmodellierungssprachen aufgezeigt, welche auf [Lich12] basiert und erweitert wurde.

Quelle	Name
[DaEA98]	Abstract Process Engine Language (APEL)
[KaJS96]	Adonis Prozessmodellierungssprache
[Chap70]	ANSI Flow Charts
[Lank13]	ArchiMate
[ChMM10]	Aspect-Oriented Business Process Modeling (AO4BPMN)
[Öste95]	Aufgabenkette
[FeSi95]	Aufgabensystem
[WS-BPEL2.0]	Business Process Execution Language for Web Services (WS-BPEL)
[BPMN2.0.2]	Business Process Model and Notation (BPMN)
[BSBE15]	BPMN for Clinical Pathways (BPMN4CP)
[ScRa14]	BPMN for Internal Controls (BPMN+C)
[StCV11]	BPMN for Ressource
[BPML0.4]	Business Process Modeling Language (BPML)
[MaKu12]	Business Process Modeling Notation Extension for Risk Handling
[CMMN1.1]	Case Management Model and Notation (CMMN)
[Jens91]	Coloured Petri Nets
[KoVZ97]	Computer Integrated Manufacturing Open System Architecture (CIMOSA)
[PeAa06]	ConDec
[GaSa79]	Datenflussdiagramm (SSA)
[DMN1.1]	Decision Model and Notation (DMN)
[BrØs06]	Domain Specfic Modeling Languages
[JaPL98]	E^3
[ebXML2.0.4]	ebXML Business Process Specification Schema
[ScRa14]	EPC for internal controls(EPC+C)
[MeNü05]	EPC Markup Language (EPML)
[KeNS92]	Ereignisgesteuerte Prozesskette (EPK)
[Rose06]	Ereignisgesteuerter Prozessgraph
[RoTu08]	Executable Process Modeling Language (EPML)
[EmGr91]	Funsoft Net
[Schm96]	Generalised Process Networks (GPN)
[Prit66]	Graphical Evaluation and Review Technique (GERT)
[BPTO+04]	Guideline Interchange Format (GLIF)
[MiMa13]	Human Cognitive Modeling Language (HCM-L)
[IDEF0]	Integraion Definition for Function Modeling (IDEF)
[IDEF3]	IDEF 3 Process Description Capture Method
[SpMJ96]	Integrated Enterprise Modelling
[FeSi95]	Interaktionsmodell (IM)
[KoLS04]	klinischer Algorithmus

Quelle	Name
[Tard92]	Merise
[BaBl03]	Metagraphs
[Fran94]	Multi-Perspective Enterprise Modelling
[RuHE07]	New Yet Another Workflow Language (newYAWL)
[Mevi08]	Performance Net
[Petr62]	Petri-Netz
[BePR07b]	PICTURE
[BaWe90]	Process Algebra
[GrWe01]	Process Landscaping
[HBHK+08]	Process Map
[AnRa08]	Process Modeling Language (PML)
[RaSN15]	Process Monitoring
[SuFo03]	PROforma
[HoRG83]	Role Activity Diagram (RAD)
[BFGL94]	Spade Language (SLANG)
[CJMN+92]	SPELL
[TCGN+07]	Standards-based Active Guideline Environment (SAGE)
[Booc91]	Unified Modeling Language - Aktivitätsdiagramm
[Sche01]	Vorgangskettendiagramm
[CeFB00]	Web modeling language (WebML)
[Sche01]	Wertschöpfungskette
[Work12]	XML Process Definition Language (XPDL)
[AaHo05]	Yet Another Workflow Language (YAWL)

Tabelle 3: Beispiele - Geschäftsprozessmodellierungssprachen

3.8 Zusammenfassung

In den vorherigen Abschnitten wurde der Begriff Modellierungssprache definiert. Zudem wurden die Modellierungssprachen Business Process Model and Notation (BPMN), Ereignisgesteuerte Prozesskette (EPK), UML-Aktivitätsdiagramm (UML-AD), Petri-Netze (PN) und die erweiterten Workflow-Netze (YAWL bzw. newYAWL) eingeführt sowie auf weitere Modellierungssprachen hingewiesen. Die vorgestellten Modellierungssprachen werden zum einen in Kapitel 5.3 wieder aufgegriffen, um die Anwendung der entwickelten Methode zur Beschreibung der Kontrollflusssemantik von Geschäftsprozessmodellierungssprachen mit Kontrollflussmustern demonstrieren zu können. Zum anderen werden im anschließenden Kapitel 4 die Kontrollflussmuster zur Beschreibung der Kontrollflusssemantik vorgestellt, deren Kontrollflusssemantik jeweils mit einer YAWL- bzw. newYAWL-Spezifikation präzise beschrieben wird. Die daraus resultierenden Parameter der YAWL- bzw. newYAWL-Spezifikation für ein Kontrollflussmuster werden zur präzisen Beschreibung der Ablaufreihenfolge der Elemente eines Geschäftsprozessmodell (vgl. Kapitel 6) benötigt.

Die anderen eben genannten Modellierungssprachen wurden ausschließlich ausgewählt, um die Anwendung der entwickelten Methoden aus Kapitel 5 und 6 demonstrieren zu können. Die Kontrollflusssemantik der Modellierungssprachen wird zunächst mit den Kontrollflussmustern aus Kapitel 4 festgelegt. Darauf aufbauend kann die Ablaufreihenfolge der Elemente im Geschäftsprozessmodell präzise beschrieben werden. Die Existenz zahlreicher Geschäftsprozessmodellierungssprachen wird unter anderem aus Kapitel 3.7 bzw. der nicht abschließenden Tabelle 3 deutlich. Die ausgewählten Geschäftsprozessmodellierungssprachen werden durch eine Vielzahl unterschiedlicher Eigenschaften charakterisiert. Beispielsweise bestehen die Modellierungssprachen EPK (vgl. Abbildung 16) und Petri-Netze (vgl. Abbildung 24) aus wenigen Syntaxelementen, deren Kontrollflusssemantik mit einfachen Grundstrukturen (sequentielle, parallele und alternative Ablaufreihenfolge) beschrieben werden kann. Die Modellierungssprache BPMN besitzt dahingegen zahlreiche Syntaxelemente, die nicht immer Notationselemente besitzen. Beispielsweise wird der Kontrollfluss zwischen den BPMN-Linkereignissen nicht über Notationselemente dargestellt (vgl. Abbildung 44). Darüber hinaus werden für die Modellierung des Kontrollflusses von Aktivitäten und deren angehefteten Ereignissen (vgl. z. B. Abbildung 33) keine Notationselemente verwendet, wie auch bei der Einbettung der Ereignis-Teilprozesse (vgl. z. B. Abbildung 37). Zusätzlich gibt es Bereiche, wie den Ad-Hoc-Teilprozess (vgl. Abbildung 9g, h), bei dem die Reihenfolge und die Ausführungshäufigkeit der einzelnen Aktivitäten nicht vorgeschrieben wird. Dementsprechend gibt es in BPMN Syntaxelemente, die eine komplexe Ablaufreihenfolge beschreiben, deren Kontrollflusssemantik ebenfalls mit einfachen Grundstrukturen beschrieben werden kann. Hierfür müssen jedoch diese Grundstrukturen entsprechend zusammengesetzt werden. Das UML-Aktivitätsdiagramm kann zwischen den Extremfällen eingeordnet werden, wie auch die erweiterten Workflow-Netze (YAWL bzw. newYAWL). Die ausgewählten Modellierungssprachen sind auch aufgrund des Verbreitungsgrades geeignete Repräsentanten [KoLL09, Aals15] um die Anwendung der entwickelten Methoden in Kapitel 5 und 6 demonstrieren zu können.

4 Kontrollflussmuster zur Beschreibung der Kontrollflusssemantik

Um ungewollte Mehrdeutigkeiten bei der Ablaufreihenfolge der Elemente in Geschäftsprozessmodellen zu vermeiden, sollten alle Stakeholder ein einheitliches Verständnis über die Kontrollflusssemantik besitzen. Dies gilt für die Analysten, die erste Entwürfe erstellen, über die technischen Entwickler, die für die Implementierung verantwortlich sind, bis hin zu den Managern, die die Prozesse verwalten und überwachen. Die Kontrollflusssemantik einer Geschäftsprozessmodellierungssprache wird durch ein Semantikschema definiert und mit der Syntax durch die Semantikzuordnung verknüpft (vgl. Abbildung 6).

Das Semantikschema kann durch eine formale oder auch natürlich-sprachliche Beschreibung definiert werden. Die Kontrollflusssemantik der Petri-Netze und YAWL bzw. newYAWL wird durch eine formale Beschreibung definiert. Dahingegen ist die Kontrollflusssemantik der Modellierungssprachen UML-Aktivitätsdiagramm [UML2.5.1], Ereignisgesteuerte Prozesskette [KeNS92] und Business Process Model and Notation [BPMN2.0.2] natürlich-sprachlich beschrieben. Die natürlich-sprachliche Beschreibung kann aufgrund von Homonymen mehrdeutig sein, so dass gleiche Aspekte von unterschiedlichen Anwendern unterschiedlich aufgefasst werden können. Darüber hinaus ist eine natürlich-sprachliche Beschreibung typischerweise lückenhaft bzw. enthält unpräzise oder mehrdeutige Aspekte, wie auch an der Modellierungssprache EPK deutlich wird. Beispielsweise haben [ChSc94] und [LaSW97] die Kontrollflusssemantik einer EPK [KeNS92] formal beschrieben. [Ritt00] hat diese unter anderem mit Hilfe der Prozessmodelle aus Abbildung 27 gegenübergestellt, mit dem Ergebnis, dass die Kontrollflusssemantik der Oder-Zusammenführung einer EPK (vgl. Abbildung 16c) divergierend definiert wurde.

Die mögliche Ablaufreihenfolge der Elemente des Geschäftsprozessmodells aus Abbildung 27a ist beim zugrunde legen der Kontrollflusssemantik von [ChSc94] und [LaSW97] identisch, da die Oder-Zusammenführung genau die Pfade der Oder-Verzweigung wieder zusammenführt. Dahingegen wurde die Kontrollflusssemantik für die Oder-Zusammenführungen von Abbildung 27b und Abbildung 27c divergierend definiert. In [ChSc94] wird vorausgesetzt, dass bei einer Oder-Zusammenführung immer eine Oder-Verzweigung vorausgeht. Aus diesem Grund ist die Kontrollflusssemantik für die Oder-Zusammenführung der Geschäftsprozessmodelle aus Abbildung 27b und Abbildung 27c nicht definiert. In [LaSW97] wird eine Oder-Zusammenführung, ohne eine vorausgehende Oder-

Verzweigung, in mehrere Entweder-Oder-Zusammenführungen aufgespalten. Ähnliche Szenarien können auch mit anderen formalen Beschreibungen für andere Geschäftsprozessmodellierungssprachen aufgezeigt werden. Unter anderem wurden zur Vermeidung von ungewollten Mehrdeutigkeiten in [MeRv10] *sieben Prozessmodellierungsrichtlinien (7PMG)* vorgeschlagen. Regel 5 besagt, dass Oder-Zusammenführungen in den Geschäftsprozessprozessmodellen nicht verwendet werden sollten, sondern nur Entweder-Oder-Verzweigungen und -Zusammenführungen sowie Und-Verzweigungen und -Zusammenführungen. Die ungewollten Mehrdeutigkeiten resultieren aus dem Nicht-Lokalitäts-Problem [Kind06], d. h., es kann nicht lokal am Zusammenführungselement entschieden werden, wann der Kontrollfluss fortgeführt werden soll.

Diese ungewollten Mehrdeutigkeiten der Kontrollflusssemantik einer Oder-Zusammenführung in Geschäftsprozessmodellen konnte auch in einer empirischen Untersuchung festgestellt werden, die im Rahmen dieser Arbeit entwickelt und durchgeführt wurde (vgl. Kapitel 7). An der Umfrage haben 152 Personen teilgenommen, bei der unter anderem die Kontrollflusssemantik der drei Geschäftsprozessmodelle aus Abbildung 27 mit vorgegebenen Antwortmöglichkeiten beschrieben werden sollte. Zur Vermeidung dieser ungewollten Mehrdeutigkeiten kann die Oder-Zusammenführung mit Kontrollflussmustern annotiert werden, um die Kontrollflusssemantik mit vorgegebenen Schablonen zu beschreiben. Die Kontrollflusssemantik sollte für die annotierten Geschäftsprozessmodelle dann erneut von den Teilnehmern mit den vorgegebenen Antwortmöglichkeiten beschrieben werden (vgl. Abbildung 90). Das Ergebnis war, dass durch die Annotation die ungewollten Mehrdeutigkeiten reduziert werden konnten und somit die Verständlichkeit der Kontrollflusssemantik der Geschäftsprozessmodelle verbessert wurde. Die Interpretation der Kontrollflusssemantik der Oder-Zusammenführung nach der Intention des Modellierers konnte für das annotierte Geschäftsprozessmodell in Abbildung 27a um 51,9 Prozentpunkte verbessert werden. Für die Geschäftsprozessmodelle aus Abbildung 27b bzw. Abbildung 27c konnte durch die Annotation der Oder-Zusammenführung des Geschäftsprozessmodells mit Kontrollflussmustern eine Verbesserung der Verständlichkeit um 21,3 Prozentpunkte auf 90% bzw. um 68,8 Prozentpunkte auf 76% erwirkt werden. Mit den Kontrollflussmustern kann aber nicht nur die Kontrollflusssemantik einer Oder-Zusammenführung beschrieben werden, sondern auch die Kontrollflusssemantik einer Geschäftsprozessmodellierungssprache (vgl. Kapitel 5) sowie die Ablaufreihenfolge der Elemente in Geschäftsprozessmodellen (vgl. Kapitel 6).

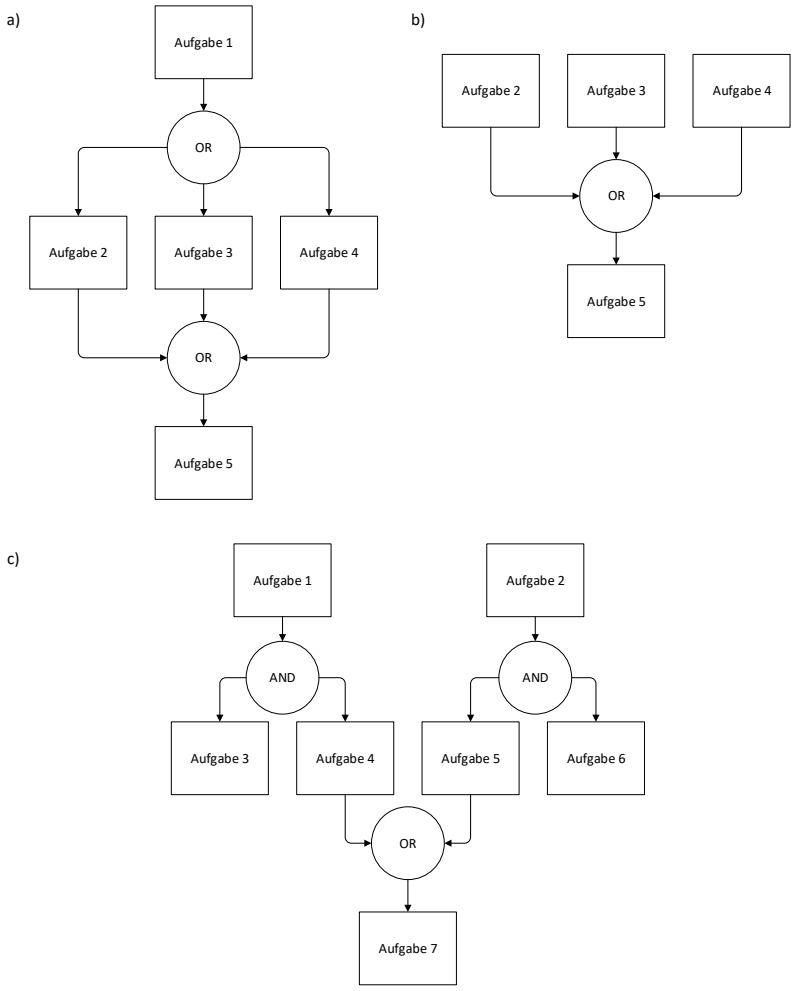


Abbildung 27: Beispiel - Oder-Zusammenführung

Die Kontrollflussmuster sind somit ein zentraler Bestandteil der in Kapitel 5 und 6 entwickelten Methoden. Aus diesem Grund wird in diesem Kapitel zunächst der Begriff Kontrollflussmuster durch die etymologische Herleitung der beiden konstruierenden Wortbestandteile Kontrollfluss und Muster eingeführt. Daran anknüpfend werden die Kontrollflussmuster von Börger und der Workflow Pattern Initiative überblickartig vorgestellt sowie miteinander verglichen. Im Anschluss daran werden die Kontrollflussmuster der Workflow Pattern Initiative detailliert vorgestellt und deren Kontrollflusssemantik mit Hilfe der newYAWL-Sepzifikation beschrieben. Für die Beschreibung der Kontrollflusssemantik von Geschäftsprozessmodellierungssprachen sowie die präzise Beschreibung der

Ablaufreihenfolge der Elemente in den Geschäftsprozessmodellen kann eine Übersetzungssemantik sowie operationelle, denotationelle oder axiomatische Semantik entwickelt werden. Dementsprechend werden jeweils die Eigenschaften einer solchen Semantik beschrieben und auf deren Anwendung in der Literatur verwiesen.

4.1 Kontrollfluss, Muster und Kontrollflussmuster

Für die Beschreibung der Kontrollflusssemantik einer Geschäftsprozessmodellierungssprache (vgl. Kapitel 5) oder die Beschreibung der Ablaufreihenfolge der Elemente eines Geschäftsprozessmodells (vgl. Kapitel 6) können Kontrollflussmuster verwendet werden. Dabei beschreibt der Kontrollfluss die sequenzielle, parallele oder alternative Ablaufreihenfolge der Elemente im Geschäftsprozessmodell. Zu solchen Elementen zählen beispielsweise Aktivitäten, die in elementarer Form eine atomare Einheit der Arbeit beschreiben [AHKB03]. In graphischen Geschäftsprozessmodellen werden die Aktivitäten sowie auch die Elemente zur Ablaufsteuerung (z. B. Verzweigung oder Zusammenführung) durch Vierecke, Kreise oder ähnliche Zeichenelemente dargestellt (vgl. Kapitel 3.1). Die Ablaufreihenfolge zwischen diesen Elementen wird typischerweise bei graphischen Geschäftsprozessmodellierungssprachen mit gerichteten Pfeilen bzw. Kanten modelliert (vgl. z. B. Kapitel 3.2).

Definition Kontrollflussperspektive:

„The control-flow perspective (or process) perspective describes activities and their execution ordering through different constructors, which permit flow of execution control, e. g. sequence, choice, parallelism and join synchronization."

[AHKB03]

Der Grundgedanke von Mustern ist, dass für wiederkehrende Entwurfsprobleme bewährte Lösungen eingesetzt werden können. Dementsprechend kann ein Muster als Schema, Schablone, Bauplan oder auch als Modell interpretiert werden. Der Begriff wurde bereits in den 70er Jahren für die Architektur von Gebäuden verwendet [AISJ77] und hielt Mitte der 90er Jahre in der objektorientierten Softwareentwicklung durch die 23 Entwurfsmuster von [Gamm94] Einzug in das Geschäftsprozessmanagement. Diese Muster wurden auch auf andere Bereiche übertragen, wie an den Analysemustern von [Fowl96] deutlich wird [JuSp06].

Definition Muster:

"Each pattern describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice."

[AISJ77]

Die Kontrollflussmuster können zusammenfassend als Schablonen zur Beschreibung der Ablaufreihenfolge der Elemente im Geschäftsprozessmodell definiert werden.

Definition Kontrollflussmuster:

Kontrollflussmuster sind Schablonen zur Beschreibung der Ablaufreihenfolge der Elemente im Geschäftsprozessmodell.

Ein Geschäftsprozessmodell, dessen Kontrollflusssemantik mit Kontrollflussmustern beschrieben wurde, besteht aus einer Sammlung von Kontrollflussmustern [ThLR07]. In der Literatur existieren zahlreiche Muster im Kontext des Geschäftsprozessmanagements zur Beschreibung der Kontrollflusssemantik, wie in [SVKF+18] aufgezeigt wird. Beispielsweise gibt es die Kontrollflussmuster der Workflow Pattern Initiative (WPI) [AHKB03, RHAM06] und die Kontrollflussmuster von Börger [Börg07]. Im Folgenden werden die Muster von [AHKB03, RHAM06] und [Börg07] im Überblick aufgezeigt und miteinander verglichen. Daran anknüpfend werden die 43 Kontrollflussmuster der Workflow Pattern Initiative ausführlich vorgestellt. Zusätzlich wird die Kontrollflusssemantik jedes Kontrollflussmusters mit einem newYAWL-Beispielsprozessmodell beschrieben. Auf dieser Grundlage werden die relevanten Parameter einer newYAWL-Spezifikation identifiziert, die zur präzisen Beschreibung der Kontrollflusssemantik des Kontrollflussmusters benötigt werden. Für die Beschreibung der Kontrollflusssemantik einer Geschäftsprozessmodellierungssprache werden zunächst die Kontrollflussmuster den Syntaxelementen zugewiesen (vgl. Kapitel 5). Darauf aufbauend können für die präzise Beschreibung der Ablaufreihenfolge der Elemente im Geschäftsprozessmodell jeweils eines der möglichen Kontrollflussmuster für ein Element ausgewählt und die entsprechenden Parameter des Kontrollflussmusters spezifiziert werden (vgl. Kapitel 6).

4.1.1 Vergleich der Kontrollflussmuster der Workflow Pattern Initiative mit den Kontrollflussmustern von Börger

Die Motivation zur Entwicklung von Kontrollflussmustern durch die Workflow Pattern Initiative resultierte aus dem Vergleich bestehender Workflow-Management-Systeme. Für

den ersten Entwurf der Kontrollflussmuster wurden 13 kommerzielle Workflow-Management-Systeme und 2 Prototypen aus der Wissenschaft auf ihre Fähigkeiten hin untersucht. Es konnten 20 Kontrollflussmuster identifiziert werden, die wiederum verwendet wurden, um diese Systeme miteinander zu vergleichen. Einige Muster repräsentieren einfache Strukturen, wie eine Sequenz. Andere sind komplexerer Natur, wie die Modellierung von Meilensteinen oder auch die verschachtelte parallele Ausführung von Bereichen eines Geschäftsprozessmodells. Eine positive Bewertung erhielt ein Workflow-Management-System nach [AHKB03], wenn das Workflow-Management-System das Muster direkt abgebilden kann. Eine negative Bewertung erhielt es, wenn es nicht direkt abgebildet werden konnte. Eine teilweise Abbildung des Musters wurde auf einer 3-Punkteskala als neutral bewertet. Dementsprechend war die Intention zum einen, die verschiedenen Workflow-Management-Systeme miteinander zu vergleichen. Zum anderen sollten die Muster als Grundlage zur Anpassung oder Verfeinerung bestehender Workflow-Management-Systeme verwendet werden können, um unterstützend auf die Entwicklung neuer Ansätze einzuwirken. Die 20 definierten Kontrollflussmuster in [AHKB03] besitzen ausschließlich eine natürlich-sprachliche Beschreibung der Kontrollflusssemantik. In [RHAM06] wurden die 20 Kontrollflussmuster um 23 weitere Muster ergänzt. Die Kontrollflusssemantik der 43 Muster wurde darüber hinaus in [RHAM06] mit gefärbten Petri-Netz-Beispielen [Jens91] präzise beschrieben. Die Muster wurden in [RHAM06] absichtlich nur mit gefärbten Petri-Netz-Beispielen vorgestellt, da diese unabhängig von einer bestimmten Geschäftsprozessmodellierungssprache definiert werden sollten [HAAR10]. Eine Auflistung der 43 Kontrollflussmuster ist aus Tabelle 4 und Tabelle 5 zu entnehmen. Dabei entsprechen die Muster 1 bis 20 den ursprünglichen Mustern [AHKB03] und die Muster 21 bis 43 den ergänzten Kontrollflussmustern [RHAM06].

Tabelle 4: Überblick - Kontrollflussmuster nach [AHKB03, RHAM06, DrKO17] - Teil 1/2

Kategorie	Nummer	Kontrollflussmustername	Erläuterung
Basic Control Flow Patterns *(Elementare Kontrollflussmuster)*	Muster 1 (wcp_1)	Sequence	Beschreiben die elementaren Beziehungen zwischen einzelnen Aktivitäten eines Geschäftsprozesses.
	Muster 2 (wcp_2)	Parallel Split	
	Muster 3 (wcp_3)	Synchronization	
	Muster 4 (wcp_4)	Exclusive Choice	
	Muster 5 (wcp_5)	Simple Merge	
Advanced Branching and Synchronization Patterns *(Erweiterte Verzweigungs- und Synchronisationsmuster)*	Muster 6 (wcp_6)	Multi-Choice	Beschreiben erweiterte Aufspaltungs- und Synchronisationsmuster, die nicht durch die elementaren Kontrollflussmuster abgebildet werden.
	Muster 7 (wcp_7)	Structured Synchronizing Merge	
	Muster 8 (wcp_8)	Multi-Merge	
	Muster 9 (wcp_9)	Structured Discriminator	
	Muster 28 (wcp_{28})	Blocking Discriminator	
	Muster 29 (wcp_{29})	Cancelling Discriminator	
	Muster 30 (wcp_{30})	Structured Partial Join	
	Muster 31 (wcp_{31})	Blocking Partial Join	
	Muster 32 (wcp_{32})	Cancelling Partial Join	
	Muster 33 (wcp_{33})	Generalized AND-Join	
	Muster 37 (wcp_{37})	Local Synchronizing Merge	
	Muster 38 (wcp_{38})	General Synchronizing Merge	
	Muster 41 (wcp_{41})	Thread Merge	
	Muster 42 (wcp_{42})	Thread Split	
Multiple Instance Patterns *(Mehrfachinstanzmuster)*	Muster 12 (wcp_{12})	Multiple Instances without Synchronization	Beschreiben das Verhalten von Aktivitäten die mehrere Instanzen besitzen können. Die Anzahl der Instanzen können zur Laufzeit oder während der Modellerstellung festgelegt werden.
	Muster 13 (wcp_{13})	Multiple Instances with a Priori Design-Time Knowledge	
	Muster 14 (wcp_{14})	Multiple Instances with a Priori Run-Time Knowledge	
	Muster 15 (wcp_{15})	Multiple Instances without a Priori Run-Time Knowledge	
	Muster 34 (wcp_{34})	Static Partial Join for Multiple Instances	
	Muster 35 (wcp_{35})	Cancelling Partial Join for Multiple Instances	
	Muster 36 (wcp_{36})	Dynamic Partial Join for Multiple Instances	

Tabelle 5: Überblick - Kontrollflussmuster nach [AHKB03, RHAM06, DrKO17] - Teil 2/2

Kategorie	Nummer	Kontrollflussmustername	Erläuterung
State-based Patterns *(Zustandsbasierte Muster)*	Muster 16 (wcp_{16}) Muster 17 (wcp_{17}) Muster 18 (wcp_{18}) Muster 39 (wcp_{39}) Muster 40 (wcp_{40})	Deferred Choice Interleaved Parallel Routing Milestone Critical Section Interleaved Routing	Beschreiben das Verhalten von Aktivitäten unter bestimmten Zuständen.
Cancellation and Force Completion Patterns *(Abbruchmuster und Muster mit erzwungenem Ende)*	Muster 19 (wcp_{19}) Muster 20 (wcp_{20}) Muster 25 (wcp_{25}) Muster 26 (wcp_{26}) Muster 27 (wcp_{27})	Cancel Task Cancel Case Cancel Region Cancel Multiple Instance Activity Complete Multiple Instance Activity	Beschreiben mögliche Abbrüche einer oder mehrerer Aktivitäten oder des gesamten Geschäftsprozesses.
Iteration Patterns *(Iterationsmuster)*	Muster 10 (wcp_{10}) Muster 21 (wcp_{21}) Muster 22 (wcp_{22})	Arbitrary Cycles Structured Loop Recursion	Beschreiben die erneute Ausführung von Aktivitäten.
Termination Patterns *(Beendigungsmuster)*	Muster 11 (wcp_{11}) Muster 43 (wcp_{43})	Implicit Termination Explicit Termination	Beschreiben Beendigungsaspekte eines Geschäftsprozesses.
Trigger Patterns *(Triggermuster)*	Muster 23 (wcp_{23}) Muster 24 (wcp_{24})	Transient Trigger Persistent Trigger	Beschreiben Einflüsse, die außerhalb des Geschäftsprozesses auftreten und notwendig sind, um eine bestimmte Aktivität auszuführen.

In [Börg07] wird ein weiterer Ansatz zur Beschreibung von Kontrollflussmustern vorgestellt. Der Grund einer alternativen Definition von Kontrollflussmustern resultiert nach [Börg12] aus einer willkürlichen Auswahl und der Art der Beschreibung der Kontrollflussmuster in [RHAM06]. Börger begründet dies unter anderem mit einer fehlenden statistischen Untersuchung, die beschreibt, welches Muster, wie oft in der Realität angewendet wird. Darüber hinaus beschreibt die Workflow Pattern Initiative nicht, nach welchen Kriterien die konkreten Muster definiert wurden. Deutlich wird dies insbesondere an der kontinuierlich wachsenden Musteranzahl, da in [AHKB03] 20 Kontrollflussmuster definiert und diese in [RHAM06] um 23 weitere Muster ergänzt wurden. Die Ergänzungen resultieren aus der Verfeinerung von bestehenden Kontrollflussmustern und einer kritischen Bewertung der Kontrollflussperspektive [RHAM06]. Darüber hinaus wurden 43 Ressourcenmuster [RAHE05], 40 Datenmuster [RHEA05] sowie 21 Ausnahmebehandlungsmuster [RuAH06] definiert. Die Menge der Kontrollflussmuster der Workflow Pattern Initiative ist nicht minimal und kann prinzipiell durch die Kombinationen von 13 Basismustern abgebildet werden, wie in [Börg07] gezeigt wird. Dabei sind 7 der 13 Muster elementare Muster, wie Sequenz (engl. sequence), Iteration (engl. iteration), Termination (engl. termination), Abbruch (engl. cancellation), Auswahl (engl. choice), Parallelität (engl. parallel split) und Verschachtelung (engl. interleaving). Der zweite Aspekt, der in [Börg12] kritisiert wird, ist die natürlich-sprachliche Beschreibung der Kontrollflusssemantik der Kontrollflussmuster, die nicht präzise ist und teilweise unvollständig beschriebene Verhaltenseigenschaften aufweist. Die ursprünglichen und erweiterten Kontrollflussmuster wurden in [RHAM06] mit gefärbten Petri-Netz-Beispielen definiert. Nach [Börg12] sind die Beispiele jedoch teilweise überspezifiziert, wodurch die Kernbedeutung des Kontrollflussmusters häufig nicht deutlich wird. Die Überspezifizierung ist nach [Börg12] auf die Syntax der gefärbten Petri-Netze zurückzuführen.

In [Börg07] werden daher 13 Basiskontrollflussmuster mit abstrakten Zustandsmaschinen definiert, die jeweils in vier sequentielle und vier parallele Kontrollflussmusterkategorien eingruppiert werden (vgl. Tabelle 6). Die 43 Kontrollflussmuster der Workflow Pattern Initiative können durch eine geeignete Kombination aus den 13 Basismustern entwickelt werden. Mit den abstrakten Zustandsmaschinen kann die Kontrollflusssemantik präzise beschrieben werden [BöTh08a]. Börger hat die abstrakten Zustandsmaschinen zur Beschreibung ausgewählt, da die Muster in einem angemessenen Detaillierungsgrad beschrieben werden können. Das Ziel von Börger bestand darin, eine plausible Klassifikation für die Muster zu entwerfen und die Abhängigkeiten zwischen den Mustern aufzuzeigen. Weiterhin sollte eine durchgängige und präzise Beschreibung der Muster geliefert und ein minimal erzeugendes System definiert werden [Börg07, Börg12].

In [AaHo12] wird die Kritik von [Börg12] aufgegriffen. Beispielsweise kritisiert [Börg12], dass die Kontrollflussmuster nur über eine informale Beschreibung verfügen, gleichzeitig aber die Beschreibung mit den gefärbten Petri-Netz-Beispielen nicht praktikabel sei. [AaHo12] erwidern, dass die Kontrollflusssemantik der Muster absichtlich nicht mit einer spezifischen Sprache beschrieben wurde, da diese unabhängig von einer bestimmten Sprache sein sollte. Darüber hinaus sind gefärbte Petri-Netze nach [AaHo12] keine geeignete Geschäftsprozessmodellierungssprache für Endanwender, sondern eine geeignete Maßnahme zur graphischen und präzisen Beschreibung der Kontrollflusssemantik der Kontrollflussmuster. Petri-Netze und YAWL bzw. newYAWL bieten im Vergleich zu abstrakten Zustandsmaschinen zusätzlich zahlreiche Analyseansätze im Geschäftsprozesskontext an. Für Petri-Netze gibt es Techniken, wie Model Checking, Lineare-Algebraische-Techniken (z. B. Invarianten), Markov Ketten, Simulation etc. Es existieren beispielsweise Analyseverfahren zur Überprüfung der Soundness-Eigenschaft [HAAR10], zur Anwendung von Process Mining Techniken [Aals16] oder auch zur Simulation von Geschäftsprozessmodellen [RWAH+09]. YAWL bzw. newYAWL und Petri-Netze zählen zu den etablierten Geschäftsprozessmodellierungssprachen, wie auch u. a. BPMN, EPK, und UML-AD im Vergleich zu den abstrakten Zustandsmaschinen [AaHo12].

Kategorie	Muster-typ	Muster [Börg07]	Zuordnung zu [RHAM06]	Erläuterung
Sequential Control Flow Pattern *(Sequenzielle Kontrollflussmuster)*	Sequence Patterns	Sequence	Sequence (wcp_1)	Nach der Beendigung einer Aktivität kann eine andere Aktivität ausgeführt werden.
		Milestone	Milestone (wcp_{18})	
	Iteration Patterns	Arbitrary Cycles	Arbitrary Cycles (wcp_{10})	Eine oder mehrere Aktivitäten können wiederholt ausgeführt werden.
		Structured Loop	Structured Loop (wcp_{21})	
		Recursion	Recursion (wcp_{22})	
	Begin/ Termination Patterns	Termination	Implicit Termination (wcp_{11}) Explicit Termination (wcp_{43})	Aktivitäten sind aktiv und sollten beendet werden oder es können keine weiteren Aktivitäten ausgeführt werden.
		Cancel Activity	Cancel Task (wcp_{19}) Cancel Case (wcp_{20}) Cancel Region (wcp_{25}) Cancel Multiple Instance Activity (wcp_{26})	
	Selection Patterns	Choice	Exclusive Choice (wcp_4) Deferred Choice (wcp_{16})	Auswahl von nachfolgenden auszuführenden Aktivitäten.
			Multi-Choice (wcp_6)	
	Parallel Split Patterns	Parallel Split	Parallel Split (wcp_2) Multiple Instances without Synchronization (wcp_{12})	Aufspaltung des Kontrollflusses, damit Aktivitäten in paralleler oder beliebiger Reihenfolge ausgeführt werden können.
		Synchronization Parallel Split	Thread Split (wcp_{42})	
	Trigger Patterns	Tigger	Transient Trigger (wcp_{23}) Persistent Trigger (wcp_{24})	Externe Einflüsse beeinflussen den Kontrollfluss.

Tabelle 6: Überblick - Kontrollflussmuster nach [Börg07] - Teil 1/2

Kategorie	Muster-typ	Muster [Börg07]	Zuordnung zu [RHAM06]	Erläuterung
Parallel Control Flow Pattern *(Parallele Kontrollflussmuster)*	Interleaving Patterns	Interleaved Parallel Routing	Interleaved Parallel Routing (wcp_{17}) Critical Section (wcp_{39}) Interleaved Routing (wcp_{40})	Ausführung von Aktivitäten in beliebiger Reihenfolge, die zur Laufzeit festgelegt werden. Es können keine zwei Aktivitäten gleichzeitig ausgeführt wird.
	Merge Pattern	Merge	Thread Merge (wcp_{41}) Simple Merge (wcp_5) Multi-Merge (wcp_8) Static Partial Join for Multiple Instances (wcp_{34}) Multiple Instances with a Priori Design-Time Knowledge (wcp_{13}) Multiple Instances with a Priori Run-Time Knowledge (wcp_{14}) Multiple Instances without a Priori Run-Time Knowledge (wcp_{15}) Cancelling Partial Join for Multiple Instances (wcp_{35}) Complete Multiple Instance Activity (wcp_{27}) Dynamic Partial Join for Multiple Instances (wcp_{36}) Structured Synchronizing Merge (wcp_7) Synchronization (wcp_3) Local Synchronizing Merge (wcp_{37}) General Synchronizing Merge (wcp_{38}) Structured Discriminator (wcp_9) Blocking Discriminator (wcp_{28}) Cancelling Discriminator (wcp_{29}) Structured Partial Join (wcp_{30}) Blocking Partial Join (wcp_{31}) Cancelling Partial Join (wcp_{32}) Generalized AND-Join (wcp_{33})	Zusammenführung des Kontrollflusses.

Tabelle 7: Überblick - Kontrollflussmuster nach [Börg07] - Teil 2/2

4.1.2 Kontrollflussmuster der Workflow Pattern Initiative

In Tabelle 4 und Tabelle 5 wurden die 43 Kontrollflussmuster der Workflow Pattern Initiative aufgelistet und in die 8 Kategorien: Kontrollflussbasismuster, erweiterte Verzweigungs- und Synchronisationsmuster, Mehrfachinstanzmuster, zustandsbasierte Muster, Abbruchmuster und Muster mit erzwungenen Ende, Iterationsmuster, Beendigungsmuster und Triggermuster gruppiert. Im Folgenden wird jede Kategorie motivierend eingeführt und die darin beinhaltenden Kontrollflussmuster vorgestellt. Darüber hinaus wird die Kontrollflusssemantik jedes Musters mit einer newYAWL-Spezifikation beschrieben, woraus die zu spezifizierenden Parameter der newYAWL-Spezifikation für die präzise Beschreibung der Ablaufreihenfolge der Elemente eines Geschäftsprozessmodells abgeleitet werden können. Aus Gründen der Übersicht werden die Elemente C, i, o, T, T_A, T_C, M, F einer newYAWL-Spezifikation nicht aufgeführt.

Für die nachfolgenden Ausführungen werden die Begriffe *Vor-* und *Nachbereich* sowie *aktiviert* benötigt. Der *Vorbereich* eines betrachteten Elementes wird definiert durch alle Elemente, die eine Kante zum betrachteten Element besitzen. Die Elemente A und D sind der Vorbereich von Element B in Abbildung 28. Der *Nachbereich* eines betrachteten Elementes wird definiert durch alle Elemente, die eine eingehende Kante vom betrachteten Element besitzen (vgl. Petri-Netze in Kapitel 3.5). Der Nachbereich von Element A sind die Elemente B und C in Abbildung 28. Ein Element ist *aktiviert*, wenn alle notwendigen Elemente im Vorbereich ausgeführt wurden und auf dessen Grundlage das zu aktivierende Element ausgeführt werden kann. Die notwendigen ausgeführten Elemente im Vorbereich ergeben sich durch die Vorbedingung des zu aktivierenden Elementes. Die möglichen nachfolgenden auszuführenden Elemente eines ausgeführten Elementes ergeben sich aus den Nachbedingungen. Die Vor- und Nachbedingungen eines Elementes werden mit Kontrollflussmustern definiert. Beispielsweise existieren die Elemente A, B, C und D (vgl. Abbildung 28). Das Element A besitzt als Nachbedingung das Kontrollflussmuster exklusive Auswahl (wcp_4) und ist mit den Elementen B und C verbunden. Das zu aktivierende Element B besitzt als Vorbedingung das Kontrollflussmuster Synchronisation (wcp_3). Das Element D besitzt das Kontrollflussmuster Sequenz (wcp_1) als Nachbedingung, wie auch das Element C als Vorbedingung. Damit das Element B aktiviert werden kann, muss zum einen nach der Ausführung des Elementes A, das Element B bei der exklusiven Auswahl ausgewählt werden. Zum anderen muss das Element D ausgeführt sein, damit das Element B aktiviert werden kann. Es können nur aktivierte Elemente ausgeführt werden. Eine Deaktivierung ist nur durch die Ausführung des Elementes möglich.

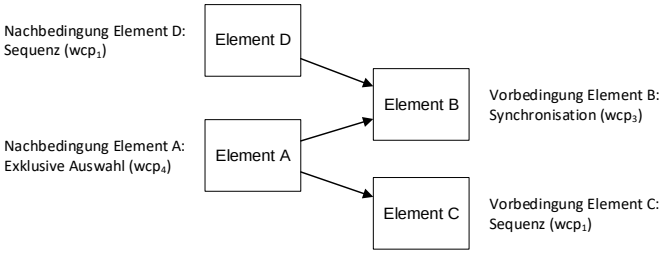


Abbildung 28: Beispiel - Kontrollflussmuster als Vor- und Nachbedingung

4.1.2.1 Kontrollflussbasismuster

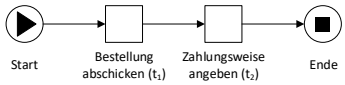
Die Kategorie der Kontrollflussbasismuster beinhaltet die Muster zur elementaren Beschreibung einer möglichen Ablaufreihenfolge von Elementen in Geschäftsprozessmodellen, wie auch aus den ersten Konzepten der Workflow Management Coalition (WfMC) hervorgeht [WfMC99]. Zu der Kategorie zählen die Muster Sequenz (engl. sequence; Workflow Control Pattern 1 - wcp_1), parallele Aufspaltung (engl. parallel split; wcp_2), Synchronisation (engl. synchronization; wcp_3), exklusive Auswahl (engl. exclusive choice; wcp_4) und einfache Zusammenführung (engl. simple merge; wcp_5). [AHKB03, RHAM06, Russ07].

Das Kontrollflussmuster *Sequenz* (Workflow Control Pattern 1 - wcp_1) kennzeichnet eine Reihe von aufeinanderfolgenden Elementen, wie beispielsweise Aktivitäten, die nacheinander ausgeführt werden. Nach der Beendigung des Elementes kann das Element im Nachbereich aktiviert werden [AHKB03, RHAM06, Russ07].

Sequenz (engl. sequence; wcp_1)

BESCHREIBUNG: Ein nachfolgendes Element kann unmittelbar aktiviert werden, wenn das vorausgehende Element ausgeführt wurde.

BEISPIEL: Auf die Aufgabe *Bestellung abschicken* folgt die Aufgabe *Zahlungsweise angeben*.



NEWYAWL: Es muss eine Kante in der Flussrelation F vorliegen. Im obigen Beispiel existiert die Kante zwischen den zwei Aufgaben t_1 und t_2, d. h. $(t_1, t_2) \in F$ mit $t_1 \in T$ *(Bestellung abschicken)* und $t_2 \in T$ *(Zahlungsweise angeben)*.

NEWYAWL-PARAMETER: keine

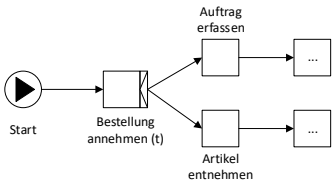
Das Kontrollflussmuster *parallele Aufspaltung* (wcp_2) beschreibt die nebenläufige Ausführung von Elementen, so dass alle Elemente im Nachbereich aktiviert werden. Für die

Vorbedingungen der Elemente im Nachbereich wird angenommen, dass diese mit dem Kontrollflussmuster Sequenz (wcp_1) definiert sind. Die aktivierten Elemente können unabhängig voneinander ausgeführt werden. Die Aufspaltung kann zu einem späteren Zeitpunkt wieder synchronisiert werden [AHKB03, RHAM06, Russ07]. Für die nachfolgenden Beschreibungen wird angenommen, dass die Vor- und Nachbedingungen jeweils mit dem Kontrollflussmuster Sequenz beschrieben sind, sofern diese nicht anders definiert werden.

Parallele Aufspaltung (engl. parallel split; wcp_2)

BESCHREIBUNG: Aufspaltung des Kontrollflusses an einem Element. Alle Elemente im Nachbereich des Aufspaltungselementes werden aktiviert, so dass diese nebenläufig und unabhängig voneinander ausgeführt werden können.

BEISPIEL: Auf die Aufgabe *Bestellung annehmen* folgt die nebenläufige Ausführung der Aufgaben *Auftrag erfassen* und *Artikel entnehmen*.



NEWYAWL: Die Split-Funktion muss für die Aufgabe $t \in T$ (*Bestellung annehmen*) mit dem Funktionswert AND spezifiziert werden, d. h. $\mathrm{Split}(t) = \mathrm{AND}$.

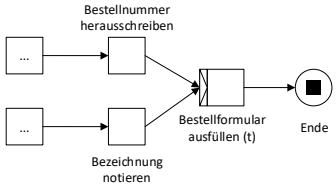
NEWYAWL-PARAMETER: Split

Das Kontrollflussmuster *Synchronisation* (wcp_3) beschreibt die synchronisierte Zusammenführung des Kontrollflusses an einem Element. An einer solchen Stelle im Geschäftsprozessmodell kann der Kontrollfluss erst fortgeführt werden, wenn alle Elemente im Vorbereich ausgeführt wurden. Die nebenläufige Ausführung der Elemente wurde typischerweise durch eine parallele Aufspaltung (wcp_2) im vorherigen Verlauf des Geschäftsprozessmodells erzeugt [AHKB03, RHAM06, Russ07].

Synchronisation (engl. synchronization; wcp_3)

BESCHREIBUNG: Synchronisation des Kontrollflusses an einem Element. Der Kontrollfluss kann erst fortgeführt werden, wenn alle Elemente im Vorbereich ausgeführt wurden.

BEISPIEL: Die Aufgaben *Bestellnummer herausschreiben* und *Bezeichnung notieren* werden nebenläufig ausgeführt und synchronisiert, bevor die Aufgabe *Bestellformular ausfüllen* ausgeführt werden kann.



NEWYAWL: Die Join-Funktion muss für die Aufgabe $t \in T$ (*Bestellformular ausfüllen*) mit dem Funktionswert AND spezifiziert werden, d. h. $\mathrm{Join}(t) = \mathrm{AND}$.

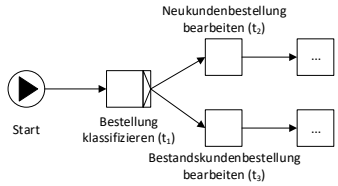
NEWYAWL-PARAMETER: Join

Das Kontrollflussmuster *exklusive Auswahl* (wcp_4) beschreibt die Verzweigung des Kontrollflusses, der aber nur an einem Element von mehreren möglichen Elementen im Nachbereich fortgesetzt werden kann. Die Auswahl des Elementes, an dem der Kontrollfluss fortgesetzt wird, basiert auf dem Ergebnis eines bereits ausgeführten Elementes, der Auswertung eines Ausdrucks, programmatischen Auswahlmechanismen oder spezifischen Datenelementen. Das Element wird somit dynamisch zur Laufzeit bestimmt [AHKB03, RHAM06, Russ07].

Exklusive Auswahl (engl. exclusive choice; wcp_4)

BESCHREIBUNG: Verzweigung des Kontrollflusses an einem Element. Der Kontrollfluss wird abhängig von den Verzweigungsbedingungen an genau einem Element im Nachbereich fortgesetzt. Die Auswahl basiert auf den Daten der Geschäftsprozessinstanz.

BEISPIEL: Nach der Aufgabe *Bestellung klassifizieren* kann entweder die Aufgabe *Neukundenbestellung bearbeiten* oder die Aufgabe *Bestandskundenbestellung bearbeiten* ausgeführt werden.



NEWYAWL: Die Split-Funktion (Split), die Reihenfolge der Auswertungssequenz ($<_{\mathrm{XOR}}$), die Kantenbedingungen (ArcCond) und die Standardkante (Default) müssen für die Aufgabe $t_1 \in T$ (*Bestellung klassifizieren*) definiert werden. Die Split-Funktion muss für die Aufgabe $t_1 \in T$ mit dem Funktionswert XOR spezifiziert werden, d. h. $\mathrm{Split}(t_1) = \mathrm{XOR}$. Zusätzlich muss die Reihenfolge der Auswertungssequenz der ausgehenden Kanten für t_1 in $<_{\mathrm{XOR}}$ festgelegt werden ($<_{\mathrm{XOR}} = \{(t_1, \{(1, t_2), (2, t_3)\})\}$). In ArcCond wird für jede ausgehende Kante von t_1 eine Kantenbedingung festgelegt, d. h. $\mathrm{ArcCond} = \{((t_1, t_2), \mathrm{condition}_1), ((t_1, t_3), \mathrm{condtion}_2)\}$. In der Menge

Default wird die Kante definiert, die ausgewählt wird, wenn keine der vorhandenen Kantenbedingungen positiv ausgewertet wird ($\mathrm{Default} = \{(t_1, t_2)\} \subset F$).

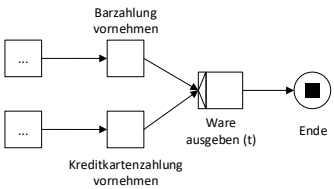
NEWYAWL-PARAMETER: Split, $<_{XOR}$, ArcCond und Default

Das Kontrollflussmuster *einfache Zusammenführung* (wcp_5) ist ein Muster, um den Kontrollfluss an einem Element ohne Synchronisation zusammenzuführen. Das Muster kann beispielsweise zur Vereinfachung eines Geschäftsprozessmodells verwendet werden. Dementsprechend kann der Kontrollfluss zusammengeführt werden, so dass das nachfolgende auszuführende Element nur einmal im Geschäftsprozessmodell modelliert werden muss [AHKB03, RHAM06, Russ07].

Einfache Zusammenführung (engl. simple merge; wcp_5)

BESCHREIBUNG: Zusammenführung des Kontrollflusses an einem Element, ohne Synchronisation.

BEISPIEL: Nach der Aufgabe *Barzahlung vornehmen* oder der Aufgabe *Kreditkartenzahlung vornehmen* kann die Aufgabe *Ware ausgeben* ausgeführt werden.



NEWYAWL: Die Join-Funktion muss für die Aufgabe $t \in T$ (*Ware ausgeben*) mit dem Funktionswert XOR spezifiziert werden, d. h. $\mathrm{Join}(t) = \mathrm{XOR}$.

NEWYAWL-PARAMETER: Join

4.1.2.2 Erweiterte Verzweigungs- und Synchronisationsmuster

Die Muster aus der Kategorie der erweiterten Verzweigung und Synchronisation beschreiben im Vergleich zur vorherigen Kategorie komplexere Kontrollflussmuster, die in den Geschäftsprozessen auftreten können. Die Kontrollflussmuster aus dieser Kategorie werden jedoch nur vereinzelt von den Modellierungssprachen unterstützt [Dres16b]. In [AHKB03] wurden zunächst die vier Kontrollflussmuster Mehrfachauswahl (engl. multi-choice; wcp_6), synchronisierte Zusammenführung, Mehrfachzusammenführung (engl. multi-merge; wcp_8) und Diskriminator eingeführt. In der überarbeiteten Fassung [RHAM06] wurden die Kontrollflussmuster Mehrfachauswahl (wcp_6) und Mehrfachzusammenführung (wcp_8) nicht verändert, sondern nur die Kontrollflusssemantik mit gefärbten Petri-Netz-Beispielen präzisiert. Dahingegen wurde das Kontrollflussmuster synchronisierte Zusammenführung in

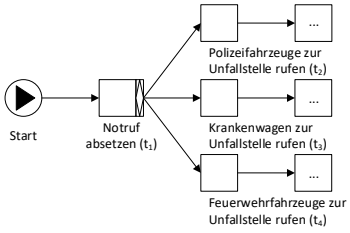
der erweiterten Fassung [RHAM06] in drei Varianten aufgegliedert, d. h. in die strukturierte synchronisierte Zusammenführung (engl. structured synchronizing merge; wcp_7), lokale synchronisierte Zusammenführung (engl. local synchronizing merge; wcp_{37}) und generalisierte synchronisierte Zusammenführung (engl. general synchronizing merge; wcp_{38}). Das Muster Diskriminator wurde in sechs Varianten aufgegliedert, d. h. in strukturierter Diskriminator (engl. structured discriminator; wcp_9), blockierender Diskriminator (engl. blocking discriminator; wcp_{28}), abbrechender Diskriminator (engl. cancelling discriminator; wcp_{29}), strukturierte partielle Zusammenführung (engl. structured partial join; wcp_{30}), blockierende partielle Zusammenführung (engl. blocking partial join; wcp_{31}) und abbrechende partielle Zusammenführung (engl. cancelling partial join; wcp_{32}). Bei der generalisierten Und-Zusammenführung (engl. generalized and-join; wcp_{33}) muss jedes Element im Vorbereich mindestens einmal ausgeführt sein, wobei diese Elemente durch verschiedene Trigger ausgelöst sein können. Die Kontrollflussmuster Thread-Zusammenführung (engl. thread merge; wcp_{41}) und Thread-Aufspaltung (engl. thread split; wcp_{42}) spalten den Kontrollfluss in Threads auf bzw. führen diese Threads später wieder zusammen (vgl. Kapitel 3.6) [AHKB03, RHAM06, Russ07].

Mit dem Kontrollflussmuster *Mehrfachauswahl* (wcp_6) wird ein logisches Oder modelliert, sodass der Kontrollfluss an beliebig vielen Elementen im Nachbereich der Mehrfachauswahl fortgesetzt werden kann. Der Kontrollfluss wird abhängig von den Verzweigungsbedingungen an den entsprechenden Elementen fortgesetzt. Die Auswahl der Elemente basiert auf den zugrundeliegenden Daten der Geschäftsprozessinstanz. Dieses Muster ist der exklusiven Auswahl (wcp_4) ähnlich, mit dem Unterschied, dass auch mehrere nachfolgende Elemente aktiviert werden können [AHKB03, RHAM06, Russ07].

Mehrfachauswahl (engl. multi choice; wcp_6)

BESCHREIBUNG: Verzweigung des Kontrollflusses an einem Element. Der Kontrollfluss wird abhängig von den Verzweigungsbedingungen an einem oder mehreren Elementen fortgesetzt. Die Auswahl basiert auf den Daten der Geschäftsprozessinstanz.

BEISPIEL: Nach der Aufgabe *Notruf absetzen* werden, abhängig vom Inhalt des Notrufes, *Polizeifahrzeuge*, *Feuerwehrfahrzeuge* und/oder ein *Krankenwagen* an der Unfallstelle benötigt.



NEWYAWL: Die Split-Funktion (Split), die Kantenbedingungen (ArcCond) und die Standardkante (Default) müssen für die Aufgabe $t_1 \in T$ (*Notruf absetzen*) definiert werden. Die Split-Funktion muss mit dem Funktionswert OR spezifiziert werden, d. h. $\mathrm{Split}(t_1) = \mathrm{OR}$. In ArcCond wird für jede ausgehende Kante von t_1 eine Kantenbedingung festgelegt, d. h. $\mathrm{ArcCond} = \{((t_1, t_2), \mathrm{condition}_1), ((t_1, t_3), \mathrm{condtion}_2), ((t_1, t_4), \mathrm{condtion}_3)\}$. In der Menge Default wird die Standardkante beschrieben, die ausgewählt wird, wenn keine der definierten Kantenbedingungen positiv ausgewertet wird ($\mathrm{Default} = \{(t_1, t_2)\} \subset F$). Im obigen Beispiel ist dies die Kante von der Aufgabe *Notruf absetzen* (t_1) zu der Aufgabe *Polizeifahrzeuge zur Unfallstelle berufen* (t_4).

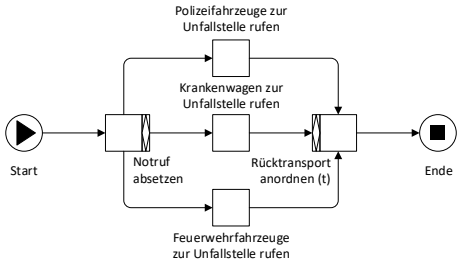
NEWYAWL-PARAMETER: Split, ArcCond und Default

Das Kontrollflussmuster *strukturierte synchronisierte Zusammenführung* (wcp_7) synchronisiert den Kontrollfluss, der durch eine Mehrfachauswahl (wcp_6) in einem vorgelagerten Prozessschritt aufgespalten wurde. Die Zusammenführung ist strukturiert. Das bedeutet, dass eine Verzweigung auch wieder zusammengeführt werden muss. Zwischen den Elementen der Mehrfachauswahl und Zusammenführung existiert eine Eins-zu-Eins-Beziehung. Der Kontrollfluss kann an der Zusammenführung erst fortgesetzt werden, wenn der verzweigte Kontrollfluss von der Mehrfachauswahl wieder synchronisiert wurde (Nicht-Lokalitäts-Prinzip [Kind06]) [AHKB03, RHAM06, Russ07].

Strukturierte synchronisierte Zusammenführung (engl. structured synchronizing merge; wcp_7)

BESCHREIBUNG: Zusammenführung des Kontrollflusses an einem Element. Der Kontrollfluss wurde an einem früheren und eindeutig identifizierbaren Prozessschritt durch eine Mehrfachauswahl (wcp_6) verzweigt. Der Kontrollfluss kann an dem Zusammenführungselement erst fortgesetzt werden, wenn der verzweigte Kontrollfluss von der Mehrfachauswahl wieder synchronisiert wurde.

BEISPIEL: Nach der Aufgabe *Notruf absetzen*, werden abhängig vom Inhalt des Notrufes *Polizeifahrzeuge*, *Feuerwehrfahrzeuge* und/oder *ein Krankenwagen zur Unfallstelle* gerufen. Nachdem alle beorderten Fahrzeuge am Unfallort eingetroffen sind, kann die Aufgabe *Rücktransport anordnen* ausgeführt werden.



NEWYAWL: Die Join-Funktion muss für die Aufgabe $t \in T$ (*Rücktransport anordnen*) mit dem Funktionswert OR spezifiziert werden, d. h. $\mathrm{Join}(t) = \mathrm{OR}$.

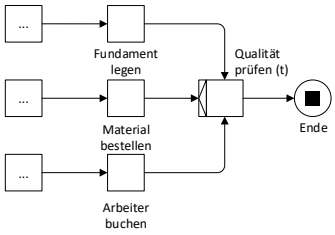
NEWYAWL-PARAMETER: Join

Das Kontrollflussmuster *Mehrfachzusammenführung* (wcp_8) kann zur Zusammenführung des Kontrollflusses verwendet werden, ohne Synchronisation. Im Unterschied zur einfachen Zusammenführung (wcp_5) besteht die Möglichkeit, dass das Mehrfachzusammenführungselement unabhängig n mal aktiviert wird, sodass dies zu einer n-maligen Aktivierung der Elemente im Nachbereich führt [AHKB03, RHAM06, Russ07].

Mehrfachzusammenführung (engl. multi merge: wcp_8)

BESCHREIBUNG: Zusammenführung des Kontrollflusses an einem Element, ohne Synchronisation. Die n-malige Aktivierung des Mehrfachzusammenführungselementes führt zu einer n-maligen Aktivierung der nachfolgenden Elemente.

BEISPIEL: Die Aufgaben *Fundament legen*, *Material bestellen* und *Arbeiter buchen* erfolgen nebenläufig zueinander. Wenn jede Aktivität beendet wurde, muss für jede der vorgelagerten Aufgaben die *Qualität überprüft* werden.



NEWYAWL: Die Join-Funktion muss für die Aufgabe $t \in T$ (*Qualität prüfen*) mit dem Funktionswert XOR spezifiziert werden, d. h. $\mathrm{Join}(t) = \mathrm{XOR}$.

NEWYAWL-PARAMETER: Join

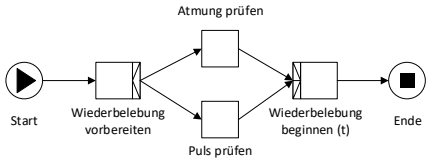
Mit dem Kontrollflussmuster *strukturierter Diskriminator* (wcp_9) kann die Zusammenführung des Kontrollflusses mit einem Diskriminator modelliert werden. Zunächst wird der Kontrollfluss mit einer parallelen Aufspaltung (wcp_2) aufgespalten. Bei der nachfolgenden Zusammenführung soll durch den Diskriminator verhindert werden, dass die nachfolgenden Elemente im Geschäftsprozessmodell mehrfach ausgeführt werden. Sobald ein Element im Vorbereich ausgeführt wurde, ist der Diskriminator aktiviert, so dass weitere ausgeführte Elemente im Vorbereich des Diskriminators den Kontrollfluss nicht beeinflussen. Nach der Aktivierung des Diskriminators können die nachfolgenden Elemente ausgeführt werden. Erst wenn alle Elemente im Vorbereich ausgeführt wurden, wird der Diskriminator

zurückgesetzt bzw. deaktiviert, wodurch eine erneute Aktivierung möglich ist. Die Zurücksetzung ermöglicht die Verwendung des Diskriminators auch in Schleifen. Das Muster ist strukturiert, da eine parallele Aufspaltung (wcp_2) vorausgehen muss [AHKB03, RHAM06, Russ07].

Strukturierter Diskriminator (engl. structured discriminator; wcp_9)

BESCHREIBUNG: Zusammenführung des Kontrollflusses an einem Element. Der Kontrollfluss wurde durch eine vorherige parallele Aufspaltung (wcp_2) aufgespalten. Der Kontrollfluss kann fortgesetzt werden, sobald eines der Elemente im Vorbereich ausgeführt wurde. Gleichzeitig wird der Diskriminator aktiviert, woraufhin weitere ausgeführte Elemente im Vorbereich den Kontrollfluss nicht beeinflussen. Der Diskriminator wird erst zurückgesetzt, wenn alle Elemente im Vorbereich ausgeführt wurden. Der Diskriminator kann nach der Zurücksetzung erneut aktiviert werden.

BEISPIEL: Die Aufgabe *Wiederbelebung vorbereiten* wird ausgeführt. Nachfolgend werden die Aufgaben *Atmung prüfen* und den *Puls prüfen* nebenläufig ausgeführt. Sobald eine der beiden Aufgaben abgeschlossen ist, kann mit der *Wiederbelegung* begonnen werden. Die Fertigstellung der anderen Aufgaben hat keinen Einfluss auf die Ausführung der Aufgabe *Wiederbelegung beginnen*.



NEWYAWL: Die Join- und Thresh-Funktion muss für die Aufgabe $t \in T$ (*Wiederbelebung beginnen*) definiert werden. Die Split-Funktion muss mit dem Funktionswert PJOIN spezifiziert werden, d. h. $\mathrm{Join}(t) = \mathrm{PJOIN}$. Zusätzlich muss der Funktionswert der Thresh-Funktion auf 1 gesetzt werden, um die Fortführung des Kontrollflusses nach der Aktivierung des Diskriminators zu beschreiben, d. h. $\mathrm{Thresh}(t) = 1$.

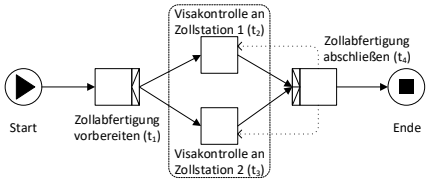
NEWYAWL-PARAMETER: Join und Thresh

Das Kontrollflussmuster *blockierender Diskriminator* (wcp_{28}) ist eine weitere Variante des strukturierten Diskriminator. Im Unterschied zum strukturierten Diskriminator existiert ein blockierender Bereich, der durch die Aktivierung des Diskriminators aktiviert wird. Aus diesem Grund ist keine strukturierte Form erforderlich. Sobald der blockierende Bereich aktiviert ist, kann dieser erst erneut aktiviert werden, wenn der Diskriminator zurückgesetzt wurde [AHKB03, RHAM06, Russ07].

Blockierender Diskriminator (engl. blocking discriminator; wcp$_{28}$)

BESCHREIBUNG: Zusammenführung des Kontrollflusses an einem Element. Der Kontrollfluss kann am Diskriminator fortgesetzt werden, sobald eines der Elemente im Vorbereich des Diskriminators ausgeführt wurde. Gleichzeitig wird der Diskriminator aktiviert, woraufhin die Elemente eines definierten Bereichs nicht erneut ausgeführt werden können. Eine erneute Ausführung der Elemente innerhalb des Bereichs ist erst möglich, wenn der Diskriminator zurückgesetzt wurde. Der Diskriminator wird zurückgesetzt, wenn alle Elemente im Vorbereich des Diskriminators ausgeführt wurden.

BEISPIEL: Zur Einreise können die Visa einer Touristengruppe von verschiedenen Zollstationen gleichzeitig kontrolliert werden. Eine später ankommende Gruppe muss warten, bis die Kontrolle der vorherigen Gruppe vollständig beendet ist.



NEWYAWL: Die Join-, Thresh- und Block-Funktion muss für die Aufgabe $t_4 \in T$ (*Zollabfertigung abschließen*) definiert werden. Die Join-Funktion muss mit dem Funktionswert PJOIN spezifiziert werden, d. h. $\mathrm{Join}(t_4) = \mathrm{PJOIN}$. Darüber hinaus muss der Funktionswert der Thresh-Funktion auf 1 gesetzt werden, um die Fortführung des Kontrollflusses nach dem ersten ausgeführten Element im Vorbereich zu beschreiben, d. h. $\mathrm{Thresh}(t_4) = 1$. Zusätzlich muss der blockierende Bereich für die Aufgabe t_4 definiert werden, der die Menge der Aufgaben umfasst, die blockiert werden müssen, d. h. $\mathrm{Block}(t_4) = \{t_2, t_3\}$.

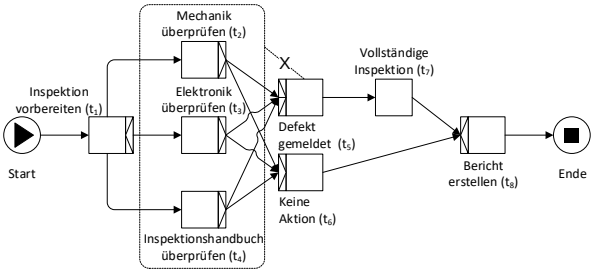
NEWYAWL-PARAMETER: Join, Thresh und Block

Das Kontrollflussmuster *abbrechender Diskriminator* (wcp$_{29}$) ist ebenfalls eine Variante des strukturierten Diskriminator, mit dem Unterschied, dass, wenn der Diskriminator aktiviert wurde, die Elemente aus einem definierten Bereich abgebrochen werden [AHKB03, RHAM06, Russ07].

Abbrechender Diskriminator (engl. cancelling discriminator; wcp$_{29}$)

BESCHREIBUNG: Zusammenführung des Kontrollflusses an einem Element. Der Kontrollfluss kann am Diskriminator fortgesetzt werden, sobald eines der Elemente im Vorbereich des Diskriminators ausgeführt wurde. Gleichzeitig wird der Diskriminator aktiviert, woraufhin die Ausführung der Elemente eines definierten Bereichs abgebrochen wird. Erst wenn alle Elemente abgebrochen sind, wird der Diskriminator zurückgesetzt.

BEISPIEL: Für die Inspektion wird die *Mechanik*, *Elektronik* und das *Inspektionshandbuch überprüft*. Die Überprüfung findet nebenläufig statt. Sofern eine der 3 Überprüfungen negativ ist, werden die anderen Aufgaben unmittelbar abgebrochen und eine vollständige Inspektion durchgeführt. Nachfolgend wird ein Bericht erstellt. Für den Fall, dass alle 3 Überprüfungen positiv sind, wird der Bericht direkt ausgestellt.



NEWYAWL: Die Join-, Thresh- und Rem-Funktion muss für die Aufgabe $t_5 \in T$ (*Defekt gemeldet*) definiert werden. Die Join-Funktion muss mit dem Funktionswert PJOIN spezifiziert werden, d. h. $\mathrm{Join}(t_5) = \mathrm{PJOIN}$. Die Thresh-Funktion muss für die Aufgabe t_5 mit dem Funktionswert 1 spezifiziert werden, um die Fortführung des Kontrollflusses nach der ersten Aktivierung zu beschreiben, d. h. $\mathrm{Thresh}(t_5) = 1$. Der Abbruchbereich wird mit der Rem-Funktion für die Aufgabe t_5 beschrieben, die die Menge der Elemente beinhaltet, die abgebrochen werden sollen ($\mathrm{Rem}(t_5) = \{t_2, t_3, t_4\}$).

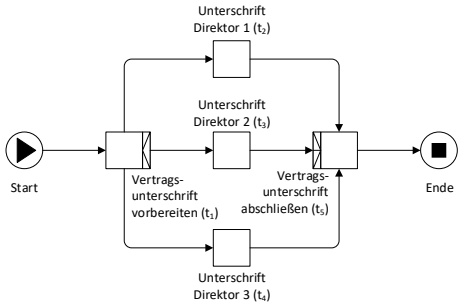
NEWYAWL-PARAMETER: Join, Thresh und Rem

Das Kontrollflussmuster *strukturierte partielle Zusammenführung* (wcp_{30}) ist eine Zusammenführung des Kontrollflusses, bei dem n der m möglichen Elemente im Vorbereich ausgeführt sein müssen, bevor der Kontrollfluss am Zusammenführungselement fortgesetzt werden kann. Das Muster kann somit als Spezialisierung einer Synchronisation (wcp_3) betrachtet werden. Sofern nur ein ausgeführtes Element im Vorbereich für die Fortführung des Kontrollflusses erforderlich ist ($n = 1$), entspricht das Kontrollflussmuster einem Diskriminator (vgl. wcp_9, wcp_{28} und wcp_{29}). Falls eine Ausführung aller Elemente im Vorbereich erforderlich ist ($n = m$), entspricht das Kontrollflussmuster der Synchronisation (wcp_3) und dem Kontrollflussmuster generalisierte UND-Zusammenführung (wcp_{33}). Das Muster liegt in strukturierter Form vor, da eine parallele Aufspaltung vorausgehen muss, die mit der strukturierten partiellen Zusammenführung in Beziehung steht [AHKB03, RHAM06, Russ07].

Strukturierte partielle Zusammenführung (engl. structured partial join; wcp_{30})

BESCHREIBUNG: Zusammenführung des Kontrollflusses an einem Element. Der Kontrollfluss wurde durch eine vorherige parallele Aufspaltung (wcp_2) aufgespalten. Der Kontrollfluss kann fortgesetzt werden, sobald n der m möglichen Elemente im Vorbereich ausgeführt wurden. Gleichzeitig wird die partielle Zusammenführung aktiviert, woraufhin weitere ausgeführte Elemente im Vorbereich den Kontrollfluss nicht beeinflussen. Die partielle Zusammenführung wird erst zurückgesetzt, wenn alle Elemente im Vorbereich ausgeführt wurden. Die partielle Zusammenführung kann nach der Zurücksetzung erneut aktiviert werden.

BEISPIEL: Ein Vertrag ist rechtskräftig, wenn zwei der drei Direktoren den Vertrag unterschrieben haben. Sobald zwei Direktoren den Vertrag unterschrieben haben, ist der Vertrag wirksam. Der dritte Direktor kann den Vertrag unterschreiben oder nicht unterschreiben, was die Fortführung des Kontrollflusses nicht beeinflusst.



NEWYAWL: Die Join- und Thresh-Funktion muss für die Aufgabe $t_5 \in T$ (*Vertragsunterschrift abschließen*) definiert werden. Die Split-Funktion muss mit dem Funktionswert PJOIN spezifiziert werden, d. h. $\mathrm{Join}(t_5) = \mathrm{PJOIN}$. Darüber hinaus muss der Funktionswert der Thresh-Funktion auf n gesetzt werden, um die Fortführung des Kontrollflusses nach n ausgeführten Elementen im Vorbereich zu beschreiben, d. h. in diesem Beispiel ist $\mathrm{Thresh}(t_5) = 2$.

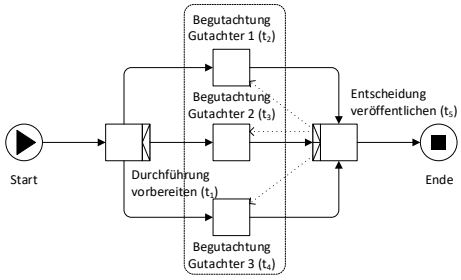
NEWYAWL-PARAMETER: Join und Thresh

Das Kontrollflussmuster *blockierende partielle Zusammenführung* (wcp_{31}) ist eine alternative Variante der strukturierten partiellen Zusammenführung. Anstatt der strukturierten Form wird ein blockierender Bereich verwendet. Der Bereich wird blockiert, sobald die partielle Zusammenführung aktiviert wurde. Der blockierende Bereich kann erst wieder erneut aktiviert werden, wenn die partielle Zusammenführung zurückgesetzt wurde. Die partielle Zusammenführung wird zurückgesetzt, wenn alle m Elemente im Vorbereich ausgeführt wurden [AHKB03, RHAM06, Russ07].

Blockierende partielle Zusammenführung (engl. blocking partial join; wcp_{31})

BESCHREIBUNG: Zusammenführung des Kontrollflusses an einem Element. Der Kontrollfluss kann an der partiellen Zusammenführung fortgesetzt werden, wenn n der m möglichen Elemente im Vorbereich ausgeführt wurden. Gleichzeitig wird die partielle Zusammenführung aktiviert, woraufhin die Elemente eines definierten Bereichs nicht erneut ausgeführt werden können. Eine erneute Ausführung der Elemente innerhalb des Bereichs ist erst möglich, wenn die partielle Zusammenführung zurückgesetzt wurde. Die partielle Zusammenführung wird zurückgesetzt, wenn alle Elemente im Vorbereich der partiellen Zusammenführung ausgeführt wurden.

BEISPIEL: Für die Begutachtung eines wissenschaftlichen Beitrags werden 3 Gutachter benötigt. Sobald zwei der drei Gutachter über die Annahme oder Ablehnung entschieden haben, werden die Gutachten mit den Entscheidungen veröffentlicht. Die beiden Gutachter, die bereits ihr Gutachten erstellt haben, können jedoch erst wieder erneut einen neuen Beitrag begutachten, wenn der dritte Gutachter die Begutachtung beendet hat.



NEWYAWL: Die Join-, Thresh- und Block-Funktion muss für die Aufgabe $t_5 \in T$ (*Entscheidung veröffentlichen*) definiert werden. Die Join-Funktion muss mit dem Funktionswert PJOIN spezifiziert werden, d. h. $\mathrm{Join}(t_5) = \mathrm{PJOIN}$. Darüber hinaus muss der Funktionswert für die Aufgabe t_5 in der Thresh-Funktion auf 2 festgelegt werden, um die Fortführung des Kontrollflusses nach zwei ausgeführten Elementen im Vorbereich zu beschreiben, d. h. $\mathrm{Thresh}(t_5) = 2$. Zusätzlich muss der blockierende Bereich mit der Block-Funktion für die Aufgabe t_5 beschrieben werden, der die Menge der Aufgaben umfasst, die blockiert werden müssen ($\mathrm{Block}(t_5) = \{t_2, t_3, t_4\}$).

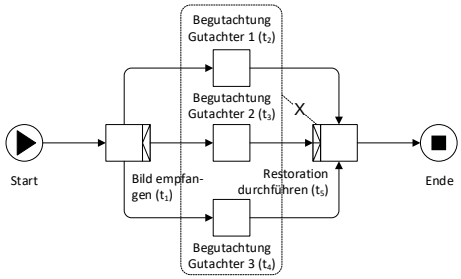
NEWYAWL-PARAMETER: Join, Thresh und Block

Das Kontrollflussmuster *abbrechende partielle Zusammenführung* (wcp_{32}) ist ebenfalls eine Variante der strukturierten partiellen Zusammenführung, mit dem Unterschied, dass, wenn die partielle Zusammenführung aktiviert wurde, die Elemente aus einem definierten Bereich abgebrochen werden [AHKB03, RHAM06, Russ07].

Abbrechende partielle Zusammenführung (engl. cancelling partial join; wcp_{32})

BESCHREIBUNG: Zusammenführung des Kontrollflusses an einem Element. Der Kontrollfluss kann an der partiellen Zusammenführung fortgesetzt werden, wenn n der m möglichen Elemente im Vorbereich ausgeführt wurden. Gleichzeitig wird die partielle Zusammenführung aktiviert, woraufhin die Ausführung der Elemente eines definierten Bereichs abgebrochen wird. Erst wenn alle Elemente abgebrochen sind, wird die partielle Zusammenführung zurückgesetzt.

BEISPIEL: Ein Kunstwerk wird empfangen und nachfolgend an drei Kunsthändler zur Begutachtung weitergegeben. Sobald zwei der drei Begutachtungen vorliegen, werden die verbleibenden Begutachtungen abgebrochen und die Restauration kann begonnen werden.



NEWYAWL: Die Join-, Thresh- und Rem-Funktion muss für die Aufgabe $t_5 \in T$ (*Restauration durchführen*) definiert werden. Die Join-Funktion muss mit dem Funktionswert PJOIN spezifiziert werden, d. h. $\mathrm{Join}(t_5) = \mathrm{PJOIN}$. Darüber hinaus muss der Funktionswert für die Aufgabe t_5 in der Thresh-Funktion auf 2 gesetzt werden, um die Fortführung des Kontrollflusses nach zwei ausgeführten Elementen im Vorbereich zu beschreiben, d. h. $\mathrm{Thresh}(t_5) = 2$. Zusätzlich muss der Abbruchbereich mit der Rem-Funktion für die Aufgabe t_5 definiert werden, der die Menge der Aufgaben umfasst, die abgebrochen werden ($\mathrm{Rem}(t_5) = \{t_2, t_3, t_4\}$).

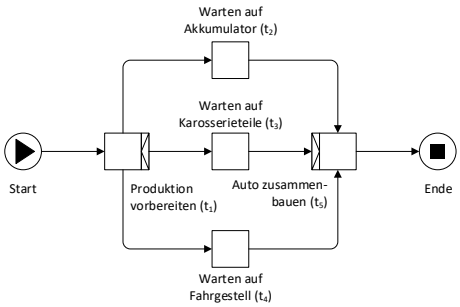
NEWYAWL-PARAMETER: Join, Thresh und Rem

Das Kontrollflussmuster *generalisierte Und-Zusammenführung* (wcp_{33}) kann für die Synchronisation verwendet werden. Dementsprechend kann der Kontrollfluss erst fortgeführt werden, wenn alle Elemente im Vorbereich ausgeführt wurden. Die parallele Ausführung von Elementen resultiert typischerweise aus einer parallelen Aufspaltung (wcp_2) in einem vorherigen Prozessschritt. Im Unterschied zum Kontrollflussmuster Synchronisation (wcp_3) können die Elemente im Vorbereich mehrfach ausgeführt worden sein, sofern z. B. ein Element im Vorbereich n-Mal ausgeführt wurde und alle anderen Elemente im Vorbereich nur 1-Mal ausgeführt wurden, dann kann das Zusammenführungselement genau einmal ausgeführt werden. Für die erneute Aktivierung des Zusammenführungselementes muss jedes Element im Vorbereich mindestens einmal ausgeführt worden sein [AHKB03, RHAM06, Russ07].

Generalisierte Und-Zusammenführung (engl. generalized and-join; wcp_{33})

BESCHREIBUNG: Synchronisation des Kontrollflusses an einem Element. Der Kontrollfluss kann erst fortgesetzt werden, wenn alle Elemente im Vorbereich mindestens einmal ausgeführt wurden. Eine mehrfache Ausführung der Elemente im Vorbereich ist möglich.

BEISPIEL: Die Produktion wird vorbereitet, indem auf die *Akkumulatoren, Karosserieteile* und das *Fahrgestell* aus den verschiedenen Produktionslinien gewartet wird. Auf die Akkumulatoren muss immer sehr lange gewartet werden, sodass während dessen bereits mehrere Fahrgestelle oder Karosserieteile eingetroffen sein können. Sofern jedes Teil einmal vorhanden ist, kann ein Auto zusammengebaut werden.



NEWYAWL: Die Join-Funktion muss für die Aufgabe $t_5 \in T$ (*Auto zusammenbauen*) mit dem Funktionswert AND spezifiziert werden, d. h. $\mathrm{Join}(t_5) = \mathrm{AND}$.

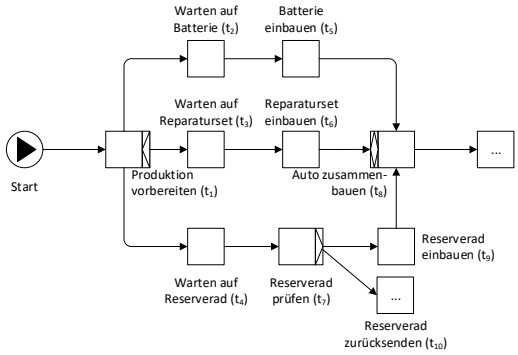
NEWYAWL-PARAMETER: Join

Das Kontrollflussmuster *lokale synchronisierte Zusammenführung* (wcp_{37}) führt den Kontrollfluss zusammen, der durch eine Mehrfachauswahl (wcp_6) aufgespalten wurde. Im Vergleich zum Kontrollflussmuster strukturierte synchronisierte Zusammenführung (wcp_7) ist das Kontrollflussmuster nicht strukturiert. Stattdessen werden die Zusammenführungsinformationen direkt beim Zusammenführungselement durch sogenannte true/false-Bedingungen ausgewertet [AHKB03, RHAM06, Russ07].

Lokale Synchronisierte Zusammenführung (engl. local synchronizing merge; wcp_{37})

BESCHREIBUNG: Synchronisation des Kontrollflusses an einem Element. Der Kontrollfluss kann erst fortgeführt werden, wenn alle erforderlichen Elemente im Vorbereich ausgeführt wurden. Die erforderlichen Elemente werden durch lokale Bedingungen am Synchronisationselement ermittelt. Eine lokale Bedingung ist wahr, wenn das Element im Vorbereich ausgeführt wurde oder das Element nicht mehr ausgeführt werden kann. Andernfalls ist die Bedingung falsch.

BEISPIEL: Für die Produktion eines Fahrzeuges wird eine *Batterie*, ein *Reserverad* und/oder ein *Reparaturset* benötigt. Es kommt häufiger vor, dass das falsche Reserverad geliefert wurde, sodass das Auto auch ohne ein Reserverad fertiggestellt werden kann. Das falsche Reserverad wird zurückgesendet.



NEWYAWL: Die Join-Funktion muss für die Aufgabe $t_8 \in T$ (*Auto zusammenbauen*) mit dem Funktionswert OR spezifiziert werden, d. h. $\mathrm{Join}(t_8) = \mathrm{OR}$. Eine zusätzliche Bedingung ist, dass keine der eingehenden Zweige von t_8 Teil eines Zyklus ist.

NEWYAWL-PARAMETER: Join

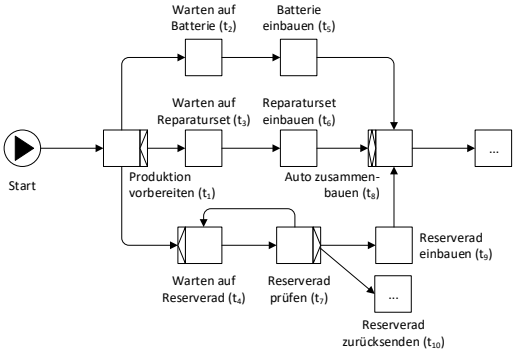
Das Kontrollflussmuster *generalisierte synchronisierte Zusammenführung* (wcp_{38}) synchronisiert den Kontrollfluss, der in einem vorherigen Prozessschritt durch eine Mehrfachauswahl (wcp_6) aufgespalten wurde. Im Vergleich zur lokalen synchronisierten Zusammenführung (wcp_{37}) werden keine lokalen true/false-Bedingungen ausgewertet. Die Auswertung der Zusammenführung basiert auf den aktuell und noch potentiell möglichen auszuführenden Elementen im Vorbereich der Zusammenführung. Dementsprechend werden für die Aktivierung der Zusammenführung die noch möglichen Zustände der Geschäftsprozessinstanz ausgewertet. Mit diesem Kontrollflussmuster wird im Vergleich zur lokalen synchronisierten Zusammenführung die Modellierung von Schleifen auch ohne die Anwendung eines strukturierten Kontrollflussmusters ermöglicht [AHKB03, RHAM06, Russ07].

Generelle Synchronisierte Zusammenführung (engl. general synchronizing merge; wcp_{38})

BESCHREIBUNG: Synchronisation des Kontrollflusses an einem Element. Der Kontrollfluss kann fortgeführt werden, wenn alle erforderlichen Elemente im Vorbereich ausgeführt wurden. Die erforderlichen Elemente werden durch die Überprüfung der gesamten Geschäftsprozessinstanz ermittelt.

BEISPIEL: Für die Produktion eines Fahrzeuges wird eine *Batterie*, ein *Reserverad* und/oder ein *Reparaturset* benötigt. Es kommt häufiger vor, dass das falsche Reserverad geliefert wurde, sodass

das Auto ohne ein Reserverad fertiggestellt wird. Es besteht aber auch die Möglichkeit, auf das richtige Reserverad zu warten. Das falsche Reserverad wird zurückgesendet.



NEWYAWL: Die Join-Funktion muss für die Aufgabe $t_8 \in T$ (*Auto zusammenbauen*) mit dem Funktionswert OR spezifiziert werden, d. h. $\mathrm{Join}(t_8) = \mathrm{OR}$.

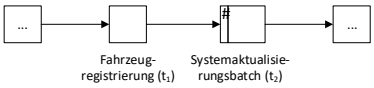
NEWYAWL-PARAMETER: Join

Mit dem Kontrollflussmuster *Thread-Zusammenführung* (wcp_{41}) können einzelne Threads wieder zusammengeführt werden. Ein Thread ist eine Ausführungsreihenfolge für die Abarbeitung eines Programms. Übertragen auf Geschäftsprozessmodelle ist ein Thread ein Bereich eines Geschäftsprozessmodells der zwischen einer Thread-Verzweigungs- (wcp_{42}) und einer -Zusammenführungsaufgabe (wcp_{41}) mehrfach ausgeführt werden kann (vgl. Kapitel 3.6). Sofern mehrere Threads nebenläufig ausgeführt werden, teilen sich diese Threads die entsprechenden Ressourcen [AHKB03, RHAM06, Russ07].

Thread-Zusammenführung (engl. thread merge; wcp_{41})

BESCHREIBUNG: Zusammenführung des Kontrollflusses durch eine bestimmte Anzahl von Threads.

BEISPIEL: Fahrzeuge werden in Tranchen im Fahrzeugregistrierungssystem aktualisiert, sodass, wenn 10 Fahrzeuge registriert wurden, das System einmal aktualisiert wird. Registrierungsprozesse laufen unabhängig voneinander ab.



NEWYAWL: Die Join- und ThreadIn-Funktion muss für die Aufgabe $t_2 \in T$ (*Systembatchaktualisierung*) definiert werden. Die Join-Funktion muss mit dem Funktionswert THREAD spezifiziert werden, d. h. $\mathrm{Join}(t_2) = \mathrm{THREAD}$. Zusätzlich muss die ThreadIn-Funktion für die Threadanzahl bis

zur Zusammenführung festgelegt werden, d. h. im obigen Beispiel ist die Threadanzahl 10, so dass $\mathrm{ThreadIn}(t_2) = 10$ gilt.

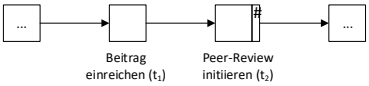
NEWYAWL-PARAMETER: Join und ThreadIn

Das Kontrollflussmuster *Thread-Aufspaltung* (wcp_{42}) ist das Pendant zum Kontrollflussmuster Thread-Zusammenführung (wcp_{41}). Über einen Zweig können mehrere Threads erzeugt werden, welche durch das Kontrollflussmuster Thread-Zusammenführung wieder zusammengeführt werden können [AHKB03, RHAM06, Russ07].

Thread-Aufspaltung (engl. thread split; wcp_{42})

BESCHREIBUNG: Aufspaltung des Kontrollflusses in eine bestimmte Anzahl von Threads.

BEISPIEL: Nach der Abgabe eines wissenschaftlichen Beitrags werden drei unabhängige Peer-Review-Aufgaben initiiert.



NEWYAWL: Die Split- und ThreadOut-Funktion muss für die Aufgabe $t_2 \in T$ (*Peer-Review initiieren*) definiert werden. Die Split-Funktion muss mit dem Funktionswert THREAD spezifiziert werden, d. h. $\mathrm{Split}(t_2) = \mathrm{THREAD}$. Zusätzlich muss die ThreadOut-Funktion für die Threadanzahl zur Aufspaltung festgelegt werden, d. h., im obigen Beispiel ist die Threadanzahl 3, so dass $\mathrm{ThreadOut}(t_2) = 3$ gilt.

NEWYAWL-PARAMETER: Split und ThreadOut

4.1.2.3 Mehrfachinstanzmuster

Die Kontrollflussmuster aus der Kategorie Mehrfachinstanzmuster beschreiben Szenarien zur Erzeugung von mehreren Instanzen desselben Elementes. Dabei werden Mehrfachinstanzen in drei verschiedenen Fällen benötigt:

1. Ein Element kann von sich selbst mehrere Instanzen erzeugen, während das Element ausgeführt wird.

2. Ein Element kann mehrfach unabhängig voneinander initiiert werden, wie z. B. in einer Schleife oder einer Geschäftsprozessinstanz, bei der mehrere Threads existieren.

3. Zwei oder mehrere Elemente definieren eine nicht überlappungsfreie Tätigkeit. Für die Ausführung der Elemente muss das Element entweder mehrfach ausgelöst werden oder über eine gemeinsame Teilprozessdefinition verfügen.

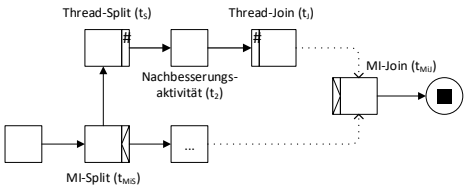
Die drei Fälle beschreiben Varianten, bei denen mehrere Instanzen von einem Element benötigt werden. Der erste Fall ist im Kontext der Kontrollflussmuster am interessantesten, da bei diesem Fall mehrere Instanzen von einem Element erzeugt werden. Die Kontrollflussmuster der Workflow Pattern Initiative aus der Kategorie Mehrfachinstanzmuster bilden Szenarien des ersten Falls ab. Die anderen Fälle können mit diesen Mustern und/oder durch eine Kombination anderer Kontrollflussmuster modelliert werden. Die Kontrollflussmuster Mehrfachinstanz ohne Synchronisation (wcp_{12}), Mehrfachinstanz mit Festlegung der Anzahl der Instanzen zur Entwurfszeit (wcp_{13}), Mehrfachinstanz mit Festlegung der Anzahl der Instanzen vor der Laufzeit des Elementes (wcp_{14}) und Mehrfachinstanz mit Festlegung der Anzahl der Instanzen während der Laufzeit des Elementes (wcp_{15}) zählen zu den ursprünglichen Mustern aus [AHKB03]. Im Rahmen der Reorganisation wurde die Kategorie um drei Kontrollflussmuster erweitert. Dies sind die statische partielle Zusammenführung für Mehrfachinstanzen (wcp_{34}), die abbrechende partielle Zusammenführung für Mehrfachinstanzen (wcp_{35}) und die dynamische partielle Zusammenführung für Mehrfachinstanzen (wcp_{36}) [AHKB03, RHAM06, Russ07].

Mit dem Kontrollflussmuster *Mehrfachinstanz ohne Synchronisation* (wcp_{12}) können mehrere Instanzen eines Elementes erzeugt werden. Die benötigte Anzahl der Instanzen wird zur Entwurfszeit festgelegt. Alle Instanzen werden unabhängig voneinander ausgeführt. Die einzelnen Instanzen werden nach der Ausführung des Elementes nicht synchronisiert.

Mehrfachinstanz ohne Synchronisation (engl. multiple instances without synchronization; wcp_{12})

BESCHREIBUNG: Erzeugung von mehreren Instanzen eines Elementes. Die Anzahl der zu erzeugenden Instanzen wird zur Entwurfszeit festgelegt. Der Kontrollfluss wird fortgesetzt, sobald die Instanzen erzeugt wurden. Die Instanzen können unabhängig voneinander ausgeführt werden. Nach der Ausführung werden diese nicht synchronisiert.

BEISPIEL: Es wird eine Liste mit Verkehrsverstößen erstellt. Für jeden Verkehrsverstoß wird die Aufgabe *Nachbesserungsaktivität* ausgeführt. Die Instanzen der Aufgabe Nachbesserungsaktivität können parallel und ohne nachträgliche Synchronisation ausgeführt werden.



NEWYAWL: Es werden vier zusätzliche Aufgaben: $t_{MiS} \in T$ *(MI-Split)*, $t_S \in T$ *(Thread-Split)*, $t_J \in T$ *(Thread-Join*) und $t_{MiJ} \in T$ *(MI-Join)* sowie die Split-, Join-, ThreadOut- und ThreadIn-Funktion

benötigt. Die Aufgabe MI-Split (t_{MiS}) spaltet den Kontrollfluss zunächst mit $\mathrm{Split}(t_{MiS}) = \mathrm{AND}$ in zwei Pfade auf. Ein Pfad repräsentiert den normalen Kontrollfluss und der andere Pfad erzeugt die Instanzen. Die Instanzen werden mit Threads durch die Aufgabe Thread-Split (t_S) erstellt. Dementsprechend muss für diese Aufgabe die Split- und ThreadOut-Funktion definiert werden, d. h. $\mathrm{Split}(t_S) = \mathrm{Thread}$ und $\mathrm{ThreadOut}(t_S) = n$. Mit n wird die Anzahl der zu erzeugenden Instanzen festgelegt, die der Anzahl der Listenelemente der Verkehrsverstöße entspricht, mit $n \in \mathbb{N}$. Somit kann n Mal die Aufgabe Nachbesserungsaktivität (t_2) ausgeführt werden. Die Threads werden dann in der Thread-Join-Aufgabe (t_J) wieder zusammengeführt. Für diese Aufgabe müssen die Join- und ThreadIn-Funktion definiert werden, d. h. $\mathrm{Join}(t_J) = \mathrm{THREAD}$ und $\mathrm{ThreadIn}(t_J) = n$. Da eine newYAWL-Spezifikation nur einen definierten Endknoten besitzen darf, wird der aufgespaltene Kontrollfluss in MI-Join (t_{MiJ}) wieder zusammengeführt. Für diese Aufgabe muss die Join-Funktion mit dem Funktionswert AND spezifiziert werden, d. h. $\mathrm{Join}(t_{MiJ}) = \mathrm{AND}$.

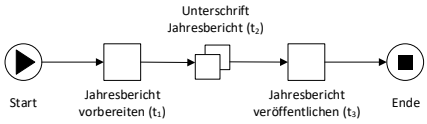
NEWYAWL-PARAMETER: Split, Join, ThreadOut und ThreadIn

Das Kontrollflussmuster *Mehrfachinstanz mit Festlegung der Anzahl der Instanzen zur Entwurfszeit* (wcp_{13}) bietet die Möglichkeit, eine vordefinierte Anzahl von Instanzen von einem Element zu erzeugen. Im Unterschied zum Kontrollflussmuster Mehrfachinstanz ohne Synchronisation (wcp_{12}) müssen erst alle Instanzen beendet sein, bevor die Ausführung des Elementes beendet ist und der Kontrollfluss fortgesetzt werden kann. Zusätzlich muss die Anzahl zu erzeugender Instanzen vor der Instanziierung der Geschäftsprozessinstanz bekannt sein [AHKB03, RHAM06, Russ07].

Mehrfachinstanz mit Festlegung der Anzahl der Instanzen zur Entwurfszeit (engl. multiple instances with a priori design-time knowledge; wcp_{13})

BESCHREIBUNG: Erzeugung von mehreren Instanzen eines Elementes. Die Anzahl der zu erzeugenden Instanzen muss vor der Instanziierung der Geschäftsprozessinstanz bekannt sein. Die Ausführung des Elementes ist beendet, wenn alle Instanzen beendet sind. Daraufhin kann der Kontrollfluss fortgesetzt werden. Die Instanzen werden unabhängig voneinander ausgeführt.

BEISPIEL: Der Jahresbericht muss von allen sechs Direktoren unterzeichnet werden, bevor dieser veröffentlicht werden kann.



NEWYAWL: Die Nofi-Funktion muss für die Aufgabe $t_2 \in T$ (*Unterschrift Jahresbericht)* definiert werden, d. h. $\mathrm{Nofi}(t_2) = (\min, \max, \mathrm{th}, \mathrm{ds}, \mathrm{canc})$ mit $\min = \max = \mathrm{th} =$ Anzahl der festgelegten Instanzen vor der Instanziierung der Geschäftsprozessinstanz; $\mathrm{ds} = \mathrm{static}$ und $\mathrm{canc} = \mathrm{non} - \mathrm{canceling}$.

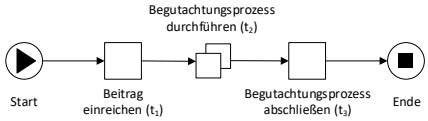
NEWYAWL-PARAMETER: Nofi

Beim Kontrollflussmuster *Mehrfachinstanz mit Festlegung der Anzahl der Instanzen vor der Laufzeit des Elementes* (wcp_{14}) muss die Anzahl der zu erzeugenden Instanzen erst unmittelbar vor der Ausführung des Elementes festgelegt werden, entgegen dem Kontrollflussmuster Mehrfachinstanz mit Festlegung der Anzahl der Instanzen zur Entwurfszeit (wcp_{13}) [AHKB03, RHAM06, Russ07].

Mehrfachinstanz mit Festlegung der Anzahl der Instanzen vor der Laufzeit des Elementes (engl. multiple instances with a priori run-time knowledge; wcp_{14})

BESCHREIBUNG: Erzeugung von mehreren Instanzen eines Elementes. Die Anzahl der zu erzeugenden Instanzen muss vor der Instanziierung des Elementes bekannt sein. Die Ausführung des Elementes ist beendet, wenn alle Instanzen beendet sind. Daraufhin kann der Kontrollfluss fortgesetzt werden. Die Instanzen werden unabhängig voneinander ausgeführt.

BEISPIEL: Der Begutachtungsprozess für einen wissenschaftlichen Beitrag bei einer Zeitschrift wird mehrfach ausgeführt. Die Ausführungshäufigkeit ist vom Inhalt der Arbeit und der Verfügbarkeit der Gutachter abhängig. Der Begutachtungsprozess kann erst fortgesetzt werden, wenn alle Gutachter ihre Bewertungen abgegeben haben.



NEWYAWL: Die Nofi-Funktion muss für die Aufgabe $t_2 \in T$ (*Begutachtungsprozess durchführen)* definiert werden, d. h. $\mathrm{Nofi}(t_2) = (\min, \max, \mathrm{th}, \mathrm{ds}, \mathrm{canc})$ mit $\min = \max = \mathrm{th} =$ Anzahl der festgelegten Instanzen vor der Instanziierung des Elementes; $\mathrm{ds} = \mathrm{static}$ und $\mathrm{canc} = \mathrm{non} - \mathrm{canceling}$.

NEWYAWL-PARAMETER: Nofi

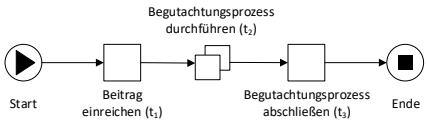
Das Kontrollflussmuster *Mehrfachinstanz mit Festlegung der Anzahl der Instanzen ohne Informationen vor der Laufzeit des Elementes* (wcp_{15}) ist eine Erweiterung des Kontrollflussmusters Mehrfachinstanz mit Festlegung der Anzahl der Instanzen vor der Laufzeit des Elementes (wcp_{14}). Dementsprechend erfolgt die Festlegung zu erzeugender Instanzen während der Laufzeit. Sofern weitere Instanzen benötigt werden, werden diese erzeugt.

Die Erzeugung weiterer Instanzen ist nur solange möglich, bis alle Instanzen des Elementes beendet sind [AHKB03, RHAM06, Russ07].

Mehrfachinstanz mit Festlegung der Anzahl der Instanzen ohne Informationen vor der Laufzeit des Elementes (engl. multiple instances without a priori run-time knowledge; wcp_{15})

BESCHREIBUNG: Erzeugung von mehreren Instanzen eines Elementes. Während der Ausführung des Elementes können beliebig viele Instanzen erzeugt werden, solange noch nicht alle Instanzen des Elementes beendet sind. Die Ausführung des Elementes ist beendet, wenn alle Instanzen beendet sind. Daraufhin kann der Kontrollfluss fortgesetzt werden. Die Instanzen werden unabhängig voneinander ausgeführt.

BEISPIEL: Der Begutachtungsprozess einer wissenschaftlichen Arbeit bei einer Zeitschrift wird mehrfach ausgeführt. Die Ausführungshäufigkeit ist vom Inhalt der Arbeit und der Verfügbarkeit der Gutachter abhängig. Während des Begutachtungsprozesses können sich jederzeit weitere Gutachter melden, die den Beitrag begutachten möchten. Der Prozess kann erst fortgesetzt werden, wenn alle Gutachter ihre Gutachten abgegeben haben.



NEWYAWL: Die Nofi-Funktion muss für die Aufgabe $t_2 \in T$ (*Begutachtungsprozess durchführen)* definiert werden, d. h. $\mathrm{Nofi}(t_2) = (\min, \max, \mathrm{th}, \mathrm{ds}, \mathrm{canc})$ mit $\mathrm{ds} = \mathrm{dynamic}$ und $\mathrm{canc} = \mathrm{non} - \mathrm{canceling}$. Die Werte $\min$, $\max$ und th werden während der Ausführung des Elementes festgelegt.

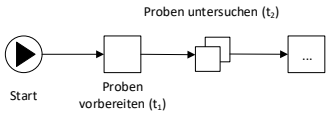
NEWYAWL-PARAMETER: Nofi

Das Kontrollflussmuster *statische partielle Zusammenführung für Mehrfachinstanzen* (wcp_{34}) ist eine Erweiterung des Kontrollflussmusters Mehrfachinstanz mit Festlegung der Anzahl der Instanzen vor der Laufzeit des Elementes (wcp_{14}). Hierbei müssen nicht alle Mehrfachinstanzen beendet sein, bevor der Kontrollfluss fortgesetzt werden kann. Stattdessen wird der Kontrollfluss nach einer festgelegten Anzahl beendeter Instanzen fortgesetzt. Alle erzeugten Instanzen des Elementes müssen dennoch vor der Beendigung der Geschäftsprozessinstanz beendet werden. Andernfalls kann die Geschäftsprozessinstanz nicht beendet werden [AHKB03, RHAM06, Russ07].

Statische partielle Zusammenführung für Mehrfachinstanzen (engl. static partial join for multiple instances; wcp$_{34}$)

BESCHREIBUNG: Erzeugung von mehreren Instanzen eines Elementes. Die Anzahl der zu erzeugenden Instanzen muss vor der Instanziierung des Elementes bekannt sein und es muss ein Schwellwert definiert werden. Der Schwellwert beschreibt die erforderliche Anzahl beendeter Instanzen, damit der Kontrollfluss fortgesetzt werden kann. Dennoch müssen alle erzeugten Instanzen beendet werden, bevor die Geschäftsprozessinstanz beendet werden kann. Die Instanzen werden unabhängig voneinander ausgeführt.

BEISPIEL: In einem Begutachtungsprozess werden immer 10 Proben auf Defekte untersucht. Sobald 7 der 10 Proben untersucht wurden, kann der Kontrollfluss fortgesetzt werden. Die übrigen 3 Proben müssen dennoch überprüft werden.



NEWYAWL: Die Nofi-Funktion muss für die Aufgabe $t_2 \in T$ (*Proben untersuchen)* definiert werden, d. h. $\mathrm{Nofi}(t_2) = (\min, \max, \mathrm{th}, \mathrm{ds}, \mathrm{canc})$ mit $\min = \max =$ Anzahl der festgelegten Instanzen vor der Instanziierung des Elementes; $\mathrm{th} =$ Schwellwert, der die Anzahl beendeter Instanzen beschreibt, die beendet sein müssen, bevor der Kontrollfluss fortgesetzt werden kann, d. h. $\mathrm{th} = 7$; $\mathrm{ds} = \mathrm{static}$ und $\mathrm{canc} = \mathrm{non} - \mathrm{canceling}$.

NEWYAWL-PARAMETER: Nofi

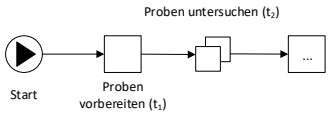
Das Kontrollflussmuster *abbrechende partielle Zusammenführung für Mehrfachinstanzen* (wcp$_{35}$) ist eine Erweiterung des Kontrollflussmusters Mehrfachinstanz mit Festlegung der Anzahl der Instanzen vor der Laufzeit des Elementes (wcp$_{14}$). Bei diesem Kontrollflussmuster müssen nicht alle Mehrfachinstanzen beendet sein, bevor der Kontrollfluss fortgesetzt werden kann. Stattdessen wird der Kontrollfluss nach einer festgelegten Anzahl beendeter Instanzen fortgesetzt. Die Anzahl wird durch einen Schwellwert definiert. Sobald der Schwellwert erreicht ist, wird der Kontrollfluss fortgesetzt. Im Unterschied zum Kontrollflussmuster statische partielle Zusammenführung für Mehrfachinstanzen (wcp$_{34}$) werden die noch nicht beendeten Instanzen abgebrochen [AHKB03, RHAM06, Russ07].

Abbrechende partielle Zusammenführung für Mehrfachinstanzen (engl. cancelling partial join for multiple instances; wcp$_{35}$)

BESCHREIBUNG: Erzeugung von mehreren Instanzen eines Elementes. Die Anzahl der zu erzeugenden Instanzen muss vor der Instanziierung des Elementes bekannt sein und es muss ein Schwellwert definiert werden. Der Schwellwert beschreibt die erforderliche Anzahl beendeter Instanzen,

damit der Kontrollfluss fortgesetzt werden kann. Nach der Erreichung des Schwellwertes werden alle noch nicht beendeten Instanzen abgebrochen. Die Instanzen werden unabhängig voneinander ausgeführt.

BEISPIEL: In einem Begutachtungsprozess werden immer 10 Proben auf Defekte untersucht. Sobald 7 der 10 Proben untersucht wurden, kann der Prozess fortgesetzt werden. Die übrigen 3, der noch nicht abgeschlossenen Überprüfungen, werden abgebrochen.



NEWYAWL: Die Nofi-Funktion muss für die Aufgabe $t_2 \in T$ (*Proben untersuchen)* definiert werden, d. h. $\mathrm{Nofi}(t_2) = (\min, \max, \mathrm{th}, \mathrm{ds}, \mathrm{canc})$ mit $\min = \max =$ Anzahl der festgelegten Instanzen vor der Instanziierung des Elementes; $\mathrm{th} =$ Schwellwert, der die Anzahl beendeter Instanzen beschreibt, die beendet sein müssen, bevor der Kontrollfluss fortgesetzt werden kann, d. h. $\mathrm{th} = 7$; $\mathrm{ds} = \mathrm{static}$ und $\mathrm{canc} = \mathrm{canceling}$.

NEWYAWL-PARAMETER: Nofi

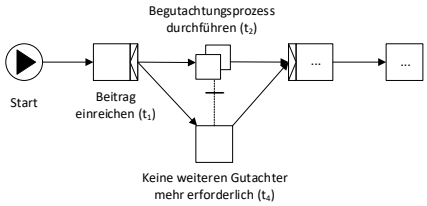
Das Kontrollflussmuster *dynamische partielle Zusammenführung für Mehrfachinstanzen* (wcp_{36}) ist eine Erweiterung des Kontrollflussmusters Mehrfachinstanz mit Festlegung der Anzahl ohne Informationen vor der Laufzeit des Elementes (wcp_{15}). Bei diesem Kontrollflussmuster müssen nicht alle Instanzen beendet sein, bevor der Kontrollfluss fortgesetzt werden kann. Während der Ausführung des Elementes können weitere Instanzen erzeugt werden. Es können solange beliebig viele Instanzen erzeugt werden, bis eine Deaktivierungsbedingung erfüllt ist, die die Erzeugung von weiteren Instanzen verhindert [AHKB03, RHAM06, Russ07].

Dynamische partielle Zusammenführung für Mehrfachinstanzen (engl. dynamic partial join for multiple instances; wcp_{36})

BESCHREIBUNG: Erzeugung von mehreren Instanzen eines Elementes. Während der Ausführung des Elementes können beliebig viele Instanzen erzeugt werden, bis die weitere Erzeugung von Instanzen deaktiviert wird. Der Kontrollfluss wird erst fortgesetzt, wenn die Beendigungsbedingung erfüllt ist. Daraufhin können auch keine weiteren Instanzen mehr erzeugt werden. Alle erzeugten Instanzen müssen beendet sein, bevor das Element erneut ausgeführt werden kann. Die Geschäftsprozessinstanz kann erst beendet werden, wenn alle Instanzen ausgeführt wurden. Die Instanzen werden unabhängig voneinander ausgeführt.

BEISPIEL: Der Begutachtungsprozess einer wissenschaftlichen Arbeit bei einer Zeitschrift wird mehrfach ausgeführt. Die Ausführungshäufigkeit ist vom Inhalt der Arbeit und der Verfügbarkeit

der Gutachter abhängig. Während des Begutachtungsprozesses können sich jederzeit weitere Gutachter melden, die den Beitrag begutachten möchten. Es können sich jedoch nur solange neue Gutachter melden, bis der Herausgeber der Zeitschrift bekundet hat, dass sich genügend Gutachter gemeldet haben.



NEWYAWL: Die Nofi- und Disable-Funktion muss definiert werden. Für die Aufgabe $t_2 \in T$ (*Begutachtungsprozess durchführen)* muss die Nofi-Funktion spezifiziert werden, d. h. $\mathrm{Nofi}(t_2) = (\min, \max, \mathrm{th}, \mathrm{ds}, \mathrm{canc})$ mit $\mathrm{ds} = \mathrm{dynamic}$ und $\mathrm{canc} = \mathrm{non} - \mathrm{canceling}$. Die Werte $\min$, $\max$ und th werden während der Ausführung des Elementes festgelegt. Die Disable-Funktion muss für die Aufgabe $t_4 \in T$ (*Keine weiteren Gutachter mehr erforderlich)* mit dem Funktionswert der Aufgabe $t_2 \in T$ (*Begutachtungsprozess durchführen)* spezifiziert werden, d. h. $\mathrm{Disable}(t_4) = t_2$.

NEWYAWL-PARAMETER: Nofi und Disable

4.1.2.4 Zustandsbasierte Muster

Die Kategorie der zustandsbasierten Kontrollflussmuster umfasst Muster, deren Ausführungsverhalten vom Zustand der Geschäftsprozessinstanz und teilweise auch von externen Faktoren abhängig ist. Die drei Kontrollflussmuster aufgeschobene Auswahl (engl. deferred choice; wcp_{16}), verschachtelte parallele Ausführung (engl. interleaved parallel routing; wcp_{17}) und Meilenstein (engl. milestone; wcp_{18}) zählen zu den ursprünglichen Mustern, die in [AHKB03] definiert wurden. Im Rahmen der Reorganisation der Kontrollflussmuster in [RHAM06] wurden vier weitere zustandsbasierte Kontrollflussmuster definiert. Das Kontrollflussmuster kritischer Bereich (engl. critical section; wcp_{39}) und die verschachtelte Ausführung (engl. interleaved routing; wcp_{40}) wurden der Kategorie der zustandsbasierten Kontrollflussmuster zugeordnet. Die zwei weiteren Kontrollflussmuster, d. h. Thread-Aufspaltung (wcp_{41}) und Thread-Zusammenführung (wcp_{42}) wurden den erweiterten Verzweigungs- und Synchronisationsmustern zugeordnet, da Threads erzeugt bzw. synchronisiert werden [AHKB03, RHAM06, Russ07].

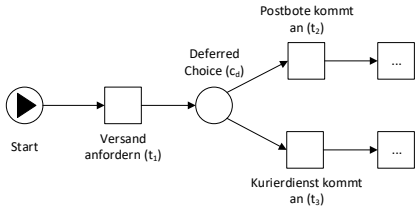
Mit dem Kontrollflussmuster *aufgeschobene Auswahl* (wcp_{16}) werden Entscheidungen modelliert, die nicht von den Informationen der Geschäftsprozessinstanz abhängen. Im Gegensatz zum Kontrollflussmuster exklusive Auswahl (wcp_4) ist die Entscheidung, welches Element im Nachbereich ausgeführt werden soll, von externen Faktoren abhängig, wie z. B.

eingehende Nachrichten, Umgebungsdaten oder Ressourcenverfügbarkeiten. Zum Zeitpunkt der Entscheidung ist nicht eindeutig bestimmbar, an welchem Element der Kontrollfluss fortgesetzt wird. Der Kontrollfluss kann aber nur an genau einem Element fortgesetzt werden [AHKB03, RHAM06, Russ07].

Aufgeschobene Auswahl (engl. deferred choice; wcp_{16})

BESCHREIBUNG: Verzweigung des Kontrollflusses an einem Element. Der Kontrollfluss wird abhängig von externen Faktoren an genau einem Element im Nachbereich fortgesetzt. Zum Zeitpunkt der Entscheidung ist nicht eindeutig bestimmbar, an welchem Element der Kontrollfluss fortgesetzt wird.

BEISPIEL: Ein Kunde fordert die Express-Versandlieferung von einem Airbag beim Paketdienst der Post sowie bei einem Kurierdienst an. Entweder wird das Paket vom Paketdienst der Post oder vom bestellten Kurierdienst ausgeliefert. Dies richtet sich danach, wer zuerst am Kundenstandort ist. Wenn beide Dienstleister gleichzeitig am Standort eintreffen, wird das Paket entweder vom Postboten oder vom Kurierdienst ausgeliefert.



NEWYAWL: Es muss eine Bedingung $c \in C$ hinzugefügt werden, die die aufgeschobene Auswahl modelliert und die dementsprechend mit allen Aufgaben aus dem Vor- und Nachbereich verbunden ist. In diesem Beispiel wurde die Bedingung $c_d \in C$ (*Deferred Choice*) eingefügt und mit den entsprechenden Aufgaben verknüpft, d. h. (t_1, c_d), (c_d, t_2), $(c_d, t_3) \in F$ mit $t_1 \in T$ *(Versand anfordern),* $t_2 \in T$ *(Postbote kommt an)* und $t_3 \in T$ *(Kurierdienst kommt an).*

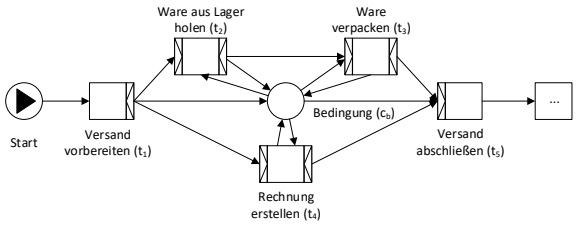
NEWYAWL-PARAMETER: keine

Das Kontrollflussmuster *verschachtelte parallele Ausführung* (wcp_{17}) ermöglicht die Modellierung von sogenannten kritischen Bereichen, sodass die Elemente dieses Bereiches nur nacheinander ausgeführt werden dürfen. Die Reihenfolge der auszuführenden Elemente des kritischen Bereichs wird durch eine partielle Ordnung bestimmt. Jedes Element muss dabei genau einmal ausgeführt werden. Dabei ist die partielle Ordnung von den kontrollflussspezifischen Anforderungen dieser Elemente abhängig. Dies liegt unter anderem vor, wenn zwei Elemente, wie z. B. die Elemente A und B, innerhalb eines Bereichs durch eine Sequenz (wcp_1) verbunden sind [AHKB03, RHAM06, Russ07].

Verschachtelte Parallele Ausführung (engl. interleaved parallel routing; wcp_{17})

BESCHREIBUNG: Es müssen alle Elemente eines definierten Bereichs ausgeführt werden. Diese Elemente unterliegen einer partiellen Ordnung, sodass die Ausführungsreihenfolge, abhängig von der partiellen Ordnung, variieren kann. Die partielle Ordnung wird durch kontrollflussspezifische Abhängigkeiten zu anderen Elementen bestimmt. Die einzelnen Elemente des Bereichs können nur nacheinander ausgeführt werden.

BEISPIEL: Vor der Versendung der Ware muss diese aus dem Lager geholt und verpackt werden. Zusätzlich muss eine Rechnung erstellt werden. Die Ware muss zuerst aus dem Lager geholt werden, bevor diese verpackt werden kann. Die Rechnung kann jederzeit erstellt werden. Es müssen alle Aufgaben ausgeführt werden, jedoch dürfen diese Aufgaben nicht gleichzeitig, sondern nur nacheinander ausgeführt werden.



NEWYAWL: Es muss eine Bedingung $c_b \in C$ hinzugefügt werden, die mit allen Aufgaben aus dem verschachtelten Bereich verbunden ist, d. h. (t_2, c_b), (c_b, t_2), (t_3, c_b), (c_b, t_3), (t_4, c_b), $(c_b, t_4) \in F$ mit $t_2 \in T$ *(Ware aus Lager holen),* $t_3 \in T$ *(Ware verpacken)* und $t_4 \in T$ *(Rechnung erstellen).* Zusätzlich müssen für diese Aufgaben die Split- und Join-Funktion definiert und mit dem Funktionswert AND spezifiziert werden, d. h. $\mathrm{Split}(t_2) = \mathrm{Join}(t_2) = \mathrm{AND}$, $\mathrm{Split}(t_3) = \mathrm{Join}(t_3) = \mathrm{AND}$ und $\mathrm{Split}(t_4) = \mathrm{Join}(t_4) = \mathrm{AND}$. Darüber hinaus muss für die entsprechende Anfangsaufgabe die Split-Funktion und für die Endaufgabe die Join-Funktion definiert sowie mit dem Funktionswert AND spezifiziert werden, d. h. $\mathrm{Split}(t_1) = \mathrm{Join}(t_5) = \mathrm{AND}$ mit $t_1 \in T$ *(Versand vorbereiten)* und $t_5 \in T$ *(Versand abschließen)*. Diese Aufgaben müssen ebenfalls mit der Bedingung $c_b \in C$ verknüpft werden, d. h. $(t_1, c_b), (c_b, t_5) \in F$.

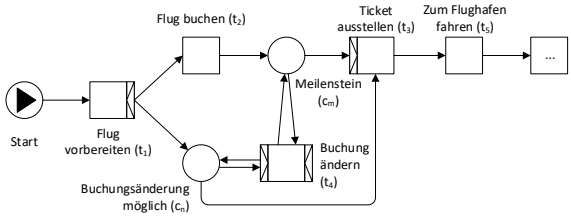
NEWYAWL-PARAMETER: Split und Join

Mit dem Kontrollflussmuster *Meilenstein* (wcp_{18}) können Szenarien modelliert werden, bei denen bestimmte Elemente nur dann ausgeführt werden können, wenn ein bestimmter Zustand in der Geschäftsprozessinstanz vorliegt. Sofern der bestimmte Zustand nicht mehr zutrifft, können die entsprechenden Elemente auch nicht mehr ausgeführt werden. Die Elemente können jederzeit wieder erneut ausgeführt werden, wenn der definierte Zustand wieder zutrifft [AHKB03, RHAM06, Russ07].

Meilenstein (engl. milestone; wcp$_{18}$)

BESCHREIBUNG: Ein oder mehrere Elemente können nur ausgeführt werden, wenn ein bestimmter Zustand vorliegt. Das Element bzw. die Elemente können nicht ausgeführt werden, wenn der bestimmte Zustand nicht zutrifft. Eine erneute Aktivierung des Elementes ist jederzeit möglich, wenn der bestimmte Zustand wieder eintritt.

BEISPIEL: Eine Fluggesellschaft ermöglicht es eine Buchung zu ändern, solange das Ticket noch nicht ausgestellt wurde.



NEWYAWL: Es muss eine Bedingung $c_m \in C$ hinzugefügt werden, die den Meilenstein darstellt. Die Aufgabe $t_4 \in T$ (*Buchung ändern)* kann nur ausgeführt werden, wenn der Meilenstein zutrifft. Für die entsprechenden Aufgaben, die nur mit dem Meilenstein ausgeführt werden können, müssen jeweils die Split- und Join-Funktion definiert werden und mit dem Funktionswert AND spezifiziert werden, d. h. $\mathrm{Split}(t_4) = \mathrm{Join}(t_4) = \mathrm{AND}$. Zusätzlich muss diese Aufgabe mit dem Meilenstein verknüpft werden, d. h. $(c_m, t_4), (t_4, c_m) \in F$. Die Aufgaben zur Aktivierung bzw. Deaktivierung des Meilensteins müssen ebenfalls mit der Bedingung verknüpft werden, d. h. $(t_2, c_m), (c_m, t_3) \in F$. Für die erneute Ausführung der Aufgabe $t_4 \in T$ muss eine Bedingung hinzugefügt werden, d. h. $c_n \in C$ und entsprechend eingebettet werden, d. h. $(t_4, c_n), (c_n, t_4), (t_1, c_n) \in F$ sowie mit der Aufgabe zur Deaktivierung des Meilensteins $t_3 \in T$, d. h. $(c_n, t_3) \in F$ verbunden werden.

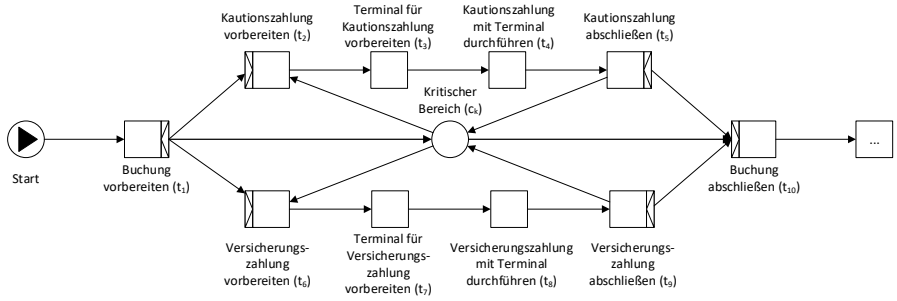
NEWYAWL-PARAMETER: Split und Join

Das Kontrollflussmuster *kritischer Bereich* (wcp$_{39}$) kann zur Modellierung von Bereichen verwendet werden, die nicht gleichzeitig ausgeführt werden dürfen. Beispielsweise kann dies der Fall sein, wenn Elemente auf die gleiche Ressource (z. B. Daten oder physische Ressource) zugreifen müssen [AHKB03, RHAM06, Russ07].

Kritischer Bereich (engl. critical section; wcp$_{39}$)

BESCHREIBUNG: Zwei oder mehrere Bereiche können als kritischer Bereich gekennzeichnet werden. Ein Bereich kann ein oder mehrere Elemente beinhalten. Sobald sich ein Element eines Bereichs in der Ausführung befindet, darf kein anderer Bereich Elemente besitzen, die aktiviert oder ausgeführt werden können. Erst nachdem kein Element mehr in dem Bereich ausgeführt wird, können die Elemente anderer Bereiche ausgeführt werden.

BEISPIEL: Für die Buchung eines Ferienhauses ist eine Kautions- und Versicherungszahlung erforderlich. Die Bezahlung ist nur mit einer Kreditkarte möglich. Die Bezahlmöglichkeit kann aus diesem Grund nur über ein Kreditkartenterminal erfolgen. Die jeweilige Kautions- und Versicherungszahlung kann nur nacheinander erfolgen.



NEWYAWL: Es muss eine Bedingung $c \in C$ hinzugefügt werden, die jeweils mit den Anfangs- und Endaufgaben der kritischen Bereiche verbunden ist. Im obigen Beispiel wurde die Bedingung $c_k \in C$ eingefügt, die mit den Anfangs- und Endaufgaben verbunden ist, d. h. $(c_k, t_2), (t_5, c_k), (c_k, t_6), (t_9, c_k) \in F$ mit $t_2 \in T$ *(Kreditkartenzahlung vorbereiten),* $t_5 \in T$ *(Kautionszahlung abschließen)*, $t_6 \in T$ *(Versicherungszahlung vorbereiten)* und $t_9 \in T$ *(Versicherungszahlung abschließen).* Für die Anfangsaufgaben muss jeweils die Split-Funktion sowie für die Endaufgabe die Join-Funktion definiert und mit dem Funktionswert AND spezifiziert werden, d. h. $\mathrm{Split}(t_2) = \mathrm{Join}(t_5) = \mathrm{AND}$ und $\mathrm{Split}(t_6) = \mathrm{Join}(t_9) = \mathrm{AND}$.

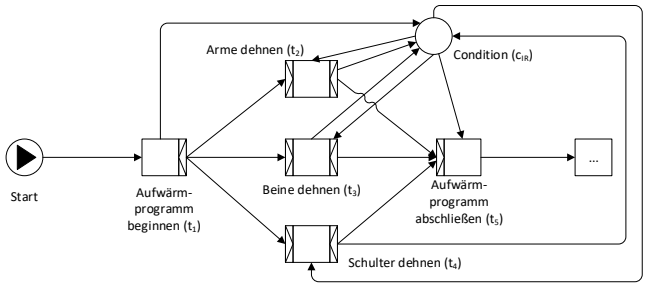
NEWYAWL-PARAMETER: Split und Join

Das Kontrollflussmuster *verschachtelte Aufspaltung* (wcp_{40}) ist ein weniger restriktives Kontrollflussmuster als die verschachtelte parallele Aufspaltung (wcp_{17}), da keine partielle Ordnung mehr existieren darf. Dementsprechend können die Elemente in beliebiger Reihenfolge ausgeführt werden [AHKB03, RHAM06, Russ07].

Verschachtelte Ausführung (engl. interleaved routing; wcp_{40})

BESCHREIBUNG: Alle Elemente eines definierten Bereichs müssen ausgeführt werden. Die Elemente dieses Bereichs können in beliebiger Reihenfolge nacheinander ausgeführt werden. Zwischen den Elementen des Bereichs bestehen keine kontrollflussspezifischen Abhängigkeiten.

BEISPIEL: Das Aufwärmprogramm eines Sportlers besteht aus dem Dehnen der Arme, dem Dehnen der Beine und dem Dehnen der Schulter. Die Reihenfolge der Übungen ist dabei irrelevant. Eine nebenläufige Ausführung ist nicht möglich.



NEWYAWL: Es muss eine Bedingung $c \in C$ hinzugefügt werden, die mit allen Aufgaben aus dem verschachtelten Bereich verbunden wird. Zusätzlich ist die Bedingung mit dem Beginn und dem Ende des verschachtelten Bereichs verbunden. Im obigen Beispiel wurde die Bedingung $c_{IR} \in C$ *(Condition)* eingefügt und mit den entsprechenden Aufgaben verbunden, d. h. (t_1, c_{IR}), (t_2, c_{IR}), (c_{IR}, t_2), (t_3, c_{IR}), (c_{IR}, t_3), (t_4, c_{IR}), (c_{IR}, t_4) $(c_{IR}, t_5) \in F$ mit $t_1 \in T$ *(Aufwärmprogramm beginnen),* $t_2 \in T$ *(Arme dehnen),* $t_3 \in T$ *(Beine dehnen),* $t_4 \in T$ *(Schulter dehnen)* und $t_5 \in T$ *(Aufwärmprogramm abschließen)*. Darüber hinaus müssen für diese Aufgaben die Split- und Join-Funktion definiert und mit dem Funktionswert AND spezifiziert werden, d. h. $\mathrm{Split}(t_2) = \mathrm{Join}(t_2) = \mathrm{AND}$, $\mathrm{Split}(t_3) = \mathrm{Join}(t_3) = \mathrm{AND}$ und $\mathrm{Split}(t_4) = \mathrm{Join}(t_4) = \mathrm{AND}$. Für die Anfangsaufgabe des verschachtelten Bereichs wird nur die Split-Funktion und für die Endaufgabe die Join-Funktion, d. h. $\mathrm{Split}(t_1) = \mathrm{Join}(t_5) = \mathrm{AND}$ benötigt.

NEWYAWL-PARAMETER: Split und Join

4.1.2.5 Abbruch und erzwungene Beendigungsmuster

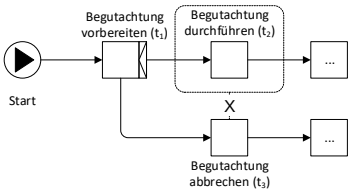
Die Kontrollflussmuster aus der Kategorie Abbruch und erzwungene Beendigung sind Muster, die einen Abbruch oder ein erzwungenes Ende von einzelnen Elementen oder Bereichen ermöglichen. In der ursprünglichen Definition der Kontrollflussmuster [AHKB03] wurden die zwei Kontrollflussmuster Element abbrechen (engl. cancel task; wcp_{19}) und Geschäftsprozessinstanz abbrechen (engl. cancel case; wcp_{20}) definiert. Im Rahmen der Reorganisation [RHAM06] wurden drei weitere derartige Kontrollflussmuster definiert: Bereich abbrechen (engl. cancel region; wcp_{25}), Mehrfachinstanz Element abbrechen (engl. cancel multiple instance activity; wcp_{26}) und Mehrfachinstanz Element beenden (engl. complete multiple instance activity; wcp_{27}) [AHKB03, RHAM06, Russ07].

Mit dem Kontrollflussmuster *Element abbrechen* (wcp_{19}) können einzelne Elemente deaktiviert werden, die sich in Ausführung befinden. Mit der Deaktivierung kann sichergestellt werden, dass die Ausführung des Elementes nicht begonnen werden kann. Sofern sich das Element bereits in der Ausführung befindet, wird die Ausführung abgebrochen [AHKB03, RHAM06, Russ07].

Element abbrechen (engl. cancel task; wcp_{19})

BESCHREIBUNG: Ein aktiviertes Element wird deaktiviert, sodass dieses nicht mehr ausgeführt werden kann oder ein Element, dass sich in der Ausführung befindet, wird abgebrochen.

BEISPIEL: Ein Bauabnehmer kann zu jedem Zeitpunkt die Begutachtung eines Gebäudes abbrechen, solange diese noch nicht beendet ist.



NEWYAWL: Es muss die Rem-Funktion für die Aufgabe $t_3 \in T$ *(Begutachtung abbrechen)* definiert werden, die $t_2 \in T$ *(Begutachtung durchführen)* als Funktionswert besitzt und somit die Aufgabe t_2 abbricht, d. h. $\mathrm{Rem}(t_3) = t_2$.

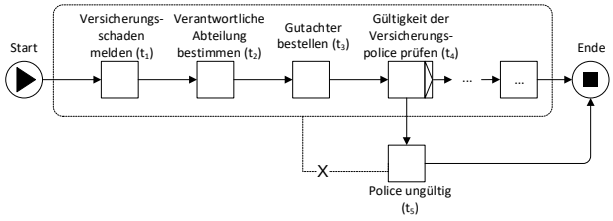
NEWYAWL-PARAMETER: Rem

Das Kontrollflussmuster *Geschäftsprozessinstanz abbrechen* (wcp_{20}) bietet nicht nur die Möglichkeit einzelne Elemente abzubrechen oder zu deaktivieren, sondern ermöglicht den Abbruch der ganzen Geschäftsprozessinstanz. Dies ist eine Erweiterung gegenüber dem Kontrollflussmuster Element abbrechen (wcp_{19}) [AHKB03, RHAM06, Russ07].

Geschäftsprozessinstanz abbrechen (engl. cancel case; wcp_{20})

BESCHREIBUNG: Eine Geschäftsprozessinstanz, die sich in der Ausführung befindet, wird abgebrochen. Die Geschäftsprozessinstanz wird als nicht erfolgreiche Ausführung abgespeichert.

BEISPIEL: Bei einem Versicherungsanspruch wird festgestellt, dass die Police abgelaufen ist. Infolgedessen werden alle Aufgaben abgebrochen, die mit der Geschäftsprozessinstanz in Verbindung stehen.



NEWYAWL: Es muss die Rem-Funktion für die Aufgabe $t_5 \in T$ *(Police ungültig)* definiert werden, die als Funktionswert alle Elemente des Geschäftsprozessmodells enthält, d. h. $\mathrm{Rem}(t_5) = \{t_1, t_2, t_3, t_4, t_6, \ldots\}$.

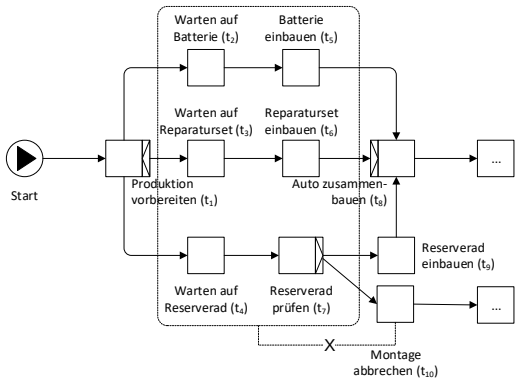
NEWYAWL-PARAMETER: Rem

Das Kontrollflussmuster *Bereich abbrechen* (wcp_{25}) ist eine Kombination der beiden Kontrollflussmuster Element abbrechen (wcp_{19}) und Geschäftsprozessinstanz abbrechen (wcp_{20}). Dementsprechend kann nicht nur ein einzelnes Element oder eine ganze Geschäftsprozessinstanz, sondern ein speziell definierter Bereich innerhalb des Geschäftsprozessmodells abgebrochen werden [AHKB03, RHAM06, Russ07].

Bereich abbrechen (engl. cancel region; wcp_{25})

BESCHREIBUNG: Ein definierter Bereich, der beliebig viele Elemente beinhalten kann, wird abgebrochen. Dementsprechend werden die aktivierten Elemente in diesem Bereich deaktiviert, so dass diese Elemente nicht mehr ausgeführt werden können. Elemente, die sich noch in der Ausführung befinden werden abgebrochen. Die Elemente des Bereichs müssen keine zusammenhängende Teilmenge bilden.

BEISPIEL: Für die Produktion eines Fahrzeuges wird eine *Batterie*, ein *Reserverad* und/oder ein *Reparaturset* benötigt. Wenn das falsche Reserverad geliefert wird, wird die gesamte Montage abgebrochen.



NEWYAWL: Es muss die Rem-Funktion für die Aufgabe $t_{10} \in T$ *(Montage abbrechen)* definiert werden, die als Funktionswert alle zu deaktivierenden bzw. abzubrechenden Elemente enthält, d. h. $\mathrm{Rem}(t_{10}) = \{t_2, t_3, t_4, t_5, t_6, t_7\}$.

NEWYAWL-PARAMETER: Rem

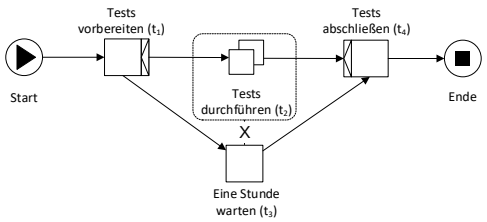
Mit dem Kontrollflussmuster *Mehrfachinstanzelement abbrechen* (wcp_{26}) kann eine Mehrfachinstanz abgebrochen werden. Dadurch können die noch nicht ausgeführten Instanzen

auch nicht mehr ausgeführt werden und Instanzen, die sich in der Ausführung befinden, werden abgebrochen. Die bereits beendeten Instanzen bleiben von dem Abbruch unberührt [AHKB03, RHAM06, Russ07].

Mehrfachinstanzelement abbrechen (engl. cancel multiple instance task; wcp_{26})

BESCHREIBUNG: Von einem Element können mehrere Instanzen existieren, die voneinander unabhängig ausgeführt werden. Die Anzahl zu erzeugender Instanzen ist zur Entwurfszeit bekannt. Beim Abbrechen eines Mehrfachinstanzelementes können noch auszuführende Instanzen nicht mehr ausgeführt werden und sich in der Ausführung befindende Instanzen werden abgebrochen. Solange noch nicht alle Instanzen beendet sind ist eine Deaktivierung bzw. ein Abbruch der Instanzen möglich. Bereits beendete Instanzen bleiben davon unberührt.

BEISPIEL: Es werden 500 Eiweißtests durchgeführt, bis entweder alle Tests beendet sind oder die Zeitspanne von einer Stunde vergangen ist. Für den Fall, dass die Tests nach einer Stunde nicht beendet sind, wird die Mehrfachinstanzaufgabe abgebrochen, d. h. die noch nicht abgeschlossenen Tests werden abgebrochen.



NEWYAWL: Es muss die Rem-Funktion für die Aufgabe $t_3 \in T$ *(Eine Stunde warten)* definiert werden, die die Mehrfachinstanzaufgabe $t_2 \in T$ *(Tests durchführen)* dann abbricht, d. h. $\mathrm{Rem}(t_3) = t_2$.

NEWYAWL-PARAMETER: Rem

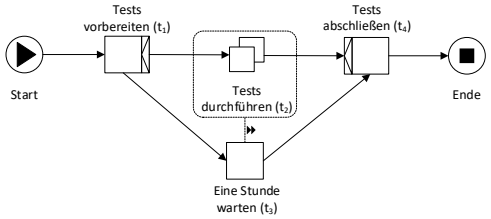
Das Kontrollflussmuster *Mehrfachinstanzelement beenden* (wcp_{27}) kann zur Beendigung einer Mehrfachinstanz verwendet werden, so dass die noch nicht beendeten Instanzen nicht mehr ausgeführt werden können. Die Anzahl zu erzeugender Instanzen ist zur Entwurfszeit bekannt. Dementsprechend werden die noch auszuführenden Instanzen des Elementes nicht mehr ausgeführt. Die bereits vollständig ausgeführten Instanzen bleiben von der Beendigung unberührt [AHKB03, RHAM06, Russ07].

Mehrfachinstanzelement beenden (engl. complete multiple instance activity; wcp_{27})

BESCHREIBUNG: Von einem Element können mehrere Instanzen existieren, die beendet werden sollen. Die Instanzen sind voneinander unabhängig. Wenn ein Mehrfachinstanzelement beendet

werden soll, dann können nachfolgend alle noch nicht ausgeführten Instanzen auch nicht mehr ausgeführt werden.

BEISPIEL: Es werden unterschiedliche Eiweißtests durchgeführt, bis entweder alle Tests beendet sind oder die Zeitspanne von einer Stunde vergangen ist. Falls nach einer Stunde nicht alle Tests durchgeführt werden konnten, soll auf die Beendigung der in der Ausführung befindenden Tests gewartet werden. Zudem sollen keine neuen Tests mehr ausgeführt werden können.



NEWYAWL: Es muss die Comp-Funktion für die Aufgabe $t_3 \in T$ (*Eine Stunde warten*) definiert werden, die die Mehrfachinstanzaufgabe $t_2 \in T$ (*Tests durchführen*) beendet, d. h. $\mathrm{Comp}(t_3) = t_2$.

NEWYAWL-PARAMETER: Comp

4.1.2.6 Iterationsmuster

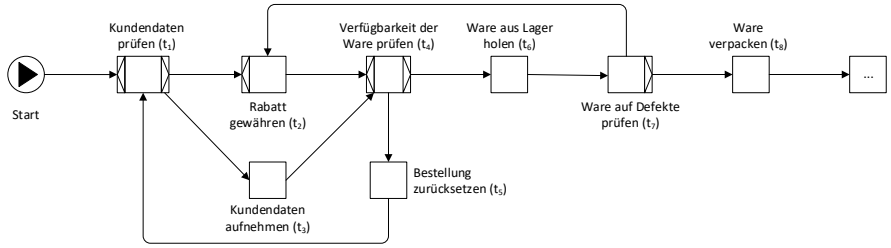
Die Iterationsmuster werden zur Beschreibung der wiederholten Ausführung von Elementen verwendet. Es wird zwischen beliebigen Zyklen (engl. arbitrary cycles; wcp_{10}), strukturierten Schleifen (engl. structured loop; wcp_{21}) und der Rekursion (engl. recursion; wcp_{22}) unterschieden. In Analogie zu den klassischen Programmiersprachen können die beliebigen Zyklen mit den goto-Anweisungen verglichen werden, die ein unstrukturiertes Programmieren ermöglichen. Die Ausführung des Programmcodes kann mit einer goto-Anweisung an einer bestimmten Stelle fortgesetzt werden. Eine strukturiertere Variante der Wiederholung kann durch den Einsatz von Schlüsselwörtern, wie `while … do` und `repeat … until` erreicht werden, die mit dem Kontrollflussmuster strukturierte Schleife modelliert werden können. Der Selbstaufruf einer Methode entspricht dem Kontrollflussmuster Rekursion [AHKB03, RHAM06, Russ07].

Das Kontrollflussmuster *beliebige Zyklen* (wcp_{10}) kann zur unstrukturierten Modellierung von Schleifen in Geschäftsprozessmodellen verwendet werden, ohne spezifische Schleifenbedingungen oder Anforderungen an das gesamte Geschäftsprozessmodell oder einzelne Elemente formulieren zu müssen [AHKB03, RHAM06, Russ07].

Beliebige Zyklen (engl. arbitrary cycles; wcp_{10})

BESCHREIBUNG: Es handelt sich um einen definierten Bereich, der mehrere Ein- und Ausstiegspunkte besitzt. Ein Punkt ist ein Element und der definierte Bereich besteht aus mehreren Elementen.

BEISPIEL: Für die Bestellabwicklung werden zunächst die Kundendaten geprüft. Sofern der Kunde bereits bekannt ist, wird diesem ein Rabatt gewährt. Bei Neukunden müssen die Kundendaten aufgenommen werden. Daran anknüpfend wird im Lager überprüft, ob die Ware verfügbar ist. Ist die Ware nicht verfügbar, so wird die Bestellung zurückgesetzt und der Bestellvorgang beginnt wieder mit der Überprüfung der Kundendaten. Ist die Ware verfügbar, so wird diese aus dem Lager geholt. Nachdem die Ware vorliegt, wird diese auf Defekte hin überprüft. Sofern keine Defekte vorliegen, wird die Ware verpackt. Andernfalls erhält der Kunde erneut einen Rabatt und die Verfügbarkeit der Ware wird erneut überprüft.



NEWYAWL: ES müssen keine Parameter in der newYAWL-Spezifikation für die Beschreibung von beliebigen Zyklen definiert werden, da diese aus der Ablaufreihenfolge der Elemente im Geschäftsprozessmodell hervorgehen.

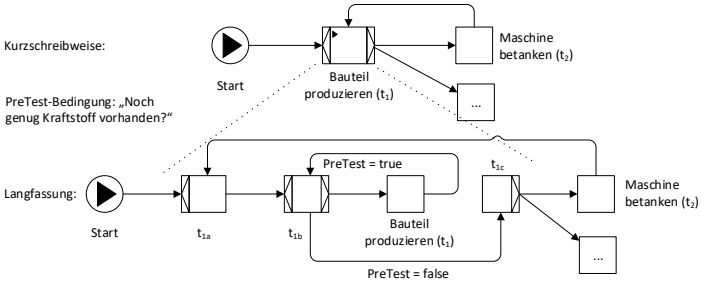
NEWYAWL-PARAMETER: keine

Für das Kontrollflussmuster s*trukturierte Schleife* (wcp_{21}) existiert ein definierter Bereich im Geschäftsprozessmodell, der in Abhängigkeit von einer Bedingung wiederholt ausgeführt werden kann. Die Bedingung kann zum einen vor der Ausführung (Pre-Test) oder erst nach der Ausführung (Post-Test) des Bereichs überprüft werden. Die Überprüfung der Bedingung vor der Ausführung des Bereichs entspricht einer klassischen `while … do`-Schleife, so dass abhängig von der Bedingung, der Bereich kein einziges Mal oder auch mehrmals ausgeführt werden kann. Der Bereich wird solange ausgeführt, bis die Bedingung als falsch ausgewertet wird. Daraufhin wird der Bereich nicht erneut ausgeführt und der Kontrollfluss wird nach diesem Bereich fortgesetzt. Im Unterschied zur Überprüfung der Bedingung nach einem Durchlauf des Bereichs erfordert diese Variante mindestens einen Durchlauf und entspricht einer `repeat … until`-Schleife. Der Bereich wird solange wiederholt ausgeführt, bis die Bedingung als wahr ausgewertet wird. Andernfalls wird der Kontrollfluss nach dem Bereich fortgesetzt [AHKB03, RHAM06, Russ07].

Strukturierte Schleife (engl. structured loop; wcp_{21})

BESCHREIBUNG: Ein Bereich im Geschäftsprozessmodell soll in Abhängigkeit von einer Bedingung wiederholt ausgeführt werden. Die Bedingung kann zum einen vor der Ausführung (Pre-Test) oder erst nach der Ausführung (Post-Test) des Bereichs überprüft werden. Beim Pre-Test wird der Bereich solange wiederholt ausgeführt, bis die referenzierte Bedingung als falsch ausgewertet wird. Im Gegensatz zum Post-Test wird der Bereich solange ausgeführt, bis die Bedingung erfüllt ist. Eine Kombination von Pre- und Post-Test ist ebenfalls möglich. Sofern der Bereich aufgrund der Bedingung nicht mehr ausgeführt werden kann, wird der Kontrollfluss nach dem Bereich fortgesetzt.

BEISPIEL: Solange in der Maschine noch Kraftstoff ist, kann ein Bauteil nach dem anderen produziert werden. Die Kraftstoffmenge wird immer vor der Produktion eines Bauteils überprüft.



NEWYAWL: Für dieses Beispiel muss die PreTest-Funktion für die Aufgabe $t_1 \in T$ *(Bauteil produzieren)* definiert werden, die als Funktionswert die Bedingung: „Noch genug Kraftstoff vorhanden?“ enthält, d. h. $\mathrm{PreTest}(t_1) = \text{"Noch genug Kraftstoff vorhanden?"}$. Analog könnte die PostTest-Funktion definiert werden, sofern die Bedingung nach der Ausführung überprüft werden soll. Falls ein Pre- und Post-Test erforderlich ist, muss jeweils die PreTest- und PostTest-Funktion definiert werden.

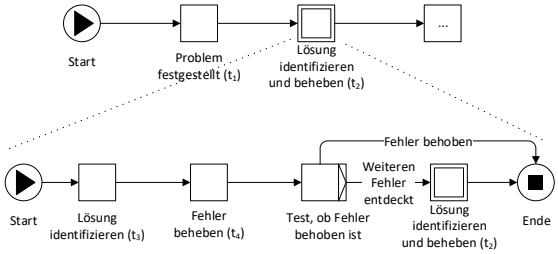
NEWYAWL-PARAMETER: PreTest und/oder PostTest

Das Kontrollflussmuster *Rekursion* (wcp_{22}) ermöglicht Bereiche wiederkehrend auszuführen, d. h., dass nicht explizit vor oder nach dem Bereich überprüft wird, ob dieser erneut ausgeführt werden soll. Stattdessen wird der Bereich erneut von sich selbst aufgerufen, so dass Hierarchieebenen zwischen den Bereichen entstehen. Es wird immer zuerst der erneut aufgerufene Bereich (niedrigere Hierarchieebene) ausgeführt, bevor eine höhere Hierarchieebene ausgeführt werden kann [AHKB03, RHAM06, Russ07].

Rekursion (engl. recursion; wcp_{22})

BESCHREIBUNG: Ein Bereich im Geschäftsprozessmodell soll wiederholt ausgeführt werden. Der Bereich besteht aus einem oder mehreren Elementen. Zur erneuten Ausführung des Bereichs wird dieser von sich selbst wieder aufgerufen.

BEISPIEL: Für jedes mechanische Problem, das in einer Produktionsanlage identifiziert wird, wird eine Aufgabe *Lösung identifizieren und beheben* initiiert. Bei der Ausführung der Aufgabe können weitere Fehler identifiziert werden, so dass eine weitere Aufgabe *Lösung identifizieren und beheben* initiiert wird. Der Hauptdefekt kann nicht gelöst werden, sofern die Unterdefekte nicht gelöst wurden.



NEWYAWL: ES müssen keine spezifischen Variablen in der newYAWL-Spezifikation definiert werden, da jede Aufgabe in einem newYAWL-Modell als zusammengesetzte Aufgabe modelliert werden kann. Innerhalb dieser zusammengesetzten Aufgabe kann wiederum auf die zusammengesetzte Aufgabe verwiesen werden.

NEWYAWL-PARAMETER: keine

4.1.2.7 Beendigungsmuster

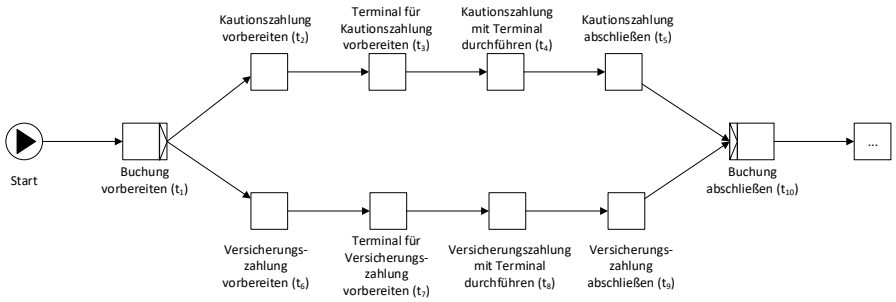
Eine weitere Kategorie bei den Kontrollflussmustern der Workflow Pattern Initiative sind die Beendigungsmuster, die Mechanismen beschreiben, wie eine Geschäftsprozessinstanz beendet werden kann. Es wird zwischen der impliziten Beendigung (engl. implicit termination; wcp_{11}) und der expliziten Beendigung (engl. explicit termination; wcp_{43}) differenziert, auf die nachfolgend eingegangen wird [AHKB03, RHAM06, Russ07].

Das Kontrollflussmuster *implizite Beendigung* (wcp_{11}) beschreibt die Beendigung einer Geschäftsprozessinstanz. Eine Geschäftsprozessinstanz gilt als erfolgreich beendet, wenn es kein Element mehr gibt, das noch ausgeführt werden kann bzw. sich in der Ausführung befindet. Darüber hinaus kann mit der impliziten Beendigung ein Ende eines Pfades beschrieben werden. Ein Pfad ist beendet, wenn kein nachfolgendes Element aktiviert werden kann. Zu diesem Kontrollflussmuster gehören keine ungewollten Zustände der Geschäftsprozessinstanz, wie z. B. ein Deadlock [AHKB03, RHAM06, Russ07].

Implizite Beendigung (engl. implicit termination; wcp$_{11}$)

BESCHREIBUNG: Es kann eine Geschäftsprozessinstanz oder auch ein Pfad implizit beendet werden. Eine Geschäftsprozessinstanz ist implizit beendet, wenn es keine Elemente mehr gibt, die noch ausgeführt werden können bzw. sich in der Ausführung befinden. Ein Pfad ist beendet, wenn kein nachfolgendes Element aktiviert werden kann. Zu diesem Kontrollflussmuster gehören keine ungewollten Zustände der Geschäftsprozessinstanz, wie z. B. ein Deadlock.

BEISPIEL: Dieses Kontrollflussmuster kann mit der Modellierungssprache YAWL bzw. newYAWL nicht modelliert werden, ohne die Syntaxanforderungen zu verletzen. Ansonsten wäre die Bedingung verletzt, dass jeder Knoten im Netz $(C \cup T, F)$ auf einem direkten Pfad von i nach o liegt. Dementsprechend wird ein Negativbeispiel aufgezeigt, das eine ungewollte Beendigung modelliert. In dem Beispiel wird der Kontrollfluss durch die Aufgabe $t_1 \in T$ *(Buchung vorbereiten)* aufgespalten, die mit der Split-Funktion und dem Funktionswert XOR spezifiziert ist, d. h. $\mathrm{Split}(t_1) = \mathrm{XOR}$. Dieser aufgespaltene Kontrollfluss soll wieder durch die Aufgabe $t_{10} \in T$ *(Buchung abschließen)* zusammengeführt werden, die mit der Join-Funktion und dem Funktionswert AND spezifiziert ist, d. h. $\mathrm{Join}(t_{10}) = \mathrm{AND}$. Die Aufgabe $t_{10} \in T$ *(Buchung abschließen)* kann jedoch nicht aktiviert werden, da entweder der Bereich der Kautionszahlung oder der Bereich der Versicherungszahlung ausgeführt wird. Es können niemals beide Pfade ausgeführt werden, die für die Aktivierung der Aufgabe *Buchung abschließen* erforderlich sind.



NEWYAWL: Das Kontrollflussmuster kann mit der Modellierungssprache YAWL bzw. newYAWL nicht modelliert werden.

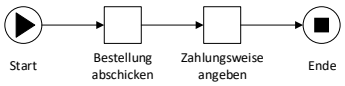
NEWYAWL-PARAMETER: keine

Das Pendant ist das Kontrollflussmuster *explizite Terminierung* (wcp$_{43}$) mit dessen Hilfe die Beendigung einer Geschäftsprozessinstanz erwirkt werden kann, wenn ein bestimmter Punkt im Geschäftsprozessmodell erreicht wird. Typischerweise befindet sich am Ende eines Geschäftsprozessmodells ein Terminierungsknoten, der die Geschäftsprozessinstanz terminiert [AHKB03, RHAM06, Russ07].

Explizite Beendigung (engl. explicit termination; wcp_{43})

BESCHREIBUNG: Eine Geschäftsprozessinstanz wird beendet, wenn ein bestimmter Punkt im Geschäftsprozessmodell erreicht wird. Jedes noch aktive Element wird daraufhin deaktiviert und Elemente, die ausgeführt werden, werden abgebrochen. Die Geschäftsprozessinstanz gilt als erfolgreich beendet.

BEISPIEL: Eine Geschäftsprozessinstanz startet mit der Aufgabe *Bestellung abschicken,* auf die die Aufgabe *Zahlungsweise angeben* folgt. Daraufhin ist die Geschäftsprozessinstanz beendet.



NEWYAWL: Es muss das Element $o \in C$ *(Ende)* verwendet werden. Sobald der Kontrollfluss das Element erreicht, ist die Geschäftsprozessinstanz beendet.

NEWYAWL-PARAMETER: keine

4.1.2.8 Trigger Patterns

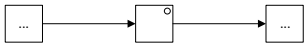
Die Kategorie Trigger Muster der Workflow Pattern Initiative beschreibt Muster, für die externe Einflüsse erforderlich sind, damit ein Element ausgeführt werden kann. Es wird zwischen den Kontrollflussmustern transienter Trigger (engl. transient trigger; wcp_{23}) und persistenter Trigger (engl. transient trigger; wcp_{24}) differenziert, die nachfolgend betrachtet werden [AHKB03, RHAM06, Russ07].

Das Kontrollflussmuster *transienter Trigger* (wcp_{23}) beschreibt das Eintreten eines Ereignisses, sodass durch das Eintreten dieses Ereignisses ein Element aktiviert wird, wenn alle anderen erforderlichen Vorbedingungen für die Aktivierung erfüllt sind. Die transienten Ereignisse haben keine Erinnerungsfunktion, d. h. falls die Ereignisse nicht sofort nach dem Eintreten behandelt werden bzw. das entsprechende Element ausgeführt wird, kann es sein, dass das Ereignis nicht mehr verfügbar ist [AHKB03, RHAM06, Russ07].

Transienter Trigger (engl. transient trigger; wcp_{23})

BESCHREIBUNG: Ein Element benötigt zusätzlich zu anderen kontrollflussspezifischen Anforderungen ein Ereignis, um aktiviert zu werden. Das benötigte Ereignis ist nur vorübergehender Natur und kann nur verwendet werden, wenn das Element zum Zeitpunkt des Empfangs darauf wartet.

BEISPIEL: Wenn ein Alarmsignal (Trigger) empfangen wird, wird die Prüfung des Sensors (Aufgabe) initialisiert.



Sensor prüfen (t_1)

NEWYAWL: Es muss die Trig-Funktion für die Aufgabe $t_1 \in T$ *(Sensor prüfen)* definiert werden. Die Trig-Funktion muss mit dem Funktionswert der TriggerID spezifiziert werden, d. h. $\mathrm{Trig}(t_1) = \mathrm{Alarmsignal}$.

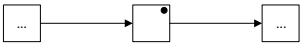
NEWYAWL-PARAMETER: Trig

Mit dem Kontrollflussmuster *persistenter Trigger* (wcp_{24}) können im Vergleich zum transienten Trigger Ereignisse modelliert werden, die von dauerhafter Natur sind. Dementsprechend wird das Ereignis einmal ausgelöst und ist solange aktiv, bis dieses von dem benötigten Element verwendet wird [AHKB03, RHAM06, Russ07].

Persistenter Trigger (engl. persistent trigger; wcp_{24})

BESCHREIBUNG: Ein Element benötigt zusätzlich zu anderen kontrollflussspezifischen Anforderungen ein Ereignis um aktiviert zu werden. Das benötigte Ereignis ist von dauerhafter Natur und ist solange aktiv, bis dieses von dem benötigten Element verwendet wird.

BEISPIEL: Bei jeder neuen Einstellung eines Mitarbeiters wird die Vorstellungsaufgabe (jetzt oder später) initialisiert.



Vorstellung des neuen Mitarbeiters (t_1)

NEWYAWL: Es muss die Trig-Funktion für die Aufgabe $t_1 \in T$ *(Vorstellung des neuen Mitarbeiters)* definiert und mit dem Funktionswert der TriggerID spezifiziert werden, d. h. $\mathrm{Trig}(t_1) = \mathrm{Neuer\ Mitarbeiter}$. Zusätzlich muss die Trigger-Aufgabe zur Menge $\mathrm{Persist}$ hinzugefügt werden, d. h. $\mathrm{Persist} = \{t_1\}$.

NEWYAWL-PARAMETER: Trig und Persist

4.2 Präzise Beschreibung der Kontrollflusssemantik

Zur Vermeidung von ungewollten Mehrdeutigkeiten der Kontrollflusssemantik im Geschäftsprozessmodell, muss die Ablaufreihenfolge der Elemente im Geschäftsprozessmodell präzise beschrieben sein. Für die Beschreibung der Kontrollflusssemantik einer Geschäftsprozessmodellierungssprache, eines Geschäftsprozessmodells oder auch eines Kontrollflussmusters kann eine operationale, denotationelle oder auch axiomatische Semantik definiert werden. Zusätzlich gibt es die Übersetzungssemantik, die das zu

spezifizierende Element oder Kontrollflussmuster mit einem semantisch äquivalenten Element verknüpft, welches bereits über eine präzise Beschreibung verfügt [Fehr89, Baum92, RiSS93]. Beispielsweise wurde für die Beschreibung der Kontrollflusssemantik der Kontrollflussmuster der Workflow Pattern Initiative die Modellierungssprache newYAWL verwendet, deren Kontrollflusssemantik mit den gefärbten Petri-Netzen in [Russ07] präzise beschrieben ist. Im Folgenden werden die vier Beschreibungsarten, die Übersetzungssemantik sowie die operationelle, denotationelle und axiomatische Semantik vorgestellt und auf entsprechende Ansätze aus der Literatur referenziert.

Bei der *Übersetzungssemantik* werden Elemente, deren Kontrollflusssemantik nicht präzise beschrieben ist, mit semantisch ähnlichen Elementen verknüpft, die wiederum eine präzise Kontrollflusssemantik besitzen. Die Modellierungssprachen, die zur Modellierung von Geschäftsprozessen eingesetzt werden und deren Kontrollflusssemantik nicht präzise beschrieben ist, werden häufig in Petri-Netze oder Varianten von Petri-Netzen, wie die gefärbten Petri-Netze (CPN, Colored Petri Net) oder Recursive extended concurrent algebraic term nets (RECATNets), transformiert [Aals99, DiDO08, Ritt99a, Dehn03, MeAa07, ChSc94, LaSW97, LaSW98, MoRo00, DeRi01, Ritt99b, Ritt99c, RPUW+07, OuLi08, KhBI17, Take08, HuLS10, RaEH10, CoPR15]. Darüber hinaus wird die Kontrollflusssemantik auch häufig mit der Modellierungssprache YAWL bzw. newYAWL spezifiziert [DDDG08, YSWS08, MMNV+06, YSSW08, MeMN06, YeSo10]. Die Kontrollflusssemantik einer Geschäftsprozessmodellierungssprache kann ebenfalls durch die Zuweisung der einzelnen Elemente zu einem Metamodell erfolgen, wie dies anhand des ADOxx Meta^2 Models [FiKa13] deutlich wird. Ein weiterer Ansatz ist die Annotation der Elemente mit Metainformationen, wie beispielsweise in [Fill11, Fill12] aufgezeigt. Hierbei werden den Elementen z. B. Wahrscheinlichkeitsverteilungen für Verzweigungen, Ausführungszeiten oder auch Risiken zugewiesen.

Mit einer *operationellen Semantik* kann die Kontrollflusssemantik einer Geschäftsprozessmodellierungssprache, eines Geschäftsprozessmodells oder eines Kontrollflussmusters in abstrakterer Weise als mit einer Übersetzungssemantik definiert werden. Eine solche Semantik basiert auf einer abstrakten oder auch einer konkreten Maschine. Die Grundidee besteht darin, dass eine Folge von Zuständen beschrieben wird, die schrittweise abgearbeitet wird, d. h. ausgehend vom Anfangszustand wird Anweisung für Anweisung abgearbeitet. Der Übergang von einem Zustand in den nächsten Zustand wird mit einer Überführungsfunktion beschrieben und ist Hauptbestandteil der Maschine, so dass viele maschinenabhängige Details in die Semantikbeschreibung mit einfließen. Damit diese allgemeingültig bleibt, wird angestrebt abstrakte Maschinen mit möglichst wenigen Elementen einzusetzen und diese dann von den konkreten Maschinen zu interpretieren. Die Abhängigkeit der Semantik vom Interpreter bleibt dennoch bestehen, sodass die operationellen Semantiken

in der Regel auch sehr implementierungsnah sind [Fehr89, Baum92, RiSS93]. Die Kontrollflusssemantik der Petri-Netze und YAWL bzw. newYAWL wurde bereits bei der Konzeption mit einer operationellen Kontrollflusssemantik ausgestattet. In der Literatur existieren zahlreiche Ansätze mit einer operationellen Semantik zur präzisen Beschreibung der Kontrollflusssemantik, wie dies beispielsweise anhand der Modellierungssprachen EPK [Rump99, Kind06, MeNü03, MeAa07], BPMN [KIGK+14, BöSö11, BöTh08b, BöTh08a, PrQZ08, CPRT15, ChCH11, GoDi13, ElBo14] und UML-AD [PoPr04, BöCR00, Eshu02, KnGo10, Sars06, SaGu06, ViKa05] deutlich wird.

Der Abstraktionsgrad einer *denotationellen Semantik* ist höher als der einer operationellen Semantik, da von der schrittweisen Zustandsänderung einer Maschine abstrahiert wird. Stattdessen wird die Semantik durch eine statische Abbildung von einem Anfangszustand in einen Endzustand präzise beschrieben. Die Abbildung ist die Zuweisung eines diskreten Objektes, einer stetigen Funktion oder einer Funktion höherer Ordnung und beschreibt die Wirkung auf die beteiligten Variablen. Dementsprechend liegt für jeden Zustand ein geordnetes Tupel vor, das sich nur in den Werten der Elemente des Tupels unterscheidet. Die Ablaufreihenfolge wird durch die Hintereinander-Ausführung der beschriebenen Funktionen deutlich. Die ersten Ansätze einer denotationellen Semantik gehen auf McCarthy [McCa62] zurück, der einfache Flussdiagramme in rekursive Gleichungssysteme mit Zustandsvektoren überführte [Fehr89, Baum92, RiSS93]. Im Kontext der Beschreibung der Kontrollflusssemantik existieren verschiedene denotationelle Semantikbeschreibungen [Herb14]. Häufig werden auch Prozesskalküle, wie das Kalkül kommunizierender sequentieller Prozesse (Communicating Sequential Processes, kurz: CSP) [WoGi08, WoGi11, CaMe12, CPPR10] oder das π-Kalkül [Puhl07, PuWe06] verwendet.

Die *axiomatische Semantik* ist die abstrakteste Möglichkeit, die Kontrollflusssemantik einer Geschäftsprozessmodellierungssprache, eines Geschäftsprozessmodels oder eines Kontrollflussmusters präzise zu beschreiben. Mit den Axiomen werden nicht die Zustandsübergänge definiert, sondern es werden die Eigenschaften bzw. die Gültigkeitsanforderungen der Zustände beschrieben [Witt07, Tuck04, GrRu10]. Dementsprechend werden keine Zustände oder Zustandsübergänge definiert, sondern lediglich die Eigenschaften der Zustände. Die Eigenschaften der Zustände werden mit logischen Aussagen formuliert, die Vor- und Nachbedingungen besitzen und als Zusicherungen bezeichnet werden. Eine logische Aussage p, die vor der Ausführung einer Anweisung A gilt, wird als Vorbedingung bezeichnet. Entsprechend gilt nach der Ausführung der Anweisung A die logische Aussage q als Nachbedingung. Der Zusammenhang wird als $\{\mathrm{p}\}\,\mathrm{A}\,\{\mathrm{q}\}$ formuliert [Fehr89, Baum92, RiSS93].

4.3 Zusammenfassung

In den vorherigen Abschnitten wurden die Begriffe Kontrollfluss, Muster und Kontrollflussmuster definiert. Daran anknüpfend wurden die Kontrollflussmuster von Börger und die Kontrollflussmuster der Workflow Pattern Initiative beschrieben und miteinander verglichen. Ausführlich wurden die Kontrollflussmuster der Workflow Pattern Initiative betrachtet, für die die Kontrollflusssemantik jeweils mit einer newYAWL-Spezifikation präzise beschrieben wurde. Zur präzisen Beschreibung einer Kontrollflusssemantik wurde auf die Übersetzungssemantik, die operationelle, denotationelle und die axiomatische Semantik eingegangen.

Die praktische Relevanz einer präzisen Kontrollflusssemantik einer Geschäftsprozessmodellierungssprache bzw. eines Geschäftsprozessmodells wird anhand der einzelnen Phasen des Geschäftsprozessmanagements (vgl. Abbildung 3) deutlich. Dementsprechend können in der Dokumentationsphase aufgrund einer präzisen Kontrollflusssemantik Gültigkeitsanfragen formuliert werden, um z. B. das Geschäftsprozessmodell auf Widerspruchsfreiheit, Redundanzfreiheit oder auch auf inhaltliche Aspekte hin überprüfen zu können. Bei der Prozessanalyse können unter anderem IT-basierte Analyseverfahren eingesetzt werden, so dass z. B. quantitative Aussagen über die Mindestdauer, den Ressourcenverbrauch oder die Kapazitätsauslastung möglich werden. Auf der Grundlage einer präzisen Kontrollflusssemantik können zudem Simulationsexperimente durchgeführt werden, um während der Prozesskonzeption mögliche Geschäftsprozessmodellalternativen miteinander vergleichen zu können. Darüber hinaus können durch die präzise Kontrollflusssemantik ungewollte Mehrdeutigkeiten bei der Implementierung vermieden werden, welche sich wiederum auf das Prozesscontrolling auswirken können [Ober96, Aals10].

Die Kontrollflussmuster der Workflow Pattern Initiative ermöglichen es komplexe Abläufe aus einfachen Grundstrukturen zusammenzusetzen. Im Folgenden wird eine auf den obigen Kontrollflussmustern basierende Methode zur Beschreibung der Kontrollflusssemantik von graphischen Geschäftsprozessmodellierungssprachen vorgestellt. Dabei werden zunächst den Syntaxelementen der Modellierungssprache Kontrollflussmuster zur Beschreibung der Kontrollflusssemantik zugeordnet. Darauf aufbauend kann für jedes Element im Geschäftsprozessmodell eines dieser Kontrollflussmuster ausgewählt werden und für die präzise Beschreibung der Kontrollflusssemantik die entsprechenden newYAWL-Parameter spezifiziert werden. Es wird zwischen musterspezifischen und modellspezifischen Parametern differenziert, wie beispielsweise am Kontrollflussmuster exklusive Auswahl (wcp_4) deutlich wird. Wenn ein Element in einem Geschäftsprozessmodell durch das Kontrollflussmuster exklusive Auswahl spezifiziert ist, müssen die Split-Funktion (Split), die Reihenfolge

der Auswertungssequenz ($<_{\mathrm{XOR}}$), die Kantenbedingungen (ArcCond) und die Standardkante (Default) definiert werden. Die Split-Funktion kann für das Element mit dem Funktionswert XOR spezifiziert werden, unabhängig von einem konkreten Element im Geschäftsprozessmodell (musterspezifischer Parameter). Dahingegen ist die Spezifikation der Reihenfolge der Auswertungssequenz ($<_{\mathrm{XOR}}$), der Kantenbedingungen (ArcCond) und der Standardkante (Default) von dem konkreten Element im Geschäftsprozessmodell abhängig (modellspezifische Parameter). Die Zuweisung der Kontrollflussmuster zu den Syntaxelementen einer Geschäftsprozessmodellierungssprache wird in Kapitel 5 betrachtet sowie die Spezifikation der muster- und modellspezifischen newYAWL-Parameter in Kapitel 6.

5 Kontrollflusssemantik für Geschäftsprozessmodellierungssprachen

Missverständnisse zwischen Domänenexperten und Modellierern entstehen typischerweise immer dann, wenn die Bedeutung einzelner Symbole einer Geschäftsprozessmodellierungssprache nicht eindeutig festgelegt ist. Die meisten in der Praxis eingesetzten Modellierungssprachen besitzen keine präzise Kontrollflusssemantik, wodurch ungewollte Mehrdeutigkeiten entstehen. Unpräzise Beschreibungen erschweren bzw. verhindern die Simulation, Analyse, Überwachung und Verbesserung der Geschäftsprozesse sowie die Kommunikation über die Geschäftsprozesse. Die Kontrollflussmuster der Workflow Pattern Initiative aus Kapitel 4.1.2 können als Hilfsmittel für die Modellierung eingesetzt werden, um komplexe Abläufe aus einfachen Grundstrukturen zusammenzusetzen. Im Folgenden wird eine auf Kontrollflussmustern basierende Methode zur Beschreibung der Kontrollflusssemantik von graphischen Geschäftsprozessmodellierungssprachen vorgestellt. Zunächst aber werden die notwendigen Anforderungen an die Geschäftsprozessmodellierungssprachen definiert. Darauf aufbauend wird das Vorgehen zur Beschreibung der Kontrollflusssemantik von Geschäftsprozessmodellierungssprachen aufgezeigt und auf die Modellierungssprachen aus Kapitel 3 angewendet. Abgerundet wird das Kapitel mit einer Zusammenfassung.

5.1 Anforderungen an die Geschäftsprozessmodellierungssprachen

Für die Anwendung der Methode zur Beschreibung der Kontrollflusssemantik von graphischen Geschäftsprozessmodellierungssprachen müssen zwei Anforderungen erfüllt werden.

(1) Zum einen müssen die Syntaxelemente der Geschäftsprozessmodellierungssprache in zwei disjunkte Mengen (Knoten- und Kantentyp) untergliedert werden können, um daraufhin die Kontrollflusssemantik für die Knotentypen, in Abhängigkeit der Kantentypen, definieren zu können (vgl. Abbildung 29).

(2) Zum anderen muss die Kontrollflusssemantik eines Knotentyps der Geschäftsprozessmodellierungssprache durch die Kontrollflusssemantik «Vor der Ausführung»,

«Während der Ausführung» und «Nach der Ausführung» beschrieben werden können (vgl. Abbildung 30).

Die Knotentypen einer Geschäftsprozessmodellierungssprache sind typischerweise Aktivitätsknoten sowie auch die Elemente zur Ablaufsteuerung (z. B. Verzweigung oder Zusammenführung). Beispielsweise besteht die Modellierungssprache EPK aus den Knotentypen: Funktion (vgl. Abbildung 16a), Ereignis (vgl. Abbildung 16b), XOR-, AND- und OR-Verknüpfungsoperator (vgl. Abbildung 16c) sowie der Prozessschnittstelle (vgl. Abbildung 16e). Mit den Kantentypen kann die Ablaufreihenfolge zwischen den Knotentypen beschrieben werden. In graphischen Geschäftsprozessmodellierungssprachen werden typischerweise Pfeile bzw. gerichtete Kanten zur Modellierung der Kantentypen verwendet (vgl. z. B. für BPMN Abbildung 13). Ungerichtete Kanten (vgl. z. B. für BPMN Abbildung 13g) sind häufig Beziehungen und zählen ebenfalls zu den Kantentypen. Die Menge der Knoten- und Kantentypen sollte außerdem nicht leer und endlich sein. Aus Praktikabilitätsgründen wird angenommen, dass die Mengen der Knoten- und Kantentypen jeweils endlich sind. Die Knoten- und Kantentypen einer Geschäftsprozessmodellierungssprache sind spezialisierte Syntaxelemente, sodass die Komponenten einer Geschäftsprozessmodellierungssprache in Abbildung 29 nach [FiKa13] um die Kanten- und Knotentypen erweitert wurden. Die Kontrollflusssemantik wird einem Knotentyp zugeordnet, mit der Folge, dass die Assoziation von der Semantikzuordnung zum Syntax (vgl. Abbildung 6) durch die Assoziation von der Semantikzuordnung zum Knotentyp ersetzt wird (vgl. Abbildung 29).

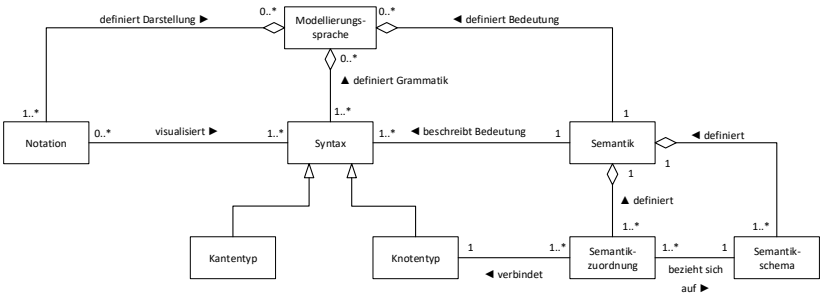


Abbildung 29: Modell - Elemente einer Geschäftsprozessmodellierungssprache (In Anlehnung an [FiKa13]) - Erweiterung um Kanten- und Knotentyp

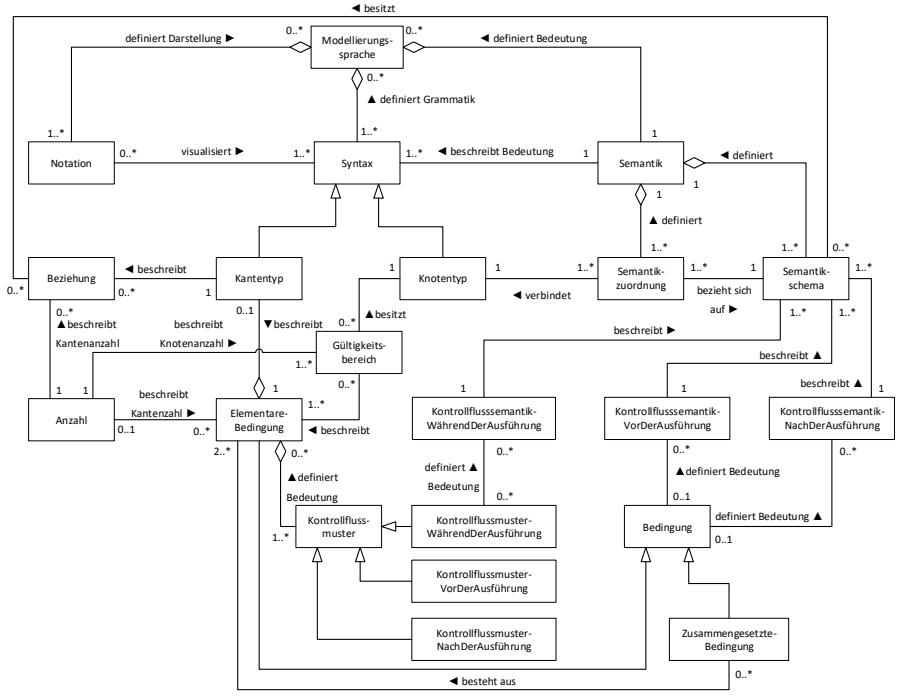


Abbildung 30: Modell - Elemente einer Geschäftsprozessmodellierungssprache (In Anlehnung an [FiKa13]) - Erweiterung um Kanten- und Knotentyp und Verfeinerung des Semantikschemas

Die zweite Anforderung ist, dass das Semantikschema einer Geschäftsprozessmodellierungssprache bzw. die Kontrollflusssemantik eines Knotentyps durch eine Kombination der

- Kontrollflusssemantik «Vor der Ausführung» (vgl. Klasse `KontrollflussemantikVorDerAusführung` in Abbildung 30),
- Kontrollflusssemantik «Während der Ausführung» (vgl. Klasse `KontrollflussemantikWährendDerAusführung` in Abbildung 30) und
- Kontrollflusssemantik «Nach der Ausführung» (vgl. Klasse `KontrollflussemantikNachDerAusführung` in Abbildung 30)

beschrieben werden kann. Die Kontrollflusssemantik wird durch die

- Kontrollflussmuster «Vor der Ausführung» (vgl. Klasse `KontrollflussmusterVorDerAusführung` in Abbildung 30),
- Kontrollflussmuster «Während der Ausführung» (vgl. Klasse `KontrollflussmusterWährendDerAusführung` in Abbildung 30) und

– Kontrollflussmuster «nach der Ausführung» (vgl. Klasse `KontrollflussmusterNachDerAusführung` in Abbildung 30)

bestimmt. Mit den Kontrollflussmustern wird eine geeignete Auswahl von Mustern zur Verfügung gestellt, um die Kontrollflusssemantik eines Knotentyps beschreiben zu können. Die Kontrollflussmuster sowie deren Kontrollflusssemantik wurde bereits ausführlich anhand von Beispielen mit der Modellierungssprache newYAWL in Kapitel 4.1.2 vorgestellt. Im Folgenden wird die Zusammensetzung der Kontrollflusssemantik «Vor der Ausführung», «Während der Ausführung» und «Nach der Ausführung» betrachtet.

Die Kontrollflusssemantik «Vor der Ausführung» (vgl. Klasse `KontrollflusssemantikVorDerAusführung` in Abbildung 30) wird auch als Vorbedingung bezeichnet, da diese Kontrollflussmuster für die Aktivierung der Elemente im Geschäftsprozessmodell verantwortlich sind. Die Kontrollflusssemantik der Vorbedingung wird durch eine elementare Bedingung (vgl. Klasse `ElementareBedingung` in Abbildung 30) oder eine zusammengesetzte Bedingung (vgl. Klasse `ZusammengesetzteBedingung` in Abbildung 30) beschrieben. Die zusammengesetzte Bedingung besteht aus zwei oder mehreren elementaren Bedingungen. Eine elementare Bedingung setzt sich typischerweise

– aus einem Kantentyp der Modellierungssprache (vgl. Klasse `Kantentyp` in Abbildung 30),
– der eingehenden Kantenanzahl (vgl. Klasse `Anzahl` in Abbildung 30) und
– dem Gültigkeitsbereich (vgl. Klasse `Gültigkeitsbereich` in Abbildung 30)

zusammen. Der Gültigkeitsbereich kann die elementare Bedingung auf bestimmte

– Knotentypen (vgl. Klasse `Knotentyp` in Abbildung 30) im Vorbereich sowie
– deren Häufigkeit (vgl. Klasse `Anzahl` in Abbildung 30)

einschränken. Eine elementare Bedingung kann mit einer beliebigen Anzahl von Kontrollflussmustern «Vor der Ausführung» (vgl. Klasse `KontrollflussmusterVorDerAusführung` in Abbildung 30) verknüpft werden. Zu den Kontrollflussmustern «Vor der Ausführung» gehören die folgenden Kontrollflussmuster der Workflow Pattern Initiative:

– Sequenz (wcp_1),
– Synchronisation (wcp_3),
– Einfache Zusammenführung (wcp_5),
– Strukturierte synchronisierte Zusammenführung (wcp_7),
– Mehrfachzusammenführung (wcp_8),
– Strukturierter Diskriminator (wcp_9),

- Blockierender Diskriminator (wcp_{28}),
- Abbrechender Diskriminator (wcp_{29}),
- Strukturierte partielle Zusammenführung (wcp_{30}),
- Blockierende partielle Zusammenführung (wcp_{31}),
- Abbrechende partielle Zusammenführung (wcp_{32}),
- Generalisierte Und-Zusammenführung (wcp_{33}),
- Lokale synchronisierte Zusammenführung (wcp_{37}),
- Generelle synchronisierte Zusammenführung (wcp_{38}) und
- Thread Zusammenführung (wcp_{41}).

Darüber hinaus wird das Kontrollflussmuster Start (wcp_{Start}) definiert und den Kontrollflussmustern «Vor der Ausführung» zugeordnet. Unter den Kontrollflussmustern der Workflow Pattern Initiative existieren keine Kontrollflussmuster, um Startknoten zu kennzeichnen. Eine Startbedingung hat keine eingehenden Kanten und ist somit auch von keinem Kantentyp und keinem Gültigkeitsbereich abhängig. Die Kontrollflusssemantik einer Startbedingung wird in einem newYAWL-Beispielsprozess in Abbildung 31 durch die Bedingung Start beschrieben.

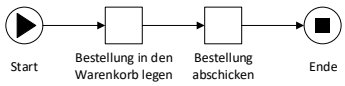


Abbildung 31: Beispiel - Kontrollflussmuster Start (wcp_{Start})

Das Pendant zur Kontrollflusssemantik «Vor der Ausführung» ist die Kontrollflusssemantik «Nach der Ausführung» (vgl. Klasse `KontrollflusssemantikNachDerAusführung` in Abbildung 30), welche auch als Nachbedingung bezeichnet wird. Die Nachbedingung beschreibt die Kontrollflusssemantik eines Elementes nach der Ausführung. Die Kontrollflusssemantik einer Nachbedingung wird, wie die Vorbedingung, durch eine elementare Bedingung oder zusammengesetzte Bedingung beschrieben. Die zusammengesetzte Bedingung besteht aus zwei oder mehreren elementaren Bedingungen. Der Unterschied zur Vorbedingung ist, dass anstatt der eingehenden Kantenanzahl, die ausgehende Kantenanzahl spezifiziert werden muss und die Kontrollflusssemantik einer Bedingung mit den Kontrollflussmustern «Nach der Ausführung» (vgl. Klasse `KontrollflussmusterNachDerAusführung` in Abbildung 30) beschrieben wird. Darüber hinaus wird der Gültigkeitsbereich für den Nachbereich definiert, anstatt für den Vorbereich. Zu den Kontrollflussmustern «Nach der Ausführung» gehören die folgenden 8 Kontrollflussmuster der Workflow Pattern Initiative:

- Sequenz (wcp_1),
- Parallele Aufspaltung (wcp_2),
- Exklusive Auswahl (wcp_4),
- Mehrfachauswahl (wcp_6),
- Implizite Beendigung (wcp_{11}),
- Aufgeschobene Auswahl (wcp_{16}),
- Thread Aufspaltung (wcp_{42}) und
- Explizite Beendigung (wcp_{43}).

Eine elementare Bedingung, deren Kontrollflusssemantik mit den Kontrollflussmustern wcp_{11} oder wcp_{43} beschrieben wird, besitzt keinen Kantentyp und ist somit auch von keiner Kantenanzahl und keinem Gültigkeitsbereich abhängig.

Die Kontrollflusssemantik «Während der Ausführung» (vgl. Klasse `KontrollflussemantikWährendDerAusführung` in Abbildung 30) beschreibt die Kontrollflusssemantik zwischen der Kontrollflusssemantik «Vor der Ausführung» und «Nach der Ausführung». Aus diesem Grund werden typischerweise nur Aktivitäten mit den Kontrollflussmustern «Während der Ausführung» spezifiziert, da in der Regel Ereignisse, wie auch Verzweigungs- und Zusammenführungselemente kein zu spezifizierendes Verhalten während der Ausführung besitzen. Die Kontrollflusssemantik «Während der Ausführung» wird nicht mit Bedingungen beschrieben, sondern direkt durch die Zuordnung einer beliebigen Anzahl von Kontrollflussmustern «Während der Ausführung» (vgl. Klasse `KontrollflussmusterWährendDerAusführung` in Abbildung 30). Zu den Kontrollflussmustern «Während der Ausführung» gehören die folgenden Kontrollflussmuster der Workflow Pattern Initiative:

- Mehrfachinstanz ohne Synchronisation (wcp_{12}),
- Mehrfachinstanz mit Festlegung der Anzahl der Instanzen zur Entwurfszeit (wcp_{13}),
- Mehrfachinstanz mit Festlegung der Anzahl der Instanzen vor der Laufzeit des Elementes (wcp_{14}),
- Mehrfachinstanz mit Festlegung der Anzahl der Instanzen ohne Informationen vor der Laufzeit des Elementes (wcp_{15}),
- verschachtelte parallele Ausführung (wcp_{17}),
- Meilenstein (wcp_{18}),
- Element abbrechen (wcp_{19}),
- Geschäftsprozessinstanz abbrechen (wcp_{20}),
- strukturierte Schleife (wcp_{21}),
- transienter Trigger (wcp_{23}),
- persistenter Trigger (wcp_{24}),

- Bereich abbrechen (wcp_{25}),
- Mehrfachinstanzelement abbrechen (wcp_{26}),
- Mehrfachinstanzelement beenden (wcp_{27}),
- statische partielle Zusammenführung für Mehrfachinstanzen (wcp_{34}),
- abbrechende partielle Zusammenführung für Mehrfachinstanzen (wcp_{35}),
- dynamische partielle Zusammenführung für Mehrfachinstanzen (wcp_{36})
- kritischer Bereich (wcp_{39}) und
- verschachtelte Ausführung (wcp_{40}).

Zu den zur Verfügung stehenden Kontrollflussmustern der Workflow Pattern Initiative wird zusätzlich das Kontrollflussmuster Teilprozess ($wcp_{Teilprozess}$) definiert und den Kontrollflussmustern «Während der Ausführung» zugeordnet. Mit dem Kontrollflussmuster Teilprozess können Bereiche eines Geschäftsprozessmodells durch ein einzelnes Element ersetzt werden. Ein Bereich ist eine beliebige Sammlung von Elementen im Geschäftsprozessmodell. In Abbildung 32 wird das Kontrollflussmuster Teilprozess mit einer newYAWL-Spezifikation beschrieben. Das Notationselement *Bonität prüfen (t_1)* ist ein Teilprozess, wobei der Kontrollfluss am *Startknoten (i_2)* fortgesetzt wird, sobald der Teilprozess *Bonität prüfen (t_1)* instanziiert wird. Der Teilprozess wird solange ausgeführt, bis der Kontrollfluss den *Endknoten (o_2)* erreicht hat, woraufhin die Ausführung des gesamten Teilprozesses abgeschlossen ist.

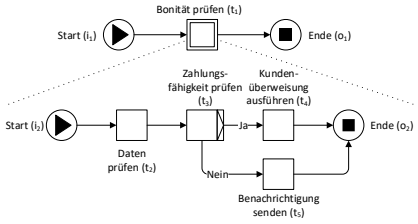


Abbildung 32: Beispiel - Kontrollflussmuster Teilprozess ($wcp_{Teilprozess}$)

Aus der vorherigen Auflistung der Kontrollflussmuster geht hervor, dass nicht alle Kontrollflussmuster der Workflow Pattern Initiative zur Beschreibung der Kontrollflusssemantik verwendet wurden. Die außer Acht gelassenen Kontrollflussmuster sind

- die beliebigen Zyklen (wcp_{10}) und
- die Rekursion (wcp_{22}).

Das Kontrollflussmuster *beliebige Zyklen* (wcp_{10}) wird nicht betrachtet, da mit diesem Kontrollflussmuster nur ein Bereich beschrieben wird, der mehrere Ein- und Ausstiegspunkte besitzt. Mit dem Kontrollflussmuster *Rekursion* (wcp_{22}) wird ein Bereich beschrieben, der wiederholt ausgeführt und von sich selbst wieder aufgerufen wird. Das Kontrollflussmuster sieht keine Bedingungen vor, nach welchen Kriterien der Bereich erneut ausgeführt werden soll. Dementsprechend müssen die einzelnen Aspekte explizit modelliert werden. Eine mögliche Variante zur Modellierung dieser Aspekte ist die Verwendung eines Teilprozesses, der innerhalb des Teilprozesses wieder erneut aufgerufen wird, wie auch aus der Beschreibung des Kontrollflussmusters Rekursion in Kapitel 4.1.2.6 hervorgeht.

Zur Beschreibung der Kontrollflusssemantik einer Geschäftsprozessmodellierungssprache mit der im Folgenden entwickelten Methode müssen die eingangs definierten Anforderungen erfüllt sein. Für alle Geschäftsprozessmodellierungssprachen aus Kapitel 3 ist die erste Anforderung erfüllt (vgl. Abbildung 29), d. h. die Syntaxelemente der Modellierungssprache können in die zwei disjunkten Mengen Knoten- und Kantentyp untergliedert werden. Die zweite Anforderung wird von den betrachteten Modellierungssprachen aus Kapitel 3, d. h. EPK, UML-AD, Petri-Netz und YAWL bzw. newYAWL erfüllt. Dementsprechend kann die Kontrollflusssemantik der Knotentypen dieser Modellierungssprachen durch die Kontrollflusssemantik «Vor der Ausführung», «Während der Ausführung» und «Nach der Ausführung» (vgl. Abbildung 30) beschrieben werden. Bei der Modellierungssprache BPMN gibt es Knotentypen bzw. Verbindungen zu anderen Knotentypen, die nicht mit den obigen Kontrollflussmustern beschrieben werden können. Für diesen Fall müssen benutzerdefinierte Kontrollflussmuster definiert werden, um die Kontrollflusssemantik vollständig beschreiben zu können. Aus diesem Grund werden im Folgenden vier benutzerdefinierte Kontrollflussmuster eingeführt (vgl. Tabelle 8).

BPMN-Element	Beschreibung der Kontrollflusssemantik
Angeheftetes unterbrechendes Ereignis an eine Aktivität (BPMN, vgl. z. B. Abbildung 33)	Unterbrechende Ereignisse können an Aktivitäten angeheftet werden, mit der Kontrollflusssemantik, sodass, wenn das angeheftete Ereignis eingetreten ist, die Aktivität abgebrochen wird.
Angeheftetes nicht-unterbrechendes Ereignis an eine Aktivität (BPMN, vgl. z. B. Abbildung 35)	Siehe Beschreibung angeheftetes unterbrechendes Ereignis an eine Aktivität, mit dem Unterschied, dass der Kontrollfluss am angehefteten Ereignis nebenläufig fortgeführt und die Aktivität somit nicht abgebrochen wird.
Ereignis-Teilprozess mit unterbrechendem Startereignis (BPMN, vgl. z. B. Abbildung 37)	Ereignis-Teilprozesse sind in Teilprozesse eingebettet und brechen den Teilprozess ab, wenn das Startereignis des Ereignis-Teilprozesses eingetreten ist. Das Startereignis des Ereignis-Teilprozesses kann jedoch nur den Ereignis-Teilprozess instanziieren, wenn der Teilprozess aktiv ist. Ein Teilprozess ist aktiv, wenn dieser instanziiert und noch nicht beendet ist.

BPMN-Element	Beschreibung der Kontrollflusssemantik
Ereignis-Teilprozess mit nicht-unterbrechendem Startereignis (BPMN, vgl. Abbildung 39)	Siehe Beschreibung Ereignis-Teilprozess mit unterbrechendem Startereignis, mit dem Unterschied, dass der Teilprozess nicht abgebrochen, sondern der Teilprozess nebenläufig zum Ereignis-Teilprozess ausgeführt wird.

Tabelle 8: Überblick - Kontrollflusssemantik in BPMN, die nicht mit den bisherigen Kontrollflussmustern abgebildet werden kann

Die Kontrollflusssemantik der angehefteten Ereignisse an Aktivitäten und der Ereignis-Teilprozesse in BPMN, jeweils mit unterbrechendem und nicht-unterbrechendem Ereignis, kann nicht mit den bereits vorgestellten Kontrollflussmustern abgebildet werden, sodass im Folgenden die Kontrollflussmuster:

- angeheftetes unterbrechendes Ereignis an eine Aktivität:
  ($wcp_{BPMNAngeheftetesUnterbrechendesEreignis}$)
- angeheftetes nicht-unterbrechendes Ereignis an eine Aktivität:
  ($wcp_{BPMNAngeheftetesNichtUnterbrechendesEreignis}$)
- Ereignis-Teilprozess mit unterbrechendem Startereignis:
  ($wcp_{BPMNEreignisTeilprozessUnterbrechendesEreignis}$)
- Ereignis-Teilprozess mit nicht-unterbrechendem Startereignis:
  ($wcp_{BPMNEreignisTeilprozessNichtUnterbrechendesEreignis}$)

definiert werden. Ein *angeheftetes unterbrechendes Ereignis an eine BPMN-Aktivität* wird in Abbildung 33 und Abbildung 34 anhand eines BPMN-Beispiels illustriert und dessen Kontrollflusssemantik wird mit einer newYAWL-Spezifikation beschrieben. Das Kontrollflussmuster wird einer BPMN-Aktivität zugeordnet und erhält die Bezeichnung $wcp_{BPMNAngeheftetesUnterbrechendesEreignis}$. Zur Spezifikation des Musters müssen die angehefteten Ereignisse ausgewählt werden. Die Ereignisse können nur die BPMN-Aktivität beeinflussen, solange diese ausgeführt wird. Die Ereignisse, die die angehefteten Ereignisse auslösen, können entweder nicht modelliert (vgl. Abbildung 33) oder explizit modelliert (vgl. Abbildung 34) sein. Die Kontrollflusssemantik des angehefteten Ereignisses wird durch die Zuordnung der Kontrollflussmuster zu diesem Ereignis realisiert. Sofern das auslösende Ereignis des angehefteten Ereignisses nicht modelliert wurde (vgl. Abbildung 33), ist das angeheftete Ereignis mit einem transienten Trigger zu spezifizieren. Andernfalls besitzt das auslösende Ereignis einen impliziten Sequenzfluss zum angehefteten Ereignis. Die Kontrollflusssemantik anderer angehefteter Ereignisse kann analog definiert werden.

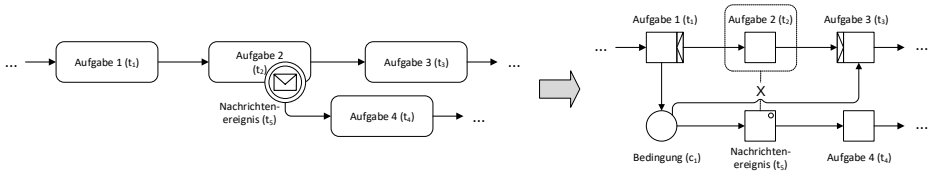


Abbildung 33: Beispiel - Angeheftetes unterbrechendes Ereignis an eine Aktivität mit nicht explizit modelliertem Auslöser

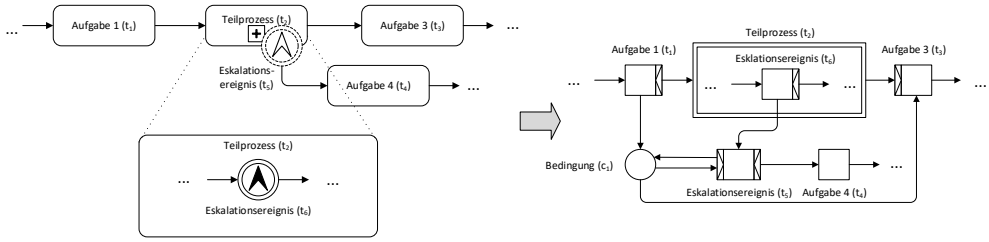


Abbildung 34: Beispiel - Angeheftetes unterbrechendes Ereignis an eine Aktivität mit explizit modelliertem Auslöser

Ein *angeheftetes nicht-unterbrechendes Ereignis an eine BPMN-Aktivität* wird in Abbildung 35 und Abbildung 36 anhand eines BPMN-Beispiels illustriert und dessen Kontrollflusssemantik mit einer newYAWL-Spezifikation beschrieben. Im Unterschied zum Kontrollflussmuster *angeheftetes unterbrechendes Ereignis an eine Aktivität* wird die Aktivität nicht abgebrochen, sondern nebenläufig ausgeführt. Das Ereignis kann eine nebenläufige Ausführung nur ermöglichen, solange die entsprechende BPMN-Aktivität ausgeführt wird. Das angeheftete Ereignis kann auch mehrfach ausgelöst werden. Das Kontrollflussmuster wird als wcp$_{\text{BPMNAngeheftetesNichtUnterbrechendesEreignis}}$ bezeichnet.

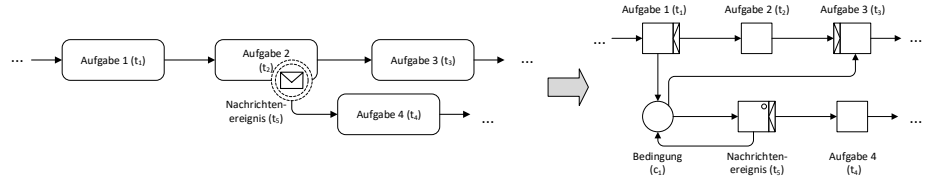


Abbildung 35: Beispiel - Angeheftetes nicht-unterbrechendes Ereignis an eine Aktivität mit nicht explizit modelliertem Auslöser

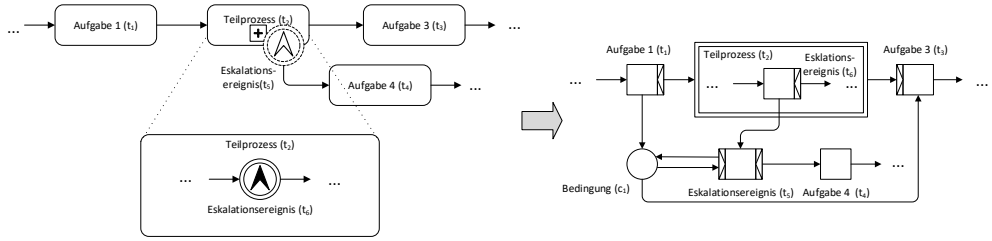


Abbildung 36: Beispiel - Angeheftetes nicht-unterbrechendes Ereignis an eine Aktivität mit explizit modelliertem Auslöser[1]

Ein *Ereignis-Teilprozess mit unterbrechendem Startereignis* wird in Abbildung 37 und Abbildung 38 anhand eines BPMN-Beispiels illustriert und dessen Kontrollflusssemantik wird mit einer newYAWL-Spezifikation beschrieben. Das Kontrollflussmuster wird den eingebetteten Teilprozessen zugeordnet. Für die Spezifikation des Musters muss der Ereignis-Teilprozess ausgewählt werden. Ein Ereignis-Teilprozess (t_5) kann nur solange instanziiert werden, wie der Teilprozess (t_1) noch aktiv ist. Nach der Instanziierung des Ereignis-Teilprozesses (t_5) werden die Elemente des Teilprozesses abgebrochen. Nach Beendigung des Ereignis-Teilprozesses ist der Teilprozess (t_1) beendet. Diese Kontrollflusssemantik wird durch das Kontrollflussmuster $wcp_{BPMNEreignisTeilprozessUnterbrechendesEreignis}$ abgebildet. Der Auslöser des Startereignisses des Ereignis-Teilprozesses kann entweder nicht modelliert (vgl. Abbildung 37) oder explizit modelliert (vgl. Abbildung 38) sein. Falls das auslösende Ereignis nicht explizit modelliert wurde, besitzt das Startereignis (i_3) des Ereignis-Teilprozesses einen Trigger (vgl. Abbildung 37). Sofern das auslösende Ereignis des Startereignisses des Ereignis-Teilprozesses explizit modelliert wurde (vgl. Abbildung 38), existiert ein impliziter Sequenzfluss vom auslösenden Element (t_6) zum Startereignis des Ereignis-Teilprozesses (i_3).

[1] Die Kante vom Eskalationsereignis (t_6) bzw. dem explizit modellierten Auslöser zum Eskalationsereignis (t_5) bzw. dem angehefteten Ereignis ist nach der Syntax der newYAWL-Spezifikation nicht zulässig. In einer newYAWL-Spezifikation darf keine Bedingung oder Aufgabe innerhalb einer zusammengesetzten Aufgabe über eine Kante mit einer Bedingung oder Aufgabe außerhalb einer zusammengesetzten Aufgabe verbunden werden. Im Rahmen des Beispiels wird diese Kante dennoch als Kurzschreibweise verwendet, um den Teilprozess zu erhalten und somit zur Verständlichkeit der Kontrollflusssemantik beizutragen. Zur korrekten vollständigen Abbildung der Syntax müsste die zusammengesetzte Aufgabe aufgelöst werden, d. h. die Bedingungen und Aufgaben der zusammengesetzten Aufgabe in den Kontrollfluss des übergeordneten Prozesses eingebettet werden.

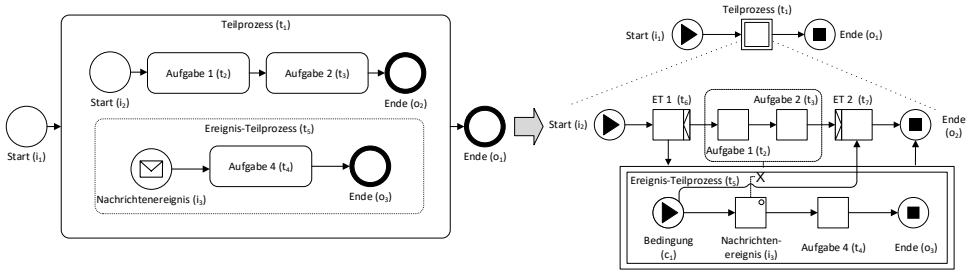


Abbildung 37: Beispiel - Ereignis-Teilprozess mit unterbrechendem Startereignis und nicht explizit modelliertem Auslöser[2]

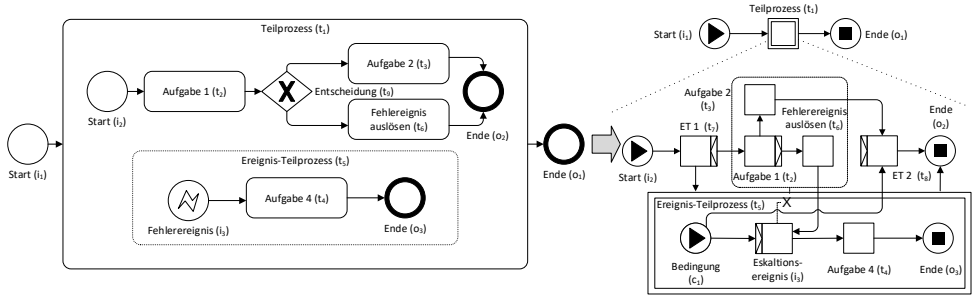


Abbildung 38: Beispiel - Ereignis-Teilprozess mit unterbrechendem Startereignis und explizit modelliertem Auslöser[3]

Ein *Ereignis-Teilprozess mit nicht-unterbrechendem Startereignis* wird in Abbildung 39 und Abbildung 40 anhand eines BPMN-Beispiels illustriert und dessen Kontrollflusssemantik wird mit einer newYAWL-Spezifikation beschrieben. Im Unterschied zum Kontrollflussmuster Ereignis-Teilprozess mit unterbrechendem Startereignis ($wcp_{BPMNEreignisTeilprozessUnterbrechendesEreignis}$) wird durch das Eintreten des Startereignisses des Ereignis-Teilprozesses (i_3) der Ereignis-Teilprozess (t_5) nebenläufig zu den Elementen im Teilprozess ausgeführt, anstatt den Teilprozess abzubrechen. Dementsprechend kann der Ereignis-Teilprozess (t_5) mehrfach ausgeführt werden. Die Kontrollflusssemantik wird durch das Kontrollflussmuster $wcp_{BPMNEreignisTeilprozessNichtUnterbrechendesEreignis}$ abgebildet. Falls das auslösende Ereignis nicht explizit modelliert wurde, ist das Startereignis des Ereignis-Teilprozesses mit einem Trigger

[2] Erklärung siehe vorherige Fußnote. Zur korrekten Abbildung der Syntax müsste die zusammengesetzte Aufgabe aufgelöst werden, d. h. die Bedingungen und Aufgaben der zusammengesetzten Aufgabe in den Kontrollfluss des übergeordneten Prozesses eingebettet werden.

[3] Siehe vorherige Fußnote.

zu spezifizieren (vgl. Abbildung 39). Sofern das auslösende Ereignis des Startereignisses des Ereignis-Teilprozesses explizit modelliert wurde (vgl. Abbildung 40), existiert ein impliziter Sequenzfluss vom auslösenden Ereignis (t_6) zum Startereignis des Ereignis-Teilprozesses (i_3).

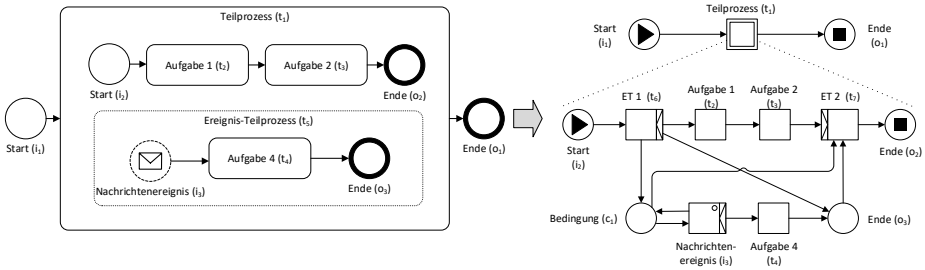


Abbildung 39: Beispiel - Ereignis-Teilprozess mit nicht-unterbrechendem Startereignis und nicht explizit modelliertem Auslöser[4]

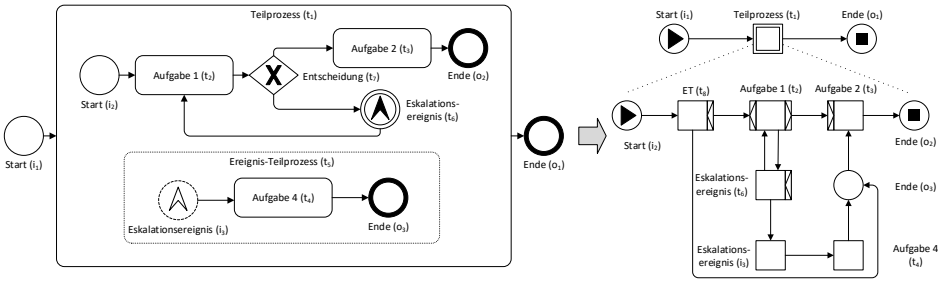


Abbildung 40: Beispiel - Ereignis-Teilprozess mit nicht-unterbrechendem Startereignis und explizit modelliertem Auslöser[5]

Die für die Modellierungssprache BPMN entwickelten Kontrollflussmuster werden zu den Kontrollflussmustern «Während der Ausführung» hinzugefügt. Abgerundet wird der Abschnitt in Abbildung 41 mit einer Übersicht der Zuordnung der Kontrollflussmuster zu den

[4] Der Ereignis-Teilprozess (t_6) wurde in der newYAWL-Darstellung zur übersichtlicheren Darstellung aufgelöst und die Bedingungen sowie die Aufgaben in den Kontrollfluss des übergeordneten Prozesses eingebettet. In dem Beispiel wird darüber hinaus aus Praktikabilitätsgründen angenommen, dass Aufgabe 4 nicht zeitgleich mehrfach ausgeführt werden kann. Andernfalls würde dies die newYAWL-Spezifikation unverhältnismäßig vergrößern, da zusätzlich geprüft werden müsste, ob der Ereignis-Teilprozess noch aktiv ist, bevor der Teilprozess beendet werden kann.

[5] Siehe vorherige Fußnote.

Kontrollflussmustern «Vor der Ausführung», «Während der Ausführung» und «Nach der Ausführung».

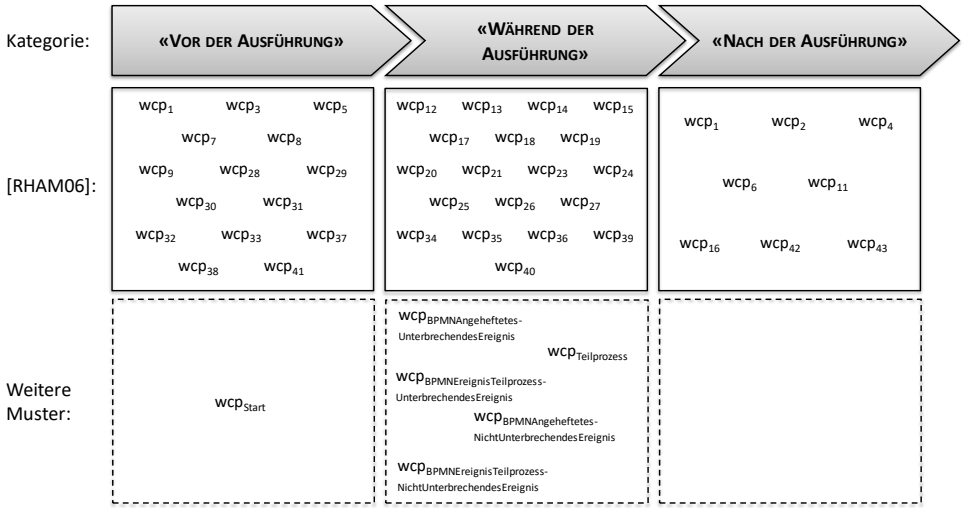


Abbildung 41: Überblick - Einordung der Kontrollflussmuster in die Mengen Kontrollflussmuster «Vor der Ausführung», «Während der Ausführung» und «Nach der Ausführung»

5.2 Vorgehen zur Beschreibung der Kontrollflusssemantik einer Geschäftsprozessmodellierungssprache

Zur Beschreibung der Kontrollflusssemantik einer Geschäftsprozessmodellierungssprache werden den Knotentypen Kontrollflussmuster (vgl. Abbildung 41) zugeordnet. Somit fungieren die Kontrollflussmuster als Bindeglied zur Beschreibung der Kontrollflusssemantik einer Geschäftsprozessmodellierungssprache. Das Vorgehen zur Beschreibung der Kontrollflusssemantik einer Geschäftsprozessmodellierungssprache wird in Abbildung 42 als BPMN-Modell dargestellt, um die Ablaufbeschreibung als Grundlage für die prototypische Realisierung des Software-Werkzeuges (vgl. Kapitel 7.2) einsetzen zu können. BPMN eignet sich aufgrund der Aufgabentypen, mit denen gekennzeichnet werden kann, welche Aufgaben vom Modellierer und welche Aufgaben von einem IT-System übernommen werden können. Zusätzlich können die BPMN-Ereignisse verwendet werden, um die ausgewählten Kontrollflussmuster für die jeweiligen Knotentypen explizit darzustellen.

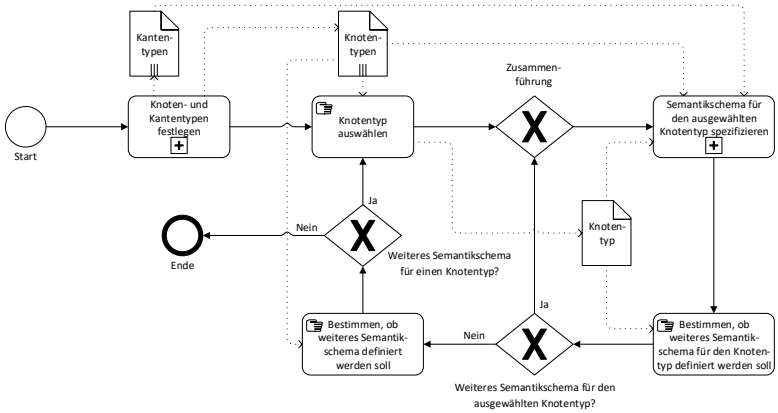


Abbildung 42: Vorgehen - Beschreibung der Kontrollflusssemantik einer Geschäftsprozessmodellierungssprache

Im ersten Schritt werden für die Beschreibung der Kontrollflusssemantik einer Geschäftsprozessmodellierungssprache die möglichen Knoten- und Kantentypen definiert (vgl. Abbildung 30). Anschließend kann einer der definierten Knotentypen ausgewählt und dessen Kontrollflusssemantik mit einem Semantikschema spezifiziert werden. Ein Semantikschema (vgl. Klasse `Semantikschema` in Abbildung 30) eines Knotentyps wird durch die

- Kontrollflusssemantik «Vor der Ausführung» (vgl. Klasse `KontrollflussemantikVorderAusführung` in Abbildung 30),
- Kontrollflusssemantik «Während der Ausführung» (vgl. Klasse `KontrollflusssemantikWährendderAusführung` in Abbildung 30),
- Kontrollflusssemantik «Nach der Ausführung» (vgl. Klasse `KontrollflussemantikNachderAusführung` in Abbildung 30) und
- Beziehung (vgl. Klasse `Beziehung` in Abbildung 30)

definiert. Nachdem die Kontrollflusssemantik für einen Knotentyp beschrieben wurde, kann ein weiteres Semantikschema für den ausgewählten Knotentyp definiert werden. Alternativ kann ein Semantikschema für einen anderen Knotentyp spezifiziert werden. Auf die detaillierten Beschreibungen, d. h. die Festlegung der Knoten- und Kantentypen (vgl. Abbildung 43) und die Spezifikation des Semantikschemas für einen Knotentyp (vgl. Abbildung 45) wird im Folgenden eingegangen.

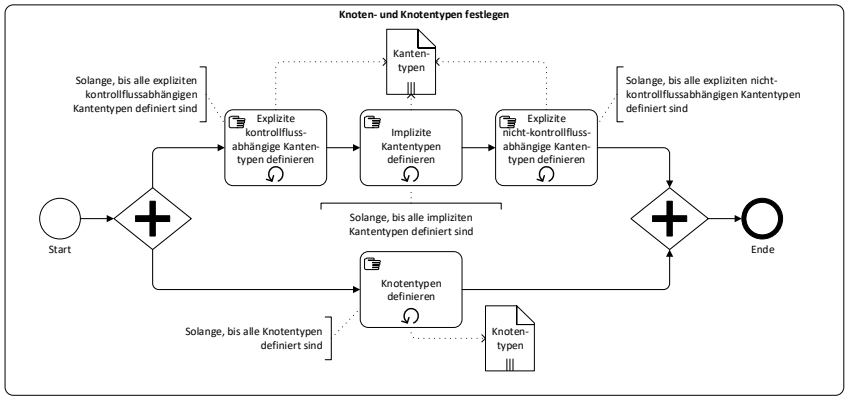


Abbildung 43: Vorgehen - Festlegung der Knoten- und Kantentypen

Der Teilprozess zur Definition der Knoten- und Kantentypen wird in Abbildung 43 illustriert. Zunächst müssen die expliziten und impliziten kontrollflussabhängigen Kantentypen sowie die expliziten nicht-kontrollflussabhängigen Kantentypen der Modellierungssprache definiert werden. Die expliziten kontrollflussabhängigen Kantentypen sind diejenigen Kantentypen, die durch visuell sichtbare Notationselemente dargestellt und zur Beschreibung des Kontrollflusses verwendet werden, wie z. B. durch Kanten bzw. Pfeile. Die impliziten Kantentypen sind ebenfalls kontrollflussabhängige Kantentypen, mit denen der Kontrollfluss zwischen Elementen im Geschäftsprozessmodell beschrieben werden kann. Im Gegensatz zu den expliziten kontrollflussabhängigen Kantentypen besitzen die impliziten kontrollflussabhängigen Kantentypen keine visuell sichtbaren Notationselemente. Die expliziten nicht-kontrollflussabhängigen Kantentypen beeinflussen den Kontrollfluss eines Elementes nicht und werden auch als Beziehungen (vgl. Klasse `Beziehungen` in Abbildung 30) bezeichnet. Beispielsweise existieren in der Modellierungssprache Business Process Model and Notation (BPMN):

– die expliziten kontrollflussabhängigen Kantentypen: Sequenzfluss (vgl. Abbildung 13a, b und c), Nachrichtenfluss (vgl. Abbildung 13d, e, f), gerichtete Assoziation (vgl. Abbildung 13h) und bidirektionale Assoziation (vgl. Abbildung 13i).

– der implizite kontrollflussabhängige Kantentyp: impliziter Sequenzfluss, der z. B. bei der Modellierung zwischen zwei Link-Ereignissen (vgl. Abbildung 44) verwendet wird.

– der explizite nicht-kontrollflussabhängige Kantentyp: Assoziation (vgl. Abbildung 13g)

Abbildung 44: Beispiel - Implizite Kante zwischen zwei Linkereignissen

Nachdem die Knoten- und Kantentypen definiert wurden, ist der Teilprozess abgeschlossen. Im nächsten Schritt kann einer der definierten Knotentypen ausgewählt und die Kontrollflusssemantik des Knotentyps durch ein Semantikschema beschrieben werden (vgl. Abbildung 45). Für das Semantikschema müssen zunächst die

- Kontrollflusssemantik «Vor der Ausführung» (vgl. Klasse `KontrollflussemantikVorDerAusführung` in Abbildung 30) und
- Kontrollflusssemantik «Nach der Ausführung» (vgl. Klasse `KontrollflussemantikNachDerAusführung` in Abbildung 30)

in Form von Vor- und Nachbedingungen (vgl. Klasse `Bedingung` in Abbildung 30) für den ausgewählten Knotentyp definiert werden. Eine Bedingung kann eine elementare Bedingung oder eine zusammengesetzte Bedingung sein. Die zusammengesetzte Bedingung besteht aus zwei oder mehreren elementaren Bedingungen. Eine elementare Bedingung ist in der Regel von

- einem kontrollflussabhängigen Kantentypen (vgl. Klasse `Kantentyp` in Abbildung 30),
- deren ein- bzw. ausgehender Kantenanzahl (vgl. Klasse `Anzahl` in Abbildung 30) sowie
- dem Gültigkeitsbereich (vgl. Klasse `Gültigkeitsbereich` in Abbildung 30)

abhängig (vgl. BPMN-Modell in Abbildung 46). Der Gültigkeitsbereich beschreibt

- die möglichen Knotentypen im Vor- bzw. Nachbereich (vgl. Klasse `Knotentyp` in Abbildung 30)
- mit deren Häufigkeit (vgl. Klasse `Anzahl` in Abbildung 30).

Darüber hinaus werden einer elementaren Bedingung einer beliebigen Anzahl von Kontrollflussmustern (vgl. Klasse `Kontrollflussmuster` in Abbildung 30) zur Beschreibung der Kontrollflusssemantik zugeordnet. Es müssen solange Bedingungen für ein Semantikschema entwickelt werden, bis alle notwendigen Vor- und Nachbedingungen definiert sind. Sofern mehrere Vor- bzw. Nachbedingungen definiert wurden, müssen im Anschluss daran jeweils alle Vorbedingungen und alle Nachbedingungen miteinander verknüpft werden (vgl. Klasse `ZusammengesetzteBedingung` in Abbildung 30). In diesem Fall entsteht eine

zusammengesetzte Bedingung bzw. ein logischer Ausdruck für die Vorbedingung und eine zusammengesetzte Bedingung bzw. ein logischer Ausdruck für die Nachbedingung des Knotentyps (vgl. Abbildung 54). Andernfalls existiert für das Semantikschema eine elementare Vorbedingung bzw. eine elementare Nachbedingung. Nachfolgend können die Beziehungen im Semantikschema (vgl. Klasse `Beziehungen` Abbildung 30) mit der

- Kantenanzahl (vgl. Klasse `Anzahl` in Abbildung 30) und
- einem nicht-kontrollflussabhängigen Kantentypen (vgl. Klasse `Kantentyp` in Abbildung 30)

definiert werden (vgl. Abbildung 30). Im nächsten Schritt wird die Kontrollflusssemantik «Während der Ausführung» (vgl. Klasse `KontrollflusssemantikWährendDerAusführung` in Abbildung 30) durch die Kontrollflussmuster «Während der Ausführung» (vgl. Klasse `KontrollflussmusterWährendDerAusführung` in Abbildung 30) beschrieben (vgl. BPMN-Modell in Abbildung 30), sofern für die Beschreibung der Kontrollflusssemantik des Knotentyps die Kontrollflussmuster «Während der Ausführung» erforderlich sind. Andernfalls werden dem Knotentyp keine Kontrollflussmuster «Während der Ausführung» zugeordnet und für die Fortsetzung des Kontrollflusses ist unmittelbar die Nachbedingung verantwortlich. Typischerweise werden nur Aktivitäten mit den Kontrollflussmustern «Während der Ausführung» spezifiziert, da in der Regel Ereignisse, wie auch Verzweigungs- und Zusammenführungsknoten kein zu spezifizierendes Verhalten während der Ausführung besitzen. Abschließend wird das Semantikschema mit dem Knotentyp gespeichert (vgl. Klasse `Semantikzuordnung` in Abbildung 30). Beispielobjekte der Klasse Semantikzuordnung werden in Kapitel 5.3 beschrieben, z. B. für BPMN in Tabelle 13. Im Folgenden werden die einzelnen Teilprozesse aus Abbildung 45 zur Beschreibung der Kontrollflusssemantik eines Knotentyps betrachtet, d. h. die Festlegung der Kontrollflusssemantik der Vor- bzw. Nachbedingung (vgl. Abbildung 46), die Verknüpfung der elementaren Bedingungen (vgl. Abbildung 54), die zu definierenden Beziehungen (vgl. Abbildung 55) sowie die Festlegung der Kontrollflussmuster «Während der Ausführung» (vgl. Abbildung 56).

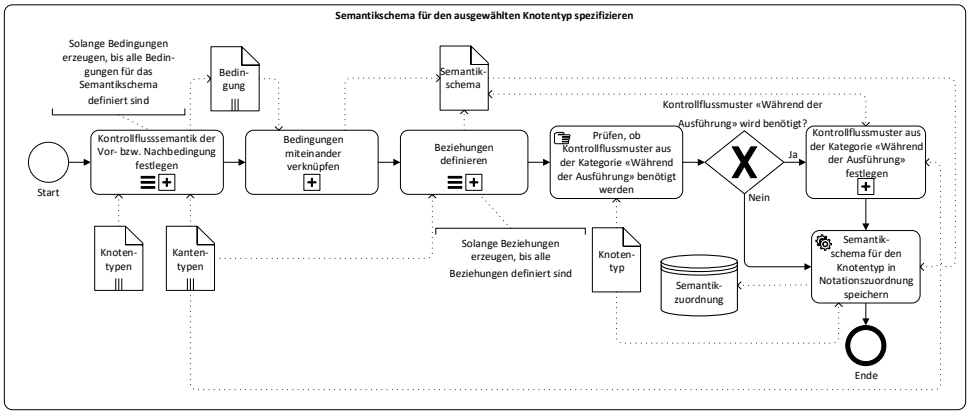


Abbildung 45: Vorgehen - Semantikschema für den ausgewählten Knotentyp spezifizieren

In Abbildung 46 wird der Teilprozess zur Beschreibung der Kontrollflusssemantik für die Vor- bzw. Nachbedingung beschrieben. Dabei müssen solange Instanzen vom Teilprozess erzeugt werden, wie Bedingungen benötigt werden. Für eine Instanz des Teilprozesses wird zunächst die Art der Bedingung festgelegt, d. h., ob eine Vorbedingung ohne Startmuster, Nachbedingung ohne Endmuster, Startmuster (Vorbedingung) oder Endmuster (Nachbedingung) formuliert werden soll. Die Kontrollflussmuster «Vor der Ausführung» können in *Startmuster* und *Kontrollflussmuster ohne Startmuster* untergliedert werden. Ein Startmuster besitzt keine eingehenden Kanten und ist somit auch von keinem Kantentyp abhängig. Es existiert auch kein Gültigkeitsbereich, da keine Knotentypen im Vorbereich existieren. Analog kann die Nachbedingung untergliedert werden, mit dem Unterschied, dass ein Endmuster keine ausgehenden Kanten und somit auch keine Knotentypen im Nachbereich besitzt. Bei der Auswahl eines Start- (vgl. Abbildung 47) oder Endmusters (vgl. Abbildung 48) muss demnach nur ein entsprechendes Muster ausgewählt werden. Sofern eine Nachbedingung ohne Endmuster ausgewählt wird, muss zunächst ein Kantentyp und die Anzahl der ausgehenden Kanten festgelegt sowie die Menge der möglichen Kontrollflussmuster ohne Endmuster aus der Menge der Kontrollflussmuster «Nach der Ausführung» ausgewählt werden. Abschließend wird der Gültigkeitsbereich der Bedingung festgelegt, indem die Bedingung auf bestimmte Knotentypen und deren Häufigkeit im Nachbereich für den Kantentyp eingeschränkt wird. Analog kann eine Vorbedingung formuliert werden, wobei ein Kontrollflussmuster aus der Menge Kontrollflussmuster «Vor der Ausführung» ohne Startmuster ausgewählt wird sowie die Knotentypen für den Vorbereich im Gültigkeitsbereich betrachtet werden (vgl. Abbildung 46).

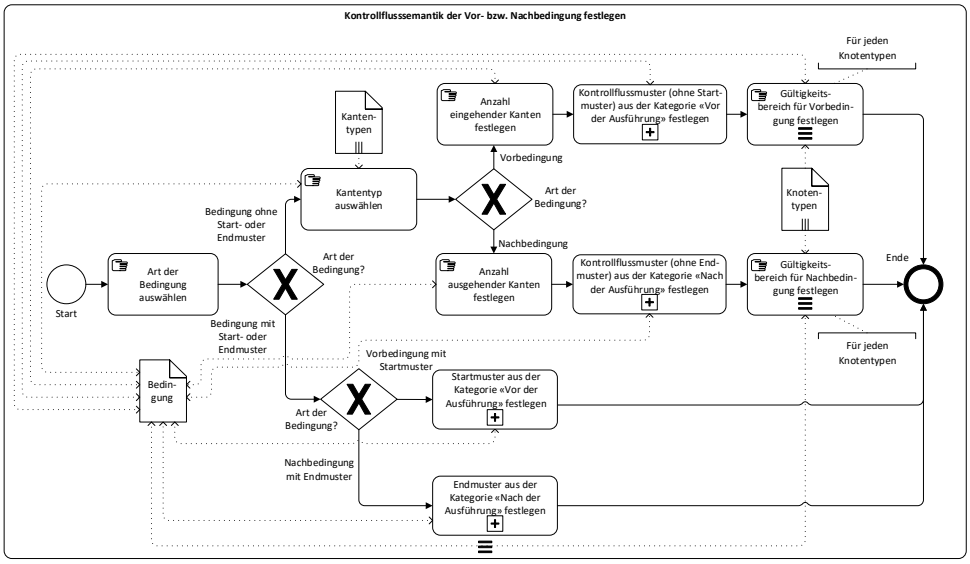


Abbildung 46: Vorgehen - Kontrollflusssemantik der Vor- bzw. Nachbedingung festlegen

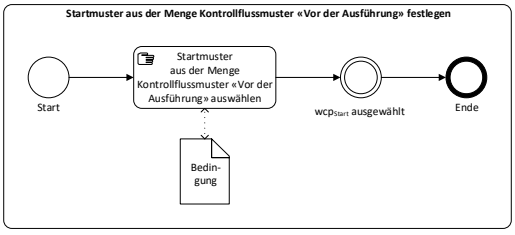


Abbildung 47: Vorgehen - Startmuster aus der Menge Kontrollflussmuster «Vor der Ausführung» festlegen

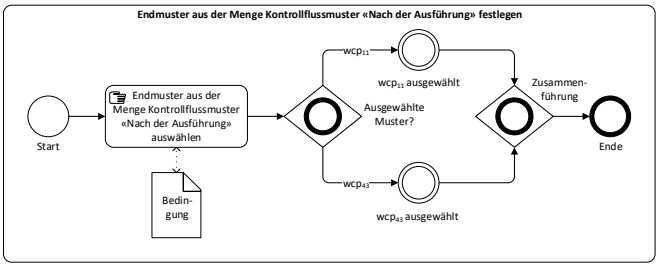


Abbildung 48: Vorgehen - Endmuster aus der Menge Kontrollflussmuster «Nach der Ausführung» festlegen

Eine elementare Bedingung (vgl. Abbildung 30) wird durch die ein- bzw. ausgehende Kantenanzahl, die Kontrollflussmuster und durch den Gültigkeitsbereich beschrieben. Die Kantenanzahl kann die Kontrollflusssemantik bzw. die Wahl der Kontrollflussmuster beeinflussen, wie beispielsweise an den BPMN-Aufgaben aus Abbildung 49a und Abbildung 49b deutlich wird. Dementsprechend kann die Kontrollflusssemantik einer BPMN-Aufgabe mit einem ausgehenden Sequenzfluss, mit dem Kontrollflussmuster Sequenz (vgl. Abbildung 49a) sowie bei mehr als einem ausgehenden Sequenzfluss, mit dem Kontrollflussmuster parallele Aufspaltung (vgl. Abbildung 49b), beschrieben werden. Analog können Beispiele für die Abhängigkeit der Anzahl der eingehenden Kanten beschrieben werden.

Abbildung 49: Beispiel - BPMN-Aufgabe - Kontrollflusssemantik

Für die Beschreibung der Anzahl der möglichen ein- bzw. ausgehenden Kanten kann eine Grammatik in der Erweiterten Backus-Naur-Form (EBNF) [Hedt12] angegeben werden. EBNF ist durch den Standard ISO 14977 definiert [ISO14977]. Mit der angegebenen EBNF-Grammatik kann die Kantenanzahl eine Zahl, ein Intervall oder auch eine beliebige Kombination aus Zahlen und Intervallen, aus der Menge der natürlichen Zahlen $\mathbb{N}_0$ sein. Sofern die Kantenanzahl beliebig groß sein darf, wird dies mit unendlich (∞) gekennzeichnet.

```
Anzahl = (Zahl | Intervall);
Intervall = ("(" | "["), Zahl, "...", Zahl, (")" | "]");
Zahl = (Ziffer, { Ziffer}) | "∞";
Ziffer = [0-9];
```

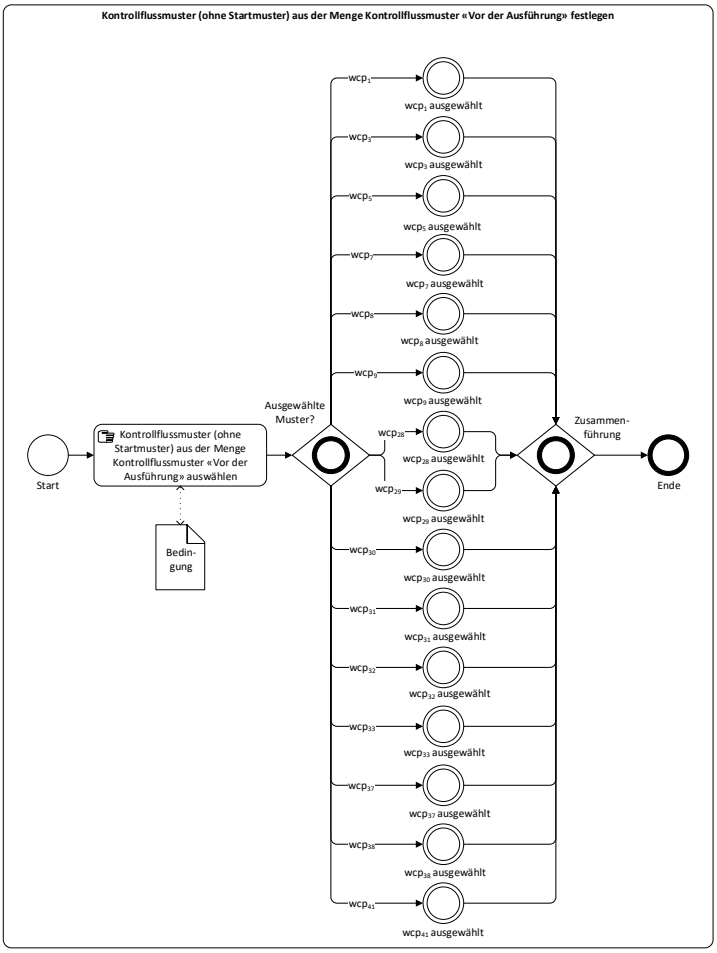


Abbildung 50: Vorgehen - Kontrollflussmuster (ohne Startmuster) aus der Menge Kontrollflussmuster «Vor der Ausführung» festlegen

Nachdem die Anzahl der ein- bzw. ausgehenden Kanten eines Kantentyps für eine Bedingung festgelegt wurde, können beliebig viele Kontrollflussmuster ohne Start- bzw. Endmuster aus der Menge Kontrollflussmuster «Vor der Ausführung» (vgl. Abbildung 50) bzw. «Nach der Ausführung» (vgl. Abbildung 51) ausgewählt werden. Beispielsweise können dem Knotentyp OR-Verknüpfungsoperator einer EPK (vgl. Abbildung 16c) bei [2…∞] eingehenden Kontrollflüssen die Kontrollflussmuster einfache Zusammenführung (wcp_5), strukturierte synchronisierte Zusammenführung (wcp_7), Mehrfachzusammenführung (wcp_8), strukturierter Diskriminator (wcp_9), lokale synchronisierte Zusammenführung (wcp_{37}) und

generelle synchronisierte Zusammenführung (wcp_{38}) aus der Menge Kontrollflussmuster «Vor der Ausführung» zugeordnet werden.

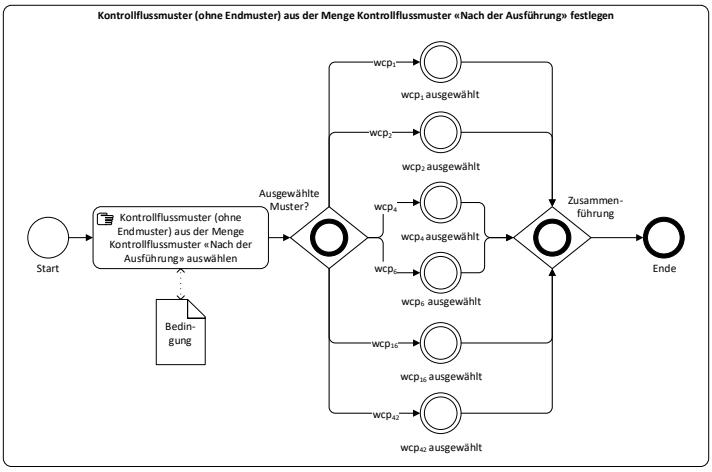


Abbildung 51: Vorgehen - Kontrollflussmuster (ohne Endmuster) aus der Menge Kontrollflussmuster «Nach der Ausführung» festlegen

Im nächsten Schritt kann der Gültigkeitsbereich (vgl. Klasse `Gültigkeitsbereich` in Abbildung 30) der spezifizierten Bedingung durch ausgewählte Knotentypen im Vor- bzw. Nachbereich und deren Häufigkeit eingeschränkt werden. Dementsprechend wird für jeden Knotentyp festgelegt, ob dieser im Vor- bzw. Nachbereich mit dem ausgewählten Kantentyp vorkommen kann und in welcher Häufigkeit der Knotentyp auftreten darf. Für die Festlegung der Häufigkeit der jeweiligen Knotentypen im Vor- bzw. Nachbereich kann die obige EBNF-Grammatik `Anzahl` verwendet werden. In Abbildung 52 wird ein Beispiel angegeben, bei dem die Kontrollflusssemantik des Knotentyps von den Knotentypen im Vorbereich abhängig ist. Es wird angenommen, dass eine Modellierungssprache existiert, die nur die zwei Knotentypen Kreis und Viereck sowie den Kantentyp Sequenzfluss besitzt. Die Kontrollflusssemantik «Vor der Ausführung» wird durch das Kontrollflussmuster Synchronisation (wcp_3) aus der Menge Kontrollflussmuster «Vor der Ausführung» definiert, sofern der Vorbereich mindestens einen Kreis und ein Viereck aufweist, d. h. $\text{Kreis} = [1 \dots \infty]$ und $\text{Viereck} = [1 \dots \infty]$. Für den Fall, dass sich nur Vierecke im Vorbereich befinden, gilt das Kontrollflussmuster Mehrfachzusammenführung (wcp_8) für die Kontrollflusssemantik «Vor der Ausführung», d. h. $\text{Kreis} = 0$ und $\text{Viereck} = [1 \dots \infty]$.

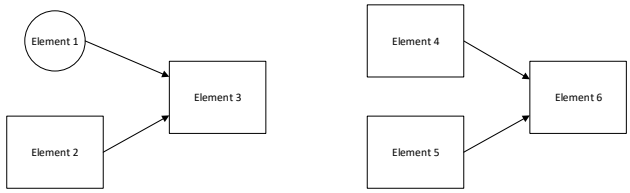


Abbildung 52: Beispiel - Kontrollflusssemantik abhängig von den Knotentypen im Vorbereich

Nachdem die Vor- und Nachbedingungen für den ausgewählten Knotentyp definiert wurden, müssen im nächsten Schritt die Bedingungen für das Semantikschema jeweils miteinander verknüpft werden (vgl. Klasse `ZusammengesetzteBedingung` in Abbildung 30 und BPMN-Modell in Abbildung 54), sofern zwei oder mehrere Bedingungen definiert wurden. Für die Verknüpfung können die logischen Grundoperatoren Konjunktion (`"AND"`), Disjunktion (`"OR"`) und Negation (`"NOT"`) sowie die zusammengesetzten Operatoren negiertes Und (`"NAND"`), negiertes Oder (`"NOR"`), exklusives Oder bzw. Antivalenz (`"XOR"`) und exklusives Nicht Oder bzw. Äquivalenz (`"XNOR"`) verwendet werden. Zusätzlich können die Bedingungen entsprechend der üblichen Regeln zur Bildung logischer Ausdrücke verknüpft werden. Die Verknüpfung der Bedingungen wird mit der EBNF-Grammatik `ZusammengesetzteBedingung` beschrieben. $\texttt{Bedingung}_1$ bis $\texttt{Bedingung}_n$ sind die erstellten Bedingungen vom Typ Vor- bzw. Nachbedingung.

```
ZusammengesetzteBedingung = ("(", Bedingung, [Operator,
                            Bedingung], {Operator, Zusammenge-
                            setzteBedingung}, ")"), {Operator, Zu-
                            sammengesetzteBedingung};
Bedingung = Bedingung1 | Bedingung2 | … | Bedingungn;
Operator = "AND" | "OR" | "NOT" | "NAND" | "NOR" | "XOR" |
                            "XNOR";
```

Im Folgenden wird die EBNF-Grammatik `ZusammengesetzteBedingung` auf die BPMN-Aufgabe in Abbildung 53 angewendet, bei der die Vorbedingungen miteinander verknüpft werden sollen. Die BPMN-Aufgabe ist mit den folgenden Vorbedingungen spezifiziert, wobei der Gültigkeitsbereich vernachlässigt wird:

- Vorbedingung mit Kantentyp Sequenzfluss (Bed1): Eine BPMN-Aufgabe mit [2…∞] eingehenden Sequenzflüssen kann mit den Kontrollflussmustern einfache Zusammenführung (wcp_5) oder Mehrfachzusammenführung (wcp_8) spezifiziert werden.

- Vorbedingung mit Kantentyp Nachrichtenfluss (Bed2): Eine BPMN-Aufgabe mit [2…∞] eingehenden Nachrichtenflüssen kann mit den Kontrollflussmustern

einfache Zusammenführung (wcp_5) oder Mehrfachzusammenführung (wcp_8) spezifiziert werden.

– Vorbedingung mit gerichteter Assoziation (Bed3): Eine BPMN-Aufgabe mit [2…∞] eingehenden gerichteten Assoziationen kann mit dem Kontrollflussmuster Synchronisation (wcp_3) spezifiziert werden.

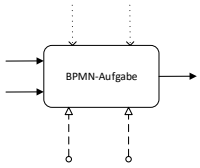


Abbildung 53: Beispiel - Zusammengesetzte Bedingung

Die Anwendung der EBNF-Grammatik auf das Beispiel in Abbildung 53 ergibt mit den obigen Bedingungen die zusammengesetzte Bedingung:

```
(Bed1 AND Bed2 AND Bed3).
```

Die Bedingungen enthalten wiederum die Kontrollflussmuster, so dass zur Erfüllung der Vorbedingung der BPMN-Aufgabe Bed1, Bed2 und Bed3 erfüllt sein muss. Bed1 und Bed2 wird unter anderem durch die Kontrollflussmuster einfache Zusammenführung (wcp_5) und Mehrfachzusammenführung (wcp_8) definiert sowie Bed3 durch das Kontrollflussmuster Synchronisation (wcp_3). Mit den Kontrollflussmustern einer Bedingung wird die Kontrollflusssemantik der Bedingung und somit des Elementes im Geschäftsprozessmodell beschrieben. Aus den möglichen Kontrollflussmustern zur Beschreibung der Kontrollflusssemantik muss genau ein Kontrollflussmuster für jede Bedingung ausgewählt werden, damit die Kontrollflusssemantik der Bedingung für das Element im Geschäftsprozessmodell eindeutig beschrieben ist (vgl. Kapitel 6). Für Bed1 und Bed2 wird für die BPMN-Aufgabe aus Abbildung 53 das Kontrollflussmuster einfache Zusammenführung (wcp_5) ausgewählt. Dementsprechend ist die Vorbedingung der BPMN-Aufgabe erfüllt, wenn jeweils ein beliebiges Element im Vorbereich ausgeführt wurde, dass mit einem Sequenzfluss und Nachrichtenfluss mit der BPMN-Aufgabe verbunden ist. Zusätzlich müssen alle Elemente im Vorbereich ausgeführt sein, die über eine gerichtete Assoziation mit der BPMN-Aufgabe verbunden sind. Die Auswahl der Kontrollflussmuster für Bedingungen und somit für die Elemente im Geschäftsprozessmodell wird in Kapitel 6 betrachtet. Analog können Nachbedingungen miteinander verknüpft werden. In Abbildung 54 existiert noch ein weiterer Fall, der eintritt, falls keine Bedingungen vom Typ Vor- bzw. Nachbedingung für den Knotentyp

definiert wurden, sodass ein Semantikschema entweder ohne Vor- bzw. Nachbedingung oder nur mit den elementaren Vor- bzw. Nachbedingungen erzeugt wird.

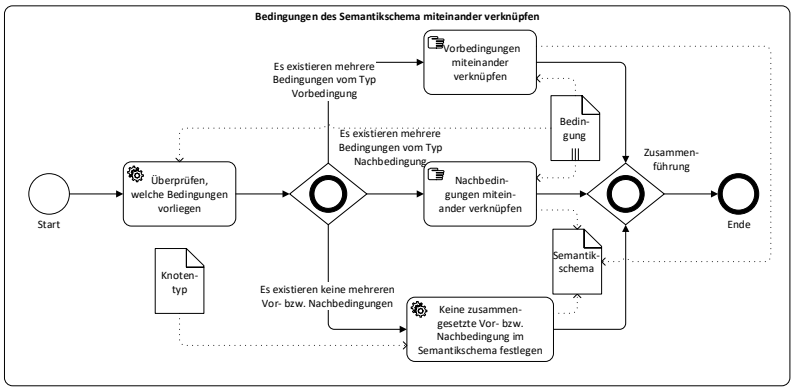


Abbildung 54: Vorgehen - Bedingungen des Semantikschema miteinander verknüpfen

Im nächsten Schritt können die Beziehungen mit den nicht-kontrollflussabhängigen Kantentypen festgelegt werden (vgl. Klasse `Beziehung` in Abbildung 30). Die Anzahl dieser Kantentypen (vgl. Klasse `Anzahl` in Abbildung 30) wird ebenfalls durch die obige EBNF-Grammatik `Anzahl` beschrieben. Beziehungen werden beispielsweise bei der Modellierungssprache newYAWL benötigt, die die Verbindung mit einem Abbruch-, Blockierungs- und/oder Beendigungsbereich beschreiben.

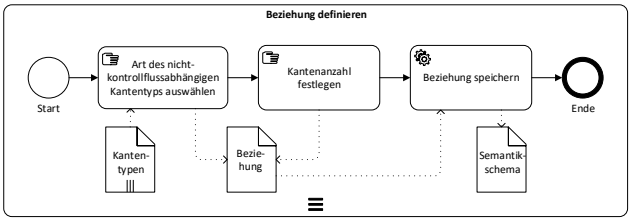


Abbildung 55: Vorgehen - Beziehung definieren

Nachfolgend können gegebenenfalls Muster aus der Menge Kontrollflussmuster «Während der Ausführung» dem Semantikschema für den Knotentyp zugeordnet werden, sofern diese Kontrollflussmuster zur Beschreibung der Kontrollflusssemantik erforderlich sind. Die möglichen Muster aus der Menge der Kontrollflussmuster «Während der Ausführung» sind in Teilmengen zusammengefasst, da einem Knotentyp mehrere Kontrollflussmuster

zugewiesen werden können. Einem Element im Geschäftsprozessmodell können aber nicht mehrere Kontrollflussmuster einer Teilmenge zugeordnet werden. Beispielsweise können einem Element nicht mehrere Kontrollflussmuster aus der Menge Mehrfachinstanz zugewiesen werden. Ein weiteres Beispiel sind die Abbruchmuster, so dass ein Element in einem Geschäftsprozessmodell nicht gleichzeitig genau anderes Element (wcp_{19}) und auch mehrere andere Elemente abbrechen (wcp_{25}) kann. Die definierten Teilmengen sind: $WCP_{Mehrfachinstanz}$, $WCP_{Abbruch}$, $WCP_{Trigger}$, wcp_{18}, wcp_{21}, wcp_{27}, wcp_{39}, $WCP_{VerschachtelteAusführung}$ und $WCP_{BenutzerdefinierteMuster}$ (vgl. Abbildung 56), mit

- $WCP_{Mehrfachinstanz} = \{wcp_{12}, wcp_{13}, wcp_{14}, wcp_{15}, wcp_{34}, wcp_{35}, wcp_{36}\}$ (vgl. Abbildung 57),
- $WCP_{Abbruch} = \{wcp_{19}, wcp_{20}, wcp_{25}, wcp_{26}\}$ (vgl. Abbildung 58),
- $WCP_{Trigger} = \{wcp_{23}, wcp_{24}\}$ (vgl. Abbildung 59),
- $WCP_{VerschachtelteAusführung} = \{wcp_{17}, wcp_{40}\}$ (vgl. Abbildung 60) und
- $WCP_{BenutzerdefinierteMuster} = \{wcp_{BPMNAngeheftetesUnterbrechendesEreignis}, wcp_{BPMNAngeheftetesNichtUnterbrechendesEreignis}, wcp_{Teilprozess}, wcp_{BPMNEreignisTeilprozessUnterbrechendesEreignis}, wcp_{BPMNEreignisTeilprozessNichtUnterbrechendesEreignis}\}$ (vgl. Abbildung 61),

wie bereits in Kapitel 5.1 sowie in Abbildung 41 beschrieben wurde. Im folgenden Abschnitt wird die Beschreibung der Kontrollflusssemantik einer Geschäftsprozessmodellierungssprache auf die Modellierungssprachen aus Kapitel 3 angewendet.

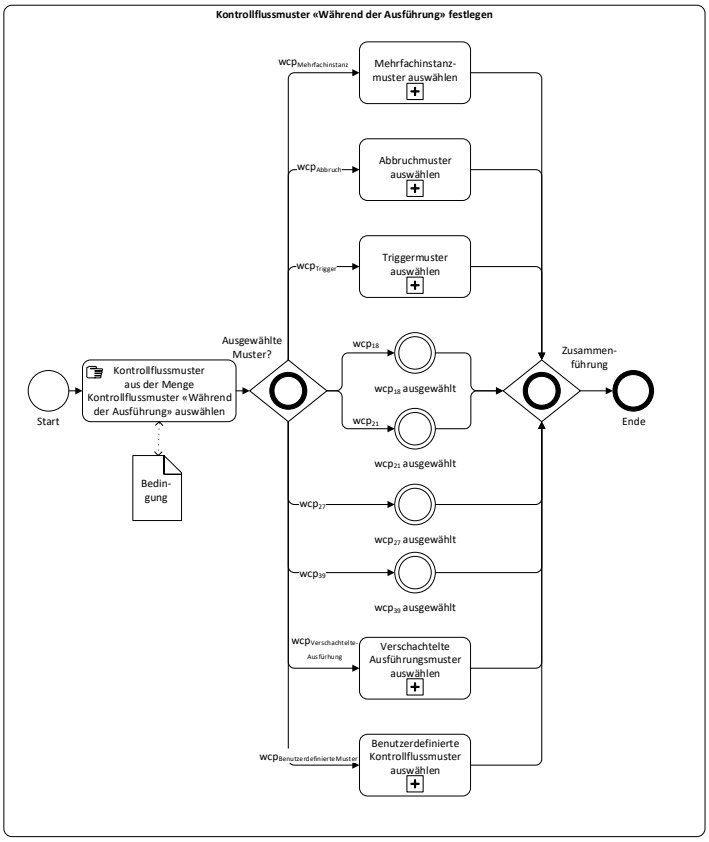


Abbildung 56: Vorgehen - Kontrollflussmuster «Während der Ausführung» festlegen

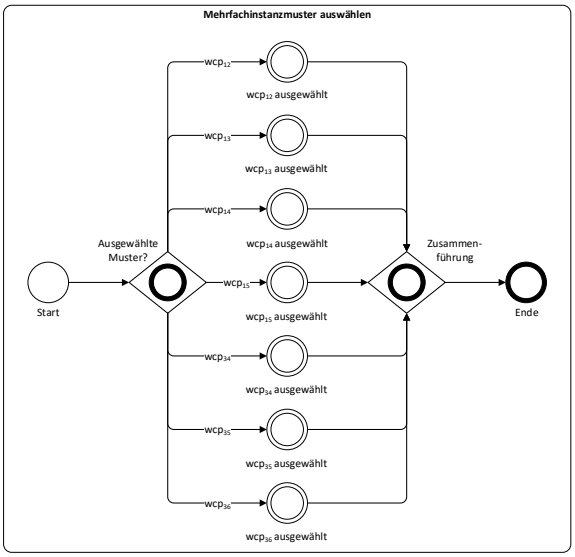


Abbildung 57: Vorgehen - Mehrfachinstanzmuster auswählen

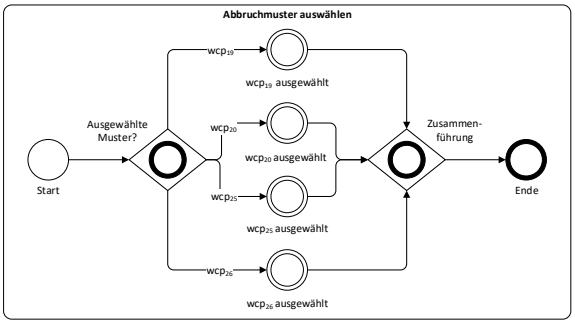


Abbildung 58: Vorgehen - Abbruchmuster auswählen

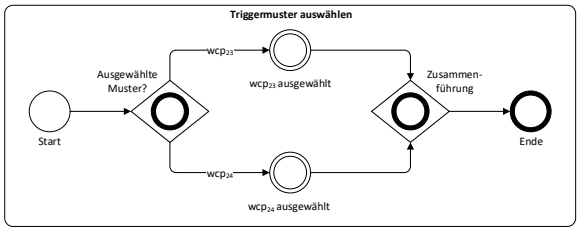


Abbildung 59: Vorgehen - Triggermuster auswählen

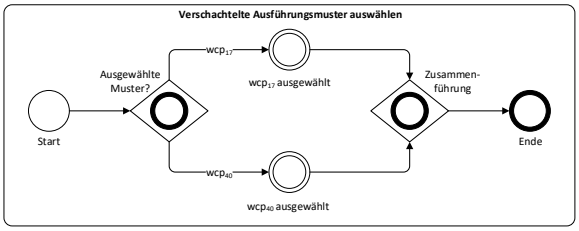


Abbildung 60: Vorgehen - Verschachtelte Ausführungsmuster auswählen

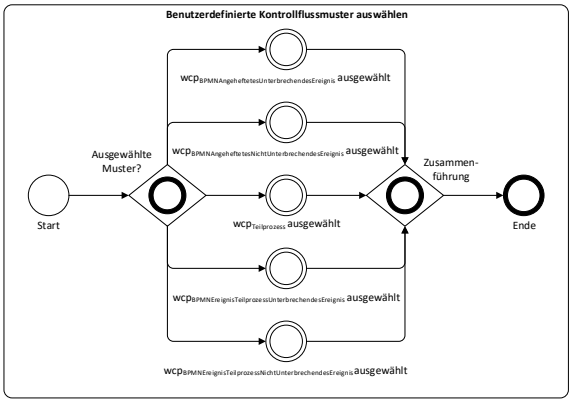


Abbildung 61: Vorgehen - Benutzerdefinierte Kontrollflussmuster auswählen

5.3 Beispielhafte Anwendung des Vorgehens

Im Folgenden wird die Methode zur Beschreibung der Kontrollflusssemantik von Geschäftsprozessmodellierungssprachen auf die Modellierungssprachen Business Process Model and Notation (BPMN) [BPMN2.0.2], Ereignisgesteuerte Prozesskette (EPK) [KeNS92], Petri-Netz [Petr62] UML-Aktivitätsdiagramm [UML2.5.1] und Yet Another Workflow Language (YAWL) bzw. new Yet Another Workflow Language (newYAWL) [RuHE07] aus Kapitel 3 angewendet. Dafür muss die Kontrollflusssemantik der Geschäftsprozessmodellierungssprachen durch eine geeignete Kombination von Mustern aus den Mengen Kontrollflussmuster «Vor der Ausführung», «Während der Ausführung» und «Nach der Ausführung» beschrieben werden können (vgl. zweite Anforderung in Kapitel 5.1). Diese Standardmodellierungssprachen bieten eine Vielzahl unterschiedlicher Varianten von Knotentypen und Kantentypen zur Beschreibung der Kontrollflusssemantik. Dennoch kann nicht gewährleistet werden, dass für alle existierenden Geschäftsprozessmodellierungssprachen die

Kontrollflusssemantik mit der entwickelten Methode beschrieben werden kann. Sofern die Kontrollflusssemantik nicht mit den vorgegebenen Kontrollflussmustern abgebildet werden kann, müssen benutzerdefinierte Kontrollflussmuster entwickelt werden, wie beispielsweise für die angehefteten Ereignisse an Aktivitäten oder die Ereignis-Teilprozesse in BPMN demonstriert wurde (vgl. Kapitel 5.1).

5.3.1 Business Process Model and Notation (BPMN)

Die Business Process Model and Notation wurde in Kapitel 3.2 vorgestellt. Die Knotentypen der Modellierungssprache werden in Tabelle 12 beschrieben und deren Notationselemente in Abbildung 8, 9, 11, 12, 14, 15 und Tabelle 2 dargestellt. Die Kantentypen gehen aus Tabelle 9 und Tabelle 10 hervor sowie deren Notationselemente aus Abbildung 13. Die Eingruppierung der Kantentypen in expliziter kontrollflussabhängiger, impliziter kontrollflussabhängiger und expliziter nicht-kontrollflussabhängiger Kantentyp wurde bereits in Kapitel 5.2 vorgenommen.

Die Kontrollflusssemantik der einzelnen Knotentypen wird im Semantikschema (vgl. Abbildung 30) durch die

- Kontrollflusssemantik «Vor der Ausführung»,
- Kontrollflusssemantik «Während der Ausführung» und
- Kontrollflusssemantik «Nach der Ausführung» sowie
- einer beliebigen Anzahl von Beziehungen

beschrieben. Beziehungen wurden durch einen Kantentyp und eine Kantenanzahl definiert. In BPMN existiert nur eine Beziehung mit dem Kantentyp Assoziation, die zu den expliziten nicht-kontrollflussabhängigen Kantentypen zählt und mit der Kantenanzahl $[0 \ldots \infty]$, wie in Tabelle 9 definiert wird.

BeziehungsID	Kantentyp	Anzahl
Bez1	Assoziation	$[0 \ldots \infty]$

Tabelle 9: Instanz - BPMN - Klasse Beziehung

Die Kontrollflusssemantik «Vor der Ausführung» und «Nach der Ausführung» wird für ein Semantikschema jeweils durch eine elementare oder zusammengesetzte Bedingung beschrieben (vgl. Abbildung 30). Die zusammengesetzte Bedingung besteht aus zwei oder mehreren elementaren Bedingungen, die über die EBNF-Grammatik `ZusammengesetzteBedingung` miteinander verknüpft sind. Eine elementare Bedingung wird durch einen Kantentyp, einen Gültigkeitsbereich, die ein- bzw. ausgehende Kantenanzahl und

Mustern aus der Menge Kontrollflussmuster «Vor der Ausführung» bzw. «Nach der Ausführung» beschrieben. Für die Beschreibung der elementaren Bedingungen sind keine Gültigkeitsbereiche erforderlich, so dass diese im Folgenden weggelassen werden. Die Instanzen der Klasse der elementare Bedingung werden in Tabelle 10 und die Instanzen der Klasse der zusammengesetzte Bedingung in Tabelle 11 definiert.

BedingungsID	Kantentyp	Anzahl	Kontrollflussmuster
Bed1	Kein Kantentyp	Keine Anzahl	wcp_{Start}
Bed2	Kein Kantentyp	Keine Anzahl	wcp_{11}
Bed3	Kein Kantentyp	Keine Anzahl	wcp_{43}
Bed4	Sequenzfluss	1	wcp_{1}
Bed5	Sequenzfluss	[2 … ∞]	wcp_{2}
Bed6	Sequenzfluss	[2 … ∞]	wcp_{3}, wcp_{33}, wcp_{37}, wcp_{38}
Bed7	Sequenzfluss	[2 … ∞]	wcp_{4}
Bed8	Sequenzfluss	[2 … ∞]	wcp_{5}, wcp_{8}
Bed9	Sequenzfluss	[2 … ∞]	wcp_{6}
Bed10	Sequenzfluss	[2 … ∞]	wcp_{7}, wcp_{37}, wcp_{38}
Bed11	Sequenzfluss	[2 … ∞]	wcp_{9}, wcp_{28}, wcp_{30}, wcp_{31}
Bed12	Sequenzfluss	[2 … ∞]	wcp_{16}
Bed13	Nachrichtenfluss	1	wcp_{1}
Bed14	Nachrichtenfluss	[2 … ∞]	wcp_{2}
Bed15	Nachrichtenfluss	[2 … ∞]	wcp_{5}, wcp_{8}
Bed16	Gerichtete Assoziation	1	wcp_{1}
Bed17	Gerichtete Assoziation	[2 … ∞]	wcp_{2}
Bed18	Gerichtete Assoziation	[2 … ∞]	wcp_{3}, wcp_{33}, wcp_{37}, wcp_{38}
Bed19	Gerichtete Assoziation	[2 … ∞]	wcp_{5}, wcp_{8}
Bed20	Gerichtete Assoziation	[2 … ∞]	wcp_{16}
Bed21	Bidirektionale Assoziation	1	wcp_{1}
Bed22	Bidirektionale Assoziation	[2 … ∞]	wcp_{2}
Bed23	Bidirektionale Assoziation	[2 … ∞]	wcp_{3}, wcp_{33}, wcp_{37}, wcp_{38}
Bed24	Bidirektionale Assoziation	[2 … ∞]	wcp_{5}, wcp_{8}
Bed25	Bidirektionale Assoziation	[2 … ∞]	wcp_{16}
Bed26	Impliziter Sequenzfluss	1	wcp_{1}
Bed27	Impliziter Sequenzfluss	[2 … ∞]	wcp_{2}
Bed28	Impliziter Sequenzfluss	[2 … ∞]	wcp_{5}, wcp_{8}

Tabelle 10: Instanzen - BPMN - Klasse elementare Bedingung

Die zusammengesetzten Bedingungen werden aufgrund der Vielzahl in aggregierter Form in Tabelle 11 als EBNF-Grammatik definiert und implizieren ausnahmsweise aus Praktikabilitätsgründen auch die nicht zusammengesetzten Bedingungen. Beispielsweise besteht die Bedingung 29: “(“(Bed1 | Bed4 | Bed8) [“AND“ (Bed13 | Bed15)] [“AND“ (Bed16 | Bed18)] [“AND“ (Bed21 | Bed23)]“)“ aus 3 elementaren Bedingungen und 78 zusammengesetzten Bedingungen.

BedingungsID	Aggregierte zusammengesetzte Bedingung
Bed29	“(“(Bed1 | Bed4 | Bed8) [“AND“ (Bed13 | Bed15)] [“AND“ (Bed16 | Bed18)] [“AND“ (Bed21 | Bed23)“)“
Bed30	“(“(Bed2 | Bed4 | Bed5) [“AND“ (Bed13 | Bed14)] [“AND“ (Bed16 | Bed17)] [“AND“ (Bed21 | Bed22)“)“
Bed31	“(“(Bed4 | Bed8) “AND“ (Bed13 | Bed15)] “)“
Bed32	“(“(Bed4 | Bed5) “AND“ (Bed13 | Bed14)] “)“
Bed33	“(“(Bed4 | Bed5) “AND“ (Bed26 | Bed27)] “)“
Bed34	“(“(Bed16 | Bed19) [“AND“(Bed21 | Bed24)] “)“
Bed35	“(“(Bed16 | Bed20) [“AND“ (Bed21 | Bed25)] “)“

Tabelle 11: Instanzen - BPMN - Klasse zusammengesetzte Bedingung

Nachdem die Bestandteile des Semantikschemas definiert wurden, d. h. die Instanzen der Klasse Beziehung (vgl. Tabelle 9) und die Instanzen der Klasse Bedingung (vgl. Tabelle 10 und Tabelle 11), wird das Semantikschema in Tabelle 12 für BPMN beschrieben. Jede Instanz der Klasse Semantikschema in BPMN besitzt die Beziehung Bez1 (vgl. Tabelle 9), welche aus Gründen der Übersichtlichkeit in Tabelle 12 jedoch ausgeblendet wurde. Abschließend werden die Semantikschemas den Knotentypen in der Klasse Semantikzuordnung (vgl. Klasse `Semantikzuordnung` in Abbildung 30) zugeordnet (vgl. Tabelle 13). Mit dieser Zuordnung ist die Kontrollflusssemantik der vorgestellten BPMN-Variante aus Kapitel 3.2 beschrieben, wobei an dieser Stelle auf eine Vollständigkeitsüberprüfung verzichtet wird.

SID	Kontrollflusssemantik «Vor der Ausführung»	Kontrollflusssemantik «Während der Ausführung»	Kontrollflusssemantik «Nach der Ausführung»
S1	Bed1	Keine Kontrollflussmuster	Bed4
S2	Bed1	Keine Kontrollflussmuster	Bed5
S3	Bed1	Keine Kontrollflussmuster	Bed12
S4	Bed1	wcp_{23}	Bed4
S5	Bed1	wcp_{23}	Bed5
S6	Bed4	Keine Kontrollflussmuster	Bed2
S7	Bed4	Keine Kontrollflussmuster	Bed3
S8	Bed4	Keine Kontrollflussmuster	Bed4
S9	Bed4	Keine Kontrollflussmuster	Bed5
S10	Bed4	Keine Kontrollflussmuster	Bed7
S11	Bed4	Keine Kontrollflussmuster	Bed9
S12	Bed4	Keine Kontrollflussmuster	Bed12
S13	Bed4	Keine Kontrollflussmuster	Bed13
S14	Bed4	Keine Kontrollflussmuster	Bed14
S15	Bed4	Keine Kontrollflussmuster	Bed26
S16	Bed4	Keine Kontrollflussmuster	Bed27
S17	Bed4	Keine Kontrollflussmuster	Bed32
S18	Bed4	Keine Kontrollflussmuster	Bed33
S19	Bed4	wcp_{23}	Bed4

SID	Kontrollflusssemantik «Vor der Ausführung»	Kontrollflusssemantik «Während der Ausführung»	Kontrollflusssemantik «Nach der Ausführung»
S20	Bed4	wcp_{23}	Bed5
S21	Bed6	Keine Kontrollflussmuster	Bed4
S22	Bed6	Keine Kontrollflussmuster	Bed5
S23	Bed8	Keine Kontrollflussmuster	Bed2
S24	Bed8	Keine Kontrollflussmuster	Bed3
S25	Bed8	Keine Kontrollflussmuster	Bed4
S26	Bed8	Keine Kontrollflussmuster	Bed5
S27	Bed8	Keine Kontrollflussmuster	Bed7
S28	Bed8	Keine Kontrollflussmuster	Bed12
S29	Bed8	Keine Kontrollflussmuster	Bed13
S30	Bed8	Keine Kontrollflussmuster	Bed14
S31	Bed8	Keine Kontrollflussmuster	Bed26
S32	Bed8	Keine Kontrollflussmuster	Bed27
S33	Bed8	Keine Kontrollflussmuster	Bed32
S34	Bed8	Keine Kontrollflussmuster	Bed33
S35	Bed8	wcp_{23}	Bed4
S36	Bed8	wcp_{23}	Bed5
S37	Bed10	Keine Kontrollflussmuster	Bed4
S38	Bed10	Keine Kontrollflussmuster	Bed9
S30	Bed11	Keine Kontrollflussmuster	Bed4
S40	Bed11	Keine Kontrollflussmuster	Bed9
S41	Bed13	Keine Kontrollflussmuster	Bed4
S42	Bed13	Keine Kontrollflussmuster	Bed5
S43	Bed15	Keine Kontrollflussmuster	Bed4
S44	Bed15	Keine Kontrollflussmuster	Bed5
S45	Bed16	Keine Kontrollflussmuster	Keine Nachbedingung
S46	Bed19	Keine Kontrollflussmuster	Keine Nachbedingung
S47	Bed22	Keine Kontrollflussmuster	Bed4
S48	Bed22	Keine Kontrollflussmuster	Bed5
S49	Bed26	Keine Kontrollflussmuster	Bed4
S50	Bed26	Keine Kontrollflussmuster	Bed5
S51	Bed26	Keine Kontrollflussmuster	Bed16
S52	Bed28	Keine Kontrollflussmuster	Bed4
S53	Bed28	Keine Kontrollflussmuster	Bed5
S54	Bed29	wcp_{13}, wcp_{14}, wcp_{21}, wcp_{34}, wcp_{36}, $wcp_{BPMNAngeheftetesUnterbrechendesEreignis}$, $wcp_{BPMNAngeheftetesNichtUnterbrechendesEreignis}$	Bed30
S55	Bed29	wcp_{13}, wcp_{14}, wcp_{17}, wcp_{21}, wcp_{34}, wcp_{36}, wcp_{40}, $wcp_{BPMNEreignisTeilprozessUnterbrechendesEreignis}$, $wcp_{BPMNEreignisTeilprozessNichtUnterbrechendesEreignis}$, $wcp_{BPMNAngeheftetesUnterbrechendesEreignis}$, $wcp_{BPMNAngeheftetesNichtUnterbrechendesEreignis}$, $wcp_{Teilprozess}$	Bed30
S56	Bed31	Keine Kontrollflussmuster	Bed4
S57	Bed31	Keine Kontrollflussmuster	Bed5

SID	Kontrollflusssemantik «Vor der Ausführung»	Kontrollflusssemantik «Während der Ausführung»	Kontrollflusssemantik «Nach der Ausführung»
S58	Bed34	Keine Kontrollflussmuster	Bed35
S59	Keine Vorbedingung	Keine Kontrollflussmuster	Bed16
S60	Keine Vorbedingung	wcp_{13}, wcp_{14}, wcp_{21}, wcp_{34}, wcp_{36}	Keine Nachbedingung
S61	Keine Vorbedingung	wcp_{23}	Bed4
S62	Keine Vorbedingung	wcp_{23}	Bed5

Tabelle 12: Instanzen - BPMN - Klasse Semantikschema

Knotentyp	SemantikschemaID (SID)
Aufgabe	S54
Teilprozess	S55
Transaktion	S55
Ereignis-Teilprozess	-
Aufruf-Aktivität	S55
Exklusives Gateway	S10, S25, S27
Ereignisbasiertes Gateway	S12, S28
Inklusives Gateway	S11, S37, S38
Paralleles Gateway	S9, S21, S22
Komplexes Gateway	S11, S39, S40
Exklusives Ereignisbasiertes Gateway (Instanziierend)	S3
Paralleles Ereignisbasiertes Gateway (Instanziierend)	S2
Pool	-
Blankoereignis	S1, S2, S6, S8, S9, S23, S25, S26
Nachrichtenereignis	S4, S5, S6, S8, S9, S13, S14, S17, S19, S20, S23, S25, S26, S29, S30, S33, S41, S42, S43, S44, S49, S50, S52, S53, S56, S57, S61, S62
Timerereignis	S4, S5, S17, S18, S35, S36, S61, S62
Eskalationsereignis	S6, S8, S9, S15, S16, S18, S23, S25, S26, S31, S32, S34, S49, S50, S52, S53, S56, S57
Bedingungsereignis	S4, S5, S19, S20, S35, S36, S61, S62
Linkereignis	S4, S5, S15, S16, S23, S31, S32, S49, S50, S52, S53
Fehlerereignis	S6, S15, S16, S23, S31, S32, S49, S50, S51, S53, S61, S62
Abbruchereignis	S6, S15, S16, S23, S31, S32, S49, S52
Kompensationsereignis	S6, S8, S9, S15, S16, S18, S23, S25, S26, S31, S32, S34, S49, S50, S51, S52, S53, S59, S60, S61
Signalereignis	S4, S5, S6, S8, S9, S15, S16, S18, S19, S20, S25, S26, S31, S32, S34, S35, S36, S43, S44, S47, S48, S56, S57, S61, S62
Mehrfachereignis	S4, S5, S6, S8, S9, S13, S14, S15, S16, S18, S19, S20, S23, S25, S26, S29, S30, S31, S32, S34, S35, S36, S43, S44, S47, S48, S56, S57, S61, S62
Mehrfach-/Parallelereignis	S4, S5, S6, S8, S9, S13, S14, S15, S16, S18, S19, S20, S23, S25, S26, S29, S30, S31, S32, S34, S35, S36, S43, S44, S47, S48, S56, S57, S61, S62
Terminierungsereignis	S7, S24
Datenobjekt	S58

Knotentyp	SemantikschemaID (SID)
Dateninputobjekt	S61, S62
Datenoutputobjekt	S45, S46
Datenspeicher	S58
Textannotation	-

Tabelle 13: Instanzen - BPMN - Klasse Semantikzuordnung

5.3.2 Ereignisgesteuerte Prozesskette (EPK)

Die Modellierungssprache Ereignisgesteuerte Prozesskette (EPK) wurde in Kapitel 3.3 vorgestellt, die aus den Knotentypen Ereignis, Funktion, XOR-Verknüpfungsoperator, OR-Verknüpfungsoperator, AND-Verknüpfungsoperator und Prozesswegweiser besteht, deren Notationselemente in Abbildung 16 illustriert werden. Die verwendete EPK-Variante besitzt als einzigen Kantentyp den Kontrollfluss (vgl. Notationselement in Abbildung 16d), welcher ein expliziter kontrollflussabhängiger Kantentyp ist.

Die Kontrollflusssemantik der einzelnen Knotentypen wird im Semantikschema durch die

- Kontrollflusssemantik «Vor der Ausführung»,
- Kontrollflusssemantik «Während der Ausführung» und
- Kontrollflusssemantik «Nach der Ausführung» sowie
- einer beliebigen Anzahl von Beziehungen

beschrieben. Instanzen der Klasse Beziehung existieren bei der betrachteten EPK-Variante nicht, so dass diese im Folgenden weggelassen werden. Die Kontrollflusssemantik «Vor der Ausführung» und «Nach der Ausführung» wird für das Semantikschema mit elementaren Bedingungen beschrieben. Eine elementare Bedingung wird durch einen Kantentyp, einen Gültigkeitsbereich, die ein- bzw. ausgehende Kantenanzahl und die Muster aus der Menge Kontrollflussmuster «Vor der Ausführung» bzw. «Nach der Ausführung» beschrieben. Die Bedingungen der verwendeten EPK-Variante müssen durch keinen Gültigkeitsbereich eingeschränkt werden, so dass dieser im Folgenden weggelassen wird. In Tabelle 14 werden die Instanzen der Klasse Bedingung einer EPK definiert sowie in Tabelle 15 die Instanzen der Klasse Semantikschema.

BedingungsID	Kantentyp	Anzahl	Kontrollflussmuster
Bed1	Kein Kantentyp	Keine Anzahl	wcp_{Start}
Bed2	Kein Kantentyp	Keine Anzahl	wcp_{11}
Bed3	Kontrollfluss	1	wcp_{1}
Bed4	Kontrollfluss	[2 … ∞]	wcp_{2}
Bed5	Kontrollfluss	[2 … ∞]	wcp_{3}, wcp_{33}, wcp_{37}, wcp_{38}

BedingungsID	Kantentyp	Anzahl	Kontrollflussmuster
Bed6	Kontrollfluss	[2 … ∞]	wcp_4
Bed7	Kontrollfluss	[2 … ∞]	wcp_5, wcp_8
Bed8	Kontrollfluss	[2 … ∞]	wcp_5, wcp_7, wcp_8, wcp_9, wcp_{37}, wcp_{38}
Bed9	Kontrollfluss	[2 … ∞]	wcp_6

Tabelle 14: Instanzen - EPK - Klasse Bedingung

SID	Kontrollflusssemantik «Vor der Ausführung»	Kontrollflusssemantik «Während der Ausführung»	Kontrollflusssemantik «Nach der Ausführung»
S1	Bed1	Keine Kontrollflussmuster	Bed2
S2	Bed1	Keine Kontrollflussmuster	Bed3
S3	Bed1	$wcp_{Teilprozess}$	Bed2
S4	Bed1	$wcp_{Teilprozess}$	Bed3
S5	Bed3	Keine Kontrollflussmuster	Bed2
S6	Bed3	Keine Kontrollflussmuster	Bed3
S7	Bed3	Keine Kontrollflussmuster	Bed4
S8	Bed3	Keine Kontrollflussmuster	Bed6
S9	Bed3	Keine Kontrollflussmuster	Bed9
S10	Bed3	$wcp_{Teilprozess}$	Bed2
S11	Bed3	$wcp_{Teilprozess}$	Bed3
S12	Bed5	Keine Kontrollflussmuster	Bed3
S13	Bed5	Keine Kontrollflussmuster	Bed4
S14	Bed7	Keine Kontrollflussmuster	Bed3
S15	Bed7	Keine Kontrollflussmuster	Bed6
S16	Bed8	Keine Kontrollflussmuster	Bed3
S17	Bed8	Keine Kontrollflussmuster	Bed9

Tabelle 15: Instanzen - EPK - Klasse Semantikschema

Die Verknüpfung der Kontrollflusssemantik im Semantikschema mit den entsprechenden Knotentypen der Modellierungssprache wird in der Klasse Semantikzuordnung (vgl. Klasse `Semantikzuordnung` in Abbildung 30) vorgenommen, wie in Tabelle 16 beschrieben wird. Mit dieser Zuordnung ist die Kontrollflusssemantik der vorgestellten EPK-Variante in Kapitel 3.3 beschrieben, wobei auf eine Vollständigkeitsüberprüfung an dieser Stelle verzichtet wird.

Knotentyp	SemantikschemaID
Ereignis	S1, S2, S5, S6
Funktion	S6
XOR-Verknüpfungsoperator	S8, S14, S15
OR-Verknüpfungsoperator	S9, S16, S17
AND-Verknüpfungsoperator	S7, S12, S13
Prozesswegweiser	S3, S4, S10, S11

Tabelle 16: Instanzen - EPK - Klasse Semantikzuordnung

5.3.3 UML-Aktivitätsdiagramm (UML-AD)

Die Modellierungssprache UML-Aktivitätsdiagramm (UML-AD) wurde in Kapitel 3.4 vorgestellt. Die Knotentypen der Modellierungssprache werden in Tabelle 21 aufgezeigt sowie deren Notationselemente in Abbildung 18, 19, 20, 21 und 22. Als Kantentypen existieren der explizite kontrollflussabhängige Kantentyp Sequenzfluss, deren Notationselemente in Abbildung 23 illustriert werden, sowie der implizite kontrollflussabhängige Kantentyp impliziter Sequenzfluss. Darüber hinaus existiert der Entscheidungseingangsfluss als expliziter nicht-kontrollflussabhängiger Kantentyp, dessen Notationselement in Abbildung 20b illustriert wird.

Die Kontrollflusssemantik der einzelnen Knotentypen wird im Semantikschema durch die

- Kontrollflusssemantik «Vor der Ausführung»,
- Kontrollflusssemantik «Während der Ausführung» und
- Kontrollflusssemantik «Nach der Ausführung» sowie
- einer beliebigen Anzahl von Beziehungen

beschrieben. Beziehungen besitzen einen Kantentyp und eine Kantenanzahl. Die UML-AD besitzt nur eine Instanz der Klasse Beziehung mit dem Kantentyp Entscheidungseingangsfluss und der Kantenanzahl 1, wie aus Tabelle 17 hervorgeht.

BeziehungsID	Kantentyp	Anzahl
Bez1	Entscheidungseingangsfluss	1

Tabelle 17: Instanz - UML-AD - Klasse Beziehung

Die Kontrollflusssemantik «Vor der Ausführung» und «Nach der Ausführung» für ein Semantikschema wird jeweils durch eine elementare oder zusammengesetzte Bedingung beschrieben. Die zusammengesetzte Bedingung besteht aus zwei oder mehreren elementaren Bedingungen, welche über die EBNF-Grammatik `ZusammengesetzteBedingung`

miteinander verknüpft werden. Eine elementare Bedingung wird durch einen Kantentyp, einen Gültigkeitsbereich, die ein- bzw. ausgehende Kantenanzahl und die Muster aus der Menge Kontrollflussmuster «Vor der Ausführung» bzw. «Nach der Ausführung» beschrieben. Die Bedingungen der verwendeten UML-AD-Variante müssen durch keinen Gültigkeitsbereich eingeschränkt werden, so dass dieser im Folgenden weggelassen wird. Die Instanzen der elementaren Bedingungen werden in Tabelle 18 und die Instanzen der zusammengesetzten Bedingungen in Tabelle 19 beschrieben.

BedingungsID	Kantentyp	Anzahl	Kontrollflussmuster
Bed1	Kein Kantentyp	Keine Anzahl	wcp_{Start}
Bed2	Kein Kantentyp	Keine Anzahl	wcp_{11}
Bed3	Kein Kantentyp	Keine Anzahl	wcp_{43}
Bed4	Sequenzfluss	1	wcp_{1}
Bed5	Sequenzfluss	[2 … ∞]	wcp_{2}
Bed6	Sequenzfluss	[2 … ∞]	wcp_{3}, wcp_{33}, wcp_{37}, wcp_{38}
Bed7	Sequenzfluss	[2 … ∞]	wcp_{4}
Bed8	Sequenzfluss	[2 … ∞]	wcp_{5}, wcp_{8}
Bed9	Sequenzfluss	[2 … ∞]	wcp_{9}, wcp_{28}, wcp_{30}, wcp_{31}
Bed10	Impliziter Sequenzfluss	1	wcp_{1}
Bed11	Impliziter Sequenzfluss	[2 … ∞]	wcp_{2}
Bed12	Impliziter Sequenzfluss	[2 … ∞]	wcp_{3}, wcp_{33}, wcp_{37}, wcp_{38}

Tabelle 18: Instanzen - UML-AD - Klasse elementare Bedingung

BedingungsID	Zusammengesetzte Bedingung
Bed13	(Bed04 AND Bed10)
Bed14	(Bed04 AND Bed11)
Bed15	(Bed05 AND Bed10)
Bed16	(Bed05 AND Bed11)
Bed17	(Bed04 AND Bed12)
Bed18	(Bed06 AND Bed10)
Bed19	(Bed06 AND Bed12)

Tabelle 19: Instanzen - UML-AD - Klasse zusammengesetzte Bedingung

Nachdem die Bestandteile der Klasse Semantikschema, d. h. die Instanz der Klasse Beziehung (vgl. Tabelle 17), die Instanzen der Klasse elementare Bedingung (vgl. Tabelle 18) und die Instanzen der Klasse zusammengesetzte Bedingung (vgl. Tabelle 19) definiert wurden, kann das Semantikschema in Tabelle 20 für UML-AD definiert werden. Die Instanzen mit der SemantikschemaID (SID) S12, S41 und S42 der Klasse Semantikschema besitzen zusätzlich die Beziehung Bez1 (vgl. Tabelle 17), welche aber aus Gründen der Übersichtlichkeit in Tabelle 20 ausgeblendet wurde. Abschließend werden die Instanzen den Knotentypen der Klasse Semantikzuordnung (vgl. Klasse `Semantikzuordnung` in Abbildung 30) zugeordnet

(vgl. Tabelle 21). Mit dieser Zuordnung ist die Kontrollflusssemantik der vorgestellten UML-AD-Variante aus Kapitel 3.4 beschrieben, wobei auf eine Vollständigkeitsprüfung an dieser Stelle verzichtet wird.

SID	Kontrollflusssemantik «Vor der Ausführung»	Kontrollflusssemantik «Während der Ausführung»	Kontrollflusssemantik «Nach der Ausführung»
S1	Bed01	Keine Kontrollflussmuster	Bed04
S2	Bed01	Keine Kontrollflussmuster	Bed05
S3	Bed01	wcp_{23}	Bed04
S4	Bed01	wcp_{23}	Bed05
S5	Bed01	$wcp_{Teilprozess}$	Bed03
S6	Bed04	Keine Kontrollflussmuster	Bed02
S7	Bed04	Keine Kontrollflussmuster	Bed03
S8	Bed04	Keine Kontrollflussmuster	Bed04
S9	Bed04	Keine Kontrollflussmuster	Bed05
S10	Bed04	Keine Kontrollflussmuster	Bed10
S11	Bed04	Keine Kontrollflussmuster	Bed11
S12	Bed04	Keine Kontrollflussmuster	Bed07
S13	Bed04	Keine Kontrollflussmuster	Bed13
S14	Bed04	Keine Kontrollflussmuster	Bed14
S15	Bed04	Keine Kontrollflussmuster	Bed15
S16	Bed04	Keine Kontrollflussmuster	Bed16
S17	Bed04	wcp_{19}, wcp_{23}, wcp_{25}	Bed04
S18	Bed04	wcp_{19}, wcp_{23}, wcp_{25}	Bed05
S19	Bed04	wcp_{21}, $wcp_{Teilprozess}$	Bed04
S20	Bed04	wcp_{21}, $wcp_{Teilprozess}$	Bed05
S21	Bed04	wcp_{21}, $wcp_{Teilprozess}$	Bed10
S22	Bed04	wcp_{21}, $wcp_{Teilprozess}$	Bed11
S23	Bed04	wcp_{23}	Bed04
S24	Bed04	wcp_{23}	Bed05
S25	Bed06	Keine Kontrollflussmuster	Bed02
S26	Bed06	Keine Kontrollflussmuster	Bed04
S27	Bed06	Keine Kontrollflussmuster	Bed05
S28	Bed06	Keine Kontrollflussmuster	Bed10
S29	Bed06	Keine Kontrollflussmuster	Bed11
S30	Bed06	Keine Kontrollflussmuster	Bed13
S31	Bed06	Keine Kontrollflussmuster	Bed14
S32	Bed06	Keine Kontrollflussmuster	Bed15
S33	Bed06	Keine Kontrollflussmuster	Bed16
S34	Bed06	wcp_{19}, wcp_{23}, wcp_{25}	Bed04
S35	Bed06	wcp_{19}, wcp_{23}, wcp_{25}	Bed05
S36	Bed06	wcp_{21}, $wcp_{Teilprozess}$	Bed04
S37	Bed06	wcp_{21}, $wcp_{Teilprozess}$	Bed05
S38	Bed06	wcp_{21}, $wcp_{Teilprozess}$	Bed10
S39	Bed06	wcp_{21}, $wcp_{Teilprozess}$	Bed11
S40	Bed06	wcp_{23}	Bed04

SID	Kontrollflusssemantik «Vor der Ausführung»	Kontrollflusssemantik «Während der Ausführung»	Kontrollflusssemantik «Nach der Ausführung»
S41	Bed06	wcp_{23}	Bed05
S42	Bed08	Keine Kontrollflussmuster	Bed02
S43	Bed08	Keine Kontrollflussmuster	Bed03
S44	Bed08	Keine Kontrollflussmuster	Bed04
S45	Bed08	Keine Kontrollflussmuster	Bed07
S46	Bed09	Keine Kontrollflussmuster	Bed04
S47	Bed09	Keine Kontrollflussmuster	Bed05
S48	Bed10	Keine Kontrollflussmuster	Bed04
S49	Bed10	Keine Kontrollflussmuster	Bed05
S50	Bed10	Keine Kontrollflussmuster	Bed10
S51	Bed10	Keine Kontrollflussmuster	Bed11
S52	Bed10	wcp_{21}, $wcp_{Teilprozess}$	Bed04
S53	Bed10	wcp_{21}, $wcp_{Teilprozess}$	Bed05
S54	Bed10	wcp_{21}, $wcp_{Teilprozess}$	Bed10
S55	Bed10	wcp_{21}, $wcp_{Teilprozess}$	Bed11
S56	Bed12	Keine Kontrollflussmuster	Bed04
S57	Bed12	Keine Kontrollflussmuster	Bed05
S58	Bed12	Keine Kontrollflussmuster	Bed10
S59	Bed12	Keine Kontrollflussmuster	Bed11
S60	Bed12	wcp_{21}, $wcp_{Teilprozess}$	Bed04
S61	Bed12	wcp_{21}, $wcp_{Teilprozess}$	Bed05
S62	Bed12	wcp_{21}, $wcp_{Teilprozess}$	Bed10
S63	Bed12	wcp_{21}, $wcp_{Teilprozess}$	Bed11
S64	Bed13	Keine Kontrollflussmuster	Bed04
S65	Bed13	Keine Kontrollflussmuster	Bed05
S66	Bed17	Keine Kontrollflussmuster	Bed04
S67	Bed17	Keine Kontrollflussmuster	Bed05
S68	Bed18	Keine Kontrollflussmuster	Bed04
S69	Bed18	Keine Kontrollflussmuster	Bed05
S70	Bed19	Keine Kontrollflussmuster	Bed04
S71	Bed19	Keine Kontrollflussmuster	Bed05
S72	Keine Vorbedingung	wcp_{19}, wcp_{23}, wcp_{25}	Bed04

Tabelle 20: Instanzen - UML-AD - Klasse Semantikschema

Knotentyp	SemantikschemaID (SID)
Aktivität	S5
Aktion	S19, S20, S21, S22, S36, S37, S38, S39, S52, S53, S54, S55, S60, S61, S62, S63, S38, S60
Senden von Signalen	S8, S9, S13, S14, S15, S16, S26, S27, S30, S31, S32, S33
Empfangen von Signalen	S3, S4, S17, S18, S26, S27, S34, S35, S64, S65, S66, S67, S68, S69, S70, S71, S72
Zeitereignis	S23, S24, S40, S41

Knotentyp	SemantikschemaID (SID)
Entscheidungs-/Zusammenführungsknoten	S12, S44, S45
Aufspaltungs-/Synchronisationsknoten	S9, S26, S27, S46, S47
Startknoten	S1, S2
Aktivitätsendknoten	S7, S43
Ablaufende	S6, S42
Objektknoten	S1, S2, S6, S8, S9, S10, S11, S25, S26, S27, S28, S29, S48, S49, S50, S51, S56, S57, S58, S59
Konnektorknoten	S10, S48
Aktivitätsgruppe	-

Tabelle 21: Instanzen - UML-AD - Klasse Semantikzuordnung

5.3.4 Petri-Netz (PN)

Die Modellierungssprache Petri-Netze wurde in Kapitel 3.5 vorgestellt und besteht aus den Knotentypen Stelle und Transition, deren Notationselemente in Abbildung 24 illustriert werden. Als einziger Kantentyp ist der Sequenzfluss (vgl. Abbildung 24) definiert, der ein expliziter kontrollflussabhängiger Kantentyp ist.

Die Kontrollflusssemantik der einzelnen Knotentypen wird im Semantikschema durch die

- Kontrollflusssemantik «Vor der Ausführung»,
- Kontrollflusssemantik «Während der Ausführung» und
- Kontrollflusssemantik «Nach der Ausführung» sowie
- einer beliebigen Anzahl von Beziehungen

beschrieben. Instanzen der Klasse Beziehung und Kontrollflusssemantik «Während der Ausführung» existieren bei der verwendeten Petri-Netz-Variante nicht, so dass diese im Folgenden weggelassen werden. Die Kontrollflusssemantik «Vor der Ausführung» und «Nach der Ausführung» für ein Semantikschema wird jeweils durch eine elementare Bedingung beschrieben. Eine elementare Bedingung wird durch einen Kantentyp, einen Gültigkeitsbereich, die ein- bzw. ausgehende Kantenanzahl und die Muster aus der Menge Kontrollflussmuster «Vor der Ausführung» bzw. «Nach der Ausführung» beschrieben. Die Bedingungen der verwendeten Petri-Netz-Variante müssen durch keinen Gültigkeitsbereich eingeschränkt werden, so dass dieser im Folgenden weggelassen wird. In Tabelle 22 werden die Instanzen der Klasse Bedingungen der Petri-Netze definiert sowie in Tabelle 23 die Instanzen der Klasse Semantikschema.

BedingungsID	Kantentyp	Anzahl	Kontrollflussmuster
Bed1	Kein Kantentyp	Keine Anzahl	wcp_{Start}
Bed2	Kein Kantentyp	Keine Anzahl	wcp_{11}
Bed3	Sequenzfluss	1	wcp_{1}
Bed4	Sequenzfluss	[2 … ∞]	wcp_{2}
Bed5	Sequenzfluss	[2 … ∞]	wcp_{3}, wcp_{33}, wcp_{37}, wcp_{38}
Bed6	Sequenzfluss	[2 … ∞]	wcp_{16}
Bed7	Sequenzfluss	[2 … ∞]	wcp_{5}, wcp_{8}

Tabelle 22: Instanzen - Petri-Netze - Klasse Bedingung

SemantikschemaID (SID)	Kontrollflusssemantik «Vor der Ausführung»	Kontrollflusssemantik «Nach der Ausführung»
S1	Bed1	Bed2
S2	Bed1	Bed3
S3	Bed1	Bed4
S4	Bed1	Bed6
S5	Bed3	Bed2
S6	Bed3	Bed3
S7	Bed3	Bed4
S8	Bed3	Bed6
S9	Bed5	Bed2
S10	Bed5	Bed3
S11	Bed5	Bed4
S12	Bed7	Bed2
S13	Bed7	Bed3
S14	Bed7	Bed6

Tabelle 23: Instanzen - Petri-Netze - Klasse Semantikschema

Die Verknüpfung der definierten Kontrollflusssemantik im Semantikschema mit den entsprechenden Knotentypen der Modellierungssprache wird in der Klasse Semantikzuordnung (vgl. Klasse `Semantikzuordnung` in Abbildung 30) vorgenommen, wie in Tabelle 16 beschrieben wird. Mit dieser Zuordnung ist die Kontrollflusssemantik der vorgestellten Petri-Netz-Variante aus Kapitel 3.5 beschrieben.

Knotentyp	SemantikschemaID
Stelle	S1, S2, S4, S5, S6, S8, S12, S13, S14
Transition	S1, S2, S3, S5, S6, S7, S9, S10, S11

Tabelle 24: Instanzen - Petri-Netze - Klasse Semantikzuordnung

5.3.5 New Yet Another Workflow Language (newYAWL)

Die Modellierungssprache YAWL bzw. newYAWL wurde in Kapitel 3.6 vorgestellt und besteht aus den Knotentypen Startbedingung, Bedingung, Endbedingung und Aufgabe, deren Notationselemente in Abbildung 25 aufgezeigt werden. Als Kantentyp existiert der Kontrollfluss (vgl. z. B. Abbildung 25h), der ein expliziter kontrollflussabhängiger Kantentyp ist. Darüber hinaus gibt es die Kantentypen

- Abbruchkante (vgl. Notationselement in Abbildung 25n),
- Beendigungskante (Notationselement siehe Abbildung 25v),
- Deaktivierungskante (Notationselement siehe Abbildung 25w) und
- Blockierungskante (Notationselement siehe Abbildung 25s),

die zu den expliziten nicht-kontrollflussabhängigen Knotentypen zählen. Die Kontrollflusssemantik der einzelnen Knotentypen wird im Semantikschema durch die

- Kontrollflusssemantik «Vor der Ausführung»,
- Kontrollflusssemantik «Während der Ausführung» und
- Kontrollflusssemantik «Nach der Ausführung» sowie
- einer beliebigen Anzahl von Beziehungen

beschrieben. Beziehungen besitzen einen Kantentyp und eine Kantenanzahl. Die Instanzen der Beziehungen werden in YAWL bzw. newYAWL in Tabelle 25 definiert.

BeziehungsID	Kantentyp	Anzahl
Bez2	Abbruchkante	[1 … ∞]
Bez4	Beendigungskante	[1 … ∞]
Bez6	Deaktivierungskante	[0 … ∞]
Bez7	Deaktivierungskante	[1 … ∞]
Bez9	Blockierungskante	[1 … ∞]

Tabelle 25: Instanzen - YAWL bzw. newYAWL - Klasse Beziehung

Die Kontrollflusssemantik «Vor der Ausführung» und «Nach der Ausführung» für ein Semantikschema wird jeweils durch eine elementare oder zusammengesetzte Bedingung beschrieben. Die zusammengesetzte Bedingung besteht wiederum aus zwei oder mehreren elementaren Bedingungen, welche über die EBNF-Grammatik `ZusammengesetzteBedingung` miteinander verknüpft sind. Eine elementare Bedingung wird durch einen Kantentyp, einen Gültigkeitsbereich, die ein- bzw. ausgehende Kantenanzahl und die Muster aus der Menge Kontrollflussmuster «Vor der Ausführung» bzw. «Nach der Ausführung» beschrieben. Die Bedingungen der verwendeten newYAWL-Variante müssen durch keinen

Gültigkeitsbereich eingeschränkt werden, so dass dieser im Folgenden weggelassen wird. Darüber hinaus existieren keine zusammengesetzten Bedingungen. Die Objekte der elementaren Bedingungen werden in Tabelle 26 definiert.

BedingungsID	Kantentyp	Anzahl	Kontrollflussmuster
Bed1	Kein Kantentyp	Keine Anzahl	wcp_{Start}
Bed2	Kein Kantentyp	Keine Anzahl	wcp_{43}
Bed3	Sequenzfluss	1	wcp_{1}
Bed4	Sequenzfluss	1	wcp_{41}
Bed5	Sequenzfluss	1	wcp_{42}
Bed6	Sequenzfluss	[2 … ∞]	wcp_{2}
Bed7	Sequenzfluss	[2 … ∞]	wcp_{3}, wcp_{33}, wcp_{37}, wcp_{38}
Bed8	Sequenzfluss	[2 … ∞]	wcp_{4}
Bed9	Sequenzfluss	[2 … ∞]	wcp_{4}, wcp_{16}
Bed10	Sequenzfluss	[2 … ∞]	wcp_{5}, wcp_{8}
Bed11	Sequenzfluss	[2 … ∞]	wcp_{6}
Bed12	Sequenzfluss	[2 … ∞]	wcp_{7}, wcp_{37}, wcp_{38}
Bed13	Sequenzfluss	[2 … ∞]	wcp_{9}
Bed14	Sequenzfluss	[2 … ∞]	wcp_{28}
Bed15	Sequenzfluss	[2 … ∞]	wcp_{29}
Bed16	Sequenzfluss	[2 … ∞]	wcp_{30}
Bed17	Sequenzfluss	[2 … ∞]	wcp_{31}
Bed18	Sequenzfluss	[2 … ∞]	wcp_{32}

Tabelle 26: Instanzen - YAWL bzw. newYAWL - Klasse elementare Bedingung

Nachdem die Bestandteile der Klasse Semantikschema, d. h. die Instanzen der Klasse Beziehung (vgl. Tabelle 9) und die Instanzen der Klasse elementare Bedingung (vgl. Tabelle 10) definiert wurden, können die Instanzen der Klasse Semantikschema in Tabelle 27 für YAWL bzw. newYAWL definiert werden. Die Bedingungen in Tabelle 27 werden aus Gründen der Übersichtlichkeit in einer EBNF-Grammatik angegeben. Abschließend werden die Semantikschemas den Knotentypen in der Klasse Semantikzuordnung (vgl. Klasse `Semantikzuordnung` in Abbildung 30) zugeordnet (vgl. Tabelle 28). Mit dieser Zuordnung ist die Kontrollflusssemantik der beschriebenen YAWL bzw. newYAWL-Variante beschrieben, wobei auf eine Vollständigkeitsüberprüfung an dieser Stelle verzichtet wird.

SID	Kontrollflusssemantik «Vor der Ausführung»	Kontrollflusssemantik «Während der Ausführung»	Kontrollflusssemantik «Nach der Ausführung»	Beziehung
S1	Bed1	Keine Kontrollflussmuster	Bed03	Keine Beziehung
S2	Bed1	Keine Kontrollflussmuster	Bed09	Keine Beziehung
S3	Bed3	Keine Kontrollflussmuster	Bed03	Keine Beziehung

SID	Kontrollflusssemantik «Vor der Ausführung»	Kontrollflusssemantik «Während der Ausführung»	Kontrollflusssemantik «Nach der Ausführung»	Beziehung
S4	Bed3	Keine Kontrollflussmuster	Bed09	Keine Beziehung
S5	Bed3	Keine Kontrollflussmuster	Bed02	Keine Beziehung
S6	(Bed3 | Bed7 | Bed10 | Bed12 | Bed13 | Bed16)	wcp_{13}, wcp_{14}, wcp_{15}, wcp_{21}, wcp_{23}, wcp_{24}, wcp_{34}, wcp_{35}, $wcp_{Teilprozess}$	(Bed3 | Bed5 | Bed6 | Bed7 | Bed11)	Bez1, Bez3, Bez6, Bez8
S7	(Bed3 | Bed77 | Bed10 | Bed12 | Bed13 | Bed15 | Bed16 | Bed18)	wcp_{13}, wcp_{14}, wcp_{15}, wcp_{19}, wcp_{20}, wcp_{21}, wcp_{23}, wcp_{24}, wcp_{25}, wcp_{26}, wcp_{34}, wcp_{35}, $wcp_{Teilprozess}$	(Bed3 | Bed5 | Bed6 | Bed7 | Bed11)	Bez2, Bez3, Bez6, Bez8
S8	(Bed3 | Bed7 | Bed10 | Bed12 | Bed13 | Bed16)	wcp_{13}, wcp_{14}, wcp_{15}, wcp_{21}, wcp_{23}, wcp_{24}, wcp_{27}, wcp_{34}, wcp_{35}, $wcp_{Teilprozess}$	(Bed3 | Bed5 | Bed6 | Bed7 | Bed11)	Bez1, Bez4, Bez6, Bez8
S9	(Bed3 | Bed7 | Bed10 | Bed12 | Bed13 | Bed16)	wcp_{13}, wcp_{14}, wcp_{15}, wcp_{21}, wcp_{23}, wcp_{24}, wcp_{34}, wcp_{35}, wcp_{36}, $wcp_{Teilprozess}$	(Bed3 | Bed5 | Bed6 | Bed7 | Bed11)	Bez1, Bez3, Bez7, Bez8
S10	(Bed3 | Bed7 | Bed10 | Bed12 | Bed13 | Bed16)	wcp_{13}, wcp_{14}, wcp_{15}, wcp_{19}, wcp_{20}, wcp_{21}, wcp_{23}, wcp_{24}, wcp_{25}, wcp_{26}, wcp_{27}, wcp_{34}, wcp_{35}, $wcp_{Teilprozess}$	(Bed3 | Bed5 | Bed6 | Bed7 | Bed11)	Bez2, Bez4, Bez6, Bez8
S11	(Bed3 | Bed7 | Bed10 | Bed12 | Bed13 | Bed16)	wcp_{13}, wcp_{14}, wcp_{15}, wcp_{21}, wcp_{23}, wcp_{24}, wcp_{34}, wcp_{35}, wcp_{36}, $wcp_{Teilprozess}$	(Bed3 | Bed5 | Bed6 | Bed7 | Bed11)	Bez2, Bez3, Bez7, Bez8
S12	(Bed3 | Bed7 | Bed10 | Bed12 | Bed13 | Bed16)	wcp_{13}, wcp_{14}, wcp_{15}, wcp_{21}, wcp_{23}, wcp_{24}, wcp_{27}, wcp_{34}, wcp_{35}, wcp_{36}, $wcp_{Teilprozess}$	(Bed3 | Bed5 | Bed6 | Bed7 | Bed11)	Bez1, Bez4, Bez7, Bez8
S13	Bed10	Keine Kontrollflussmuster	Bed2	Keine Beziehung
S14	Bed10	Keine Kontrollflussmuster	Bed3	Keine Beziehung
S15	Bed10	Keine Kontrollflussmuster	Bed9	Keine Beziehung
S16	(Bed14 | Bed17)	wcp_{13}, wcp_{14}, wcp_{15}, wcp_{21}, wcp_{23}, wcp_{24}, wcp_{34}, wcp_{35}, $wcp_{Teilprozess}$	(Bed3 | Bed5 | Bed6 | Bed7 | Bed11)	Bez1, Bez3, Bez6, Bez9

SID	Kontrollflusssemantik «Vor der Ausführung»	Kontrollflusssemantik «Während der Ausführung»	Kontrollflusssemantik «Nach der Ausführung»	Beziehung
S17	(Bed14 | Bed17)	wcp_{13}, wcp_{14}, wcp_{15}, wcp_{19}, wcp_{20}, wcp_{21}, wcp_{23}, wcp_{24}, wcp_{25}, wcp_{26}, wcp_{34}, wcp_{35}, $wcp_{Teilprozess}$	(Bed3 | Bed5 | Bed6 | Bed7 | Bed11)	Bez1, Bez3, Bez6, Bez9
S18	(Bed14 | Bed17)	wcp_{13}, wcp_{14}, wcp_{15}, wcp_{21}, wcp_{23}, wcp_{24}, wcp_{27}, wcp_{34}, wcp_{35}, $wcp_{Teilprozess}$	(Bed3 | Bed5 | Bed6 | Bed7 | Bed11)	Bez1, Bez3, Bez6, Bez9
S19	(Bed14 | Bed17)	wcp_{13}, wcp_{14}, wcp_{15}, wcp_{21}, wcp_{23}, wcp_{24}, wcp_{34}, wcp_{35}, wcp_{36}, $wcp_{Teilprozess}$	(Bed3 | Bed5 | Bed6 | Bed7 | Bed11)	Bez1, Bez3, Bez6, Bez9
S20	(Bed14 | Bed17)	wcp_{13}, wcp_{14}, wcp_{15}, wcp_{19}, wcp_{20}, wcp_{21}, wcp_{23}, wcp_{24}, wcp_{25}, wcp_{26}, wcp_{27}, wcp_{34}, wcp_{35}, $wcp_{Teilprozess}$	(Bed3 | Bed5 | Bed6 | Bed7 | Bed11)	Bez1, Bez3, Bez6, Bez9
S21	(Bed14 | Bed17)	wcp_{13}, wcp_{14}, wcp_{15}, wcp_{21}, wcp_{23}, wcp_{24}, wcp_{34}, wcp_{35}, wcp_{36}, $wcp_{Teilprozess}$	(Bed3 | Bed5 | Bed6 | Bed7 | Bed11)	Bez1, Bez3, Bez6, Bez9
S22	(Bed14 | Bed17)	wcp_{13}, wcp_{14}, wcp_{15}, wcp_{21}, wcp_{23}, wcp_{24}, wcp_{27}, wcp_{34}, wcp_{35}, wcp_{36}, $wcp_{Teilprozess}$	(Bed3 | Bed5 | Bed6 | Bed7 | Bed11)	Bez1, Bez3, Bez6, Bez9

Tabelle 27: Instanzen - YAWL bzw. newYAWL - Klasse Semantikschema

Knotentyp	SID
Bedingung	S3, S4, S14, S15
Startbedingung	S1, S2
Endbedingung	S5, S13
Aufgabe	S6, S7, S8, S9, S10, S11, S12, S16, S17, S18, S19, S20, S21, S22
Abbruchbereich	-
Blockierungsbereich	-
Beendigungsbereich	-

Tabelle 28: Instanzen - YAWL bzw. newYAWL - Klasse Semantikzuordnung

5.4 Zusammenfassung

In den vorherigen Abschnitten wurde eine auf Kontrollflussmustern basierende Methode zur Beschreibung der Kontrollflusssemantik von graphischen Geschäftsprozessmodellierungssprachen vorgestellt. Mit der Kontrollflusssemantik werden die Methoden zur

Verbesserung der Abläufe und Strukturen erweitert, so dass unter anderem die Analyse (z. B. in Form einer Simulation), Überwachung (z. B. mittels einer Workflow Engine) und Verbesserung (z. B. im Kontext von Kosten, Zeit, Ressourcen) der dokumentierten Geschäftsprozesse möglich wird. Eine weitere Möglichkeit, die durch die entwickelte Methode entsteht, ist die Austauschbarkeit der Notationselemente einer Geschäftsprozessmodellierungssprache. Zum Beispiel können die Notationselemente in Abhängigkeit von der Anwendungsdomäne ausgetauscht werden bzw. eine Modellierungssprache kann auf eine andere Modellierungssprache zur Vereinheitlichung abgebildet oder auch zur Verbesserung der Verständlichkeit eingesetzt werden. Die Anwendung der Methode zur Beschreibung der Kontrollflusssemantik einer Modellierungssprache adressiert daher hauptsächlich Designer von Geschäftsprozessmodellierungssprachen und Modellierer von Geschäftsprozessmodellen, die die Methoden zur Analyse der Prozessmodelle erweitern wollen sowie auch Software-Werkzeugbauer, die beispielsweise eine IT-gestützte Analyse von Geschäftsprozessmodellen ermöglichen möchten.

Für die Anwendung der entwickelten Methode wurden einführend zwei Anforderungen definiert. Zum einen müssen die Syntaxelemente der Geschäftsprozessmodellierungssprache durch die zwei disjunkten Mengen Knoten- und Kantentyp beschrieben werden können (vgl. Abbildung 29). Zum anderen muss die Kontrollflusssemantik der Knotentypen durch die Kontrollflusssemantik «Vor der Ausführung», «Während der Ausführung» und «Nach der Ausführung» beschrieben werden können (vgl. Abbildung 30). Die vorgestellten Modellierungssprachen aus Kapitel 3 wurden daraufhin auf die Erfüllung dieser Anforderungen hin überprüft, mit dem Ergebnis, dass die Modellierungssprachen EPK, UML-AD, Petri-Netze und YAWL bzw. newYAWL die Anforderungen erfüllen. BPMN erfüllte nur die erste Anforderung. Für die Erfüllung der zweiten Anforderung wurden vier benutzerdefinierte Kontrollflussmuster entwickelt.

Im nächsten Schritt wurde das Vorgehen zur Beschreibung der Kontrollflusssemantik einer Geschäftsprozessmodellierungssprache beschrieben. Das Vorgehen wurde durch ein BPMN-Modell (vgl. Abbildung 42) definiert, um die Ablaufbeschreibung auch als Grundlage für die prototypische Realisierung des Software-Werkzeuges (vgl. Kapitel 7.2) einsetzen zu können. Darauf aufbauend wurde die Kontrollflusssemantik für die Modellierungssprachen aus Kapitel 3 beschrieben. Die Kontrollflusssemantik der Modellierungssprachen wurden mit dem Modell aus Abbildung 30 definiert. Beispielsweise wurde das Semantikschema für BPMN in Tabelle 12 beschrieben, welches auf die Beziehungen in Tabelle 9 und die Bedingungen in Tabelle 10 bzw. Tabelle 11 referenziert. Mit der Semantikzuordnung in Tabelle 13 wurde das Semantikschema den Knotentypen zugeordnet, wodurch die Kontrollflusssemantik der BPMN beschrieben ist. Analog wurde dies auch für die anderen Geschäftsprozessmodellierungssprachen aus Kapitel 3 durchgeführt. Bei anderen Varianten der

verwendeten Geschäftsprozessmodellierungssprachen müssen die entsprechenden Semantikschemas angepasst werden. Für die Beschreibung der Kontrollflusssemantik von anderen Modellierungssprachen müssen die gleichen Schritte durchgeführt werden.

Nachdem die Kontrollflusssemantik der Geschäftsprozessmodellierungssprachen in diesem Kapitel beschrieben wurde, kann auf dessen Grundlage die Ablaufreihenfolge der Elemente eines Geschäftsprozessmodells präzise beschrieben werden (vgl. Kapitel 6). Es wurde eine Differenzierung vorgenommen, da in diesem Kapitel zunächst der mögliche Wertebereich der Kontrollflusssemantik eines Syntaxelementes definiert wurde, weil ein Knotentyp mehrere Semantikschemas besitzen kann. Ein Semantikschema kann wiederum auch auf mehrere Kontrollflussmuster referenzieren. Beispielsweise können in der Semantikzuordnung für einen Knotentyp zwei Semantikschemas definiert sein. Das erste Semantikschema referenziert als Nachbedingung auf das Kontrollflussmuster Sequenz (wcp_1). Das zweite Semantikschema referenziert auf die Kontrollflussmuster exklusive Auswahl (wcp_4) und parallele Aufspaltung (wcp_2). Damit die Kontrollflusssemantik des Elementes im Geschäftsprozessmodell eindeutig ist, darf eine Ausprägung eines Knotentyps, d. h. ein Element im Geschäftsprozessmodell nur mit einem der beiden Semantikschemas spezifiziert werden. Beim zweiten Semantikschema muss eines der beiden möglichen Kontrollflussmuster ausgewählt werden. Weiterführend müssen die entsprechenden Parameter für das Kontrollflussmuster (vgl. Kapitel 4.1.2) spezifiziert werden. Die Auswahl eines Semantikschemas für ein Element im Geschäftsprozessmodell und deren Kontrollflussmuster sowie die Spezifikation der Parameter der Kontrollflussmuster werden in Kapitel 6 thematisiert.

6 Ablaufreihenfolge der Elemente in Geschäftsprozessmodellen

Nachdem im vorherigen Kapitel die Kontrollflusssemantik einer Geschäftsprozessmodellierungssprache definiert wurde, kann im nächsten Schritt die Ablaufreihenfolge der Elemente eines Geschäftsprozessmodells festgelegt werden. Die Beschreibung der Kontrollflusssemantik einer Geschäftsprozessmodellierungssprache und die Beschreibung der Ablaufreihenfolge der Elemente eines Geschäftsprozessmodells wurde untergliedert, da ein Knotentyp durch mehrere Semantikschemas (vgl. z. B. Tabelle 28) spezifiziert sein kann und ein Semantikschema wiederum auch auf mehrere Kontrollflussmuster (vgl. z. B. Tabelle 27) verweisen kann. Im Geschäftsprozessmodell darf ein Knoten für die eindeutige Beschreibung der Kontrollflusssemantik jedoch nur mit höchstens einem Semantikschema spezifiziert sein. Darüber hinaus dürfen die elementaren Bedingungen des Semantikschemas nur mit einem Kontrollflussmuster spezifiziert sein. Sofern eine elementare Bedingung auf mehrere Kontrollflussmuster verweist, muss eines dieser Kontrollflussmuster ausgewählt werden. Für das ausgewählte Kontrollflussmuster sind die entsprechenden Parameter des Kontrollflussmusters (vgl. Kapitel 4.1.2) festzulegen. Hierbei wird zwischen muster- und modellspezifischen Parametern differenziert. Beispielsweise wird die Unterscheidung am Kontrollflussmuster exklusive Auswahl (wcp_4) deutlich. Wenn für einen Knoten im Geschäftsprozessmodell ein Semantikschema ausgewählt wird, welches beispielsweise bei der Kontrollflusssemantik «Nach der Ausführung» mit einer elementaren Bedingung spezifiziert ist und diese auf das Kontrollflussmuster exklusive Auswahl verweist, dann müssen die Split-Funktion (Split), die Reihenfolge der Auswertungssequenz ($<_{XOR}$), die Kantenbedingungen (ArcCond) und die Standardkante (Default) definiert werden (vgl. Kapitel 4.1.2.1). Bei der Auswahl des Kontrollflussmusters muss die Split-Funktion immer den Funktionswert XOR besitzen, d. h., die Split-Funktion kann unabhängig von dem zu spezifizierenden Knoten mit dem Funktionswert XOR spezifiziert werden und ist somit nur von dem Muster (musterspezifische Parameter) abhängig. Dahingegen können die Reihenfolge der Auswertungssequenz ($<_{XOR}$), die Kantenbedingungen (ArcCond) und die Standardkante (Default), abhängig von dem zu spezifizierenden Knoten im Geschäftsprozessmodell (modellspezifische Parameter) variieren.

Im Folgenden werden zunächst die Anforderungen und darauf aufbauend das Vorgehen für die Beschreibung der Ablaufreihenfolge der Elemente in Geschäftsprozessmodellen definiert. Insbesondere wird auf die Spezifikation der muster- und modellspezifischen

Parameter jedes Kontrollflussmusters eingegangen. Abgerundet wird das Kapitel durch die Beschreibung der Ablaufreihenfolge der Elemente von ausgewählten Geschäftsprozessmodellen aus [DrKO17] und [RuQu12].

6.1 Anforderungen an die Geschäftsprozessmodelle

Für die Beschreibung der Ablaufreihenfolge der Elemente eines Geschäftsprozessmodells, muss

(1) das Geschäftsprozessmodell mit einer Geschäftsprozessmodellierungssprache erstellt worden sein, dessen Kontrollflusssemantik vollständig mit der Methode aus Kapitel 5 beschrieben wurde.

(2) das Geschäftsprozessmodell durch Knoten, Kanten und Beziehungen beschrieben werden können, so dass jedem Knoten jeweils genau ein Knotentyp und auch jeder Kante bzw. Beziehung jeweils genau ein Kantentyp zugeordnet werden kann.

(3) jeder Knoten mit genau einem Semantikschema des entsprechenden Knotentyps verknüpft und die Kontrollflusssemantik jeder elementaren Bedingung eines Semantikschemas mit genau einem Kontrollflussmuster beschrieben werden können.

Die erste Anforderung stellt sicher, dass die Ablaufreihenfolge der Elemente des Geschäftsprozessmodells mit der Kontrollflusssemantik der Geschäftsprozessmodellierungssprache beschrieben werden kann. In Kapitel 5 wurde die Kontrollflusssemantik von Knotentypen von Geschäftsprozessmodellierungssprachen definiert, so dass mit dieser Kontrollflusssemantik die Ablaufreihenfolge der Knoten in einem Geschäftsprozessmodell beschrieben werden kann. Für die Beschreibung der Ablaufreihenfolge der Elemente von Geschäftsprozessmodellen mit den Semantikschemas aus Kapitel 5 muss das Geschäftsprozessmodell durch

- Knoten (vgl. Klasse `G_Knoten` in Abbildung 62),
- Kanten (vgl. Klasse `G_Kante` in Abbildung 62) und
- Beziehungen (vgl. Klasse `G_Beziehung` in Abbildung 62)

beschrieben werden können, wie aus der zweiten Anforderung und Abbildung 62 hervorgeht.

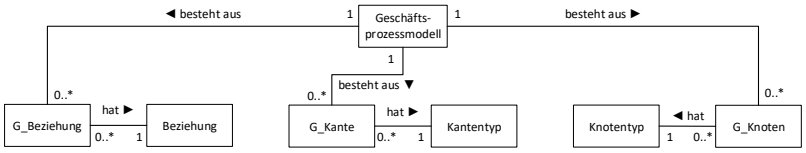


Abbildung 62: Model - Beschreibung der Bestandteile eines Geschäftsprozessmodells

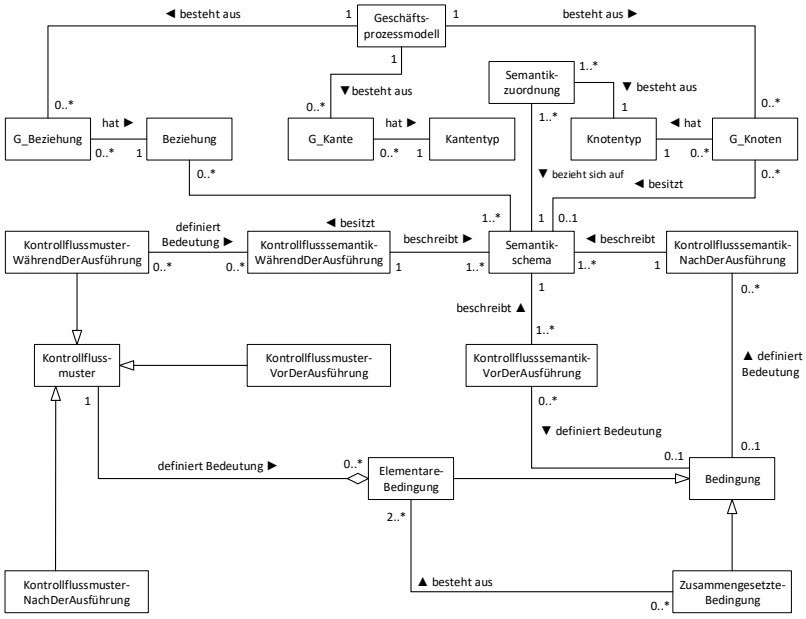


Abbildung 63: Modell - Beschreibung der Ablaufreihenfolge der Elemente eines Geschäftsprozessmodels

Ausgehend von dem zugeordneten Knotentyp der Geschäftsprozessmodellierungssprache zu den jeweiligen Knoten im Geschäftsprozessmodell kann jeweils ein Semantikschema einem Knoten zugeordnet werden. Die möglichen Semantikschemas für einen Knoten gehen aus der Semantikzuordnung zu den Knotentypen (vgl. Kapitel 5) hervor. Jedem Knoten im Geschäftsprozessmodell darf höchstens ein Semantikschema zugeordnet werden. Typischerweise besitzt jeder Knoten mindestens ein Semantikschema. Es gibt aber beispielsweise Knoten in Geschäftsprozessmodellen, die keine Kontrollflusssemantik besitzen, wie Textannotationen in BPMN. Im Unterschied zum Semantikschema einer Geschäftsprozessmodellierungssprache (vgl. Abbildung 30) muss jede elementare Bedingung des Semantikschemas mit genau einem Kontrollflussmuster spezifiziert werden. Sofern mehrere Kontrollflussmuster mit einer elementaren Bedingung verknüpft sind, muss eines der

verknüpften Kontrollflussmuster ausgewählt werden (vgl. Abbildung 63). Die Anforderungen zur Beschreibung der Ablaufreihenfolge der Elemente eines Geschäftsprozessmodells werden als Modell in Abbildung 63 zusammenfassend mit einem UML-Klassendiagramm [UML2.5.1] dargestellt. Nachfolgend wird das Vorgehen für die Beschreibung der Ablaufreihenfolge der Elemente von Geschäftsprozessmodellen vorgestellt. Für die Anwendung des Vorgehens müssen die oben beschriebenen Anforderungen erfüllt sein.

6.2 Vorgehen zur Beschreibung der Ablaufreihenfolge der Elemente eines Geschäftsprozessmodells

Das Vorgehen zur Beschreibung der Ablaufreihenfolge der Elemente eines Geschäftsprozessmodells wird mit dem BPMN-Modell in Abbildung 64 illustriert, um diese Beschreibung als Grundlage für die prototypische Realisierung des Software-Werkzeuges (vgl. Kapitel 7.2) einsetzen zu können.

Im ersten Schritt zur Beschreibung der Ablaufreihenfolge der Elemente von Geschäftsprozessmodellen muss eine Geschäftsprozessmodellierungssprache ausgewählt werden, dessen Kontrollflusssemantik mit der Methode aus Kapitel 5 beschrieben wurde (vgl. Anforderung 1). Hieraus resultieren die Knoten- und Kantentypen sowie die Semantikzuordnungen von Knotentypen zu den Semantikschemas der Geschäftsprozessmodellierungssprache. Im nächsten Schritt können die Knoten, Kanten und Beziehungen des Geschäftsprozessmodells definiert werden. Zudem wird jedem Knoten jeweils genau ein Knotentyp und jeder Kante bzw. Beziehung jeweils genau ein Kantentyp zugeordnet (vgl. Anforderung 2). Zur Veranschaulichung wurde beispielsweise das Geschäftsprozessmodell aus Abbildung 65 mit dem Modell aus Abbildung 63 in Tabelle 29 beschrieben. Im nächsten Schritt können die impliziten Abhängigkeiten zwischen den Knoten im Geschäftsprozessmodell mit den impliziten Kantentypen definiert werden. Dieser Schritt ist erforderlich, da es Abhängigkeiten zwischen Knoten in Geschäftsprozessmodellen geben kann, die nicht mit den Notationselementen der Geschäftsprozessmodellierungssprache visualisiert sind. Beispielsweise existieren implizite Abhängigkeiten zwischen BPMN-Linkereignissen (vgl. Abbildung 44). Es können solange implizite Abhängigkeiten definiert werden, bis alle Abhängigkeiten beschrieben sind. Abschließend wird jedem Knoten im Geschäftsprozessmodell ein Semantikschema zugeordnet, wobei für jede elementare Bedingung des Semantikschemas genau ein Kontrollflussmuster ausgewählt werden muss (vgl. Anforderung 3).

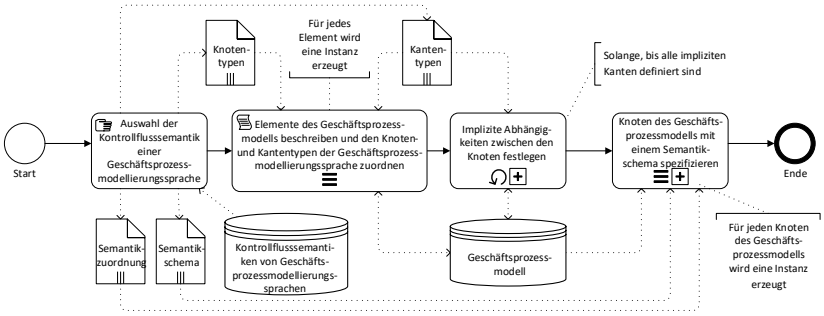


Abbildung 64: Vorgehen - Beschreibung der Ablaufreihenfolge der Elemente eines Geschäftsprozessmodells

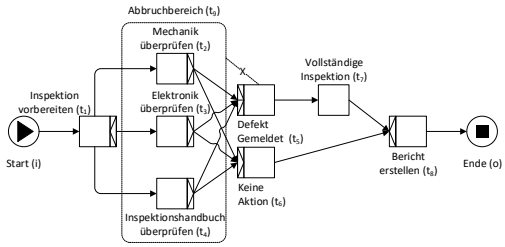


Abbildung 65: Beispiel - newYAWL - Inspektion

Klasse	Instanzen
Knoten	i, t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, t_9, o
Knotentypen	Startbedingung, Endbedingung, Aufgabe, Abbruchbereich, Blockierungsbereich, Beendigungsbereich
G_Knoten	(i, Startbedingung), (t_1, Aufgabe), (t_2, Aufgabe), (t_3, Aufgabe), (t_4, Aufgabe), (t_5, Aufgabe), (t_6, Aufgabe), (t_7, Aufgabe), (t_8, Aufgabe), (t_9, Abbruchbereich), (o, Endbedingung),
Kanten	(i, t_1), (t_1, t_2), (t_1, t_3), (t_1, t_4), (t_2, t_5), (t_2, t_6), (t_3, t_5), (t_3, t_6), (t_4, t_5), (t_4, t_6), (t_5, t_7), (t_7, t_8), (t_6, t_8), (t_8, o)
Kantentypen	Kontrollfluss (Sq), Abbruchkante (Ak), Beendigungskante (Bk), Deaktivierungskante (Dk), Blockierungskante (BIK)
G_Kante	(i, t_1, Sq), (t_1, t_2, Sq), (t_1, t_3, Sq), (t_1, t_4, Sq), (t_2, t_5, Sq), (t_2, t_6, Sq), (t_3, t_5, Sq), (t_3, t_6, Sq), (t_4, t_5, Sq), (t_4, t_6, Sq), (t_5, t_7, Sq), (t_7, t_8, Sq), (t_6, t_8, Sq), (t_8, o, Sq)
Beziehung	(t_5, t_9)
G_Beziehung	(t_5, t_9, Ak)

Tabelle 29: Beispiel - newYAWL - Beschreibung von Abbildung 65 mit dem Modell aus Abbildung 63

Mit dem Teilprozess «Implizite Abhängigkeiten zwischen Knoten festlegen» können die impliziten Abhängigkeiten zwischen Knoten im Geschäftsprozessmodell beschrieben werden.

Zur Definition einer impliziten Abhängigkeit wird im ersten Schritt ein impliziter Kantentyp ausgewählt. Nachdem ein impliziter Kantentyp aus der entsprechenden Geschäftsprozessmodellierungssprache ausgewählt wurde, muss der Start- (x) und Endknoten (y) ausgewählt werden, der die beiden Knoten mit dem ausgewählten impliziten Kantentyp verbindet. Abschließend wird das Geschäftsprozessmodell um diese Kante (x,y), in `G_Kante`, mit dem Kantentyp erweitert. Der Teilprozess wird solange wiederholt, bis alle impliziten Abhängigkeiten definiert wurden. Sofern beispielsweise keine impliziten Kanten existieren, wie bei den Geschäftsprozessmodellierungssprachen EPK, Petri-Netzen oder YAWL bzw. newYAWL, ist die Schleifenbedingung des Teilprozesses erfüllt, so dass der Teilprozess übersprungen wird.

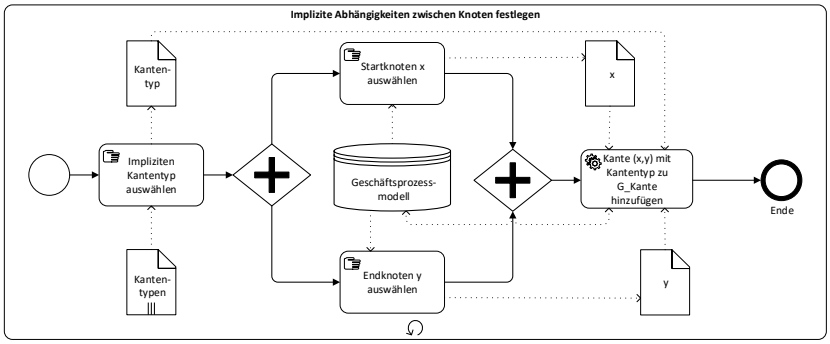


Abbildung 66: Vorgehen - Implizite Abhängigkeiten zwischen Knoten festlegen

Abgerundet wird das Vorgehen zur Beschreibung der Ablaufreihenfolge der Elemente eines Geschäftsprozessmodells mit der Zuweisung eines der möglichen Semantikschemas zu den einzelnen Knoten im Geschäftsprozessmodell. Damit einhergehend sind, die Auswahl der Kontrollflussmuster sowie die Spezifikation der muster- und modellspezifischen Parameter. Der Teilprozess aus Abbildung 67 wird für jeden Knoten des Geschäftsprozessmodells ausgeführt und beginnt mit der Ermittlung der möglichen Semantikschemas des betrachteten Knotens. Die Grundgesamtheit der möglichen Semantikschemas für einen Knoten wird durch die Semantikzuordnung zu dem Knotentyp definiert, die aber aufgrund von Rahmenbedingungen reduziert werden kann. Für einen Knoten werden die Rahmenbedingungen durch die ein- und ausgehende Kantenanzahl unter Berücksichtigung des entsprechenden Kantentyps festgelegt. Die Identifizierung der Rahmenbedingungen und der Abgleich mit der Grundgesamtheit könnte durch ein IT-System unterstützt werden. Aus diesem Grund wurde in Abbildung 67 die BPMN-Aufgabe zur Ermittlung der Semantikschemas mit dem Aufgabentyp Skript gekennzeichnet. Die BPMN-Aufgabe aus Abbildung 49a besitzt beispielsweise die folgenden Rahmenbedingungen:

- Kontrollflusssemantik «Vor der Ausführung»: Eine elementare Bedingung mit dem Kantentyp Sequenzfluss und der Kantenanzahl 1.
- Kontrollflusssemantik «Nach der Ausführung»: Eine elementare Bedingung mit dem Kantentyp Sequenzfluss und der Kantenanzahl 1.
- Beziehung: keine Beziehungen

Aus Tabelle 13 geht hervor, dass nur ein Semantikschema für eine BPMN-Aufgabe ermittelt werden konnte, das auf die Rahmenbedingungen der Kontrollflusssemantik «Vor der Ausführung» und «Nach der Ausführung» sowie die Beziehungen überprüft werden muss. Das Semantikschema der BPMN-Aufgabe wurde jedoch in aggregierter Form definiert, so dass eigentlich 6.561 Semantikschemas für die BPMN-Aufgabe existieren, die auf die Rahmenbedingungen hin überprüft werden müssen. Die 6.561 Semantikschemas sind die möglichen Kombinationen der Bedingungen 29 und 30 (vgl. Tabelle 11), die bei einer BPMN-Aufgabe als Vor- bzw. Nachbedingung fungieren können. Diese beiden Bedingungen enthalten mit einer formulierten EBNF-Grammatik jeweils 81 Bedingungen. Auf die oben definierten Rahmenbedingungen trifft trotzdem nur genau ein Semantikschema zu:

- Kontrollflusssemantik «Vor der Ausführung»: Bed4
- Kontrollflusssemantik «Während der Ausführung»: wcp_{13}, wcp_{14}, wcp_{17}, wcp_{21}, wcp_{34}, wcp_{36}, wcp_{40}, $wcp_{BPMNEreignisTeilprozessUnterbrechendesEreignis}$, $wcp_{BPMNEreignisTeilprozessNichtUnterbrechendesEreignis}$, $wcp_{BPMNAngeheftetesUnterbrechendesEreignis}$, $wcp_{BPMNAngeheftetesNichtUnterbrechendesEreignis}$
- Kontrollflusssemantik «Nach der Ausführung»: Bed4
- Beziehung: Bez1

Ein weiteres Beispiel ist die BPMN-Aufgabe aus Abbildung 53, deren Kontrollflusssemantik «Vor der Ausführung» aus einer zusammengesetzten Bedingung besteht, welche sich aus drei elementaren Bedingungen zusammensetzt. Die Kontrollflusssemantik «Nach der Ausführung» wird durch eine elementare Bedingung definiert. Die Rahmenbedingungen für das zu ermittelnde Semantikschema werden wie folgt definiert:

- Kontrollflusssemantik «Vor der Ausführung»: Eine zusammengesetzte Bedingung, die aus den drei folgenden elementaren Bedingungen besteht, d. h.
  - eine elementare Bedingung mit dem Kantentyp Sequenzfluss und der Kantenanzahl 2,
  - eine elementare Bedingung mit dem Kantentyp Nachrichtenfluss und der Kantenanzahl 2 sowie
  - eine elementare Bedingung mit dem Kantentyp gerichtete Assoziation und der Kantenanzahl 2.

– Kontrollflusssemantik «Nach der Ausführung»: Eine elementare Bedingung mit dem Kantentyp Sequenzfluss und der Kantenanzahl 1.
– Beziehung: Keine Beziehungen.

Von den 6.561 möglichen Semantikschemas für BPMN-Aufgaben erfüllt nur das folgende Semantikschema die erforderlichen Rahmenbedingungen:

– Kontrollflusssemantik «Vor der Ausführung»: Bed4 AND Bed15 AND Bed18
– Kontrollflusssemantik «Während der Ausführung»: wcp_{13}, wcp_{14}, wcp_{17}, wcp_{21}, wcp_{34}, wcp_{36}, wcp_{40}, $wcp_{BPMNEreignisTeilprozessUnterbrechendesEreignis}$, $wcp_{BPMNEreignisTeilprozessNichtUnterbrechendesEreignis}$, $wcp_{BPMNAngeheftetesUnterbrechendesEreignis}$, $wcp_{BPMNAngeheftetesNichtUnterbrechendesEreignis}$
– Kontrollflusssemantik «Nach der Ausführung»: Bed4
– Beziehung: Bez1

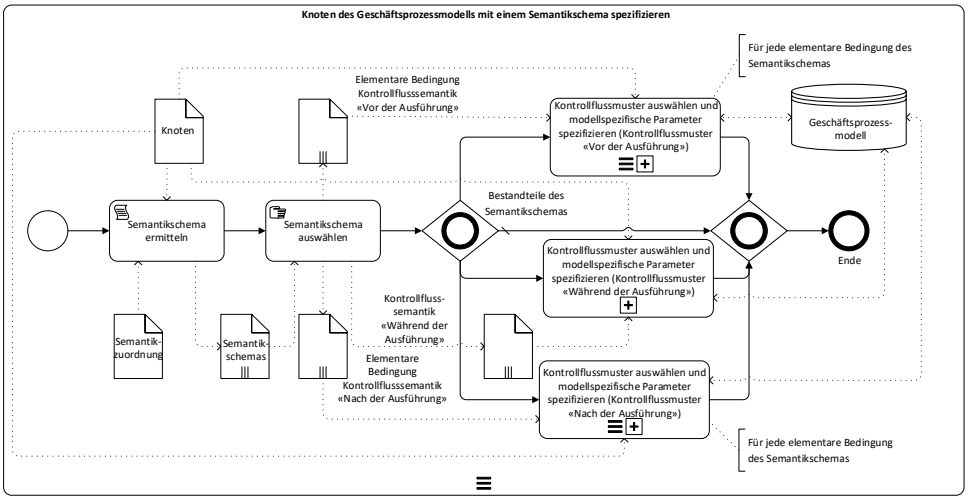


Abbildung 67: Vorgehen - Knoten des Geschäftsprozessmodells mit einem Semantikschema spezifizieren

Nachdem die möglichen Semantikschemas für einen Knoten im Geschäftsprozessmodell ermittelt wurden, muss bei mehreren zutreffenden Semantikschemas genau ein Semantikschema zur Beschreibung der Kontrollflusssemantik des Knotens ausgewählt werden. Abschließend können die Kontrollflussmuster bestimmt sowie deren muster- und modellspezifischen Parameter spezifiziert werden. Dabei muss für jede elementare Bedingung der Kontrollflusssemantik «Vor der Ausführung» und «Nach der Ausführung» genau ein Kontrollflussmuster ausgewählt werden. Die identifizierten elementaren Bedingungen der

Kontrollflusssemantik «Vor der Ausführung» und «Nach der Ausführung» werden in Abbildung 67 durch die Datenobjekte

- Elementare Bedingung Kontrollflusssemantik «Vor der Ausführung» und
- Elementare Bedingung Kontrollflusssemantik «Nach der Ausführung»

illustriert. Die möglichen Kontrollflussmuster der Kontrollflusssemantik «Während der Ausführung» werden in Abbildung 67 durch das Datenobjekt Kontrollflusssemantik «Während der Ausführung» dargestellt. Die Teilprozesse zur Auswahl der Kontrollflussmuster und die Spezifikation der jeweiligen notwenigen modellspezifischen Parameter der Kontrollflussmuster werden

- für die Kontrollflusssemantik «Vor der Ausführung» in Abbildung 68,
- für die Kontrollflusssemantik «Während der Ausführung» in Abbildung 76 und
- für die Kontrollflusssemantik «Nach der Ausführung» in Abbildung 72

aufgezeigt. Für die elementaren Bedingungen der Kontrollflusssemantik «Vor der Ausführung» kann jeweils nur eines der möglichen Kontrollflussmuster ausgewählt werden. Die möglichen auszuwählenden Kontrollflussmuster werden durch die zugrundeliegende elementare Bedingung definiert. Sofern die elementare Bedingung auf mehrere Kontrollflussmuster verweist, muss genau eines dieser möglichen Kontrollflussmuster ausgewählt werden, wie auch an dem exklusiven BPMN-Gateway des Teilprozesses in Abbildung 68 deutlich wird. Abhängig von dem ausgewählten Kontrollflussmuster müssen keine Parameter oder modell- und/oder musterspezifische Parameter spezifiziert werden. Die zu spezifizierenden Parameter der jeweiligen Kontrollflussmuster wurden in Kapitel 4.1.2 mit einer newYAWL-Spezifikation definiert. Für die Beschreibung der Kontrollflusssemantik «Vor der Ausführung» werden die Parameter Join, Thresh, Block, Rem und ThreadIn der newYAWL-Spezifikation benötigt. Für die Festlegung des strukturierten Knotens der Kontrollflussmuster wcp_7, wcp_9, wcp_{30} wird die partielle Funktion Struc eingeführt, die vom betrachteten Knoten auf den entsprechenden strukturierten Knoten verweist, mit $\mathrm{Struc}: \mathrm{Knoten} \nrightarrow \mathrm{Knoten}$. Mit Knoten wird die Menge der Knoten im Geschäftsprozessmodell beschrieben. Zudem wird eine partielle Funktion verwendet, da nicht jeder Knoten im Geschäftsprozessmodell einen strukturierten Knoten besitzt. Eine partielle Funktion ist eine rechtseindeutige, aber keine linkstotale Relation, d. h. eine Relation, bei der kein Element des Definitionsbereichs auf mehr als ein Element des Wertebereichs abgebildet wird [Hoff18].

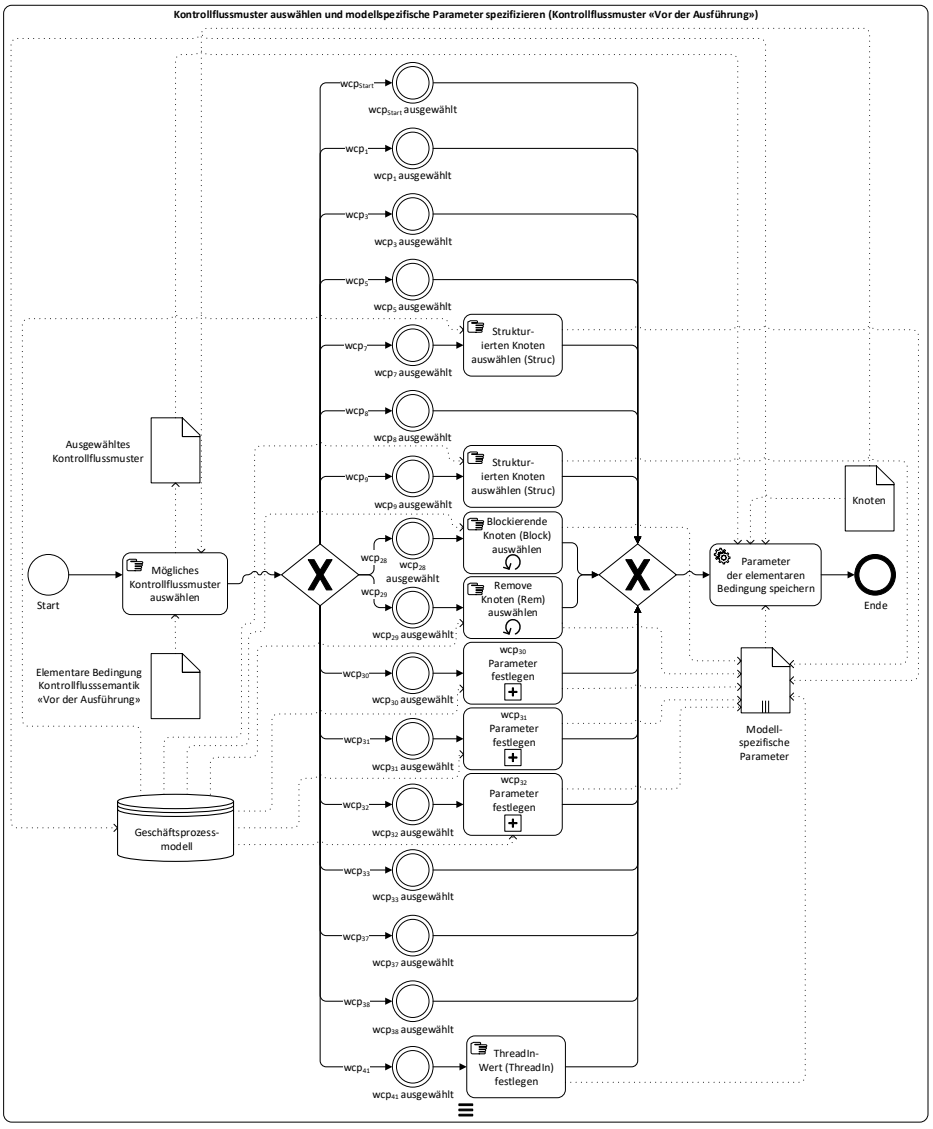


Abbildung 68: Vorgehen - Kontrollflussmuster auswählen und modellspezifische Parameter spezifizieren (Kontrollflussmuster «Vor der Ausführung»)

In Tabelle 30 werden die in Kapitel 4.1.2 definierten Parameter der Kontrollflussmuster zusammenfassend in muster- und modellspezifische Parameter untergliedert. Dabei ist $v \in \text{Knoten}$, der zu spezifizierende Knoten, dem das Kontrollflussmuster zugewiesen wird. Darüber hinaus werden in Abbildung 68 die Auswahl des Kontrollflussmusters und die

Spezifikation der modellspezifischen Parameter für die Kontrollflussmuster «Vor der Ausführung» illustriert. Die Spezifikation der modellspezifischen Parameter der Kontrollflussmuster strukturierte partielle Zusammenführung (wcp_{30}), blockierende partielle Zusammenführung (wcp_{31}) und abbrechende partielle Zusammenführung (wcp_{32}) wurde aus Gründen der Übersichtlichkeit in die Teilprozesse ausgelagert (siehe Abbildung 69, bis Abbildung 71).

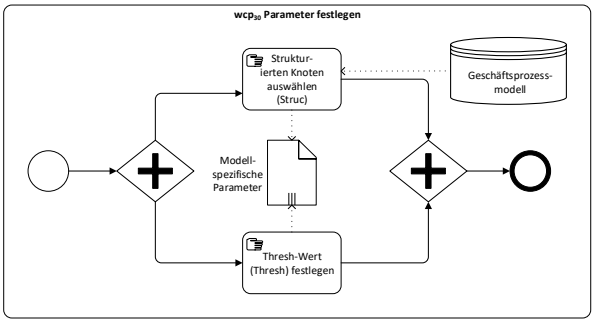


Abbildung 69: Vorgehen - Modellspezifische Parameter für das Kontrollflussmuster wcp_{30} festlegen

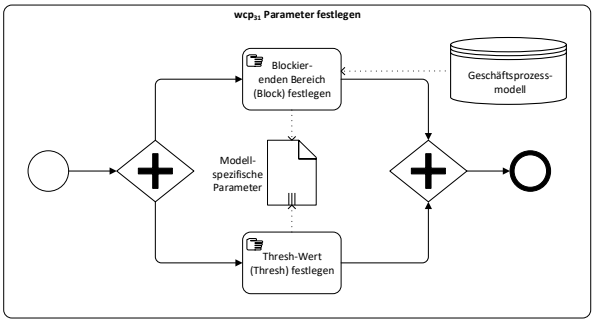


Abbildung 70: Vorgehen - Modellspezifische Parameter für das Kontrollflussmuster wcp_{31} festlegen

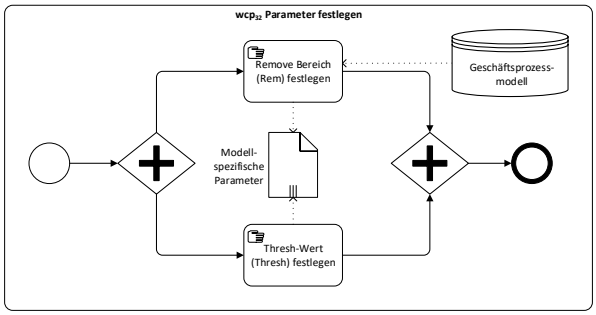


Abbildung 71: Vorgehen - Modellspezifische Parameter für das Kontrollflussmuster wcp_{32} festlegen

Kontroll-flussmuster	Name des Kontrollflussmusters	Musterspezifische Parameter	Modellspezifische Parameter
wcp_1	Sequenz	-	-
wcp_3	Synchronisation	Join(v) = AND	-
wcp_5	Einfache Zusammenführung	Join(v) = XOR	-
wcp_7	Strukturierte synchronisierte Zusammenführung	Join(v) = OR	Struc(v)
wcp_8	Mehrfach Zusammenführung	Join(v) = XOR	-
wcp_9	Strukturierter Diskriminator	Join(v) = PJOIN, Thresh(v) = 1	Struc(v)
wcp_{28}	Blockierender Diskriminator	Join(v) = PJOIN, Thresh(v) = 1	Block(v)
wcp_{29}	Abbrechender Diskriminator	Join(v) = PJOIN, Thresh(v) = 1	Rem(v)
wcp_{30}	Strukturierte partielle Zusammenführung	Join(v) = PJOIN	Struc(v), Thresh(v)
wcp_{31}	Blockierende partielle Zusammenführung	Join(v) = PJOIN	Struc(v), Block(v)
wcp_{32}	Abbrechende partielle Zusammenführung	Join(v) = PJOIN	Struc(v), Rem(v)
wcp_{33}	Verallgemeinerte Synchronisation	Join(v) = AND	-
wcp_{37}	Lokale synchronisierte Zusammenführung	Join(v) = OR	-
wcp_{38}	Verallg. synchronisierte Zusammenführung	Join(v) = OR	-
wcp_{41}	Thread-Zusammenführung	Join(v) = Thread	ThreadIn(v)
wcp_{Start}	Startmuster	-	-

Tabelle 30: Modell - Muster- und modellspezifische Parameter für die Kontrollflussmuster «Vor der Ausführung»

Nachdem die zu spezifizierenden muster- und modellspezifischen Parameter für die Kontrollflussmuster «Vor der Ausführung» definiert wurden, soll im Folgenden auf die Spezifikation der muster- und modellspezifischen Parameter für die Kontrollflussmuster «Nach der Ausführung» eingegangen werden. Hierbei gilt, wie auch bei der Kontrollflusssemantik «Vor der Ausführung», sofern die elementare Bedingung auf mehrere Kontrollflussmuster

verweist, muss genau eines der möglichen Kontrollflussmuster für den Knoten ausgewählt werden. Deutlich wird dies an dem exklusiven BPMN-Gateway des Teilprozesses in Abbildung 72. Abhängig von dem ausgewählten Kontrollflussmuster müssen keine Parameter oder modell- und/oder musterspezifische Parameter spezifiziert werden. Die möglichen zu spezifizierenden Parameter wurden aus der newYAWL-Spezifikation abgeleitet, d. h. Split, Rem, ArcCond, $<_{XOR}$, Default und ThreadOut.

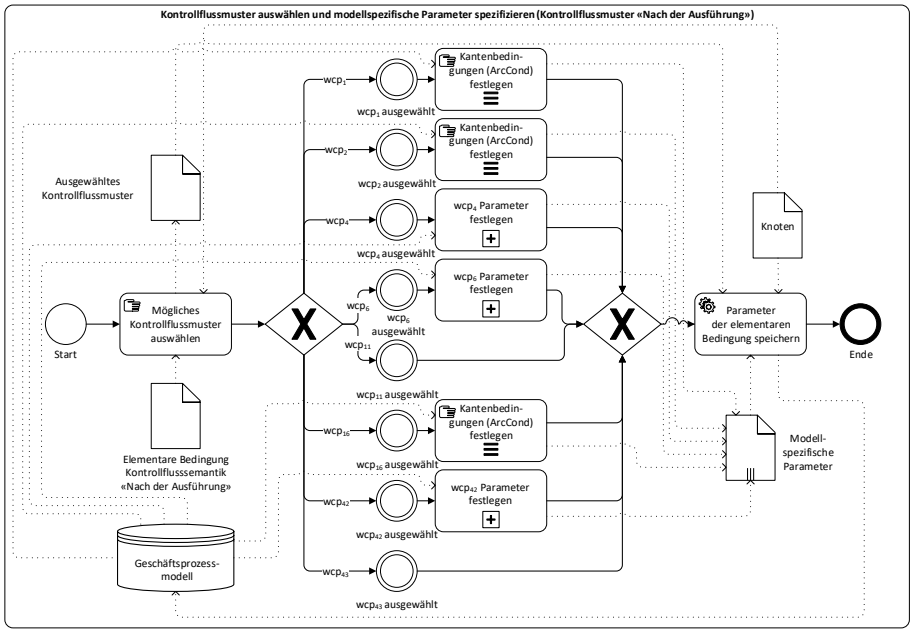


Abbildung 72: Vorgehen - Kontrollflussmuster auswählen und modellspezifische Parameter spezifizieren (Kontrollflussmuster «Nach der Ausführung»)

In Abbildung 72 werden die zu spezifizierenden modell- und musterspezifischen Parameter für die Kontrollflussmuster «Nach der Ausführung» zusammenfassend beschrieben. Dabei ist $v \in Knoten$ in Tabelle 31, der zu spezifizierende Knoten, dem das Kontrollflussmuster zugewiesen wird. Für jedes Kontrollflussmuster «Nach der Ausführung», mit Ausnahme der Endmuster (wcp_{11} und wcp_{43}), müssen modellspezifische Parameter spezifiziert werden, da eine ausgehende Kante eine Kantenbedingung (ArcCond) besitzen kann. Die Kantenbedingungen (ArcCond) beschreiben zusätzlich zum Kontrollflussmuster eine prüfende Eigenschaft, ob der Kontrollfluss an dieser Kante fortgesetzt werden soll. Sofern eine Kante keine spezifische Kantenbedingung besitzt, wird der Funktionswert von ArcCond auf `true` festgelegt. Die Kontrollflussmuster «Nach der Ausführung» werden somit im Vergleich zu den

vorgestellten Kontrollflussmustern in Kapitel 4.1.2 um Kantenbedingungen (ArcCond) erweitert. Die Erweiterung wurde eingeführt, da es beispielsweise den Kantentyp Sequenzfluss in BPMN gibt und für den Kantentyp Kantenbedingungen definiert werden können. Weitere modellspezifische Parameter müssen bei den Kontrollflussmustern exklusive Auswahl (wcp_4), Mehrfachauswahl (wcp_6) und Thread-Aufspaltung (wcp_{42}) festgelegt werden, wie aus Tabelle 31 hervorgeht und in Abbildung 73, Abbildung 74 und Abbildung 75 illustriert wird.

Kontrollfluss-muster	Name des Kontrollflussmusters	Musterspezifische Parameter	Modellspezifische Parameter
wcp_1	Sequenz	-	ArcCond
wcp_2	Parallele Aufspaltung	Split(v) = AND	ArcCond
wcp_4	Exklusive Auswahl	Split(v) = XOR	$<_{XOR}$, ArcCond, Default
wcp_6	Mehrfachauswahl	Split(v) = OR	ArcCond
wcp_{11}	Implizite Beendigung	-	-
wcp_{16}	Aufgeschobene Auswahl	-	ArcCond
wcp_{42}	Thread-Aufspaltung	Split(v) = THREAD	ArcCond, ThreadOut(v)
wcp_{43}	Explizite Beendigung	Rem(v) = Knoten	-

Tabelle 31: Modell - Muster- und modellspezifische Parameter für die Kontrollflusssemantik «Nach der Ausführung»

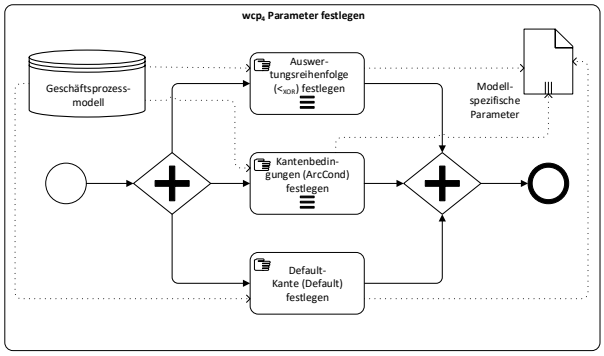


Abbildung 73: Vorgehen - Modellspezifische Parameter für das Kontrollflussmuster wcp_4 festlegen

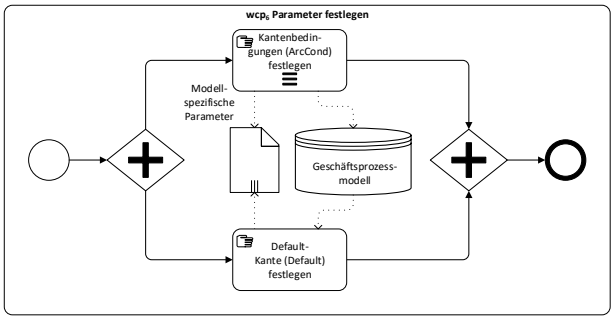


Abbildung 74: Vorgehen - Modellspezifische Parameter für das Kontrollflussmuster wcp_6 festlegen

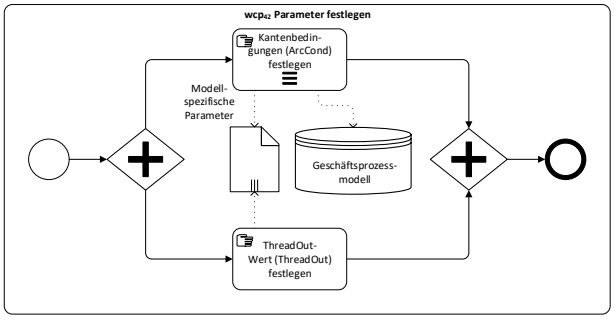


Abbildung 75: Vorgehen - Modellspezifische Parameter für das Kontrollflussmuster wcp_{42} festlegen

Nachdem die muster- und modellspezifischen Parameter der Kontrollflussmuster «Nach der Ausführung» betrachtet wurden, werden im Folgenden die muster- und modellspezifischen Parameter der Kontrollflussmuster «Während der Ausführung» definiert. Im Unterschied zu den elementaren Bedingungen der Kontrollflusssemantik «Vor der Ausführung» und «Nach der Ausführung» muss bei der Kontrollflusssemantik «Während der Ausführung» keines der möglichen Kontrollflussmuster ausgewählt werden. In Tabelle 32 wird für jedes Kontrollflussmuster «Während der Ausführung» die Aufgliederung in modell- und musterspezifische Parameter beschrieben, wobei $v \in \text{Knoten}$ der zu spezifizierende Knoten ist, d. h. dem das Kontrollflussmuster zugewiesen wird. Die zu spezifizierenden Parameter der jeweiligen Kontrollflussmuster wurden in Kapitel 4.1.2 mit einer newYAWL-Spezifikation definiert. Die zu spezifizierenden Parameter für die Kontrollflussmuster «Während der Ausführung» werden aus der newYAWL-Spezifikation abgeleitet, d. h. ThreadOut, Nofi, Disable, Rem, Trig, Pre und Post. Zusätzlich wird für die Festlegung der erforderlichen ausgeführten Knoten für das Kontrollflussmuster Meilenstein (wcp_{18}) die partielle Funktion Meilenstein eingeführt, die jeweils vom betrachteten Knoten auf ein

Element der Potenzmenge bzw. die ausgeführten Knoten des Geschäftsprozessmodells verweist, mit $\text{Meilenstein}: \text{Knoten} \rightharpoonup \mathbb{P}(\text{Knoten})$. Für die Spezifikation der Parameter des Kontrollflussmusters Kritischer Bereich (wcp_{39}) müssen Bereiche mit Knoten definiert werden, d. h. $\text{KritischerBereich}_i \subseteq \mathbb{P}(\text{Knoten})$, so dass keine zwei Knoten eines kritischen Bereichs gleichzeitig ausgeführt werden. Jeder kritische Bereich $i \in \mathbb{N}$ besteht aus einer Teilmenge der Potenzmenge der Knoten des Geschäftsprozessmodells. Für die Kontrollflussmuster verschachtelte parallele Ausführung (wcp_{17}) und verschachtelte Ausführung (wcp_{40}) müssen ebenfalls Bereiche definiert werden, wobei die Knoten innerhalb eines Bereichs i nur nacheinander ausgeführt werden können und jeder Knoten des Bereichs ausgeführt werden muss, d. h. $\text{verschachtelteAusführung}_i \in \mathbb{P}(\text{Knoten})$ mit $i \in \mathbb{N}$. Für die Spezifikation der benutzerdefinierten Kontrollflussmuster werden folgende partielle Funktionen benötigt:

- Angeheftetes unterbrechendes BPMN-Ereignis ($wcp_{\text{BPMNAngeheftetesNichtUnterbrechendesEreignis}}$): $\text{BPMNAngeheftetesUnterbrechendesEreignis}: \text{Knoten} \rightharpoonup \mathbb{P}(\text{Knoten})$
- Angeheftetes nicht-unterbrechendes BPMN-Ereignis ($wcp_{\text{BPMNAngeheftetesNichtUnterbrechendesEreignis}}$): $\text{BPMNAngeheftetesNichtUnterbrechendesEreignis}: \text{Knoten} \rightharpoonup \mathbb{P}(\text{Knoten})$
- Teilprozess ($wcp_{\text{Teilprozess}}$): $\text{Teilprozess}: \text{Knoten} \rightharpoonup \text{Prozessmodell}$
- Ereignis-Teilprozess mit unterbrechendem Startereignis ($wcp_{\text{BPMNEreignisTeilprozessUnterbrechendesEreignis}}$): $\text{BPMNEreignisTeilprozessUnterbrechendesEreignis}: \text{Knoten} \rightharpoonup \text{Prozessmodell}$
- Ereignis-Teilprozess mit nicht-unterbrechendem Startereignis ($wcp_{\text{BPMNEreignisTeilprozessNichtUnterbrechendesEreignis}}$): $\text{BPMNEreignisTeilprozessNichtUnterbrechendesEreignis}: \text{Knoten} \rightharpoonup \text{Prozessmodell}$

Im Vergleich zu den elementaren Bedingungen der Kontrollflusssemantik «Vor der Ausführung» und «Nach der Ausführung» kann die Kontrollflusssemantik «Während der Ausführung» durch mehrere Kontrollflussmuster beschrieben werden. Die Menge der Kontrollflussmuster «Während der Ausführung» untergliedert sich in Teilmengen, wobei aus jeder Teilmenge maximal eines der möglichen Kontrollflussmuster ausgewählt werden kann, mit Ausnahme der benutzerdefinierten Kontrollflussmuster. Eine Teilmenge der Menge Kontrollflussmuster «Während der Ausführung» ist eine Menge von Kontrollflussmustern, die sich gegenseitig ausschließt und einem Knoten somit nicht gleichzeitig zugewiesen werden kann. Beispielsweise können einem Knoten nicht mehrere Kontrollflussmuster aus der Teilmenge Mehrfachinstanzmuster zugeordnet werden. Ein weiteres Beispiel sind die

Abbruchmuster, sodass ein Knoten in einem Geschäftsprozessmodell nicht gleichzeitig mehrere Abbruchmuster besitzen kann. Die definierten Teilmengen sind: Mehrfachinstanzmuster ($WCP_{Mehrfachinstanz}$), Abbruchmuster ($WCP_{Abbruch}$), Triggermuster ($WCP_{Trigger}$), Meilenstein (wcp_{18}), strukturierte Schleife (wcp_{21}), Mehrfachinstanzelement beenden (wcp_{27}), kritischer Bereich (wcp_{39}), verschachtelte Bereichsmuster ($WCP_{VerschachtelteAusführung}$) und benutzerdefinierte Kontrollflussmuster ($WCP_{BenutzerdefinierteMuster}$). Die Auswahl der Kontrollflussmuster wird in Abbildung 76 illustriert. Die zu spezifizierenden modellspezifischen Parameter werden nach den Teilmengen gruppiert und detailliert in Abbildung 77 bis Abbildung 84 beschrieben. In Tabelle 32 werden die zu spezifizierenden modell- und musterspezifischen Parameter zusammengefasst.

Kontrollflussmuster	Musterspezifische Parameter	Modellspezifische Parameter
wcp_{12}	-	ThreadOut(v)
wcp_{13}	Nofi(v) = (min, max, th, static, non-canceling)	min = max = th
wcp_{14}	Nofi(v) = (min, max, th, static, non-canceling)	min = max = th
wcp_{15}	Nofi(v) = (min, max, th, dynamic, non-canceling)	min = max = th während der Laufzeit spezifizieren
wcp_{17}	-	verschachteleteAusführung(v)
wcp_{18}	-	Meilenstein(v)
wcp_{19}	-	Rem(v)
wcp_{20}	-	Rem(v)
wcp_{21}	-	PreTest(v) und/oder PostTest(v)
wcp_{23}	-	Trig(v)
wcp_{24}	Persist = Persit {v}	Trig(v)
wcp_{25}	-	Rem(v)
wcp_{26}	-	Rem(v)
wcp_{27}		Comp(v)
wcp_{34}	Nofi(v) = (min, max, th, static, non-canceling)	min, max, th
wcp_{35}	Nofi(v) = (min, max, th, static, canceling)	min, max, th
wcp_{36}	Nofi(v) = (min, max, th, dynamic, non-canceling)	min = max = th Disable
wcp_{39}	-	$KritischerBereich_i$(v)
wcp_{40}	-	$VerschachteleteAusführung_i$(v)
$wcp_{Teilprozess}$	-	Teilprozess(v)
$wcp_{BPMNAngeheftetes-UnterbrechendesEreignis}$	-	BPMNAngeheftetes-UnterbrechendesEreignis(v)
$wcp_{BPMNAngeheftetes-NichtUnterbrechendesEreignis}$	-	BPMNAngeheftetesNicht-UnterbrechendesEreignis(v)

Kontrollflussmuster	Musterspezifische Parameter	Modellspezifische Parameter
$wcp_{BPMNEreignisTeilprozess-UnterbrechendesEreignis}$	-	BPMNEreignisTeilprozess-UnterbrechendesEreignis
$wcp_{BPMNEreignisTeilprozess-NichtUnterbrechendesEreignis}$	-	BPMNEreignisTeilprozess-NichtUnterbrechendesEreignis(v)

Tabelle 32: Modell - Muster- und modellspezifische Parameter für die Kontrollflussmuster «Während der Ausführung»

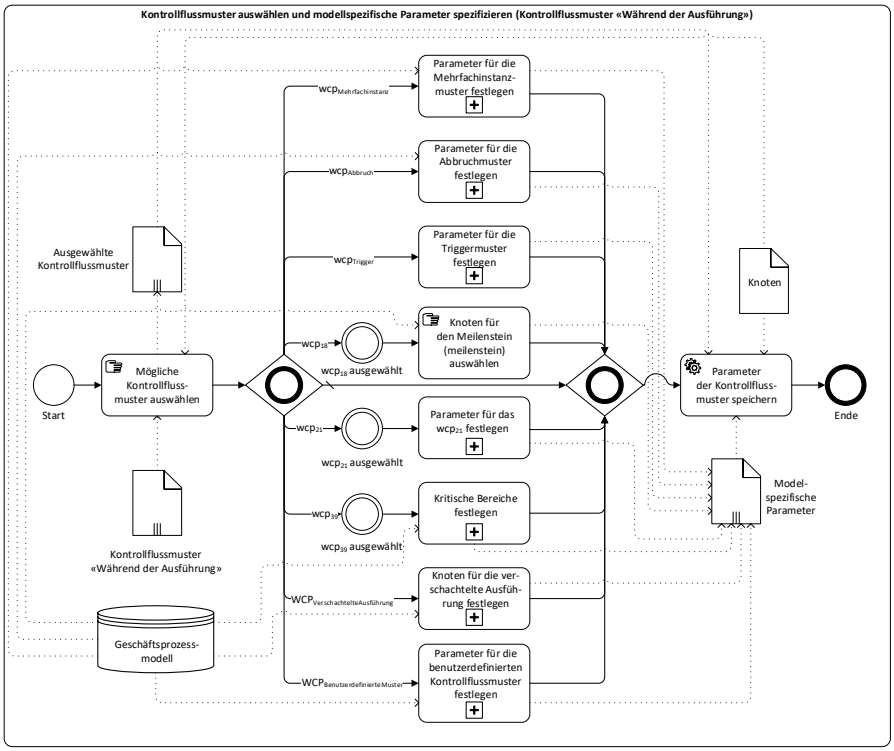


Abbildung 76: Vorgehen - Kontrollflussmuster auswählen und modellspezifische Parameter spezifizieren (Kontrollflussmuster «Während der Ausführung»)

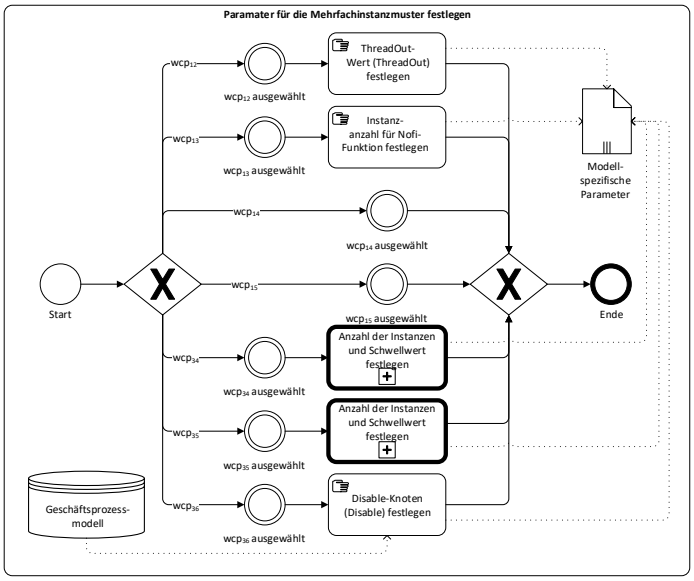


Abbildung 77: Vorgehen - Modellspezifische Parameter für die Kontrollflussmuster $WCP_{Mehrfachinstanz}$ festlegen

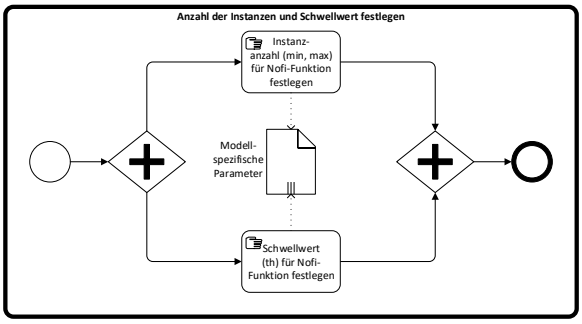


Abbildung 78: Vorgehen - Modellspezifische Parameter für die Kontrollflussmuster wcp_{34} und wcp_{35} festlegen

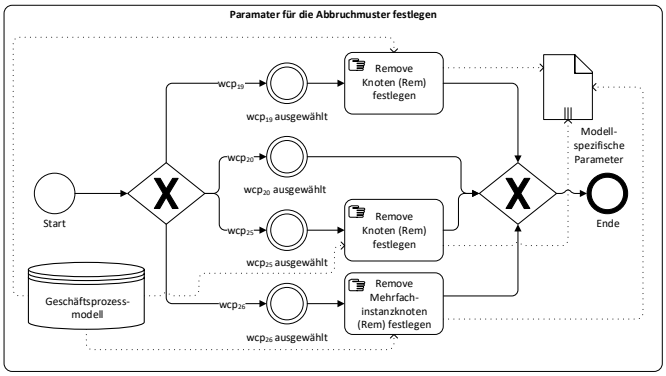


Abbildung 79: Vorgehen - Modellspezifische Parameter für die Kontrollflussmuster $WCP_{Abbruch}$ festlegen

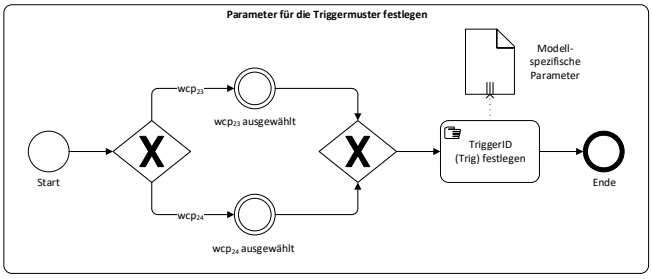


Abbildung 80: Vorgehen - Modellspezifische Parameter für die Kontrollflussmuster $WCP_{Trigger}$ festlegen

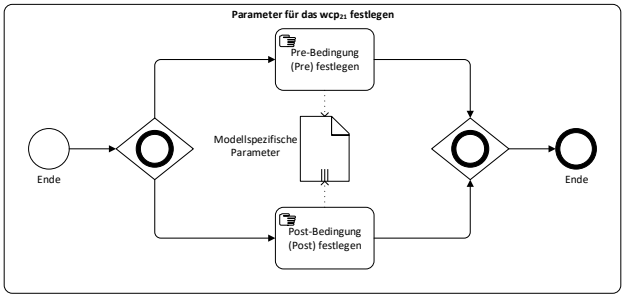


Abbildung 81: Vorgehen - Modellspezifische Parameter für das Kontrollflussmuster wcp_{21} festlegen

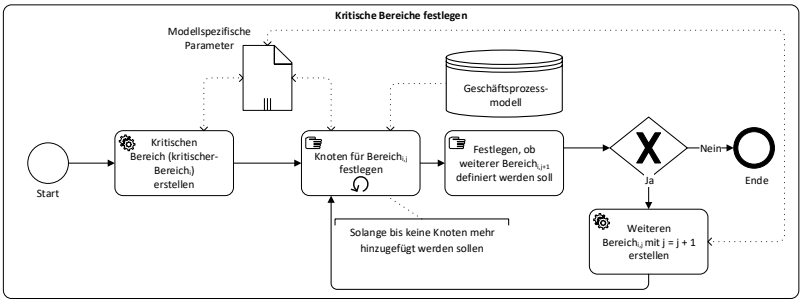


Abbildung 82: Vorgehen - Modellspezifische Parameter für das Kontrollflussmuster wcp_{39} festlegen

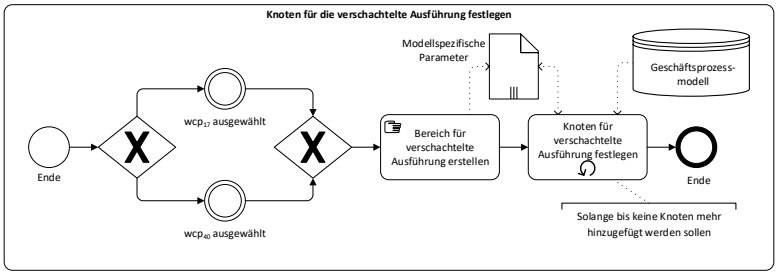


Abbildung 83: Vorgehen - Modellspezifische Parameter für die Kontrollflussmuster wcp_{17} und wcp_{40} festlegen

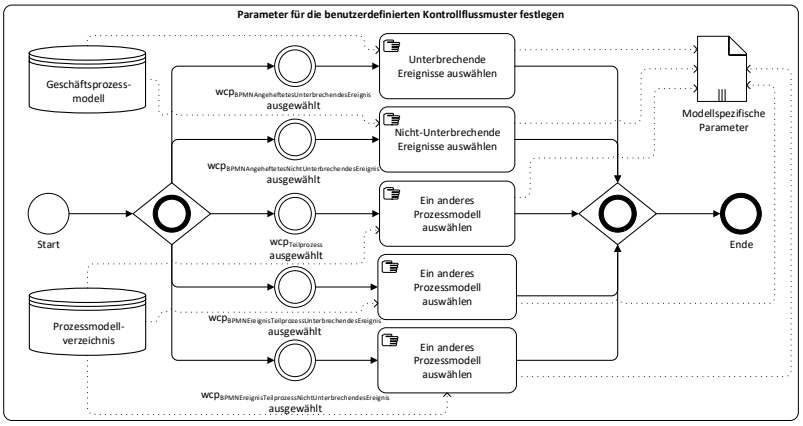


Abbildung 84: Vorgehen - Modellspezifische Parameter für die Kontrollflussmuster $WCP_{BenutzerdefinierteMuster}$ festlegen

6.3 Beispielhafte Anwendung des Vorgehens

Im vorherigen Kapitel wurde das Vorgehen zur Beschreibung der Ablaufreihenfolge von Elementen in Geschäftsprozessmodellen vorgestellt, was im Folgenden auf ausgewählte Geschäftsprozessmodelle aus [DrKO17] und [RuQu12] angewendet werden soll. Dabei wird für jede Geschäftsprozessmodellierungssprache aus Kapitel 3 ein Beispiel zur Beschreibung der Ablaufreihenfolge der Knoten des Geschäftsprozessmodells vorgestellt. Die Kontrollflusssemantik dieser Geschäftsprozessmodellierungssprachen wurde in Kapitel 5.3 definiert.

6.3.1 Business Process Model and Notation (BPMN)

Für die Modellierungssprache BPMN wird ein Geschäftsprozessmodell verwendet, das die Vorbereitung eines Weihnachtsdinners modelliert (vgl. Abbildung 85). Für die Beschreibung der Kontrollflusssemantik des Geschäftsprozessmodells muss nach dem Vorgehen aus Abbildung 64 zunächst die definierte Kontrollflusssemantik der verwendeten Geschäftsprozessmodellierungssprache ausgewählt werden. Hierfür wird die Kontrollflusssemantik aus Kapitel 5.3.1 mit dem Semantikschema aus Tabelle 12 und der Semantikzuordnung aus Tabelle 13 verwendet. Nachfolgend können die Knoten, Kanten und Beziehungen des Geschäftsprozessmodells mit dem Modell aus Abbildung 63 beschrieben und die Knoten- und Kantentypen diesen Knoten zugeordnet werden (vgl. Tabelle 33 bis Tabelle 36). Im nächsten Schritt können die impliziten Abhängigkeiten definiert werden. Eine implizite Abhängigkeit existiert zwischen den Knoten t_{39} und t_{40} mit dem Kantentyp impliziter Sequenzfluss, so dass eine neue Instanz der Klasse `G_Kante` erzeugt werden kann, mit der Kante (t_{39}, t_{40}) und mit dem Kantentyp impliziter Sequenzfluss (IS). Abschließend können die möglichen Semantikschemas für die einzelnen Knoten ermittelt und zugewiesen werden (vgl. Tabelle 37) sowie mit den entsprechenden muster- und modellspezifischen Parametern beschrieben werden (vgl. Tabelle 38).

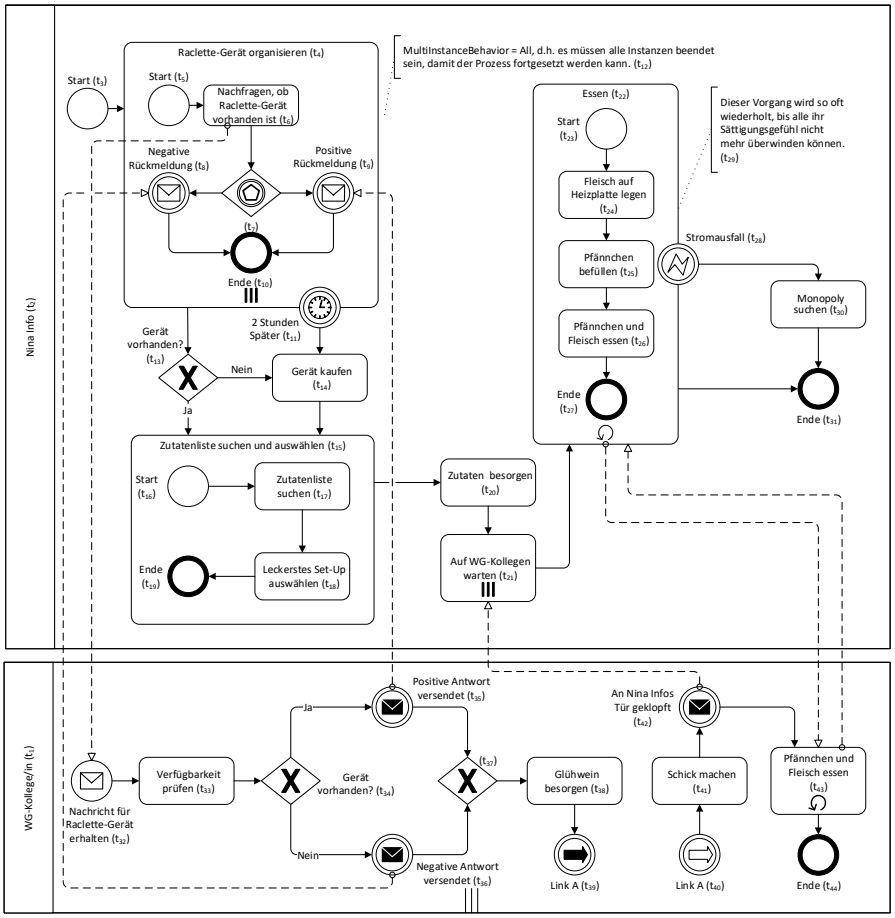


Abbildung 85: Beispiel - BPMN - Weihnachtsdinner [DrKO17]

Klasse	Instanzen
Knoten	t_1, t_2, t_3, t_4, t_{11}, t_{12}, t_{13}, t_{14}, t_{15}, t_{20}, t_{21}, t_{22}, t_{28}, t_{29}, t_{30}, t_{31}, t_{32}, t_{33}, t_{34}, t_{35}, t_{36}, t_{37}, t_{38}, t_{39}, t_{40}, t_{41}, t_{42}, t_{43}, t_{44}
Knotentyp	Aufgabe, Teilprozess, Transaktion, Ereignis-Teilprozess, Aufruf-Aktivität, Exklusives Ereignis, Ereignisbasiertes Gateway, Inklusives Gateway, Paralleles Gateway, Komplexes Gateway, Exklusives Ereignisbasiertes Gateway (Instanziierend), Paralleles Ereignisbasiertes Gateway (Instanziierend), Pool, Blankoereignis, Nachrichtenereignis, Timerereignis, Eskalationsereignis, Bedingungsereignis, Linkereignis, Fehlerereignis, Abbruchereignis, Kompensationsereignis, Signalereignis, Mehrfachereignis, Mehrfach-/Parallelereignis, Terminierungsereignis, Datenobjekt, Dateninputobjekt, Datenoutputobjekt, Datenspeicher, Textannotation
G_Knoten	(t_1, Pool), (t_2, Pool), (t_3, Blankoereignis) (t_4, Teilprozess), (t_{11}, Timerereignis), (t_{12}, Textannotation), (t_{13}, Exklusives Gateway), (t_{14}, Aufgabe), (t_{15}, Teilprozess), (t_{20},

Klasse	Instanzen
	Aufgabe), (t_{21}, Aufgabe), (t_{22}, Teilprozess), (t_{28}, Fehlerereignis), (t_{29}, Textannotation), (t_{30}, Aufgabe), (t_{31}, Blankoereignis), (t_{32}, Nachrichtenereignis), (t_{33}, Aufgabe), (t_{34}, Exklusives Gateway), (t_{35}, Nachrichtenereignis), (t_{36}, Nachrichtenereignis), (t_{37}, Exklusives Gateway), (t_{38}, Aufgabe), (t_{39}, Linkereignis), (t_{40}, Linkereignis), (t_{41}, Aufgabe), (t_{42}, Nachrichtenereignis), (t_{43}, Aufgabe), (t_{44}, Blankoereignis)
Kanten	(t_3, t_4), (t_4, t_{13}), (t_{11}, t_{14}), (t_{13}, t_{14}), (t_{13}, t_{15}), (t_{14}, t_{15}), (t_{15}, t_{20}), (t_{20}, t_{21}), (t_{21}, t_{22}), (t_{28}, t_{30}), (t_{30}, t_{31}), (t_{22}, t_{31}), (t_{32}, t_{33}), (t_{33}, t_{34}), (t_{34}, t_{35}), (t_{34}, t_{36}), (t_{36}, t_{37}), (t_{35}, t_{37}), (t_{37}, t_{38}), (t_{38}, t_{39}), (t_{40}, t_{41}), (t_{41}, t_{42}), (t_{42}, t_{43}), (t_{43}, t_{44}), (t_6, t_{32}), (t_{35}, t_9), (t_{36}, t_8), (t_{42}, t_{21}), (t_{22}, t_{43}), (t_{43}, t_{22})
Kantentyp	Sequenzfluss (S), Nachrichtenfluss (N), Gerichtete Assoziation (GA), Bidirektionale Assoziation (BA), Impliziter Sequenzfluss (IS)
G_Kante	(t_3, t_4, S), (t_4, t_{13}, S), (t_{11}, t_{14}, S), (t_{13}, t_{14}, S), (t_{13}, t_{15}, S), (t_{14}, t_{15}, S), (t_{15}, t_{20}, S), (t_{20}, t_{21}, S), (t_{21}, t_{22}, S), (t_{28}, t_{30}, S), (t_{30}, t_{31}, S), (t_{22}, t_{31}, S), (t_{32}, t_{33}, S), (t_{33}, t_{34}, S), (t_{34}, t_{35}, S), (t_{34}, t_{36}, S), (t_{36}, t_{37}, S), (t_{35}, t_{37}, S), (t_{37}, t_{38}, S), (t_{38}, t_{39}, S), (t_{40}, t_{41}, S), (t_{41}, t_{42}, S), (t_{42}, t_{43}, S), (t_{43}, t_{44}, S), (t_6, t_{32}, N), (t_{35}, t_9, N), (t_{36}, t_8, N), (t_{42}, t_{21}, N), (t_{22}, t_{43}, N), (t_{43}, t_{22}, N), (t_{39}, t_{40}, IS)
Beziehung	(t_4, t_{12}), (t_{22}, t_{29})
G_Beziehung	(t_4, t_{12}, BA), (t_{22}, t_{29}, BA)

Tabelle 33: Beispiel - BPMN - Beschreibung von Abbildung 85 mit dem Modell aus Abbildung 63

Klasse	Instanzen
Knoten	t_5, t_6, t_7, t_8, t_9, t_{10}
Knotentyp	Aufgabe, Teilprozess, Transaktion, Ereignis-Teilprozess, Aufruf-Aktivität, Exklusives Ereignis, Ereignisbasiertes Gateway, Inklusives Gateway, Paralleles Gateway, Komplexes Gateway, Exklusives Ereignisbasiertes Gateway (Instanziierend), Paralleles Ereignisbasiertes Gateway (Instanziierend), Pool, Blankoereignis, Nachrichtenereignis, Timerereignis, Eskalationsereignis, Bedingungsereignis, Linkereignis, Fehlerereignis, Abbruchereignis, Kompensationsereignis, Signalereignis, Mehrfachereignis, Mehrfach-/Parallelereignis, Terminierungsereignis, Datenobjekt, Dateninputobjekt, Datenoutputobjekt, Datenspeicher, Textannotation
G_Knoten	(t_5, Blankoereignis), (t_6, Aufgabe), (t_7, Ereignisbasiertes Gateway), (t_8, Nachrichtenereignis), (t_9, Nachrichtenereignis), (t_{10}, Blankoereignis)
Kanten	(t_5, t_6), (t_6, t_7), (t_7, t_8), (t_7, t_9), (t_8, t_{10}), (t_9, t_{10})
Kantentyp	Sequenzfluss (S), Nachrichtenfluss (N), Gerichtete Assoziation (GA), Bidirektionale Assoziation (BA), Impliziter Sequenzfluss (IS)
G_Kante	(t_5, t_6, S), (t_6, t_7, S), (t_7, t_8, S), (t_7, t_9, S), (t_8, t_{10}, S), (t_9, t_{10}, S)
Beziehung	-
G_Beziehung	-

Tabelle 34: Beispiel - BPMN - Beschreibung des BPMN-Teilprozesses „Raclette-Gerät organisieren (t_4)“ von Abbildung 85 mit dem Modell aus Abbildung 63

Klasse	Instanzen
Knoten	t_{16}, t_{17}, t_{18}, t_{19}
Knotentyp	Aufgabe, Teilprozess, Transaktion, Ereignis-Teilprozess, Aufruf-Aktivität, Exklusives Ereignis, Ereignisbasiertes Gateway, Inklusives Gateway, Paralleles Gateway, Komplexes Gateway, Exklusives Ereignisbasiertes Gateway (Instanziierend), Paralleles

Klasse	Instanzen
	Ereignisbasiertes Gateway (Instanziierend), Pool, Blankoereignis, Nachrichtenereignis, Timerereignis, Eskalationsereignis, Bedingungsereignis, Linkereignis, Fehlerereignis, Abbruchereignis, Kompensationsereignis, Signalereignis, Mehrfachereignis, Mehrfach-/Parallelereignis, Terminierungsereignis, Datenobjekt, Dateninputobjekt, Datenoutputobjekt, Datenspeicher, Textannotation
G_Knoten	(t_{16}, Blankoereignis), (t_{17}, Aufgabe), (t_{18}, Aufgabe), (t_{19}, Blankoereignis)
Kanten	(t_{16}, t_{17}), (t_{17}, t_{18}), (t_{18}, t_{19})
Kantentyp	Sequenzfluss (S), Nachrichtenfluss (N), Gerichtete Assoziation (GA), Bidirektionale Assoziation (BA), Impliziter Sequenzfluss (IS)
G_Kante	(t_{16}, t_{17}, S), (t_{17}, t_{18}, S), (t_{18}, t_{19}, S)
Beziehung	-
G_Beziehung	-

Tabelle 35: Beispiel - BPMN - Beschreibung des BPMN-Teilprozesses „Zutatenliste suchen und auswählen (t_{15})“ von Abbildung 85 mit dem Modell aus Abbildung 63

Klasse	Instanzen
Knoten	t_{23}, t_{24}, t_{25}, t_{26}, t_{27}
Knotentyp	Aufgabe, Teilprozess, Transaktion, Ereignis-Teilprozess, Aufruf-Aktivität, Exklusives Ereignis, Ereignisbasiertes Gateway, Inklusives Gateway, Paralleles Gateway, Komplexes Gateway, Exklusives Ereignisbasiertes Gateway (Instanziierend), Paralleles Ereignisbasiertes Gateway (Instanziierend), Pool, Blankoereignis, Nachrichtenereignis, Timerereignis, Eskalationsereignis, Bedingungsereignis, Linkereignis, Fehlerereignis, Abbruchereignis, Kompensationsereignis, Signalereignis, Mehrfachereignis, Mehrfach-/Parallelereignis, Terminierungsereignis, Datenobjekt, Dateninputobjekt, Datenoutputobjekt, Datenspeicher, Textannotation
G_Knoten	(t_{23}, Blankoereignis), (t_{24}, Aufgabe), (t_{25}, Aufgabe), (t_{26}, Aufgabe), (t_{27}, Blankoereignis)
Kanten	(t_{23}, t_{24}), (t_{24}, t_{25}), (t_{25}, t_{26}), (t_{26}, t_{27})
Kantentyp	Sequenzfluss (S), Nachrichtenfluss (N), Gerichtete Assoziation (GA), Bidirektionale Assoziation (BA), Impliziter Sequenzfluss (IS)
G_Kante	(t_{23}, t_{24}, S), (t_{24}, t_{25}, S), (t_{25}, t_{26}, S), (t_{26}, t_{27}, S)
Beziehung	-
G_Beziehung	-

Tabelle 36: Beispiel - BPMN - Beschreibung des BPMN-Teilprozesses „Essen (t_{22})“ von Abbildung 85 mit dem Modell aus Abbildung 63

Knoten	SemantikschemaID (SID)
t_1, t_2	S53
t_3, t_5, t_{16}, t_{23}	S1
t_4, t_{15}, t_{22}	S55
t_6, t_{14}, t_{17}, t_{18}, t_{20}, t_{21}, t_{24}, t_{25}, t_{26}, t_{30}, t_{33}, t_{38}, t_{41}, t_{43}	S54
t_7	S12
t_8, t_9	S56
t_{10}, t_{19}, t_{27}, t_{31}, t_{44}	S6

Knoten	SemantikschemaID (SID)
t_{11}, t_{28}	S61
t_{12}, t_{29}	-
t_{13}, t_{34}	S12
t_{32}	S41
t_{35}, t_{36}, t_{42}	S17
t_{37}	S25
t_{39}	S15
t_{40}	S49

Tabelle 37: Beispiel - BPMN - Zuweisung der Knoten von Abbildung 85 zum Semantikschema

Knoten	Kontrollflussmuster	Modellspezifische Parameter
t_1	«Während der Ausführung»: wcp_{13}	Nofi(t_1) = (3, 3, 3, static, non-canceling)
t_2	-	-
t_3	«Nach der Ausführung»: wcp_1	ArcCond={((t_3,t_4), true)}
t_4	«Während der Ausführung»: $wcp_{Teilprozess}$	Teilprozess(t_4) = Raclette-Gerät organisieren
	«Während der Ausführung»: wcp_{13}	Nofi(t_4) = (3, 3, 3, static, non-canceling)
	«Während der Ausführung»: $wcp_{BPMNAngeheftetesUnterbrechendesEreignis}$	(t_4, {t_{11}})
	«Nach der Ausführung»: wcp_1	ArcCond = {((t_4,t_{13}), true)}
t_5	«Nach der Ausführung»: wcp_1	ArcCond = {((t_5,t_6), true)}
t_6	«Nach der Ausführung» (Bed. mit Kantentyp Sq): wcp_1	ArcCond = {((t_6,t_7), true)}
	«Nach der Ausführung» (Bed. mit Kantentyp N): wcp_1	ArcCond = {((t_6,t_{32}), true)}
t_7	«Nach der Ausführung»: wcp_{16}	ArcCond = {((t_7,t_8), true), ((t_7,t_9), true)}
t_8	«Nach der Ausführung»: wcp_1	ArcCond = {((t_8,t_{10}), true)}
t_9	«Nach der Ausführung»: wcp_1	ArcCond = {((t_9,t_{10}), true)}
t_{10}	-	-
t_{11}	«Während der Ausführung»: wcp_{23}	Trig = {"nach 2 Stunden"}
	«Nach der Ausführung»: wcp_1	ArcCond={((t_9,t_{10}), true)}
t_{12}	-	-
t_{13}	«Nach der Ausführung»: wcp_4	$<_{XOR}$ = {(t_{13}, {(1, t_{14}), (2, t_{15})})} ArcCond = {((t_{13}, t_{14}), "Nein"), ((t_{13}, t_{15}), "Ja")} Default = {(t_{13}, t_{14})}
t_{14}	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{14},t_{15}), true)}
t_{15}	«Während der Ausführung»: $wcp_{Teilprozess}$	Teilprozess(t_{15}) = Zutatenliste suchen und auswählen
t_{16}	«Nach der Ausführung»: wcp_1	ArcCond={(t_{16},t_{17}), true}
t_{17}	«Nach der Ausführung»: wcp_1	ArcCond={(t_{17},t_{18}), true}
t_{18}	«Nach der Ausführung»: wcp_1	ArcCond={(t_{18},t_{19}), true}
t_{19}	-	-
t_{20}	«Nach der Ausführung»: wcp_1	ArcCond={(t_{20},t_{21}), true}
t_{21}	«Während der Ausführung»: wcp_{13}	Nofi(t_4) = (3, 3, 3, static, non-canceling)

Knoten	Kontrollflussmuster	Modellspezifische Parameter
	«Nach der Ausführung»: wcp_1	ArcCond={(t_{20},t_{27}), true}
t_{22}	«Während der Ausführung»: $wcp_{Teilprozess}$	Teilprozess(t_{22}) = Essen
	«Während der Ausführung»: wcp_{21}	PostTest={"Stättigungsgefühl" = true}
	«Während der Ausführung»: wcp_{23}	Trig={"Stromausfall"}
	«Während der Ausführung»: $wcp_{BPMNAngeheftetesUnterbrechendesEreignis}$	BPMNAngeheftetes-Unterbrechendes-Ereignis(t_{22}) = {t_{28}}
	«Nach der Ausführung» (Bed. mit Kantentyp Sq): wcp_1	ArcCond = {((t_{22},t_{31}), true)}
	«Nach der Ausführung» (Bed. mit Kantentyp N): wcp_1	ArcCond = {((t_{22},t_{43}), true)}
t_{23}	«Nach der Ausführung»: wcp_1	ArcCond={((t_{23},t_{24}), true)}
t_{24}	«Nach der Ausführung»: wcp_1	ArcCond={((t_{24},t_{25}), true)}
t_{25}	«Nach der Ausführung»: wcp_1	ArcCond={((t_{25},t_{26}), true)}
t_{26}	«Nach der Ausführung»: wcp_1	ArcCond={((t_{26},t_{27}), true)}
t_{27}	-	-
t_{28}	«Während der Ausführung»: wcp_{23}	Trig = {"nach 2 Stunden"}
	«Nach der Ausführung»: wcp_1	ArcCond={((t_{28},t_{30}), true)}
t_{29}	-	-
t_{30}	«Nach der Ausführung»: wcp_1	ArcCond={((t_{30},t_{31}), true)}
t_{31}	-	-
t_{32}	«Nach der Ausführung»: wcp_1	ArcCond={((t_{32},t_{33}), true)}
t_{33}	«Nach der Ausführung»: wcp_1	ArcCond={((t_{33},t_{34}), true)}
t_{34}	«Nach der Ausführung»: wcp_4	$<_{XOR}$ = {(t_{34}, {(1, t_{35}), (2, t_{36})})} ArcCond = {((t_{34}, t_{35}),), "Ja"), ((t_{34}, t_{36}), "Nein")} Default = {(t_{34}, t_{36})}
t_{35}	«Nach der Ausführung» (Bed. mit Kantentyp Sq): wcp_1	ArcCond = {((t_{35},t_{37}), true)}
	«Nach der Ausführung» (Bed. mit Kantentyp N): wcp_1	ArcCond = {((t_{35},t_9), true)}
t_{36}	«Nach der Ausführung» (Bed. mit Kantentyp Sq): wcp_1	ArcCond = {((t_{36},t_{37}), true)}
	«Nach der Ausführung» (Bed. mit Kantentyp N): wcp_1	ArcCond = {((t_{36},t_8), true)}
t_{37}	«Nach der Ausführung»: wcp_1	ArcCond={((t_{37},t_{38}), true)}
t_{38}	«Nach der Ausführung»: wcp_1	ArcCond={((t_{38},t_{39}), true)}
t_{39}	«Nach der Ausführung»: wcp_1	ArcCond={((t_{39},t_{40}), true)}
t_{40}	«Nach der Ausführung»: wcp_1	ArcCond={((t_{40},t_{41}), true)}
t_{41}	«Nach der Ausführung»: wcp_1	ArcCond={((t_{41},t_{42}), true)}
t_{42}	«Nach der Ausführung» (Bed. mit Kantentyp Sq): wcp_1	ArcCond = {((t_{42},t_{43}), true)}
	«Nach der Ausführung» (Bed. mit Kantentyp N): wcp_1	ArcCond = {((t_{42},t_{21}), true)}
t_{43}	«Während der Ausführung»: wcp_{21}	PostTest={"Stättigungsgefühl" = true}
	«Nach der Ausführung» (Bed. mit Kantentyp Sq): wcp_1	ArcCond = {((t_{43},t_{44}), true)}
	«Nach der Ausführung» (Bed. mit Kantentyp N): wcp_1	ArcCond = {((t_{43},t_{22}), true)}
t_{44}	-	-

Tabelle 38: Beispiel - BPMN - Modellspezifische Parameter für die Knoten aus Abbildung 85 festlegen

6.3.2 Ereignisgesteuerte Prozesskette (EPK)

Zur Modellierungssprache EPK wird als Beispiel ein Geschäftsprozessmodell verwendet, das die Aufstellung einer Würstchenbude modelliert (vgl. Abbildung 86). Hierfür muss im ersten Schritt nach dem Vorgehen aus Abbildung 64 die definierte Kontrollflusssemantik der verwendeten Geschäftsprozessmodellierungssprache ausgewählt werden, wobei die Kontrollflusssemantik aus Kapitel 5.3.2 mit dem Semantikschema aus Tabelle 15 und der Semantikzuordnung aus Tabelle 16 verwendet wird. Nachfolgend können die Knoten, Kanten und Beziehungen des Geschäftsprozessmodells mit dem Modell aus Abbildung 63 beschrieben sowie die Knoten- und Kantentypen diesen Knoten zugeordnet werden (vgl. Tabelle 39). Für die EPK existieren keine impliziten Kantentypen, so dass die möglichen Semantikschemas für die einzelnen Knoten unmittelbar ermittelt und zugewiesen (vgl. Tabelle 40) sowie mit den entsprechenden muster- und modellspezifischen Parametern spezifiziert werden können (vgl. Tabelle 41).

Klasse	Instanzen
Knoten	t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, t_9, t_{10}, t_{11}, t_{12}, t_{13}, t_{14}, t_{15}, t_{16}, t_{17}, t_{18}, t_{19}, t_{20}, t_{21}, t_{22}, t_{23}, t_{24}, t_{25}, t_{26}, t_{27}, t_{28}, t_{29}, t_{30}, t_{31}, t_{32}, t_{33}, t_{34}, t_{35}
Knotentyp	Ereignis, Funktion, XOR-Verknüpfungsoperator, AND-Verknüpfungsoperator, OR-Verknüpfungsoperator, Prozesswegweiser
G_Knoten	(t_1, Ereignis), (t_2, XOR-Verknüpfungsoperator), (t_3, Funktion) (t_4, XOR-Verknüpfungsoperator), (t_5, Ereignis), (t_6, AND-Verknüpfungsoperator), (t_7, Funktion), (t_8, Funktion), (t_9, XOR-Verknüpfungsoperator), (t_{10}, Ereignis), (t_{11}, Timerereignis), (t_{12}, Funktion), (t_{13}, Funktion), (t_{14}, OR-Verknüpfungsoperator), (t_{15}, Ereignis), (t_{16}, Ereignis), (t_{17}, OR-Verknüpfungsoperator), (t_{18}, Funktion), (t_{19}, Ereignis), (t_{20}, XOR-Verknüpfungsoperator), (t_{21}, Ereignis), (t_{22}, Funktion), (t_{23}, AND-Verknüpfungsoperator), (t_{24}, Ereignis), (t_{25}, Funktion), (t_{26}, Ereignis), (t_{27}, Ereignis), (t_{28}, Funktion), (t_{29}, XOR-Verknüpfungsoperator), (t_{30}, Ereignis), (t_{31}, Funktion), (t_{32}, Ereignis), (t_{33}, Ereignis), (t_{34}, Funktion), (t_{35}, Ereignis)
Kanten	(t_1, t_2), (t_2, t_3), (t_3, t_4), (t_4, t_5), (t_5, t_6), (t_6, t_7), (t_7, t_{23}), (t_6, t_8), (t_8, t_9), (t_9, t_{10}), (t_{10}, t_{12}), (t_{12}, t_{21}), (t_{21}, t_{20}), (t_9, t_{11}), (t_{11}, t_{13}), (t_{13}, t_{14}), (t_{14}, t_{15}), (t_{14}, t_{16}), (t_{15}, t_{17}), (t_{16}, t_{17}), (t_{17}, t_{18}), (t_{18}, t_{19}), (t_{19}, t_{20}), (t_{20}, t_{22}), (t_{22}, t_{23}), (t_{23}, t_{24}), (t_{24}, t_{25}), (t_{25}, t_{26}), (t_4, t_{27}), (t_{27}, t_{28}), (t_{28}, t_{29}), (t_{29}, t_{30}), (t_{30}, t_{31}), (t_{31}, t_{32}), (t_{29}, t_{33}), (t_{33}, t_{34}), (t_{34}, t_{35}), (t_{35}, t_2)
Kantentyp	Kontrollfluss (K)
G_Kante	(t_1, t_2, K), (t_2, t_3, K), (t_3, t_4, K), (t_4, t_5, K), (t_5, t_6, K), (t_6, t_7, K), (t_7, t_{23}, K), (t_6, t_8, K), (t_8, t_9, K), (t_9, t_{10}, K), (t_{10}, t_{12}, K), (t_{12}, t_{21}, K), (t_{21}, t_{20}, K), (t_9, t_{11}, K), (t_{11}, t_{13}, K), (t_{13}, t_{14}, K), (t_{14}, t_{15}, K), (t_{14}, t_{16}, K), (t_{15}, t_{17}, K), (t_{16}, t_{17}, K), (t_{17}, t_{18}, K), (t_{18}, t_{19}, K), (t_{19}, t_{20}, K), (t_{20}, t_{22}, K), (t_{22}, t_{23}, K), (t_{23}, t_{24}, K), (t_{24}, t_{25}, K), (t_{25}, t_{26}, K), (t_4, t_{27}, K), (t_{27}, t_{28}, K), (t_{28}, t_{29}, K), (t_{29}, t_{30}, K), (t_{30}, t_{31}, K), (t_{31}, t_{32}, K), (t_{29}, t_{33}, K), (t_{33}, t_{34}, K), (t_{34}, t_{35}, K), (t_{35}, t_2, K)
Beziehung	-
G_Beziehung	-

Tabelle 39: Beispiel - EPK - Beschreibung von Abbildung 86 mit dem Modell aus Abbildung 63

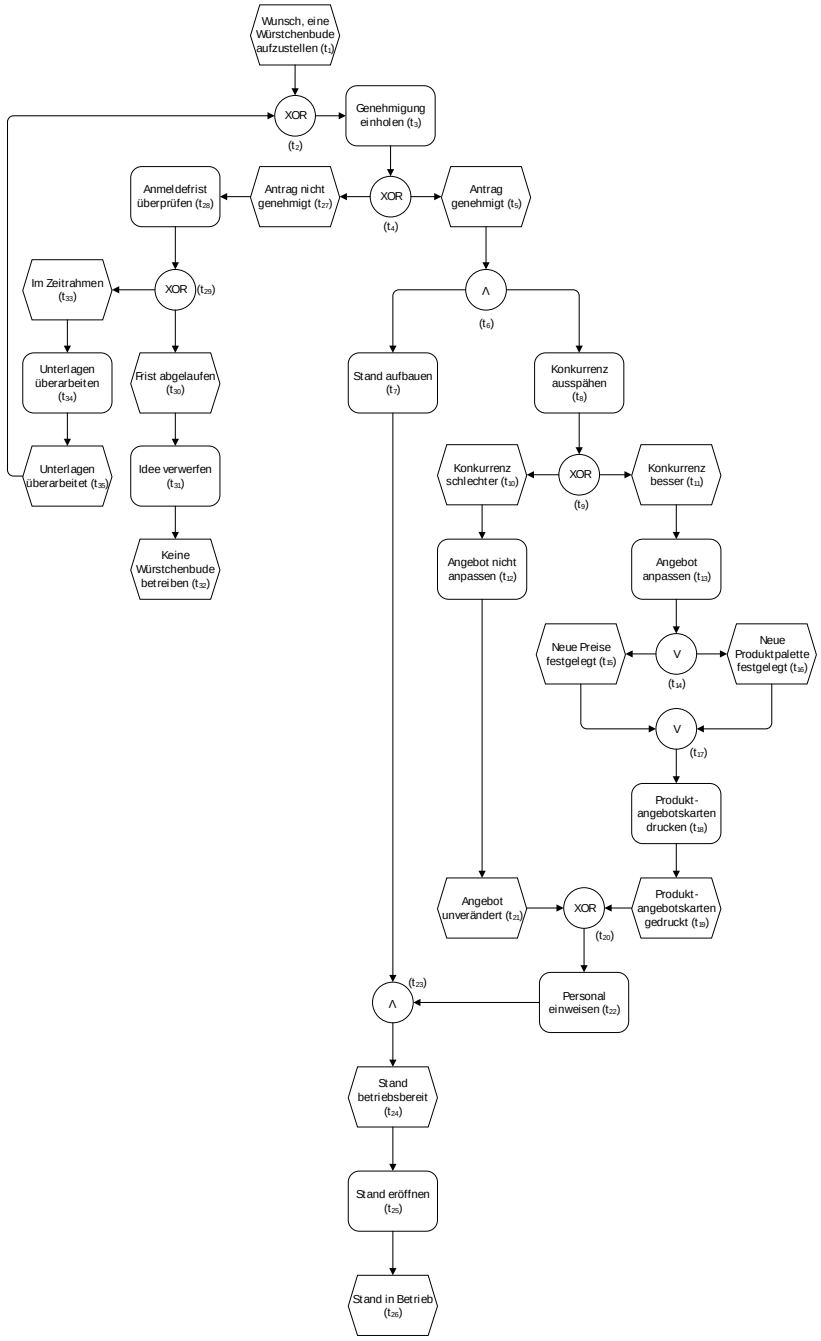


Abbildung 86: Beispiel - EPK - Aufstellung einer Würstchenbude [DrKO17]

Knoten	SemantikschemaID (SID)
t_1, t_3, t_5, t_7, t_8, t_{10}, t_{11}, t_{12}, t_{13}, t_{15}, t_{16}, t_{18}, t_{19}, t_{21}, t_{22}, t_{24}, t_{25}, t_{26}, t_{27}, t_{28}, t_{30}, t_{31}, t_{32}, t_{33}, t_{34}, t_{35}	S6
t_2, t_{20}	S14
t_4, t_9, t_{29}	S8
t_6	S7
t_{14}	S9
t_{17}	S16
t_{23}	S12

Tabelle 40: Beispiel - EPK - Zuweisung der Knoten von Abbildung 86 zum Semantikschema

Knoten	Kontrollflussmuster	Modellspezifische Parameter
t_1	«Nach der Ausführung»: wcp_1	ArcCond = {((t_1,t_2), true)}
t_2	«Nach der Ausführung»: wcp_1	ArcCond = {((t_2,t_3), true)}
t_3	«Nach der Ausführung»: wcp_1	ArcCond = {((t_3,t_4), true)}
t_4	«Nach der Ausführung»: wcp_4	<$_{XOR}$ = {(t_4, {(1, t_{27}), (2, t_5)})} ArcCond = {((t_4, t_{27}), "Antrag nicht genehmigt"), ((t_4, t_5), "Antrag genehmigt")}
t_5	«Nach der Ausführung»: wcp_1	ArcCond = {((t_5,t_6), true)}
t_6	«Nach der Ausführung»: wcp_2	ArcCond = {((t_6,t_7), true), ((t_6,t_8), true)}
t_7	«Nach der Ausführung»: wcp_1	ArcCond = {((t_7,t_{23}), true)}
t_8	«Nach der Ausführung»: wcp_1	ArcCond = {((t_8,t_9), true)}
t_9	«Nach der Ausführung»: wcp_4	<$_{XOR}$ = {(t_9, {(1, t_{10}), (2, t_{11})})} ArcCond = {((t_9, t_{10}), "Konkurrenz schlechter"), ((t_9, t_{11}), "Konkurrenz besser")} Default = {(t_9, t_{11})}
t_{10}	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{10},t_{12}), true)}
t_{11}	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{11},t_{13}), true)}
t_{12}	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{12},t_{21}), true)}
t_{13}	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{13},t_{14}), true)}
t_{14}	«Nach der Ausführung»: wcp_6	ArcCond = {((t_{14}, t_{15}), "Neue Preise festgelegt"), ((t_{14}, t_{16}), "neue Produktpalette festgelegt")} Default = {(t_{14}, t_{15})}
t_{15}	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{15},t_{17}), true)}
t_{16}	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{16},t_{17}), true)}
t_{17}	«Vor der Ausführung»: wcp_1	Struc(t_{17}) = t_{14}
	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{17},t_{18}), true)}
t_{18}	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{18},t_{19}), true)}
t_{19}	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{19},t_{20}), true)}
t_{20}	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{20},t_{22}), true)}
t_{21}	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{21},t_{20}), true)}
t_{22}	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{22},t_{23}), true)}
t_{23}	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{23},t_{24}), true)}
t_{24}	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{24},t_{25}), true)}
t_{25}	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{25},t_{26}), true)}

Knoten	Kontrollflussmuster	Modellspezifische Parameter
t_{26}	-	-
t_{27}	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{27},t_{28}), true)}
t_{28}	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{28},t_{29}), true)}
t_{29}	«Nach der Ausführung»: wcp_1	$<_{XOR}$ = {(t_{29}, {(1, t_{30}), (2, t_{33})})} ArcCond = {((t_{29}, t_{30}), "Frist abgelaufen"), ((t_{29}, t_{33}), "Im Zeitrahmen")} Default = {(t_{29}, t_{33})}
t_{30}	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{30},t_{31}), true)}
t_{31}	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{31},t_{32}), true)}
t_{32}	-	-
t_{33}	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{33},t_{34}), true)}
t_{34}	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{34},t_{35}), true)}
t_{35}	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{35},t_2), true)}

Tabelle 41: Beispiel - EPK - Modellspezifische Parameter für die Knoten aus Abbildung 86 festlegen

6.3.3 UML-Aktivitätsdiagramm (UML-AD)

Als Beispiel für ein UML-Aktivitätsdiagramm wird ein Geschäftsprozessmodell verwendet, das die Bedienung eines Zigarettenautomaten modelliert (vgl. Abbildung 87), wobei im ersten Schritt, entsprechend dem Vorgehen in Abbildung 64, die definierte Kontrollflusssemantik der Geschäftsprozessmodellierungssprache ausgewählt werden muss. Hierfür wird die Kontrollflusssemantik aus Kapitel 5.3.3 mit dem Semantikschema in Tabelle 20 und der Semantikzuordnung in Tabelle 21 verwendet. Nachfolgend können die Knoten, Kanten und Beziehungen des Geschäftsprozessmodells mit dem Modell aus Abbildung 63 beschrieben sowie die Knoten- und Kantentypen diesen Knoten zugeordnet werden (vgl. Tabelle 42 und Tabelle 43). In dem Geschäftsprozessmodell gibt es keine impliziten Abhängigkeiten, so dass die möglichen Semantikschemas für die einzelnen Knoten unmittelbar ermittelt und zugewiesen (vgl. Tabelle 44) sowie mit den entsprechenden muster- und modellspezifischen Parametern spezifiziert werden können (vgl. Tabelle 45).

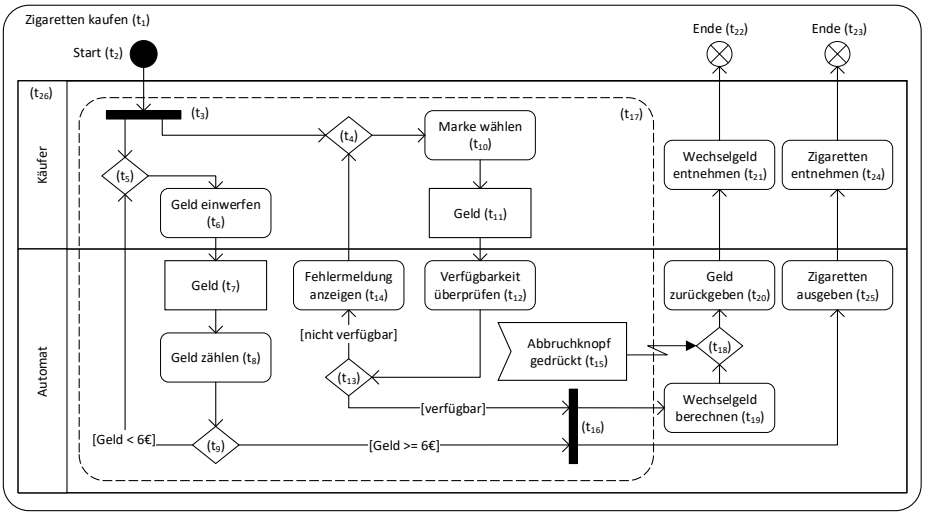


Abbildung 87: Beispiel - UML-AD - Bedienung eines Zigarettenautomaten [RuQu12]

Klasse	Instanzen
Knoten	t_1
Knotentyp	Aktion, Senden von Signalen, Empfangen von Signalen, Zeitereignis, Entscheidungs-/Zusammenführungsknoten, Aufspaltungs-/Synchronisationsknoten, Startknoten, Aktivitätsendknoten, Ablaufende, Objektknoten, Konnektorknoten, Aktivitätsgruppe
G_Knoten	(t_1, Aktivität)
Kanten	-
Kantentyp	Sequenzfluss (Sq), impliziter Sequenzfluss (IS)
G_Kante	-
Beziehung	-
G_Beziehung	-

Tabelle 42: Beispiel - UML-AD - Beschreibung von Abbildung 87 mit dem Modell aus Abbildung 63

Klasse	Instanzen
Knoten	t_2, t_3, t_4, t_5, t_6, t_7, t_8, t_9, t_{10}, t_{11}, t_{12}, t_{13}, t_{14}, t_{15}, t_{16}, t_{17}, t_{18}, t_{19}, t_{20}, t_{21}, t_{22}, t_{23}, t_{24}, t_{25}, t_{26}, t_{27}, t_{28}, t_{29}
Knotentyp	Aktion, Senden von Signalen, Empfangen von Signalen, Zeitereignis, Entscheidungs-/Zusammenführungsknoten, Aufspaltungs-/Synchronisationsknoten, Startknoten, Aktivitätsendknoten, Ablaufende, Objektknoten, Konnektorknoten, Aktivitätsgruppe
G_Knoten	(t_2, Startknoten), (t_3, Aufspaltungs-/Synchronisationsknoten) (t_4, Entscheidungs-/Zusammenführungsknoten), (t_5, Entscheidungs-/Zusammenführungsknoten), (t_6, Aktion), (t_7, Objektknoten), (t_8, Aktion), (t_9, Entscheidungs-/Zusammenführungs-knoten), (t_{10}, Aktion), (t_{11}, Objektknoten), (t_{12}, Aktion), (t_{13}, Entscheidungs-/Zusammenführungsknoten), (t_{14}, Aktion), (t_{15}, Empfangen von Signalen), (t_{16},

Klasse	Instanzen
	Aufspaltungs-/Synchronisationsknoten), (t_{17}, OR-Verknüpfungsoperator), (t_{18}, Unterbrechungsbereich), (t_{19}, Aktion), (t_{20}, Entscheidungs-/Zusammenführungs-knoten), (t_{21}, Aktion), (t_{22}, Konnektorknoten), (t_{23}, Konnektorknoten), (t_{24}, Aktion), (t_{25}, Ablaufende), (t_{26}, Aktion), (t_{27}, Aktion), (t_{28}, Ablaufende), (t_{29}, Aktivitätsgruppe)
Kanten	(t_2, t_3), (t_3, t_4), (t_4, t_{10}), (t_{10}, t_{11}), (t_{11}, t_{12}), (t_{12}, t_{13}), (t_{13}, t_{14}), (t_{14}, t_4), (t_{13}, t_{16}), (t_3, t_5), (t_5, t_6), (t_6, t_7), (t_7, t_8), (t_8, t_9), (t_9, t_5), (t_9, t_{16}), (t_{16}, t_{19}), (t_{16}, t_{25}), (t_{25}, t_{24}), (t_{24}, t_{23}), (t_{19}, t_{18}), (t_{15}, t_{18}), (t_{18}, t_{20}), (t_{20}, t_{21}), (t_{21}, t_{22})
Kantentyp	Sequenzfluss (Sq), impliziter Sequenzfluss (IS)
G_Kante	(t_2, t_3, Sq), (t_3, t_4, Sq), (t_4, t_{10}, Sq), (t_{10}, t_{11}, Sq), (t_{11}, t_{12}, Sq), (t_{12}, t_{13}, Sq), (t_{13}, t_{14}, Sq), (t_{14}, t_4, Sq), (t_{13}, t_{16}, Sq), (t_3, t_5, Sq), (t_5, t_6, Sq), (t_6, t_7, Sq), (t_7, t_8, Sq), (t_8, t_9, Sq), (t_9, t_5, Sq), (t_9, t_{16}, Sq), (t_{16}, t_{19}, Sq), (t_{16}, t_{25}, Sq), (t_{25}, t_{24}, Sq), (t_{24}, t_{23}, Sq), (t_{19}, t_{18}, Sq), (t_{15}, t_{18}, Sq), (t_{18}, t_{20}, Sq), (t_{20}, t_{21}, Sq), (t_{21}, t_{22}, Sq)
Beziehung	-
G_Beziehung	-

Tabelle 43: Beispiel - UML-AD - Beschreibung des UML-AD-Teilprozesses „Zigaretten kaufen (t_1)“ von Abbildung 87 mit dem Modell aus Abbildung 63

Knoten	SemantikschemaID (SID)
t_1	S5
t_2	S1
t_3	S9
t_4, t_5, t_{18}	S44
t_{22}, t_{23}	S6
t_6, t_7, t_8, t_{10}, t_{11}, t_{12}, t_{14}, t_{19}, t_{20}, t_{21}, t_{24}, t_{25}	S8
t_9, t_{13}	S12
t_{15}	S72
t_{16}	S27
t_{17}, t_{26}	-

Tabelle 44: Beispiel - UML-AD - Zuweisung der Knoten von Abbildung 87 zum Semantikschema

Knoten	Kontrollflussmuster	Modellspezifische Parameter
t_1	«Während der Ausführung»: $wcp_{Teilprozess}$	Teilprozess(t_4) = Zigaretten kaufen
t_2	«Nach der Ausführung»: wcp_1	ArcCond = {(t_2,t_3), true)}
t_3	«Nach der Ausführung»: wcp_1	ArcCond={(t_3,t_4), true), (t_3,t_5), true)}
t_4	«Nach der Ausführung»: wcp_1	ArcCond = {(t_4,t_{10}), true)}
t_5	«Nach der Ausführung»: wcp_1	ArcCond = {(t_5,t_6), true)}
t_6	«Nach der Ausführung»: wcp_1	ArcCond = {(t_6,t_7), true)}
t_7	«Nach der Ausführung»: wcp_1	ArcCond = {(t_7,t_8), true)}
t_8	«Nach der Ausführung»: wcp_1	ArcCond = {(t_8,t_9), true)}
t_9	«Nach der Ausführung»: wcp_4	$<_{XOR}$ = {(t_9, {(1, t_5), (2, t_{16})})} ArcCond = {(t_9, t_5), "Geld < 6€"), (t_9, t_{16}), "Geld >= 6€")} Default = {(t_9, t_5)}

Knoten	Kontrollflussmuster	Modellspezifische Parameter
t_{10}	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{10},t_{11}), true)}
t_{11}	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{11},t_{12}), true)}
t_{12}	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{12},t_{13}), true)}
t_{13}	«Nach der Ausführung»: wcp_1	$<_{XOR}$ = {(t_{13}, {(1, t_{14}), (2, t_{16})})} ArcCond = {((t_{13}, t_{14}), "nicht verfügbar"), ((t_{13}, t_{16}), "verfügbar")} Default = {(t_{13}, t_{14})}
t_{14}	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{14},t_4), true)}
t_{15}	«Während der Ausführung»: wcp_{23}	Trig(t_{15}) = {"Abbruchknopf gedrückt"}
	«Während der Ausführung»: wcp_{25}	Rem(t_{15}) = {t_3, t_4, t_5, t_6, t_7, t_8, t_9, t_{10}, t_{11}, t_{12}, t_{13}, t_{14}, t_{16}, t_{17}}
	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{15},t_{18}), true)}
t_{16}	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{16},t_{19}), true), ((t_{16},t_{25}), true)}
t_{17}	-	-
t_{18}	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{18},t_{20}), true)}
t_{19}	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{19},t_{18}), true)}
t_{20}	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{20},t_{21}), true)}
t_{21}	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{21},t_{22}), true)}
t_{22}	-	-
t_{23}	-	-
t_{24}	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{24},t_{23}), true)}
t_{25}	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{25},t_{24}), true)}
t_{26}	-	-

Tabelle 45: Beispiel - UML-AD - Modellspezifische Parameter für die Knoten aus Abbildung 87 festlegen

6.3.4 Petri-Netz (PN)

Für Petri-Netze wird exemplarisch ein Geschäftsprozessmodell verwendet, das den Verkaufs- und Beratungsprozess eines Lampengeschäfts modelliert (vgl. Abbildung 88), wobei im ersten Schritt die definierte Kontrollflusssemantik der verwendeten Geschäftsprozessmodellierungssprache ausgewählt werden muss (vgl. Abbildung 64). Hierfür wird die Kontrollflusssemantik aus Kapitel 5.3.4 mit dem Semantikschema in Tabelle 23 und der Semantikzuordnung in Tabelle 24 verwendet. Nachfolgend können die Knoten, Kanten und Beziehungen des Geschäftsprozessmodells mit dem Modell aus Abbildung 63 beschrieben sowie die Knoten- und Kantentypen diesen Knoten zugeordnet werden (vgl. Tabelle 46). Es existieren keine impliziten Abhängigkeiten im Geschäftsprozessmodell, so dass die möglichen Semantikschemas für die einzelnen Knoten unmittelbar ermittelt und zugewiesen (vgl. Tabelle 47) sowie mit den entsprechenden muster- und modellspezifischen Parametern spezifiziert werden können (vgl. Tabelle 48).

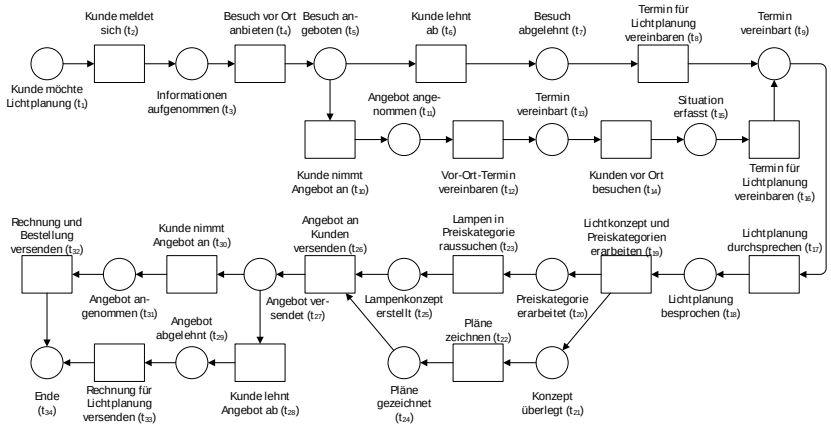


Abbildung 88: Beispiel - Petri-Netz - Verkaufs- und Beratungsprozess eines Lampengeschäfts [DrKO17]

Klasse	Instanzen
Knoten	t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, t_9, t_{10}, t_{11}, t_{12}, t_{13}, t_{14}, t_{15}, t_{16}, t_{17}, t_{18}, t_{19}, t_{20}, t_{21}, t_{22}, t_{23}, t_{24}, t_{25}, t_{26}, t_{27}, t_{28}, t_{29}, t_{30}, t_{31}, t_{32}, t_{33}, t_{34}
Knotentyp	Stelle, Transition
G_Knoten	(t_1, Stelle), (t_2, Transition), (t_3, Stelle) (t_4, Transition), (t_5, Stelle), (t_6, Transition), (t_7, Stelle), (t_8, Transition), (t_9, Stelle), (t_{10}, Transition), (t_{11}, Stelle), (t_{12}, Transition), (t_{13}, Stelle), (t_{14}, Transition), (t_{15}, Stelle), (t_{16}, Transition), (t_{17}, Stelle), (t_{18}, Stelle), (t_{19}, Transition), (t_{20}, Stelle), (t_{21}, Stelle), (t_{22}, Transition), (t_{23}, Transition), (t_{24}, Stelle), (t_{25}, Stelle), (t_{26}, Transition), (t_{27}, Stelle), (t_{28}, Transition), (t_{29}, Stelle), (t_{30}, Transition), (t_{31}, Stelle), (t_{32}, Transition), (t_{33}, Transition), (t_{34}, Stelle)
Kanten	(t_1, t_2), (t_2, t_3), (t_3, t_4), (t_4, t_5), (t_5, t_6), (t_6, t_7), (t_7, t_8), (t_8, t_9), (t_5, t_{10}), (t_{10}, t_{11}), (t_{11}, t_{12}), (t_{12}, t_{13}), (t_{13}, t_{14}), (t_{14}, t_{15}), (t_{15}, t_{16}), (t_{16}, t_9), (t_9, t_{17}), (t_{17}, t_{18}), (t_{18}, t_{19}), (t_{19}, t_{20}), (t_{20}, t_{23}), (t_{23}, t_{25}), (t_{25}, t_{26}), (t_{19}, t_{21}), (t_{21}, t_{22}), (t_{22}, t_{24}), (t_{24}, t_{26}), (t_{26}, t_{27}), (t_{27}, t_{28}), (t_{28}, t_{29}), (t_{29}, t_{33}), (t_{33}, t_{34}), (t_{27}, t_{30}), (t_{30}, t_{31}), (t_{31}, t_{32}), (t_{32}, t_{34})
Kantentyp	Sequenzfluss (Sq)
G_Kante	(t_1, t_2, Sq), (t_2, t_3, Sq), (t_3, t_4, Sq), (t_4, t_5, Sq), (t_5, t_6, Sq), (t_6, t_7, Sq), (t_7, t_8, Sq), (t_8, t_9, Sq), (t_5, t_{10}, Sq), (t_{10}, t_{11}, Sq), (t_{11}, t_{12}, Sq), (t_{12}, t_{13}, Sq), (t_{13}, t_{14}, Sq), (t_{14}, t_{15}, Sq), (t_{15}, t_{16}, Sq), (t_{16}, t_9, Sq), (t_9, t_{17}, Sq), (t_{17}, t_{18}, Sq), (t_{18}, t_{19}, Sq), (t_{19}, t_{20}, Sq), (t_{20}, t_{23}, Sq), (t_{23}, t_{25}, Sq), (t_{25}, t_{26}, Sq), (t_{19}, t_{21}, Sq), (t_{21}, t_{22}, Sq), (t_{22}, t_{24}, Sq), (t_{24}, t_{26}, Sq), (t_{26}, t_{27}, Sq), (t_{27}, t_{28}, Sq), (t_{28}, t_{29}, Sq), (t_{29}, t_{33}, Sq), (t_{33}, t_{34}, Sq), (t_{27}, t_{30}, Sq), (t_{30}, t_{31}, Sq), (t_{31}, t_{32}, Sq), (t_{32}, t_{34}, Sq)
Beziehung	-
G_Beziehung	-

Tabelle 46: Beispiel - Petri-Netz - Beschreibung von Abbildung 88 mit dem Modell aus Abbildung 63

Knoten	SemantikschemaID (SID)
t_1	S2
t_{34}	S5
t_5, t_{27}	S8

Knoten	SemantikschemaID (SID)
t_9	S13
t_{19}	S7
t_{26}	S10
t_2, t_3, t_4, t_6, t_7, t_8, t_{10}, t_{11}, t_{12}, t_{13}, t_{14}, t_{15}, t_{16}, t_{17}, t_{18}, t_{20}, t_{21}, t_{22}, t_{23}, t_{24}, t_{25}, t_{28}, t_{29}, t_{30}, t_{31}, t_{32}, t_{33}	S6

Tabelle 47: Beispiel - Petri-Netz - Zuweisung der Knoten von Abbildung 88 zum Semantikschema

Knoten	Kontrollflussmuster	Parameter
t_1	«Nach der Ausführung»: wcp_1	ArcCond = {(t_1,t_2), true)}
t_2	«Nach der Ausführung»: wcp_1	ArcCond = {(t_2,t_3), true)}
t_3	«Nach der Ausführung»: wcp_1	ArcCond = {(t_3,t_4), true)}
t_4	«Nach der Ausführung»: wcp_1	ArcCond = {(t_4,t_5), true)}
t_5	«Nach der Ausführung»: wcp_4	$<_{XOR}$ = {(t_5, {(1, t_6), (2, t_{10})})} ArcCond = {(t_5, t_6), "Kunde lehnt angebot ab"), (t_5, t_{10}), "Kunde nimmt angebot an")} Default = {(t_5, t_6)}
t_6	«Nach der Ausführung»: wcp_1	ArcCond = {(t_6,t_7), true)}
t_7	«Nach der Ausführung»: wcp_1	ArcCond = {(t_7,t_8), true)}
t_8	«Nach der Ausführung»: wcp_1	ArcCond = {(t_8,t_9), true)}
t_9	«Nach der Ausführung»: wcp_1	ArcCond = {(t_9,t_{17}), true)}
t_{10}	«Nach der Ausführung»: wcp_1	ArcCond = {(t_{10},t_{11}), true)}
t_{11}	«Nach der Ausführung»: wcp_1	ArcCond = {(t_{11},t_{12}), true)}
t_{12}	«Nach der Ausführung»: wcp_1	ArcCond = {(t_{12},t_{13}), true)}
t_{13}	«Nach der Ausführung»: wcp_1	ArcCond = {(t_{13},t_{14}), true)}
t_{14}	«Nach der Ausführung»: wcp_1	ArcCond = {(t_{14},t_{15}), true)}
t_{15}	«Nach der Ausführung»: wcp_1	ArcCond = {(t_{15},t_{16}), true)}
t_{16}	«Nach der Ausführung»: wcp_1	ArcCond = {(t_{16},t_9), true)}
t_{17}	«Nach der Ausführung»: wcp_1	ArcCond = {(t_{17},t_{18}), true)}
t_{18}	«Nach der Ausführung»: wcp_1	ArcCond = {(t_{18},t_{19}), true)}
t_{19}	«Nach der Ausführung»: wcp_2	ArcCond = {(t_{19},t_{20}), true), (t_{19},t_{21}), true)}}
t_{20}	«Nach der Ausführung»: wcp_1	ArcCond = {(t_{20},t_{23}), true)}
t_{21}	«Nach der Ausführung»: wcp_1	ArcCond = {(t_{21},t_{22}), true)}
t_{22}	«Nach der Ausführung»: wcp_1	ArcCond = {(t_{22},t_{24}), true)}
t_{23}	«Nach der Ausführung»: wcp_1	ArcCond = {(t_{23},t_{25}), true)}
t_{24}	«Nach der Ausführung»: wcp_1	ArcCond = {(t_{24},t_{26}), true)}
t_{25}	«Nach der Ausführung»: wcp_1	ArcCond = {(t_{25},t_{26}), true)}
t_{26}	«Nach der Ausführung»: wcp_1	ArcCond = {(t_{26},t_{27}), true)}
t_{27}	«Nach der Ausführung»: wcp_4	$<_{XOR}$ = {(t_{27}, {(1, t_{28}), (2, t_{30})})} ArcCond = {(t_{27}, t_{28}), "Kunde lehnt Angebot ab"), (t_{27}, t_{30}), "Kunde nimmt Angebot an")} Default = {(t_{27}, t_{28})}
t_{28}	«Nach der Ausführung»: wcp_1	ArcCond = {(t_{28},t_{29}), true)}
t_{29}	«Nach der Ausführung»: wcp_1	ArcCond = {(t_{29},t_{33}), true)}
t_{30}	«Nach der Ausführung»: wcp_1	ArcCond = {(t_{33},t_{34}), true)}

Knoten	Kontrollflussmuster	Parameter
t_{31}	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{31},t_{32}), true)}
t_{32}	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{32},t_{34}), true)}
t_{33}	«Nach der Ausführung»: wcp_1	ArcCond = {((t_{33},t_{34}), true)}
t_{34}	-	-

Tabelle 48: Beispiel - Petri-Netz - Modellspezifische Parameter für die Knoten aus Abbildung 88 festlegen

6.3.5 New Yet Another Workflow Language (newYAWL)

Für newYAWL wird ein Geschäftsprozessmodell verwendet, das einen Inspektionsprozess beschreibt (vgl. Abbildung 65), wobei im ersten Schritt die definierte Kontrollflusssemantik der verwendeten Geschäftsprozessmodellierungssprache ausgewählt werden muss (vgl. Abbildung 64). Hierfür wird die Kontrollflusssemantik aus Kapitel 5.3.5 mit dem Semantikschema in Tabelle 27 und der Semantikzuordnung in Tabelle 28 verwendet. Nachfolgend können die Knoten, Kanten und Beziehungen des Geschäftsprozessmodells mit dem Modell aus Abbildung 63 beschrieben sowie die Knoten- und Kantentypen diesen Knoten zugeordnet werden (vgl. Tabelle 29). Es existieren keine impliziten Abhängigkeiten, so dass abschließend die möglichen Semantikschemas für die einzelnen Knoten ermittelt und zugewiesen (vgl. Tabelle 49) sowie mit den entsprechenden Parametern spezifiziert werden können (vgl. Tabelle 50).

Knoten	Semantikschema (SID)
i	S1
t_1, t_2, t_3, t_4 t_6, t_7, t_8	S6
t_5	S7
t_9	-
o	S5

Tabelle 49: Beispiel - newYAWL - Zuweisung der Knoten von Abbildung 65 zum Semantikschema

Knoten	Kontrollflussmuster	Parameter
i	«Nach der Ausführung»: wcp_1	ArcCond={((i,t_1), true)}
t_1	« Nach der Ausführung»: wcp_2	ArcCond={((t_1,t_2), true), ((t_1,t_3), true), ((t_1,t_4), true)}
t_2	« Nach der Ausführung»: wcp_4	$<_{XOR}$ = {(t_2, {(1, t_5), (2, t_6)})} ArcCond = {((t_2, t_5), "Defekt"), ((t_2, t_6), "Kein Defekt")} Default = {(t_2, t_5)}
t_3	« Nach der Ausführung»: wcp_4	$<_{XOR}$ = {(t_3, {(1, t_5), (2, t_6)})} ArcCond = {((t_3, t_5), "Defekt"), ((t_3, t_6), "Kein Defekt")} Default = {(t_3, t_5)}

Knoten	Kontrollflussmuster	Parameter
t_4	« Nach der Ausführung»: wcp_4	$<_{XOR}$ = {(t_4, {(1, t_5), (2, t_6)})} ArcCond = {((t_4, t_5), "Defekt"), ((t_4, t_6), "Kein Defekt")} Default = {(t_4, t_5)}
t_5	«Vor der Ausführung»: wcp_{29}	Rem = {t_2, t_3, t_4}
	«Nach der Ausführung»: wcp_1	ArcCond = {((t_5,t_7), true)}
t_6	«Nach der Ausführung»: wcp_1	ArcCond = {((t_6,t_8), true)}
t_7	«Nach der Ausführung»: wcp_1	ArcCond = {((t_7,t_8), true)}
t_8	«Nach der Ausführung»: wcp_1	ArcCond = {((t_8,o), true)}
t_9	-	-
o	-	-

Tabelle 50: Beispiel - newYAWL - Modellspezifische Parameter für die Knoten aus Abbildung 65 festlegen

6.4 Zusammenfassung

Es wurde ein Vorgehen zur Beschreibung der Ablaufreihenfolge der Elemente eines Geschäftsprozessmodells vorgestellt, für das zunächst drei Anforderungen definiert wurden. Daran anknüpfend wurde das Vorgehen exemplarisch auf ausgewählte Geschäftsprozessmodelle aus [DrKO17] und [RuQu12] angewendet.

Die erste Anforderung beschreibt, dass das Geschäftsprozessmodell mit einer Geschäftsprozessmodellierungssprache erstellt worden sein muss, dessen Kontrollflusssemantik vollständig mit der Methode aus Kapitel 5 beschrieben wurde. Die zweite Anforderung definiert, dass das erstellte Geschäftsprozessmodell durch Knoten, Kanten und Beziehungen beschrieben werden kann, so dass jeder Knoten jeweils mit genau einem Knotentyp und auch jede Kante bzw. Beziehung jeweils mit genau einem Kantentyp spezifiziert werden kann (vgl. Abbildung 62). Die dritte Anforderung verlangt, dass die Kontrollflusssemantik des Knotens mit genau einem der möglichen Semantikschemas spezifiziert werden kann (vgl. Abbildung 63). Die möglichen Semantikschemas werden durch die Zuordnung des Knotens zum Knotentyp bestimmt. Für jede elementare Bedingung des Semantikschemas muss genau eines der möglichen Kontrollflussmuster ausgewählt werden. Die möglichen Kontrollflussmuster werden durch die elementare Bedingung des Semantikschemas der Geschäftsprozessmodellierungssprache beschrieben, die auf mehrere Kontrollflussmuster verweisen kann (vgl. Abbildung 30).

Nachdem die Anforderungen zur Beschreibung der Ablaufreihenfolge der Elemente eines Geschäftsprozessmodells beschrieben wurden, wurde im nächsten Abschnitt das Vorgehen mit BPMN-Modellen definiert (vgl. Abbildung 64). Die BPMN-Modelle werden als Grundlage für die prototypische Realisierung des Software-Werkzeuges (vgl. Kapitel 7.2)

eingesetzt. Im Wesentlichen besteht das Vorgehen aus vier Schritten. Zunächst muss die Kontrollflusssemantik für die verwendete Geschäftsprozessmodellierungssprache ausgewählt werden (Anforderung 1). Anschließend können die Knoten, Kanten und Beziehungen des Geschäftsprozessmodells mit dem Modell aus Abbildung 63 beschrieben sowie mit den entsprechenden Knoten- und Kantentypen der ausgewählten Geschäftsprozessmodellierungssprache spezifiziert werden (Anforderung 2). Im nächsten Schritt können die impliziten Abhängigkeiten definiert werden, da diese nicht mit den Notationselementen der Modellierungssprache visualisiert sind. Abschließend kann jedem Knoten des Geschäftsprozessmodells ein Semantikschema zugewiesen werden. Die Parameter der jeweiligen Kontrollflussmuster wurden bereits in Kapitel 4.1.2 definiert sowie in Abschnitt 6.2 punktuell für die einzelnen Kontrollflussmuster erweitert. Beispielsweise wurde die partielle Funktion Struc eingeführt, um auf die zu spezifizierenden strukturierten Knoten der Kontrollflussmuster wcp_7, wcp_9 und wcp_{30} verweisen zu können. Die Parameter der Kontrollflussmuster wurden in modell- und musterspezifische Parameter untergliedert, da es Parameter gibt, die von dem zu spezifizierenden Knoten unabhängig und somit nur von dem ausgewählten Kontrollflussmuster abhängig sind. Deutlich wird dies unter anderem an dem Kontrollflussmuster exklusive Auswahl (wcp_4), bei dem die Parameter Split, $<_{XOR}$, ArcCond und Default spezifiziert werden müssen. Die Split-Funktion muss mit dem Funktionswert XOR spezifiziert werden, wobei der Funktionswert von dem zu spezifizierenden Knoten unabhängig und somit nur vom Kontrollflussmuster (musterspezifische Parameter) abhängig ist. Die modellspezifischen Parameter sind dahingegen vom Knoten abhängig und können von Knoten zu Knoten variieren, wie beispielsweise die Reihenfolge der Auswertungssequenz ($<_{XOR}$), die Kantenbedingungen (ArcCond) und die Standardkante (Default) des Kontrollflussmusters exklusive Auswahl (wcp_4).

Abgerundet wurde das Kapitel mit der Anwendung des entwickelten Vorgehens zur Beschreibung der Ablaufreihenfolge der Elemente von Geschäftsprozessmodellen an ausgewählten Geschäftsprozessmodellen aus [DrKO17] und [RuQu12]. Für jede vorgestellte Geschäftsprozessmodellierungssprache aus Kapitel 3, deren Kontrollflusssemantik in Kapitel 5.3 definiert wurde, wurde ein Geschäftsprozessmodell illustriert und die mögliche Ablaufreihenfolge der Elemente der Geschäftsprozessmodelle beschrieben. Aus der Anwendung der Methode wurde deutlich, dass die Beschreibung der Ablaufreihenfolge der Elemente eines Geschäftsprozessmodells sehr aufwendig sein kann, wenn die Beschreibung der Kontrollflusssemantik des Geschäftsprozessmodells nicht durch ein IT-System unterstützt wird. Eine Vielzahl der notwendigen Spezifikationsschritte kann von einem IT-System automatisch übernommen werden, wenn die 3 definierten Anforderungen erfüllt sind. Beispielsweise kann dies anhand des Petri-Netz-Geschäftsprozessmodells verdeutlicht werden. Die Beschreibung des Petri-Netz-Geschäftsprozessmodells mit dem Modell aus Abbildung 63

(vgl. Tabelle 46) kann automatisiert durch ein IT-System erfolgen, wenn für die Knoten- und Kantentypen die Notationselemente definiert sind. Dementsprechend müssen nur für die Knotentypen Stelle und Transition sowie für den Kantentyp Sequenzfluss Notationselemente definiert werden, mit denen auch das Geschäftsprozessmodell erstellt sein muss. Die Zuweisung der Knoten zu den Semantikschemas (vgl. Tabelle 47) kann ebenfalls durch das IT-System automatisch übernommen werden, wenn die Semantikschemas der ausgewählten Geschäftsprozessmodellierungssprache dem IT-System zur Verfügung stehen. Ein manueller Eingriff ist nur erforderlich, wenn für einen Knoten im Geschäftsprozessmodell mehrere Semantikschemas ermittelt werden. Dieses Szenario kann bei den ausgewählten Geschäftsprozessmodellen nicht eintreten, da die Semantikschemas in Kapitel 5.3 so definiert wurden, dass für einen Knoten nur ein Semantikschema zutreffen kann. Der Aufwand der Spezifikation der modellspezifischen Paramater ist von den ausgewählten Kontrollflussmustern abhängig und davon, ob für die Kantentypen Kantenbedingungen existieren. Bei Petri-Netzen existieren keine Kantenbedingungen für die Kantentypen und es existiert auch keine zu spezifizierende Kontrollflusssemantik «Während der Ausführung», sodass mit dieser Angabe, die Tabelle 48 auf zwei Zeilen bzw. die Spezifikation der exklusiven Auswahl (wcp_4) für die Knoten t_5 und t_{27} reduziert werden kann. Die signifikante Reduktion der zu spezifizierenden Aspekte, kann auch für die anderen Geschäftsprozessmodelle demonstriert werden. Die Anwendung der Methode ist aus diesem Grund nur mit einem IT-System zu empfehlen, dass den Anwender bei der Beschreibung der Ablaufreihenfolge der Elemente eines Geschäftsprozessmodells unterstützt. Ein mögliches IT-System wird in Kapitel 7.2 vorgestellt.

7 Evaluation der Beschreibung der Kontrollflusssemantik

In den vorherigen Kapiteln wurden zwei Methoden vorgestellt. Eine Methode zur Beschreibung der Kontrollflusssemantik von Geschäftsprozessmodellierungssprachen (vgl. Kapitel 5) und eine darauf aufbauende Methode für die präzise Beschreibung der Ablaufreihenfolge der Elemente in den Geschäftsprozessmodellen (vgl. Kapitel 6). Mit dieser Beschreibung sollen unter anderem ungewollte Mehrdeutigkeiten in den Geschäftsprozessmodellen bei der Ablaufreihenfolge der Elemente vermieden werden. Häufig entstehen ungewollte Mehrdeutigkeiten bei dem OR-Verknüpfungsoperator einer EPK [KeNS92] und dem inklusiven Gateway der BPMN [BPMN2.0.2] aufgrund der Nicht-Lokalität einer Oder-Zusammenführung (siehe [Kind06]). Aus diesem Grund wird nachfolgend die entwickelte Methode aus Kapitel 6 am Beispiel einer Oder-Zusammenführung evaluiert. Hierbei soll in den Geschäftsprozessmodellen die Ablaufreihenfolge der Elemente mit und ohne Anwendung der vorgestellten Methode unter Berücksichtigung der Ergebnisse einer empirischen Untersuchung verglichen werden. Darüber hinaus soll ein Feedback von potenziellen Anwendern eingeholt werden, um erste Anhaltspunkte für die Vermutungen der einfachen Benutzbarkeit, leichten Erlernbarkeit, einfachen Beschreibung der Kontrollflusssemantik (bzw. Erwartungshaltung), Nützlichkeit und leichten Verständlichkeit zu sammeln. Die prototypische Realisierung der Methode in einem Software-Werkzeug wurde auf einer Fachtagung [Dres18] vorgestellt und evaluiert [Lund01]. Das Werkzeug zur präzisen Beschreibung der Ablaufreihenfolge der Elemente eines Geschäftsprozessmodells wird in Kapitel 7.2 vorgestellt, dass unter anderem auch eine interaktive Simulation von Geschäftsprozessmodellen ermöglicht. Die durchgeführten Untersuchungen in diesem Kapitel dienen ausschließlich als erster Entwurf, um Anhaltspunkte für Vermutungen über die Methode oder auch für den Prototyp von potenziellen Anwendern zu erhalten. Auf dieser Grundlage können detaillierte empirische Untersuchungen durchgeführt werden.

7.1 Empirische Untersuchung

Es wird geprüft, ob durch den Einsatz der entwickelten Methode die ungewollten Mehrdeutigkeiten einer Oder-Zusammenführung reduziert werden können. Dabei wird untersucht, ob eine Oder-Zusammenführung entgegen der Empfehlung der sieben Prozessmodellierungsrichtlinien (7PMG) von [MeRv10] doch zur Modellierung von

Geschäftsprozessen verwendet werden kann. Zudem werden die Variablen Benutzbarkeit, Erlernbarkeit, Erwartungshaltung, Nützlichkeit und Verständlichkeit betrachtet. Besonderen Wert wird dabei auf die Nachvollziehbarkeit der Ergebnisse gelegt, sodass die Untersuchung in Anlehnung an die fünf Phasen von [Maye13, Atte10, Diek17], vgl. Abbildung 89 durchgeführt wird.

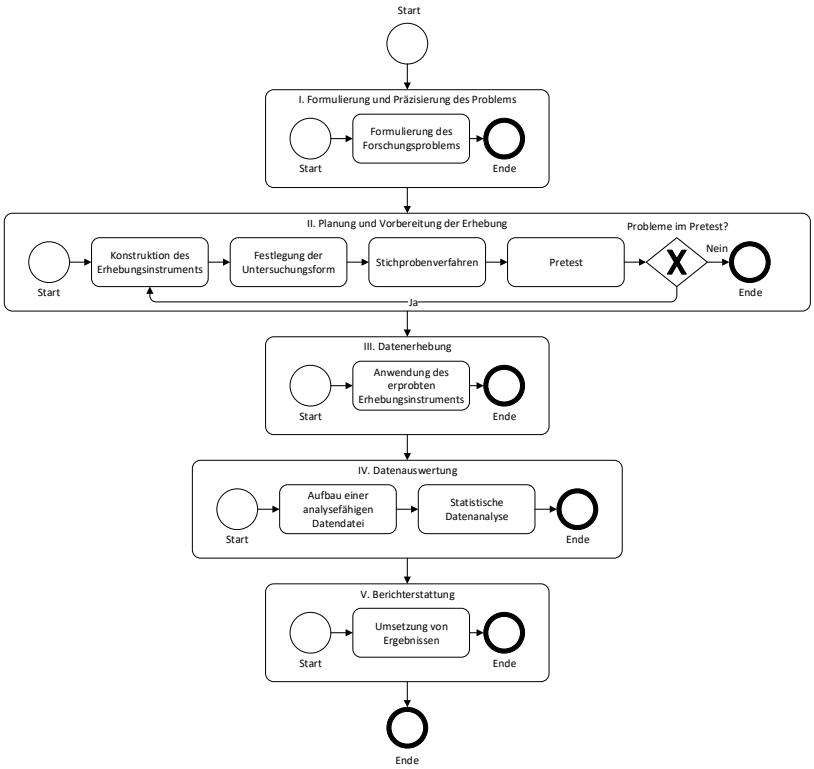


Abbildung 89: Überblick - Phasen einer empirischen Untersuchung (In Anlehnung an [Diek17])

7.1.1 Formulierung und Präzisierung des Problems

Zur Herleitung von Hypothesen, die als Grundlage für die empirische Untersuchung verwendet werden, wurden verwandte Arbeiten einbezogen. Nachfolgend werden relevante Arbeiten betrachtet. Die Arbeit von [MeNA07] stellt sieben Prozessmodellierungsrichtlinien (7PMG) auf. Durch die Anwendung der Richtlinien sollen ungewollte Mehrdeutigkeiten vermieden werden. Regel 5 der 7PMG lautet „Avoid OR routing elements“ [MeRv10].

Mit dieser Regel wird empfohlen Oder-Verzweigungen und -Zusammenführungen in Geschäftsprozessmodellen nicht zu verwenden, da aufgrund der Nicht-Lokalität einer Oder-Zusammenführung (siehe [Kind06]) ungewollte Mehrdeutigkeiten entstehen. Beispielsweise könnte der Kontrollfluss fortgeführt werden, wenn eines der Elemente im Vorbereich ausgeführt wurde (Kontrollflussmuster: strukturierter Diskriminator (wcp9)). Es kann aber auch die Möglichkeit bestehen, dass bei der Zusammenführung auf die Ausführung eines zweiten Elementes im Vorbereich gewartet werden muss (Kontrollflussmuster: strukturierte synchronisierte Zusammenführung (wcp7)). Diese Entscheidungen können möglicherweise von einer Oder-Verzweigung abhängen, so dass zur Fehlerreduktion empfohlen wird, die Oder-Zusammenführung entweder nicht zu verwenden oder um eine Zusammenführungsbedingung zu erweitern [Ritt00]. Basierend auf diesen Erkenntnissen soll die Gültigkeit der Regel 5 der 7PMG durch eine Geschäftsprozessmodellannotation überprüft werden. Hierfür wird die folgende Hypothese formuliert:

– *Hypothese 1:* Die Annotation der Elemente in den Geschäftsprozessmodellen mit den Kontrollflussmustern der Workflow Pattern Initiative verbessert die Verständlichkeit der Kontrollflusssemantik einer Oder-Zusammenführung.

Die Kontrollflussmuster können nicht nur zur Formulierung einer Zusammenführungsbedingung einer Oder-Zusammenführung verwendet werden, sondern auch, um die Kontrollflusssemantik von Geschäftsprozessmodellierungssprachen (vgl. Kapitel 5) zu definieren und die Ablaufreihenfolge der Elemente in den Geschäftsprozessmodellen (vgl. Kapitel 6) präzise zu beschreiben. Um für die entwickelte Methode zur Beschreibung der Ablaufreihenfolge der Elemente in Geschäftsprozessmodellen ein erstes Feedback zu erhalten, sollen die Vermutungen der einfachen Benutzbarkeit, leichten Erlernbarkeit, einfachen Beschreibung der Kontrollflusssemantik (bzw. Erwartungshaltung), Nützlichkeit und leichten Verständlichkeit evaluiert werden. Hierfür können die folgenden Hypothesen formuliert werden:

– *Hypothese 2* (Erlernbarkeit): Die Methode zur Beschreibung der Ablaufreihenfolge der Elemente der Geschäftsprozessmodelle mit den Kontrollflussmustern der Workflow Pattern Initiative ist einfach zu erlernen.

– *Hypothese 3* (Benutzbarkeit): Die Benutzung der Methode zur Beschreibung der Ablaufreihenfolge der Elemente der Geschäftsprozessmodelle mit den Kontrollflussmustern der Workflow Pattern Initiative ist einfach.

– *Hypothese 4* (Verständlichkeit): Die Methode zur Beschreibung der Ablaufreihenfolge der Elemente der Geschäftsprozessmodelle mit den Kontrollflussmustern der Workflow Pattern Initiative ist verständlich.

– *Hypothese 5* (Verständlichkeit): Die Methode zur Beschreibung der Ablaufreihenfolge der Elemente der Geschäftsprozessmodelle mit den Kontrollflussmustern der Workflow Pattern Initiative ist für Anfänger und Experten gleichermaßen verständlich.

– *Hypothese 6* (Erwartungshaltung): Die Methode zur Beschreibung der Ablaufreihenfolge der Elemente der Geschäftsprozessmodelle ermöglicht eine einfache Beschreibung der Kontrollflusssemantik.

– *Hypothese 7* (Nützlichkeit): Die Annotation der Elemente der Geschäftsprozessmodelle mit den Kontrollflussmustern ist nützlich, um die Kontrollflusssemantik der Geschäftsprozessmodelle besser zu verstehen.

Die sieben aufgestellten Hypothesen sollen mit Hilfe einer empirischen Untersuchung verifiziert oder falsifiziert werden. Ausgehend von diesen ermittelten ersten Anhaltspunkten können detaillierte empirische Untersuchungen durchgeführt werden.

7.1.2 Planung und Vorbereitung der Erhebung

Bei der Planung und Vorbereitung der Erhebung sind zunächst die verwendeten Begriffe zu definieren. Die relevanten Begriffe sind in diesem Zusammenhang das Geschäftsprozessmanagement, die Kontrollflusssemantik und -muster sowie die entwickelte Methoden aus Kapitel 6. Um für die Teilnehmer der empirischen Untersuchung ein gemeinsames Grundverständnis zu schaffen, wurden für die relevanten Begriffe Schulungsmaterialien erstellt. Mit einem circa 4-minütigem Informationsvideo wurden die Begriffe Geschäftsprozessmanagement, Kontrollflusssemantik und das Konzept der entwickelten Methode aus Kapitel 6 vorgestellt. Ebenfalls wurden ausgewählte Kontrollflussmuster beschrieben. Es wurde das audiovisuelle Medium Video ausgewählt, da alle Teilnehmer dieselben erforderlichen Grundkenntnisse für die Untersuchung in der gleichen Form verständlich vorgestellt bekommen sollten. Durch die gleichzeitige Ansteuerung von mehreren Sinnen, d. h. Sehen und Hören, kann das Gehirn die Informationen tendenziell besser aufnehmen [Maye99]. Für die Erstellung des Informationsvideos wurden die Mitarbeiter der Forschungsgruppe Betriebliche Informationssysteme am Institut für Angewandte Informatik und Formale Beschreibungsverfahren (AIFB) des Karlsruher Instituts für Technologie (KIT) in

Feedbackzyklen mit einbezogen, um ein qualitativ hochwertiges Informationsvideo zur Verfügung stellen zu können.

Nachdem die Grundvoraussetzungen zur Verständlichkeit der Untersuchung geschaffen wurden, kann mit der Operationalisierung begonnen werden. Mit der Operationalisierung werden Phänomene direkt messbar gemacht, indem die Variablen und Indikatoren für die relevanten Begriffe der Hypothese definiert werden. Die Ausprägungsgrade eines Phänomens können bei der Datenerhebung erfasst werden [Atte10]. Bei der ersten Hypothese soll überprüft werden, ob sich die Verständlichkeit der Kontrollflusssemantik einer Oder-Zusammenführung durch die Annotation mit Kontrollflussmustern verbessert. Hieraus kann der Begriff Verbesserung der Verständlichkeit der Oder-Zusammenführung abgeleitet werden. Darüber hinaus kann die Bekanntheit der Kontrollflussmuster der Workflow Pattern Initiative bei der Betrachtung der Hypothese berücksichtigt werden. Als Indikatoren für die Bekanntheit der Kontrollflussmuster werden Ja und Nein für die Nominalskala festgelegt. Zur Messbarmachung der Verbesserung der Verständlichkeit der Oder-Zusammenführung können Geschäftsprozessmodelle mit und ohne Annotation von Kontrollflussmustern definiert werden, deren Kontrollflusssemantik interpretiert werden muss. Als Indikatoren dieser Variable kann eine Nominalskala für die mögliche Kontrollflusssemantik der Oder-Zusammenführung festgelegt werden. Mögliche Geschäftsprozessmodelle mit einer Oder-Zusammenführung und die Ablaufreihenfolge der Elemente in den Geschäftsprozessmodellen können aus [Ritt00] abgeleitet werden. Die zu bewertenden Geschäftsprozessmodelle werden zum einen ohne die Annotation von Kontrollflussmustern in Abbildung 27 sowie zum anderen mit der Annotation von Kontrollflussmustern in Abbildung 90 illustriert. Die mögliche Ablaufreihenfolge der Elemente im ersten Geschäftsprozessmodell (Abbildung 27a bzw. Abbildung 90a) besteht daraus, dass

(i) die Aufgaben 2, 3 und 4 ausgeführt sein müssen, damit Aufgabe 5 ausgeführt werden kann.

(ii) nach der ersten ausgeführten Aufgabe (Aufgabe 2, 3 oder 4), Aufgabe 5 ausgeführt werden kann.

(iii) abhängig davon, welche Aufgaben von der Oder-Verzweigung (OR) aktiviert wurden, entsprechend die Aufgabe 2, 3 und/oder 4 ausgeführt sein müssen, damit Aufgabe 5 ausgeführt werden kann.

(iv) die Kontrollflusssemantik der Oder-Zusammenführung (OR) nicht eindeutig ist.

(v) 2 der 3 Aufgaben ausgeführt sein müssen, damit Aufgabe 5 ausgeführt werden kann.

Die mögliche Ablaufreihenfolge der Elemente im zweiten Geschäftsprozessmodell (Abbildung 27b bzw. Abbildung 90b) kann analog definiert werden, mit Ausnahme von Punkt (iii), da keine Oder-Verzweigung existiert. Im dritten Geschäftsprozessmodell (Abbildung 27c bzw. Abbildung 90c) besteht die mögliche Ablaufreihenfolge der Elemente daraus, dass

(i) die Aufgaben 4 und 5 ausgeführt sein müssen, damit Aufgabe 7 ausgeführt werden kann.

(ii) nach der ersten ausgeführten Aufgabe (Aufgabe 4 oder 5), Aufgabe 7 ausgeführt werden kann.

(iii) die Kontrollflusssemantik der Oder-Zusammenführung (OR) nicht eindeutig ist.

Bei der zweiten bis siebten Hypothese sollen die Vermutungen der einfachen Benutzbarkeit, leichten Erlernbarkeit, einfachen Beschreibung der Kontrollflusssemantik (bzw. Erwartungshaltung), Nützlichkeit und leichten Verständlichkeit für die entwickelte Methode aus Kapitel 6 betrachtet werden. Um differenziertere Aussagen über die Vermutungen treffen zu können, sollten die Kenntnisse im Geschäftsprozessmanagement, die Bekanntheit von Geschäftsprozessmodellierungssprachen und die Bekanntheit der Kontrollflussmuster der Workflow Pattern Initiative der Probanden berücksichtigt werden. Die Kenntnisse im Geschäftsprozessmanagement können mit einer Ordinalskala nach dem Schulnotensystem festgelegt werden, d. h. die Zahl 1 bedeutet sehr gute Kenntnisse und die Zahl 6 bedeutet keine Kenntnisse im Geschäftsprozessmanagement. Für die Bekanntheit von Geschäftsprozessmodellierungssprache können die Standardsprachen aus Kapitel 3 definiert werden. Für die Kontrollflussmuster der Workflow Pattern Initiative kann identifiziert werden, ob diese bekannt oder nicht bekannt sind. Dementsprechend können die Indikatoren Ja und Nein für die Nominalskala festgelegt werden.

Für die Evaluierung der einfachen Benutzbarkeit, leichten Erlernbarkeit, einfachen Beschreibung der Kontrollflusssemantik (bzw. Erwartungshaltung), Nützlichkeit und leichten Verständlichkeit existieren verschiedene etablierte Werkzeuge, auf die im Folgenden zurückgegriffen werden soll:

– Die *Erlernbarkeit* soll messen, ob die Methode leicht zu erlernen ist und diese trotz seltener Verwendung fehlerfrei anwendet werden kann. In Anlehnung an den International Organization for Standardization Norm (IsoNorm) Fragebogen [PrAn93] lässt sich die Aussage *die Methode ist gut ohne fremde Hilfe erlernbar* formulieren. Darüber hinaus kann in Anlehnung an den International Organization for Standardization Metrics (IsoMetrics) Fragebogen [WiGH96], die Aussage *auch bei seltenem Gebrauch ist es kein Problem die Methode wieder anzuwenden*

abgeleitet werden. Die ursprüngliche Aussage lautet: *„Auch bei seltenem Gebrauch ist es kein Problem sich wieder in die Software hineinzufinden.“*. Die Bewertung kann mit einer Ordinalskala im Schulnotensystem erfolgen.

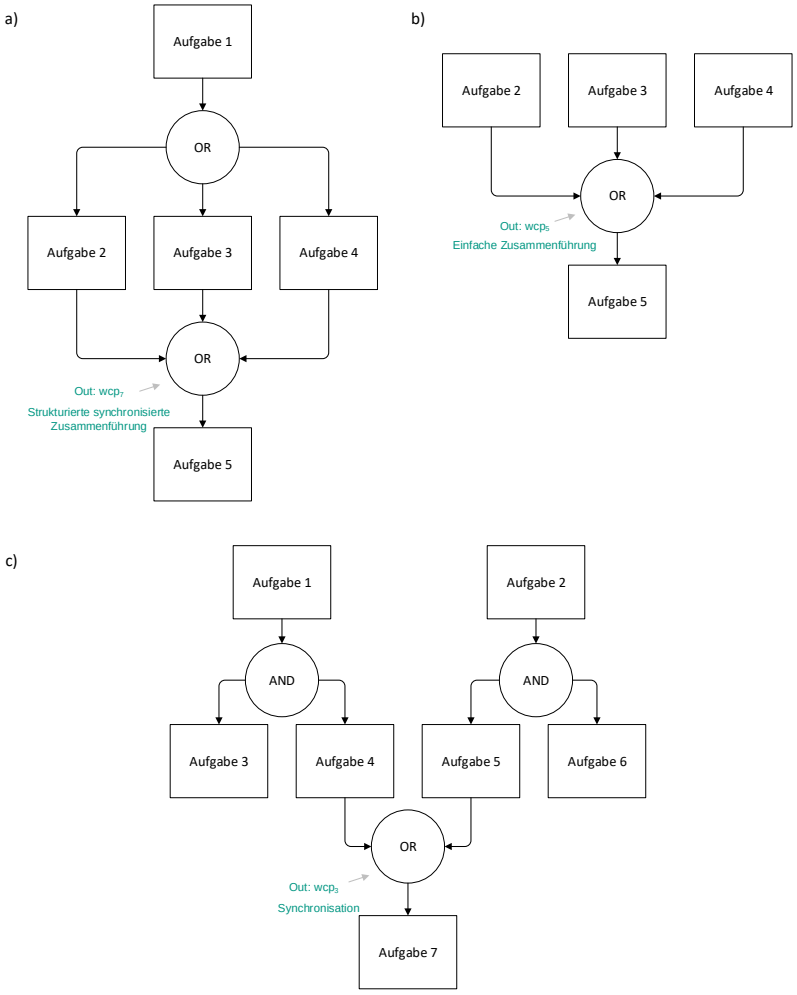


Abbildung 90: Beispiel - Oder-Zusammenführung mit Kontrollflussmustern annotiert

– Die *Benutzbarkeit* ermöglicht die Messung der Anwendbarkeit durch den Anwender, um zu identifizieren, ob die Anwendung der Methode für den Anwender problematisch ist oder ob die Kontrollflussmuster den Elementen einfach zugewiesen werden können. In Anlehnung an den modular evaluation of key Components of User Experience (meCUE) Fragebogen [MiRi13] kann die zu bewertende

Aussage *die Methode lässt sich einfach benutzen* abgeleitet werden. Die Bewertung kann mit einer Ordinalskala im Schulnotensystem erfolgen.

– Die *Verständlichkeit* zur Beschreibung der Ablaufreihenfolge der Elemente in den Geschäftsprozessmodellen kann in Anlehnung an den Fragebogen User Experience Questionnaire (UEQ) [RaST13] evaluiert werden, mit der Aussage *die Methode ist verständlich*. Die Verständlichkeit kann abhängig von verschiedenen Zielgruppen bewertet werden. Die Methode sollte dementsprechend nicht nur von Personen verstanden werden, die fortgeschrittene Kenntnisse im Geschäftsprozessmanagement haben, sondern auch von Personen, die nur Grundkenntnisse besitzen. Daraus kann die zu bewertende Aussage *die Methode ist für Anfänger und Experten gleichermaßen verständlich* abgeleitet werden, die sich in Anlehnung an den International Organization for Standardization Norm (IsoNorm) Fragebogen [PrAn93] definieren lässt. Die Bewertung kann mit einer Ordinalskala im Schulnotensystem erfolgen.

– Mit der *Erwartungshaltung* kann gemessen werden, ob die avisierten Erwartungen des Anwenders erfüllt werden. In der Untersuchung soll festgestellt werden, ob die Methode den Erwartungen zur einfachen Beschreibung der Kontrollflusssemantik entspricht. In Anlehnung an den Fragebogen User Experience Questionnaire (UEQ) [RaST13] zur Bewertung eines Produktes kann die Aussage *die Methode erfüllt meine Erwartungen an eine einfache Beschreibung der Kontrollflusssemantik* abgeleitet werden. Die Bewertung kann mit einer Ordinalskala im Schulnotensystem erfolgen.

– Die *Nützlichkeit* ermöglicht u. a. die Messung, ob die Kontrollflusssemantik der Geschäftsprozessmodelle durch die Anwendung der Methode besser zu verstehen ist und somit zur Reduktion der ungewollten Mehrdeutigkeiten in Geschäftsprozessmodellen beiträgt. In Anlehnung an den UEQ Fragebogen [RaST13] zur Bewertung eines Produktes kann die zu bewertende Aussage *die Methode ist nützlich, um die Kontrollflusssemantik besser zu verstehen* abgeleitet werden. Eine weitere Frage, die aus dem meCUE Fragebogen [MiRi13] abgeleitet werden kann, lautet: *ich würde die Methode gerne nutzen, da ich die Nützlichkeit sehe*, wobei der ursprüngliche Wortlaut lautet: *„Ich kann es kaum erwarten, das Produkt erneut zu verwenden.“* [MiRi13]. Die Bewertung dieser beiden Fragen kann mit einer Ordinalskala im Schulnotensystem erfolgen.

Nachdem die verwendeten Begriffe in den Hypothesen operationalisiert wurden, wird deutlich, dass sich die Befragung als Erhebungsinstrument eignet. Damit die Befragung

einem möglichst breiten Teilnehmerkreis mit angemessenem Aufwand zur Verfügung gestellt werden kann, wurde als Form der Online-Fragebogen ausgewählt. Aufgrund der Anonymität eines Fragebogens wurden zusätzlich die personenbezogenen Daten *Geschlecht* und *Altersgruppe* erfragt, um allgemeine Informationen über die Teilnehmer zu erhalten. Darüber hinaus sollte der Fragebogen dazu verwendet werden, um ein freies *Feedback* bzw. *Verbesserungsvorschläge* für die entwickelte Methode zu erhalten. Zur Motivation der zu akquirierenden Personen wurden zwei 25 Euro Gutscheine unter allen Teilnehmern der Befragung zufällig verlost. Die Teilnahme an der Verlosung wurde durch ein nicht verknüpftes Feld mit der Angabe der eMail-Adresse ermöglicht (vgl. Abbildung 91, Seite 8). Diese Möglichkeit wurde als Entscheidungsvariante gewählt, um zu gewährleisten, dass die Mindeststichprobenanzahl von 30 bis 50 Teilnehmern erreicht wird [Hill98].

Der Aufbau des Fragebogens, der sich in 8 Seiten untergliedert, ist in Abbildung 91 illustriert:

- Die erste Seite beinhaltet allgemeine Informationen und Hinweise zur Untersuchung sowie die Abfrage der *Altersgruppe* und des *Geschlechts*. Darüber hinaus wird erfragt, ob die *Kontrollflussmuster* der Workflow Pattern Initiative bekannt sind.

- Die zweite Seite beinhaltet das *Informationsvideo* zu den Begriffen Geschäftsprozessmanagement, Kontrollflussmuster und der entwickelten Methode aus Kapitel 6. Darauf aufbauend wird erfragt, ob alle Informationen in dem Video verstanden wurden. Zusätzlich werden die *Kenntnisse im Geschäftsprozessmanagement* abgefragt sowie gefragt, welche *Modellierungssprachen* zur Modellierung von Geschäftsprozessmodellen von den Teilnehmern eingesetzt werden.

- Auf der dritten Seite des Fragebogens werden die Aussagen zur entwickelten Methode bewertet, d. h. *die Methode ist nützlich, um die Kontrollflusssemantik besser zu verstehen*; *die Methode lässt sich einfach benutzen; die Methode ist verständlich; auch bei seltenem Gebrauch ist es kein Problem die Methode wieder anzuwenden; die Methode ist gut ohne fremde Hilfe erlernbar; die Methode ist für Anfänger und Experten gleichermaßen verständlich* und *die Methode erfüllt meine Erwartungen an eine einfache Beschreibung der Kontrollflusssemantik*.

- Auf der vierten, fünften und sechsten Seite soll die Ablaufreihenfolge der Elemente in den Geschäftsprozessmodellen aus Abbildung 27 und Abbildung 90 jeweils mit den oben beschriebenen Antwortmöglichkeiten beschrieben werden.

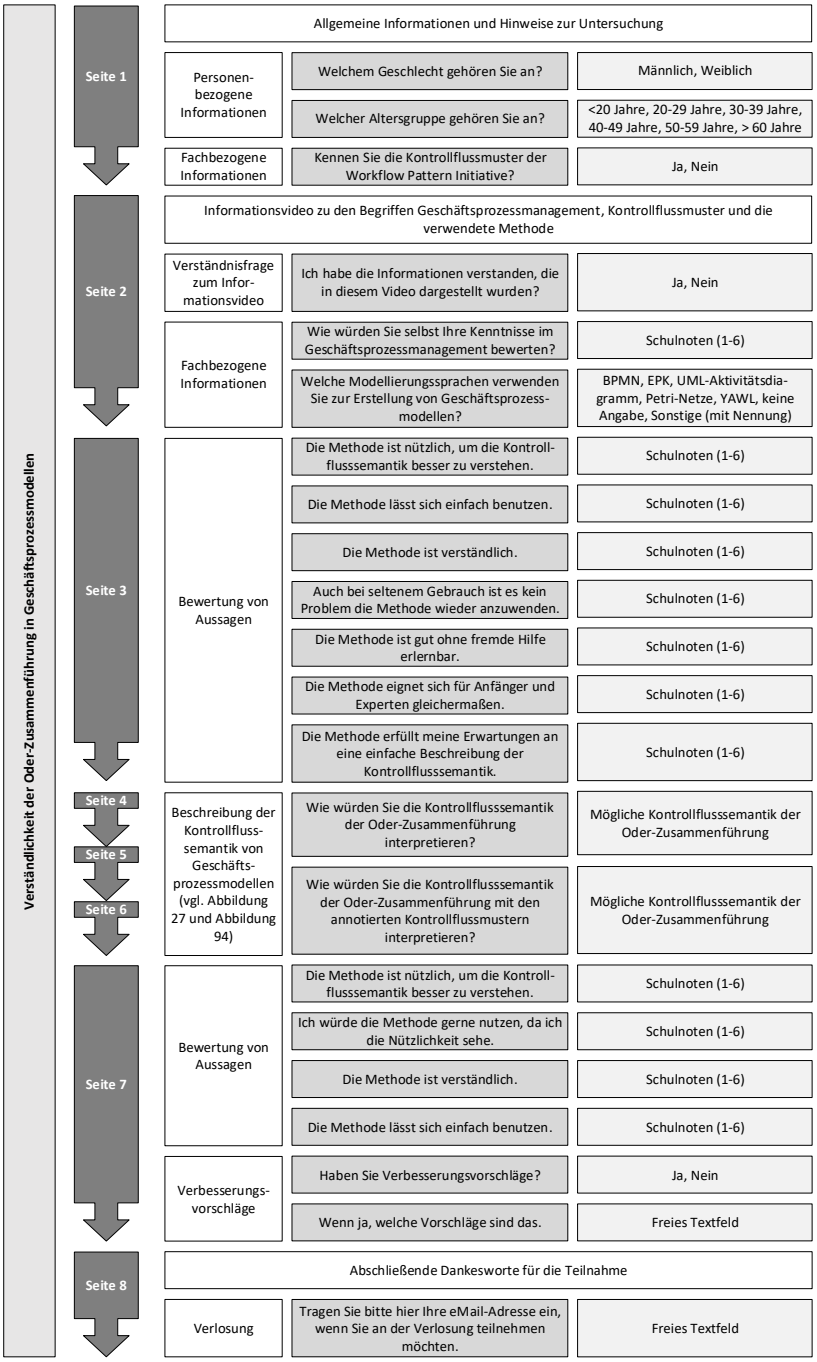


Abbildung 91: Überblick - Fragebogen zur entwickelten Methode

– Auf der siebten Seite werden ausgewählte Fragen erneut gestellt, um zu identifizieren, ob es nach der Interpretation der Ablaufreihenfolge der Elemente in den Geschäftsprozessmodellen zu Meinungsänderungen in den Aussagen gekommen ist. Die meisten Änderungen werden bei den folgenden drei Aussagen vermutet: *die Methode ist nützlich, um die Kontrollflusssemantik besser zu verstehen; die Methode ist verständlich* und *die Methode lässt sich einfach benutzen*. Aus diesem Grund werden auch nur diese Aussagen erneut bewertet. Darüber hinaus wird nach der Interpretation der Ablaufreihenfolge der Elemente in den Geschäftsprozessmodellen die zu bewertende Aussage, *ich würde die Methode gerne nutzen, da ich die Nützlichkeit sehe*, abgefragt. Zusätzlich sind das *Feedback* bzw. die *Verbesserungsvorschläge* der Methode in diese Seite integriert.

– Abgerundet wird der Online-Fragebogen auf der achten Seite mit abschließenden Dankesworten und der Möglichkeit der Teilnahme an einer *Verlosung*.

Nachdem der Fragebogen entwickelt wurde, wurden die Rahmenbedingungen für die Durchführung der empirischen Untersuchung festgelegt. Dementsprechend wurde eine nicht-experimentelle Querschnittsuntersuchung durchgeführt, da der Untersuchungsprozess nicht wiederholt wird und darüber hinaus auch keine Vergleichs- oder Kontrollgruppe vorgesehen ist. Zudem handelt es sich um eine originäre Primäruntersuchung, da die konkrete Untersuchung erstmalig und ohne bisher existierende Datenbasis vollzogen wird. Als potenzielle Untersuchungsobjekte wurden

– Doktoranden mit einem verwandten Forschungsschwerpunkt,
– Studierende des Karlsruher Instituts für Technologie (KIT), die Veranstaltungen mit dem Bezug auf das Geschäftsprozessmanagement belegt haben,
– Mitarbeiter der Dienstleistungseinheit Organisationsentwicklung und Prozesse (OEP) des KITs,
– Mitarbeiter der Promatis Software AG, die die Methode praktisch einsetzen und
– Personen durch eigene Kontakte, mit und ohne Erfahrung im Geschäftsprozessmanagement

definiert. Darüber hinaus wurde dazu aufgerufen, die Umfrage an Personen weiterzuleiten, die an der Untersuchung ein Interesse haben könnten. Der Feldzugang kann als halboffen definiert werden, da der Ersteller der empirischen Untersuchung derzeit selbst am KIT tätig ist und die Veranstaltungen *Modellierung von Geschäftsprozessen* und *Workflow-Management* betreut sowie die Mitarbeiter der Dienstleistungseinheit Organisationsentwicklung und Prozesse (OEP) des KIT als externer Berater unterstützt. Abgerundet wurde diese Phase mit einem Pretest zur Sicherung der Qualität der durchzuführenden Untersuchung, damit

Fehler und Missverständnisse vor der Durchführung der Befragung identifiziert werden können. Der entwickelte Fragebogen wurde durch die Mitarbeiter der Forschungsgruppe Betriebliche Informationssysteme am AIFB des KIT getestet, bevor der Fragebogen mit dem Informationsvideo für die Teilnehmer aktiviert wurde. Unklarheiten bei der Fragenformulierung, Rechtschreibfehler, Funktionen des Fragebogens, die Dauer der Beantwortung des vollständigen Fragebogens sowie eine generelle thematische Einschätzung wurden erprobt. Aufgrund des Tests wurden sprachliche Verbesserungen vorgenommen. Im Anschluss daran wurde der Fragebogen erneut durch die Mitarbeiter der Forschungsgruppe getestet.

7.1.3 Datenerhebung

Die Datenerhebung erfolgte über ein Online-Umfragetool, bei dem die gespeicherten Antworten exportiert werden können. Die Online-Umfrage wurde im Umfragezeitraum vom 23.04.2018 bis zum 07.05.2018 durchgeführt. Innerhalb der ersten drei Tage konnte die festgelegte Mindestanzahl von 50 Teilnehmern bereits erreicht werden. Der zeitliche Verlauf der Teilnehmerzahlen wird in Abbildung 92 illustriert. Dabei wurden die Antworten täglich geprüft, um Fehlentwicklungen, die trotz durchgeführter Pretests eintreten können, so früh wie möglich zu erkennen. Bei der täglichen Kontrolle konnten allerdings keine Fehlentwicklungen festgestellt werden.

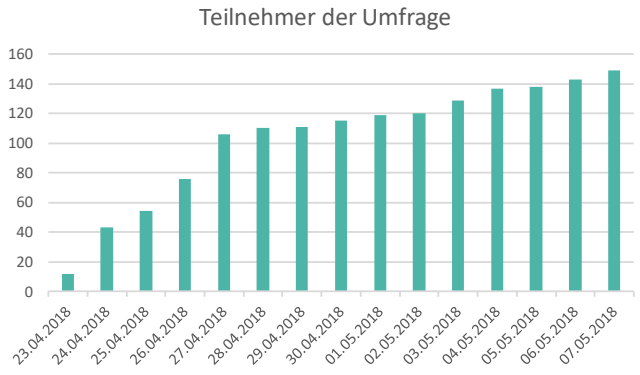


Abbildung 92: Evaluation - Anzahl der Teilnehmer im Befragungszeitraum

7.1.4 Datenauswertung

Die Ergebnisse der Antworten des Online-Fragebogens lassen sich von der verwendeten Plattform als Microsoft Excel (XLS)-Dokument, Comma Separated Values (CSV)-Dokument und als Portable Document Format (PDF)-Dokument exportieren. Aus diesem Grund wurde das Programm Microsoft Excel für die Datenauswertung verwendet. Die zugrundeliegenden 152 Datensätze wurden mit Vorfilterungen bereinigt, indem inhaltlich leere Datensätze (Anzahl: 3), Teilnehmer, die das Informationsvideo nicht verstanden haben (Anzahl: 5) und Teilnehmer, die die Frage zum Verständnis des Informationsvideos nicht beantwortet haben (Anzahl: 12), entfernt wurden. Inhaltlich leere Datensätze sind durch Teilnehmer entstanden, die keine inhaltliche Frage beantwortet, sondern nur die personenbezogenen Daten angegeben haben. Unvollständig beantwortete Fragebögen (Anzahl: 38) wurden berücksichtigt, da nicht beantwortete Fragen durch die Teilnehmer wahrscheinlich absichtlich nicht beantwortet wurden. Von den 38 unvollständig ausgefüllten Fragebögen haben 3 Teilnehmer die Informationen des Videos nicht verstanden. Die nachfolgenden Auswertungen beziehen sich daher auf 132 ausgefüllte Fragebögen, die für die Untersuchung in drei Auswertungsgruppen untergliedert wurden, das heißt,

(1) alle Teilnehmer der Umfrage sowie

(2) Teilnehmer, die aufgrund ihrer Selbsteinschätzung nur Grundkenntnisse im Geschäftsprozessmanagement besitzen und

(3) Teilnehmer, die aufgrund ihrer Selbsteinschätzung fortgeschrittene Kenntnisse im Geschäftsprozessmanagement besitzen.

Es wurden Teilnehmer in die Auswertungsgruppe Grundkenntnisse eingeordnet, wenn diese die Kenntnisse im Geschäftsprozessmanagement mit den Noten ausreichend bis ungenügend bewertet haben (Anzahl der Teilnehmer: 54). In die Auswertungsgruppe fortgeschrittene Kenntnisse wurden die Teilnehmer eingeordnet, wenn sie angegeben haben, dass sie befriedigende bis sehr gute Kenntnisse im Geschäftsprozessmanagement besitzen (Anzahl der Teilnehmer: 78). Die Aufgliederung wurde vorgenommen, um differenzierte Auswertungen vornehmen zu können, insbesondere in Bezug auf die Benutzbarkeit, Erlernbarkeit, Erwartungshaltung, Nützlichkeit und Verständlichkeit sowie zur Identifizierung von Unterschieden bei der Interpretation der Ablaufreihenfolge der Elemente in den Geschäftsprozessmodellen mit einer Oder-Zusammenführung.

Bei den auszuwertenden Datensätzen handelt es sich um 61 weibliche und 71 männliche Teilnehmer, d. h. dass beide Teilnehmergruppen ähnlich stark vertreten sind (vgl.

Abbildung 93, Links). Dahingegen ist die Verteilung der Altersgruppen ungleichmäßig, da etwas mehr als die Hälfte der Teilnehmer (52%) der Altersgruppe 20-29 Jahre angehören, woraufhin aufgrund der definierten Untersuchungsobjekte auf Studierende geschlossen werden kann (vgl. Abbildung 93, Mitte). Die Verteilung der Kenntnisse im Geschäftsprozessmanagement ist annähernd gleich verteilt, mit leichten Abweichungen bei der Angabe von guten und befriedigenden Kenntnissen, wie aus Abbildung 93 (Rechts) hervorgeht. Dementsprechend umfasst die Auswertungsgruppe der fortgeschrittenen Kenntnisse 24 Teilnehmer mehr als die Auswertungsgruppe mit Grundkenntnissen im Geschäftsprozessmanagement. Bei der Datenauswertung wird nicht chronologisch vorgegangen, sondern in Anlehnung an die aufgestellten Hypothesen.

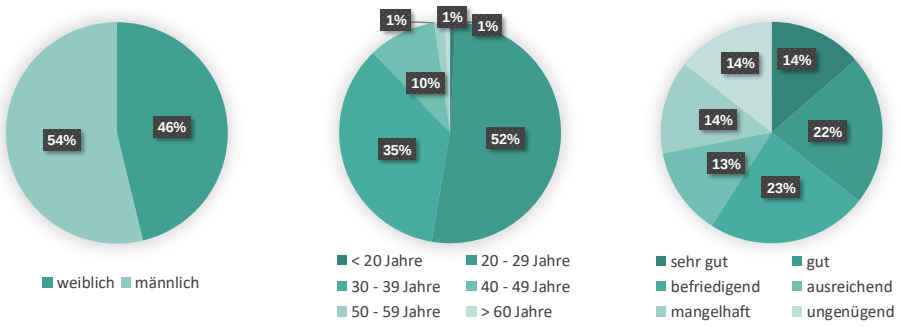


Abbildung 93: Evaluation - Aufteilung der Teilnehmer nach Geschlecht (Links), Altersgruppen (Mitte) und Kenntnissen im Geschäftsprozessmanagement (Rechts)

Für die erste Hypothese wird die Verbesserung der Verständlichkeit der Kontrollflusssemantik einer Oder-Zusammenführung durch die Annotation mit Kontrollflussmustern analysiert. Jedoch wird zunächst die Bekanntheit der Kontrollflussmuster der Workflow Pattern Initiative betrachtet. Die Ereignisverteilung wird in Abbildung 94 und Abbildung 95 visualisiert. Insgesamt haben 42 Teilnehmer bzw. 32% der Teilnehmer angegeben, dass sie die Kontrollflussmuster kennen und 90 Teilnehmer bzw. 68% der Teilnehmer haben die Frage nach der Bekanntheit der Kontrollflussmuster mit Nein beantwortet. Bei der Betrachtung nach dem Kenntnisstand bezüglich des Geschäftsprozessmanagements wurde erwartungsgemäß deutlich, dass die Mehrheit der Teilnehmer mit fortgeschrittenen Kenntnissen die Kontrollflussmuster kennen (53%). Die Teilnehmer, die nur Grundkenntnisse im Geschäftsprozessmanagement besitzen (9%), sind dahingegen sichtlich in der Minderheit. Aufgeschlüsselt nach den einzelnen Kenntnisstufen wird die abnehmende Teilnehmerzahl bei geringeren Kenntnissen ebenfalls deutlich. Außerdem überwiegen die Nein-Antworten in allen Kenntnisstufen (vgl. Abbildung 95).

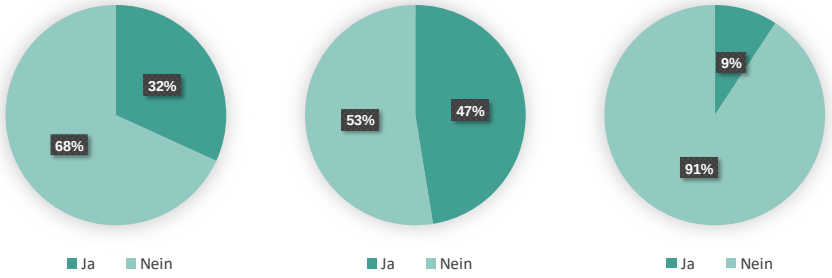


Abbildung 94: Evaluation - Bekanntheit der Kontrollflussmuster der Workflow Pattern Initiative (Links: alle Teilnehmer, Mitte: Teilnehmer mit fortgeschrittenen Kenntnissen im Geschäftsprozessmanagement, Rechts: Teilnehmer mit Grundkenntnissen im Geschäftsprozessmanagement)

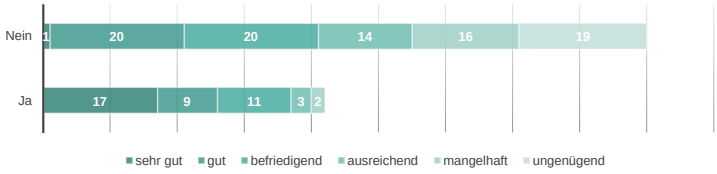


Abbildung 95: Evaluation - Bekanntheit der Kontrollflussmuster der Workflow Pattern Initiative in Abhängigkeit zu den Kenntnissen im Geschäftsprozessmanagement

Im Folgenden werden die Ergebnisse der drei Geschäftsprozessmodelle interpretiert. Dabei wird zunächst das Geschäftsprozessmodell aus Abbildung 27a betrachtet, deren Ergebnisverteilung zur Interpretation der Ablaufreihenfolge der Elemente in Abbildung 96 visualisiert ist. Das gleiche Geschäftsprozessmodell wurde mit Kontrollflussmustern annotiert (vgl. Abbildung 90a), welches mit den gleichen Antwortmöglichkeiten bewertet werden sollte. Die Ergebnisverteilung wird in Abbildung 97 veranschaulicht. Im Folgenden werden die Antwortmöglichkeiten mit deren Ergebnisverteilung in absoluten Zahlen, in Abhängigkeit der Auswertungsgruppen sowie mit bzw. ohne Annotation der Kontrollflussmuster dargestellt. Zum Beispiel beschreibt das Ergebnis 54 / 16, dass 54 Teilnehmer die Antwort für das Geschäftsprozessmodell ohne Kontrollflussmuster sowie 16 Personen die Antwort für das Geschäftsprozessmodell mit Kontrollflussmustern ausgewählt haben.

– **Antwort 1:** Die Aufgaben 2, 3 und 4 müssen ausgeführt sein, damit Aufgabe 5 ausgeführt werden kann.
  Ergebnis: 1. Auswertungsgruppe: 2 / 2 Teilnehmer; 2. Auswertungsgruppe: 0 / 2 Teilnehmer und 3. Auswertungsgruppe: 2 / 0 Teilnehmer

- **Antwort 2:** Nach der ersten ausgeführten Aufgabe (Aufgabe 2, 3 oder 4) kann Aufgabe 5 ausgeführt werden.
  Ergebnis: 1. Auswertungsgruppe: 54 / 16 Teilnehmer; 2. Auswertungsgruppe: 31 / 7 Teilnehmer und 3. Auswertungsgruppe: 23 / 9 Teilnehmer
- **Antwort 3:** 2 der 3 Aufgaben müssen ausgeführt sein, damit Aufgabe 5 ausgeführt werden kann.
  Ergebnis: 1. Auswertungsgruppe: 0 / 2 Teilnehmer; 2. Auswertungsgruppe: 0 / 1 Teilnehmer und 3. Auswertungsgruppe: 2 / 1 Teilnehmer
- **Antwort 4:** Abhängig davon, welche Aufgaben von der Oder-Verzweigung (OR) aktiviert wurden, müssen die Aufgaben 2, 3 und/oder 4 ausgeführt sein, damit Aufgabe 5 ausgeführt werden kann.
  Ergebnis: 1. Auswertungsgruppe: 29 / 84 Teilnehmer; 2. Auswertungsgruppe: 21 / 53 Teilnehmer und 3. Auswertungsgruppe: 8 / 31 Teilnehmer
- **Antwort 5:** Die Kontrollflusssemantik der Oder-Zusammenführung (OR) ist nicht eindeutig.
  Ergebnis: 1. Auswertungsgruppe: 24 / 1 Teilnehmer; 2. Auswertungsgruppe: 14 / 1 Teilnehmer und 3. Auswertungsgruppe: 10 / 0 Teilnehmer

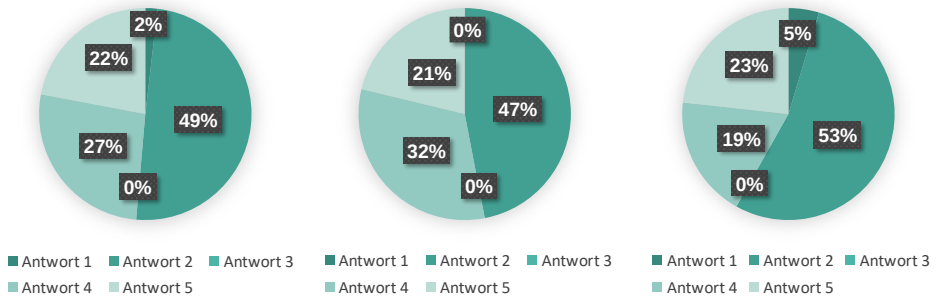


Abbildung 96: Evaluation - Erstes Geschäftsprozessmodell ohne Annotation von Kontrollflussmustern (Links: alle Teilnehmer, Mitte: Teilnehmer mit fortgeschrittenen Kenntnissen im Geschäftsprozessmanagement, Rechts: Teilnehmer mit Grundkenntnissen im Geschäftsprozessmanagement)

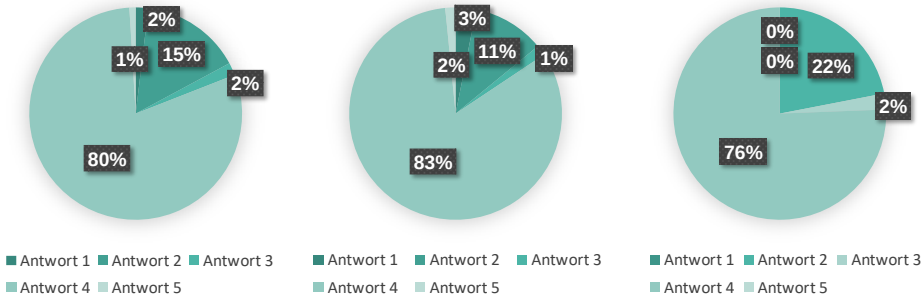


Abbildung 97: Evaluation - Erstes Geschäftsprozessmodell mit Annotation von Kontrollflussmustern (Links: alle Teilnehmer, Mitte: Teilnehmer mit fortgeschrittenen Kenntnissen im Geschäftsprozessmanagement, Rechts: Teilnehmer mit Grundkenntnissen im Geschäftsprozessmanagement)

Aus Abbildung 96 geht hervor, dass fast die Hälfte der Teilnehmer die zweite Antwort ausgewählt hat, so dass nach der ersten ausgeführten Aufgabe (Aufgabe 2, 3 oder 4), Aufgabe 5 ausgeführt werden kann. Die andere Hälfte hat fast zu gleichen Teilen angegeben, dass die Kontrollflusssemantik der Oder-Zusammenführung an dieser Stelle nicht eindeutig sei (22%) und die Ausführung von Ausgabe 5 von der Oder-Verzweigung abhängig ist (27%). Die anderen Antworten können aufgrund der geringen Anzahl mit 2% und 0% bei der Interpretation der Ergebnisse vernachlässigt werden. Bei der Differenzierung nach dem Kenntnisstand im Geschäftsprozessmanagement wird deutlich, dass mehr Teilnehmer mit fortgeschrittenen Kenntnissen eine Verbindung zur Oder-Verzweigung herstellen (32%) als die Teilnehmer, die nur Grundkenntnisse besitzen (19%). Analog wird diese Differenz auch bei Antwort 2 deutlich, das heißt, dass Aufgabe 5 ausgeführt werden kann, sobald eine der Aufgaben 2, 3 oder 4 ausgeführt worden ist. Aus der Ergebnisverteilung in Abbildung 96 wird deutlich, warum die Regel 5 der sieben Prozessmodellierungsrichtlinien (7PMG) [MeRv10] bei der Geschäftsprozessmodellierung berücksichtigt werden sollte. Signifikant anders sieht die Ergebnisverteilung bei dem Geschäftsprozessmodell mit Kontrollflussmustern (vgl. Abbildung 90a) aus. Das zugewiesene Kontrollflussmuster beschreibt die Ablaufreihenfolge der Elemente des Geschäftsprozessmodells so, dass die Ausführung von Ausgabe 5 von der Oder-Verzweigung abhängig ist. Die Interpretation der Ablaufreihenfolge der Elemente nach der Intention des Modellierers konnte durch die Annotation des Kontrollflussmusters von 27% auf 80% gesteigert werden (vgl. Abbildung 97). Nur noch 15% der Teilnehmer sind der Meinung, dass nach der ersten ausgeführten Aufgabe, die Aufgabe 5 ausgeführt werden kann. Bei der Differenzierung nach dem Kenntnisstand im Geschäftsprozessmanagement wird deutlich, dass mehr Teilnehmer mit fortgeschrittenen Kenntnissen durch die Annotation beeinflusst werden konnten als die Teilnehmer, die nur Grundkenntnisse im Geschäftsprozessmanagement besitzen. Es sind trotzdem noch 11% der

Teilnehmer mit fortgeschrittenen Kenntnissen der Meinung, dass nach der ersten ausgeführten Aufgabe, Aufgabe 5 unmittelbar ausgeführt werden kann. Gleichwohl vertreten aber auch 22% der Teilnehmer mit Grundkenntnissen im Geschäftsprozessmanagement diese Meinung. Das kann unter anderem auf die fehlenden Vorkenntnisse in Bezug auf die Bekanntheit der Kontrollflussmuster zurückgeführt werden. Die Interpretation der Ablaufreihenfolge der Elemente des Geschäftsprozessmodells nach der Intention des Modellierers konnte durch die Annotation von Kontrollflussmustern um 53 Prozentpunkte verbessert werden. Dementsprechend wurde die Verständlichkeit der Oder-Zusammenführung durch die Annotation mit einem Kontrollflussmuster der Workflow Pattern Initiative signifikant verbessert.

Das zweite zu interpretierende Geschäftsprozessmodell wird in Abbildung 27b illustriert, deren Ergebnisverteilung zur Interpretation der Ablaufreihenfolge der Elemente in Abbildung 98 visualisiert ist. Wie auch beim ersten Geschäftsprozessmodell wurde das Geschäftsprozessmodell aus Abbildung 27b mit Kontrollflussmustern annotiert (vgl. Abbildung 90b), woraufhin die gleichen Antwortmöglichkeiten bewertet werden sollten. Die Ergebnisverteilung des Geschäftsprozessmodells mit Kontrollflussmustern wird in Abbildung 99 visualisiert. Im Folgenden werden zunächst die Antwortmöglichkeiten mit deren Ergebnisverteilung in absoluten Zahlen, in Abhängigkeit der Auswertungsgruppen sowie mit bzw. ohne Annotation der Kontrollflussmuster, dargestellt:

- **Antwort 1:** Die Aufgaben 2, 3 und 4 müssen ausgeführt sein, damit Aufgabe 5 ausgeführt werden kann.
  Ergebnis: 1. Auswertungsgruppe: 3 / 6 Teilnehmer; 2. Auswertungsgruppe: 2 / 3 Teilnehmer und 3. Auswertungsgruppe: 1 / 3 Teilnehmer
- **Antwort 2:** Nach der ersten ausgeführten Aufgabe (Aufgabe 2, 3 oder 4) kann Aufgabe 5 ausgeführt werden.
  Ergebnis: 1. Auswertungsgruppe: 79 / 103 Teilnehmer; 2. Auswertungsgruppe: 42 / 62 Teilnehmer und 3. Auswertungsgruppe: 37 / 41 Teilnehmer
- **Antwort 3:** 2 der 3 Aufgaben müssen ausgeführt sein, damit Aufgabe 5 ausgeführt werden kann.
  Ergebnis: 1. Auswertungsgruppe: 3 / 2 Teilnehmer; 2. Auswertungsgruppe: 2 / 1 Teilnehmer und 3. Auswertungsgruppe: 1 / 1 Teilnehmer
- **Antwort 4:** Die Kontrollflusssemantik der Oder-Zusammenführung (OR) ist nicht eindeutig.
  Ergebnis: 1. Auswertungsgruppe: 31 / 3 Teilnehmer; 2. Auswertungsgruppe: 23 / 2 Teilnehmer und 3. Auswertungsgruppe: 8 / 1 Teilnehmer

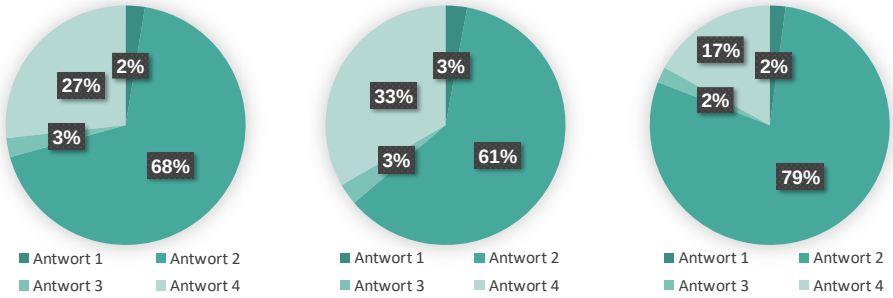


Abbildung 98: Evaluation - Zweites Geschäftsprozessmodell ohne Annotation von Kontrollflussmustern (Links: alle Teilnehmer, Mitte: Teilnehmer mit fortgeschrittenen Kenntnissen im Geschäftsprozessmanagement, Rechts: Teilnehmer mit Grundkenntnissen im Geschäftsprozessmanagement)

Im Vergleich zum ersten Geschäftsprozessmodell wird die Ablaufreihenfolge der Elemente im Geschäftsprozessmodell mit und ohne Annotation der Kontrollflussmuster überwiegend gleich interpretiert. Etwas mehr als zwei Drittel der Teilnehmer (68%) interpretieren die Oder-Zusammenführung so, dass nach der ersten ausgeführten Aufgabe die Ausführung der Aufgabe 5 möglich ist. Eine weitere häufig genannte Antwort bei den Teilnehmern ist mit 27%, dass die Kontrollflusssemantik der Oder-Zusammenführung an dieser Stelle nicht eindeutig sei. Die anderen beiden Antworten können mit 2% und 3% aufgrund der geringen Nennung vernachlässigt werden. Bei der detaillierten Betrachtung der Ergebnisverteilung, unter Berücksichtigung der Kenntnisse im Geschäftsprozessmanagement, wird deutlich, dass fast dreimal so viele Teilnehmer mit fortgeschrittenen Kenntnissen die Ablaufreihenfolge der Elemente des Geschäftsprozessmodells als nicht eindeutig bezeichnen. Prozentual ist der Unterschied jedoch nur doppelt so groß, da die Auswertungsgruppe mit fortgeschrittenen Kenntnissen 24 Teilnehmer mehr umfasst als die Auswertungsgruppe mit Grundkenntnissen im Geschäftsprozessmanagement. In diesem Zusammenhang hat sich auch die Anzahl der Antworten reduziert bzw. erhöht, so dass nach der ersten ausgeführten Aufgabe, Aufgabe 5 ausgeführt werden kann. 79% der Teilnehmer mit Grundkenntnissen im Geschäftsprozessmanagement haben angegeben, dass die Aufgabe 5 nach der ersten ausgeführten Aufgabe ausgeführt werden kann. Ebenso haben das auch 61% der Teilnehmer mit fortgeschrittenen Kenntnissen angegeben. Nach der Intention des Modellierers wurde die Kontrollflusssemantik der Oder-Zusammenführung in Abbildung 90b mit dem Kontrollflussmuster der einfachen Zusammenführung annotiert, so dass nach der ersten ausgeführten Aufgabe 2, 3 oder 4 unmittelbar Aufgabe 5 ausgeführt werden kann. Durch die Verwendung des Kontrollflussmusters konnte die Interpretation der Ablaufreihenfolge der Elemente des Geschäftsprozessmodells nach der Intention des Modellierers von 68% auf 90% gesteigert werden. Mit der Annotation der Kontrollflussmuster sind keine

nennenswerten Abweichungen mehr bei der Abgabe der Antworten mit 91% und 89%, abhängig vom Kenntnisstand im Geschäftsprozessmanagement, zu verzeichnen. Auch an diesem Beispiel wird insgesamt deutlich, dass durch die Einführung der Zusammenführungsbedingung in Form von Kontrollflussmustern die ungewollten Mehrdeutigkeiten der Oder-Zusammenführung signifikant reduziert werden konnten und somit die Verständlichkeit verbessert wurde, obwohl bereits etwa zwei Drittel der Teilnehmer die Ablaufreihenfolge der Elemente des Geschäftsprozessmodells ohne Kontrollflussmuster nach der Intention des Modellierers verstanden haben.

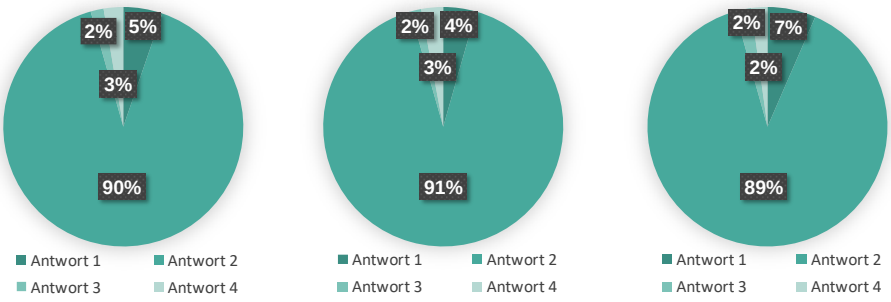


Abbildung 99: Evaluation - Zweites Geschäftsprozessmodell mit Annotation von Kontrollflussmustern (Links: alle Teilnehmer, Mitte: Teilnehmer mit fortgeschrittenen Kenntnissen im Geschäftsprozessmanagement, Rechts: Teilnehmer mit Grundkenntnissen im Geschäftsprozessmanagement)

Das dritte zu interpretierende Geschäftsprozessmodell ist in Abbildung 27c illustriert, deren Ergebnisverteilung zur Interpretation der Ablaufreihenfolge der Elemente im Geschäftsprozessmodell in Abbildung 100 visualisiert wird. Das gleiche Geschäftsprozessmodell wurde auch mit Kontrollflussmustern annotiert, wie dies auch in Abbildung 90c dargestellt ist. Die Ablaufreihenfolge der Elemente des Geschäftsprozessmodells sollte mit den gleichen Antwortmöglichkeiten beschrieben werden. Die Ergebnisverteilung ist in Abbildung 101 visualisiert. Im Folgenden werden die Antwortmöglichkeiten mit deren Ergebnisverteilung in absoluten Zahlen dargelegt, in Abhängigkeit der Auswertungsgruppen sowie mit bzw. ohne Annotation der Kontrollflussmuster:

- **Antwort 1:** Die Aufgaben 5 und 6 müssen ausgeführt sein, damit Aufgabe 7 ausgeführt werden kann.
  Ergebnis: 1. Auswertungsgruppe: 7 / 85 Teilnehmer; 2. Auswertungsgruppe: 3 / 51 Teilnehmer und 3. Auswertungsgruppe: 4 / 34 Teilnehmer
- **Antwort 2:** Nach der ersten ausgeführten Aufgabe (Aufgabe 5 oder 6) kann Aufgabe 7 ausgeführt werden.

Ergebnis: 1. Auswertungsgruppe: 77 / 25 Teilnehmer; 2. Auswertungsgruppe: 44 / 14 Teilnehmer und 3. Auswertungsgruppe: 33 / 11 Teilnehmer

– **Antwort 3:** Die Kontrollflusssemantik der Oder-Zusammenführung (OR) ist nicht eindeutig.
  Ergebnis: 1. Auswertungsgruppe: 29 / 2 Teilnehmer; 2. Auswertungsgruppe: 20 / 1 Teilnehmer und 3. Auswertungsgruppe: 9 / 1 Teilnehmer

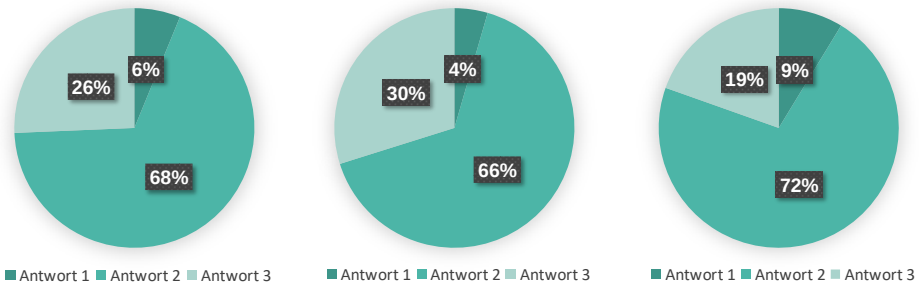


Abbildung 100: Evaluation - Drittes Geschäftsprozessmodell ohne Annotation von Kontrollflussmustern (Links: alle Teilnehmer, Mitte: Teilnehmer mit fortgeschrittenen Kenntnissen im Geschäftsprozessmanagement, Rechts: Teilnehmer mit Grundkenntnissen im Geschäftsprozessmanagement)

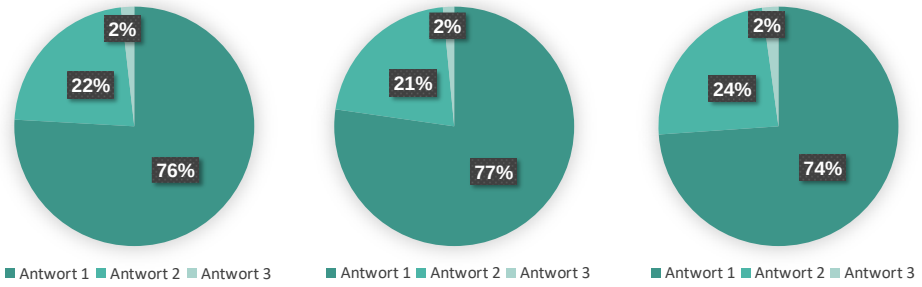


Abbildung 101: Evaluation - Drittes Geschäftsprozessmodell mit Annotation von Kontrollflussmustern (Links: alle Teilnehmer, Mitte: Teilnehmer mit fortgeschrittenen Kenntnissen im Geschäftsprozessmanagement, Rechts: Teilnehmer mit Grundkenntnissen im Geschäftsprozessmanagement)

Durch die Ergebnisauswertung in Abbildung 100 wird mit der Interpretation der Ablaufreihenfolge der Elemente des dritten Geschäftsprozessmodells (vgl. Abbildung 27c) deutlich, dass etwa zwei Drittel der Teilnehmer angegeben haben, dass nach der ersten ausgeführten Aufgabe (Aufgabe 5 oder 6), Aufgabe 7 ausgeführt werden kann. 26% der Teilnehmer

sind der Meinung, dass die Ablaufreihenfolge der Elemente des Geschäftsprozessmodells nicht eindeutig ist sowie 6% der Teilnehmer, dass für die Ausführung von Aufgabe 7, die Aufgaben 5 und 6 ausgeführt sein müssen. Wie auch beim ersten und zweiten Geschäftsprozessmodell ohne Kontrollflussmuster ist eine Abweichung aufgrund des Kenntnisstandes im Geschäftsprozessmanagement ersichtlich. Dementsprechend sind 30% der Teilnehmer mit fortgeschrittenen Kenntnissen der Meinung, dass die Ablaufreihenfolge der Elemente des Geschäftsprozessmodells nicht eindeutig ist, wohingegen nur 19% der Teilnehmer mit Grundkenntnissen im Geschäftsprozessmanagement dieser Meinung sind. Die Aufgaben 5 und 6 müssen ausgeführt sein, bevor Aufgabe 7 ausgeführt werden kann, haben 4% der Teilnehmer mit fortgeschrittenen Kenntnissen angegeben. Dieser Meinung sind auch 9% der Teilnehmer mit Grundkenntnissen im Geschäftsprozessmanagement. Analog gilt dies mit 66% bzw. 72% für die dritte Antwortmöglichkeit. Die Oder-Zusammenführung wurde in Abbildung 91c mit dem Kontrollflussmuster Synchronisation annotiert, so dass die Aufgaben 5 und 6 ausgeführt sein müssen, damit Aufgabe 7 ausgeführt werden kann. Anstatt der Annotation der Oder-Zusammenführung mit dem Kontrollflussmuster Synchronisation (wcp_3) hätte im Geschäftsprozessmodell auch eine Und-Zusammenführung verwenden werden können. Zum einen sollte identifiziert werden, inwiefern die Annotation des Geschäftsprozessmodells mit Kontrollflussmustern einen Einfluss auf die Interpretation der Ablaufreihenfolge der Elemente von Geschäftsprozessmodellen haben kann. Zum anderen ist die Synchronisation aber auch ein zulässiger Fall einer Oder-Zusammenführung. Aus der Ergebnisverteilung in Abbildung 101 geht hervor, dass die Annotation des Geschäftsprozessmodells mit Kontrollflussmustern einen großen Einfluss auf die Interpretation der Ablaufreihenfolge der Elemente von Geschäftsprozessmodellen hat. Die Interpretation der Ablaufreihenfolge der Elemente des Geschäftsprozessmodells konnte nach der Intention des Modellierers durch die Annotation mit Kontrollflussmustern von 6% auf 76% verbessert werden. Es sind nur noch 22% der Teilnehmer der Meinung, dass nach der ersten ausgeführten Aufgabe die Aufgabe 7 ausgeführt werden kann. Ein möglicher Grund dafür, warum die Prozentzahl trotz der Annotation des Geschäftsprozessmodells mit Kontrollflussmustern so hoch ist, ist, dass mit einer Oder-Zusammenführung normalerweise keine Und-Zusammenführung modelliert wird. Die Anzahl der Teilnehmer, die der Meinung sind, dass die Kontrollflusssemantik der Oder-Zusammenführung nicht eindeutig ist, konnte von 26% auf 2% reduziert werden. Die Auswertung besitzt aufgrund des Kenntnisstandes der Teilnehmer im Geschäftsprozessmanagement keine nennenswerten Abweichungen mehr, ebenso wie auch beim nicht annotierten Geschäftsprozessmodell. Die fehlenden Vorkenntnisse in Bezug auf die Kontrollflussmuster haben beim Geschäftsprozessmodell mit Kontrollflussmustern keinen Einfluss auf die ausgewählten Antworten der Teilnehmer. Wahrscheinlich wurden alle notwendigen Informationen durch das einführende Informationsvideo ausreichend erklärt, mit Ausnahme einer Oder-

Zusammenführung, die von einer Oder-Verzweigung abhängig ist, wie aus dem ersten Geschäftsprozessmodell hervorgeht. Insgesamt wird auch an diesem Beispiel deutlich, dass durch die Annotation des Geschäftsprozessmodells mit Kontrollflussmustern die ungewollten Mehrdeutigkeiten der Oder-Zusammenführung signifikant reduziert werden konnten und somit die Verständlichkeit der Kontrollflusssemantik verbessert wurde. Darüber hinaus konnte im Rahmen des dritten Geschäftsprozessmodells festgestellt werden, dass durch die Annotation der Geschäftsprozessmodelle mit Kontrollflussmustern die Interpretation der Ablaufreihenfolge der Elemente stark beeinflusst werden kann.

Für alle drei Beispielgeschäftsprozessmodelle konnte gezeigt werden, dass durch die Annotation der Oder-Zusammenführung, mit den Kontrollflussmustern der Workflow Pattern Initiative, die ungewollten Mehrdeutigkeiten der Kontrollflusssemantik reduziert werden können und die Verständlichkeit der Kontrollflusssemantik verbessert werden kann. Insbesondere gilt das für die Teilnehmer, die nur Grundkenntnisse im Geschäftsprozessmanagement besitzen. Mit der Annotation sind keine signifikanten Abweichungen mehr zwischen den Teilnehmern mit fortgeschrittenen Kenntnissen und Grundkenntnissen im Geschäftsprozessmanagement erkennbar. Somit kann die erste aufgestellte Hypothese durch diese Untersuchung gestützt werden. Aus den Erkenntnissen kann geschlossen werden, dass die entwickelte Methode aus Kapitel 6 zur Verständlichkeit beitragen kann. Jedoch bestanden die Geschäftsprozessmodelle zum einen nur aus wenigen Elementen und zum anderen wurde ausschließlich die Verständlichkeit der Oder-Zusammenführung betrachtet. Darüber hinaus ist die Frage zu beantworten, wie die Kontrollflusssemantik einer Oder-Zusammenführung in Geschäftsprozessmodellen prinzipiell interpretiert wird. Die Antwort nach der ersten ausgeführten Aufgabe kann die Aufgabe nach einer Oder-Zusammenführung ausgeführt werden, ist gehäuft aufgetreten. In detaillierten Untersuchungen sollten diese Aspekte unter anderem auch mit realitätsnäheren und komplexeren Geschäftsprozessmodellen sowie auch anderen Kontrollflussmustern evaluiert werden, um zu überprüfen, ob die entwickelte Methode auch hier zur Verständlichkeit beitragen kann.

Im nächsten Schritt werden die Benutzbarkeit, Erlernbarkeit, Erwartungshaltung, Nützlichkeit und Verständlichkeit betrachtet. Für differenzierte Auswertungen wurden die Kenntnisse im Geschäftsprozessmanagement (vgl. Abbildung 93 Rechts) berücksichtigt. Darüber hinaus wurde ermittelt, welche Modellierungssprachen zur Modellierung von Geschäftsprozessen eingesetzt werden. Abhängig vom Kenntnisstand im Geschäftsprozessmanagement wird das Ergebnis in Abbildung 102 visualisiert. Bei der Frage nach den Modellierungssprachen war eine Mehrfachnennung möglich. Aus der Ergebnisverteilung geht hervor, dass die Modellierungssprache BPMN am häufigsten verwendet wird, gefolgt von den Petri-Netzen (PN), UML-Aktivitätsdiagrammen (UML-AD) und der Ereignisgesteuerten Prozesskette (EPK). Die Modellierungssprache YAWL wird nur von Teilnehmern eingesetzt,

die sehr gute Kenntnisse im Geschäftsprozessmanagement besitzen. Mit abnehmendem Kenntnisstand nimmt auch die Anzahl der Antwort im Fragebogen „keine Angabe“ zu, woraus man schlussfolgern kann, dass ein Großteil dieser Teilnehmer wahrscheinlich keine Modellierungssprache einsetzt. Aus den Ergebnissen geht darüber hinaus auch hervor, dass Teilnehmer im Geschäftsprozessmanagement trotz geringer Kenntnisse, die Modellierungssprachen BPMN, EPK, UML-AD und PN dennoch einsetzen.

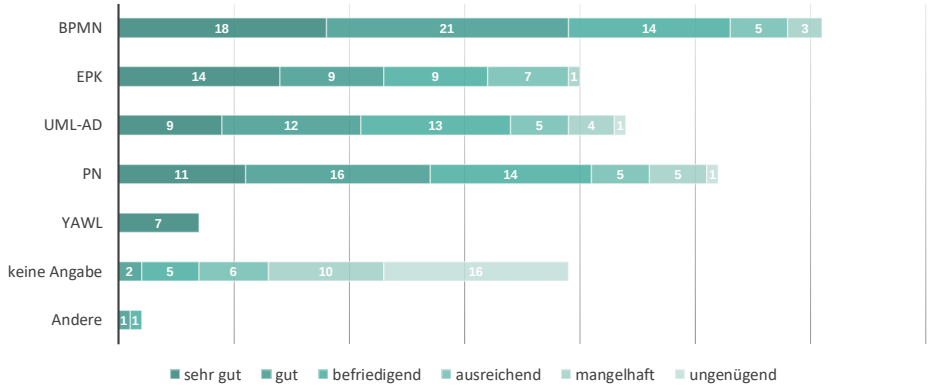


Abbildung 102: Evaluation - Anwendung der Geschäftsprozessmodellierungssprachen der Teilnehmer in Abhängigkeit von den Kenntnissen im Geschäftsprozessmanagement

Im Folgenden werden die Hypothesen zwei bis sieben betrachtet. In diesem Zusammenhang wurde die einfache Benutzbarkeit, leichte Erlernbarkeit, einfache Beschreibung der Kontrollflusssemantik (bzw. Erwartungshaltung), Nützlichkeit und leichte Verständlichkeit messbar gemacht, indem Aussagen formuliert wurden, die von den Teilnehmern mittels Schulnoten bewertet werden sollten. Die Ergebnisse, der zu bewertenden Aussagen vor der Interpretation der Ablaufreihenfolge der Elemente der Geschäftsprozessmodelle, werden in Abbildung 103 illustriert. In Abbildung 104 sind die Ergebnisse der zu bewertenden Aussagen nach der Interpretation der Ablaufreihenfolge der Elemente dargestellt. In Abbildung 105 und Abbildung 106 werden nur die Teilnehmer mit fortgeschrittenen Kenntnissen im Geschäftsprozessmanagement betrachtet sowie in Abbildung 107 und Abbildung 108 nur die Teilnehmer mit Grundkenntnissen im Geschäftsprozessmanagement. Im Folgenden werden die einzelnen Hypothesen betrachtet:

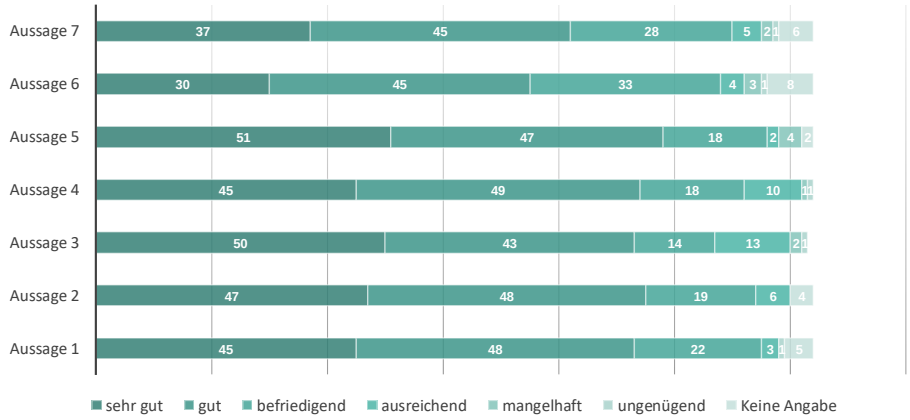


Abbildung 103: Evaluation - Bewertung von Aussagen von den Teilnehmern vor der Interpretation der Ablaufreihenfolge der Elemente der Geschäftsprozessmodelle

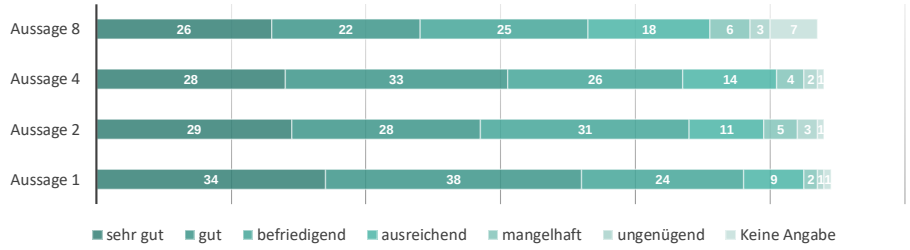


Abbildung 104: Evaluation - Bewertung von Aussagen von den Teilnehmern nach der Interpretation der Ablaufreihenfolge der Elemente der Geschäftsprozessmodelle

– *Hypothese 2* (Erlernbarkeit): Die Methode zur Beschreibung der Ablaufreihenfolge der Elemente der Geschäftsprozessmodelle mit den Kontrollflussmustern der Workflow Pattern Initiative ist einfach zu erlernen.

  Die zu bewertende Aussage für diese Hypothese war, *die Methode ist gut ohne fremde Hilfe erlernbar* (Aussage 5). Durchschnittlich wurde unter allen Teilnehmern die Methode als gut ohne fremde Hilfe erlernbar bewertet (Durchschnittliche Note: 1,86 mit der Standardabweichung: 0,04). Dabei haben 79,03% der Teilnehmer die Aussage mit gut bzw. sehr gut beurteilt. Analog zu den anderen Aussagen wurde auch diese Aussage von den Teilnehmern mit fortgeschrittenen Kenntnissen (Durchschnittliche Note: 1,72 mit der Standardabweichung: 0,08)

tendenziell besser bewertet als von den Teilnehmern mit Grundkenntnissen im Geschäftsprozessmanagement (Durchschnittliche Note: 2,08 mit der Standardabweichung: 0,11).

Die Aussage 4, *auch bei seltenem Gebrauch ist es kein Problem die Methode wieder anzuwenden,* wurde von 75,8% der Teilnehmer mit gut bis sehr gut bewertet. Weniger als 2% haben die Aussage mit mangelhaft oder ungenügend bewertet. Durchschnittlich wurde die Aussage mit der Note 2,0 und einer Standardabweichung von 0,04 bewertet. In Bezug auf die Betrachtung nach dem Kenntnisstand im Geschäftsprozessmanagement wird deutlich, dass 80% der Teilnehmer mit fortgeschrittenen Kenntnissen die Aussage mit gut bzw. sehr gut bewertet haben, dahingegen aber nur 69,38% der Teilnehmer mit Grundkenntnissen. Durchschnittlich wurde die Aussage mit der Note 1,93 und einer Standardabweichung von 0,07 (fortgeschrittene Kenntnisse im Geschäftsprozessmanagement) bzw. mit der Note 2,1 und einer Standardabweichung von 0,11 (Grundkenntnisse im Geschäftsprozessmanagement) bewertet. Diese Aussage wurde ebenfalls nach der Interpretation der Ablaufreihenfolge der Elemente der Geschäftsprozessmodelle erneut von den Teilnehmern bewertet. Hierbei sowie auch bei den anderen Aussagen wird deutlich, dass sich die durchschnittlich vergebene Note von 2,0 auf 2,42 verschlechtert hat. Die Standardabweichung verändert sich erst ab der dritten Nachkommastelle. Diese Verschlechterung kann hauptsächlich auf die schlechtere Bewertung der Teilnehmer mit Grundkenntnissen im Geschäftsprozessmanagement zurückgeführt werden. Die Teilnehmer haben die Aussage durchschnittlich mit der Note 2,71 bewertet und einer Standardabweichung von 0,1. Die Teilnehmer mit fortgeschrittenen Kenntnissen im Geschäftsprozessmanagement bewerten die Aussage durchschnittlich mit der Note 2,24 und einer Standardabweichung von 0,07. Besonders interessant an den Ergebnissen ist, dass die Aussage nach der Interpretation der Kontrollflusssemantik um eine halbe Note schlechter bewertet wurde sowie eine größere Standardabweichung vorliegt als vor der Interpretation der Kontrollflusssemantik. Darüber hinaus haben die Teilnehmer mit nur Grundkenntnissen im Geschäftsprozessmanagement eine deutlich schlechtere Bewertung abgegeben, so dass insbesondere diese Aspekte in einer weiterführenden Untersuchung betrachtet werden sollten.

– *Hypothese 3* (Benutzbarkeit): Die Benutzung der Methode zur Beschreibung der Ablaufreihenfolge der Elemente der Geschäftsprozessmodelle mit den Kontrollflussmustern der Workflow Pattern Initiative ist einfach.

Die zu bewertende Aussage für die Hypothese war: *die Methode lässt sich einfach benutzen* (Aussage 2). Die Aussage wurde von 76,61% der Teilnehmer mit gut bis sehr gut bewertet, so dass die Methode als einfach benutzbar eingestuft werden kann. Kein Teilnehmer hat die Aussage mit mangelhaft oder ungenügend beantwortet, so dass die durchschnittliche Note unter allen Teilnehmern 1,89 ist, mit einer Standardabweichung von 0,04. Die Teilnehmer mit fortgeschrittenen Kenntnissen haben die Methode als einfacher benutzbar bewertet (Durchschnittliche Note: 1,77 mit einer Standardabweichung: 0,08) als die Teilnehmer mit Grundkenntnissen im Geschäftsprozessmanagement (Durchschnittliche Note: 2,02 mit einer Standardabweichung: 0,12). Nach der Interpretation der Ablaufreihenfolge der Elemente der Geschäftsprozessmodelle wurde diese Aussage von den Teilnehmern erneut bewertet. Dabei konnte festgestellt werden, dass die einfache Benutzung durchschnittlich nur noch mit der Note 2,47 und einer Standardabweichung von 0,04 bewertet wurde. Somit wurde die Benutzbarkeit um etwas mehr als eine halbe Note schlechter bewertet als vor der Interpretation der Ablaufreihenfolge der Elemente der Geschäftsprozessmodelle. Die Verschlechterung ist überwiegend auf die Teilnehmer mit Grundkenntnissen im Geschäftsprozessmanagement zurückzuführen. Dennoch kann die Methode zusammenfassend als gut benutzbar eingestuft werden. Wie auch bei der vorherigen Hypothese sollte diese Veränderung der Bewertung vor und nach der Interpretation der Geschäftsprozessmodelle detaillierter betrachtet werden, um entsprechende Rückschlüsse treffen zu können.

– *Hypothese 4* (Verständlichkeit): Die Methode zur Beschreibung der Ablaufreihenfolge der Elemente der Geschäftsprozessmodelle mit den Kontrollflussmustern der Workflow Pattern Initiative ist verständlich.

  Die Teilnehmer sollten nicht nur die Kontrollflusssemantik von Geschäftsprozessmodellen interpretieren, sondern darüber hinaus auch bewerten, ob sie die Methode verständlich finden. Dafür sollte die Aussage *die Methode ist verständlich* (Aussage 3) bewertet werden. Wie auch bei den vorherigen Aussagen haben 75,6% der Teilnehmer die Aussage mit gut bzw. sehr gut bewertet sowie 2,4% der Teilnehmer mit mangelhaft bzw. ungenügend. Durchschnittlich wurde die Aussage mit der Note 2,0 und einer Standardabweichung von 0,04 bewertet. In Anlehnung an die anderen Aussagen war auch bei dieser Aussage die durchschnittlich vergebene Note der Teilnehmer mit fortgeschrittenen Kenntnissen im Geschäftsprozessmanagement (Durchschnittliche Note: 1,9 mit der Standardabweichung: 0,07) besser als die durchschnittlich vergebene Note der Teilnehmer mit Grundkenntnissen im Geschäftsprozessmanagement (Durchschnittliche Note:

2,08 mit einer Standardabweichung: 0,11). Die Methode kann aufgrund der zugrundeliegenden Ergebnisse als verständlich bezeichnet werden.

– *Hypothese 5* (Verständlichkeit): Die Methode zur Beschreibung der Ablaufreihenfolge der Elemente der Geschäftsprozessmodelle mit den Kontrollflussmustern der Workflow Pattern Initiative ist für Anfänger und Experten gleichermaßen verständlich.

  Die sechste zu bewertende Aussage betrachtet ebenfalls die Verständlichkeit durch die Aussage, *die Methode ist für Anfänger als auch für Experten gleichermaßen geeignet*. Diese Aussage wurde von allen Teilnehmern mit der durchschnittlichen Note von 2,2 und einer Standardabweichung von 0,04 bewertet. Aufgrund der Ergebnisse der vorherigen Aussagen ist dieses Ergebnis plausibel, da die Teilnehmer mit Grundkenntnissen die Aussagen tendenziell schlechter bewertet haben als die Teilnehmer mit fortgeschrittenen Kenntnissen im Geschäftsprozessmanagement. Dennoch haben 50% der Teilnehmer mit Grundkenntnissen im Geschäftsprozessmanagement die Aussage mit gut bzw. sehr gut bewertet. Für diese Hypothese wäre eine detaillierte Betrachtung bezüglich der Frage, inwiefern Teilnehmer mit Grundkenntnissen der Meinung sind, dass sich die Methode für Teilnehmer mit Grundkenntnissen und für Teilnehmer mit fortgeschrittenen Kenntnissen im Geschäftsprozessmanagement eignet. Analog gilt das für die Teilnehmer mit fortgeschrittenen Kenntnissen.

– *Hypothese 6* (Erwartungshaltung): Die Methode zur Beschreibung der Ablaufreihenfolge der Elemente der Geschäftsprozessmodelle ermöglicht eine einfache Beschreibung der Kontrollflusssemantik.

  In einer weiteren Hypothese sollte die Erwartungshaltung evaluiert werden, ob die Methode die Anforderungen an eine einfache Beschreibung der Kontrollflusssemantik erfüllt. Die zu bewertende Aussage war *die Methode erfüllt meine Erwartungen an eine einfache Beschreibung der Kontrollflusssemantik*. Durchschnittlich wurde diese Aussage mit der Note 2,09 und einer Standardabweichung von 0,04 bewertet, wobei 70% der Teilnehmer mit fortgeschrittenen Kenntnissen im Geschäftsprozessmanagement die Aussage mit gut bzw. sehr gut bewertet haben, dahingegen jedoch nur 59,18% der Teilnehmer mit Grundkenntnissen im Geschäftsprozessmanagement. Durchschnittlich wurde die Aussage von den Teilnehmern mit fortgeschrittenen Kenntnissen mit der Note 1,98 und einer Standardabweichung von 0,07 bewertet sowie mit der Note 2,26 und einer Standardabweichung von 0,11 von den Teilnehmern mit Grundkenntnissen im

Geschäftsprozessmanagement. Zusammenfassend kann die Erwartungshaltung als gut bezeichnet werden.

– *Hypothese 7* (Nützlichkeit): Die Annotation der Elemente der Geschäftsprozessmodelle mit den Kontrollflussmustern ist nützlich, um die Kontrollflusssemantik der Geschäftsprozessmodelle besser zu verstehen.

  Die Aussage, *die Methode ist nützlich, um die Kontrollflusssemantik besser zu verstehen* (Aussage 1), haben insgesamt 75% der Teilnehmer mit gut bis sehr gut bewertet. Durchschnittlich wird die Nützlichkeit mit der Note 1,89 und einer Standardabweichung von 0,04 bewertet. Unter Berücksichtigung des Kenntnisstandes im Geschäftsprozessmanagement wird deutlich, dass die Teilnehmer mit fortgeschrittenen Kenntnissen im Geschäftsprozessmanagement die Methode nützlicher einstufen (Durchschnittliche Note: 1,77 mit einer Standardabweichung: 0,07) als die Teilnehmer, die nur Grundkenntnisse im Geschäftsprozessmanagement (Durchschnittliche Note: 2,09 mit einer Standardabweichung: 0,12) besitzen. Die Nützlichkeit wird überwiegend als gut bis sehr gut bewertet und sollte auch nach der Interpretation der Ablaufreihenfolge der Elemente der Geschäftsprozessmodelle von den Teilnehmern erneut bewertet werden. Aus der Ergebnisverteilung wird deutlich, dass sich der Wert der durchschnittlichen Nützlichkeit verringert hat. Dementsprechend wurde die Nützlichkeit nur noch mit einer durchschnittlichen Note von 2,16 und einer Standardabweichung von 0,04 bewertet, anstatt der durchschnittlichen Note von 1,89 und einer Standardabweichung von 0,04. Die Nützlichkeit wurde sowohl von den Teilnehmern mit fortgeschrittenen Kenntnissen als auch von den Teilnehmern mit Grundkenntnissen im Geschäftsprozessmanagement gleichermaßen um eine Drittel-Note schlechter bewertet. Dennoch wurde die Nützlichkeit noch als gut eingestuft.

  Die Aussage 8, *ich würde die Methode gerne nutzen, da ich die Nützlichkeit sehe*, wurde im Anschluss an die Interpretation der Ablaufreihenfolge der Elemente der Geschäftsprozessmodelle befragt. Unter allen Teilnehmern wurde die Aussage durchschnittlich mit der Note 2,65 und einer Standardabweichung von 0,04 bewertet. Dabei ist im Rahmen der Betrachtung zu erkennen, dass nur 54% der Teilnehmer mit fortgeschrittenen Kenntnissen diese Aussage mit gut bis sehr gut bewertet haben und nur 30% der Teilnehmer mit Grundkenntnissen im Geschäftsprozessmanagement. Die Teilnehmer mit Grundkenntnissen haben die Aussage mit der durchschnittlichen Note von 3,1 und einer Standardabweichung von 0,11 bewertet, was als mittelmäßig zu bewerten ist. Aus den beiden Aussagen

kann abgeleitet werden, dass die Methode zwar nicht gerne angewendet wird, aber dennoch eine gewisse Nützlichkeit unterstellt werden kann.

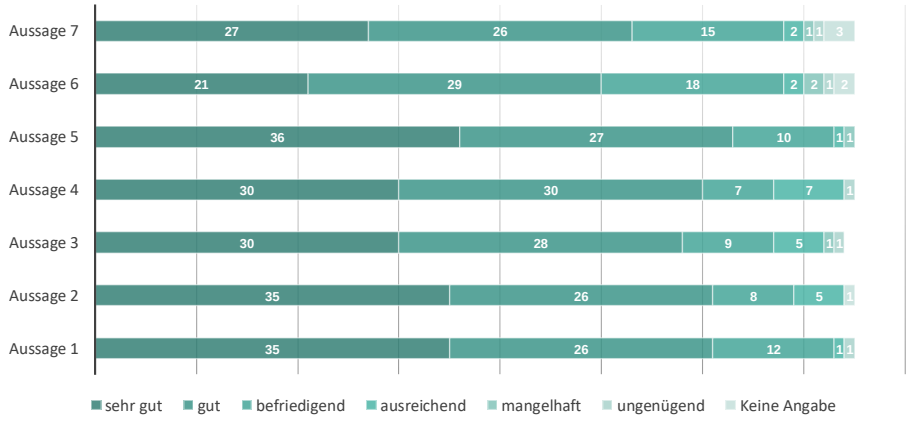


Abbildung 105: Evaluation - Bewertung von Aussagen von den Teilnehmern vor der Interpretation der Ablaufreihenfolge der Elemente der Geschäftsprozessmodelle in Abhängigkeit der Teilnehmer mit fortgeschrittenen Kenntnissen im Geschäftsprozessmanagement

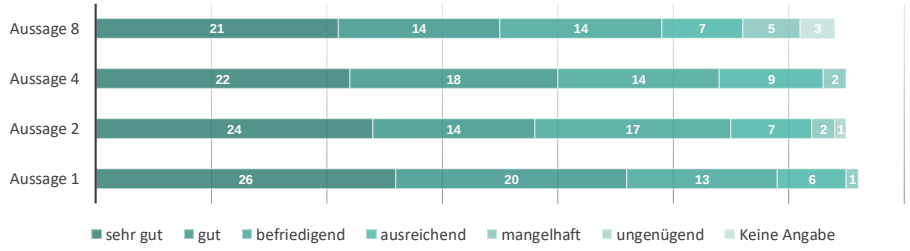


Abbildung 106: Evaluation - Bewertung von Aussagen von den Teilnehmern nach der Interpretation der Ablaufreihenfolge der Elemente der Geschäftsprozessmodelle in Abhängigkeit der Teilnehmer mit fortgeschrittenen Kenntnissen im Geschäftsprozessmanagement

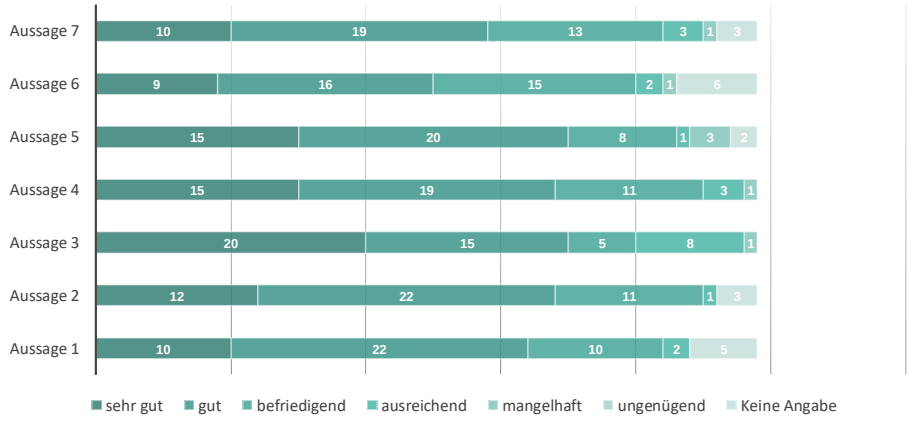


Abbildung 107: Evaluation - Bewertung von Aussagen von den Teilnehmern vor der Interpretation der Ablaufreihenfolge der Elemente der Geschäftsprozessmodelle in Abhängigkeit der Teilnehmer mit Grundkenntnissen im Geschäftsprozessmanagement

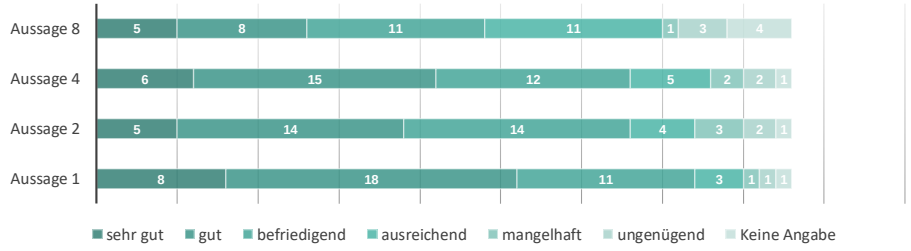


Abbildung 108: Evaluation - Bewertung von Aussagen von den Teilnehmern nach der Interpretation der Ablaufreihenfolge der Elemente der Geschäftsprozessmodelle in Abhängigkeit der Teilnehmer mit Grundkenntnissen im Geschäftsprozessmanagement

Abschließend kann aus den Aussagen abgeleitet werden, dass eine einfache Benutzbarkeit, leichte Erlernbarkeit, einfache Beschreibung der Kontrollflusssemantik (bzw. Erwartungshaltung), Nützlichkeit und leichte Verständlichkeit unterstellt werden kann, da alle betrachteten Aussagen unter allen Teilnehmern mit durchschnittlich gut bewertet wurden. Eine Ausnahme bildet die achte Aussage, woraus abgeleitet wurde, dass die Methode zwar nicht gerne angewendet wird, aber dennoch eine gewisse Nützlichkeit unterstellt werden kann. Die entwickelten Hypothesen können somit gestützt werden, wobei detaillierte Untersuchungen durchgeführt werden sollten, um beispielsweise die prinzipiell höhere Standardabweichung der Teilnehmer mit Grundkenntnissen im Geschäftsprozessmanagement

erklären zu können und darüber hinaus die schlechtere Bewertung der Aussagen nach der Interpretation der Kontrollflusssemantik zu untersuchen.

Die letzte Frage ermöglichte freie Anmerkungen bzw. die Angabe von *Verbesserungsvorschlägen*, die von zahlreichen Teilnehmern auch genutzt wurden. 13 Teilnehmer haben ein Feedback bzw. Verbesserungsvorschläge abgegeben. Beispielsweise wurde von einem Teilnehmer angemerkt, dass die einsetzbaren Annotationen in den Geschäftsprozessmodellen in einer leicht verständlichen Sprache wiedergegeben werden sollten. Dieser Aspekt sollte eigentlich durch den Pretest abgefangen werden. Wahrscheinlich ist diese Anmerkung darauf zurückzuführen, dass keine fachfremden Teilnehmer mit in die Phase der Pretests des Informationsvideos und des Fragebogens involviert waren. Ein weiterer Teilnehmer hat angemerkt, dass die Methode didaktisch besser aufbereitet werden sollte, da das Informationsvideo von dem Teilnehmer bzw. der Teilnehmerin nur aufgrund von Vorkenntnissen verstanden werden konnte. Dies kann ein Grund sein, für die tendenziell schlechteren Bewertungen der Aussagen der Teilnehmer, die nur Grundkenntnisse im Geschäftsprozessmanagement vorweisen können. Dementsprechend sollte das Informationsvideo in Zusammenarbeit mit fachfremden Personen noch einmal überarbeitet werden. Die anderen Anmerkungen bzw. Verbesserungsvorschläge gehen in eine ähnliche Richtung, sodass keine Verbesserungsvorschläge zur Methode selbst gemacht wurden, sondern eher zur didaktischen Aufbereitung und Darstellung sowie Beschreibung der Kontrollflussmuster in den Geschäftsprozessmodellen.

7.1.5 Berichterstattung

Die Ergebnisse der empirischen Untersuchung sowie das überarbeitete Informationsvideo wurden veröffentlicht.

7.2 Prototypische Realisierung in Microsoft Visio

Für die prototypische Realisierung in einem Software-Werkzeug der entwickelten Methoden wurde ein Microsoft Visio (MS Visio) Add-In entwickelt. MS Visio ist ein Zeichenprogramm zur schnellen und einfachen Erstellung von Modellen mit vorgegebenen Formen.

Für die Erstellung von Geschäftsprozessmodellen in MS Visio werden Formen verwendet, die in Schablonen verankert sind. Die Formen können aus den Schablonen via Drag & Drop auf die Zeichenfläche gelegt werden. In den Schablonen werden Formen zusammengefasst, sodass unter anderem in MS Visio Schablonen für die Geschäftsprozess-

modellierungssprachen BPMN 2.0, EPK und UML-Aktivitätsdiagramme existieren. Spezielle Schablonen für die Notationselemente der Modellierungssprachen YAWL bzw. newYAWL oder auch Petri-Netze existieren nicht. Dennoch gibt es typischerweise Formen bzw. Notationselemente der Modellierungssprachen auch in anderen Schablonen. Beispielsweise können die Formen *Feld* und *Kreis* aus der Schablone *Blöcke* für die Notationselemente eines Petri-Netzes verwendet werden. Für den Startknoten einer newYAWL-Spezifikation kann die Form *Start* aus der Schablone *Container und Abschlusszeichen - SharePoint 2013 - Workflow* verwendet werden. MS Visio bietet bereits eine Vielzahl von Formen an, die auch in neuen Schablonen zusammengestellt werden können. Sofern eine Form nicht existiert, können neue Formen entwickelt werden. Zur Realisierung des Software-Werkzeuges wurde MS Visio ausgewählt. Zum einen aufgrund der praktischen Relevanz des Produktes [KJHP15] und zum anderen aufgrund der Vielzahl der vorhandenen Formen, für die die Kontrollflusssemantik beschrieben werden kann. Demnach müssen typischerweise keine neuen Formen für eine zu spezifizierende Geschäftsprozessmodellierungssprache entwickelt werden. Zudem kann ein vorhandenes Modellierungswerkzeug zur Implementierung der entwickelten Methode verwendet werden.

Für die Erweiterung der Funktionalität von MS Visio können Add-Ins entwickelt werden. Hierfür muss das von Microsoft bereitgestellte Produkt Visual Studio verwendet werden, welches Projektvorlagen zur Erweiterung von MS Visio zur Verfügung stellt. Die Projektvorlagen für MS Visio werden durch die Office Entwicklungstools für Visual Studio bereitgestellt. Hierdurch kann unter anderem auf das Office Objektmodell zugegriffen werden, um neben der Funktionserweiterung auch die Benutzeroberfläche anpassen zu können. Die Programmiersprache ist wahlweise Visual Basic oder auch Visual C#. Für die im Folgenden vorgestellte prototypische Realisierung wurde ein MS Visio Add-In mit Visual Studio 2017 entwickelt und unterstützend das .NET Framework 4.6.1, die Office Developer Tools 2017 und die Programmiersprache Visual C# 2017 eingesetzt.

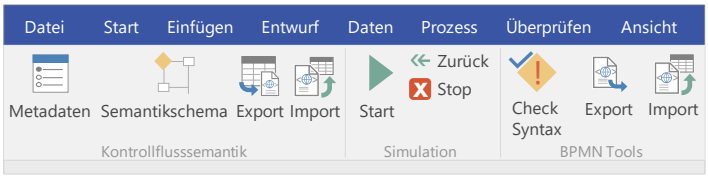


Abbildung 109: Prototyp - Menü des Microsoft Visio Add-Ins

Das MS Visio Add-In besteht aus den 3 Komponenten: (1) Kontrollflusssemantik, (2) Simulation und (3) BPMN Tools, wie in Abbildung 109 illustriert. Mit der ersten Komponente,

die in Kapitel 7.2.1 beschrieben wird, kann die Kontrollflusssemantik einer Geschäftsprozessmodellierungssprache (vgl. Kapitel 5) und die Ablaufreihenfolge der Elemente von Geschäftsprozessmodellen (vgl. Kapitel 6) festgelegt werden. Alternativ kann auch eine bereits beschriebene Kontrollflusssemantik einer Geschäftsprozessmodellierungssprache importiert werden, sofern diese bereits für ein anderes Geschäftsprozessmodell erstellt und exportiert wurde. Auf der Grundlage der beschriebenen Ablaufreihenfolge der Elemente eines Geschäftsprozessmodells können mit der zweiten Komponente, die in Kapitel 7.2.2 vorgestellt wird, die möglichen Zustände eines Geschäftsprozessmodells interaktiv betrachtet werden. Mit der dritten Komponente, die bereits in [Dres16a] vorgestellt wurde, können Geschäftsprozessmodelle, die mit der Modellierungssprache BPMN erstellt wurden, exportiert und importiert werden. Darüber hinaus kann die Syntax von BPMN-Modellen überprüft werden.

7.2.1 Beschreibung der Kontrollflusssemantik

Das Vorgehen zur Beschreibung der Kontrollflusssemantik einer Geschäftsprozessmodellierungssprache ist in Abbildung 42 illustriert, wobei zunächst die Kanten- und Knotentypen festgelegt werden müssen, um darauf aufbauend die einzelnen Knotentypen auswählen und deren Semantikschema spezifizieren zu können. Eine Spezifikation der Knotentypen ist bei der prototypischen Realisierung nicht erforderlich, da die möglichen Knotentypen durch die Formen in MS Visio bereits definiert sind. In der prototypischen Realisierung entspricht das Notationselement dem Syntaxelement, so dass eine 1 zu 1 Beziehung zwischen Notations- und Syntaxelement besteht. Dahingegen müssen die kontrollflussabhängigen, impliziten und nicht-kontrollflussabhängigen Kantentypen explizit festgelegt werden. Aus Gründen der Benutzerfreundlichkeit werden die möglichen Kantentypen für die zu definierenden Bedingungen der Kontrollflusssemantik «Vor der Ausführung» und «Nach der Ausführung» vorgegeben, da MS Visio eine Vielzahl von Formen als Kantentypen besitzt. Die Kantentypen können durch das Klicken auf den Button *Metadaten* aus der Gruppe *Kontrollflusssemantik* (vgl. Abbildung 109) definiert werden. In Abbildung 110 wird das entsprechende Dialogfenster zur Eingabe dieser Daten am Beispiel der BPMN Kantentypen illustriert. Darüber hinaus kann bei den Metadaten der Name der Kontrollflusssemantik festgelegt werden. Im Dialogfenster in Abbildung 110 können nur die Namen der Kantentypen definiert werden, sodass abweichend von dem Vorgehen aus Abbildung 42 zusätzlich die Kantentypen den Notationselementen der Modellierungssprache bzw. den Formen von MS Visio zugeordnet werden müssen. Die Zuordnung eines Notationselementes zu dem definierten Kantentyp erfolgt durch das Markieren des Notationselementes und dem nachfolgenden Anklicken des Buttons *Semantikschema* aus der Gruppe *Kontrollflusssemantik* (vgl.Abbildung 109). Daraufhin öffnet sich das Dialogfenster in Abbildung 111, bei dem das

ausgewählte Notationselement einem definierten Kantentyp zugewiesen werden kann. Der Name des Notationselementes ist der von MS Visio vorgegebene Name der Form. Die Zuordnung zum Notationselement wird nicht für die Instanz der Form auf der Zeichenfläche vorgenommen, sondern für die dazugehörige Master-Form. Eine Form auf der Zeichenfläche wird aus einer Master-Form erzeugt, welche in einer Schablone verankert ist, sodass alle Formen, die von der gleichen Master-Form abstammen, durch die gleiche Master-Form UID (Unique Identifier) identifiziert werden können. Die Master-Form UID reicht zur Identifizierung jedoch nicht immer aus, da es in MS Visio beispielsweise Formen gibt, wie die BPMN-Gateways oder die BPMN-Verbindungselemente, die von der gleichen Master-Form abstammen, aber unterschiedliche Kanten- oder Knotentypen bzw. Notationselemente darstellen. Aus diesem Grund muss zusätzlich die Konfiguration der Form gespeichert werden. Mögliche Konfigurationen für ein BPMN-Gateway sind unter anderem das exklusive, inklusive oder parallele Gateway. Für die BPMN-Verbindungselemente sind das beispielsweise der Sequenzfluss und Nachrichtenfluss sowie die Assoziation. Für die eindeutige Zuordnung der Kontrollflusssemantik zu einer MS Visio Form wird daher die Master-Form UID sowie deren entsprechende Konfiguration verwendet.

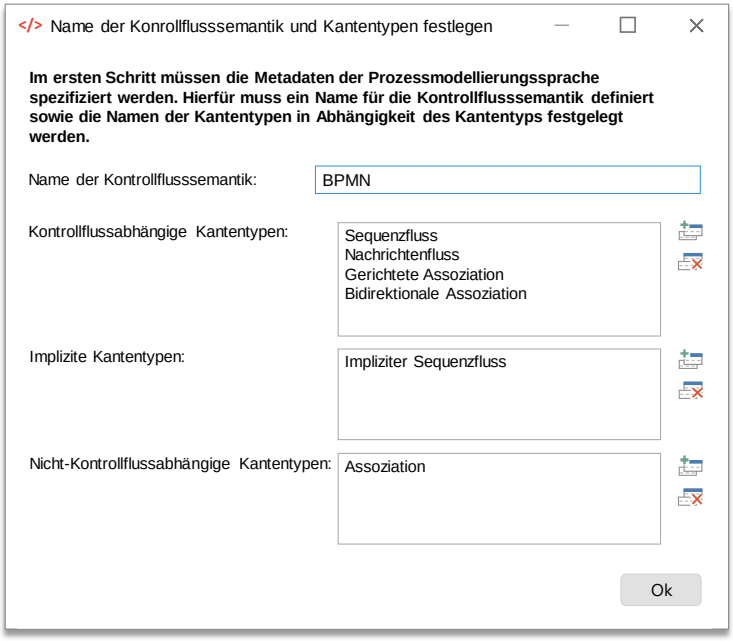


Abbildung 110: Prototyp - Name der Kontrollflusssemantik und der Kantentypen festlegen

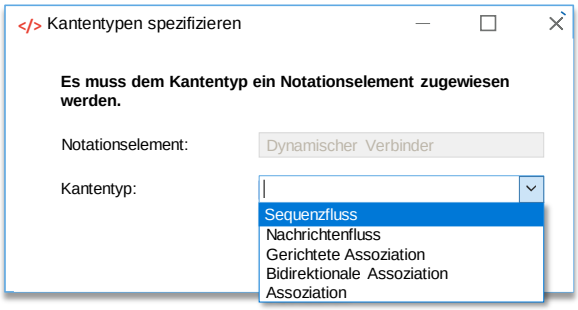


Abbildung 111: Prototyp - Kantentyp spezifizieren

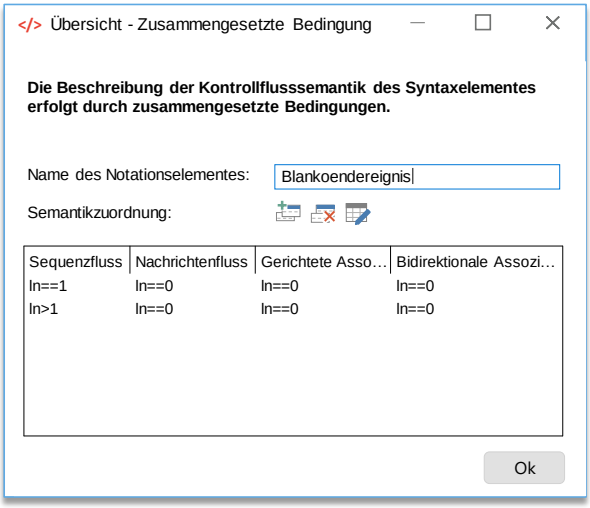


Abbildung 112: Prototyp - Übersicht der zusammengesetzten Bedingungen

Nachdem die Kantentypen sowie deren Notationselemente definiert wurden, können die Semantikschemas für die Knotentypen definiert werden (vgl. Abbildung 42). Das Semantikschema eines Knotentyps besteht im Rahmen der prototypischen Realisierung ausschließlich aus den zusammengesetzten Bedingungen (vgl. Abbildung 30). Dementsprechend wird für die Bildung des Semantikschemas nur die Kontrollflusssemantik «Vor der Ausführung» und «Nach der Ausführung» berücksichtigt. Der Gültigkeitsbereich der elementaren Bedingungen wird nicht implementiert, da in Kapitel 3 keine Modellierungssprachen betrachtet wurden, die einen Gültigkeitsbereich besitzen. Die Kontrollflusssemantik «Während der Ausführung» wurde bei der prototypischen Realisierung ebenfalls nicht implementiert. Hinsichtlich der Demonstration der Nützlichkeit und Evaluation der Zufriedenheit, der

Benutzerfreundlichkeit sowie zur Anwendung der Methode wurden diese Aspekte als nicht erforderlich angesehen, um analysefähige Geschäftsprozessmodelle zu erzeugen.

Das Semantikschema kann für einen Knotentyp definiert werden, indem zunächst eine Form auf der Zeichenfläche ausgewählt wird und sich durch das Anklicken des Buttons *Semantikschema* aus der Gruppe *Kontrollflusssemantik* (vgl. Abbildung 109) das Dialogfenster aus Abbildung 112 öffnet. Wie bereits oben beschrieben, wird der Button auch für die Zuordnung von Kantentypen in Bezug auf die Notationselemente verwendet. Die Differenzierung, welches Dialogfenster sich öffnet, ist von der ausgewählten Form abhängig. Bei einer eindimensionalen Form wird davon ausgegangen, dass es sich um einen Kantentyp handelt, wohingegen man bei einer mehrdimensionalen Form davon ausgehen kann, dass es sich um einen Knotentyp handelt. Bei einer mehrdimensionalen Form öffnet sich das Dialogfenster aus Abbildung 112, welches eine Übersicht der bereits definierten zusammengesetzten Bedingungen für den Knotentyp darstellt. Jede Zeile beschreibt eine zusammengesetzte Bedingung, die aus elementaren Bedingungen besteht. Diese Bedingungen sind jeweils mit einem logischen Und zu einer zusammengesetzten Bedingung verknüpft (vgl. EBNF-Grammatik `ZusammengesetzteBedingung`). In Abbildung 112 werden beispielsweise die Bedingungen eines BPMN-Endereignisses dargestellt. Im Unterschied zu Kapitel 5.3.1 wurde der Knotentyp Blankoereignis in Blankostart-, Blankozwischen- und Blankoendereignis untergliedert, da die Beschreibung des Semantikschemas in MS Visio über die Formen bzw. Notationselemente der Modellierungssprache vorgenommen wird. Die Bedingungen in Abbildung 112 beschreiben eine zusammengesetzte Bedingung der Kontrollflusssemantik «Vor der Ausführung», was an dem Signalwort *In* deutlich wird. In der jeweiligen Spalte des Kantentyps ist die mögliche Kantenanzahl eingetragen, d. h. die erste zusammengesetzte Bedingung beschreibt die Kontrollflusssemantik eines Blankoendereignisses, das genau einen eingehenden Sequenzfluss (In==1) und keinen eingehenden Nachrichtenfluss (In==0), keine gerichtete Assoziation (In==0) und auch keine bidirektionale Assoziation (In==0) besitzt. Weitere Details der jeweiligen elementaren Bedingungen werden erst bei der Auswahl der entsprechenden zusammengesetzten Bedingung deutlich (vgl. Abbildung 113). Für die Spezifikation der Kurzschreibweise, der möglichen ein- bzw. ausgehenden Kantenanzahl, werden die Operatoren größer (>), größer gleich (>=), gleich (==), ungleich (!=), kleiner gleich (<=) und kleiner (<) verwendet. Neben der Übersicht, der bereits definierten zusammengesetzten Bedingungen können weitere zusammengesetzte Bedingungen erzeugt sowie vorhandene zusammengesetzte Bedingungen bearbeitet oder gelöscht werden. Bei der Definition einer neuen zusammengesetzten Bedingung oder beim Bearbeiten dieser Bedingung öffnet sich das Dialogfenster aus Abbildung 113.

Bei einer neuen zusammengesetzten Bedingung muss zunächst festgelegt werden, ob es sich um eine Vor- oder Nachbedingung handelt bzw. ob die Bedingung der

Kontrollflusssemantik «Vor der Ausführung» oder «Nach der Ausführung» zugeordnet werden soll. Im Dialogfenster kann das Verhalten mit einem Dropdown-Feld ausgewählt werden. Weiterführend kann in Anlehnung an Abbildung 46 für jede elementare Bedingung ein Kantentyp, die mögliche ein- bzw. ausgehende Kantenanzahl sowie das Kontrollflussmuster festgelegt werden. Im Unterschied zur vorgestellten Methode kann jeder Bedingung nur ein Kontrollflussmuster zugeordnet werden. Sofern eine Bedingung durch mehrere Kontrollflussmuster definiert werden soll, muss für jedes Kontrollflussmuster eine separate Bedingung erzeugt werden. Derzeit unterstützt das entwickelte Add-In die Kontrollflussmuster Sequenz (wcp_1), parallele Aufspaltung (wcp_2), Synchronisation (wcp_3), exklusive Auswahl (wcp_4), einfache Zusammenführung (wcp_5), Mehrfachauswahl (wcp_6), strukturierte synchronisierte Zusammenführung (wcp_7), Mehrfachzusammenführung (wcp_8) und strukturierter Diskriminator (wcp_9). Darüber hinaus wurden die Kontrollflussmuster Start (wcp_{Start}) und implizite Beendigung (wcp_{11}) implementiert. Für die implementierten Kontrollflussmuster müssen die modellspezifischen Parameter Struc, ArcCond, $<_{XOR}$ und Default spezifiziert werden (vgl. Kapitel 6). Im Rahmen des Prototyps wurde nur die Spezifikation des strukturierten Knotens (Struc) implementiert, da für die Anwendung der implementierten interaktiven Simulation (vgl. Kapitel 7.2.2) die anderen Parameter nicht erforderlich sind. Die Kantenbedingungen (ArcCond), die Auswertungsreihenfolge der Kantenbedingungen ($<_{XOR}$) und die Default-Kante werden durch den Anwender der interaktiven Simulation festgelegt. Das Semantikschema der Knotentypen wird durch die Kombination der zusammengesetzten Bedingungen definiert, d. h. jeweils aus

– einer zusammengesetzten Bedingung aus der Kontrollflusssemantik «Vor der Ausführung» und
– einer zusammengesetzten Bedingung aus der Kontrollflusssemantik «Nach der Ausführung».

Die Kontrollflusssemantik des Knotentyps, wie auch die Kontrollflusssemantik des Kantentyps wird nicht für die Instanz der Form auf der Zeichenfläche spezifiziert, sondern für die Master-Form (bzw. die Form in der Schablone) in der entsprechenden verwendeten Konfiguration auf der Zeichenfläche. Nachdem ein Semantikschema für einen Knotentyp erzeugt wurde, können in Anlehnung an das Vorgehen aus Abbildung 42 zur Beschreibung der Kontrollflusssemantik von Geschäftsprozessmodellierungssprachen weitere Semantikschemas definiert werden. Es können aber auch Semantikschemas für andere Knotentypen festgelegt werden. Sobald die Kontrollflusssemantik für eine Geschäftsprozessmodellierungssprache bzw. für die Knoten- und Kantentypen beschrieben wurde, kann die Kontrollflusssemantik in ein Extensible Markup Language (XML)-Format exportiert werden. Beispielsweise wird im Folgenden die erstellte präzise Kontrollflusssemantik für Petri-Netze im XML-Format dargestellt, die in ein Software-Werkzeug importiert werden kann, um

darauf aufbauend Geschäftsprozessmodelle interaktiv zu simulieren. Alternativ könnte diese Kontrollflusssemantik auch eine Grundlage bilden, um die Kontrollflusssemantik von anderen Modellierungssprachen zu beschreiben. Dabei wurde als Notationselement für den Kantentyp Sequenzfluss die MS Visio Form BPMN-Verbindungselement in der Konfiguration Sequenzfluss verwendet. Für den Knotentyp Stelle wurde die MS Visio Form BPMN-Ereignis in der Konfiguration *Blankostartereignis* verwendet und für den Knotentyp Transition die MS Visio Form *Feld* aus der Schablone Blöcke eingesetzt.

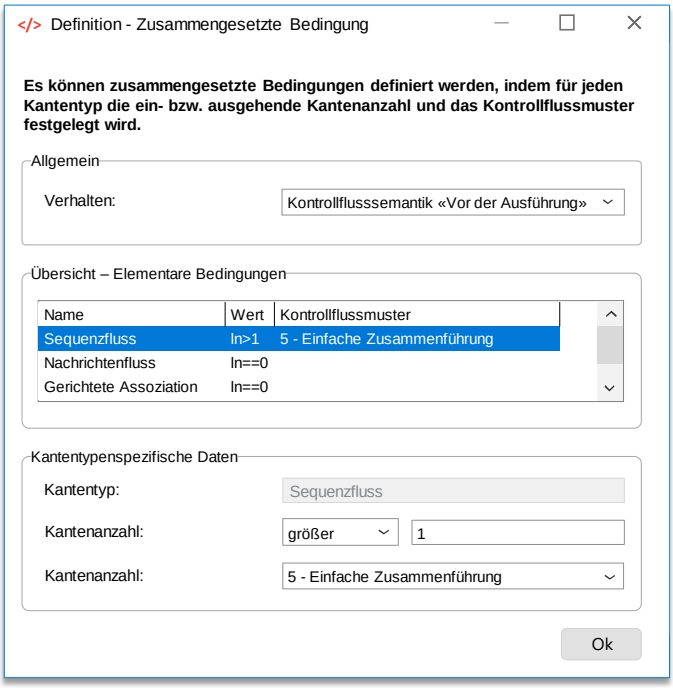


Abbildung 113: Prototyp - Definition einer zusammengesetzten Bedingung

```
<?xml version="1.0"?>
<Kontrollflusssemantik xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://www.dre-
scher.org/kontrollflusssemantik">
  <Metadaten Schemaname="Petri-Netz">
    <Kantentyp ID="f1cd3ee5-2e77-49b6-a546-4483d9e04994" Name="Sequenzfluss">
      <Mastershape Mastershape-ID="3f4bf83f-000b-0000-8e40-00608cf305b2">
        <Konfiguration Bedingung="Actions.SequenceFlow.Checked">1</Konfiguration>
        <Konfiguration Bedingung="Actions.MessageFlow.Checked">0</Konfiguration>
        <Konfiguration Bedingung="Actions.Association.Checked">0</Konfiguration>
        <Konfiguration Bedingung="Actions.ConditionType.Checked">0</Konfiguration>
        <Konfiguration Bedingung="Actions.ConditionNone.Checked">1</Konfiguration>
```

```
        <Konfiguration Bedingung="Actions.ConditionExpression.Checked">0</Konfigu-
ration>
        <Konfiguration Bedingung="Actions.ConditionDefault.Checked">0</Konfigura-
tion>
        <Konfiguration Bedingung="Actions.Direction.Checked">0</Konfiguration>
      </Mastershape>
    </Kantentyp>
  </Metadaten>
  <Knotentypen>
    <Knotentyp ID="e381321c-fe65-424a-bfbb-5d6f6d4a766c" Name="Transition">
      <Mastershape Mastershape-ID="287c7801-0002-0000-8e40-00608cf305b2" />
      <Semantikschemas>
        <Semantikschema ID="89d13f3c-a6bb-11e8-98d0-529269fb1459" Verhalten="In">
          <Kantentyp ID="f1cd3ee5-2e77-49b6-a546-4483d9e04994" Operator="gleich"
Value="0" Kontrollflussmuster="0 - Start"/>
        </Semantikschema>
        <Semantikschema ID="87191493-1671-48c7-a1c1-285442a3837e" Verhalten="In">
          <Kantentyp ID="f1cd3ee5-2e77-49b6-a546-4483d9e04994" Operator="gleich"
Value="1" Kontrollflussmuster="1 - Sequenz"/>
        </Semantikschema>
        <Semantikschema ID="ace17573-3624-4614-9ff3-074be29d857a" Verhalten="In">
          <Kantentyp ID="f1cd3ee5-2e77-49b6-a546-4483d9e04994" Operator="größer"
Value="1" Kontrollflussmuster="3 - Synchronsation"/>
        </Semantikschema>
        <Semantikschema ID="a5d5b952-740f-4c5d-a3e4-e222fb68c809"
Verhalten="Out">
          <Kantentyp ID="f1cd3ee5-2e77-49b6-a546-4483d9e04994" Operator="gleich"
Value="1" Kontrollflussmuster="1 - Sequenz"/>
        </Semantikschema>
        <Semantikschema ID="9561a7aa-55d8-40ee-a983-7f8a55d76f57"
Verhalten="Out">
          <Kantentyp ID="f1cd3ee5-2e77-49b6-a546-4483d9e04994" Operator="größer"
Value="1" Kontrollflussmuster="2 - Parallele Aufspaltung"/>
        </Semantikschema>
        <Semantikschema ID="aadcd97a-a6bb-11e8-98d0-529269fb1459"
Verhalten="Out">
          <Kantentyp ID="f1cd3ee5-2e77-49b6-a546-4483d9e04994" Operator="gleich"
Value="0" Kontrollflussmuster="11 – Implizite Beendigung "/>
        </Semantikschema>
       </Semantikschemas>
    </Knotentyp>
    <Knotentyp ID="21221df1-f923-4112-a239-6c12b0ab3b12" Name="Stelle">
      <Mastershape Mastershape-ID="3f4bb759-0007-0000-8e40-00608cf305b2">
        <Konfiguration Bedingung="Actions.EventType.Checked">0</Konfiguration>
        <Konfiguration Bedingung="Actions.Start.Checked">1</Konfiguration>
        <Konfiguration Bedingung="Actions.Intermediate.Checked">0</Konfiguration>
        <Konfiguration Bedingung="Actions.IntermediateThrowing.Checked">0</Konfigu-
ration>
        <Konfiguration Bedingung="Actions.End.Checked">0</Konfiguration>
        <Konfiguration Bedingung="Actions.TriggerResult.Checked">0</Konfiguration>
        <Konfiguration Bedingung="Actions.NoTriggerResult.Checked">1</Konfigura-
tion>
        <Konfiguration Bedingung="Actions.Message.Checked">0</Konfiguration>
        <Konfiguration Bedingung="Actions.Timer.Checked">0</Konfiguration>
        <Konfiguration Bedingung="Actions.Error.Checked">0</Konfiguration>
        <Konfiguration Bedingung="Actions.Cancel.Checked">0</Konfiguration>
        <Konfiguration Bedingung="Actions.Compensation.Checked">0</Konfiguration>
```

```
        <Konfiguration Bedingung="Actions.Conditional.Checked">0</Konfiguration>
        <Konfiguration Bedingung="Actions.Link.Checked">0</Konfiguration>
      </Mastershape>
      <Semantikschemas>
        <Semantikschema ID="44ecc92c-a6bb-11e8-98d0-529269fb1459" Verhalten="In">
          <Kantentyp ID="f1cd3ee5-2e77-49b6-a546-4483d9e04994" Operator="gleich"
Value="0" Kontrollflussmuster="0 - Start"/>
        </Semantikschema>
        <Semantikschema ID="11bb3c36-a52c-4e7d-85c4-20ee4bd9d4b7" Verhalten="In">
          <Kantentyp ID="f1cd3ee5-2e77-49b6-a546-4483d9e04994" Operator="gleich"
Value="1" Kontrollflussmuster="1 - Sequenz"/>
        </Semantikschema>
        <Semantikschema ID="9de9b246-1995-4fd0-9e63-a91e61b2bcd9" Verhalten="In">
          <Kantentyp ID="f1cd3ee5-2e77-49b6-a546-4483d9e04994" Operator="größer"
Value="1" Kontrollflussmuster="5 - Einfache Zusammenführung"/>
        </Semantikschema>
        <Semantikschema ID="2dd878a9-622e-4b3e-a279-d3d2438a861f"
Verhalten="Out">
          <Kantentyp ID="f1cd3ee5-2e77-49b6-a546-4483d9e04994" Operator="gleich"
Value="1" Kontrollflussmuster="1 - Sequenz"/>
        </Semantikschema>
        <Semantikschema ID="a9c37ad6-6b48-4e46-90a2-807be521e62e"
Verhalten="Out">
          <Kantentyp ID="f1cd3ee5-2e77-49b6-a546-4483d9e04994" Operator="größer"
Value="1" Kontrollflussmuster="4 - Exklusive Auswahl"/>
        </Semantikschema>
        <Semantikschema ID="55d9a89a-a6bb-11e8-98d0-529269fb1459"
Verhalten="Out">
          <Kantentyp ID="f1cd3ee5-2e77-49b6-a546-4483d9e04994" Operator="gleich"
Value="0" Kontrollflussmuster="11 – Implizite Beendigung"/>
        </Semantikschema>
      </Semantikschemas>
    </Knotentyp>
  </Knotentypen>
</Kontrollflusssemantik>
```

Durch den Ex- und Import der Kontrollflusssemantik besteht jederzeit die Möglichkeit die Kontrollflusssemantik anzupassen oder auch auf andere dokumentierte Geschäftsprozessmodelle in MS Visio zu übertragen bzw. anzuwenden. Darüber hinaus kann die Kontrollflusssemantik in dem XML-Format verwendet werden, um die Kontrollflusssemantik mit anderen Software-Werkzeugen auszutauschen.

7.2.2 Interaktive Simulation von Geschäftsprozessmodellen

Mit der Anwendung der entwickelten Methode zur Beschreibung der Ablaufreihenfolge der Elemente eines Geschäftsprozessmodells werden systematische Untersuchungen möglich. Beispielsweise kann das Geschäftsprozessmodell mit einer interaktiven Simulation analysiert werden. In diesem Zusammenhang können ungewollte Mehrdeutigkeiten vermieden werden, indem Schritt für Schritt die mögliche Ablaufreihenfolge betrachtet wird.

Alternativ können andere Analyseverfahren für die Geschäftsprozessmodelle angewendet werden, um die inhaltliche Korrektheit zu überprüfen [Dres18].

Für die interaktive Simulation muss das Geschäftsprozessmodell um Zustände erweitert werden. Dementsprechend erhalten im Rahmen der prototypischen Realisierung die Knoten auf der Zeichenfläche in MS Visio einen Zustand. Bei der prototypischen Realisierung der interaktiven Simulation besitzt ein Knoten genau einen der vier möglichen Zustände. Die möglichen Zustände sind in den Geschäftsprozessmodellen in Abbildung 116 und Abbildung 117 illustriert. Der Zustand eines Knotens im Geschäftsprozessmodell ist:

(1) «unmarkiert», wenn der Knoten aufgrund des aktuellen Zustandes der Geschäftsprozessinstanz nicht ausgeführt werden kann und auch nicht auf die Ausführung von anderen Knoten wartet, damit der Knoten ausgeführt werden kann. Die Knoten, die im Geschäftsprozessmodell den Zustand «unmarkiert» besitzen, haben keine spezielle visuelle Kennzeichnung.

(2) «markiert», wenn der Knoten aktiviert ist. Ein Knoten ist aktiviert, wenn alle notwendigen Knoten im Vorbereich ausgeführt wurden. Die notwendigen ausgeführten Knoten im Vorbereich ergeben sich aus der Vorbedingung bzw. aus der Kontrollflusssemantik «Vor der Ausführung» des zu aktivierenden Knotens. Ein Knoten, der den Zustand «markiert» besitzt, wird visuell mit Grün hervorgehoben.

(3) «markiert, nicht ausführbar», wenn der Knoten aufgrund von noch notwendigen auszuführenden Knoten im Vorbereich noch nicht aktiviert werden kann. Die notwendigen ausgeführten Knoten im Vorbereich ergeben sich aus der Vorbedingung bzw. der Kontrollflusssemantik «Vor der Ausführung» des zu aktivierenden Knotens. Ein Knoten, der den Zustand «markiert, nicht ausführbar» besitzt, wird visuell mit Rot hervorgehoben.

(4) «hervorgehoben», wenn der Knoten den Zustand «markiert» erhalten würde, aber der Knoten nur solange ausgeführt werden kann, sofern bestimmte andere Knoten noch nicht ausgeführt wurden. Dementsprechend kann der Zustand nur in Verbindung mit den implementierten Kontrollflussmustern exklusive Auswahl (wcp_4) oder Mehrfachauswahl (wcp_6) auftreten. Ein Knoten, der den Zustand «hervorgehoben» besitzt, wird visuell mit Gelb hervorgehoben.

Das Vorgehen einer interaktiven Simulation eines Geschäftsprozessmodells ist in Abbildung 114 mit einem BPMN-Modell visualisiert. Zunächst wird beim Start der interaktiven Simulation eines Geschäftsprozessmodells eine Instanz des Geschäftsprozessmodells erzeugt, wodurch die Knoten im Geschäftsprozessmodell den Zustand «unmarkiert» erhalten.

Nachfolgend kann der Startzustand bzw. können die Startknoten ermittelt werden. Bei mehreren Startknoten muss der Anwender der interaktiven Simulation einen Startknoten aus der Menge der möglichen Startknoten auswählen (vgl. Abbildung 115). Abschließend kann die Ablaufreihenfolge der Elemente des Geschäftsprozessmodells interaktiv simuliert werden (vgl. Abbildung 118). Im Folgenden wird auf die Auswahl und Ermittlung der Startknoten eingegangen sowie das Vorgehen der interaktiven Simulation vorgestellt.

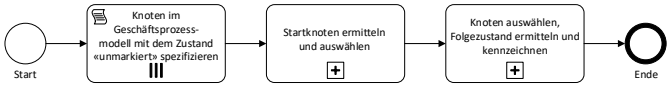


Abbildung 114: Vorgehen - Interaktive Simulation von Geschäftsprozessmodellen

Für die Ermittlung der Startknoten (vgl. Abbildung 115) müssen die zutreffenden Semantikschemas der Knoten bzw. die zutreffenden zusammengesetzten Bedingungen identifiziert werden. Eine zusammengesetzte Bedingung trifft zu, wenn jede elementare Bedingung der zusammengesetzten Bedingung zutrifft. Eine elementare Bedingung trifft zu, wenn die Anzahl der verbundenen ein- bzw. ausgehenden Kanten des Kantentyps im Bereich der erforderlichen definierten Kantenanzahl für den Kantentyp liegt. Alle zutreffenden Bedingungen, die mit dem Kontrollflussmuster Start spezifiziert sind, sind mögliche Startknoten. Sofern mehrere Startknoten existieren, werden diese Knoten mit dem Zustand «hervorgehoben» gekennzeichnet. Vom Anwender kann im nächsten Schritt daraufhin manuell ein Startknoten ausgewählt werden, wie auch an dem BPMN-Aufgabentyp in Abbildung 115 deutlich wird. Der ausgewählte Startknoten erhält den Zustand «markiert» und die anderen Startknoten erhalten den Zustand «unmarkiert». Falls nur ein Startknoten identifiziert werden konnte, wird dieser Knoten mit dem Zustand «markiert» spezifiziert.

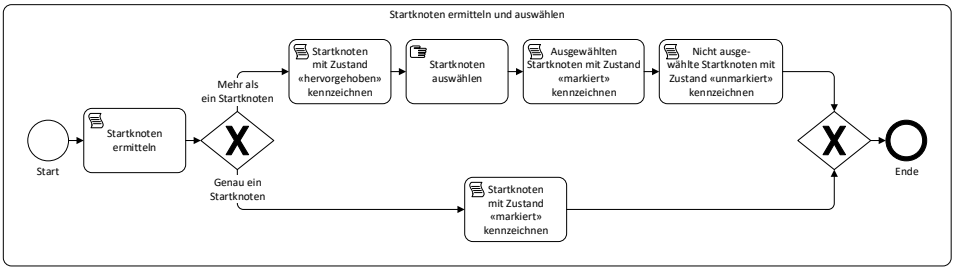


Abbildung 115: Vorgehen - Startknoten ermitteln und auswählen

Ausgehend von dem Startzustand können die Folgezustände der Geschäftsprozessinstanz ermittelt werden, wie in Abbildung 118 visualisiert wird. Ein Folgezustand der Geschäftsprozessinstanz kann ermittelt werden, indem zunächst ein Knoten im Geschäftsprozessmodell ausgewählt wird. Besitzt der ausgewählte Knoten den Zustand «unmarkiert» oder «markiert, nicht ausführbar», dann ändert sich der Zustand der Geschäftsprozessinstanz nicht und es kann erneut ein Knoten ausgewählt werden. Wenn der ausgewählte Knoten den Zustand «hervorgehoben» besitzt, dann werden zunächst die Zustände der Knoten zurückgesetzt, die von dem ausgewählten Knoten abhängen. Beispielsweise besitzen die Aufgaben 5 und 6 in Abbildung 116 den Zustand «hervorgehoben» sowie auch in Abbildung 117 die Transitionen t_6 und t_9. Sofern Aufgabe 5 bzw. Transition t_6 als Knoten ausgewählt wird, wird der Zustand für Aufgabe 6 bzw. die Transition t_9 auf den Zustand «unmarkiert» zurückgesetzt. Falls ein Knoten mit dem Zustand «markiert» ausgewählt wird, fällt dieser Schritt weg. Im Anschluss können die Knoten ermittelt werden, die für eine Zustandsänderung betrachtet werden müssen. Ausgehend von der zutreffenden zusammengesetzten Bedingung der Kontrollflusssemantik «Nach der Ausführung» des ausgewählten Knotens, wird die Menge der zu betrachtenden Knoten für eine Zustandsänderung definiert. Für die Knoten aus dieser Menge wird überprüft, ob der Zustand abhängig von der zutreffenden zusammengesetzten Bedingung der Kontrollflusssemantik «Vor der Ausführung» geändert werden muss. Beispielsweise geht aus Abbildung 116 hervor, dass die Aufgabe 4 ausgeführt wurde und die Menge der zu überprüfenden Knoten aus dem zusammenführenden inklusiven Gateway besteht. Aufgrund des definierten Kontrollflussmusters der strukturierten synchronisierten Zusammenführung (wcp_7) des inklusiven Gateways der elementaren Bedingung und der Auswahl der Aufgaben 3 und 4 mit dem verzweigenden inklusiven Gateway wird dem zusammenführenden inklusiven Gateway der Zustand «markiert, nicht ausführbar» zugewiesen. Sobald Aufgabe 3 ausgeführt wurde, erhält der Knoten den Zustand «markiert». Analog gilt dies für das Beispiel in Abbildung 117, wobei der Knoten t_5 für die Kontrollflusssemantik «Vor der Ausführung» das Kontrollflussmuster Synchronisation (wcp_3) in der elementaren Bedingung besitzt. Die Transition t_5 besitzt den Zustand «markiert, nicht ausführbar», da die Transition erst schalten bzw. ausgeführt werden kann, wenn alle Knoten im Vorbereich ausgeführt wurden. Zusammenfassend werden die möglichen Knoten für eine Zustandsänderung vom ausgeführten Knoten und deren zutreffenden zusammengesetzten Bedingungen der Kontrollflusssemantik «Nach der Ausführung» definiert. Für die Knoten aus dieser Menge wird abhängig von der zutreffenden zusammengesetzten Bedingung der Kontrollflusssemantik «Vor der Ausführung» und den Knoten im Vorbereich eine Zustandsänderung durchgeführt.

Das Vorgehen der interaktiven Simulation wurde ebenfalls in dem Prototyp realisiert und kann gestartet werden, indem auf den Button *Start* aus der Gruppe *Simulation* (vgl. Abbildung 109) geklickt wird.

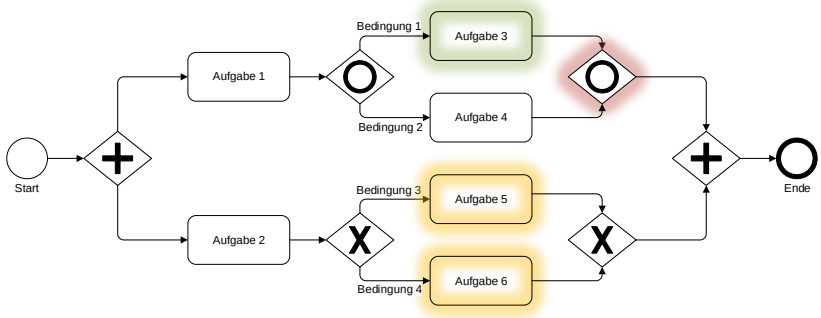


Abbildung 116: Beispiel - BPMN - Zustände eines Prozessmodells

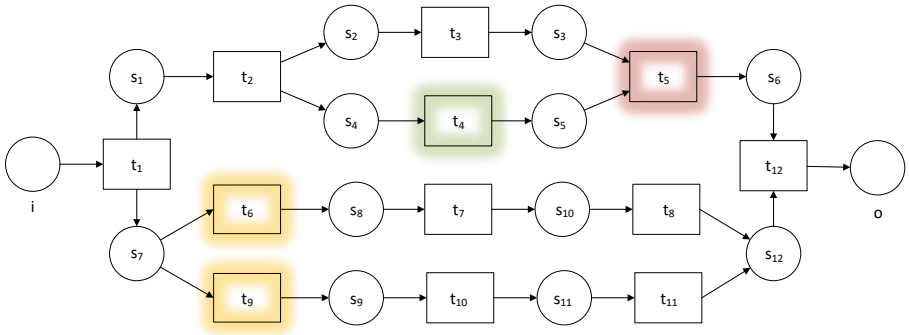


Abbildung 117: Beispiel - Petri-Netz - Zustände eines Prozessmodells

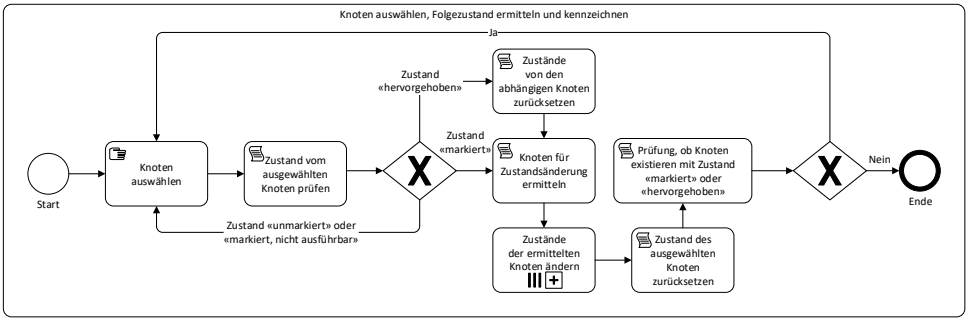


Abbildung 118: Vorgehen - Knoten auswählen, Folgezustand ermitteln und kennzeichnen

Auf der Tagung Modellierung 2018, der Gesellschaft für Informatik e.V. [SKVM+18], wurde der Prototyp durch Fachexperten auf der Grundlage des USE Fragebogens [Lund01] evaluiert. USE steht für Nützlichkeit (Usefulness), Zufriedenheit (Satisfaction) und Anwendbarkeit (Ease of use). Bei dem Fragebogen müssen die Teilnehmer Aussagen über die Nützlichkeit, Anwendbarkeit, Erlernbarkeit und Zufriedenheit mit einer Bewertungsskala von 1 bis 7 bewerten. Zusätzlich besteht die Möglichkeit, die Aussage mit NA (no answer bzw. keine Antwort) zu beantworten. Bei der Bewertung einer Aussage mit Ziffer 1 (Disagree) gibt der Teilnehmer an, dass die Aussage gar nicht zutrifft oder mit Ziffer 7 (Agree), dass die Aussage voll zutrifft. Bei der Untersuchung konnten 12 Fachexperten befragt werden. Die Auswertung der Fragen wird in Tabelle 51 bis Tabelle 54 aufgezeigt, an der insgesamt zu erkennen ist, dass die Fachexperten die Aussagen zur Methode und Software positiv bewerten. Dies wird jeweils an der hohen durchschnittlichen Bewertung der jeweiligen Aussagen und der geringen Standardabweichung deutlich. Darüber hinaus haben 3 Teilnehmer ein Interesse an dem vorgestellten Software-Werkzeug geäußert, welches derzeit unter anderem in einem Beratungsunternehmen eingesetzt wird.

Aussage	Durchschnitt	Standardabweichung	Bewertungen
Die Methode hilft mir effektiver zu sein.	6,2	0,79	10
Die Methode hilft mir produktiver zu sein.	6,45	0,69	11
Die Methode ist nützlich.	6	0,95	12
Die Methode macht es einfacher Dinge zu erledigen, die ich erreichen möchte.	5,83	1,47	12
Die Methode spart Zeit, wenn ich sie benutze.	6,33	0,78	12
Die Methode erfüllt meine Bedürfnisse.	5,73	1,01	11
Die Methode tut alles, was ich von ihr erwarte.	5,91	1,16	12

Tabelle 51: Evaluation - Nützlichkeit der Methode

Aussage	Durchschnitt	Standardabweichung	Bewertungen
Ich konnte die Methode schnell einsetzen.	6,42	0,90	12
Ich erinnere mich leicht daran, wie man die Methode benutzen kann.	6,42	0,67	12
Es ist leicht zu lernen, die Methode zu benutzen.	6,42	0,67	12
Ich war mit der Methode schnell vertraut.	6,67	0,65	12

Tabelle 52: Evaluation - Erlernbarkeit der Methode

Aussage	Durchschnitt	Standardabweichung	Bewertungen
Die Software ist leicht zu bedienen.	6,42	0,79	12
Die Software ist benutzerfreundlich.	6,08	1	12
Die Software erfordert wenige Schritte, um das zu erreichen, was ich damit machen möchte.	6,18	0,6	11
Die Software ist flexibel.	5,5	1,31	12
Die Verwendung der Software ist einfach.	6,42	0,79	12
Ich kann die Software ohne schriftliche Anweisungen verwenden.	6,17	0,94	12
Ich sehe keine Unstimmigkeiten, wie ich die Software benutze.	6	0,6	12
Sowohl gelegentliche als auch regelmäßige Benutzer würden die Software mögen.	5,5	1,09	12
Ich kann Fehler schnell und einfach beheben.	6,11	0,78	9
Ich kann die Software jeder Zeit problemlos verwenden.	6,09	0,83	11

Tabelle 53: Evaluation - Benutzerfreundlichkeit der Software

Aussage	Durchschnitt	Standardabweichung	Bewertungen
Ich bin mit der Software zufrieden.	6,25	0,62	12
Ich würde die Software einem Freund empfehlen.	6,08	0,79	12
Es macht Spaß die Software zu benutzen.	6,27	0,65	11
Die Software funktioniert so, wie ich es will.	6,33	0,65	12
Die Software ist wundervoll.	6,36	1,03	11
Ich glaube, dass ich die Software haben muss.	6,0	1,04	12
Die Software ist angenehm zu benutzen.	6,33	0,89	12

Tabelle 54: Evaluation - Zufriedenheit mit der Software

7.3 Zusammenfassung

In den vorherigen Abschnitten wurde zum einen eine empirische Untersuchung beschrieben. Der Untersuchungsgegenstand war die entwickelte Methode um die Ablaufreihenfolge der Elemente von Geschäftsprozessmodellen präzise beschreiben zu können (vgl.

Kapitel 6). Zum anderen wurde die prototypische Realisierung der entwickelten Methode in einem Software-Werkzeug vorgestellt, welche auch im Rahmen der Tagung Modellierung 2018 von der Gesellschaft für Informatik e.V. durch Fachexperten evaluiert wurde.

Im ersten Abschnitt wurden erste Erkenntnisse für die Verbesserung der Verständlichkeit der Kontrollflusssemantik einer Oder-Zusammenführungen in Geschäftsprozessmodellen gesammelt. Dabei wurden im Rahmen einer Online-Umfrage Teilnehmer befragt, die die Ablaufreihenfolge der Elemente in den Geschäftsprozessmodellen mit und ohne Anwendung der entwickelten Methode (vgl. Abbildung 27 und Abbildung 90) anhand von vorgegebenen Antwortmöglichkeiten angeben sollten. Die Interpretation der Ablaufreihenfolge der Elemente im Geschäftsprozessmodell durch die Teilnehmer konnte im Ergebnis nach der Intention des Modellierers mit der Annotation von Kontrollflussmustern

- für das erste zu bewertende Geschäftsprozessmodell von 27% auf 80%,
- für das zweite zu bewertende Geschäftsprozessmodell von 66% auf 90% und
- für das dritte zu bewertende Geschäftsprozessmodell von 6% auf 76%

verbessert werden. Weiterführend wurden erste Anhaltspunkte für die Vermutungen der einfachen Benutzbarkeit, leichten Erlernbarkeit, einfachen Beschreibung der Kontrollflusssemantik (bzw. Erwartungshaltung), Nützlichkeit und leichten Verständlichkeit der in Kapitel 6 entwickelten Methode anhand von zu bewertenden Aussagen in der Online-Umfrage gesammelt. Im Allgemeinen wurden alle aus den Vermutungen formulierten Aussagen mit sehr gut bis gut bewertet, mit Ausnahme der Aussage *ich würde die Methode gerne nutzen, da ich die Nützlichkeit sehe* (Durchschnittliche Note: 2,65 mit einer Standardabweichung: 0,04). Die Nützlichkeit der Methode wurde mit der Note 1,89 (Standardabweichung: 0,04) bewertet, so dass im Allgemeinen eine gewisse Nützlichkeit unterstellt werden kann, dementgegen die Methode aber nicht gerne angewendet wird. Nach der Interpretation der Ablaufreihenfolge der Elemente in den Geschäftsprozessmodellen sollten die Teilnehmer die Aussagen über die Nützlichkeit, Benutzbarkeit und Verständlichkeit erneut bewerten. Im Vergleich zur vorherigen Bewertung wurden die Aussagen durchschnittlich um eine Drittel bis halbe Note schlechter bewertet, dennoch wurde aber überwiegend die Note gut vergeben. Die schlechtere Bewertung der Aussagen nach der Interpretation der Kontrollflusssemantik ist wahrscheinlich auf das Informationsvideo zurückzuführen, wie aus einzelnen Kommentaren von Teilnehmern hervorgeht. Dementsprechend wurde das Informationsvideo überarbeitet. Einzelne Aspekte, wie die tendenziell schlechtere Bewertung der Aussagen nach der Anwendung der Methode oder auch die generelle Verständlichkeit einer Oder-Zusammenführung, unabhängig von Geschäftsprozessmodellen, sollten betrachtet werden. Die durchgeführte empirische Untersuchung liefert lediglich erste Anhaltspunkte für die aufgestellten Hypothesen. Es sollten detailliertere Untersuchungen

durchgeführt werden, wobei auch Korrelationen zwischen den einzelnen Vermutungen zu betrachten sind.

Im zweiten Abschnitt wurde eine prototypische Realisierung der entwickelten Methoden in Microsoft Visio vorgestellt. Dabei wurde zum einen beschrieben, wie mit einem Software-Werkzeug die Kontrollflusssemantik einer Geschäftsprozessmodellierungssprache und die Ablaufreihenfolge der Elemente eines Geschäftsprozessmodells beschrieben werden kann. Zum anderen wurde dadurch die Möglichkeit der Analyse von Geschäftsprozessmodellen geschaffen, die am Beispiel einer interaktiven Simulation von Geschäftsprozessmodellen demonstriert wurde. Für die interaktive Simulation wurden zunächst Zustände für die Knoten im Geschäftsprozessmodell eingeführt. Mit diesen Zuständen kann eine Geschäftsprozessinstanz interaktiv simuliert werden. Für eine umfassendere Simulation von Geschäftsprozessmodellen, z. B. um quantitative Aussagen über die Mindestdauer, den Ressourcenverbrauch oder die Kapazitätsauslastung tätigen zu können, müssen die Zustände um Attribute (z. B. Kosten) erweitert werden. Das Ziel des Software-Werkzeuges war nicht nur, die Anwendbarkeit der Methode zu demonstrieren, sondern auch durch die Anwendung der Methode analysefähige Geschäftsprozessmodelle zu schaffen.

8 Zusammenfassung und Ausblick

Das abschließende Kapitel fasst die einzelnen Ergebnisse dieser Arbeit zusammen und beschreibt den originären Beitrag der Arbeit. Darüber hinaus werden die Grenzen der entwickelten Methode kritisch betrachtet, um daraus neue und weiterführende Fragestellungen für den Ausblick abzuleiten.

8.1 Zusammenfassung

Das Ziel der Arbeit war es, eine Methode zur Beschreibung der Kontrollflusssemantik von Geschäftsprozessmodellierungssprachen zu entwickeln, um die Ablaufreihenfolge der Elemente in den Geschäftsprozessmodellen präzise beschreiben zu können. Zur Verfolgung des Ziels wurden im Wesentlichen Schablonen bzw. Kontrollflussmuster mit den Syntaxelementen der Modellierungssprache verknüpft, wodurch die Kontrollflusssemantik einer Modellierungssprache beschrieben wird.

Es wurden einführend zunächst die zentralen Begriffe: Geschäftsprozessmanagement, Geschäftsprozessmodellierungssprachen und Kontrollflusssemantik für Geschäftsprozessmodellierungssprachen vorgestellt. Ebenso wurde auf den bisherigen Stand der Forschung und Technik zur Beschreibung der Kontrollflusssemantik von Geschäftsprozessmodellierungssprachen eingegangen. Besonders wurden die Kontrollflussmuster der Workflow Pattern Initiative betrachtet, deren Kontrollflusssemantik mit gefärbten Petri-Netzen präzise beschrieben ist, woraus von [Russ07] die Modellierungssprache newYAWL entwickelt wurde. Die Kontrollflusssemantik der Kontrollflussmuster wurde mit Beispielprozessmodellen vorgestellt. Anhand der Beispiele wurden für jedes Kontrollflussmuster die notwendigen Parameter der newYAWL-Spezifikation zur präzisen Beschreibung der Kontrollflusssemantik der Kontrollflussmuster abgeleitet. Beispielsweise wurden für das Kontrollflussmuster exklusive Auswahl die Parameter Split, $<_{XOR}$, ArcCond und Default der newYAWL-Spezifikation identifiziert, wodurch die Kontrollflusssemantik der Kontrollflussmuster präzise beschrieben werden konnte.

Auf der Grundlage dieser Kontrollflussmuster wurde eine Methode zur Beschreibung der Kontrollflusssemantik von Geschäftsprozessmodellierungssprachen (vgl. Abbildung 30) vorgestellt. Darauf aufbauend wurde eine Methode zur präzisen Beschreibung der Ablaufreihenfolge der Elemente in Geschäftsprozessmodellen entwickelt (vgl. Abbildung 63). Die

Vorgehensweise zur Beschreibung der Kontrollflusssemantik für Geschäftsprozessmodellierungssprachen (vgl. Abbildung 42) und die Beschreibung der Ablaufreihenfolge der Elemente in Geschäftsprozessmodellen (vgl. Abbildung 64) wurde mit BPMN-Modellen definiert, um diese bei der prototypischen Realisierung in einem Software-Werkzeug einsetzen zu können. Es wurde eine Differenzierung zwischen den Geschäftsprozessmodellierungssprachen und Geschäftsprozessmodellen vorgenommen, da ein Knotentyp einer Geschäftsprozessmodellierungssprache unterschiedliche Bedeutungen besitzen kann. In den Geschäftsprozessmodellen muss zur präzisen Beschreibung der Kontrollflusssemantik jeder Knoten eine eindeutige Kontrollflusssemantik besitzen. Beispielsweise kann die Kontrollflusssemantik einer Oder-Zusammenführung mit einem der folgenden Kontrollflussmuster einfache Zusammenführung, strukturierte synchronisierte Zusammenführung, Mehrfachzusammenführung, strukturierter Diskriminator, lokale synchronisierte Zusammenführung oder generelle synchronisierte Zusammenführung abgebildet werden. Im Geschäftsprozessmodell darf eine Oder-Zusammenführung nur mit einem dieser Kontrollflussmuster beschrieben sein.

Die Kontrollflusssemantik eines Knotentyps einer Geschäftsprozessmodellierungssprache wird mit Semantikschemas definiert. In den Geschäftsprozessmodellen wird jedem Knoten eines dieser Semantikschemas zur Beschreibung der Kontrollflusssemantik zugewiesen. Ein Semantikschema bzw. die Kontrollflusssemantik eines Knotentyps wird durch die Kontrollflusssemantik «Vor der Ausführung», «Während der Ausführung» und «Nach der Ausführung» beschrieben (vgl. Abbildung 30). Die Kontrollflusssemantik «Vor der Ausführung» beschreibt die Anforderungen zur Aktivierung eines Knotens. Sobald alle Anforderungen erfüllt sind bzw. die erforderlichen Knoten im Vorbereich ausgeführt wurden, kann der Knoten aktiviert werden. Nur aktivierte Knoten können ausgeführt werden. Mit der Kontrollflusssemantik «Nach der Ausführung» werden die möglichen zu aktivierenden Knoten im Nachbereich nach der Ausführung des Knotens definiert. Die Kontrollflusssemantik «Während der Ausführung» beschreibt das Verhalten des Knotens während der Ausführung. Beispielsweise kann zur Ausführung des Knotens ein Trigger erforderlich sein oder es werden mehrere Instanzen erzeugt. Die Kontrollflusssemantik des Semantikschemas wird im Wesentlichen mit Kontrollflussmustern beschrieben, wobei für die Kontrollflusssemantik «Vor der Ausführung» und «Nach der Ausführung» zusätzlich Rahmenbedingungen definiert wurden. In den Rahmenbedingungen wird festgelegt, ob die Kontrollflusssemantik durch eine elementare oder zusammengesetzte Bedingung beschrieben wird. Eine zusammengesetzte Bedingung besteht aus zwei oder mehreren elementaren Bedingungen. Für jede elementare Bedingung müssen zusätzlich zum Kontrollflussmuster der Gültigkeitsbereich, die ein- bzw. ausgehende Kantenanzahl und der Kantentyp festgelegt werden. Zur Beschreibung der Kontrollflusssemantik des Kontrollflussmusters müssen die Parameter

der Kontrollflussmuster spezifiziert werden. Beispielsweise sind das die newYAWL-Parameter Split, $<_{XOR}$, ArcCond und Default für das Kontrollflussmuster exklusive Auswahl. Die Kontrollflusssemantik des Geschäftsprozessmodells wird durch die Spezifizierung der Parameter der Kontrollflussmuster der Semantikschemas präzise beschrieben. Das Vorgehen zur Anwendung dieser Methode wurde für ausgewählte Geschäftsprozessmodelle aus [DrKO17] und [RuQu12] demonstriert, die mit den Modellierungssprachen BPMN, EPK, UML-Aktivitätsdiagramm, Petri-Netze und YAWL bzw. newYAWL dokumentiert wurden.

Anschließend wurde eine empirische Untersuchung durchgeführt, um erste Anhaltspunkte für diverse Vermutungen zu erhalten. Zum einen wurde betrachtet, ob durch die Annotation der Elemente in den Geschäftsprozessmodellen mit den Kontrollflussmustern der Workflow Pattern Initiative die Verständlichkeit der Kontrollflusssemantik verbessert wird. Es wurden ebenfalls erste Erkenntnisse für die Vermutungen gesammelt, ob die Methode zur präzisen Beschreibung der Ablaufreihenfolge der Elemente eines Geschäftsprozessmodells einfach zu benutzen, leicht zu erlernen, zur einfachen Beschreibung der Kontrollflusssemantik beiträgt, nützlich für die Verständlichkeit und leicht zu verstehen ist. Besonderen Wert wurde dabei auf die Nachvollziehbarkeit der Ergebnisse gelegt, so dass die Untersuchung in Anlehnung an die fünf Phasen von [Maye13, Atte10, Diek17] durchgeführt wurde. Dementsprechend wurde eine Online-Umfrage entwickelt, an der 132 Personen teilgenommen haben. Mit der Umfrage wurden sieben Hypothesen überprüft. Das Ergebnis der Untersuchung ist, dass alle aufgestellten Hypothesen durch die Online-Umfrage gestützt werden können, jedoch sind detaillierte Untersuchungen für fundierte Schlussfolgerungen erforderlich. Das Ziel der empirischen Untersuchung war ausschließlich die Sammlung erster Anhaltspunkte für die aufgestellten Vermutungen. Eine prototypische Realisierung zur Beschreibung der Kontrollflusssemantik von Geschäftsprozessmodellierungssprachen und der Ablaufreihenfolge der Elemente eines Geschäftsprozessmodells rundet diese Arbeit ab. Hierdurch wurde die Möglichkeit zur Analyse von Geschäftsprozessmodellen geschaffen, die am Beispiel einer interaktiven Simulation von Geschäftsprozessmodellen demonstriert wurde. In diesem Zusammenhang wurden zunächst Zustände für die einzelnen Knoten eines Geschäftsprozessmodells eingeführt und darauf aufbauend das generische Vorgehen einer interaktiven Simulation beschrieben.

8.2 Beitrag

Die entwickelte Methode liefert einen Beitrag für das Geschäftsprozessmanagement, wie anhand der einzelnen Phasen des Geschäftsprozessmanagements (vgl. Abbildung 3) in Kapitel 2 beschrieben wurde. Durch die Verknüpfung der Syntaxelemente einer

Geschäftsprozessmodellierungssprache mit Schablonen bzw. Kontrollflussmustern wird eine Beschreibung der Kontrollflusssemantik für Geschäftsprozessmodellierungssprachen sowie auch die Beschreibung der Ablaufreihenfolge der Elemente in Geschäftsprozessmodellen ermöglicht. Häufig wird in der Literatur zur präzisen Beschreibung der Kontrollflusssemantik einer Modellierungssprache eine Übersetzungssemantik oder operationelle Kontrollflusssemantik entwickelt. Die Modellierungssprache wird in diesem Zusammenhang typischerweise in ihrer Ausdrucksmächtigkeit zur Modellierung von Sachverhalten eingeschränkt, im Gegensatz zur hier vorgestellten Methode. Zudem muss keine von Grund auf neue operationelle, denotationelle oder axiomatische Kontrollflusssemantik entwickelt werden. Durch die entwickelte Methode können sich weitere Einsatzmöglichkeiten ergeben, die bisher in dieser Arbeit nicht berücksichtigt oder beachtet wurden. Es wurden die folgenden Beiträge entwickelt:

– Die Kontrollflusssemantik der Kontrollflussmuster der Workflow Pattern Initiative wurde unter Zuhilfenahme der Modellierungssprache YAWL bzw. newYAWL präzise beschrieben. Es wurden muster- und modellspezifischen Parameter aus der YAWL- bzw. newYAWL-Spezifikation zur Beschreibung der Kontrollflusssemantik der Kontrollflussmuster abgeleitet und um fehlende Parameter zur Beschreibung der Kontrollflusssemantik erweitert.

– Zudem wurde eine auf Kontrollflussmustern basierende Methode zur Beschreibung der Kontrollflusssemantik von Geschäftsprozessmodellierungssprachen entwickelt. Insbesondere adressiert die Beschreibung der Kontrollflusssemantik von Geschäftsprozessmodellierungssprachen domänenspezifische Modellierungssprachen und Standardmodellierungssprachen mit domänenspezifischen Erweiterungen. Die Kontrollflusssemantik einer standardisierten Modellierungssprache (wie z. B. BPMN, UML-AD, etc.) wird in einem Spezifikationsdokument beschrieben. Diese Beschreibung ist jedoch oftmals nur natürlich-sprachlich und enthält daher gewollte oder auch ungewollte Mehrdeutigkeiten. Um diesen Mangel zu beheben kann die entwickelte Methode auch für die Beschreibung der Kontrollflusssemantik von Standardmodellierungssprachen eingesetzt werden. Mit der beschriebenen Kontrollflusssemantik wird unter anderem eine Grundlage für die Austauschbarkeit der Notationselemente der Geschäftsprozessmodellierungssprache in Abhängigkeit der Anwendungsdomäne geschaffen. Beispielsweise kann durch andere Notationselemente die Verständlichkeit der einzelnen Elemente einer Geschäftsprozessmodellierungssprache verbessert werden. Aus diesem Grund erweitert die Methode unter anderem die Möglichkeiten der Designer von Geschäftsprozessmodellierungssprachen.

– Es wurde eine Methode zur präzisen Beschreibung der Ablaufreihenfolge von Elementen in Geschäftsprozessmodellen entwickelt, die auf der Methode zur Beschreibung der Kontrollflusssemantik von Geschäftsprozessmodellierungssprachen basiert. Auf diese Weise wird das Verständnis der Ablaufreihenfolge der Elemente eines Geschäftsprozessmodells verbessert. Die dokumentierten Geschäftsprozessmodelle können aufgrund dieser Methode mit den realen Geschäftsprozessen verglichen werden, zum Beispiel mit den Methoden des Process Minings. Darüber hinaus können die Zusammenhänge der Elemente auf Vollständigkeit und Korrektheit überprüft werden. Auf Grundlage der präzisen Kontrollflusssemantik könnten Gültigkeitsanfragen formuliert werden, um z. B. das Geschäftsprozessmodell auf Widerspruchsfreiheit, Redundanzfreiheit oder auch auf inhaltliche Aspekte hin überprüfen zu können. Die damit einhergehende Qualitätssicherung lässt Fehler vor der Implementierung erkennen, um Folgekosten durch erzwungene Eingriffe während der Laufzeit zu vermeiden. Es können aber auch IT-basierte Analyseverfahren eingesetzt werden, so dass z. B. quantitative Aussagen über die Mindestdauer, den Ressourcenverbrauch oder die Kapazitätsauslastung möglich werden. Während der Prozesskonzeption können mögliche Geschäftsprozessmodellalternativen miteinander verglichen werden. Aus diesem Grund erweitert die entwickelte Methode unter anderem die Möglichkeiten eines Modellierers von Geschäftsprozessmodellen, der beispielsweise die Methoden zur Analyse der Geschäftsprozessmodelle anwenden möchte. Darüber hinaus adressiert die entwickelte Methode auch die Software-Werkzeugbauer, die beispielsweise eine IT-gestützte Analyse für Geschäftsprozessmodelle ermöglichen möchten.

– Die Anwendbarkeit der Methode wurde demonstriert. Zum einen wurde die Kontrollflusssemantik der Geschäftsprozessmodellierungssprachen BPMN, EPK, UML-Aktivitätsdiagramm, Petri-Netze und YAWL bzw. newYAWL mit der entwickelten Methode definiert, so dass auf dieser Grundlage die Ablaufreihenfolge der Elemente von ausgewählten Geschäftsprozessmodellen aus [DrKO17] und [RuQu12] präzise beschrieben werden konnte. Zum anderen wurde die Anwendbarkeit der Methode mit einer Implementierung demonstriert, welche über eine interaktive Simulation verfügt. Auf dieser Grundlage werden Auswirkungsanalysen (z. B. für die Kennzahlen, Mindestdauer, Ressourcenverbrauch oder Kapazitätsauslastung) ermöglicht, deren Ergebnisse zur Verbesserung der Geschäftsprozesse (z. B. im Kontext der Produktqualität, Durchlaufzeiten und Kundenbedürfnissen) eingesetzt werden können.

– Im Rahmen einer empirischen Untersuchung wurden erste Erkenntnisse gesammelt, dass durch die Anwendung der entwickelten Methode die ungewollten Mehrdeutigkeiten einer Oder-Zusammenführung in Geschäftsprozessmodellen reduziert werden können. Dementsprechend kann entgegen der Empfehlungen der sieben Prozessmodellierungsrichtlinien (7PMG) von [MeRv10] eine Oder-Zusammenführung in Geschäftsprozessmodellen in Kombination mit der entwickelten Methode eingesetzt werden.

8.3 Grenzen

Die entwickelte Methode wurde auf graphische Geschäftsprozessmodellierungssprachen eingeschränkt, mit denen die Ablaufreihenfolge von Aktivitäten in Geschäftsprozessen beschrieben werden kann. Zum einen wurde eine Einschränkung auf Geschäftsprozessmodellierungssprachen vorgenommen sowie darüber hinaus gefordert, dass diese Sprachen über graphische Elemente bzw. Notationselemente verfügen müssen. Eine Übertragung auf andere Sprachen, wie Programmiersprachen, sollte möglich sein, da die Ablaufreihenfolge der Elemente mit den vorgestellten Geschäftsprozessmodellierungssprachen beschrieben werden kann und ein Notationselement zudem nur eine graphische Repräsentation eines Syntaxelementes ist. Ferner wird die Kontrollflusssemantik für die Syntaxelemente der Sprache definiert und nicht für die Notationselemente. Die Kontrollflusssemantik der Syntaxelemente wurde ausschließlich im entwickelten Prototyp über die Notationselemente definiert. Dennoch kann die Untergliederung der Syntaxelemente in Knoten- und Kantentypen insbesondere bei nicht graphischen Sprachen problematisch sein. Darüber hinaus bildet beispielsweise das BPMN-Konversationsdiagramm auch eine Ausnahme, da mit dem Konversationsdiagramm nur eine informale Darstellung des Nachrichtenaustausches beschrieben wird und keine Ablaufreihenfolge von Elementen dargestellt werden soll, wie auch mit den Pools in BPMN. Es wird bei der entwickelten Methode nur die Kontrollflussperspektive für Geschäftsprozessmodellierungssprachen und die Ablaufreihenfolge der Elemente in Geschäftsprozessmodellen betrachtet. Andere Perspektiven, wie beispielsweise die Ressourcen- [RAHE05] oder Datenperspektive [RHEA05] wurden nicht betrachtet, wie es beispielsweise für die vollständige Beschreibung der Kontrollflusssemantik der UML-Aktivitätsdiagramme erforderlich wäre.

Die Methode zur Beschreibung der Kontrollflusssemantik von Geschäftsprozessmodellierungssprachen basiert auf der Grundannahme, dass die Kontrollflusssemantik durch ein Semantikschema (vgl. Abbildung 30) beschrieben werden kann, welches aus der Kontrollflusssemantik «Vor der Ausführung», «Während der Ausführung» und «Nach der

Ausführung» sowie einer beliebigen Anzahl von Beziehungen besteht. Aus der Anwendung der entwickelten Methode auf die Geschäftsprozessmodellierungssprache BPMN wurde eine Grenze aufgezeigt, da mit existierenden Kontrollflussmustern die Kontrollflusssemantik der BPMN nicht vollständig beschrieben werden konnte. Bei Szenarien, bei denen die Kontrollflusssemantik nicht mit den vorhandenen Kontrollflussmustern beschrieben werden kann, muss die Methode um die entsprechenden Kontrollflussmuster erweitert werden, damit die spezifische Kontrollflusssemantik abgebildet werden kann. Für die Modellierungssprache BPMN wurden vier individuelle Kontrollflussmuster entwickelt.

Bei der Entwicklung der Methode wurde angenommen, dass ein Geschäftsprozessmodell nur genau einen Startknoten besitzen darf und ist somit eine Grundannahme, um die entwickelte Methode zur präzisen Beschreibung der Ablaufreihenfolge der Elemente von Geschäftsprozessmodellen anwenden zu können. Es existiert demnach auch nur ein Kontrollflussmuster zur Beschreibung der Kontrollflusssemantik eines Startknotens im Geschäftsprozessmodell. Bei der Existenz von mehreren Startknoten in einem Geschäftsprozessmodell werden Kontrollflussmuster benötigt, um Abhängigkeiten zwischen diesen Startknoten formulieren zu können. Zum Beispiel, wenn der Prozess an einem Startknoten instanziiert wurde, dass die anderen Startknoten für diese Geschäftsprozessinstanz nicht mehr ausgeführt werden können. Unter anderem wurde dieser Aspekt bereits bei dem Prototyp berücksichtigt. Darüber hinaus wurde in der entwickelten Methode auch nicht vorgesehen, dass für die Ausführung des Geschäftsprozessmodells Input- oder Output-Bedingungen erforderlich sein können, wie beispielsweise bei der Modellierungssprache YAWL bzw. newYAWL und UML-Aktivitätsdiagramm. Für die Abschwächung der Anforderung müssten entsprechende Kontrollflussmuster entwickelt werden, um Abhängigkeiten zu anderen Startknoten, Input- und Output-Bedingungen sowie Vor- und Nachbedingungen für einzelne Knoten in den Geschäftsprozessmodellen formulieren zu können.

Die Anwendung der entwickelten Methode ist durch die Modellierungssprachen BPMN, EPK, UML-Aktivitätsdiagramm, Petri-Netze und YAWL bzw. newYAWL demonstriert worden, wohingegen die Akzeptanz und die Verständlichkeit der Methode nicht umfassend überprüft wurde. So wurde beispielsweise die Art der Darstellung der zugeordneten Kontrollflussmuster zu den Knoten im Geschäftsprozessmodell nicht betrachtet. Die entwickelte Methode kann nicht das generelle Problem der Mehrdeutigkeiten von Notationselementen lösen, da diese Aspekte auf die Konzeption der Modellierungssprache zurückzuführen sind. Dennoch können die ungewollten Mehrdeutigkeiten durch die Annotation der Elemente des Geschäftsprozessmodells mit Kontrollflussmustern entfernt werden, wie in der Evaluation in Kapitel 7 gezeigt wurde. Zudem wurde auch nicht der Frage nachgegangen, ob der Aufwand der Anwendung der Methode in einem geeigneten Verhältnis zum Nutzen steht. Beispielsweise müssen für die Geschäftsprozess-

modellierungssprache BPMN mehrere tausend elementare Bedingungen und Semantikschemas definiert werden, um die Kontrollflusssemantik der Modellierungssprache beschreiben zu können. Für andere Geschäftsprozessmodellierungssprachen, wie den Petri-Netzen, müssen beispielsweise nur 7 Bedingungen und 15 Semantikschemas definiert werden. Die praktische Anwendung zur Beschreibung der Kontrollflusssemantik einer Geschäftsprozessmodellierungssprache und der damit erstellten Geschäftsprozessprozessmodelle wurde ebenfalls nicht untersucht. In der empirischen Untersuchung wurden lediglich erste Anhaltspunkte für eine bessere Verständlichkeit einer Oder-Zusammenführung durch die Annotation mit Kontrollflussmustern gesammelt. Darüber hinaus wurden erste Erkenntnisse gesammelt, ob die Methode zur präzisen Beschreibung der Kontrollflusssemantik einfach benutzbar, leicht erlernbar, nützlich, leicht verständlich und eine einfache Beschreibung der Kontrollflusssemantik ermöglicht.

Mit den entwickelten Methoden kann zum einen die Kontrollflusssemantik einer Geschäftsprozessmodellierungssprache beschrieben sowie zum anderen die Ablaufreihenfolge der Elemente in Geschäftsprozessmodellen definiert werden. In dem entwickelten Software-Werkzeug wurden zur interaktiven Simulation Zustände eingeführt, die mit den möglichen Zuständen eines Bedingungs-/Ereignis-Systems (vgl. Kapitel 3.5) vergleichbar sind. Eine Verallgemeinerung, wie zum Beispiel zur interaktiven Simulation eines Stellen/Transitions-Systems, wird nicht geliefert, so dass entsprechend Kapazitäten und Kantengewichte eingeführt werden müssten. Dies würde beispielsweise auch die interaktive Simulation der verschiedenen Datenobjekte mit Kapazitäten in einem UML-Aktivitätsdiagramm ermöglichen.

8.4 Ausblick

Die entwickelte Methode und das Software-Werkzeug sind die Grundlage für weiterführende Anwendungen und praxisrelevante Erweiterungen. Insbesondere im Kontext des Geschäftsprozessmanagements bildet die Methode eine wichtige Grundlage, um analysefähige Geschäftsprozessmodelle zu schaffen (z. B. in Form einer Simulation), die die Überwachung von Geschäftsprozessinstanzen mittels einer Workflow Engine ermöglichen, wodurch die dokumentieren Geschäftsprozesse verbessert werden können (z. B. im Kontext von Kosten, Zeit und Ressourcen). Aus diesem Grund können die tatsächlichen Gegebenheiten dokumentiert werden, um darauf aufbauend die weiterführenden Methoden des Geschäftsprozessmanagements anwenden zu können. Neben der interaktiven Simulation von Geschäftsprozessmodellen können weitere Analyse- und Überwachungsmethoden entwickelt werden. Beispielsweise könnten die realen Geschäftsprozesse mit den

dokumentierten Geschäftsprozessmodellen verglichen oder auch die Zusammenhänge zwischen Elementen im Geschäftsprozessmodell auf Vollständigkeit und Korrektheit überprüft werden.

Mit der entwickelten Methode wird die Kontrollflusssemantik einer Geschäftsprozessmodellierungssprache beschrieben, so dass sich als weitere Fragestellung anbietet, ob dem Anwender der Methode nicht ein Repository zur Verfügung gestellt werden sollte. Das Repository würde die Beschreibung der Kontrollflusssemantik von verschiedenen Geschäftsprozessmodellierungssprachen beinhalten, so dass beispielsweise ein Semantikschema aus einer Geschäftsprozessmodellierungssprache auf eine andere Modellierungssprache übertragen bzw. die Kontrollflusssemantik einer anderen Geschäftsprozessmodellierungssprache damit beschrieben werden kann. Darüber hinaus könnten Vorschlagsmechanismen entwickelt werden, um anhand der zugrundeliegenden Semantikschemas ein entsprechendes Semantikschema vorzuschlagen, damit die Kontrollflusssemantik von Knotentypen beschrieben werden kann. Mit dieser Erweiterung müssten nicht immer neue Semantikschemas zur Beschreibung der Kontrollflusssemantik definiert werden, sondern es können vorhandene Semantikschemas verwendet werden.

Die zugrundeliegenden Kontrollflussmuster zur Beschreibung der Kontrollflusssemantik wurden nicht auf Vollständigkeit überprüft. Wie bereits im vorherigen Abschnitt bei den Grenzen aufgezeigt wurde, ist davon auszugehen, dass die Liste der definierten Kontrollflussmuster nicht abschließend ist. Es wurde beispielsweise ein Kontrollflussmuster für die Beschreibung der Kontrollflusssemantik von angehefteten nicht-unterbrechenden Ereignissen an BPMN-Aktivitäten eingeführt, sobald das Ereignis eintritt wird die Aufgabe unterbrochen. Sofern zwei Ereignisse eingetreten sein müssen, um die BPMN-Aufgabe zu unterbrechen, kann dies ohne eine Erweiterung der Methode oder das Hinzufügen von Kontrollflussmustern nicht abgebildet werden. In diesem Zusammenhang bietet sich eine empirische Untersuchung an, die zum einen vorhandene Geschäftsprozessmodellierungssprachen identifiziert und zum anderen die identifizierten Geschäftsprozessmodellierungssprachen auf die beiden erforderlichen Anforderungen (vgl. Kapitel 5.1) hin prüft. Sofern beispielsweise weitere Kontrollflussmuster benötigt werden, sollten diese definiert werden, wie dies auch anhand der vier benutzerdefinierten Kontrollflussmuster für BPMN in dieser Arbeit demonstriert wurde. Durch die Untersuchung würde zudem eine umfassende Qualitätssicherung vorgenommen werden, da die entwickelte Methode nur auf die ausgewählten Geschäftsprozessmodellierungssprachen BPMN, EPK, UML-Aktivitätsdiagramm, Petri-Netz und YAWL bzw. newYAWL angewendet wurde. Eine Überprüfung der entwickelten Methoden unter realen Bedingungen und einer damit einhergehenden Beobachtung ist anzustreben. Dies könnte beispielsweise im Rahmen eines Usability Tests durchgeführt werden, indem die Anwender der Methode bei der Anwendung beobachtet werden.

In der durchgeführten empirischen Untersuchung konnten zwar erste Erkenntnisse gesammelt werden, die gleichwohl aber stark von der Teilnehmergruppe der Studierenden beeinflusst sind. Eine erneute Durchführung der empirischen Untersuchung mit dem überarbeiteten Informationsvideo und anderen Teilnehmern, insbesondere einem größeren Teilnehmerkreis aus Unternehmen und eine detaillierte Betrachtung der einzelnen Vermutungen sowie die Korrelation zwischen den einzelnen Vermutungen, könnte zu einem grundlegend anderen Ergebnis führen. Darüber hinaus könnte unter anderem in weiteren Schritten die Art der Darstellung der zugeordneten Kontrollflussmuster zu den Knoten im Geschäftsprozessmodell untersucht sowie das Kosten-Nutzenverhältnis für die Anwendung der Methode evaluiert werden.

9 Literaturverzeichnis

[AaHo05] Aalst, W. M. P. v. d.; Hofstede, A. H. M. t.: YAWL: Yet Another Workflow Language. In Information Systems, 2005, 30; S. 245–275.

[AaHo12] Aalst, W. M. P. v. d.; Hofstede, A. H. M. t.: Workflow patterns put into context. In Software and System Modeling, 2012, 11; S. 319–323.

[Aals07] Aalst, W. M. P. v. d.: Challenges in Business Process Analysis. In (Filipe, J.; Cordeiro, J.; Cardoso, J. Hrsg.): International Conference on Enterprise Information Systems (ICEIS). Springer, Berlin, Heidelberg, 2007; S. 27–42.

[Aals10] Aalst, W. M. P. v. d.: Business Process Simulation Revisited. Workshop. In (Barjis, J. Hrsg.): Enterprise and Organizational Modeling and Simulation (EOMAS). Springer, Berlin, Heidelberg, 2010; S. 1–14.

[Aals12] Aalst, W. M. P. v. d.: A Decade of Business Process Management Conferences. Personal Reflections on a Developing Discipline. In (Barros, A. P.; Gal, A.; Kindler, E. Hrsg.): Business Process Management (BPM). Springer, Berlin, Heidelberg, 2012; S. 1–16.

[Aals13] Aalst, W. M. P. v. d.: Business Process Management. A Comprehensive Survey. In International Scholarly Research Notices (ISRN) Software Engineering, 2013; S. 37.

[Aals16] Aalst, W. M. P. v. d.: Process Mining. Data Science in Action. Springer, Berlin, Heidelberg, 2016.

[Aals97] Aalst, W. M. P. v. d.: Verification of Workflow Nets. In (Azéma, P.; Balbo, G. Hrsg.): International Conference on Applications and Theory of Petri Nets and Concurrency (ICATPN). Springer, Berlin, Heidelberg, 1997; S. 407–426.

[Aals99] Aalst, W. M. P. v. d.: Formalization and Verification of Event-driven Process Chains. In Information and Software Technology, 1999, 41; S. 639–650.

[AAMA+12] Aalst, W. M. P. v. d. et al.: Process Mining Manifesto. In (Daniel, F.; Barkaoui, K.; Dustdar, S. Hrsg.): Business Process Management (BPM) Workshops. Springer, Berlin, Heidelberg, 2012; S. 169–194.

[AaSt11] Aalst, W. M. P. v. d.; Stahl, C.: Modeling Business Processes. A Petri Net-Oriented Approach. MIT Press, Cambridge, Massachusetts, 2011.

[AaWM04] Aalst, W. M. P. v. d.; Weijters, T.; Maruster, L.: Workflow Mining. Discovering Process Models from Event Logs. In IEEE Transactions on Knowledge and Data Engineering, 2004, 16; S. 1128–1142.

[AcUA12] Accorsi, R.; Ullrich, M.; Aalst, W. M. P. v. d.: Process Mining. In Informatik Spektrum, 2012, 35; S. 354–359.

[Agui04] Aguilar-Savén, R. S.: Business Process Modelling. Review and Framework. In International Journal of Production Economics, 2004, 90; S. 129–149.

[AHFL12] Aubertin, I. et al.: Stand der Lehrbuchliteratur zum Geschäftsprozessmanagement. Eine quantitative Analyse, Saarbrücken, 2012.

[AHKB03] Aalst, W. M. P. v. d. et al.: Workflow Patterns. In Distributed and Parallel Databases, 2003, 14; S. 5–51.

[AISJ77] Alexander, C. et al.: A Pattern Language. Towns, Buildings, Construction. Oxford University Press (OUP), New York, 1977.

[Allw15] Allweyer, T.: BPMN 2.0 - Business Process Model and Notation. Einführung in den Standard für die Geschäftsprozessmodellierung. Books on Demand, Norderstedt, 2015.

[AlSc17] Algermissen, L.; Schwall, J.: Prozessveränderung leicht gemacht. In (Hinz, E. Hrsg.): Regieren in Kommunen. Herausforderungen besser bewältigen - Außen- und Binnenorientierung beeinflussen. Springer, Wiesbaden, 2017; S. 169–182.

[AnRa08] Anderl, R.; Raßler, J.: PML, an Object Oriented Process Modeling Language. In (Cascini, G. Hrsg.): Computer-Aided Innovation (CAI). Springer, Bosten, 2008; S. 145–156.

[Atte10] Atteslander, P.: Methoden der empirischen Sozialforschung. Erich Schmidt, Berlin, 2010.

[BaBl03] Basu, A.; Blanning, R. W.: Synthesis and Decomposition of Processes in Organizations. In Information Systems Research, 2003, 14; S. 337–355.

[BaKr96] Bause, F.; Kritzinger, P. S.: Stochastic Petri Nets. An Introduction to the Theory. Vieweg, Wiesbaden, 1996.

[Baum92] Baumann, P.: Software-Bewertung. Ein semantischer Ansatz für Informationsmaße. Springer, Berlin, Heidelberg, 1992.

[BaWe90] Baeten, J. C. M.; Weijland, W. P.: Process Algebra. Cambridge University Press, Cambridge, 1990.

[BBSG+16] Beheshti, S. M. R. et al.: Process Analytics. Concepts and Techniques for Querying and Analyzing Process Data, 2016.

[BePR07a] Becker, J.; Pfeiffer, D.; Räckers, M.: Domain Specific Process Modelling in Public Administrations. The PICTURE-Approach. In (Wimmer, M. A.; Scholl, H. J.; Grönlund, Å. Hrsg.): Electronic Government (EGOV). Springer, Berlin, Heidelberg, 2007; S. 68–79.

[BePR07b] Becker, J.; Pfeiffer, D.; Räckers, M.: PICTURE. A new Approach for Domain-Specific Process Modelling. In (Eder, J. et al. Hrsg.): Conference on Advanced Information Systems Engineering (CAiSE) Forum. CEUR-WS.org, Aachen, 2007; S. 45–48.

[BePV12] Becker, J.; Probandt, W.; Vering, O.: Grundsätze ordnungsmäßiger Modellierung. Konzeption und Praxisbeispiel für ein effizientes Prozessmanagement. Springer, Berlin, Heidelberg, 2012.

[BePZ11] Bentakouk, L.; Poizat, P.; Zaïdi, F.: Checking the Behavioral Conformance of Web Services with Symbolic Testing and an SMT Solver. In (Gogolla, M.; Wolff, B. Hrsg.): Tests and Proofs (TAP). Springer, Berlin, Heidelberg, 2011; S. 33–50.

[BFGL94] Bandinelli, S. et al.: SPADE. An Environment for Software Process Analysis, Design, and Enactment. In (Nuseibeh, B.; Finkelstein, A.; Kramer, J. Hrsg.): Software Process Modelling and Technology. Research Studies Press, Taunton, 1994; S. 223–247.

[BiBK02] Bieger, T.; Bickhoff, N.; Knyphausen-Aufseß, D. z.: Einleitung. In (Bieger, T. et al. Hrsg.): Zukünftige Geschäftsmodelle. Konzept und Anwendung in der Netzökonomie. Springer, Berlin, Heidelberg, 2002; S. 1–15.

[BKFL11] Balzert, S. et al.: Vorgehensmodelle im Geschäftsprozessmanagement. Operationalisierbarkeit von Methoden zur Prozesserhebung, Saarbrücken, 2011.

[BöCR00] Börger, E.; Cavarra, A.; Riccobene, E.: An ASM Semantics for UML Activity Diagrams. In (Rus, T. Hrsg.): Algebraic Methodology and Software Technology (AMAST). Springer, Berlin, Heidelberg, 2000; S. 293–308.

[Boeh79] Boehm, B. W.: Software Engineering. As it is. In (Bauer, F. L.; Stucki, L. G.; Lehman, M. M. Hrsg.): International Conference on Software Engineering (ICSE). IEEE Computer Society, Los Alamitos, 1979; S. 11–21.

[Boeh81] Boehm, B. W.: Software Engineering Economics. Prentice-Hall, Englewood Cliffs, NJ, 1981.

[Booc91] Booch, G.: Object Oriented Design with Applications. Benjamin-Cummings, Redwood City, 1991.

[Börg07] Börger, E.: Modeling Workflow Patterns from First Principles. In (Parent, C. et al. Hrsg.): Conceptual Modeling (ER). Springer, Berlin, Heidelberg, 2007; S. 1–20.

[Börg12] Börger, E.: Approaches to modeling business processes. A critical analysis of BPMN, Workflow Patterns and YAWL. In Software and System Modeling, 2012, 11; S. 305–318.

[BöSö11] Börger, E.; Sörensen, O.: BPMN Core Modeling Concepts. Inheritance-Based Execution Semantics. In (Embley, D. W.; Thalheim, B. Hrsg.): Handbook of Conceptual Modeling. Theory, Practice and Research Challenges. Springer, Berlin, Heidelberg, 2011; S. 287–334.

[BöTh08a] Börger, E.; Thalheim, B.: Modeling Workflows, Interaction Patterns, Web Services and Business Processes. The ASM-Based Approach. In (Börger, E. et al. Hrsg.): Abstract State Machines, B and Z. Springer, Berlin, Heidelberg, 2008; S. 24–38.

[BöTh08b] Börger, E.; Thalheim, B.: A Method for Verifiable and Validatable Business Process Modeling. In (Börger, E.; Antonio, C. Hrsg.): Advances in Software Engineering. Springer, Berlin, Heidelberg, 2008; S. 59–115.

[BPMI04] Business Process Management Initiative (BPMI): Business Process Modeling Notation (BPMN). http://www.omg.org/bpmn/Documents/BPMN_V1-0_May_3_2004.pdf.

[BPML0.4] Arkin, A.: Business Process Modeling Language (BPML). Specification. http://xml.coverpages.org/bpml.html, 04.09.2018.

[BPMN2.0.2] Object Management Group (OMG): Business Process Model and Notation (BPMN). www.omg.org/spec/BPMN/2.0.2/PDF/, 04.09.2018.

[BPRF+15] Becker, J. et al.: Semantic Business Process Modelling and Analysis. In (Brocke, J. v.; Rosemann, M. Hrsg.): Handbook on Business Process Management 1. Introduction, Methods and Information Systems. Springer, Berlin, Heidelberg, 2015; S. 187–217.

[BPTO+04] Boxwala, A. A. et al.: Guideline Interchange Format (GILF) 3. A representation format for sharable computer-interpretable clinical practice guidelines. In Biomedical Informatics, 2004, 37; S. 147–161.

[BrØs06] Brahe, S.; Østerbye, K.: Business Process Modeling. Defining Domain Specific Modeling Languages by Use of UML Profiles. In (Rensink, A.; Warmer, J. Hrsg.): Model Driven Architecture. Foundations and Applications (ECMDA-FA). Springer, Berlin, Heidelberg, 2006; S. 241–255.

[BSBE14] Braun, R. et al.: BPMN4CP. Design and Implementation of a BPMN Extension for Clinical Pathways. In (Zheng, H.; Cho, K.-H.; Wang, Y. Hrsg.): Bioinformatics and Biomedicine (BIBM). IEEE Computer Society, Los Alamitos, 2014; S. 9–16.

[BSBE15] Braun, R. et al.: Extending a Business Process Modeling Language for Domain-Specific Adaptation in Healthcare. In (Thomas, O.; Teuteberg, F. Hrsg.): Smart Enterprise Engineering. Wirtschaftsinformatik (WI). Universität Osnabrück, Osnabrück, 2015; S. 468–481.

[BSBE16] Braun, R. et al.: BPMN4CP Revised. Extending BPMN for Multi-perspective Modeling of Clinical Pathways. In (Bui, T. X.; Sprague, R. H., JR. Hrsg.): Hawaii International Conference on System Sciences (HICSS). IEEE Computer Society, Los Alamitos, 2016; S. 3249–3258.

[Burk99] Burkhardt, R.: UML - Unified Modeling Language. Objektorientierte Modellierung für die Praxis. Addison-Wesley, Bonn, 1999.

[CaMe12] Capel, M. I.; Mendoza, L. E.: Automating the Transformation from BPMN Models to CSP+T Specifications. In (Bowen, J. P.; Zhu, H.; Hinchey, M. Hrsg.): Software Engineering Workshop (SEW). IEEE Computer Society, Los Alamitos, 2012; S. 100–109.

[CeFB00] Ceri, S.; Fraternali, P.; Bongio, A.: Web Modeling Language (WebML). A Modeling Language for Designing Web Sites. In Computer Networks, 2000, 33; S. 137–157.

[CeMa99] Cerone, A.; Maggiolo-Schettinim, A.: Time-Based Expressivity of Time Petri Nets for System Specification. In Theoretical Computer Science, 1999, 216; S. 1–53.

[Chan06] Chang, J. F.: Business Process Management Systems. Strategy and Implementation. Auerbach Publications, Boca Raton, 2006.

[Chap70] Chapin, N.: Flowcharting with the ANSI Standard. A Tutorial. In Computing Surveys (CSUR), 1970, 2; S. 119–146.

[ChCH11] Christiansen, D. R.; Carbone, M.; Hildebrandt, T.: Formal Semantics and Implementation of BPMN 2.0 Inclusive Gateways. In (Bravetti, M.; Bultan, T. Hrsg.): Web Services and Formal Methods (WS-FM). Springer, Berlin, Heidelberg, 2011; S. 146–160.

[Chen02] Chen, P. P.: Entity-Relationship Modeling. Historical Events, Future Trends, and Lessons Learned. In (Broy, M.; Denert, E. Hrsg.): Software Pioneers. Contributions to Software Engineering. Springer, Berlin, Heidelberg, 2002; S. 297–310.

[Chen76] Chen, P. P.: The Entity-Relationship Model. Toward a Unified View of Data. In Transactions on Database Systems, 1976, 1; S. 9–36.

[ChMM10] Charfi, A.; Müller, H.; Mezini, M.: Aspect-Oriented Business Process Modeling with AO4BPMN. In (Kühne, T. et al. Hrsg.): European Conference Modelling Foundations and Applications (ECMFA). Springer, Berlin, Heidelberg, 2010; S. 48–61.

[ChSc94] Chen, R.; Scheer, A.-W.: Modellierung von Prozeßketten mittels Petri-Netz-Theorie, 1994.

[CJMN+92] Conradi, R. et al.: Design, Use and Implementation of SPELL. A language for Software Process Modelling and Evolution. In (Derniame, J.-C. Hrsg.):

European Workshop Software Process Technology (EWSPT). Springer, Berlin, Heidelberg, 1992; S. 167–177.

[CMMN1.1] Object Management Group (OMG): Case Management Model and Notation (CMMN). http://www.omg.org/spec/CMMN/1.1/PDF/, 08.09.2018.

[CoPR15] Corradini, F.; Polini, A.; Re, B.: Inter-Organizational Business Process Verification in Public Administration. In Business Process Management Journal (BPMJ), 2015, 21; S. 1040–1065.

[CPPR10] Corradini, F. et al.: Business Processes Verification for e-Government Service Delivery. In Information Systems Management, 2010, 27; S. 293–308.

[CPRT15] Corradini, F. et al.: An Operational Semantics of BPMN Collaboration. In (Braga, C.; Ölveczky, P. C. Hrsg.): Formal Aspects of Component Software (FACS). Springer, Berlin, Heidelberg, 2015; S. 161–180.

[DaEA98] Dami, S.; Estublier, J.; Amiour, M.: Apel. A Graphical Yet Executable Formalism for Process Modeling. In Automated Software Engineering, 1998, 5; S. 61–96.

[Dave93] Davenport, T. H.: Process Innovation. Reengineering Work through Information Technology. Harvard Business School Press, Boston, 1993.

[DDDG08] Decker, G. et al.: Transforming BPMN Diagrams into YAWL Nets. In (Dumas, M.; Reichert, M.; Shan, M. C. Hrsg.): Business Process Management (BPM). Springer, Berlin, Heidelberg, 2008; S. 386–389.

[Dehn03] Dehnert, J.: Four Systematic Steps Towards Sound Business Process Models. In (Ehrig, H. et al. Hrsg.): Petri Net Technology for Communication-Based Systems. Advances in Petri Nets. Springer, Berlin, Heidelberg, 2003; S. 66–82.

[DeRe98] Desel, J.; Reisig, W.: Place/Transition Petri Nets. In (Reisig, W.; Rozenberg, G. Hrsg.): Lectures on Petri Nets I. Basic Models. Advances in Petri Nets. Springer, Berlin, Heidelberg, 1998; S. 122–173.

[DeRi01] Dehnert, J.; Rittgen, P.: Relaxed Soundness of Business Processes. In (Dittrich, K. R.; Geppert, A.; Norrie, M. C. Hrsg.): Conference on Advanced Information Systems Engineering (CAiSE). Springer, Berlin, Heidelberg, 2001; S. 157–170.

[DiDO08] Dijkman, R. M.; Dumas, M.; Ouyang, C.: Semantics and Analysis of Business Process Models in BPMN. In Information and Software Technology, 2008, 50; S. 1281–1294.

[Diek17] Diekmann, A.: Empirische Sozialforschung. Grundlagen, Methoden, Anwendungen. Rowohlt, Reinbek, 2017.

[DLMM+12] Dumas, M. et al.: Understanding Business Process Models. The Costs and Benefits of Structuredness. In (Ralyté, J. et al. Hrsg.): Conference on Advanced Information Systems Engineering (CAiSE). Springer, Berlin, Heidelberg, 2012; S. 31–46.

[DMN1.1] Object Management Group (OMG): Decision Model and Notation (DMN). http://www.omg.org/spec/DMN/1.1/, 05.09.2016.

[Dole14] Doleski, O. D.: Integriertes Geschäftsmodell. Anwendung des St. Galler Management-Konzepts im Geschäftsmodellkontext. Springer Gabler, Wiesbaden, 2014.

[Dres16a] Drescher, A.: Business Process Model and Notation (BPMN). Ein Ansatz zur systematischen Untersuchung von BPMN-Modellen. AV Akademikerverlag, Saarbrücken, 2016.

[Dres16b] Drescher, A.: Modellierungssprachenunabhängige IT-basierte Geschäftsprozessanalyse. In (Nissen, V. et al. Hrsg.): Multikonferenz Wirtschaftsinformatik (MKWI). Universitätsverlag Illmenau, Ilmenau, 2016; S. 1131–1142.

[Dres16c] Drescher, A.: Modellierungssprachenunabhängige Anwendung des Geschäftsprozessmanagements. In (Betz, S.; Reimer, U. Hrsg.): Modellierung. Workshopband. Köllen, Bonn, 2016; S. 79–84.

[Dres18] Drescher, A.: Eine musterbasierte Kontrollflusssemantik zur interaktiven Simulation von Geschäftsprozessmodellen. In (Schaefer, I. et al. Hrsg.): Modellierung. Köllen, Bonn, 2018; S. 315–319.

[DrKO17] Drescher, A.; Koschmider, A.; Oberweis, A.: Modellierung und Analyse von Geschäftsprozessen. Grundlagen, Klausuren mit Lösungen. De Gruyter, Oldenbourg, 2017.

[ebXML2.0.4] Organization for the Advancement of Structured Information Standards (OASIS): Electronic Business using Extensible Markup Language (ebXML). Business Process Specification Schema. http://docs.oasis-open.org/ebxml-bp/2.0.4/OS/spec/ebxmlbp-v2.0.4-Spec-os-en.pdf, 04.09.2018.

[EDOC04] Object Management Group (OMG): Enterprise Distributed Object Computing (EDOC). https://www.omg.org/spec/EDOC, 05.09.2018.

[ElBo14] El-Saber, N. A. S.; Boronat, A.: BPMN Formalization and Verification using Maude. In (Roubtsova, E. et al. Hrsg.): Behaviour Modelling - Foundations and Applications (BM-FA). Workshop. Association for Computing Machinery (ACM), New York, 2014; S. 1–12.

[EmGr91] Emmerich, W.; Gruhn, V.: FUNSOFT nets. A Petri-Net based Software Process Modeling Language. In (Institute of Electrical and Electronics

Engineers (IEEE) Hrsg.): Workshop on Software Specification and Design. IEEE Computer Society, Los Alamitos, 1991; S. 175–184.

[Eshu02] Eshuis, R.: Semantics and Verfication of UML Activity Diagrams for Workflow Modelling. Doktorarbeit, Twente, 2002.

[FaBF12] Fares, E.; Bodeveix, J.-P.; Filali, M.: Design of a BPEL Verification Tool. In (Carbone, M.; Petit, J. M. Hrsg.): Web Services and Formal Methods (WS-FM). Springer, Berlin, Heidelberg, 2012; S. 95–110.

[Fehr89] Fehr, E.: Semantik von Programmiersprachen. Springer, Berlin, Heidelberg, 1989.

[FeSi95] Ferstl, O. K.; Sinz, E. J.: Der Ansatz des Semantischen Objektmodells (SOM) zur Modellierung von Geschäftsprozessen. In Wirtschaftsinformatik, 1995, 37; S. 209–220.

[FeZa14] Fellmann, M.; Zasada, A.: State-of-the-Art of Business Process Compliance Approaches. In (Avital, M.; Leimeister, J. M.; Schultze, U. Hrsg.): European Conference on Information Systems (ECIS). Association for Information Systems (AIS), Atlanta, 2014.

[FeZa16] Fellmann, M.; Zasada, A.: State-of-the-Art of Business Process Compliance Approaches. A Survey (Extended Abstract). In (Mendling, J.; Rinderle-Ma, S. Hrsg.): Enterprise Modeling and Information Systems Architectures (EMISA). Workshop. CEUR-WS.org, Aachen, 2016; S. 60–63.

[Fied14] Fiedler, R.: Organisation kompakt. De Gruyter, München, 2014.

[FiKa13] Fill, H.-G.; Karagiannis, D.: On the Conceptualisation of Modelling Methods Using the ADOxx Meta Modelling Platform. In Enterprise Modeling and Information Systems Architectures (EMISA), 2013, 8; S. 4–25.

[FiLa11] Figl, K.; Laue, R.: Cognitive Complexity in Business Process Modeling. In (Mouratidis, H.; Rolland, C. Hrsg.): Conference on Advanced Information Systems Engineering (CAiSE). Springer, Berlin, Heidelberg, 2011; S. 452–466.

[Fill11] Fill, H.-G.: Using Semantically Annotated Models for Supporting Business Process Benchmarking. In (Grabis, J.; Kirikova, M. Hrsg.): Perspectives in Business Informatics Research (BIR). Springer, Berlin, Heidelberg, 2011; S. 29–43.

[Fill12] Fill, H.-G.: An Approach for Analyzing the Effects of Risks on Business Processes Using Semantic Annotations. In (Pries-Heje, J. et al. Hrsg.): European Conference on Information Systems (ECIS). Association for Information Systems (AIS), Atlanta, 2012.

[Fowl96] Fowler, M.: Analysis Patterns. Reusable Object Models. Addison-Wesley, Amsterdam, 1996.

[Fran94] Frank, U.: Multiperspektivische Unternehmensmodellierung. Theoretischer Hintergrund und Entwurf einer objektorientierten Entwicklungsumgebung. Oldenbourg, München, Wien, 1994.

[FrRü17] Freund, J.; Rücker, B.: Praxishandbuch BPMN. Mit Einführung in CMMN und DMN. Hanser, München, 2017.

[Gait12] Gaitanides, M.: Prozessorganisation. Entwicklung, Ansätze und Programme des Managements von Geschäftsprozessen. Franz Vahlen, München, 2012.

[Gamm94] Gamma, E.: Design patterns. Elements of Reusable Object-Oriented Software. Addison-Wesley, Boston, 1994.

[GaSa79] Gane, C.; Sarson, T.: Structured Systems Analysis. Tools and Techniques. Prentice-Hall, Englewood Cliffs, 1979.

[GBFN16] Goby, N. et al.: Business Intelligence for Business Processes. The Case of IT Incident Management. In (Benbasat, I.; Bjørn-Andersen, N.; Sencer, A. Hrsg.): European Conference on Information Systems (ECIS). Association for Information Systems (AIS), Atlanta, 2016.

[GeKP98] Geisler, R.; Klar, M.; Pons, C.: Dimensions and Dichotomy in Metamodeling: Northern Formal Methods. British Computer Society, Swinton, 1998.

[GoDi13] Gorp, P. v.; Dijkman, R. M.: A visual token-based formalization of BPMN 2.0 based on in-place transformations. In Information & Software Technology, 2013, 55; S. 365–394.

[GoLa10] Gottschalk, F.; La Rosa, M.: Process Configuration. In (Hofstede, A. H. M. t. et al. Hrsg.): Modern Business Process Automation. YAWL and its Support Environment. Springer, Berlin, Heidelberg, 2010; S. 459–487.

[Gord79] Gordon, M. J. C.: The Denotational Description of Programming Languages. An Introduction. Springer, New York, 1979.

[GrLa07] Gruhn, V.; Laue, R.: What business process modelers can learn from programmers. In Science of Computer Programming, 2007, 65; S. 4–13.

[GrRu10] Grönniger, H.; Rumpe, B.: Systemmodell-basierte Definition objektbasierter Modellierungssprachen mit semantischen Variationspunkten. Shaker, Aachen, 2010.

[GrWe01] Gruhn, V.; Wellen, U.: Process Landscaping. Modelling Distributed Processes and Proving Properties of Distributed Process Models. In (Ehrig, H. et al. Hrsg.): Unifying Petri Nets. Advances in Petri Nets. Springer, Berlin, Heidelberg, 2001; S. 103–125.

[HAAR10] Hofstede, A.H.M.t. et al. Hrsg.: Modern Business Process Automation. YAWL and its Support Environment. Springer, Berlin, Heidelberg, 2010.

[HaCh93] Hammer, M.; Champy, J.: Reengineering the Corporation. A Manifesto for Business Revolution. Harvard Business School, New York, 1993.

[HäFe15] Hänel, T.; Felden, C.: An Empirical Investigation of Operational Business Intelligence Perspectives to Support an Analysis and Control of Business Processes. In (Bui, T. X.; Sprague, R. H. Hrsg.): Hawaii International Conference on System Sciences (HICSS). IEEE Computer Society, Los Alamitos, 2015; S. 4722–4731.

[Hall02] Halle, B. v.: Business Rules Applied. Business Better Systems Using the Business Rules Approach. Wiley Computer, New York, 2002.

[Harm16] Harmon, P.: The State of Business Process Managamenet. http://www.bptrends.com/bpt/wp-content/uploads/2015-BPT-Survey-Report.pdf, 06.09.2018.

[Hask22] Haskell, A. C.: Graphic Charts in Business. How to make and use them. Codex Book, New York, 1922.

[HBHK+08] Heinrich, B. et al.: SEMPA. A Semantic Business Process Management Approach for the Planning of Process Models. In Wirtschaftsinformatik, 2008, 50; S. 445–460.

[Hedt12] Hedtstück, U.: Einführung in die theoretische Informatik. Formale Sprachen und Automatentheorie. Oldenbourg, München, 2012.

[HeÖs04] Hess, T.; Österle, H.: Methoden des Business Process Redesign. Aktueller Stand und Entwicklungsperspektiven. In (Österle, H. et al. Hrsg.): Business Engineering. Die ersten 15 Jahre. Springer, Berlin, Heidelberg, 2004; S. 151–170.

[Herb14] Herbert, L.: Specification, Verification and Optimisation of Business Processes. A Unified Framework. Dissertation. Technical University of Denmark, Kongens Lyngby, 2014.

[Hill98] Hill, R.: What sample size is “enough” in internet survey research. In Interpersonal Computing and Technology Journal, 1998, 6; S. 1–10.

[HKKR05] Hitz, M. et al.: UML@Work. Objektorientierte Modellierung mit UML2. dpunkt, Heidelberg, 2005.

[HKPT10] Hillah, L.-M. et al.: PNML Framework. An Extendable Reference Implementation of the Petri Net Markup Language. In (Lilius, J.; Penczek, W. Hrsg.): Applications and Theory of Petri Nets. Springer, Berlin, Heidelberg, 2010; S. 318–327.

[Hoar69] Hoare, C. A. R.: An Axiomatic Basis for Computer Programming. In Communications of the Association for Computing Machinery (ACM), 1969, 12; S. 576–580.

[Hoff18] Hoffmann, D. W.: Grenzen der Mathematik. Eine Reise durch die Kerngebiete der mathematischen Logik. Springer, Berlin, Heidelberg, 2018.

[HoFL10] Houy, C.; Fettke, P.; Loos, P.: Empirical research in business process management. Analysis of an emerging field of research. In Business Process Management Journal (BPMJ), 2010, 16; S. 619–661.

[Holt68] Holt, A. W.: Final Report of the Information System Theory Project, New York, 1968.

[HoRG83] Holt, A. W.; Ramsey, H. R.; Grimes, J. D.: Coordination system technology as the basis for a programming environment. In Electrical Communication, 1983, 57; S. 307–314.

[HuLS10] Huai, W.; Liu, X.; Sun, H.: Towards Trustworthy Composite Service Through Business Process Model Verification. In (Institute of Electrical and Electronics Engineers (IEEE) Hrsg.): Ubiquitous, Autonomic and Trusted Computing (UIC-ATC). IEEE Computer Society, Los Alamitos, 2010; S. 422–427.

[HVSS+15] Höhne, M.; Vogel, J.; Schnägelberger, S.; Stark, M.; Schrodt, S.; Richter, C.; Eckhardt, V.; Simmion, S.: Business Process Management Studie. Messbare Verbesserung der Leistungsfähigkeit durch Prozessmanagement. http://www.bpmo.de/bpmo/export/sites/default/de/know_how/content/bpm_studie/bpm-studie/BPM_Studie_2015_DE_final.pdf, 07.09.2018.

[HZLH12] Han, Z. et al.: Control-Flow Pattern Based Transformation from UML Activity Diagram to YAWL. In (Sinderen, M. v. et al. Hrsg.): International Federation for Information Processing (IFIP). Springer, Berlin, Heidelberg, 2012; S. 129–145.

[IDEF0] Knowledge Based Systems, Inc. (KBSI): IDEFØ Method Report. http://www.idef.com/pdf/idef0.pdf, 07.09.2018.

[IDEF1a] Knowledge Based Systems, Inc. (KBSI): IDEF1 Method Report. Part 1. http://www.idef.com/pdf/IDEF1MR-part1.pdf, 07.09.2018.

[IDEF1b] Knowledge Based Systems, Inc. (KBSI): IDEF1 Method Report. Part 2. http://www.idef.com/pdf/IDEF1MR-part2.pdf, 07.09.2018.

[IDEF1X] Knowledge Based Systems, Inc. (KBSI): IDEF1X Method Report. http://www.idef.com/pdf/Idef1x.pdf, 07.09.2018.

[IDEF3] Knowledge Based Systems, Inc. (KBSI): IDEF3 Method Report. http://www.idef.com/pdf/Idef3_fn.pdf, 07.09.2018.

[IDEF4] Knowledge Based Systems, Inc. (KBSI): IDEF4 Method Report. http://www.idef.com/pdf/Idef4.pdf, 07.09.2018.

[IDEF5] Knowledge Based Systems, Inc. (KBSI): IDEF5 Method Report. http://www.idef.com/pdf/Idef5.pdf, 07.09.2018.

[ISO14977] International Organization for Standardization (ISO): Information technology - Syntactic metalanguage - Extended BNF, 1996.

[ISO15000-1] International Organization for Standardization (ISO): Electronic business eXtensible Markup Language (ebXML) - Part 1: Collaboration-protocol profile and agreement specification (ebCPP), 2004.

[ISO15000-2] International Organization for Standardization (ISO): Electronic business eXtensible Markup Language (ebXML) - Part 2: Message service specification (ebMS), 2004.

[ISO15000-3] International Organization for Standardization (ISO): Electronic business eXtensible Markup Language (ebXML) - Part 3: Registry information model specification (ebRIM), 2004.

[ISO15000-4] International Organization for Standardization (ISO): Electronic business eXtensible Markup Language (ebXML) - Part 4: Registry services specification (ebRS), 2004.

[ISO15000-5] International Organization for Standardization (ISO): Electronic business eXtensible Markup Language (ebXML) - Part 5: Core Components Specification, 2014.

[ISO15909-1] Internationale Organisation für Normung (ISO): Systems and software engineering - High-level Petri nets - Part 1: Concepts, definitions and graphical notation, 2004.

[ISO15909-2] Internationale Organisation für Normung (ISO): Systems and software engineering - High-level Petri nets - Part 2: Transfer format, 2011.

[ISO9000] Deutsches Institut für Normung e. V.: Qualitätsmanagementsysteme - Grundlagen und Begriffe, 2005.

[JaCJ92] Jacobson, I.; Christerson, M.; Jonsson, P.: Object-Oriented Software Engineering. A Use Case Driven Approach. A Use Case Approach. Addison-Wesley, Wokingham, 1992.

[JaPL98] Jaccheri, M. L.; Picco, P. G.; Lago, P.: Eliciting Software Process Models with the E3 Language. In Transactions on Software Engineering and Methodology, 1998, 7; S. 368–410.

[Jens91] Jensen, K.: Coloured Petri Nets. A High Level Language for System Design and Analysis. In (Rozenberg, G. Hrsg.): Advances in Petri Nets. Springer, Berlin, Heidelberg, 1991; S. 342–416.

[Jens92] Jensen, K.: Coloured Petri Nets. Basic Concepts. Volume 1. Springer, Berlin, Heidelberg, 1992.

[Jens95] Jensen, K.: Coloured Petri Nets. Analysis Methods. Volume 2. Springer, Berlin, Heidelberg, 1995.

[Jens97] Jensen, K.: Coloured Petri Nets. Practical Use. Volume 3. Springer, Berlin, Heidelberg, 1997.

[JKRT+16] Jannaber, S. et al.: Invigorating Event-driven Process chains. Towards an integrated meta model for EPC standardization. In (Betz, S.; Reimer, U. Hrsg.): Modellierung. Workshopband. Köllen, Bonn, 2016; S. 13–22.

[JuSp06] Jung, J.; Sprenger, J.: Muster für die Geschäftsprozessmodellierung. In (Schelp, J. et al. Hrsg.): Integration, Informationslogistik und Architektur. Köllen, Bonn, 2006; S. 189–204.

[KaBe02] Kalpic, B.; Bernus, P.: Business process modelling in industry. the powerful tool in enterprise management. In Computers in Industry, 2002, 47; S. 299–318.

[KaJS96] Karagiannis, D.; Junginger, S.; Strobl, R.: Introduction to Business Process Management Systems Concepts. In (Scholz-Reiter, B.; Stickel, E. Hrsg.): Business Process Modelling. Springer, Berlin, Heidelberg, 1996; S. 81–106.

[KaKü02] Karagiannis, D.; Kühn, H.: Metamodelling Platforms. In (Bauknecht, K.; Tjoa, A. M.; Quirchmayr, G. Hrsg.): E-Commerce and Web Technologies (EC-Web). Springer, Berlin, Heidelberg, 2002; S. 182.

[KaMM16] Karagiannis, D.; Mayr, H.C.; Mylopoulos, J. Hrsg.: Domain-Specific Conceptual Modeling. Concepts, Methods and Tools. Springer, Zürich, 2016.

[KBBR+16] Karagiannis, D. et al.: Fundamental Conceptual Modeling Languages in OMiLAB. In (Karagiannis, D.; Mayr, H. C.; Mylopoulos, J. Hrsg.): Domain-Specific Conceptual Modeling. Concepts, Methods and Tools. Springer, Zürich, 2016; S. 3–30.

[Kell99] Keller, G.: SAP R/3 prozeßorientiert anwenden. Iteratives Prozeß-Prototyping mit ereignisgesteuerten Prozeßketten und Knowledge Maps. Addison-Wesley, Bonn, 1999.

[KeNS92] Keller, G.; Nüttgens, M.; Scheer, A.-W.: Semantische Prozessmodellierung auf der Grundlage „Ereignisgesteuerter Prozeßketten (EPK)“, Saarbrücken, 1992.

[KeTe97] Keller, G.; Teufel, T.: SAP R/3 prozeßorientiert anwenden. Iteratives Prozeß-Prototyping zur Bildung von Wertschöpfungsketten. Addison-Wesley, 1997.

[KhBI17] Kheldoun, A.; Barkaoui, K.; Ioualalen, M.: Formal verification of complex business processes based on high-level Petri nets. In Information Sciences, 2017, 385-386; S. 39–54.

[KhSe12] Khademhosseinieh, B.; Seigerroth, U.: Towards Evaluating Efficiency of Enterprise Modeling Methods. In (Skersys, T.; Butleris, R.; Butkiene, R. Hrsg.): Information and Software Technologies (ICIST). Springer, Berlin, Heidelberg, 2012; S. 74–86.

[KIGK+14] Kossak, F. et al.: A Rigorous Semantics for BPMN 2.0 Process Diagrams. Springer, Berlin, Heidelberg, 2014.

[Kind06] Kindler, E.: On the semantics of EPCs. Resolving the vicious circle. In Data & Knowledge Engineering, 2006, 56; S. 23–40.

[KJHP15] Kocbek, M. et al.: Business Process Model and Notation. The Current State of Affairs. In Computer Science and Information Systems, 2015, 12; S. 509–539.

[KnGo10] Knieke, C.; Goltz, U.: An Executable Semantics for UML 2 Activity Diagrams. In (Heričko, M.; Živkovič, A. Hrsg.): European Conference on Object-Oriented Programming (ECOOP). Association for Computing Machinery (ACM), New York, 2010.

[KoLS04] Kopp, W.; Lorenz, W.; Selbmann, H.-K.: Methodische Empfehlungen zur Leitlinienerstellung. http://www.awmf.org/fileadmin/user_upload/Leitlinien/Werkzeuge/Publikationen/methoden.pdf, 07.09.2018.

[KoVZ97] Kosanke, K.; Vernadat, F. B.; Zelm, M.: CIMOSA process model for enterprise modelling. In (Goossenaerts, J.; Kimura, F.; Wortmann, H. Hrsg.): Information Infrastructure Systems for Manufacturing. Springer, Boston, 1997; S. 59–68.

[KrBL13] Krallmann, H.; Bobrik, A.; Levina, O.: Systemanalyse im Unternehmen. Prozessorientierte Methoden der Wirtschaftsinformatik. Oldenbourg, München, 2013.

[Krog12] Krogstie, J.: Model-Based Development and Evolution of Information Systems. A Quality Approach. Springer, London, 2012.

[LaCB96] Lakin, R.; Capon, N.; Botten, N.: BPR enabling software for the financial services industry. In Management services, 1996, 40; S. 18–20.

[LaHo13] Laue, R.; Hogrebe, F.: Zur Verständlichkeit graphischer Symbole in Geschäftsprozessmodellierungssprachen. In (Horbach, M. Hrsg.): Informatik angepasst an Mensch, Organisation und Umwelt. Köllen, Bonn, 2013; S. 693–705.

[LaMe08] Laue, R.; Mendling, J.: The Impact of Structuredness on Error Probability of Process Models. In (Kaschek, R. et al. Hrsg.): Information Systems and e-Business Technologies. Springer, Berlin, Heidelberg, 2008; S. 585–590.

[Lank13] Lankhorst, M.: Enterprise Architecture at Work. Modelling, Communication and Analysis. Springer, Berlin, Heidelberg, 2013.

[LaSW97] Langner, P.; Schneider, C.; Wehler, J.: Prozeßmodellierung mit ereignisgesteuerten Prozeßketten (EPKs) und Petri-Netzen. In Wirtschaftsinformatik, 1997, 39; S. 479–489.

[LaSW98] Langner, P.; Schneider, C.; Wehler, J.: Petri Net Based Certification of Event-Driven Process Chains. In (Desel, J.; Silva, M. Hrsg.): Application and Theory of Petri Nets. Springer, Berlin, Heidelberg, 1998; S. 286–305.

[LeMO05] Lenz, K.; Mevius, M. v.; Oberweis, A.: Process-oriented Business Performance Management. In (Cheung, W.; Hsu, J. Hrsg.): e-Technology, e-Commerce, and e-Services. IEEE Computer Society, Los Alamitos, 2005; S. 89–92.

[LeSM10] Leopold, H.; Smirnov, S.; Mendling, J.: Refactoring of Process Model Activity Labels. In (Hopfe, C. J. et al. Hrsg.): Natural Language Processing and Information Systems. Springer, Berlin, Heidelberg, 2010; S. 268–276.

[Lich12] Lichtenegger, W.: Methoden zur teilautomatischen Konstruktion von Ist-Prozessmodellen mittels Process Mining sowie zur Integration manuell konstruierter und automatisch generierter Ist-Prozessmodelle. Logos, Berlin, 2012.

[LoAl98] Loos, P.; Allweyer, T.: Process Orientation and Object-Orientation. An Approach for Integrating UML and Event-Driven Process Chains (EPC), Saarbrücken, 1998.

[LOVeM95] IBM Redbooks: Business Process Reengineering and Beyond. Vervante, San Jose, 1995.

[Lund01] Lund, A.: Measuring Usability with the USE Questionnaire. In Usability Interface, 2001, 8; S. 3–6.

[MaKu12] Marcinkowski, B.; Kuciapski, M.: A Business Process Modeling Notation Extension for Risk Handling. In (Cortesi, A. et al. Hrsg.): Computer Information Systems and Industrial Management (CISIM). Springer, Berlin, Heidelberg, 2012; S. 374–381.

[MaPo06] Maes, A.; Poels, G.: Evaluating Quality of Conceptual Models Based on User Perceptions. In (Embley, D. W.; Olivé, A.; Ram, S. Hrsg.): Conceptual Modeling (ER). Springer, Berlin, Heidelberg, 2006; S. 54–67.

[Maye13] Mayer, H. O.: Interview und schriftliche Befragung. Grundlagen und Methoden empirischer Sozialforschung. Oldenbourg, München, 2013.

[Maye99] Mayer, R. E.: Multimedia aids to problem-solving transfer. In International Journal of Educational Research, 1999, 31; S. 611–623.

[McCa62] McCarthy, J.: Towards a Mathematical Science of Computation. In (Colburn, T. R.; Fetzer, J. H.; Rankin, T. L. Hrsg.): Program Verification. Fundamental Issues in Computer Science. Springer, Dordrecht, 1962; S. 21–28.

[MDAW04] Medeiros, A.K.A. d. et al.: Process Mining. Extending the alpha-algorithm to Mine Short Loops, Eindhoven, 2004.

[MeAa07] Mendling, J.; Aalst, W. M. P. v. d.: Formalization and Verification of EPCs with OR-Joins Based on State and Context. In (Krogstie, J.; Opdahl, A. L.; Sindre, G. Hrsg.): Conference on Advanced Information Systems Engineering (CAiSE). Springer, Berlin, Heidelberg, 2007; S. 439–453.

[MeMN06] Mendling, J.; Moser, M.; Neumann, G.: Transformation of yEPC business process models to YAWL. In (Haddad, H. Hrsg.): Symposium on Applied Computing (SAC). Association for Computing Machinery (ACM), New York, 2006; S. 1262–1266.

[MeNA07] Mendling, J.; Neumann, G.; Aalst, W. M. P. v. d.: Understanding the Occurrence of Errors in Process Models Based on Metrics. In (Meersman, R.; Tari, Z. Hrsg.): On the Move to Meaningful Internet Systems (OTM). CoopIS, DOA, ODBASE, GADA and IS. Springer, Berlin, Heidelberg, 2007; S. 113–130.

[MeNü03] Mendling, J.; Nüttgens, M.: EPC Modelling based on Implicit Arc Types. In (Godlevsky, M. et al. Hrsg.): Information Systems Technology and its Applications (ISTA). Köllen, Bonn, 2003; S. 131–142.

[MeNü05] Mendling, J.; Nüttgens, M.: EPC Markup Language (EPML). An XML-Based Interchange Format for Event-Driven Process Chains (EPC), Wien, 2005.

[MeOb05] Mevius, M. v.; Oberweis, A.: A Petri-Net Based Approach to Performance Management of Collaborative Business Processes. In (Tjoa, A. M.; Wagner, R. R. Hrsg.): Workshop on Database and Expert Systems Applications (DEXA). IEEE Computer Society, Los Alamitos, 2005; S. 987–991.

[MeRC07] Mendling, J.; Reijers, H. A.; Cardoso, J.: What Makes Process Models Understandable? In (Alonso, G.; Dadam, P.; Rosemann, M. Hrsg.): Business Process Management (BPM). Springer, Berlin, Heidelberg, 2007; S. 48–63.

[MeRe08] Mendling, J.; Reijers, H. A.: The Impact of Activity Labeling Styles on Process Model Quality. In (Hesse, W.; Oberweis, A. Hrsg.): Symposium on Analysis, Design, Use and Societal Impact of Information Systems (SIGSAND). Köllen, Bonn, 2008; S. 117–128.

[MeRv10] Mendling, J.; Reijers, H. A.; van der Aalst, W. M. P.: Seven process modeling guidelines (7PMG). In Information & Software Technology, 2010, 52; S. 127–136.

[MeSt08] Mendling, J.; Strembeck, M.: Influence Factors of Understanding Business Process Models. In (Abramowicz, W.; Fensel, D. Hrsg.): Business Information Systems (BIS). Springer, Berlin, Heidelberg, 2008; S. 142–153.

[Mevi06] Mevius, M. v.: Kennzahlenbasiertes Management von Geschäftsprozessen mit Petri-Netzen. Dr. Hut, München, 2006.

[Mevi08] Mevius, M. v.: A Novel Modeling Language for Tool-based Business Process Engineering. In (Wainwright, R. L.; Haddad, H. Hrsg.): Symposium on Applied Computing (SAC). Association for Computing Machinery (ACM, New York, 2008; S. 590–591.

[MiMa13] Michael, J.; Mayr, H. C.: Conceptual Modeling for Ambient Assistance. In (Ng, W.; Storey, V. C.; Trujillo, J. C. Hrsg.): Conceptual Modeling (ER). Springer, 2013; S. 403–413.

[MiRi13] Minge, M.; Riedel, L.: meCUE. Ein modularer Fragebogen zur Erfassung des Nutzungserlebens. In (Boll, S.; Maaß, S.; Malaka, R. Hrsg.): Mensch & Computer. Interaktive Vielfalt. Oldenbourg, München, 2013; S. 89–98.

[MKPC14] Morais, R. M. de et al.: An Analysis of BPM Lifecycles. From a Literature Review to a Framework Proposal. In Business Process Management Journal (BPMJ), 2014, 20; S. 412–432.

[MMNV+06] Mendling, J. et al.: Faulty EPCs in the SAP Reference Model. In (Dustdar, S.; Fiadeiro, J. L.; Sheth, A. P. Hrsg.): Business Process Management (BPM). Springer, Berlin, Heidelberg, 2006; S. 451–457.

[MMRS10] Melcher, J. et al.: On Measuring the Understandability of Process Models. In (Rinderle-Ma, S.; Sadiq, S. W.; Leymann, F. Hrsg.): Business Process Management (BPM) Workshops. Springer, Berlin, Heidelberg, 2010; S. 465–476.

[MOF2.5.1] Object Management Group (OMG): Meta Object Facility (MOF). https://www.omg.org/spec/MOF/2.5.1/PDF/, 08.09.2018.

[MoRo00] Moldt, D.; Rodenhagen, J.: Ereignisgesteuerte Prozessketten und Petri-Netze zur Modellierung von Workflows. In (Giese, H.; Philippi, S. Hrsg.): Visuelle Verhaltensmodellierung verteilter und nebenläufiger Software-Systeme. Workshop, Münster, 2000; S. 57–63.

[MüBö15] Müller, C.; Bösing, K. D.: Vergleich von Simulationsfunktionalitäten in Werkzeugen zur Modellierung von Geschäftsprozessen. In (Barton, T. et al. Hrsg.): Prozesse, Technologie, Anwendungen, Systeme und Management. Mana-Buch, Heide, 2015; S. 9–19.

[MüLa17] Müller, C.; Laue, R.: Simulation von Geschäftsprozessen. Werkzeuge, Standards und Fallstricke. In (Barton, T.; Müller, C.; Seel, C.

Hrsg.): Geschäftsprozesse. Von der Modellierung zur Implementierung. Springer Gabler, Wiesbaden, 2017; S. 25–43.

[MVDA+08] Mendling, J. et al.: Detection and Prediction of Errors in EPCs of the SAP Reference Model. In Data & Knowledge Engineering, 2008, 64; S. 312–329.

[Neus06] Neuschulz, J.: Petri-Netz-gestützte Steuerung komplexer Controlling-Prozesse in einer Management-Holding. Eine anwendungsorientierte Betrachtung aus der Koordinationsperspektive. Dissertation, Braunschweig, 2006.

[NüFZ98] Nüttgens, M.; Feld, T.; Zimmermann, V.: Business Process Modeling with EPC and UML. Transformation or Integration? In (Schader, M.; Korthaus, A. Hrsg.): The Unified Modeling Language. Technical Aspects and Applications. Physica-Verlag, Heidelberg, 1998; S. 250–261.

[NüRu02] Nüttgens, M.; Rump, F. J.: Syntax und Semantik Ereignisgesteuerter Prozessketten (EPK). In (Desel, J.; Weske, M. Hrsg.): Prozessorientierte Methoden und Werkzeuge für die Entwicklung von Informationssystemen (Promise). Köllen, Bonn, 2002; S. 64–77.

[Ober90] Oberweis, A.: Zeitstrukturen für Informationssysteme. Dissertation, Mannheim, 1990.

[Ober96] Oberweis, A.: Modellierung und Ausführung von Workflows mit Petri-Netzen. Vieweg+Teubner, Stuttgart, Leipzig, 1996.

[OCL2.4] Object Management Group (OMG): Object Constraint Language (OCL). http://www.omg.org/spec/OCL/2.4/PDF/, 08.09.2018.

[OeSc13] Oestereich, B.; Scheithauer, A.: Analyse und Design mit der UML 2.5. Objektorientierte Softwareentwicklung. Oldenbourg, München, 2013.

[Ollo74] Ollongren, A.: Definition of Programming Languages by Interpreting Automata. Academic Press, London, 1974.

[Öste95] Österle, H.: Business Engineering Prozess- und Systementwicklung. Band 1. Entwurfstechniken. Springer, Berlin, Heidelberg, 1995.

[OuLi08] Ou-Yang, C.; Lin, Y. D.: BPMN-based business process model feasibility analysis. A petri net approach. In International Journal of Production Research, 2008, 46; S. 3763–3781.

[Özsu15] Özsucu, Ö.: Ein Geschäftsmodell für Logistikdienstleister im Umfeld von KMU. Wettbewerbsvorteile durch horizontale Kooperation und Revenue-Sharing. Springer Gabler, Wiesbaden, 2015.

[Pati06] Patig, S.: Die Evolution von Modellierungssprachen. Frank & Timme, Berlin, 2006.

[PeAa06] Pesic, M.; Aalst, W. M. P. v. d.: A Declarative Approach for Flexible Business Processes Management. In (Eder, J.; Dustdar, S. Hrsg.): Business Process Management (BPM). Workshops BPD, BPI, ENEI, GPWW, DPM, semantics4ws. Springer, Berlin, Heidelberg, 2006; S. 169–180.

[Pete81] Peterson, J. L.: Petri net theory and the modeling of systems. Prentice-Hall, Englewood Cliffs, 1981.

[Petr62] Petri, C. A.: Kommunikation mit Automaten. Dissertation, Bonn, 1962.

[Petr73] Petri, C. A.: Concepts of Net Theory. In (Becvar, J. et al. Hrsg.): Mathematical Foundations of Computer Science (MFCS). Slovak Academy of Sciences, Bratislava, 1973; S. 137–146.

[Petr76] Petri, C. A.: General Net Theory. In (Shaw, B. Hrsg.): Computing System Design, 1976; S. 131–169.

[PoPr04] Pokozy-Korenblat, K.; Priami, C.: Toward Extracting π-calculus from UML Sequence and State Diagrams. In Electronic Notes in Theoretical Computer Science, 2004, 101; S. 51–72.

[PrAn93] Prümper, J.; Anft, M.: Die Evaluation von Software auf Grundlage des Entwurfs zur internationalen Ergonomie-Norm ISO 9241 Teil 10 als Beitrag zur partizipativen Systemgestaltung. Ein Fallbeispiel. In (Rödiger, K.-H. Hrsg.): Software-Ergonomie. Von der Benutzungsoberfläche zur Arbeitsgestaltung. Vieweg+Teubner, Wiesbaden, 1993; S. 145–156.

[Prit66] Pritsker, A. A. B.: GERT. Graphical Evaluation and Review Technique, Santa Monica, 1966.

[PrQZ08] Prandi, D.; Quaglia, P.; Zannone, N.: Formal Analysis of BPMN Via a Translation into COWS. In (Lea, D.; Zavattaro, G. Hrsg.): Coordination Models and Languages (Coordination). Springer, Berlin, Heidelberg, 2008; S. 249–263.

[Puhl07] Puhlmann, F.: Soundness Verification of Business Processes Specified in the Pi-Calculus. In (Meersman, R.; Tari, Z. Hrsg.): On the Move to Meaningful Internet Systems (OTM). CoopIS, DOA, ODBASE, GADA and IS. Springer, Berlin, Heidelberg, 2007; S. 6–23.

[PuWe06] Puhlmann, F.; Weske, M.: Investigations on Soundness Regarding Lazy Activities. In (Dustdar, S.; Fiadeiro, J. L.; Sheth, A. P. Hrsg.): Business Process Management (BPM). Springer, Berlin, Heidelberg, 2006; S. 145–160.

[RaEH10] Ramadan, M.; Elmongui, H. G.; Hassan, R.: BPMN Formalisation using Coloured Petri Nets. In (Rinderle-Ma, S.; Sadiq, S. W.; Leymann, F. Hrsg.): Business Process Management (BPM) Workshops. Springer, Berlin, Heidelberg, 2010; S. 5–16.

[RAHE05] Russell, N. et al.: Workflow Resource Patterns. Identification, Representation and Tool Support. In (Pastor, O.; Cunha,, J. F. e. Hrsg.): Conference on Advanced Information Systems Engineering (CAiSE). Springer, Berlin, Heidelberg, 2005; S. 216–232.

[RaSN15] Radloff, M.; Schultz, M.; Nüttgens, M.: Extending different Business Process Modeling Languages with Domain Specific Concepts. The Case of Internal Controls in EPC and BPMN. In (Kolb, J.; Leopold, H.; Mendling, J. Hrsg.): Enterprise Modelling and Information Systems Architectures (EMISA). Köllen, Bonn, 2015; S. 45–58.

[RaST13] Rauschenberger, M.; Schrepp, M.; Thomaschewski, J.: User Experience mit Fragebögen messen. Durchführung und Auswertung am Beispiel des UEQ. In (Brau, H. et al. Hrsg.): Usability Professionals (UP). German UPA e.V., Stuttgart, 2013; S. 72–77.

[RBPE+91] Rumbaugh, J. et al.: Object-Oriented Modeling and Design. Prentice-Hall, Englewood Cliffs, 1991.

[ReDr07] Recker, J.; Dreiling, A.: Does It Matter Which Process Modelling Language We Teach or Use? An Experimental Study on Understanding Process Modelling Languages without Formal Education. In (Toleman, M.; Cater-Steel, A.; Roberts, D. Hrsg.): Australasian Conference on Information Systems (ACIS). Association for Information Systems (AIS), Atlanta, 2007; S. 356–366.

[Reis90] Reisig, W.: Petrinetze. Eine Einführung. Springer, Berlin, Heidelberg, 1990.

[ReMR15] Reijers, H. A.; Mendling, J.; Recker, J.: Business Process Quality Management. In (Brocke, J. v.; Rosemann, M. Hrsg.): Handbook on Business Process Management 1. Introduction, Methods and Information Systems. Springer, Berlin, Heidelberg, 2015; S. 167–185.

[RHAM06] Russell, N. et al.: Workflow Control-Flow Patterns. A Revised View, 2006.

[RHEA05] Russell, N. et al.: Workflow Data Patterns. Identification, Representation and Tool Support. In (Delcambre, L. M. L. et al. Hrsg.): Conceptual Modeling (ER). Springer, Berlin, Heidelberg, 2005; S. 353–368.

[RiSS93] Richter, R.; Sander, P.; Stucky, W.: Grundkurs Angewandte Informatik II. Problem - Algorithmus - Programm. Vieweg+Teubner, Wiesbaden, 1993.

[Ritt00] Rittgen, P.: Quo vadis EPK in ARIS? Ansätze zu syntaktischen Erweiterungen und einer formalen Semantik. In Wirtschaftsinformatik, 2000, 42; S. 27–35.

[Ritt99a] Rittgen, P.: Modified EPCs and Their Formal Semantics, Koblenz-Landau, 1999.

[Ritt99b] Rittgen, P.: Vom Prozessmodell zum elektronischen Geschäftsprozess, Koblenz-Landau, 1999.

[Ritt99c] Rittgen, P.: From Process Model to Electronic Business Process. In (Pries-Heje, J. et al. Hrsg.): European Conference on Information Systems (ECIS). Copenhagen Business School, Kopenhagen, 1999; S. 616–629.

[RoAa08] Rozinat, A.; Aalst, W. M. P. v. d.: Conformance checking of processes based on monitoring real behavior. In Information Systems, 2008, 33; S. 64–95.

[RoEn98] Rozenberg, G.; Engelfriet, J.: Elementary Net Systems. In (Reisig, W.; Rozenberg, G. Hrsg.): Lectures on Petri Nets I. Basic Models. Advances in Petri Nets. Springer, Berlin, Heidelberg, 1998; S. 12–121.

[Rose06] Rosenkranz, F.: Geschäftsprozesse. Modell- und computergestützte Planung. Springer, Berlin, 2006.

[Rose96] Rosemann, M.: Komplexitätsmanagement in Prozeßmodellen. Methodenspezifische Gestaltungsempfehlungen für die Informationsmodellierung. Gabler Verlag, Wiesbaden, 1996.

[RosettaNet] RosettaNet: PIDX Implementation Guideline (RosettaNet). http://www.pidx.org/wp-content/uploads/2014/07/PIDX-XML-Implementation-Guideline-_RosettaNet_-version-2_0-2.pdf, 08.09.2018.

[RoTu08] Rossi, D.; Turrini, E.: EPML. An Executable Process Modeling Language for Process-Aware Applications. In (Wainwright, R. L.; Haddad, H. Hrsg.): Symposium on Applied Computing (SAC). Association for Computing Machinery (ACM), New York, 2008; S. 132–133.

[Roze97] Rozenberg, G.: Handbook of Graph Grammars and Computing by Graph Transformation. Volume 1. Foundations. World Scientific, Singapore, 1997.

[RPUW+07] Raedts, I. et al.: Transformation of BPMN Models for Behaviour Analysis. In (Augusto, J. C.; Barjis, J.; Ultes-Nitsche, U. Hrsg.): Modelling, Simulation, Verification and Validation of Enterprise Information Systems (MSVVEIS). Insticc, Setúbal, 2007; S. 126–137.

[RuAH06] Russell, N.; Aalst, W. M. P. v. d.; Hofstede, A. H. M. t.: Exception Handling Patterns in Process-Aware Information Systems. http://www.workflowpatterns.com/documentation/documents/BPM-06-04.pdf, 08.09.2018.

[RuHA07] Russell, N.; Hofstede, A. H. M. t.; Aalst, W. M. P. v. d.: newYAWL. Specifying a Workflow Reference Language using Coloured Petri Nets. In (Jensen, K. Hrsg.): Practical Use of Coloured Petri Nets and CPN Tools. Department of Computer Science, Aarhus, 2007; S. 107–126.

[RuHE07] Russell, N.; Hofstede, A. H. M. t.; Edmond, D.: newYAWL. Achieving Comprehensive Patterns Support in Workflow for the Control-Flow, Data and Resource Perspectives, BPMcenter.org, 2007.

[Rump99] Rump, F. J.: Geschäftsprozessmanagement auf der Basis ereignisgesteuerter Prozessketten. Formalisierung, Analyse und Ausführung von EPKs. Vieweg+Teubner, Stuttgart, 1999.

[RuQu12] Rupp, C.; Queins, S.: UML 2 Glasklar. Praxiswissen für die UML-Modellierung. Carl Hanser, München, 2012.

[Russ07] Russell, N.: Foundations of Process-Aware Information Systems. Dissertation, Brisbane, 2007.

[RWAH+09] Rozinat, A. et al.: Workflow Simulation for Operational Decision Support. In Data & Knowledge Engineering, 2009, 68; S. 834–850.

[SaGu06] Sarstedt, S.; Guttmann, W.: An ASM Semantics of Token Flow in UML 2 Activity Diagrams. In (Virbitskaite, I.; Voronkov, A. Hrsg.): Perspectives of Systems Informatics (PSI). Springer, Berlin, Heidelberg, 2006; S. 349–362.

[Sars06] Sarstedt, S.: Semantic Foundation and Tool Support for Model-Driven Development with UML 2 Activity Diagrams. Dissertation, Ulm, 2006.

[ScGr06] Schacher, M.; Grässle, P.: Agile Unternehmen durch Business Rules. Der Business Rules Ansatz. Springer, Berlin, Heidelberg, 2006.

[Sche01] Scheer, A.-W.: ARIS. Modellierungsmethoden, Metamodelle, Anwendungen. Springer, Berlin, Heidelberg, 2001.

[Sche02] Scheer, A.-W.: ARIS. Vom Geschäftsprozess zum Anwendungssystem. Springer, Berlin, Heidelberg, 2002.

[Sche94] Scheer, A.-W.: Business Process Engineering. Reference Models for Industrial Enterprises. Springer, Berlin, Heidelberg, 1994.

[Sche99] Scheer, A.-W.: ARIS. Business Process Frameworks. Springer, Berlin, Heidelberg, 1999.

[Schm96] Schmidt, G.: Scheduling Models for Workflow Management. In (Scholz-Reiter, B.; Stickel, E. Hrsg.): Business Process Modelling. Springer, Berlin, Heidelberg, 1996; S. 67–80.

[Schu12] Schuster, T.: Modellierung, Integration und Analyse von Ressourcen in Geschäftsprozessen. KIT Scientific Publishing, Karlsruhe, 2012.

[ScNZ97] Scheer, A.-W.; Nüttgens, M.; Zimmermann, V.: Objektorientierte Objektorientierte Ereignisgesteuerte Prozeßkette (oEPK). Methode und Anwendung, Saarbrücken, 1997.

[ScRa14] Schultz, M.; Radloff, M.: Modeling Concepts for Internal Controls in Business Processes. An Empirically Grounded Extension of BPMN. In (Sadiq, S. W.; Soffer, P.; Völzer, H. Hrsg.): Business Process Management (BPM). Springer, Berlin, Heidelberg, 2014; S. 184–199.

[Silv08] Silver, B.: Ten Tips for Effective Process Modeling. http://www.bpminstitute.org/articles/article/article/bpms-watch-ten-tips-for-effective-process-modeling.html, 08.09.2018.

[ŠkTr13] Škrinjar, R.; Trkman, P.: Increasing process orientation with business process management: Critical practices'. In International Journal of Information Management, 2013, 33; S. 48–60.

[SKVM+18] Schaefer, I. et al. Hrsg.: Modellierung. Köllen, Bonn, 2018.

[SpMJ96] Spur, G.; Mertins, K.; Jochem, R.: Integrated Enterprise Modelling. Developments for the standardization of CIM. Beuth, Berlin, 1996.

[Stac73] Stachowiak, H.: Allgemeine Modelltheorie. Springer, Wien, 1973.

[StCV11] Stroppi, L. J. R.; Chiotti, O.; Villarreal, P. D.: Extending BPMN 2.0: Method and Tool Support. In (Dijkman, R. M.; Hofstetter, J.; Koehler, J. Hrsg.): Business Process Model and Notation (BPMN). Springer, Berlin, Heidelberg, 2011.

[Stör05] Störrle, H.: UML 2 erfolgreich einsetzen. Einführung und Referenz. Addison-Wesley-Longmen Publishing, München, Boston, 2005.

[Stra98] Stragringer, S.: Ein sprachbasierter Metamodellbegriff und seine Verallgemeinerung durch das Konzept des Metaisierungsprinzips. In (Pohl, K.; Schürr, A.; Vossen, G. Hrsg.): Modellierung. Workshop. CEUR-WS.org, Aachen, 1998; S. 15–20.

[SuFo03] Sutton, D. R.; Fox, J.: The Syntax and Semantics of the PROforma Guideline Modeling Language. In Journal of the American Medical Informatics Association (JAMIA), 2003, 10; S. 433–443.

[SVKF+18] Schoknecht, A. et al.: Business Process Model Patterns. State-of-the art, Research Classification and Taxonomy. In Business & Information Systems Engineering, 2018.

[SVOK12] Schönthaler, F. et al.: Business Processes for Business Communities. Modeling Languages, Methods, Tools. Springer, Berlin, Heidelberg, 2012.

[SWFP15] Speck, A. et al.: Tool-based Checking of Business Process Models. In (Ehlers, J.; Thalheim, B. Hrsg.): Subject-Oriented Business Process Management (S-BPM). Association for Computing Machinery (ACM), New York, 2015.

[SWMR09] Schrepfer, M. et al.: The Impact of Secondary Notation on Process Model Understanding. In (Persson, A.; Stirna, J. Hrsg.): The Practice of Enterprise Modeling (PoEM). Springer, Berlin, Heidelberg, 2009; S. 161–175.

[Take08] Takemura, T.: Formal Semantics and Verification of BPMN Transaction and Compensation. In (Chao, H.-C.; Tsai, J.; Cappello, F. Hrsg.): Asia-Pacific

Services Computing Conference (APSCC). IEEE Computer Society, Los Alamitos, 2008; S. 284–290.

[Tard92] Tardieu, H.: Issues for Dynamic Modelling through Recent Development in European Methods. In (Sol, H. G.; Crosslin, R. L. Hrsg.): Dynamic Modelling of Information Systems II. North-Holland, Amsterdam, 1992; S. 3–23.

[TBCB12] Turki, S. H. et al.: Modeling Security Requirements in Service Based Business Processes. In (Bider, I. et al. Hrsg.): Enterprise, Business-Process and Information Systems Modeling (BPMDS). Springer, Berlin, Heidelberg, 2012; S. 76–90.

[TCGN+07] Tu, S. W. et al.: The SAGE Guideline Model. Achievements and Overview. In Journal of the American Medical Informatics Association (JAMIA), 2007, 14; S. 589–598.

[Thia87] Thiagarajan, P. S.: Elementary Net Systems. In (Brauer, W.; Reisig, W.; Rozenberg, G. Hrsg.): Petri Nets. Central Models and Their Properties. Advances in Petri Nets. Springer, Berlin, Heidelberg, 1987; S. 26–59.

[ThLR07] Thom, L. H.; Lochpe, C.; Reichert, M. U.: Workflow Patterns for Business Process Modeling. In (Pernici, B. Hrsg.): Conference on Advanced Information Systems Engineering (CAiSE). AOIS, BPMDS, BUSITAL, EMMSAD, UMICS, WISM and Doctoral Consortium. Tapir Academic, Trondheim, 2007; S. 349–358.

[Tuck04] Tucker, A. B.: Computer Science. Handbook. Chapman & Hall/CRC, Boca Raton, 2004.

[UHH18] Universität Hamburg: Literaturliste Petri-Netze. http://www.informatik.uni-hamburg.de/TGI/pnbib/, 10.09.2018.

[UML2.5.1] Object Management Group (OMG): Unified Modeling Language (UML). Spezifikation. http://www.omg.org/spec/UML/2.5.1/, 10.05.2018.

[VaFC91] Valero Ruiz, V.; Frutos-Escrig, D. de; Cuartero, F.: Simulation of Timed Petri Nets by Ordinary Petri Nets and Applications to Decidability of the Timed Reachability Problem and other Related Problems. In (Institute of Electrical and Electronics Engineers (IEEE) Hrsg.): Petri Nets and Performance Models (PNPM) Workshop. IEEE Computer Society, Los Alamitos, 1991; S. 154–163.

[ViKa05] Vitolins, V.; Kalnins, A.: Semantics of UML 2.0 Activity Diagram for Business Modeling by Means of Virtual Machine. In (Sinderen, M. J. v. et al. Hrsg.): Enterprise Distributed Object Computing Conference (EDOC). IEEE Computer Society, Los Alamitos, 2005; S. 181–194.

[Wesk12] Weske, M.: Business Process Management. Concepts, Languages, Architectures. Springer, Berlin, Heidelberg, 2012.

[WFHS+15] Witt, S. et al.: Visualization of Checking Results for Graphical Validation Rules. In (Fujita, H.; Guizzi, G. Hrsg.): Intelligent Software Methodologies, Tools and Techniques (SoMeT). Springer, Zürich, 2015; S. 120–136.

[WfMC99] Workflow Management Coalition (WfMC): Workflow Management Coalition Terminology & Glossary. http://www.wfmc.org/standards/docs/TC-1011_term_glossary_v3.pdf, 10.09.2018.

[Whit04] White, S. A.: Introduction to BPMN. http://www.bpmn.org/Documents/Introduction%20to%20BPMN.pdf, 04.08.2016.

[WhMi08] White, S. A.; Miers, D.: BPMN Modeling and Reference Guide. Understanding and Using BPMN. Future Strategies, Lighthouse Point, 2008.

[WiGH96] Willumeit, H.; Gediga, G.; Hamborg, K.-C.: IsoMetrics. Ein Verfahren zur formativen Evaluation von Software nach ISO 9241/10. In Ergonomie und Informatik, 1996, 27; S. 5–12.

[WiSa15] Wiebring, J.; Sandkuhl, K.: Selecting the "Right" Notation for Business Process Modeling: Experiences from an Industrial Case. In (Matulevicius, R.; Dumas, M. Hrsg.): Perspectives in Business Informatics Research (BIR). Springer, Berlin, Heidelberg, 2015; S. 129–144.

[Witt07] Wittenburg, A.: Softwarekartographie. Modelle und Methoden zur systematischen Visualisierung von Anwendungslandschaften. Dissertation, München, 2007.

[WoGi08] Wong, P. Y. H.; Gibbons, J.: A Process Semantics for BPMN. In (Liu, S.; Maibaum, T.; Araki, K. Hrsg.): International Conference on Formal Engineering Methods (ICFEM). Springer, Berlin, Heidelberg, 2008; S. 355–374.

[WoGi11] Wong, P. Y. H.; Gibbons, J.: Formalisations and applications of BPMN. In Science of Computer Programming, 2011, 76; S. 633–650.

[Work12] Workflow Management Coalition (WfMC): XML Process Definition Language (XPDL). http://www.xpdl.org/standards/xpdl-2.2/XPDL%202.2%20(2012-08-30).pdf, 09.05.2016.

[WS-BPEL2.0] Organization for the Advancement of Structured Information Standards (OASIS): Web Service Business Prtocess Exceution Language (WSBPEL). Version 2.0. http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.pdf, 10.09.2018.

[YeSo10] Ye, J. H.; Song, W.: Transformation of BPMN Diagrams to YAWL Nets. In Journal of Software, 2010, 5; S. 396–404.

[YSSW08] Ye, J. H. et al.: Formal Semantics of BPMN Process Models Using YAWL. In (Zhou, Q.; Luo, J. Hrsg.): Intelligent Information Technology Application (IITA). IEEE Computer Society, Los Alamitos, 2008; S. 70–74.

[YSWS08] Ye, J. et al.: Transformation of BPMN to YAWL. In (Zhou, H.; Shen, P. Hrsg.): Computer Science and Software Engineering (CSSE). IEEE Computer Society, Los Alamitos, 2008; S. 354–359.

Geschäftsprozessmodelle bilden eine wichtige Grundlage für das Geschäftsprozessmanagement. Zur Modellierung dieser Modelle werden in der Praxis zahlreiche Geschäftsprozessmodellierungssprachen eingesetzt, die meist keine präzise Kontrollflusssemantik besitzen. Durch den Kontrollfluss wird die Ablaufreihenfolge der Elemente im Geschäftsprozessmodell festgelegt. Unerwünschte Interpretationsspielräume ergeben sich immer dann, wenn die Kontrollflusssemantik unpräzise definiert ist. Dies ist beispielsweise der Fall, wenn die Bedeutung einzelner Symbole nicht eindeutig festgelegt ist. Missverständnisse werden zwischen Domänenexperten und Modellierern beispielsweise auch durch unpräzise Beschreibungen gefördert. In dieser Arbeit wird eine auf Kontrollflussmustern basierende Methode zur Beschreibung der Kontrollflusssemantik von graphischen Geschäftsprozessmodellierungssprachen vorgestellt. Auf dieser Grundlage kann die Ablaufreihenfolge der Elemente in den Modellen präzise beschrieben werden.

ISBN 978-3-7315-0913-4

Gedruckt auf FSC-zertifiziertem Papier