MASTERARBEIT

Instanziierung von BPMN-Prozessen in ArchiMate-Modellen

Möglichkeiten und Grenzen von ArchiMate sowie BPMN zur Instanziierung von Geschäftsprozessen in Unternehmensarchitekturen

ausgeführt am

Studiengang
Informationstechnologien und Wirtschaftsinformatik

Von: Dipl.-Ing. Raphael Hackensöllner, BSc
Personenkennzeichen: 51841312

Graz, am 9. Juni 2025 .....

Raphael Hackensöllner

EHRENWÖRTLICHE ERKLÄRUNG

Ich erkläre ehrenwörtlich, dass ich

- die vorliegende Arbeit selbständig und ohne fremde Hilfe verfasst,
- andere als die angegebenen Quellen nicht benutzt,
- die den Quellen wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich
- gemacht,
- den Einsatz von generativen KI-Modellen (z.B. ChatGPT) kenntlich gemacht
- und mich sonst keiner unerlaubten Hilfsmittel bedient habe.

Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht. Die vorliegende Fassung entspricht der eingereichten elektronischen Version.

Graz, am 9. Juni 2025 .....

Raphael Hackensöllner

DANKSAGUNG

Ich möchte meinen herzlichsten Dank an meinen Betreuer, Herrn Mag. (FH) Dr. Michael Amann-Langeder aussprechen.

Vielen Dank für die kontinuierliche Unterstützung sowie die fachlich fundierten Feedbackgespräche, aber vor allem auch der äußerst humorvolle Austausch miteinander. So hat das Erarbeiten der Masterarbeit mir wirklich viel Spaß und Freude bereitet. Außerdem möchte ich mich zusätzlich bedanken, dass ich die Cockpit365-Plattform vollständig nutzen durfte. Mit dieser tollen Applikation konnten alle Anforderungen der Arbeit unkompliziert und ohne lange Einarbeitungszeit umgesetzt werden. Vielen Dank auch für den schnellen und ausführlichen Support bei meinen technischen Fragen.

Abschließend möchte ich mich bei Ihnen, den Leserinnen und Lesern, bedanken und hoffe, dass Sie einen Mehrwert aus dieser Arbeit ziehen können.

KURZFASSUNG

Eine Unternehmensarchitektur bietet einen umfassenden Überblick über ein Unternehmen inklusive dessen Infrastruktur, Anwendungen und Geschäftsprozessen. Aufgrund von Silo-Denken und unzureichender Kommunikation können jedoch Diskrepanzen zwischen den in der Unternehmensarchitektur dokumentierten Geschäftsprozessen und den tatsächlich etablierten Prozessen auftreten. Diese Arbeit zielt darauf ab, jene Diskrepanzen zu beseitigen, indem eine technische Lösung vorgeschlagen wird, um die Unternehmensarchitekturen mit den Geschäftsprozessen in Einklang zu bringen.

ArchiMate, eine ikonografische Modellierungssprache für Unternehmensarchitekturen, und BPMN, der de facto Standard für die Modellierung von Geschäftsprozessen, wurden verwendet. Durch die Anwendung der Design Science Research Methode, insbesondere des sechsstufigen Ansatzes, wurde initial Literatur gesichtet und folglich ein Artefakt erstellt, demonstriert und bewertet. Das resultierende Artefakt besteht aus einem Mapping zwischen ArchiMate und instanziierbaren BPMN-Konzepten sowie einem Java-basierten Programm, das die automatische Instanziierung von BPMN-Prozessen aus ArchiMate-Modellen ermöglicht.

Um das Artefakt zu evaluieren, wurden 20 standardisierte Workflow Patterns und zwei beispielhafte ITIL-Praktiken, darunter das „Monitoring und Event Management“ sowie das „Change Management“, in ArchiMate modelliert, automatisch transformiert und als BPMN-Prozessinstanzen ausgeführt. Von den 20 Workflow Patterns sind nur 17 in BPMN selbst möglich, jedoch konnten 14 von diesen erfolgreich modelliert und instanziiert werden. Dies zeigt, dass trotz der begrenzten Anzahl von ArchiMate und folglich auch BPMN-Konzepten, anspruchsvolle Geschäftsprozesse in der Unternehmensarchitektur selbst abgebildet werden können. Die erfolgreiche Instanziierung und Ausführung der ITIL-Praktiken bestätigten zudem die praktische Anwendbarkeit des Artefakts in realen Szenarien.

Zusammenfassend zeigt diese Arbeit, dass ArchiMate-Modelle erfolgreich in BPMN-Prozesse transformiert werden können, sodass instanziierbare und semantisch sinnvolle Prozessmodelle entstehen. Das entwickelte Artefakt ermöglicht die automatisierte Umsetzung und Ausführung von Geschäftsprozessen direkt aus Unternehmensarchitekturmodellen. Dieser Ansatz fördert die Konsistenz zwischen den dokumentierten Unternehmensarchitekturen und realen Abläufen.

ABSTRACT

An enterprise architecture provides a comprehensive overview of a company, containing its infrastructure, applications, and business processes. However, discrepancies can arise between the documented business processes within the enterprise architecture and the actual processes due to siloed thinking and inadequate communication. This thesis aims to address these misalignments by proposing a technical solution to align the enterprise architectures with business processes.

ArchiMate, an iconographic modeling language for enterprise architectures, and BPMN, the de facto standard for business process modeling, were utilized. Through the use of the Design Science Research Methodology, in particular the six-step approach, literature was reviewed and an artifact was created, demonstrated and evaluated. The resulting artifact consists of a mapping between ArchiMate and instantiable BPMN concepts, alongside a Java-based program that enables the automatic instantiation of BPMN processes from ArchiMate models.

To evaluate the artifact, 20 standardized workflow patterns and two sample ITIL practices, including the “Monitoring and Event Management”, as well as “Change Management”, were modeled in ArchiMate, automatically transformed, and then executed as BPMN process instances. Of the 20 workflow patterns, only 17 are possible within BPMN itself, however 14 of them were successfully modeled and instantiated. This indicates that despite the limited number of ArchiMate and therefore also BPMN concepts, sophisticated business processes can be depicted within the enterprise architecture itself. The successful execution of the ITIL practices further validated the artifact's practical applicability to real-world scenarios.

In conclusion, this research demonstrates that ArchiMate models can be successfully mapped to BPMN in a way that produces instantiable and semantically meaningful process models. The developed artifact enables the automated transformation and execution of business processes directly from enterprise architecture models. This approach promotes consistency between documented enterprise architectures and real-world operations.

INHALTSVERZEICHNIS

1 EINLEITUNG

Im Sinne der Modellierung von Unternehmensarchitekturen, Geschäftsprozessen, Entity-Relationship-Modellen, Softwareapplikationen, sowie in anderen Domänen, werden diverse Modellierungssprachen und Techniken eingesetzt, um diese zu visualisieren (M. M. Lankhorst et al., 2017). Darunter zählt die ikonografische Modellierungssprache ArchiMate, welche entwickelt wurde, um holistische Unternehmensarchitekturen mit einer geringen Anzahl an Elementen zu modellieren (The Open Group Standard, 2022a). Die Stärke dieser Sprache besteht darin, dass die mittels ArchiMate entwickelten Modelle und Architekturen mit den spezifischen Domänen anderer Modellierungssprachen verknüpft werden können (Penicina, 2013). Eine solche Domäne wäre der Bereich der Geschäftsprozessmodellierung. Für die detaillierte Modellierung als auch Automatisierung von Geschäftsprozessen hat sich Business Process Model and Notation, kurz BPMN, als de facto Standard erwiesen (Göpfert & Lindenbach, 2013). Durch BPMN werden Geschäftsprozesse in verschiedenen Detaillierungsgraden modelliert und durch geeignete Softwareapplikationen automatisiert.

Folglich bestehen zwei Sichtweisen, welche miteinander harmonieren können. Zum einen bietet ArchiMate einen abstrakten Überblick über die gesamte Unternehmensarchitektur, wodurch sich Kardinalitäten und Abhängigkeiten der einzelnen Prozesse, Applikationen und IT-Systeme im Unternehmen analysieren und definieren lassen (M. M. Lankhorst et al., 2017). Zum anderen dient BPMN zur Darstellung, Dokumentation sowie Automatisierung der in einem Unternehmen vorhandenen Geschäftsprozesse (Göpfert & Lindenbach, 2013).

Um diese zwei Sichtweisen direkt miteinander zu verbinden, wird in dieser Arbeit ein Artefakt nach der Design Science Research Methode nach Peffers et al. (2007) entwickelt. Das Artefakt inkludiert ein Mapping zwischen ArchiMate und BPMN, als auch die Möglichkeit, automatisiert aus ArchiMate-Modellen BPMN-Prozesse zu instanziieren. Mit dieser direkten Interaktion zwischen der Unternehmensarchitektur und den Geschäftsprozessen können effizientere Entscheidungen über Prozessänderungen getroffen werden, oder neue Geschäftsprozesse vollautomatisiert aus der Unternehmensarchitektur instanziiert werden.

1.1 Problemstellung und Motivation

Die Harmonisierung zwischen den Geschäftsprozessen und den IT-Infrastrukturen stellt einen zentralen Faktor für eine effiziente Organisation in Unternehmen dar (Buchta, 2009). Die Modellierungssprache ArchiMate wurde hierbei entwickelt, um als zentrale Anlaufstelle für verschiedene spezifische Domänen wie beispielsweise den Geschäftsprozessen zu dienen (M. M. Lankhorst et al., 2017). Durch die Integration dieser unterschiedlichen Domänen in einem

konsolidierten Modell kann eine „Single-Source-of-Truth“ geschaffen werden, welche es ermöglicht, unternehmensweite Transparenz und Nachvollziehbarkeit über alle enthaltenen Bereiche zu gewährleisten. Eine solche Transparenz ist essenziell, um fundierte und effiziente Entscheidungen über Anpassungen in der Organisation zu treffen, da die Kardinalitäten und Abhängigkeiten zwischen den Geschäftsprozessen, Applikationen und IT-Infrastruktur auf einem zentralen Punkt dargestellt werden (Winter & Fischer, 2006).

In der Praxis zeigt sich, dass Unternehmen vor dem grundlegenden Problem stehen, dass ein Dissens zwischen den Bereichen der IT-Architektur und den Geschäftsprozessen besteht. Es kommt zur Bildung von Silos, welche autark über ihre Bereiche und Domänen verfügen, wodurch beispielsweise IT-Landschaften häufig autonom entwickelt werden, ohne eine enge Verzahnung mit den zugrundeliegenden Geschäftsprozessen (Yildiz, 2022). Diese Fragmentierung führt zu dem Business-IT-Gap, welcher die Diskrepanz zwischen den strategischen Zielen des Unternehmens und der operativen Umsetzung in den IT-Systemen definiert (Ross et al., 2006).

Während die Geschäftsprozesse in der Regel auf Effizienz und Kundennutzen ausgerichtet sind, um diesen einen Mehrwert zu bieten, folgen IT-Infrastrukturen oft einem technologie- oder kostengetriebenen Ansatz, der die strategischen Bedürfnisse nicht adäquat berücksichtigt (M. M. Lankhorst et al., 2017). Dies stellt ein organisatorisches oder auch unternehmenskulturelles Problem dar, bei welchem diese Silos durch einen Mangel an Kommunikation zwischen den einzelnen Bereichen entstehen (Yildiz, 2022).

Der Aspekt der durch die Silos entstandenen Diskrepanz wird zusätzlich dadurch verschärft, dass in den einzelnen Bereichen der Unternehmen oft in separaten Tools und Frameworks gearbeitet wird (Yildiz, 2022). So wird etwa der Bereich der Geschäftsprozesse detailliert in BPMN-Modellen beschrieben, wohingegen die Modellierung der IT-Infrastruktur und Applikationslandschaft in abstrakteren Architekturbeschreibungen erfolgt. Ein direkter Abgleich dieser Modelle ist häufig nicht möglich, da eine einheitliche Datenbasis fehlt (M. M. Lankhorst et al., 2017).

Ein weiteres zentrales Problem, das aus der Bildung von Silos resultiert, ist die doppelte Modellierung von Unternehmensarchitekturen und Geschäftsprozessen sowie deren Inkonsistenz untereinander. In einem Modell einer Unternehmensarchitektur, welches beispielsweise in ArchiMate modelliert ist, sind die Geschäftsprozesse in einer abstrakteren Form visualisiert, um die strategischen Absichten des Unternehmens aufzuzeigen. Gleichzeitig erfolgt im Bereich der Geschäftsprozessmodellierung eine detailliertere Beschreibung der Prozesse, insbesondere für die Automatisierung von BPMN-Prozessen, um operative Abläufe zu optimieren (Göpfert & Lindenbach, 2013).

Diese doppelte Modellierung ist grundsätzlich notwendig, da die abstraktere Unternehmensarchitektur die im Unternehmen vorhandenen Geschäftsprozesse enthalten muss. Aufgrund der organisatorischen Silos entstehen jedoch Inkonsistenzen, da die beiden Darstellungen häufig nicht aufeinander abgestimmt sind und unterschiedliche Prioritäten verfolgen. Ohne eine klare Harmonisierung zwischen den beiden Perspektiven erfolgt in den Modellen ein zusätzlicher oder gar doppelter Pflegeaufwand und im schlimmsten Fall vollkommene Widersprüche (Harmon, 2019).

Die doppelte Modellierung erschwert die Synchronisation von strategischen und operativen Zielen, da Änderungen in einem Modell nicht automatisch im anderen reflektiert werden. Eine fehlende Integration kann zudem zu Entscheidungen führen, die nicht den tatsächlichen Anforderungen der Geschäftsprozesse oder der IT-Landschaft entsprechen. Die Notwendigkeit, eine solche automatisierte Integration zwischen den einzelnen Modellen zu schaffen, ist daher eine zentrale Motivation für diese Arbeit. Durch die Entwicklung eines integrierten Artefakts sollen die verschiedenen Modellierungssilos der Geschäfts- beziehungsweise Prozesssicht und der IT-Architektur miteinander kombiniert werden. Damit kann nicht nur eine „Single-Source-of-Truth“ etabliert, sondern auch die Interaktionen zwischen Geschäftsprozessen, Applikationen und Infrastruktur klar und nachvollziehbar dargestellt werden. Eine solche Integration ermöglicht es, das Unternehmen als kohärentes System zu betrachten und effizientere Entscheidungen über dessen Änderungen und Anpassungen zu treffen (M. M. Lankhorst et al., 2017).

1.2 Forschungsfrage und Zielsetzung

Um dem im vorherigen Unterkapitel geschilderten Problem des Aufkommens von Silos ohne ausreichende Interaktion zwischen der IT-Architektur und den Geschäftsprozessen entgegenzuwirken, wird eine explizite Forschungsfrage mit einer dementsprechenden Zielsetzung abgeleitet. Aus der Problemstellung wird ersichtlich, dass ein organisatorischer Dissens zwischen den Bereichen als auch Inkonsistenzen zwischen den einzelnen Modellen sowie deren doppelte Modellierung besteht. Der Rahmen dieser Arbeit fokussiert sich auf den Aspekt der Inkonsistenzen zwischen den Modellen der Unternehmensarchitektur und den Geschäftsprozessen.

Folglich soll eine Lösung erarbeitet werden, welche einen automatischen Abgleich der Modelle von Unternehmensarchitekturen und deren zugehörigen Geschäftsprozesse ermöglicht. Zusätzlich sollen die in den Modellen der Unternehmensarchitekturen enthaltenen operativen Geschäftsprozesse direkt instanziierbar sein. Dies erhöht die Effizienz der im Unternehmen verankerten Geschäftsprozesse, da diese über ein einzelnes Modell realisiert und direkt als konkrete Prozess-Instanz gestartet werden können.

Um jenen Anforderungen gerecht zu werden, wird ein konkretes Artefakt entwickelt, wessen methodische Vorgehensweise im nachfolgenden Unterkapitel detaillierter erläutert wird. Bevor aus dem Modell einer Unternehmensarchitektur die darin enthaltenen Geschäftsprozesse instanziiert werden können, muss ein Abgleich zwischen den unterschiedlichen Modellierungssprachen erfolgen. Demnach muss ein Mapping zwischen den einzelnen Konzepten der Unternehmensarchitektur und den Geschäftsprozessen etabliert werden. Dazu dient in dieser Arbeit ArchiMate als Modellierungssprache für Unternehmensarchitekturen und BPMN zur Visualisierung als auch nachfolgender Instanziierung der Geschäftsprozesse. Daher ist das erste Ziel des Artefakts zu beschreiben, wie ein syntaktischer sowie semantischer Vergleich zwischen den einzelnen Konzepten der beiden Sprachen erfolgt und wie folglich ein entsprechendes Mapping untereinander entwickelt werden könnte.

Aufbauend auf Basis dieses Mappings können BPMN-Modelle und demnach auch Prozess-Instanzen direkt aus der Unternehmensarchitektur gestartet werden. Dadurch soll dem Problem der Inkonsistenzen zwischen den einzelnen Architekturen und Modellen entgegengewirkt werden. Mit dem entwickelten Artefakt soll ein weiteres Ziel dieser Arbeit eruiert werden, nämlich welche neuen Möglichkeiten durch das Artefakt entstehen, aber auch welche Grenzen folglich gegeben sind. Mit diesen auf Basis der Problemstellung abgeleiteten Zielen wird die Forschungsfrage für die vorliegende Masterarbeit wie folgt formuliert.

Wie erfolgt ein syntaktischer sowie semantischer Vergleich zwischen den Konzepten von ArchiMate und BPMN, um darauf aufbauend ein Mapping zu implementieren, sodass BPMN-Prozesse innerhalb von ArchiMate-Architekturen instanziierbar werden und welche Möglichkeiten und Grenzen entstehen dadurch?

1.3 Methodische Vorgehensweise und Aufbau der Arbeit

Zur Erarbeitung und Beantwortung der zuvor gestellten Forschungsfrage sowie Zielsetzung dienen die Methoden der gestaltungsorientierten Forschung. In dieser ist abstrakt gesprochen ein gewisses Konstruktionsproblem gegeben (Österle, Winter, et al., 2010). Um ein solches Problem lösen zu können, wird auf die Erschaffung und Entwicklung von Artefakten gesetzt, welche Konstrukte, Modelle oder tatsächliche Implementierungen sein können (Hevner et al., 2004). Gemäß Vaishnavi und Küchler (2015) kategorisieren sich solche Artefakte in drei Abstraktionsstufen, nämlich der Gestaltungstheorie, Gestaltungswissen sowie der anwendungsspezifischen Instanziierungen.

Im Rahmen dieser Arbeit wird ein Artefakt als anwendungsspezifische Instanziierung umgesetzt, da ein technischer Prototyp konzeptioniert und folglich ausprogrammiert werden soll. Die Erarbeitung des Artefakts, beziehungsweise des technischen Prototyps, bedient sich des allgemeinen Erkenntnisprozesses der gestaltungsorientierten Forschung. In diesem wird beschrieben, wie das Artefakt entwickelt wird und zugleich eine wissenschaftliche Rigorosität aufweist (Österle, Becker, et al., 2010). Dazu wird ein Prozess mit vier Schritten durchlaufen, nämlich der initialen Analyse, Entwurf, Evaluation sowie Diffusion. Um diesen Prozess weiter zu vertiefen, als auch um eine konkrete Vorgehensweise aus der gestaltungsorientierten Forschung auszuwählen, wird in dieser Arbeit auf das Vorgehensmodell nach Peffers et al. (2007) gesetzt.

Das Vorgehensmodell, genauer gesagt die Design Science Research Methode, kurz DSRM, nach Peffers et al. (2007) besteht aus sechs Schritten, der Problemidentifikation & Motivation, Definition der Lösungsziele, Design & Entwicklung, Demonstration, Evaluation und Kommunikation. Beginnend mit der Problemidentifikation und der Motivation, wurde diese bereits in Kapitel 1.1 einleitend angeführt, um die Relevanz dieser Arbeit zu zeigen.

Um das zugrunde liegende Problem hier erneut kurz anzuführen, besteht ein Dissens zwischen den detailliert ausmodellierten Geschäftsprozessen in beispielsweise BPMN und der abstrakten Architekturbeschreibung in ArchiMate-Modellen, ohne einer eindeutigen „Single-Source-of-Truth“ (M. M. Lankhorst et al., 2017). Mit einem Artefakt, welches diese beiden Modellierungssilos miteinander kombiniert, könnte eine einheitliche Sicht sowie der tatsächliche Abgleich zwischen den Geschäftsprozessen und den dahinterliegenden Applikationen sowie Infrastruktur gegeben werden. Dies bietet den Mehrwert an, dass beispielsweise effiziente Entscheidungen über Änderungen in einem Unternehmen getroffen werden können, da die miteinander agierenden Komponenten und Prozesse klar ersichtlich sind (M. M. Lankhorst et al., 2017).

Im zweiten Schritt des Vorgehensmodells von Peffers et al. (2007) werden die Lösungsziele zur Erarbeitung des Problems definiert, welche erneut bereits teilweise zuvor in Kapitel 1.2 erfolgten. Anhand der Problemstellung wurde die Forschungsfrage abgeleitet und ausformuliert. Es wurde als übergeordnetes Ziel gesetzt, dass das Artefakt zum einen ein Mapping zwischen ArchiMate und BPMN inkludiert und zum anderen auf diesem aufbauend, Geschäftsprozesse in BPMN automatisiert von ArchiMate-Architekturen aus instanziiert werden können. Für die detaillierten und tatsächlich technisch realisierbaren Anforderungen des Artefakts dient das Kapitel 4.1.

Nach der Definition der Lösungsziele stehen folglich die Anforderungen für die Phase des Designs und der Entwicklung fest. In dieser wird das Artefakt selbst entwickelt (Peffers et al., 2007). Da das freie, ad-hoc Erarbeiten eines Artefaktes nicht den Kriterien der wissenschaftlichen Rigorosität erfüllen würde, muss der aktuelle Stand der Forschung und Technik berücksichtigt werden (Österle, Becker, et al., 2010). Dazu dient die Wissensbasis, welche die Grundlagen und Methoden der einzelnen Forschungsbereiche inkludiert (Hevner et al., 2004).

Als Einblick in die bereits vorhandene Wissensbasis erfolgt in Kapitel 2 eine theoretische Übersicht über den Bereich der Modellierungssprachen. Hierbei wird initial erläutert, was Modellierungssprachen sind und auf welche Bestandteile und Eigenschaften diese aufbauen. Dazu werden die abstrakten Begriffe und Konzepte der Modelle, Metamodelle, Syntax, Semantik sowie die Instanziierbarkeit von Modellen angeführt und beschrieben. Des Weiteren werden die konkreten Modellierungssprachen ArchiMate und BPMN inhaltlich zusammengefasst. Für die beiden Sprachen werden deren Einsatzbereiche sowie Historie prägnant umrissen als auch deren Struktur und Verwendung. Da es sich bei ArchiMate und BPMN um visuelle Modellierungssprachen handelt, werden einzelne Grundkonzepte mittels visueller Diagramme beispielhaft abgebildet.

Um von der allgemeinen Wissensbasis zum aktuellen Forschungsstand überleiten zu können, dient das Kapitel 3. In diesem wird der erste Teil des Artefakts, nämlich das Mapping zwischen ArchiMate und BPMN, erarbeitet. Dazu werden fünf Literaturquellen aufgearbeitet und gemeinsam aggregiert. Aus dieser zusammenfassenden Sicht werden Schlüsse für die tatsächliche Implementierung des ersten Teils des Artefakts gezogen. Anhand dieser Schlüsse als auch auf den erarbeiteten Inhalten der Wissensbasis aufbauend, wird ein eigenes, für den Anwendungsfall dieser Arbeit domänenspezifisches Mapping entwickelt. Hierbei unterscheidet sich das zu entwickelnde Mapping zu den bereits vorhandenen, indem insbesondere auf den Aspekt der Instanziierbarkeit eingegangen wird.

Mit dem definierten Mapping zwischen ArchiMate und BPMN gilt der erste Teil des angestrebten Artefaktes und der Design und Entwicklungsphase nach Peffers et al. (2007) als abgeschlossen. Der zweite Teil des Artefakts bezieht sich darauf, dass mittels des Mappings direkt aus ArchiMate-Modellen eigene BPMN-Prozesse instanziiert werden können. Mit dem entwickelten Artefakt soll sichergestellt werden, dass die Interaktion zwischen ArchiMate und BPMN nicht nur theoretisch, sondern auch praktisch umsetzbar ist. Dessen tatsächlich technischen Anforderungen werden wie bereits erwähnt in Kapitel 4.1 beschrieben. Die prototypische Programmierung des Artefaktes erfolgt in Kapitel 4.2, wodurch die gesamte Phase des Designs und Entwicklung als abgeschlossen gilt.

Mit dem entwickelten Artefakt wird die Phase der Demonstration gestartet. Peffers et al. (2007) führen an, dass durch die Nutzung des Artefakts gezeigt werden soll, dass mit diesem die erarbeiteten Probleme tatsächlich gelöst werden können. Analog zur Entwicklung wäre die ad-hoc Demonstration eines Modells nicht zielführend und nicht rigoros. Diesbezüglich werden auf standardisierte Workflow Patterns nach Van Der Aalst et al. (2003) gesetzt, um zu demonstrieren, welche Workflow Patterns in ArchiMate als BPMN-Modelle instanziierbar sind. Zur praxisbezogenen Darstellung des Artefakts dienen ausgewählte Praktiken aus der Information Technology Infrastructure Library, kurz ITIL (AXELOS, 2019).

Die vorletzte Phase im Vorgehensmodell von Peffers et al. (2007) ist die Evaluation, welche in Kapitel 5.2 sowie in Kapitel 5.3 erfolgt. In dieser soll bewertet werden, ob sich das anzustrebende Artefakt tatsächlich zur Lösung der Probleme eignet und an welche Grenzen dieses stößt. Da auf standardisierte Workflow Patterns und auf ein allgemein anerkanntes Framework gesetzt wird, können diese objektiv evaluiert und theoretisch mit anderen Lösungsansätzen verglichen werden. Zusätzlich können aus diesen Ergebnissen Schlüsse für einen iterativen Verbesserungsprozess des Artefakts gezogen werden. Dies erfolgt abschließend in den Schlussbetrachtungen in Kapitel 6. Um die Schlüsse und Erkenntnisse auch öffentlich darzustellen, sieht der DSRM Ansatz als letzte Phase deren akademische Publikation vor, was in Form dieser Masterarbeit erfolgt.

2 EIN EINBLICK IN DIE MODELLIERUNGSSPRACHEN ARCHIMATE UND BPMN

Die Kombination der beiden Artefakte ArchiMate und BPMN stellt den Mittelpunkt dieser Arbeit dar. Demnach dient dieses Kapitel als Einleitung und allgemeine Übersicht für die Einsatzbereiche und Grundlagen von ArchiMate sowie BPMN. ArchiMate dient zur Visualisierung von Unternehmensarchitekturen, um mittels einer minimalen Anzahl an verständlichen Piktogrammen holistische Unternehmensarchitekturen für verschiedene Stakeholder darzustellen (The Open Group Standard, 2022a). Dies inkludiert etwa die technische Server-Infrastruktur, die darauf aufbauende Applikations-Landschaft, als auch die Geschäftsprozesse, welche auf Basis dieser Komponenten realisiert werden. Für einen detaillierteren Einblick in die Geschäftsprozesse dient die Modellierungssprache BPMN, welche als Vorreiter im Bereich der Geschäftsprozessmodellierung gilt (Silver, 2011). Über BPMN können Prozesse nicht nur von technischen IT-Abteilungen, sondern auch von spezifischen Fachbereichen entwickelt und verstanden werden, was zu einer transparenteren Darstellung und Verständnis dieser führt (Object Management Group, Inc, 2013). Zusätzlich können BPMN-Prozesse technisch automatisiert und folglich instanziiert werden.

Bevor jedoch auf die beiden ikonografischen Modellierungssprachen eingegangen werden kann, müssen die allgemeinen Konzepte hinter einer Modellierungssprache erläutert werden. Daher beginnt dieses Kapitel mit den allgemeinen Bestandteilen und Eigenschaften von Modellierungssprachen. Dem folgend wird ArchiMate sowie BPMN im Detail beschrieben.

2.1 Bestandteile und Eigenschaften von Modellierungssprachen

Modellierungssprachen dienen abstrakt gesprochen als Werkzeugkasten für die Entwicklung und Abbildung von Modellen (Harel & Rumpe, 2003). Um dies zu ermöglichen, bestehen Modellierungssprachen aus diversen Bestandteilen, welche in den nachfolgenden Unterkapiteln erläutert werden. Initial muss der Begriff des Modells selbst definiert werden, welches allgemein gesprochen als Teilbild der Realität angesehen werden kann (Seidewitz, 2003).

Zusätzlich wird nach der allgemeinen Definition eines Modells auf das dahinterliegende Konzept verwiesen, wie Modelle konstruiert werden. Dazu muss die Modellierungssprache ein sogenanntes Metamodell aufweisen. Mittels des Metamodells kann die Verwendung der im Modell enthaltenen Konzepte auf deren korrekten Einsatz validiert werden, was als Syntax bezeichnet wird. Des Weiteren implizieren Modelle, beziehungsweise die enthaltenen Konzepte, eine gewisse Bedeutung, nämlich die sogenannte Semantik (Harel & Rumpe, 2003). Neben den allgemeinen Bestandteilen von Modellierungssprachen wird erweiternd auf die Instanziierbarkeit von konstruierten Modellen eingegangen. Das Konzept der tatsächlichen Instanziierung von Prozess-Modellen stellt einen elementaren Bestandteil dieser Arbeit dar.

2.1.1 Modell

Ein Modell wird als „Abstraktion der Realität nach einer bestimmten Konzeptionalisierung“ bezeichnet (Guizzardi, 2005, S. 1). Die Konzeptionalisierung selbst entspricht der Summe der einzelnen Konzepte in einem gewissen Bereich oder Domäne (Henderson-Sellers, 2012). Vereinfacht ausgedrückt, repräsentiert somit ein konkretes Modell einen Ausschnitt der Realität mit dem primären Fokus, die als „wichtig“ gekennzeichneten Aspekte verständlich darzustellen. Ein konkretes Modell kann folglich als Werkzeug für die Vermittlung von Informationen über einen Bereich dienen (Guizzardi, 2005).

Um ein Modell zu konstruieren, beziehungsweise abbilden zu können, wird ein geeignetes Mittel benötigt. Dafür dienen eigene Sprachen, genauer gesagt Modellierungssprachen. Auf Basis der definierten Modellierungssprachen werden textuelle oder visuelle Modelle unter der Einhaltung von bestimmten Regeln modelliert. Der Einsatzzweck von Modellierungssprachen dient folglich der Konstruktion von Modellen für bestimmte Bereiche und Aufgaben (Henderson-Sellers, 2012).

Ein Beispiel dafür ist in Abbildung 1 veranschaulicht, welches die modellhafte Darstellung einer Unternehmensarchitektur mit dem Einsatz der Modellierungssprache ArchiMate zeigt. Weitere Modellierungssprachen, welche explizit für den Bereich der Wirtschaftsinformatik relevant sind, sind unter anderem BPMN, UML, TOGAF oder SysML (Fleischmann et al., 2018). Im Rahmen dieser Arbeit wird primär auf ArchiMate sowie BPMN zur Entwicklung und Abbildung von Modellen gesetzt.

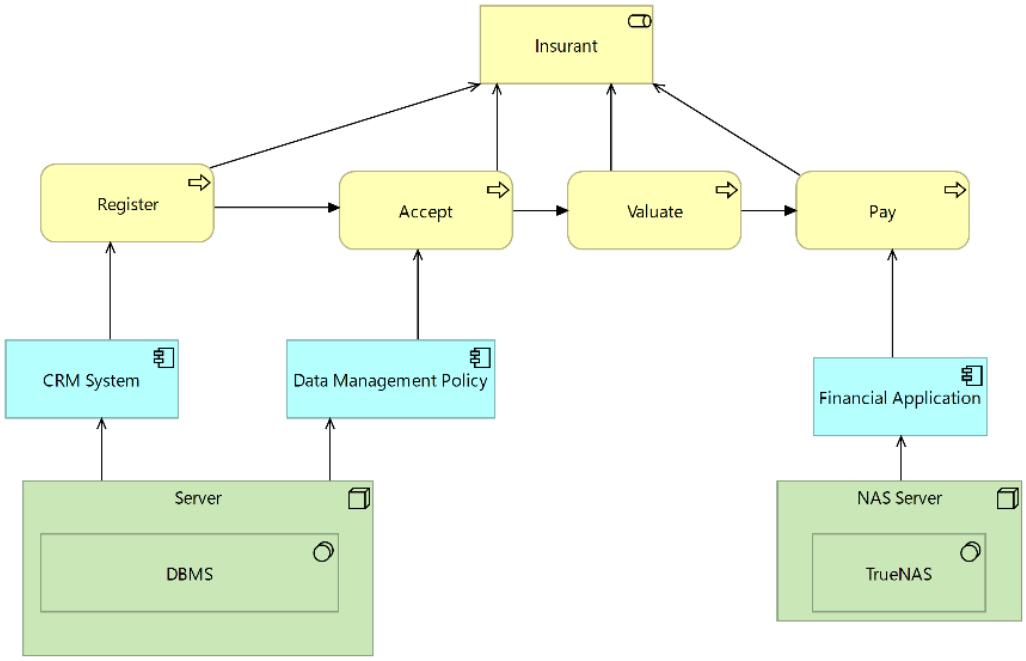

Abbildung 1: Modell einer Unternehmensarchitektur in ArchiMate, in Anlehnung an (LeanIX, 2023)

Damit ein Modell als ein solches anerkannt wird, müssen drei Hauptmerkmale erfüllt sein. Das erste Merkmal eines Modells ist das Abbildungsmerkmal. Das Abbildungsmerkmal besagt, dass das Modell selbst die Abbildung von „etwas“ ist. Der Begriff „etwas“ kann hierbei ein natürliches oder künstliches Original sein, als auch ein weiteres Modell. Es ist demnach möglich, mit einem

Modell ein anderes Modell abzubilden. Es muss jedoch stets eine Vorgabe oder ein Original gegeben sein, unabhängig von dessen Ausprägung. Ein alleinstehendes Modell, welches keinen Bezug zu einem Original oder einem anderen Konstrukt aufweist, ist nicht möglich. Zusätzlich können mehrere Modelle zu ein und demselben Original vorhanden sein (Stachowiak, 1973).

Das zweite Merkmal, welches ein Modell aufweisen muss, ist das Verkürzungsmerkmal. Im einleitenden Absatz dieses Unterkapitels wurde beschrieben, dass ein Modell gemäß Guizzardi (2005) einen Ausschnitt der Realität darstellt und sich auf die „wichtigen“ Aspekte oder Merkmale konzentriert. Dies beschreibt bereits implizit das Verkürzungsmerkmal. Es können nicht alle Aspekte und Merkmale des Originales oder der Realität abgebildet werden, lediglich jene, welche für den Einsatzzweck des Modells relevant sind. Ein Modell ohne Verkürzungsmerkmal kann nicht existieren und wäre auch nicht sinnvoll, da dies das Original selbst wäre. Welche Attribute oder Eigenschaften des Originales relevant sind, ist abhängig von der Verwendung und Zielpublikum des konkreten Modells (Stachowiak, 1973).

Mit dem zweiten Merkmal ist bereits indirekt das dritte Merkmal von Modellen, das pragmatische Merkmal, im weitesten Sinn gegeben. Dies bezieht sich darauf, wie die relevanten Attribute des Originals selektiert werden. Die Selektion selbst kann, wenn keine Spezifikation der pragmatischen Gesichtspunkte vorliegt, zufällig erfolgen, da die „wichtigen“ Aspekte nicht spezifiziert wurden. Wenn eine klare Zielsetzung und dahinterliegende Spezifikation für das Modell vorliegt, erfolgt die Selektion der Attribute nach einer strengen Zweckbestimmtheit. Das pragmatische Merkmal selbst beschreibt diesen Zweck oder was mit dem Modell erreicht werden soll. Das Modell dient als Ersatz des Originals für bestimmte Subjekte, in einem gewissen Zeitrahmen, sowie unter Einschränkungen (Stachowiak, 1973).

Vereinfacht ausgedrückt, wird mit dem pragmatischen Merkmal beschrieben, „für wen“, „wann“ und „wozu“ ein Modell dient. Anhand dieser Fragestellungen leiten sich die „wichtigen“ Attribute des Modells ab. Unterschiedliche Stakeholder haben differierende Anforderungen an ein Modell, was das „für wen“ widerspiegelt. Zusätzlich kann ein Modell für einen gewissen Zeitraum gültig sein, was operativ beispielsweise durch die Einführung einer Versionierung des Modells gekennzeichnet werden könnte. Die Frage des „wozu“ spiegelt die Effektivität des Modells wider. Ein Modell, welches den angeforderten Zweck nicht erfüllt und die falschen Attribute inkludiert, ist implizit unnötig und wertlos (Stachowiak, 1973).

2.1.2 Metamodell

Damit eine Modellierungssprache für die Modellierung von Modellen nutzbar ist, muss spezifiziert sein, welche Konzepte in der Modellierungssprache enthalten sind und wie diese eingesetzt werden dürfen. Dazu dienen sogenannte Metamodelle (Seidewitz, 2003). Der Begriff „Metamodell“ setzt sich aus dem Präfix „Meta“ sowie „Modell“ zusammen. Im vorherigen Unterkapitel wurde bereits erläutert, dass ein Modell eine abstrakte Darstellung eines Teilaspektes von einem Original ist (Stachowiak, 1973). Zum Begriff „Meta“ beschreibt Kühne (2006), dass eine gewisse Operation zweimal ausgeführt wird. Hierbei führt Kühne (2006) als

Beispiel an, dass die Diskussion darüber, wie Diskussionen geführt werden, als „Meta-Diskussion“ bezeichnet werden kann. Im für diese Arbeit relevanten Kontext wird auf Metamodelle im Bereich der Modellierung gesetzt. Nach der vorliegenden Begriffsdefinition werden für Metamodelle die Aspekte der Modellierung zweimal durchgeführt. Dies findet sich angedeutet in der Definition eines Metamodells nach dem Meta Object Facility, kurz MOF, Ansatz wieder (OMG, 2019).

„A metamodel is a model used to model modeling itself.” (OMG, 2019, S. 3)

Die Modellierung wird zweimal durchgeführt, einmal für das Modell selbst, welches einen konkreten Zweck erfüllt, und einmal als (Meta-)Modell, welches beschreibt, wie das Modell selbst modelliert werden kann. Der Ansatz oder Architektur des MOF-Frameworks findet sich insbesondere in der Informatik zur Softwaremodellierung wieder (Seidewitz, 2003). Da BPMN auf UML als Metamodell setzt (Silver, 2011), was sich wiederum nach dem MOF-Framework orientiert, und ArchiMate ein UML-ähnliches Metamodell besitzt (WIERDA, 2021), wird folglich das MOF-Framework und dessen Stufen erläutert.

Als Meta Object Facility wird das standardisierte Framework zur Spezifikation, Erstellung und Verwaltung von Modellierungssprachen und Metadaten bezeichnet. Das MOF-Framework wurde von der Object Management Group, kurz OMG, entwickelt und gewartet (OMG, 2019). Innerhalb des Frameworks wird auf Ebenen gesetzt, welche untereinander hierarchisch verbunden sind. Nach dem Standard müssen mindestens zwei Ebenen gegeben sein, ohne eine Obergrenze an verfügbaren Ebenen. In Abbildung 2 ist die vier Ebenen Architektur nach dem klassischen MOF-Ansatz dargestellt.

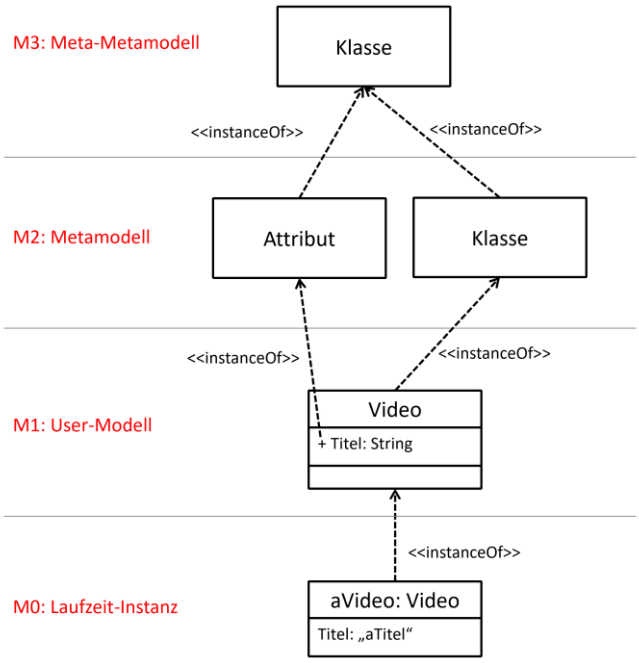

Abbildung 2: Vier Ebenen MOF-Architektur, in Anlehnung an (Object Management Group, 2011, S. 20)

Anhand der Abbildung 2 wird ersichtlich, dass die Elemente einer hierarchisch untergeordneten Ebene die Instanzen der direkt übergeordneten Ebene sind. Der Aspekt der Instanziierbarkeit wird nachfolgend in Kapitel 2.1.5 erweitert erläutert. Die hierarchisch gesehen höchste Ebene ist die Ebene M3, das Meta-Metamodell (OMG, 2019). Mit dem doppelten Präfix „Meta“ wird gezeigt, dass eine Operation, in diesem Fall die Modellierung, dreifach ausgeführt wird. Die Ebene M3 inkludiert die Spezifikation, um die Modellierungssprache selbst als Metamodell auszudrücken. In den Spezifikationen der Object Management Group ist dies das Meta Object Facility selbst (Seidewitz, 2003).

In der Ebene M2 befindet sich das Metamodell. Das Metamodell definiert die Modellierungssprache. Es liefert Aussagen darüber, welche Konzepte in einer Modellierungssprache enthalten sind und wie diese eingesetzt werden dürfen. Anhand des Metamodells kann ein Modell aus den darunterliegenden Ebenen syntaktisch validiert werden (Henderson-Sellers, 2012). Die Gesamtheit der Aussagen oder Regeln, welche für die Modellierungssprache erlaubt sind, wird als abstrakte Syntax bezeichnet (Seidewitz, 2003). Diese wird im nachfolgenden Unterkapitel detailliert untersucht. Im Fall von BPMN wird in der Ebene M2 zusätzlich die Semantik, die Bedeutung der Konzepte, definiert (Singer, 2019). Ein Einblick in die Semantik einer Modellierungssprache erfolgt in Kapitel 2.1.4.

Anhand des Metamodells aus der Ebene M2 werden konkrete Modelle in der Ebene M1 modelliert. Diese Modelle orientieren sich primär an der Darstellung einzelner zweckbezogener Anwendungen, welche für eine definierte Gruppe an Stakeholdern relevant sind (Poernomo, 2006). Zur Entwicklung der einzelnen Modelle sind diverse Programme und Editoren vorhanden, welche die im Metamodell enthaltenen Konzepte grafisch aufbereiten und auf einem Canvas visualisieren. Ein einfaches Modell aus der Abbildung 2 beschreibt eine Klasse namens „Video“ welche als Attribut einen „Titel“ des Typs „String“ aufweist (Object Management Group, 2011). Diese zwei Elemente sind jeweils Instanzen von der darüberliegenden, abstrakteren Ebene M2.

Die hierarchisch und visuell gesprochen unterste Ebene des klassischen MOF-Frameworks ist die Ebene M0, welche die Instanzen der Modelle aus M1 während der Laufzeit repräsentiert. Diese Instanzen können verschiedene Arten von Werten enthalten, abhängig von der spezifischen Modellierungssprache. Mögliche Ausprägungen sind Datenwerte, instanziierte Klassenobjekte oder Funktionsdefinitionen (Poernomo, 2006). Um erneut auf das Beispiel nach Abbildung 2 zurückzugreifen, würde konkret während der Ausführung des Modells ein Klassenobjekt vom Typ „Video“ instanziiert werden, was die dynamische Repräsentation des Modells auf der M0-Ebene darstellt.

2.1.3 Syntax

Die Syntax ist ein essenzieller Bestandteil einer jeglichen Modellierungssprache. Mittels dieser wird der valide Einsatz der enthaltenen Konzepte definiert. Hierbei kann der Begriff eines Konzeptes, je nach Modellierungssprache, mit unterschiedlichen Begriffen substituiert werden (Harel & Rumpe, 2003). So könnten beispielsweise die Terminologien „Wörter“, „Sätze“,

„Diagramme“, „Konstrukte“ oder ähnliches eingesetzt werden. Welche Terminologie konkret verwendet wird, ist abhängig vom Typ der Sprache. Demnach würden „Wörter“ und „Sätze“ die primären syntaktischen Elemente einer textuellen Sprache darstellen, während auf „Symbole“ oder „Diagramme“ im Bereich der ikonografischen Sprachen gesetzt wird. Aus einer abstrakten Sichtweise betrachtet, kann die Syntax einer Sprache zusammenfassend als „Grammatik“ bezeichnet werden, unabhängig von der tatsächlichen Ausprägung des verwendeten Begriffs (Harel & Rumpe, 2004).

Welche Konzepte oder Konstrukte in einer Modellierungssprache enthalten sind, wird in dessen Metamodell definiert. Wie im vorherigen Unterkapitel beschrieben, ist in diesem ein Regelwerk abgebildet, welches definiert, wie die Konstrukte miteinander verbunden werden dürfen (Seidewitz, 2003). Diese endliche Anzahl an Konzepten, Konstrukten oder etwaigen Begriffen, sowie dessen Regelwerk wird als abstrakte Syntax bezeichnet. Die abstrakte Syntax stellt den ersten Teilbereich der Syntax dar. Sie ist unabhängig von der tatsächlichen Notation, welche als konkrete Syntax bezeichnet wird und den zweiten Teilbereich inkludiert (Hogrebe & Nüttgens, 2009).

Mittels der abstrakten Syntax werden somit die allgemeingültigen Konstrukte und Konstrukt-Verbindungen definiert, ohne durch die visuelle Repräsentation beeinflusst zu werden. Dadurch ist die Möglichkeit gegeben, unterschiedliche Notationen, beziehungsweise eine alternative konkrete Syntax einzusetzen, welche jedoch dem gleichen Konzept der abstrakten Syntax zugeordnet sind (Hogrebe & Nüttgens, 2009). Dies ist folglich in Abbildung 3 dargestellt.

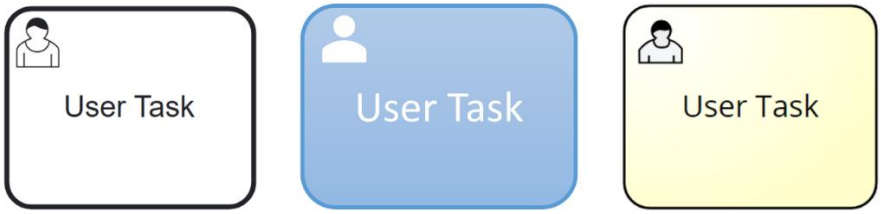

Abbildung 3: Unterschiedliche konkrete Syntax eines BPMN User Tasks, eigene Darstellung

Nach der Abbildung 3 wird das Konzept eines BPMN „User Tasks“ mittels drei verschiedener Notationen dargestellt. Die dahinterliegende abstrakte Syntax ist jedoch identisch. Folglich besteht eine Abhängigkeit von der konkreten Syntax zur abstrakten Syntax (Harel & Rumpe, 2004). Die visuelle Repräsentation der einzelnen Konstrukte kann demnach, wie in Abbildung 3 demonstriert, je nach der eingesetzten Modellierungssoftware unterschiedlich dargestellt sein. Hierbei muss jedoch die Verknüpfung zu den Konzepten der abstrakten Syntax gegeben sein, und die Notation darf dem definierten Regelwerk nicht widersprechen (Harel & Rumpe, 2004).

Die Art der Notation, oder konkrete Syntax, unterscheidet sich nach dem Typ der Modellierungssprache. Für textuelle Sprachen erfolgt der Einsatz von einzelnen Symbolen, welche primär linear und nach einer in der abstrakten Syntax definierten Reihenfolge gesetzt werden. Im Bereich der ikonografischen Modellierungssprachen dienen Piktogramme sowie Pfeile als gängige Notation, welche zusätzlich zweidimensional angeordnet werden können (Harel & Rumpe, 2004). Als Beispiel dafür dient Abbildung 4, welche die abstrakte Syntax als auch die konkrete Syntax textuell und visuell exemplarisch beschreibt.

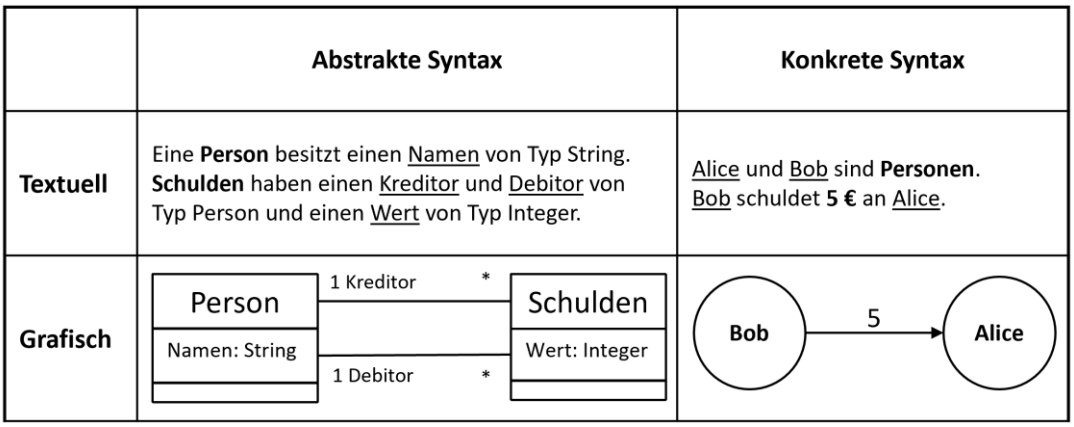

Abbildung 4: Art der Syntax, in Anlehnung an (Génova, 2009, S. 24)

Im linken oberen Quadranten von Abbildung 4 werden die zwei Konstrukte „Person“ und „Schulden“ sowie deren Konstrukt-Verbindung mittels Text in der deutschen Sprache beschrieben. Eine syntaktisch valide „Person“ muss einen „Namen“ von Typ String aufweisen, während die „Schulden“ zwei „Personen“ und einen numerischen Wert inkludieren müssen. Eine mögliche konkrete Notation ist im rechten oberen Quadranten enthalten. Es werden Alice sowie Bob als „Personen“ klassifiziert und Bob schuldet Alice einen numerischen Betrag von fünf Euro.

Analog zur textuellen Ausprägung zeigt Abbildung 4 dieselbe abstrakte Syntax, welche in diesem Fall durch ein UML-Klassendiagramm definiert wird. Das Klassendiagramm inkludiert die beiden Klassen „Person“ und „Schulden“, welche visuell durch Rechtecke repräsentiert werden. Um die syntaktisch validen Beziehungen zwischen den beiden Klassen zu definieren, dient das UML-Element der „Assoziation“ (Object Management Group, 2011). Zusätzlich wird die erlaubte Multiplizität mit den unteren und oberen Schranken angegeben. Hierbei muss für die Klasse „Schulden“ ein „Kreditor“ und ein „Debitor“ gegeben sein. Als visuelle, konkrete Syntax dient das Diagramm im rechten unteren Quadranten. Die Klasse „Person“ wird mittels eines Kreises sowie dem darin enthaltenen Namen abgebildet. Die „Schulden“, und damit repräsentativ die Verbindung zwischen Alice und Bob, erfolgt über einen gerichteten Pfeil.

2.1.4 Semantik

Mit der im vorherigen Unterkapitel dargestellten Syntax werden die enthaltenen Konzepte sowie deren validen Konstrukt-Verbindungen für eine Modellierungssprache definiert (Harel & Rumpe, 2003). Mit dem autarken Einsatz der Syntax können syntaktisch korrekte Modelle entwickelt werden. Diese lediglich auf die Syntax bezogenen Modelle liefern jedoch keine Aussage über deren Bedeutung oder Intention. Dazu wird ein weiterer wichtiger Bestandteil jeder Modellierungssprache benötigt, nämlich die Semantik (Seidewitz, 2003).

Die Semantik einer Modellierungssprache definiert die Bedeutung oder Aussagekraft der enthaltenen Konzepte (Harel & Rumpe, 2003). Mit der Semantik werden diese Konstrukte an Bedeutung bereichert, damit die entwickelten Modelle von den Zielgruppen inhaltlich verstanden werden. Hierbei besteht insofern das Problem, dass die abgebildeten Konzepte subjektiv

interpretiert werden (Fleischmann et al., 2018). Um die subjektiven Differenzen in der Interpretation zu eliminieren oder zumindest zu minimieren, muss die Semantik, analog zur Syntax, rigoros definiert sein. Um dies zu ermöglichen, wird die Semantik in zwei Teilbereiche untergliedert, die semantische Domäne sowie das semantische Mapping (Harel & Rumpe, 2003).

Die semantische Domäne definiert den Bereich oder Rahmen, in welchem sich die enthaltenen Konstrukte befinden (Harel & Rumpe, 2003). Ohne einer klaren Abgrenzung, welchem Bereich die definierten Konstrukte einer Modellierungssprache zuzuordnen sind, können Missverständnisse in deren Interpretation auftreten. Um die semantische Domäne beispielhaft zu demonstrieren, dient die Abbildung 5. Auf der linken sowie rechten Seite der Abbildung wird die numerische Zahl „10“ dargestellt. Obwohl die visuelle Notation dieser Zahl identisch ist, wird eine unterschiedliche Bedeutung vermittelt. Um die Bedeutung der Zahl einzugrenzen, wird die semantische Domäne definiert, welche für die Aussage 10 als $S = \mathbb{N}$ angeführt werden könnte. Die semantische Domäne S der Sprache wird im Rahmen der natürlichen Zahlen eingeschränkt (Harel & Rumpe, 2003).

Alternativ dazu wird die Zahl auf der rechten Seite von Abbildung 5 mit einem roten Rahmen umrandet. Auf Basis von bereits etabliertem Wissen und Erfahrungen kann angenommen werden, dass für diese Notation die semantische Domäne der Straßenverkehrsordnung interpretiert wird. Eine solche Visualisierung wird als semantisch transparent bezeichnet, da die eingesetzte Notation die korrekte Bedeutung impliziert (Petre, 1995).

Abbildung 5: Beispiel für unterschiedliche semantische Domänen, eigene Darstellung

Die Bedeutung der Zahl 10 nach Abbildung 5 ist folglich abhängig von der dahinterliegenden semantischen Domäne. Dass die semantischen Domänen zum einen die natürlichen Zahlen und zum anderen die Straßenverkehrsordnung sind, wurde im vorherigen Absatz lediglich vermutet. Es muss jedoch stets eine eindeutig definierte Zuweisung zwischen den implementierbaren Konzepten und der einschränkenden, semantischen Domäne gegeben sein. Um diese Korrelationen abzubilden, dient der zweite Teilbereich der Semantik, das semantische Mapping.

Als semantisches Mapping wird die Zuordnung zwischen den syntaktischen Konstrukten zur semantischen Domäne definiert. Mit dem semantischen Mapping ist der Konnex für die Bedeutung oder Aussage einer visuellen Darstellung gegeben. Das Mapping selbst kann unter variierender Formalität erfolgen (Harel & Rumpe, 2003). Für das Beispiel der natürlichen Zahl in

Abbildung 5 würde ein Mapping M mathematisch als $M: \langle Exp \rangle \rightarrow \mathbb{N}$ definiert werden. In Bezug auf das Verkehrszeichen der Geschwindigkeitsbegrenzung erfolgt das Mapping in der deutschen Sprache nach § 52 10a StVO.

2.1.5 Instanziierbarkeit

Im Unterkapitel 2.1.2 wurde in Abbildung 2 die Struktur des Meta Object Facility Frameworks gezeigt. In diesem werden hierarchisch aufgebaute Ebenen dargestellt, welche Elemente enthalten. Diese Elemente spiegeln Instanzen der übergeordneten Ebene wider. (OMG, 2019). Um auf die Terminologie des Begriffes der „Instanz“ weiter einzugehen, wird auf das Werk der Design Patterns nach Gamma (1995) zurückgegriffen, welches als Standardwerk in der objektorientierten Programmierung angesehen wird.

In der Definition des MOF-Frameworks wird erweitert beschrieben, dass die enthaltenen Elemente in den Ebenen Klassen und Objekte darstellen. Gemäß Gamma (1995) können Objekte als Bündel von Daten und deren möglichen Operationen gesehen werden. Die Definition, welche Daten und Operationen für Objekte möglich sind, erfolgt über Klassen. Eine Klasse kann als übergeordneter Bauplan eines Objektes verstanden werden. Aus der Klasse wird folglich ein Objekt instanziiert, welches mit dem Term „Instanz“ versehen wird. Dies bedeutet, dass bei der Instanziierung einer Klasse die Zuweisung von Speicherplatz für die internen Daten des Objektes erfolgt und diese mit den Methoden der Klasse verknüpft werden. Eine Klasse besitzt hierbei Instanzvariablen, welche sich zwischen den Objekten unterscheiden. Dadurch können aus einer Klasse mehrere Objekte instanziiert werden, welche unterschiedliche interne Daten aufweisen, jedoch auf dieselben, in der Klasse definierten, Operationen zugreifen (Gamma, 1995).

Um die Instanziierung eines Objektes aus einer Klasse zu veranschaulichen, dient die folgende Abbildung 6. Diese zeigt ein ähnliches UML-Klassendiagramm zu dem aus Abbildung 4, wobei der Klasse „Person“ die Methoden „gehen“ und „schlafen“ hinzugefügt wurden. Aus der einen Klasse werden die beiden konkreten Objekte „Alice“ und „Bob“ instanziiert. Anhand der Instanzvariable „Namen“ ist ersichtlich, dass diese mit zwei unterschiedlichen Werten befüllt wurde. Dies geschieht während des Prozesses der Instanziierung (Gamma, 1995). Da „Alice“ sowie „Bob“ als Objekte aus der Klasse „Person“ instanziiert wurden, besitzen beide deren definierten Methoden. Sowohl „Alice“ als auch „Bob“ können „gehen“ und „schlafen“ ausführen.

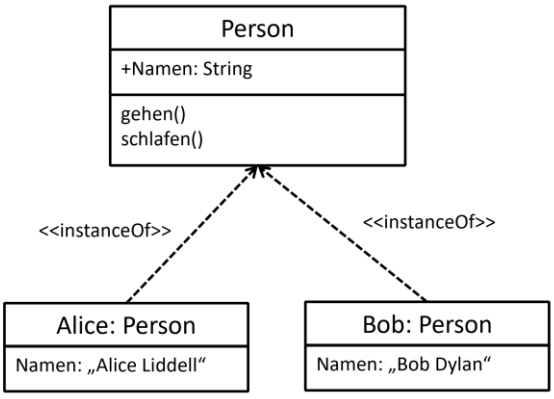

Abbildung 6: Instanziierte Objekte aus einer Klasse, eigene Darstellung

In diesem Beispiel nach Abbildung 6 wurde die Grundlage der Instanziierung in der objektorientierten Softwareentwicklung gezeigt. Um dieses Konzept auf den Bereich der Ausführung von Geschäftsprozessen anzuwenden, wird wieder auf das MOF-Framework aus Abbildung 2 zurückgegriffen. Innerhalb der Ebene M2 wird die Syntax und Semantik für die Modellierung des Prozess-Modells definiert. Hierbei müssen diese bereits die Möglichkeiten für die konkrete Ausführung von Prozessen inkludieren (Weske, 2019).

BPMN wäre beispielsweise eine Modellierungssprache, welche eine formale Definition für die Ausführung von Geschäftsprozessen bietet (Object Management Group, Inc, 2013). Mit der Ebene M1 werden Prozess-Modelle abgebildet, welche Ausführungsbeschränkungen der darauffolgenden Prozess-Instanzen definieren. Mit den Prozess-Modellen wird geregelt, in welcher Reihenfolge und nach welchen weiteren Restriktionen diese ausgeführt werden können. Gegeben sei hier ein fiktives Prozess-Modell, welches aus den Aktivtäten A und B besteht. Aus dem Prozessmodell wird eine Prozess-Instanz erzeugt, welche unter einer detaillierteren Betrachtung die Aktivitätsinstanzen für A und B besitzt. Im Prozessmodell wird $A \rightarrow B$ modelliert, wodurch diese Ausführungsbeschränkung besagt, dass die Aktivitätsinstanz von B erst starten kann, wenn die Aktivitätsinstanz von A erfolgreich terminiert wurde. Die Modelle der Ebene M1 geben demnach vor, wie der Sequenzfluss in der Prozess-Instanz zu erfolgen hat (Weske, 2019).

2.2 ArchiMate

ArchiMate ist eine ikonografische Modellierungssprache zur strukturierten Visualisierung von Unternehmensarchitekturen. Die Modellierungssprache wurde im Zuge eines niederländischen Forschungsprojekts entwickelt und im Jahr 2009 von der Open Group übernommen (M. M. Lankhorst et al., 2010). Diese wartet und adaptiert den ArchiMate Standard kontinuierlich, mit der zum Zeitpunkt des Schreibens dieser Arbeit, aktuellen Version 3.2 (The Open Group Standard, 2022a).

Einleitend zu diesem Unterkapitel werden die Einsatzbereiche von ArchiMate dargestellt, da der Begriff der Unternehmensarchitektur selbst ein größeres Spektrum an Bereichen abdeckt und unterschiedlich definiert werden kann (Saint-Louis et al., 2019). Zusätzlich werden die generellen Designkriterien und das abstrakte Metamodell von ArchiMate als Modellierungssprache dargestellt. Auf diese Designkriterien und allgemeine Struktur aufbauend, wird das sogenannte ArchiMate „Core Framework“ beschrieben.

Im „Core Framework“ sind die einzelnen Architekturen eines Unternehmens aus Technik-, Applikations- und Prozesssicht enthalten (The Open Group Standard, 2022a). Für eine erweiterte Sichtweise, welche auch die gesamte Strategie und Motivation eines Unternehmens abbildet, dient das „Full Framework“, welches im darauffolgenden Unterkapitel beschrieben wird. Aufgrund der Einheitlichkeit und Übersicht wurden die englischen Begriffe der ArchiMate Modellierungssprache gemäß dem offiziellen Glossar (2022a) der Open Group in die deutsche Sprache übersetzt.

2.2.1 Einsatzbereiche von ArchiMate zur Darstellung von Unternehmensarchitekturen

Mittels ArchiMate werden Unternehmensarchitekturen modelliert, welche einen holistischen Überblick eines Unternehmens ermöglichen (M. M. Lankhorst et al., 2010). Hierbei ist jedoch anzumerken, dass der Begriff der „Unternehmensarchitektur“ keine eindeutige und allgemein anerkannte Definition aufweist (Saint-Louis et al., 2019). Dazu führten Saint-Louis et al. (2019) eine strukturierte Literaturrecherche durch, um die unterschiedlichen Definitionen dieses Begriffes zu analysieren. Es wurden 160 Definitionen gesammelt und miteinander verglichen, welche in deren Detaillierungsgrad unterschiedlich ausgeprägt waren, jedoch primär Adaptionen von einzelnen, historischen Definitionen sind. Darunter enthalten sind beispielsweise die Werke nach Lankhorst (2013), Ross et al. (2006), Schekkerman (2006) sowie Zachman (1997).

Zusammenfassend beschreiben diese Definitionen die Unternehmensarchitektur als „Master Plan“ oder Gesamtheit aller wesentlichen Strukturen und Beziehungen der Schlüsselelemente, welche das Unternehmen ausmachen (Saint-Louis et al., 2019). Als konkretes Beispiel wird die Definition von Lankhorst (2013) angeführt, welcher den Begriff der Unternehmensarchitektur wie folgt aufschlüsselt *„coherent whole of principles, methods, and models that are used in the design and realisation of an enterprise’s organisational structure, business processes, information systems, and infrastructure“* (S. 3).

Mit der Anzahl an diversen Definitionen musste schließlich für ArchiMate eine spezifische gewählt werden. ArchiMate selbst setzt hierbei nicht nur auf eine Definition, sondern orientiert sich an einem gesamten Framework namens TOGAF (The Open Group Standard, 2022a). TOGAF ist die Abkürzung für „The Open Group Architecture Framework“ und wurde analog zu ArchiMate von der Open Group entwickelt und verwaltet. Als Architekturframework bietet TOGAF ein strukturiertes Vorgehen an, wie Unternehmensarchitekturen geplant, entwickelt, implementiert und verwaltet werden (The Open Group, 2022b). TOGAF definiert hierbei die sogenannte „Architecture Development Method“, kurz ADM, welche einen iterativen Prozess zur Entwicklung und Verbesserung der Unternehmensarchitektur darstellt (Tupper, 2011).

Mittels TOGAF sollen die Anforderungen und Bedürfnisse der einzelnen Zielgruppen abgebildet und nach Best Practices für den aktuellen Stand, als auch vorausschauend für die Zukunft implementiert werden (The Open Group, 2022b). Zusammenfassend dient TOGAF als theoretische Grundlage, wie Unternehmensarchitekturen entwickelt werden und was in diesen inkludiert ist, während ArchiMate selbst als Werkzeug zur Modellierung und Visualisierung dient. Die mittels ArchiMate entwickelten Modelle dienen als zentraler „Hub“, um die in den vorherigen Absätzen beschriebene gesamtheitliche Sicht der Unternehmensarchitektur zu visualisieren. Demnach werden in den ArchiMate-Modellen die Bereiche der Geschäftsprozesse, Strategie, IT-Infrastruktur oder Softwarearchitekturen abstrakt modelliert. Deren detaillierte Abbildung erfolgt schließlich in den einzelnen domänenspezifischen Modellierungssprachen (M. M. Lankhorst et al., 2017).

2.2.2 Designkriterien und Struktur von ArchiMate

Die primäre Grundfunktion von ArchiMate ist die Modellierung von Unternehmensarchitekturen. Die Unternehmensarchitektur stellt hierbei die höchste Architekturebene des Unternehmens dar, weshalb ArchiMate diese auf eine holistische und abstrakte Weise abbilden soll. Trotzdem muss eine gewisse Detailtiefe gegeben sein, jedoch nicht in der spezifischen Ausprägung wie bei beispielsweise BPMN für Geschäftsprozesse (Gericke, 2017). Um diesen Anforderungen gerecht zu werden, wurden mehrere Designkriterien an ArchiMate als Modellierungssprache gestellt (M. M. Lankhorst et al., 2010). ArchiMate ist eine Modellierungssprache und muss daher alle Aspekte und Bestandteile einer solchen nach Kapitel 2.1 erfüllen.

Ein weiteres Designkriterium von ArchiMate ist die Implementierung des Minimalprinzips. Im ArchiMate Standard selbst wird darauf verwiesen, dass mit der enthaltenen Anzahl von Konzepten 80 % aller Modellierungsanforderungen erfüllbar sein sollen (2022a). Trotz der limitierten Anzahl an Konzepten sollen die unterschiedlichen Perspektiven und Anforderungen der einzelnen Zielgruppen an die Unternehmensarchitektur modellierbar sein. Dies inkludiert die Abbildung der tatsächlich vorhandenen Technologien und Applikationen, als auch die Visualisierung von Prozessen und deren dahinterliegenden strategischen Ziele und Intentionen (M. M. Lankhorst et al., 2010).

In Bezug auf die Struktur, genauer gesagt das dahinterliegende Metamodell, orientiert sich ArchiMate an UML und weist daher Ähnlichkeiten und teils auch Überschneidungen zu UML auf (Gericke, 2017). Abbildung 7 zeigt folglich die übergeordnete Struktur der ArchiMate Modellierungssprache. Die in der Grafik visualisierten Elemente beziehen sich auf abstrakte Konzepte. In der Modellierung wird daher auf explizit definierte Konkretisierungen dieser abstrakten Konzepte gesetzt. Zusätzlich ist anhand der Abbildung 7 zu erkennen, dass ArchiMate auf einer hierarchischen Struktur basiert. Ein Modell stellt hierbei eine Ansammlung von Konzepten dar. Die Gesamtheit der modellierten Konzepte ergibt somit ein Modell. Ein Konzept selbst kann entweder ein Element, eine Beziehung oder ein Beziehungskonnektor sein (The Open Group Standard, 2022a).

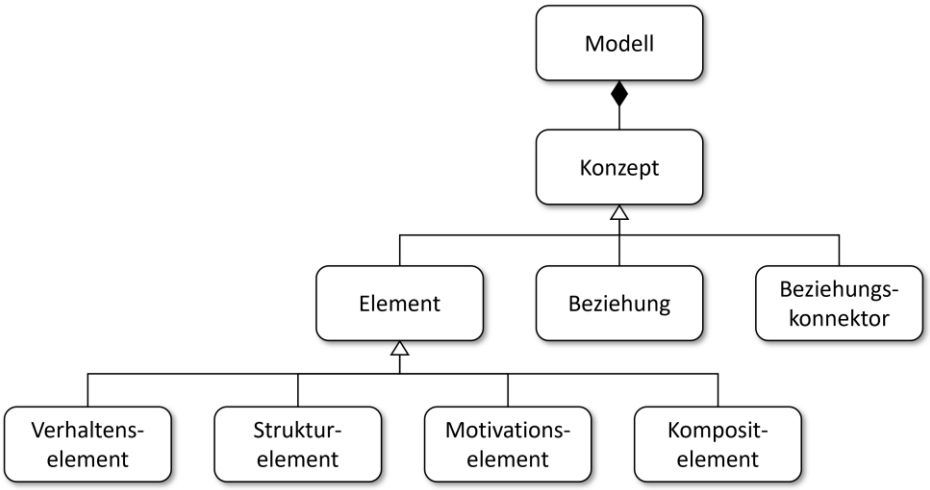

Abbildung 7: ArchiMate Metamodell, in Anlehnung an (The Open Group Standard, 2022a, S. 7)

Das Element ist ein weiterer Überbegriff für hierarchisch untergeordnete Konzepte. Diese sind das Verhaltens-, Struktur-, Motivations- und Kompositelement. Die Verhaltenselemente definieren und visualisieren das Verhalten der anderen Elemente. Mittels dieser kann modelliert werden, welche Elemente welches Verhalten aufweisen oder ausführen. Dadurch ist implizit eine gewisse Dynamik in der Abbildung von sonst starren Modellen möglich (The Open Group Standard, 2022a).

Damit ein Verhalten ausgeübt werden kann, sind Akteure notwendig. Diese werden durch Strukturelemente repräsentiert, die in aktive und passive unterteilt werden. Ein aktives Strukturelement kann ein Verhalten auf ein passives ausführen. Motivationselemente ergänzen die übrigen Konzepte, sind jedoch keiner konkreten Ressource oder Technologie direkt zuordenbar. Sie stehen für das „Warum“ beziehungsweise den Zweck der Architektur. Da sie die Architektur nicht direkt modellieren, sind sie nicht Teil des ArchiMate „Core Frameworks“, sondern des „Full Frameworks“. Dies wird in den zwei folgenden Unterkapiteln näher betrachtet. Die letzte Ausprägung der Elemente sind die Kompositelemente, die als Aggregation mehrerer Konzepte fungieren. Sie fassen nicht nur Elemente, sondern auch andere Konzepte wie Elemente mit Beziehungen zusammen (The Open Group Standard, 2022a).

Um Konzepte miteinander zu verbinden, werden Beziehungen eingesetzt. Eine Beziehung kann eine definierte Gruppe an Quellelementen mit Zielelementen verbinden. Zusätzlich sind in bestimmten Fällen auch Beziehungen mit anderen Beziehungen verknüpfbar. Durch den Einsatz von unterschiedlichen Beziehungen kann die Struktur, das Verhalten und Abhängigkeiten einer Unternehmensarchitektur modelliert werden (The Open Group Standard, 2022a). Abstrakt formuliert definieren sie, welche Elemente wie miteinander korrelieren und ermöglichen eine dynamische Sichtweise. Selbst in der Abbildung 7 wurden bereits Beziehungen eingesetzt. Eine Kompositionsbeziehung definiert, dass sich ein Modell aus mehreren Konzepten zusammensetzt und sich ein Konzept des Weiteren in drei Konzepte detaillierter spezialisiert.

Die letzte Ausprägung der Konzepte sind die Beziehungskonnektoren. Sie bestehen aus dem „And“ und „Or Verbindungspunkt“. Ein Verbindungspunkt wird eingesetzt, um Beziehungen miteinander zu verknüpfen. In der Modellierung weisen die Verbindungspunkte eine ähnliche Funktion wie die der logischen Gatter auf. Sie besitzen einen definierten Input, wodurch sich ein Output ergibt (The Open Group Standard, 2022a). Als Beispiel für einen „Or Verbindungspunkt“ wird in Abbildung 8 ein Zutritt beantragt und dieser entweder genehmigt oder abgelehnt. Der „Or Verbindungspunkt“ modelliert somit eine Entscheidung in der Architektur.

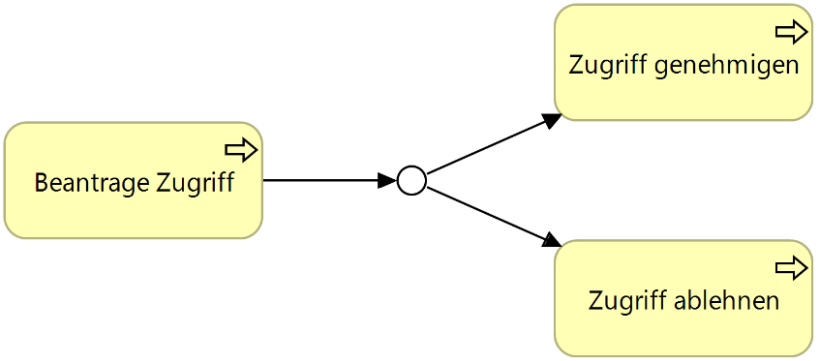

Abbildung 8: Or Verbindungspunkt, in Anlehnung an (The Open Group Standard, 2022a, S. 36)

2.2.3 Aufbau nach dem ArchiMate Core Framework

Mittels ArchiMate können die verschiedenen Sichtweisen von diversen Stakeholdern visualisiert werden. Dafür ist das sogenannte ArchiMate „Core Framework“ gegeben, welches den Aufbau des Unternehmens selbst repräsentiert. Dies gelingt durch die Unterteilung der Architektur in einzelne Ebenen sowie Aspekte, wie folglich in Abbildung 9 gezeigt. Die Ebenen stellen einen strukturierten Aufbau dar, bei welchem Elemente aus einer übergeordneten Ebene auf die Elemente der darunterliegenden Ebenen aufbauen. Mittels der Ebenen wird folglich die Unternehmensarchitektur aus einer technologischen, applikationsbezogenen, sowie geschäftsorientierten Sichtweise betrachtet. Zusätzlich zu den Ebenen erfolgt eine Gliederung der Elemente in Aspekte, welche sich nach dem Metamodell aus dem vorherigen Unterkapitel orientieren (The Open Group Standard, 2022a).

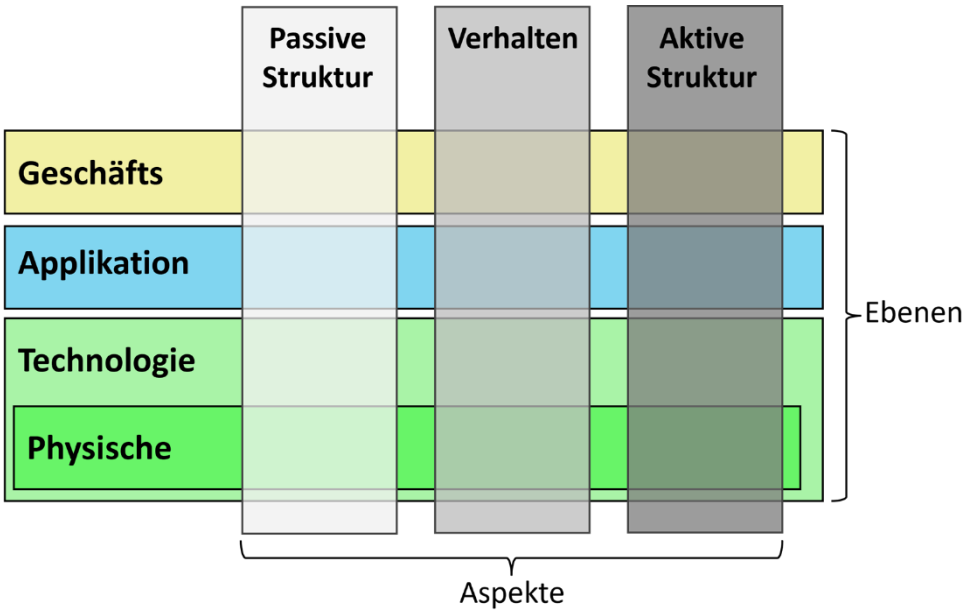

Abbildung 9: ArchiMate Core Framework, in Anlehnung an (The Open Group Standard, 2022a, S. 8)

Die unterste Ebene im ArchiMate „Core Framework“ nach Abbildung 9 ist die Technologieebene. Innerhalb dieser Ebene wird die Technologiearchitektur des Unternehmens modelliert. Demnach bildet sie das Verhalten und Zusammenspiel der einzelnen technischen Komponenten ab. Mögliche Konzepte aus dieser Ebene wären Geräte, Kommunikations-Netzwerke oder Systemsoftware. Das Konzept der Geräte dient beispielsweise zur Darstellung physischer IT-Ressourcen, welche eine bestimmte Systemsoftware verwenden und über Kommunikations-Netzwerke Daten senden sowie empfangen können (The Open Group Standard, 2022a).

Seit der ArchiMate Version 2 wurde die Technologieebene mit der physischen Ebene ergänzt (WIERDA, 2021). Sie soll analog zu einzelnen Elementen aus der Technologieebene die physische Welt darstellen, jedoch in einer nicht technisch-orientierten Sichtweise. Ein Element aus der physischen Ebene ist beispielsweise die Einrichtung. Eine solche Einrichtung könnte ein Gebäude sein, etwa eine Produktionshalle oder ein Lagerplatz (The Open Group Standard, 2022a).

Auf die Technologieebene aufbauend, werden in der Applikationsebene Elemente modelliert, welche die Struktur, das Verhalten sowie die Interaktion der einzelnen Anwendungen des Unternehmens beschreiben. Die Anwendungen dienen dazu, um die Geschäftsprozesse des Unternehmens zu realisieren, welche in ArchiMate als Applikationskomponenten modelliert werden. Eine Applikationskomponente ist eine in sich selbst abgeschlossene Einheit, welche eine gewisse Funktionalität anbietet und ersetzbar ist (The Open Group Standard, 2022a). Ein mögliches Beispiel dafür wäre das Textverarbeitungsprogramm „Microsoft Word“ welches durch die alternative Anwendung „LibreOffice Writer“ ersetzt werden könnte.

Die oberste Ebene des ArchiMate „Core Framework“ bildet die Geschäftsebene. Sie nutzt Konzepte aus der Applikations- und Technologieebene, um Geschäftsservices für definierte Zielgruppen bereitzustellen. Diese Services werden durch Geschäftsprozesse realisiert, die sich primär auf operative Unternehmensabläufe beziehen und aufbauend von konkreten Technologien modelliert werden (The Open Group Standard, 2022a).

Neben den drei Ebenen werden im ArchiMate „Core Framework“ nach Abbildung 9 zusätzlich drei Aspekte definiert. Diese erstrecken sich über alle Ebenen und sind folglich in jeder einzelnen vorhanden. Ausgeschrieben sind dies der passive Strukturaspekt, Verhaltensaspekt sowie der aktive Strukturaspekt. Die Aspekte richten sich nach den im vorherigen Unterkapitel 2.2.1 beschriebenen Verhaltenselementen und Strukturelementen, welche in Abbildung 7 schematisch visualisiert wurden. Im ArchiMate Standard (2022a) wurden die drei Aspekte an die natürliche Sprache angelehnt und können in der Form eines semantischen Tripletts notiert werden. Ein semantisches Triplett beschreibt einen Satz, welcher aus einem Subjekt, Prädikat und Objekt besteht (Ceran et al., 2012). Da ArchiMate eine ikonografische Modellierungssprache ist und die einzelnen Elemente zweidimensional angeordnet werden, muss für die Modellierung die Reihenfolge des semantischen Tripletts nicht eingehalten werden (The Open Group Standard, 2022a).

Der aktive Strukturaspekt entspricht nach dem semantischen Triplett die Subjekte. Innerhalb von ArchiMate sind dies strukturelle Elemente, welche ein Verhalten auf andere Elemente ausüben können (The Open Group Standard, 2022a). Dies wären beispielsweise die in den vorherigen Absätzen erwähnten Elemente „Geräte“ oder „Applikationskomponenten“. Welches Verhalten von den Subjekten ausgeübt wird, wird durch die Elemente des Verhaltensaspektes definiert. Schematisch und visuell sind dies eigene Konzepte, welche dem Prädikat des semantischen Tripletts zuzuordnen sind. Wierda (2021) fügt hierbei an, dass jene Trennung zwischen Struktur und Verhalten ein relevantes Modellierungskriterium von ArchiMate sei.

Der dritte und letzte Aspekt ist der passive Strukturaspekt. Elemente, welche nach diesem Aspekt klassifiziert werden, entsprechen den Objekten des semantischen Tripletts. Auf diese passiven Strukturelemente wird das Verhalten der aktiven Strukturelemente ausgeübt. Dies könnten etwa Informationsobjekte in der Geschäftsebene sein, welche sich im Zuge eines Geschäftsprozesses verändern (The Open Group Standard, 2022a).

2.2.4 Das ArchiMate Full Framework als Erweiterung

Abhängig von den darzustellenden Sichtweisen sowie dem gewünschten Detaillierungsgrad wird entweder das ArchiMate „Core“ oder das „Full Framework“ verwendet. Das „Core Framework“ wurde im vorherigen Unterkapitel beschrieben und repräsentiert die Architektur eines Unternehmens (The Open Group Standard, 2022a). Seit der ArchiMate Version 3, wurde der Standard mit der Möglichkeit erweitert, die generelle Strategie eines Unternehmens zu modellieren. Bereits davor kamen in Version 2 die Implementierungs- und Migrationselemente hinzu. Die Strategie sowie die Implementierungs- und Migrationsperspektive wurden als weitere Ebenen dem Core Framework hinzugefügt, woraus das „Full Framework“ resultiert (WIERDA, 2021). In Abbildung 10 wird das „Full Framework“ dargestellt, wobei dieses mit zwei zusätzlichen Ebenen erweitert wurde, sowie die Motivationsperspektive als neuer Aspekt.

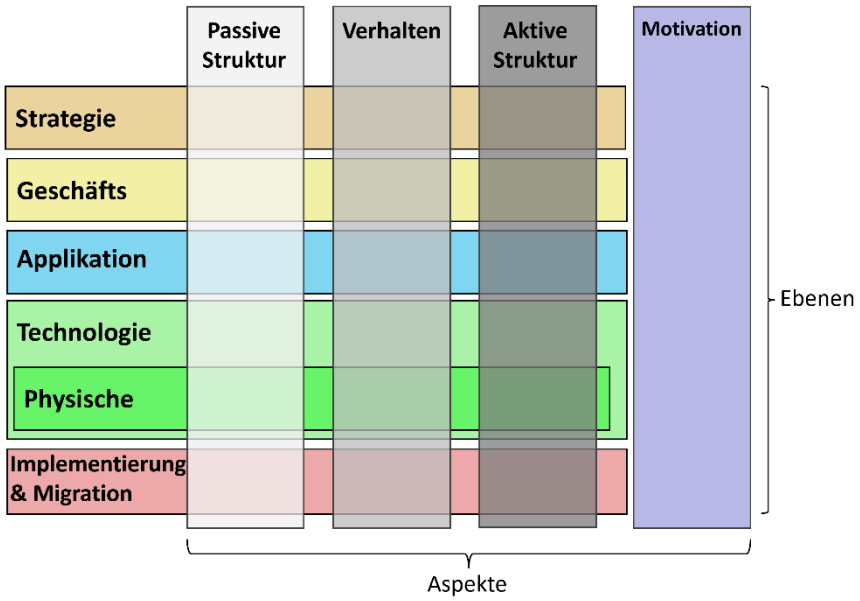

Abbildung 10: ArchiMate Full Framework, in Anlehnung an (The Open Group Standard, 2022a, S. 10)

Die Implementierungs- und Migrationsebene ist die global gesehen unterste Ebene im gesamten Standard. Mit den Elementen dieser Ebene soll das Thema des Programm- und Portfoliomanagements modelliert werden. Daher wäre ein Element aus dieser Ebene beispielsweise ein „Arbeitspaket“. Es identifiziert und plant eine Reihe von Aktionen, die durchgeführt werden sollen, um ein definiertes Ziel zu erreichen. Das Ergebnis aus dem „Arbeitspaket“ kann als Element des „Lieferobjektes“ modelliert werden (The Open Group Standard, 2022a). Zusätzlich soll in dieser Ebene das allgemeine Thema des organisatorischen „Change-Management“ inkludiert werden. Hierbei eignen sich die Elemente „Plateau“ und „Abweichung“. Ein „Plateau“ bezeichnet einen zeitlich begrenzten, jedoch stabilen Zustand einer Unternehmensarchitektur. Abstrakt könnte dies auch als Phase bezeichnet werden, etwa wäre ein „Plateau“ die Startup-Phase. Mittels der „Abweichung“ wird die Differenz zwischen zwei „Plateaus“ modelliert. In der Modellierung repräsentiert dies primär das Ergebnis einer Gap-Analyse, um die Veränderung zu visualisieren (WIERDA, 2021). Ein Modell mit den Elementen aus der Implementierungs- und Migrationsebene ist beispielhaft in Abbildung 11 visualisiert.

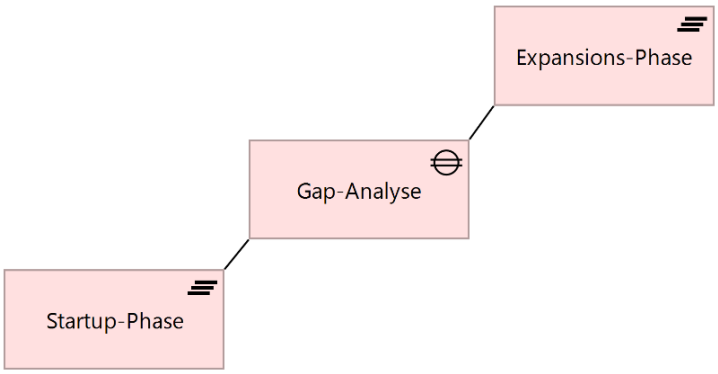

Abbildung 11: Implementierungs- und Migrationsebene, eigene Darstellung

Eine im Verhältnis gesehen neue Ebene ist die Strategieebene. Sie wurde eingeführt, um eine holistische Sichtweise über die strategische Ausrichtung eines Unternehmens zu erlangen. Mit der Strategieebene wird, die im vorherigen Unterkapitel beschriebene Geschäftsebene erweitert, indem die übergeordnete Strategie abstrakt modelliert wird. Nach Wierda (2021) erfolgt dies primär auf eine narrative Art und Weise. Des Weiteren folgt die Strategieebene nicht vollständig den durch die Aspekte etablierten semantischen Tripletts (WIERDA, 2021). In der Ebene sind Elemente enthalten, wie eine „Ressource“ und „Fähigkeit“. Mittels „Ressourcen“ werden Vermögenswerte der Organisation modelliert, welche materiell oder immateriell sein können. Auf Basis der „Ressourcen“ können „Fähigkeiten“ ausgeübt werden, welche die Kompetenzen des Unternehmens darstellen. Für die strategische Ausrichtung wird das Element „Handlungsweise“ eingesetzt. Ein beispielhaftes Modell für den Einsatz der Strategieebene ist in Abbildung 12 visualisiert.

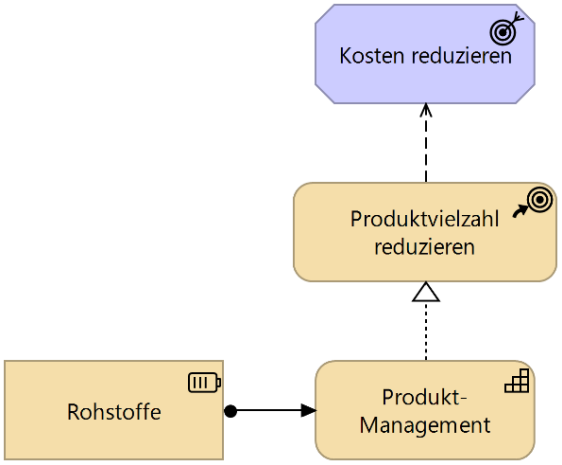

Abbildung 12: Strategie- und Motivationsebene, eigene Darstellung

In Abbildung 12 wurde bereits ein Element aus dem Bereich des Motivationsaspekts inkludiert. Die Elemente der Motivation beschäftigen sich mit der allgemeinen Frage des „Warum“. Diese Elemente können sich über alle Ebenen strecken und sind in der Modellierung nicht an einzelne Ebenen oder Konzepte gebunden. Analog zu den Elementen der Strategieebene sind sie primär narrativ und dienen als abstrakte Konzepte (WIERDA, 2021). Ein solches Konzept ist beispielsweise das visualisierte „Resultat“. Weil etwa die Produktvielzahl durch das Produkt-Management reduziert wurde, konnten auch die Kosten als Resultat daraus reduziert werden.

2.3 BPMN

Business Process Model and Notation oder kurz BPMN ist ein von der Object Management Group entwickelter Standard, um Geschäftsprozesse zu modellieren. Mittels BPMN sollte eine herstellerunabhängige, standardisierte Modellierungssprache für die Darstellung von Geschäftsprozessen entwickelt werden, welche für alle Stakeholder verständlich ist. Im Jahr 2006 wurde folglich die erste Iteration von BPMN in der Version 1.0 von der OMG veröffentlicht. Mit dieser Version wurde ein verständlicher Werkzeugkasten angeboten, welcher die elementaren Konzepte für die Prozessmodellierung anbot. Aufgrund dieser Simplizität konnte sich BPMN als eine der führenden Methoden für die Modellierung von Geschäftsprozessen etablieren (Silver, 2011). Der Aspekt der Geschäftsprozesse wird im nachfolgenden Unterkapitel angesprochen, in welchem die allgemeinen Konzepte hinter der Geschäftsprozessmodellierung beschrieben werden.

Über die Jahre hinweg wurde BPMN stetig weiterentwickelt und die Version 2.0 veröffentlicht, welche kompliziertere Elemente für fortgeschrittene Modellierungsaufgaben implementierte. Zusätzlich wurde ein dahinterliegendes Metamodell auf Basis von UML ausgearbeitet, welches die Syntax sowie Semantik der enthaltenen Elemente definiert (Silver, 2011). Auf die Elemente des 2.0 Standards wird im Unterkapitel 2.3.2 überblicksmäßig eingegangen, da über 150 verschiedene Symbole im Standard enthalten sind, was den Rahmen der Arbeit überziehen würde. Eine zusätzliche Erweiterung der Version 2.0 ist die Möglichkeit, BPMN-Prozesse über eigene Softwareapplikationen auszuführen. Dies bietet den Mehrwert der Automation von Workflows und folglich der Standardisierung von Prozessen (Göpfert & Lindenbach, 2013). Wie die Ausführung, beziehungsweise die Instanziierung von BPMN-Prozessen technisch erfolgt, wird im folgenden Unterkapitel 2.3.3 durchleuchtet.

2.3.1 Einsatzbereiche von BPMN zur Geschäftsprozessmodellierung

BPMN hat sich als de facto Standard für die Modellierung und grafische Visualisierung von Geschäftsprozessen sowie deren Automatisierung etabliert. Ein Grund dafür ist, dass BPMN eine Struktur anbietet, mit welcher Geschäftsprozesse einheitlich modelliert und folglich auch verstanden werden können. Daraus resultiert, dass zwei unterschiedliche Sphären, nämlich die Fachbereiche und die IT-Welt, auf eine gemeinsame Basis zugreifen können, um Geschäftsprozesse zu entwickeln und diese in weiterer Form auch ausführbar werden. Einerseits bringt die visuelle Notation der Prozesse einen tieferen Einblick oder Verständnis für die Intention und den Nutzen der Geschäftsprozesse. Auf der anderen Seite können diese Prozesse ausgeführt, beziehungsweise die Prozessmodelle instanziiert werden, was wiederum ein Antrieb für die Automatisierung und Standardisierung ist (Göpfert & Lindenbach, 2013).

Das allgemeine Ziel von BPMN ist es folglich, einen strukturierten Werkzeugkasten für die Entwicklung, Optimierung und Implementierung von Geschäftsprozessen zu bieten (Göpfert & Lindenbach, 2013). Dieses Ziel wird allgemein anerkannt und findet sich auch explizit im offiziellen Standard (2013) wieder. Analog zum Kapitel 2.2.1, im Bereich der

Unternehmensarchitekturen, fehlt jedoch auch für den Anwendungsbereich von BPMN eine allgemeingültige Definition des dahinterliegenden Konzeptes der Geschäftsprozesse. Die Geschäftsprozesse selbst, beziehungsweise die Geschäftsprozessmodellierung, sind Teilbereiche der übergeordneten Disziplin des Geschäftsprozessmanagements (Weske, 2019). Die Grundlage des Geschäftsprozessmanagements besteht darin, die in einem Unternehmen enthaltenen Prozesse zu verbessern und mit den Bedürfnissen von interessierten Parteien abzugleichen (Singer, 2019). Eine fundierte Erarbeitung des Begriffs des „Geschäftsprozessmanagements“ würde jedoch den definierten Rahmen dieser Arbeit überschreiten.

Weske (2019) führt diverse Definitionen und Erweiterungen des Begriffes der Geschäftsprozesse an, wobei ein allgemeiner Konsens über diesen gebildet werden kann, nämlich, dass sie eine Abhandlung von aneinandergereihten Aktivitäten darstellen. Die Aktivitäten selbst stehen in einem zeitlichen und logischen Bezug zueinander und können manuell, als auch automatisiert ausgeführt werden (Singer, 2019). Zusätzlich zu der Eigenschaft der Verbindung von Aktivitäten werden in anderen Definitionen, wie beispielsweise nach Hammer und Champy (2003), der primäre Fokus auf den Input und Output für einen Prozess gelegt, wobei der Output einen Mehrwert generieren muss. Hierbei ist anzumerken, dass ein Geschäftsprozess implizit effektiv und möglichst effizient sein sollte, da ein Geschäftsprozess ohne Mehrwert lediglich eine Ressourcen- und Zeitverschwendung ist (Weske, 2019).

In Bezug auf die Definition für den Term des „Geschäftsprozesses“ wird im Glossar des BPMN Standards (2013) dieser wie folgt angeführt, nämlich als „*defined set of buiness activities that represent the steps required to achieve a business objective*“ (Object Management Group, Inc, 2013, S. 499). In dieser Definition spiegeln sich die zusammenarbeitenden Aktivitäten wider, welche den Weg zum Erreichen eines Businessziels skizzieren. Um folglich diesen Weg abzubilden, werden die enthaltenen Konzepte und Elemente von BPMN eingesetzt, um einen visualisierten und teils auch automatisierbaren Prozess zu zeichnen.

2.3.2 Grundlagen der BPMN-Notation

Der BPMN 2.0 Standard (Object Management Group, Inc, 2013) umfasst rund 530 Seiten und beschreibt über 150 einzelne Symbole. Eine detaillierte Ausarbeitung aller spezifizierten Konzepte würde den Rahmen dieser Arbeit sprengen, weshalb lediglich die übergeordneten Kategorien der in BPMN enthaltenen Objekte umrissen werden. Zusätzlich wurde von Arevalo et al. (2016) eine zusammenfassende Studie durchgeführt, in welcher herausgefunden wurde, dass lediglich ein Bruchteil der möglichen BPMN-Elemente in der Praxis eingesetzt wird. Der Grund dafür ist die bereits genannte Vielzahl an enthaltenen Elementen sowie die monetären Kosten des Trainings zur korrekten Anwendung in Modellen (Arevalo et al., 2016). Folglich wurde von Zur Muehlen & Recker (2013) durch eine Reihe an Meetings drei Konformitätsklassen entwickelt, welche nachfolgend im BPMN 2.0 Standard (Object Management Group, Inc, 2013) offiziell verankert wurden. Diese Konformitätsklassen werden weiter spezifiziert in die beschreibende, analytische und allgemein ausführbare Subklasse.

Die beschreibende Subklasse ähnelt in ihrer Implementierung den Flussdiagrammen. Sie inkludiert lediglich eine Teilmenge der möglichen Elemente und kann Geschäftsprozesse abstrakt und simplifiziert darstellen. Dadurch sollten die mittels dieser Subklasse modellierten Prozesse selbst ohne tiefgreifende BPMN-Kenntnisse verständlich sein (Arevalo et al., 2016). Als Erweiterung dazu dient die analytische Subklasse, welche auch als Level 2 Palette bezeichnet wird (Silver, 2011). Sie fügt Konzepte wie Nachrichten oder Zwischenereignisse hinzu (Arevalo et al., 2016). Die letzte Subklasse definiert, welche BPMN-Elemente ausführbar sind und in einem ausführbaren Prozess enthalten sein können (Silver, 2011). Dem Aspekt der Ausführbarkeit, beziehungsweise der Instanziierung von BPMN-Prozessen wird das nachfolgende Unterkapitel 2.3.3 gewidmet.

Unabhängig von den Konformitätsklassen werden die in BPMN enthaltenen Objekte in fünf primäre Gruppen oder Kategorien eingeteilt (Göpfert & Lindenbach, 2013), welche wie folgt in Abbildung 13 gelistet sind. Diese Auflistung zeigt die allgemeine Notation von BPMN, wobei sich deren visuelle Repräsentation unterscheiden kann. In den nachfolgenden Absätzen werden jene Oberbegriffe erläutert und mit einem praktischen Beispiel der analytischen Konformitätsklasse in Abbildung 14 beschrieben.

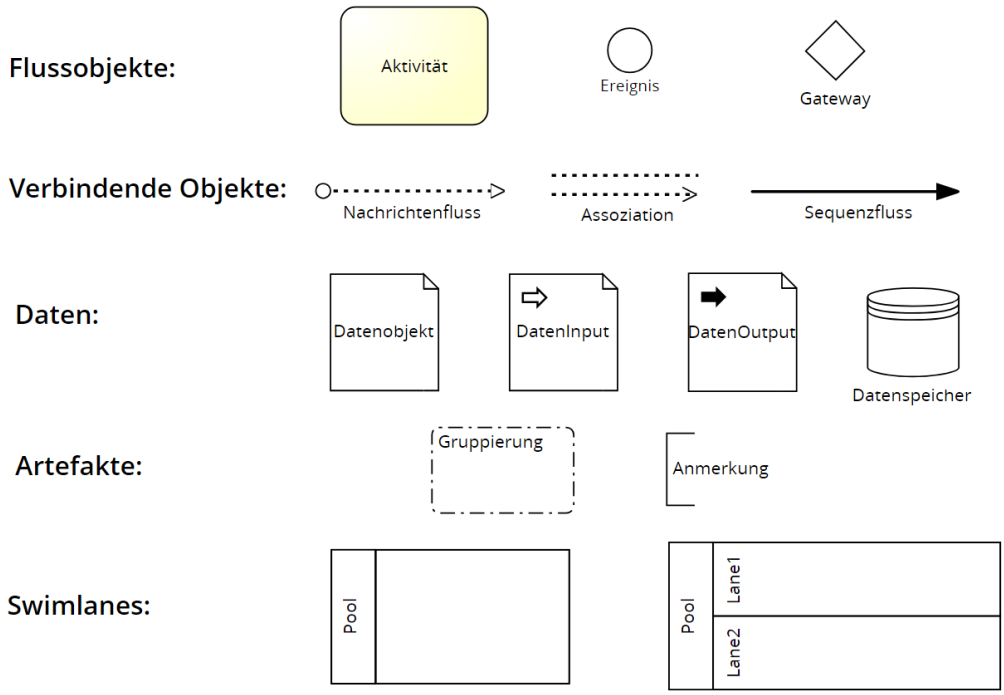

Abbildung 13: Kategorien der BPMN-Elemente, in Anlehnung an (Göpfert & Lindenbach, 2013, S. 5)

Beginnend mit den Flussobjekten nach Abbildung 13, definieren sie die Knoten in einem Prozessdiagramm. Darin enthalten sind Ereignisse, Aktivitäten und Gateways. Ein Ereignis wird visuell durch einen Kreis notiert und beschreibt, beispielsweise, dass ein Prozess gestartet oder beendet wird oder ein bestimmtes Vorkommnis, wie das Eintreffen einer Nachricht, erfolgt. Aktivitäten sind grundlegende Elemente in Geschäftsprozessen, welche Aufgaben, Subprozesse oder Transaktionen darstellen können, die innerhalb des Prozesses ausgeführt werden. Sie visualisieren den Ablauf von Handlungen und Entscheidungen und definieren abstrakt gesprochen, „was“ der Prozess tut. Sie werden als abgerundete Rechtecke dargestellt und sollten

in der Form „Objekt + Verb“ beschriftet werden. Das letzte Element der Flussobjekte ist das Gateway. Ein Gateway wird eingesetzt, um eine Kontrolle und folglich Steuerung des Prozesses zu modellieren. Durch Gateways werden alternative Pfade beim Durchlaufen des Prozesses möglich. Sie sind als Raute inklusive eines zusätzlichen Symbols innerhalb dieser notiert (Göpfert & Lindenbach, 2013).

Um die Reihenfolge des Prozessablaufes zu modellieren und um BPMN-Elemente miteinander in Beziehung zu bringen, werden verbindende Objekte eingesetzt. Mit dem Sequenzfluss wird die zeitliche und logische Reihenfolge der Flussobjekte etabliert. Daher ist beispielsweise ein Startereignis der Vorgänger einer Aktivität, welche als Nachfolger bereits ein Endereignis besitzen könnte. Hierbei ist anzumerken, dass ein Sequenzfluss lediglich innerhalb eines Pools erlaubt ist. Ein Pool repräsentiert die Verantwortlichkeiten der darin enthaltenen Aktivitäten. Mittels eines Pools können die eigenständigen Geschäftseinheiten abgebildet und zusätzlich anhand von Schwimmbahnen unterteilt werden. Schwimmbahnen sind untergeordnete Geschäftseinheiten, welche am Prozess beteiligt sind (Göpfert & Lindenbach, 2013).

Ein Unternehmen könnte beispielsweise als gesamter Pool dargestellt werden, welcher sich in die Teilbereiche „Produktion“, „Vertrieb“ und „Marketing“ als Schwimmbahnen aufteilt. Um die Kommunikation oder den Austausch zwischen Pools zu ermöglichen, wird das verbindende Objekt des Nachrichtenflusses aus Abbildung 13 eingesetzt. Hierbei werden etwa Nachrichten zwischen dem Unternehmen A und dem Unternehmen B ausgetauscht.

Mit den Flussobjekten, verbindenden Objekten und den Pools sowie Schwimmbahnen werden die Fragen beantwortet, „wer“, „was“, in welchem Ablauf im Prozess ausführt, und welche Entscheidungen und Ereignisse getroffen werden und auftreten können. Die letzten beiden Kategorien nach Abbildung 13 definieren nicht direkt den Prozessablauf selbst, sondern fügen zusätzliche Informationen hinzu, welche rein informativ sein können oder für die Ausführung des Prozesses relevant sind (Freund & Rücker, 2019). Dazu dienen die Artefakte lediglich als informative Unterstützung, um mit der Hilfe von entsprechenden Labels die subjektive Interpretation der modellierten Konzepte zu reduzieren (Henderson-Sellers, 2012). Mittels der Datenobjekte können die Ein- und Ausgaben für das Ausführen eines Prozesses abgebildet und gespeichert werden. Dadurch ist es möglich, denselben Prozess mit unterschiedlichen Eingaben oder Inputparametern zu starten, was zu alternativen Wegen im Prozess führen kann (Von Rosing et al., 2015).

Um die in den vorherigen Absätzen beschriebenen Grundlagen der BPMN-Notation zu veranschaulichen, werden diese Basis-BPMN-Elemente in einem konkreten Beispiel modelliert. Dazu dient der Prozess einer Pizza-Bestellung welcher direkt von der OMG entwickelt wurde und auch im BPMN-Standard angegeben wird (Object Management Group, Inc, 2013). Dieser Prozess wurde gewählt, da die zuvor beschriebenen Basiselemente enthalten sind und eine einheitliche Modellierungsetikette eingehalten wird. Die folgende Abbildung 14 visualisiert den Pizza-Bestellprozess.

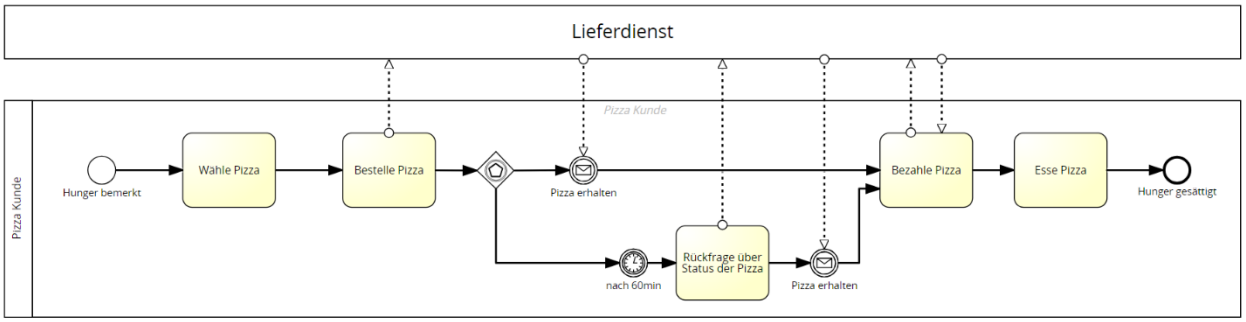

Abbildung 14: Beispielhafter BPMN-Prozess, in Anlehnung an (Freund & Rücker, 2019, S. 84)

Im BPMN-Standard (2013) wird keine Richtung in Bezug auf die Anordnung der einzelnen Elemente vorgeschrieben. Der Prozess in Abbildung 14 implementiert einen horizontalen Verlauf von links nach rechts beginnend. Freund & Rücker (2019) beschreiben, dass jede Anordnung technisch akzeptabel ist, solange diese einheitlich verwendet wird. Zusätzlich folgen die modellierten Aktivitäten dem von Göpfert & Lindenbach (2013) angesprochenen „Verb + Objekt“ Muster. Im Bestellprozess werden zwei Pools abgebildet, zum einen die bestellende Person und ein Lieferdienst. Hierbei ist eine weitere Designentscheidung ersichtlich, nämlich, dass Pools auf- oder zugeklappt sein können. Aus der Sicht der bestellenden Person sind die Aktivitäten und Entscheidungen des Lieferdienstes irrelevant, lediglich die ein- und ausgehenden Nachrichten sind von Bedeutung. Daher wird der Lieferdienst als zusammengeklappter Pool, oder auch „Blackbox“ genannt, modelliert (Freund & Rücker, 2019).

Ein Prozess benötigt stets einen Start und zumindest ein Ende (Singer, 2019). In Abbildung 14 wird dies mit jeweils einem Start- und einem Endereignis modelliert. Nach dem Start des Prozesses erfolgt ein Sequenzfluss, bis eine Pizza bestellt wurde. Folglich teilt sich der Prozess, indem eine Nachricht, in diesem Fall die Bestellung, an den Lieferdienst gesendet wird. Im Pool der bestellenden Person wird ein Gateway erreicht. Dieses Gateway ist das sogenannte „Ereignisbasierende Gateway“ und es stellt eine spezielle Ausprägung des allgemeinen Gateways dar. Das „Ereignisbasierende Gateway“ modelliert, dass unterschiedliche Ereignisse eintreten könnten, jedoch der Sequenzfluss bei dem Ereignis fortgesetzt wird, welches als Erstes eintritt (Von Rosing et al., 2015).

Beim Bestellprozess der Pizza nach Abbildung 14 kann entweder die Pizza erhalten oder nach einer Stunde Wartezeit der Lieferdienst kontaktiert werden. Sollte die Pizza innerhalb einer Stunde ankommen, so wird diese bezahlt, gegessen und der Prozess beendet. Alternativ erfolgt die Rückfrage an den Lieferdienst über den Status der Pizzabestellung. Die Nachricht, dass die Pizza erhalten wurde, sowie die eine Stunde Wartezeit sind spezielle Ausprägungen des allgemeinen Ereignis-Konzepts. Da sie weder den Start noch das Ende eines Prozesses markieren, werden sie als Zwischenereignisse bezeichnet (Göpfert & Lindenbach, 2013). Konkret wird ein „eingehendes Nachrichten-Zwischenereignis“ und ein „Timer-Zwischenereignis“ abgebildet.

2.3.3 Instanziierbarkeit von BPMN-Prozessen

Mittels BPMN können Prozesse nicht nur visualisiert, sondern auch durch spezielle Programme, sogenannte Engines, ausgeführt werden. Auf diese Weise bleiben Prozessmodelle nicht bei einer rein statischen Darstellung, sondern können beispielsweise softwaregesteuert Daten verarbeiten, Automatismen durchführen oder Formulare als Eingabemasken anbieten. Dies gelingt dadurch, dass jedes BPMN-Diagramm in einer XML-Struktur serialisiert wird und zusätzliche Attribute für dessen Ausführung besitzt (Silver, 2011). Hierbei ist jedoch anzumerken, dass primär Elemente aus der allgemein ausführbaren Subklasse mittels einer Engine ausführbar sind (Object Management Group, Inc, 2013). Ein modellierter und zur Ausführung syntaktisch valider Prozess kann folglich über die Engine als einzelne Prozess-Instanz gestartet werden.

Die Prozess-Instanz selbst ist demnach die konkrete, mit Daten gefüllte Abbildung des Prozesses während der Laufzeit (Freund & Rücker, 2019). Um den Ablauf dieser zur Laufzeit entstandenen Prozess-Instanzen technisch abbilden zu können, wird auf das sogenannte Token-Konzept gesetzt. Das Token-Konzept selbst ist ein theoretisches Konzept, bei welchem einzelne Tokens generiert werden und diese sich zwischen den Elementen des Prozessmodells bewegen. Mittels der Token kann das Verhalten eines Prozesses veranschaulicht werden, indem visualisiert wird, wie, beziehungsweise welchen Weg ein oder mehrere Token im modellierten Prozess durchlaufen (Object Management Group, Inc, 2013).

Bei der Instanziierung einer Prozess-Instanz aus einem modellierten Prozess werden initial so viele Token wie Endereignisse erzeugt. Dies bedingt syntaktisch, dass mindestens ein Token generiert wird, da ein valider BPMN-Prozess zumindest ein Endereignis vorweisen muss (Singer, 2019). Das Token selbst beschreibt einen Punkt in der Ausführung der Prozess-Instanz und ist stets einem Element zugeordnet. Daher können sich Tokens auf Aktivitäten, Ereignissen, Entscheidungen sowie Sequenzflüssen befinden. Dasjenige Element, auf welchem sich ein Token befindet, wird von der dahinterliegenden Workflow-Engine ausgeführt. Um das Verhalten des Prozesses folglich abbilden zu können, müssen diese Tokens zwischen den einzelnen Elementen verschoben werden. Dies gelingt mittels der Sequenzflüsse. Ein Sequenzfluss stellt die Verbindung zwischen den Elementen, wie etwa zwei Aktivitäten, dar, um das Token zu transportieren (Object Management Group, Inc, 2013).

Neben der Generierung beim Start des Prozesses und dem Transport während der Ausführung können Tokens auch gelöscht, genauer gesagt konsumiert werden. Dies erfolgt beispielsweise durch Endereignisse (Object Management Group, Inc, 2013). Endereignisse besitzen keine ausgehenden Sequenzflüsse und müssen im Prozessmodell angeführt werden, da die Instanz ansonsten nie abgeschlossen werden könnte. Eine Prozess-Instanz gilt folglich als beendet, wenn alle zum Startzeitpunkt vorhandenen oder während der Ausführung generierten Tokens konsumiert wurden (Göpfert & Lindenbach, 2013).

Um während der Ausführung neue Tokens zu erzeugen, sind diverse Möglichkeiten gegeben. Hierbei könnten „Parallele Gateways“ eingesetzt werden, welche einen eingehenden Sequenzfluss in zwei parallele ausgehende Sequenzflüsse aufteilen, oder durch spezielle mehrfach ausführende Aktivitäten (Object Management Group, Inc, 2013). Es gilt jedoch, dass

bereits während der Modellierung von Elementen, welche im Prozess neue Tokens generieren, darauf geachtet wird, dass diese folglich auch konsumiert werden. Ansonsten könnten bei der Ausführung des Prozessmodells nicht abschließbare Prozess-Instanzen entstehen.

Um den Ablauf eines instanziierten Prozesses inklusive des enthaltenen Token-Konzeptes zu demonstrieren, dient die folgende Abbildung 15. In dieser wird ein Terminfindungsprozess abgebildet und der Weg durch den Prozess mittels Momentaufnahmen visualisiert. In diesem Beispiel wird ein Token durch das Startereignis generiert und schließlich im Endereignis konsumiert. Die Momentaufnahmen 1 bis 15 zeigen den Transfer des Tokens durch den Prozess.

In der ersten Momentaufnahme nach Abbildung 15 wird das Token generiert und über den Sequenzfluss an die nachfolgende Aktivität übermittelt. Mittels weiterer Sequenzflüsse gelangt das Token bis zur Momentaufnahme 7, wo diese sich auf einem „Exklusiven Gateway“ befindet. Je nach den Input-Daten für die Prozess-Instanz sowie den zuvor durchgeführten Aktivitäten wird die Bedingung mit „Ja“ oder „Nein“ erfüllt. Daher erfolgt eine Abzweigung in der Prozess-Instanz entweder nach 8a oder 8b, jedoch nicht beide. Folglich bleibt das eine Token erhalten und es wird kein neues generiert. Zum Ende des Prozesses gelangt das Token in Momentaufnahme 15 in das Endereignis. Dieses konsumiert das initial generierte Token und die Prozess-Instanz ist beendet (Göpfert & Lindenbach, 2013). Obwohl aus dem Endereignis eine Nachricht gesendet wird, generiert dieses kein Token, da einem Nachrichtenfluss eine Nachricht und kein Token folgt (Object Management Group, Inc, 2013).

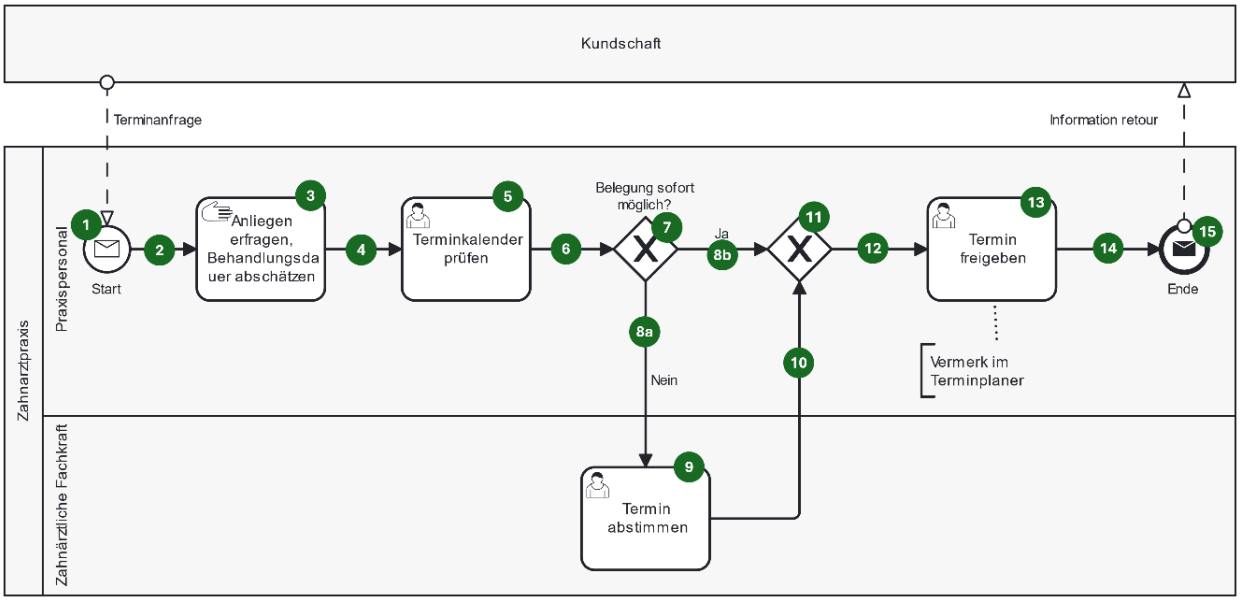

Abbildung 15: Ablauf des Token-Konzepts, in Anlehnung an (Göpfert & Lindenbach, 2013, S. 11)

Durch die in Abbildung 15 dargestellten Momentaufnahmen wird gezeigt, wie mittels Tokens das Verhalten eines Prozesses verständlich visualisierbar ist. Allein in der Abbildung 15 kann der modellierte Prozess zwei unterschiedliche Abzweigungen während der Ausführung annehmen. Abstrakter formuliert bedeutet dies, dass aus einem Prozess-Diagramm, je nach Komplexität, n instanziierte Prozesse impliziert werden (Freund & Rücker, 2019). Mittels einer entsprechenden Engine, welche die Tokens während der Ausführung visualisieren kann, ist das Verhalten des Prozesses, genauer gesagt, der jeweiligen Prozess-Instanz nachvollziehbar.

3 ÜBERSCHNEIDUNGEN ZWISCHEN ARCHIMATE UND BPMN

Ein Ziel dieser Arbeit ist es aufzuzeigen, wie die beiden Modellierungssprachen ArchiMate und BPMN miteinander kombiniert werden können. Im vorangegangenen Kapitel wurden bereits die jeweiligen Einsatzbereiche sowie die konzeptionellen Grundlagen beider Sprachen erläutert. In diesem Kapitel soll folglich aufgezeigt werden, welche Überschneidungen, beziehungsweise Interaktionen zwischen den beiden Modellierungssprachen bereits in der Literatur vorhanden sind. Dazu wurden diverse Werke identifiziert, welche nachfolgend einzeln betrachtet werden. Mittels dieser Literaturrecherche soll ein gemeinsamer Nenner zwischen den Überschneidungen von ArchiMate und BPMN gefunden werden, um darauf aufbauend ein, zumindest für diese Arbeit domänenspezifisches, Mapping zu definieren. Dazu ist anzumerken, dass in dieser Arbeit explizit auf die Instanziierung von BPMN-Prozessen aus ArchiMate-Modellen fokussiert wird, wodurch gewisse Restriktionen durch die Instanziierbarkeit der einzelnen Konzepte selbst resultieren.

Der offizielle ArchiMate Standard (2022a) als auch Lankhorst et al. (2017) deuten an, dass ein Mapping zwischen ArchiMate und BPMN einfach sei, was jedoch von Wierda (2021) widerlegt wird. Wierda (2021) beschreibt, dass ein allgemeingültiges und eindeutiges Mapping zwischen den beiden Sprachen nicht möglich sei. Um die grundlegende Problematik in der Schwierigkeit eines solchen Mappings dazustellen, dienen die folgenden Beispiele. Zum einen sind in beiden Modellierungssprachen die Konzepte eines Geschäftsprozesses sowie eines Geschäftsereignis oder eines allgemeinen Ereignisses gegeben. Das alleinige Konzept eines Geschäftsprozesses in ArchiMate kann jedoch einen gesamten Prozess in BPMN darstellen oder lediglich eine im Prozess enthaltene Aktivität. Im Falle der einzelnen Aktivität erfolgt deren logische Abfolge in ArchiMate über Fluss- und auslösenden Beziehungen. Zusätzlich sind für Entscheidungen innerhalb des Prozesses die Beziehungskonnektoren einsetzbar. Sie entsprechen den Gateways, wobei nicht alle in BPMN enthaltenen Gateways abbildbar sind (The Open Group Standard, 2022a).

Ein weiteres Beispiel ist die Korrespondenz einer Geschäftsrolle oder Geschäftsakteurs mit einem Teilnehmer oder Pool aus BPMN. Bereits anhand dieser einen Verbindung wird ersichtlich, dass ein direktes eins zu eins Mapping zwischen den beiden Modellierungssprachen sich als schwierig erweist. Die beiden genannten Konzepte der Geschäftsrolle oder Geschäftsakteurs könnten dem selbigen BPMN-Element zugeordnet werden. Demnach gilt das im Rahme dieser Arbeit entwickelte Mapping als eine, in Bezug auf die implementierten Anwendungsfälle, domänenspezifische Lösung, welche sich von anderen Überschneidungen differenzieren kann. Zusätzlich wird der Aspekt der Instanziierbarkeit explizit betrachtet.

3.1 Übersicht literaturbasierter Mappings und Methoden

Im Zuge der Literaturrecherche für diese Arbeit wurden diverse Quellen identifiziert, welche bereits mögliche Überschneidungen zwischen der ArchiMate Modellierungssprache und der BPMN-Notation anführen. Die hierbei vorhandenen Werke nach Penicina (2013), Qumer Gill und Atif Qureshi (2015), Lankhorst et al. (2017), Marugán Cancio (2018) sowie Wierda (2021) werden inhaltlich prägnant exzerpiert, da diese jeweils unterschiedliche Ansätze für die Eruierung des Mappings zwischen ArchiMate und BPMN präsentieren. Anschließend wird im nachfolgenden Unterkapitel eine visuelle Aufbereitung der gesammelten Werke vorgenommen, um daraus gezielte Schlüsse für die konkrete Implementierung dieser Arbeit ableiten zu können.

Das zeitlich älteste Werk in der Analyse stammt von Penicina (2013) und verfolgt einen ontologischen Ansatz. Dabei wird zu ArchiMate und BPMN ein weiteres Artefakt hinzugefügt, nämlich die Bunge-Wand-Weber, kurz BWW, Ontologie. Die BWW-Ontologie dient als theoretisches Rahmenwerk um formale Informationssysteme zu modellieren und um sicherzustellen, dass alle aus der Realität relevanten Konzepte in diesen Systemen enthalten sind (Wand & Weber, 1990).

Das Ziel der Arbeit nach Penicina (2013) ist nicht nur die Entwicklung eines Mappings zwischen ArchiMate und BPMN, sondern auch die Analyse der Vollständigkeit und Rechtmäßigkeit von Geschäftsprozessmodellen. Der Term „Vollständigkeit“ bedeutet hierbei, dass Modelle alle für ein Informationssystem notwendigen Elemente enthalten und mit „Rechtmäßigkeit“ wird beschrieben, dass die geltenden Regeln innerhalb des Systems eingehalten werden (Penicina, 2013). Anhand dieses theoretischen Rahmenwerks wurde, analog zu einem bereits existierenden Mapping von der BWW-Ontologie nach BPMN, die Verbindung von BWW zu ArchiMate konzeptioniert. Dadurch ist die gesamtheitliche Kombination von ArchiMate, BPMN und der BWW-Ontologie gegeben, wodurch die direkte Verbindung zwischen den Konzepten von ArchiMate und BPMN auf Metamodell-Ebene realisierbar war. Hierbei gilt jedoch anzumerken, dass das definierte Mapping den veralteten ArchiMate 2.0 Standard nutzt.

In dem Werk von Qumer Gill und Atif Qureshi (2015) wird ein mathematischer Ansatz implementiert, um die Metamodelle der beiden Artefakte miteinander zu vergleichen. Dazu erfolgte der Einsatz der syntaktischen, semantischen und strukturellen Analyse. Es wurden eigene Kennzahlen erhoben, welche prozentuell angeben, wie ähnlich die einzelnen Konzepte zwischen ArchiMate und BPMN sind.

Die syntaktische Analyse vergleicht die Bezeichnungen der Konzepte, wie sie in den jeweiligen Metamodellen von ArchiMate und BPMN festgelegt sind. Der alleinige Vergleich auf Basis von identen oder ähnlichen Wörtern ist jedoch nicht ausreichend für ein vollständiges Mapping, da dies zu Fehlinterpretationen führen kann. Qumer Gill und Atif Qureshi (2015) führen diesbezüglich als Beispiel an, dass die Konzepte „Confirmation“ und „Notification“ anhand der verwendeten Wörter ähnlicher sind als wie „Confirmation“ und „Verification“. Semantisch betrachtet sind die letzten beiden Terme jedoch ähnlicher. Daher wurde die semantische Analyse durchgeführt, in welcher für jedes Konzept Synonyme definiert wurden. Jedes Konzept sowie dessen Synonyme wurden folglich miteinander verglichen und als weitere Kennzahl erhoben.

Der letzte Indikator für die Ähnlichkeit der Konzepte zwischen ArchiMate und BPMN wurde anhand der strukturellen Analyse erhoben (Qumer Gill & Atif Qureshi, 2015). ArchiMate als auch BPMN weisen eine hierarchische Struktur in deren Metamodellen auf. In diesen Strukturen sind jeweils übergeordnete, nachfolgende und auf gleicher Ebene befindende Konzepte enthalten. Demnach konnten diese hierarchischen Ähnlichkeiten der beiden Modellierungssprachen miteinander verglichen werden. Mittels der erhobenen Kennzahlen wurde ein Score errechnet, wie ähnlich die Konzepte von ArchiMate zu BPMN sind. Als finale Überarbeitung führten Qumer Gill und Atif Qureshi (2015) eine ontologische Analyse, ähnlich zu Penicina (2013), durch. In dieser wurde das zuvor eruierte Mapping erneut mit dem Fokus auf die Metamodelle der Konzepte evaluiert. Sie beschränkten sich auf die Geschäftsebene von ArchiMate und beinhalten daher keine Konzepte aus den anderen Ebenen, mit Ausnahme des Datenobjekts (Qumer Gill & Atif Qureshi, 2015).

Lankhorst et al. (2017) führen an, dass ArchiMate als Dachbegriff oder übergeordnete Modellierungssprache für die Integration weiterer Konzepte dient. Folglich werden in diesem Werk zusätzliche Mappings zu anderen Modellierungssprachen und Frameworks angeführt. Darin enthalten sind unter anderem der Business Model Canvas, Balanced Score Card, UML, als auch BPMN. Es wird dargestellt, dass ein Mapping von ArchiMate zu BPMN als unkompliziert und einfach beschrieben wird, wobei zugleich darauf hingewiesen wird, dass ein einzelnes ArchiMate-Konzept in unterschiedliche BPMN-Konzepte überführt werden kann. (M. M. Lankhorst et al., 2017). Für die dargestellte Mapping-Tabelle werden keine zusätzlichen Informationen oder zugrunde liegende Methoden angeführt.

Ein weiteres, auf Basis der Evaluierung von Metamodellen entwickeltes, Werk ist nach Marugán Cancio (2018). Es wurden die semantischen Ähnlichkeiten der Metamodelle zwischen ArchiMate 3.0 und BPMN 2.0 verglichen. Auf diese Ähnlichkeiten aufbauend, wurde ein Mapping etabliert, welches zusätzlich mittels der Atlas Transformation Language, kurz ATL, serialisiert wurde. Die ATL ist eine modellbasierte Transformationssprache, die im Rahmen des Eclipse Modeling Framework entwickelt wurde. Sie ermöglicht die Definition von Transformationsregeln, um Quellmodelle in Zielmodelle zu überführen, und unterstützt dabei sowohl die automatische als auch die manuelle Modelltransformation (Wagelaar, 2024).

Anhand der ATL wurden Transformationsregeln entwickelt, um ArchiMate Konzepte syntaktisch korrekt in BPMN-Elemente zu transformieren. Es wird jedoch angeführt, dass die konzeptionierten Transformationsregeln primär als Pseudocode verschriftlicht wurden und nicht in Eclipse technisch getestet wurden. Das Werk soll als Grundlage für die Entwicklung von Applikationen dienen, welche ArchiMate Modelle automatisiert in BPMN-Diagramme transformieren können (Marugán Cancio, 2018).

Die letzte evaluierte Literatur zur Kombination von ArchiMate und BPMN ist nach Wierda (2021). In diesem Fall wird keine eindeutige Mapping-Tabelle angeführt, sondern in Form von Textpassagen und narrativen als auch modellierten Beispielen. Konträr zu Lankhorst et al. (2017) wird beschrieben, dass ein Mapping von ArchiMate zu BPMN kein einfaches Unterfahren sei, und diverse Möglichkeiten und Ansätze bestehen. Hierbei werden diverse Fragen gestellt, welche sich primär auf Detailfälle in der Modellierung beziehen.

Zusätzlich fokussiert sich Wierda (2021) auf eine technische Implementierung zur Kombination der beiden Standards. Dazu wurde die BPMN-Seite erweitert, indem sogenannte „Other Domain“ Objekte hinzugefügt wurden. Innerhalb dieser spiegeln sich die ArchiMate Konzepte wider und werden mit den BPMN-Elementen verknüpft. Diese Verknüpfung wird in den BPMN-Elementen gespeichert. Dies bedeutet beispielsweise, dass, wenn eine ArchiMate „Applikationsschnittstelle“ zu einer BPMN-Aktivität verknüpft wird, die Schnittstelle selbst von der ausführenden Einheit der BPMN-Aktivität genutzt wird. Alternativ, wenn ein „Applikationsservice“ zu einer BPMN-Aktivität verknüpft wird, bedeutet dies, dass innerhalb der Aktivität der Service genutzt wird (WIERDA, 2021).

Anhand des dargestellten Beispiels wird ersichtlich, dass ein eins zu eins Mapping zwischen ArchiMate und BPMN sich als schwierig erweist und ArchiMate primär beschreibt, was für die Ausführung der BPMN-Aktivität notwendig ist. Innerhalb der BPMN-Modellierungsumgebung werden dann diese „Other Domain“ Objekte den BPMN-Elementen hinzugefügt. Hierbei können zu einem BPMN-Element mehrere ArchiMate Konzepte verbunden werden. Wierda (2021) fügt jedoch die Möglichkeit einer „direkten Äquivalenz“, wie von den zuvor beschriebenen Werken, hinzu. Dies wäre beispielsweise, dass eine BPMN-Lane direkt einer ArchiMate „Geschäftsrolle“ entspricht. Dazu wird in ArchiMate als auch BPMN dasselbe Label als eindeutige Identifikation genutzt.

3.2 Aus der Literatur aggregiertes Mapping

Aus den fünf im vorherigen Unterkapitel exzerpierten Literaturquellen wurde eine zusammenfassende Mapping-Tabelle entwickelt, welche die Zuweisung der ArchiMate-Konzepte auf diverse BPMN-Elemente beinhaltet. Die nachfolgende Tabelle 1 zeigt diese Korrelationen. Hierbei befinden sich in der ersten Spalte alle in der Literatur gefundenen ArchiMate-Konzepte, welche ein BPMN-Äquivalent aufweisen. Mit wenigen Ausnahmen, wie beispielsweise der „Gruppierung“ und dem „Standort“, sind primär ArchiMate Konzepte aus der Geschäftsebene, der Applikationsebene, Beziehungen sowie den Beziehungskonnektoren vorhanden. Die Geschäftsebene als auch die Beziehungskonnektoren waren zu erwarten, da insbesondere für die Beziehungskonnektoren der ArchiMate Standard (2022a) selbst anführt, dass diese an BPMN Gateways angelehnt sind.

Aus der Technologieebene sind die beiden Konzepte „Gerät“ und „Artefakt“ gegeben. Jede weitere Spalte zeigt die Mappings aus der jeweiligen Literatur, beginnend mit Penicina (2013) und endend mit Wierda (2021). Im Fall von Wierda (2021) ist anzumerken, dass keine eindeutige, tabellarische Auflistung des Mappings vorhanden war, sondern diese aus Textpassagen und Abbildungen eruiert wurden.

Tabelle 1: Aus der Literatur aggregierte Mapping-Tabelle, eigene Darstellung

ArchiMate Konzept	**Penicina (2013)**	**Qumer Gill und Atif Qureshi (2015)**	**Lankhorst et al. (2017)**	**Marugán Cancio (2018)**	**Wierda (2021)***
Gruppierung				Gruppierung	
Standort		Pool		Pool	
Geschäftsereignis	Ereignis	Ereignis		Ereignis	Ereignis
Geschäftsservice		Aktivität		Aktivität, Pool, Subprozess	
Geschäftsschnittstelle		Interface			
Geschäftsprozess	Prozess,Diagramm, Pool, Lane	Aktivität	Prozess	Aktivität, Pool, Subprozess	Prozess, Aktivität, Subprozess, Pool
Geschäftsfunktion	Aktiviät, Subprozess			Aktivität, Pool, Subprozess	
Geschäftsinteraktion	Kollaboration Diagramm			Aktivität, Pool, Subprozess	
Geschäftsobjekt	Datenobjekt			Datenobjekt, Nachricht	Datenobjekt
Produkt				Datenobjekt, Nachricht	
Geschäftsakteur		Teilnehmer	Teilnehmer/Pool, Lane		
Geschäftsrolle	Lane	Partnerrolle	Teilnehmer/Pool, Lane		Lane
Geschäftskollaboration		Kollaboration	Kollaboration		
Applikationskomponente			Teilnehmer/Pool,Lane		Lane, Service Aktivität
Applikationskollaboration			Kollaboration		
Applikationsprozess			Prozess		
Applikationsfunktion	Service Aktivität, Skript Aktivität				
Applikationsservice					Aktivität, Prozess
Datenobjekt	Datenobjekt	Datenobjekt			Datenobjekt
Gerät	Datenspeicher				
Artefakt	Datenobjekte	Artefakt			Datenspeicher
Auslösen-Beziehung			Sequenzfluss	Sequenzfluss	
Zugriffsbeziehung			Daten-Assoziation	Daten-Assoziation, Nachrichtenfluss	
Flussbeziehung				Sequenzfluss	
Assoziationsbeziehung				Daten-Assoziation	
And Verbindungspunkt			Inklusives und -Parallels Gateway	Paralleles Gateway	Paralleles Gateway
Or Verbindungspunkt			Exklusives und eventbasierendes Gateway	Exklusives Gateway	Exklusives Gateway

Anhand der Tabelle 1 ist zu erkennen, dass sich die einzelnen Korrespondenzen zwischen den ArchiMate- und BPMN-Konzepten von Literatur zu Literatur unterscheiden. Zusätzlich kann ein ArchiMate-Konzept in einer Literatur synonym für mehrere BPMN-Elemente gesehen werden. Dies ist insbesondere für das ArchiMate Konzept „Geschäftsprozess“ der Fall. Hierbei führen Penicina (2013), Marugán Cancio (2018) sowie Wierda (2021) drei bis vier verschiedene BPMN-Elemente für ein ArchiMate-Konzept an. Dies spiegelt das bereits in der Einleitung zu diesem Kapitel angeführte Problem wider, dass ein allgemeines eins zu eins Mapping schwierig und nicht für jeden Anwendungsfall sinnvoll ist. Abgesehen von dem „Geschäftsprozess“ ist kein weiteres ArchiMate-Konzept in allen Literaturquellen vorhanden.

Das Spaltenlayout von Tabelle 1 orientiert sich nach den einzelnen Literaturquellen. Um diese Ergebnisse literaturunabhängig, strukturierter und quantitativ darzustellen, wurde daraus eine Matrix in Form einer Heatmap abgeleitet. Diese ist folglich in Tabelle 2 visualisiert und zeigt auf der Y-Achse die ArchiMate-Konzepte und auf der X-Achse die BPMN-Elemente.

Tabelle 2: Heatmap der aggregierten Mapping-Tabelle, eigene Darstellung

	Gruppierung	Pool	Ereignis	Aktivität	Subprozess	Interface	Prozess Diagramm	Lane	Prozess	Kollaboration Diagramm	Datenobjekt	Nachricht	Teilnehmer	Partnerrolle	Kollaboration	Service Aktivität	Skript Aktivität	Datenspeicher	Datenobjekte	Artefakt	Sequenzfluss	Daten-Assoziation	Nachrichtenfluss	Inklusives Gateway	Paralleles Gateway	Eventbasierendes Gateway	Exklusives Gateway	Anzahl BPMN Konzepte
Gruppierung	1																											1
Standort		2																										1
Geschäftsereignis			4																									1
Geschäftsservice		1		2	1																							3
Geschäftsschnittstelle						1																						1
Geschäftsprozess		3		3	1		1	1	2																			6
Geschäftsfunktion		1		2	2																							3
Geschäftsinteraktion		1		1	1					1																		4
Geschäftsobjekt											3	1																2
Produkt											1	1																2
Geschäftsakteur		1						1					2															3
Geschäftsrolle		1						3					1	1														4
Geschäftskollaboration															2													1
Applikationskomponente		1						2					1			1												4
Applikationskollaboration															1													1
Applikationsprozess									1																			1
Applikationsfunktion																1	1											2
Applikationsservice				1					1																			2
Datenobjekt											3																	1
Gerät																		1										1
Artefakt																		1	1	1								3
Auslösen-Beziehung																					2							1
Zugriffsbeziehung																						2	1					2
Flussbeziehung																					1							1
Assoziationsbeziehung																						1						1
And Verbindungspunkt																								1	3			2
Or Verbindungspunkt																										1	3	2
Anzahl ArchiMate Konzepte	1	8	1	5	4	1	1	4	3	1	3	2	3	1	2	2	1	2	1	1	2	2	1	1	1	1	1	

Die Zahlen innerhalb von Tabelle 2 ergeben sich aus der aggregierten Anzahl der Mappings für ein bestimmtes Konzept. Ein Beispiel dafür wäre das ArchiMate-Konzept „Geschäftsereignis“. In Tabelle 1 wurde das „Geschäftsereignis“ viermal in den Literaturquellen angeführt und einheitlich als BPMN-Konzept „Ereignis“ deklariert. Daher steht in der Tabelle 2 in der Zeile „Geschäftsereignis“ und Spalte „Ereignis“ die Zahl vier. Zusätzlich befindet sich in der untersten Zeile sowie der rechtesten Spalte die Summenanzahl der ArchiMate oder BPMN-Konzepte. Für das „Geschäftsereignis“ ist dies in beiden Fällen eins. Demnach ist das „Geschäftsereignis“ lediglich einem BPMN-Konzept zugeordnet und vice versa das „Ereignis“ nur einem ArchiMate-Konzept.

Das gegenüberliegende Extrem wäre beispielsweise der ArchiMate „Geschäftsprozess“. Dieser kann anhand der rechtesten Spalte nach Tabelle 2 auf sechs verschiedene BPMN-Konzepte gemappt werden, wobei der „Pool“ und die „Aktivität“ die höchste Ausprägung aufweisen. Auf der transponierten Seite weist das BPMN-Element „Pool“ die meisten unterschiedlichen Zuweisungen zu ArchiMate-Konzepten auf. Hierbei könnte ein „Pool“ mittels acht verschiedener ArchiMate-Konzepte dargestellt werden, wobei der „Geschäftsprozess“ die zahlreichste Wahl ist. Der ersichtliche visuelle Trend der Zahlen von links oben nach rechts unten hat keine Signifikanz und ist lediglich ein Nebenprodukt der Anordnung der ArchiMate- und BPMN-Elemente.

3.3 Instanziierbarkeit im Mapping

Mit der aggregierten Darstellung der Mappings zwischen ArchiMate und BPMN aus Tabelle 1 wurde die vorhandene Literatur allgemein zusammengefasst. Für diese Arbeit gilt es, wie auch in der Forschungsfrage in Kapitel 1.2 beschrieben, ein Mapping zwischen den beiden Artefakten zu finden, welches auch den Aspekt der Instanziierbarkeit inkludiert. Das Konzept der Instanziierbarkeit, als auch dessen Zusammenhang mit BPMN-Prozessen wurde bereits in Kapitel 2.3.3 erläutert und ist folglich für die Erarbeitung des eigenen Mappings relevant.

Unter Betrachtung der Instanziierbarkeit ist es nicht möglich, direkt die Mappings aus Tabelle 1 als auch Tabelle 2 zu verwenden. Um dies plakativ darzustellen, dient wieder das ArchiMate „Geschäftsereignis“ welches gemäß der Tabelle 2 eindeutig dem BPMN „Ereignis“ widerspiegelt. Ein „Ereignis“ ist jedoch lediglich ein Überbegriff, welcher indiziert, dass etwas im Laufe des Prozesses erfolgt (Silver, 2011). Gemäß dem BPMN-Standard (Object Management Group, Inc, 2013) ist das „Ereignis“ ein abstraktes Konzept, welches sich in drei weiteren Subtypen, dem Start-, End- und Zwischenereignis aufteilt. Damit jedoch ein jeglicher BPMN-Prozess als syntaktisch valide gilt, muss mindestens ein Start- und ein End-Ereignis gegeben sein (Singer, 2019). Demnach steht bereits mit dieser Einschränkung fest, dass ein eins zu eins Mapping zwischen den einzelnen ArchiMate- und BPMN-Konzepten, in Bezug auf die Instanziierbarkeit, nicht möglich ist.

Zur Lösung des beschriebenen Problems und zur Umsetzung des Artefakts in Form einer technischen Implementierung ist ein Ansatz erforderlich, der die Zuordnung eines ArchiMate-Konzepts zu mehreren Subtypen eines BPMN-Konzepts ermöglicht. Diesbezüglich bietet der

ArchiMate Standard (The Open Group Standard, 2022a) selbst zwei Mechanismen an. Diese sind das Hinzufügen von Attributen sowie die Spezialisierung einzelner Konzepte. Über die jeweilige Modellierungssoftware können modellierte Konzepte mit Attributen erweitert werden. Die Attribute besitzen definierte Datentypen und können mit konkreten Werten versehen werden (WIERDA, 2021). Demnach können sich zwei modellierte „Geschäftsereignisse“ mittels diverser Attribute unterscheiden, umso etwa den Subtypen-Äquivalenten in BPMN widerzuspiegeln.

Die zweite Möglichkeit, die Spezialisierung, agiert ähnlich der Vererbung. Es werden die Strukturen einer übergeordneten Klasse weitervererbt und mit spezifischen Eigenschaften versehen. Dadurch wäre beispielsweise eine alternative Notation möglich (The Open Group Standard, 2022a).

Aufgrund der Simplizität in der Implementierung und der Verwendung wird für die Entwicklung des Artefakts auf das Hinzufügen von Attributen gesetzt. Zusätzlich müssen dadurch keine erweiterten, konkreten Notationen definiert werden. Mit dem Hinzufügen von Attributen ist folglich seitens ArchiMate die Möglichkeit gegeben, die insgesamt geringere Anzahl an ArchiMate-Konzepten auf die jeweiligen BPMN-Elemente zu mappen. Demnach kann beispielsweise das ArchiMate „Geschäftsereignis“ auf das instanziierbare BPMN Start-Ereignis mittels eines zusätzlichen Attributs zugewiesen werden.

Um den nächsten Schritt im Mapping zwischen ArchiMate und BPMN zu setzen, muss eruiert werden, welche BPMN-Elemente instanziierbar sind. Die beiden im vorherigen Unterkapitel angeführten Tabellen treffen keine Aussagen über die tatsächliche Instanziierbarkeit der jeweiligen BPMN-Elemente, da dies in den angeführten Literaturquellen nicht der Fokus war. Die Instanziierbarkeit von BPMN-Elementen wurde bereits in den Unterkapiteln 2.3.2 sowie 2.3.3 angeführt und besteht darin, dass ein Token-Konzept für die Ausführung von Prozess-Instanzen eingesetzt wird.

Die Definition der instanziierbaren BPMN-Elemente selbst erfolgt über die allgemein ausführbare Subklasse (Object Management Group, Inc, 2013). Die in dieser Konformitätsklasse enthaltenen Elemente sind in der nachfolgenden Abbildung 16 dargestellt. Diese wurden dem offiziellen BPMN-Standard (2013) entnommen und bilden somit die Grundlage für diverse BPMN-Engines. Die konkrete Umsetzung dieser Elemente kann je nach Implementierung der jeweiligen Prozess-Engine variieren. Dabei können für bestimmte Elemente proprietäre Attribute realisiert werden, die nicht im Standard abgedeckt sind.

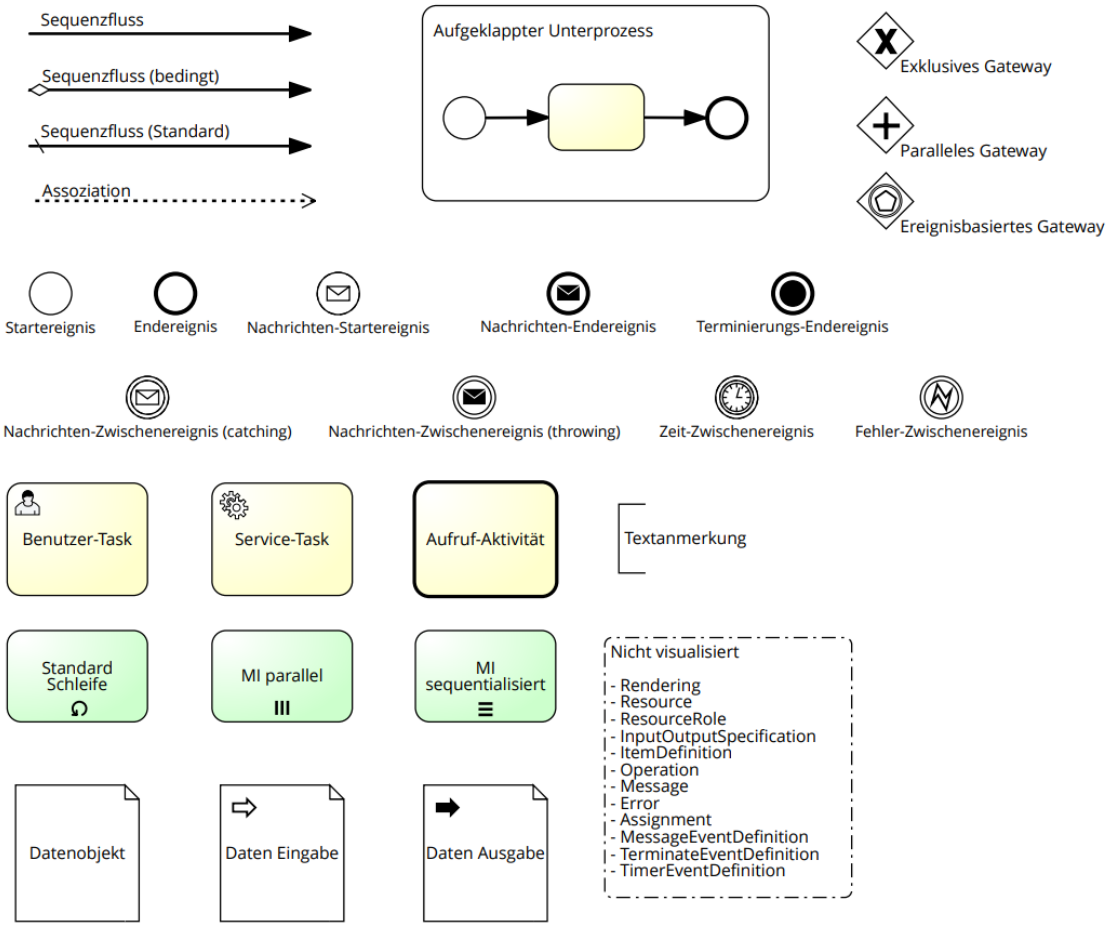

Abbildung 16: Allgemein ausführbare Subklasse, in Anlehnung an (Object Management Group, Inc, 2013, S. 6–8)

Die Abbildung 16 visualisiert alle 38 BPMN-Elemente, welche der allgemein ausführbaren Subklasse entsprechen, wobei nicht alle grafisch notiert werden können (Object Management Group, Inc, 2013). Neben den durch Symbolen notierten Elementen werden in der allgemein ausführbaren Subklasse zusätzliche Unterklassen und Attribute angeführt, welche für die Instanziierbarkeit notwendig sind (Silver, 2011). Ein Beispiel dafür ist das Zeit-Zwischenereignis. Dieses besitzt eine konkrete, hierbei ikonografisch als Ziffernblatt dargestellte Syntax, als auch das nicht visualisierbare Attribut „TimerEventDefinition“.

Wenn die Abbildung 16 detaillierter betrachtet wird, ist zu erkennen, dass diverse Basiselemente aus der beschreibenden Subklasse fehlen. Insbesondere die Elemente „Pool“ und „Lane“ sind nicht in der ausführbaren Subklasse enthalten. Zusätzlich ist lediglich eine limitierte Anzahl an Zwischenereignissen gegeben. Für die Implementierung des Artefakts dient diese limitierte Anzahl an Elementen nach Abbildung 16, inklusive zweier Ausnahmen. Im vorherigen Unterkapitel wurde in Tabelle 2 mittels Zahlen verdeutlicht, dass gewisse Gemeinsamkeiten zwischen ArchiMate und BPMN in Bezug auf die Elemente „Pool“ und „Lane“ bestehen. Zusätzlich wird durch diese ein essenzieller Mehrwert in der Modellierung von Geschäftsprozessen generiert, nämlich, dass die zugehörigen Verantwortlichkeiten repräsentiert werden. Angesichts dessen werden in den nachfolgenden Schritten sowie der Entwicklung des Artefakts die beiden Elemente „Pool“ und „Lane“ explizit berücksichtigt.

3.4 Definition des Mappings zwischen ArchiMate und BPMN

In den vorherigen Unterkapiteln wurde schrittweise der Weg zum Mapping zwischen ArchiMate und BPMN erarbeitet. Initial wurde vorhandene Literatur gesichtet und deren Ergebnisse sowie Methoden beschrieben und aggregiert. Des Weiteren wurde auf der Seite von ArchiMate das Hinzufügen von Attributen als Lösung identifiziert, um den vorhandenen Einschränkungen entgegenzuwirken. Im Fall von BPMN bestehen Limitierungen in Bezug auf die instanziierbaren Elemente (Object Management Group, Inc, 2013). Mit der gesammelten Literatur und deren sequenziellen Aufarbeitung ist die fundierte Basis gegeben, um den ersten Teil des Artefaktes gemäß der wissenschaftlichen Rigorosität nach Österle et al. (2010) zu implementieren.

Der erste Teil des Artefaktes adressiert einen Unterpunkt aus der in Kapitel 1.2 etablierten Forschungsfrage. Im Fokus steht dabei der Aufbau eines Mappings zwischen ArchiMate und BPMN unter Betrachtung der Instanziierbarkeit. Aus den fünf erarbeiteten Literaturquellen sowie den technischen Spezifikationen von ArchiMate und BPMN stehen die folgenden Informationen zur Verfügung.

- Eine Liste der aus der Literatur identifizierten ArchiMate-Konzepte
- Eine Liste der aus der Literatur identifizierten BPMN-Elemente
- Eine Zuordnung der ArchiMate-Konzepte zu BPMN-Elementen, inklusiver einer aggregierten Gewichtung
- Eine Liste der ausführbaren BPMN-Elemente

Jeder dieser angeführten Punkte agiert als eine Art Filter auf die für ein Mapping möglichen Konzepte. Aus der Gesamtmenge aller bestehenden ArchiMate als auch BPMN-Konzepte sind lediglich diejenigen relevant, welche in der Literatur anhand von Tabelle 2 identifiziert wurden. Da fünf Literaturquellen mit teilweise detaillierten Ansätzen und Methoden exerziert wurden, ist die Wahrscheinlichkeit gering, dass weitere nicht in den Tabellen enthaltene Konzepte für ein Mapping relevant wären. Demnach gilt die Tabelle 2 als Ausgangsbasis für die möglichen Konzepte und filtert somit bereits alle ArchiMate und BPMN-Elemente auf eine kleinere Teilmenge herunter. Um diese reduzierte Konzept-Liste zusätzlich im Hinblick auf die Gewichtung sowie die Instanziierbarkeit der Konzepte dazustellen, dient die nachfolgende Tabelle 3.

Tabelle 3: Erweiterung der Mapping-Tabelle um die Instanziierbarkeit, eigene Darstellung

ArchiMate-Konzept	BPMN-Konzept	Häufigkeit	Instanziierbarkeit
Gruppierung	Gruppierung	1	Nein
Standort	Pool	2	Schema
Geschäftsereignis	Ereignis	4	Ja
Geschäftsservice	Pool	1	Schema
Geschäftsservice	Aktivität	2	Ja
Geschäftsservice	Subprozess	1	Ja
Geschäftsschnittstelle	Interface	1	Nein
Geschäftsprozess	Pool	3	Schema
Geschäftsprozess	Aktivität	3	Ja
Geschäftsprozess	Subprozess	1	Ja
Geschäftsprozess	Prozess Diagramm	1	Nein
Geschäftsprozess	Lane	1	Schema
Geschäftsprozess	Prozess	2	Nein
Geschäftsfunktion	Pool	1	Schema
Geschäftsfunktion	Aktivität	2	Ja
Geschäftsfunktion	Subprozess	2	Ja
Geschäftsinteraktion	Pool	1	Schema
Geschäftsinteraktion	Aktivität	1	Ja
Geschäftsinteraktion	Subprozess	1	Ja
Geschäftsinteraktion	Kollaboration Diagramm	1	Nein
Geschäftsobjekt	Datenobjekt	3	Ja
Geschäftsobjekt	Nachricht	1	Ja
Produkt	Datenobjekt	1	Ja
Produkt	Nachricht	1	Ja
Geschäftsakteur	Pool	1	Schema
Geschäftsakteur	Lane	1	Schema
Geschäftsakteur	Teilnehmer	2	Nein
Geschäftsrolle	Pool	1	Schema
Geschäftsrolle	Lane	3	Schema
Geschäftsrolle	Teilnehmer	1	Nein
Geschäftsrolle	Partnerrolle	1	Nein
Geschäftskollaboration	Kollaboration	2	Nein
Applikationskomponente	Pool	1	Schema
Applikationskomponente	Lane	2	Schema
Applikationskomponente	Teilnehmer	1	Nein
Applikationskomponente	Service Aktivität	1	Ja
Applikationskollaboration	Kollaboration	1	Nein
Applikationsprozess	Prozess	1	Nein
Applikationsfunktion	Service Aktivität	1	Ja
Applikationsfunktion	Skript Aktivität	1	Nein
Applikationsservice	Aktivität	1	Ja
Applikationsservice	Prozess	1	Nein
Datenobjekt	Datenobjekt	3	Ja
Gerät	Datenspeicher	1	Nein
Artefakt	Datenspeicher	1	Nein
Artefakt	Datenobjekte	1	Ja
Artefakt	Artefakt	1	Nein
Auslösen-Beziehung	Sequenzfluss	2	Ja
Zugriffsbeziehung	Daten-Assoziation	2	Ja
Zugriffsbeziehung	Nachrichtenfluss	1	Nein
Flussbeziehung	Sequenzfluss	1	Ja
Assoziationsbeziehung	Daten-Assoziation	1	Ja
And Verbindungspunkt	Inklusives Gateway	1	Nein
And Verbindungspunkt	Paralleles Gateway	3	Ja
Or Verbindungspunkt	Eventbasierendes Gateway	1	Ja
Or Verbindungspunkt	Exklusives Gateway	3	Ja

Die Tabelle 3 stellt noch einmal alle ArchiMate und BPMN-Elemente aus der Literatur in der Form einer Liste dar. Hierbei werden in der ersten Spalte die ArchiMate Konzepte angeführt, wobei diesen mehreren BPMN-Elementen zugeordnet werden könnten und daher öfter aufgezählt sind. Zusätzlich wird in der dritten Spalte die Häufigkeit angegeben, wie oft eine Verbindungskombination in der gesichteten Literatur enthalten ist. In der letzten Spalte wird der Aspekt der Instanziierbarkeit angeführt. Bei diesem ist zu beachten, dass das angeführte Konzept sowie dessen untergeordnete Konzepte betrachtet wurden.

Als Beispiel dient die dritte Zeile der Tabelle 3 mit dem „Geschäftsereignis“ und dem BPMN „Ereignis“. Das übergeordnete Konzept „Ereignis“ ist nicht direkt instanziierbar, jedoch dessen untergeordnete Konzepte, wie ein „Startereignis“. Da das „Startereignis“ als ausführbares Konzept gemäß der Abbildung 16 angeführt wird, wird für diese Liste das gesamte Konzept mit „Ja“ bewertet. Konzepte, welche nicht instanziierbar sind, werden mit „Nein“ gekennzeichnet. Die explizit im vorherigen Unterkapitel angesprochenen Konzepte der „Lane“ und „Pool“ werden mit „Schema“ versehen. Diese sind nicht instanziierbar, sind jedoch im BPMN-Schema enthalten und liefern aus der Sicht von ArchiMate eine gewisse Struktur, weshalb sie gesondert betrachtet werden müssen.

Im nächsten Schritt der Erarbeitung eines Mappings zwischen ArchiMate und BPMN sind lediglich die Einträge aus Tabelle 3 relevant, welche instanziierbar oder ein „Schema“ sind. Demnach wurden die nicht instanziierbaren aus der Liste entfernt und die Spalte der BPMN-Konzepte alphabetisch sortiert. Dadurch wurden die einzelnen BPMN-Konzepte gruppiert. Innerhalb dieser Gruppen kann mittels der erarbeiteten Häufigkeit bestimmt werden, welches ArchiMate-Konzept am Geigentesten für die Zuordnung zu einem BPMN-Konzept wäre. Diese Gruppierung sowie erste Zuordnungen sind in der nachfolgenden Tabelle 4 visualisiert.

Tabelle 4: Filterung und Gruppierung der Mapping-Tabelle

ArchiMate-Konzept	BPMN-Konzept	Häufigkeit	Instanziierbarkeit
Geschäftsservice	Aktivität	2	Ja
Geschäftsprozess	**Aktivität**	**3**	**Ja**
Geschäftsfunktion	Aktivität	2	Ja
Geschäftsinteraktion	Aktivität	1	Ja
Applikationsservice	Aktivität	1	Ja
Zugriffsbeziehung	**Daten-Assoziation**	**2**	**Ja**
Assoziationsbeziehung	Daten-Assoziation	1	Ja
Geschäftsobjekt	Datenobjekt	3	Ja
Produkt	Datenobjekt	1	Ja
Datenobjekt	Datenobjekt	3	Ja
Artefakt	Datenobjekte	1	Ja
Geschäftsereignis	**Ereignis**	**4**	**Ja**
Or Verbindungspunkt	Eventbasierendes Gateway	1	Ja
Or Verbindungspunkt	**Exklusives Gateway**	**3**	**Ja**
Geschäftsprozess	Lane	1	Schema
Geschäftsakteur	Lane	1	Schema
Geschäftsrolle	**Lane**	**3**	**Schema**
Applikationskomponente	Lane	2	Schema
Geschäftsobjekt	Nachricht	1	Ja
Produkt	Nachricht	1	Ja
And Verbindungspunkt	**Paralleles Gateway**	**3**	**Ja**

Standort	Pool	2	Schema
Geschäftsservice	Pool	1	Schema
Geschäftsprozess	Pool	3	Schema
Geschäftsfunktion	Pool	1	Schema
Geschäftsinteraktion	Pool	1	Schema
Geschäftsakteur	Pool	1	Schema
Geschäftsrolle	Pool	1	Schema
Applikationskomponente	Pool	1	Schema
Auslösen-Beziehung	**Sequenzfluss**	**2**	**Ja**
Flussbeziehung	Sequenzfluss	1	Ja
Applikationskomponente	Service Aktivität	1	Ja
Applikationsfunktion	Service Aktivität	1	Ja
Geschäftsservice	Subprozess	1	Ja
Geschäftsprozess	Subprozess	1	Ja
Geschäftsfunktion	**Subprozess**	**2**	**Ja**
Geschäftsinteraktion	Subprozess	1	Ja

Anhand der Tabelle 4 sind die gefundenen Konzeptpaare visuell mit fetter Schrift und einem grünen Hintergrund dargestellt. Hierbei sind gewisse BPMN-Gruppen gegeben, welche lediglich ein Konzeptpaar besitzen und daher direkt genutzt werden können. Dies sind beispielsweise das „Geschäftsereignis“ mit der einzigen Zuordnung zu einem BPMN-Ereignis sowie der „And Verbindungspunkt“ als „Paralleles Gateway“. Bei BPMN-Gruppen, welche mehrere Konzeptpaare aufweisen, wurde folglich das Paar mit der höchsten Häufigkeit ausgewählt. Für die Gruppe der BPMN-Aktivität sind beispielsweise fünf Paare gegeben, wobei das Paar „Geschäftsprozess“ zu „Aktivität“ die höchste Zahl, drei in diesem Fall, besitzt.

Eine Ausnahme stellt jedoch die „Pool“ Gruppe dar. Hierbei zeigt die Tabelle 4, dass das Paar „Geschäftsprozess“ zu „Pool“ die Häufigkeit von drei besitzt, jedoch trotzdem das ArchiMate Konzept „Standort“ ausgewählt wurde. Dazu wurde eine weitere Gewichtung definiert, nämlich, dass die tatsächlich instanziierbaren Konzepte einen höheren Stellenwert aufweisen als die speziellen Konzepte des „Pool“ und „Lane“. Semantisch als auch syntaktisch ist die zweitbeste Auswahl „Standort“ valide. In der abstrakten Syntax beider Modellierungssprachen können innerhalb eines „Standortes“ sowie eines „Pools“ weitere Konzepte platziert werden. Semantisch bezeichnet der ArchiMate Standard (2022a) den „Standort“ als einen konzeptuellen oder physischen Raum, in welchem sich weitere Konzepte befinden oder ausgeführt werden. Demnach könnten in diesem, weitere Konzepte aus der physischen Ebene enthalten sein, oder als abstrakte Einordnung von Geschäftsprozessen dienen.

Die Tabelle 4 zeigt jedoch, dass für gewisse Gruppen noch keine Zuordnung getroffen werden konnte, da ein Gleichstand in der Häufigkeit der einzelnen Paare besteht. Diese sind die „Datenobjekt(e)“, das „Eventbasierendes Gateway“, die „Nachricht“, sowie die „Service Aktivität“. Für diese Konzeptpaare muss eine detailliertere Analyse erfolgen.

Beginnend mit dem „Datenobjekt“ stehen vier Konzeptpaare zur Verfügung, wobei aufgrund der Häufigkeit lediglich das „Geschäftsobjekt“ sowie das „Datenobjekt“ als relevant angesehen werden. Unter der Betrachtung der Semantik des ArchiMate-Konzepts „Geschäftsobjekt“ repräsentiert dies ein allgemeines Konzept innerhalb einer gewissen Geschäftsdomäne (The Open Group Standard, 2022a). Das „Geschäftsobjekt“ könnte eine einzelne Instanz von Informationen sein, was jedoch nicht der primär technischen Ansicht eines „Datenobjektes“ in

BPMN entspricht. Das ArchiMate „Datenobjekt“ hingegen beschreibt strukturierte Daten, welche für die Automatisierung von Prozessen verwendet werden können (The Open Group Standard, 2022a). Dies entspricht dem Konzept in BPMN. Hierbei ist der Wortlaut als auch die Semantik der beiden Konzepte ident, wobei ein identer Wortlaut alleinig nicht ausreichend für eine Zuordnung wäre.

Das Nächste im Detail anzusprechende Konzept ist das „Eventbasierende Gateway“. Obwohl lediglich ein Konzeptpaar vorhanden ist, muss dieses genauer betrachtet werden, da der „Or-Verbindungspunkt“ bereits dem „Exklusiven Gateway“ zugeordnet ist. Wird folglich dem „Or-Verbindungspunkt“ ein weiteres Konzept als Mapping zugeordnet, so entsteht die Anomalie des sogenannten „Symbol Overload“.

Bei einem „Symbol Overload“ werden mehrere semantische Konstrukte, etwa die beiden Gateways, einem einzelnen Symbol zugeordnet (Moody, 2009). Diese Anomalie wurde bereits bewusst akzeptiert, da ein eins zu eins Mapping nicht möglich ist. Zusätzlich wurde die Anomalie bereits bei dem Konzeptpaar „Geschäftsereignis“ und „Ereignis“ eingesetzt. Semantisch ist die Zuordnung valide, da ein „Eventbasierendes Gateway“ ähnlich einem „Exklusiven Gateway“ agiert, nämlich, dass lediglich eine nachfolgende Abzweigung aktiviert wird. Beim „Eventbasierenden Gateway“ erfolgt die Auswahl basierend auf dem zuerst eintreffenden Ergebnis (Object Management Group, Inc, 2013).

Das BPMN-Konzept der „Nachricht“ stellt einen eigenen Sonderfall dar. Eine „Nachricht“ ist ein instanziierbares Konzept, jedoch steht in der Modellierung kein visuelles Konzept zur Verfügung. Ohne die Möglichkeit der Visualisierung kann kein ArchiMate-Konzept zugeordnet werden, wobei anzumerken ist, dass die beiden Konzepte „Geschäftsobjekt“ sowie „Produkt“ eine BPMN „Nachricht“ nicht repräsentieren würden. Obwohl kein Mapping besteht, darf das Konzept der „Nachricht“ nicht außer Acht gelassen werden.

Für die „Nachricht“ führt Silver (2021) ausführlich an, dass für ausführbare BPMN-Prozesse das Konzept anders zu betrachten ist als bei rein visuellen Diagrammen. Für ausführbare Prozesse können „Nachrichten“ als Aufrufe von bestimmten Webservices angesehen werden, welche dann an die richtige Prozess-Instanz geroutet werden. Innerhalb dieser Prozess-Instanz wird die „Nachricht“ als „Datenobjekt“ gespeichert und so im Prozess weiterverarbeitet. Die richtige Zuordnung der „Nachricht“ an die richtige Prozess-Instanz sowie nachfolgend an die richtige, abfangende Stelle, obliegt der Software, welche den BPMN-Prozess verwaltet (Silver, 2021). Im Rahmen dieser Arbeit wird das „Nachrichten“ Element bei der Transformation von ArchiMate zu BPMN für gewisse Konzepte lediglich im Hintergrund automatisiert angelegt.

Das nach Tabelle 4 zuletzt verbleibende Konzept ist die „Service Aktivität“. Eine Zuordnung zum Konzeptpaar „Geschäftsereignis“ und „Ereignis“ wäre möglich, jedoch semantisch nicht stimmig, da die „Service Aktivität“ primär der Automatisierung dient. Dazu passender ist die „Applikationsfunktion“, die ein automatisiertes Verhalten beschreibt und folglich eine Aktion ausführt. Das erste Konzeptpaar, die „Applikationskomponente“, beschreibt eine autonome und austauschbare Einheit, ist jedoch ein aktives Strukturelement, das keine Aktion ausführt. Zur vollständigen syntaktischen Korrektheit müsste zusätzlich zur „Applikationsfunktion“ die „Applikationskomponente“ angegeben werden, was in diesem Fall bewusst ignoriert wurde.

Auf Basis der bereits aus Tabelle 4 etablierten Konzeptpaare sowie der vertieften Betrachtung der restlichen Konzepte kann ein finales, im Rahmen dieser Arbeit domänenspezifisches, Mapping etabliert werden. Dies ist folglich in Tabelle 5 gezeigt. Die erste Spalte enthält das ArchiMate-Konzept und die zweite Spalte das zugehörige BPMN-Konzept. Da die BPMN-Konzepte sich in Subklassen aufteilen können, sind diese in der dritten Spalte einzeln angeführt. Hierbei werden lediglich tatsächlich instanziierbare Konzepte dargestellt, mit der Ausnahme des Pools sowie der Lane. Auf Grundlage dieses Mappings erfolgt die Implementierung des zweiten Teils des Artefakts im nachfolgenden Kapitel.

Tabelle 5: Finale Mapping-Tabelle, eigene Darstellung

ArchiMate-Konzept	BPMN-Konzept	Instanziierabare (Sub-)klassen
Geschäftsereignis	Ereignis	Startereignis
		Endereignis
		Nachrichten-Startereignis
		Nachrichten-Endereignis
		Terminierungs-Endereignis
		Nachrichten-Zwischenereignis (catching)
		Nachrichten-Zwischenereignis (throwing)
		Zeit-Zwischenereignis
		Fehler-Zwischenereignis
Applikationsfunktion	Service Aktivität	Service Aktivität
Geschäftsprozess	Aktivität	Benutzer-Task
		Aufruf-Aktivität
Geschäftsfunktion	Subprozess	Subprozess
Or Verbindungspunkt	Exklusives Gateway	Exklusives Gateway
Or Verbindungspunkt	Eventbasierendes Gateway	Eventbasierendes Gateway
And Verbindungspunkt	Paralleles Gateway	Paralleles Gateway
Datenobjekt	Datenobjekt	Datenobjekt
		Daten-Eingabe
		Daten-Ausgabe
Standort	Pool	n.a.
Geschäftsrolle	Lane	n.a.
Zugriffsbeziehung	Daten-Assoziation	Daten-Assoziation
Auslösen-Beziehung	Sequenzfluss	Sequenzfluss
		Sequenzfluss (bedingt)
		Sequenzfluss (Standard)

4 INSTANZIIERUNG VON BPMN-PROZESSEN AUS ARCHIMATE-ARCHITEKTUREN

Mit der allgemeinen Wissensbasis aus Kapitel 2, dem aktuellen Forschungsstand aus Kapitel 3 sowie dem erarbeiteten Mapping zwischen ArchiMate und BPMN nach Tabelle 5 erfolgt der zweite Teil der Design- und Entwicklungsphase nach Peffers et al. (2007). Das Ziel des zweiten Teils dieser Phase besteht darin, ein Artefakt zu kreieren, welches die Instanziierung von BPMN-Prozessen aus ArchiMate Architekturen ermöglicht. Demnach wird ein Prototyp programmiert, welcher ArchiMate Architekturen einlesen, anhand des definierten Mappings transformieren und in BPMN-Prozesse konvertieren kann.

Die sukzessive Entwicklung des angesprochenen Prototyps wird in diesem Kapitel verschriftlicht. Initial werden die definierten Anforderungen und Designkriterien erläutert. Anschließend werden die für die Entwicklung eingesetzten Applikationen beschrieben, sowie die tatsächlich programmierten Bestandteile des Prototyps. In den entsprechenden Unterkapiteln wird die Funktionalität der einzelnen Bestandteile erläutert. Abschließend werden die während der Entwicklung aufgetretenen Herausforderungen und bewusst definierten Grenzen zusammengefasst.

4.1 Definierte Anforderungen und Designkriterien des Prototyps

Um den Anforderungen der wissenschaftlichen Rigorosität für die Entwicklung eines Artefaktes in der gestaltungsorientierten Forschung gerecht zu werden, muss das Artefakt zur Lösung der definierten Lösungsziele dienen (Peffers et al., 2007). In Kapitel 1.2 wurde definiert, dass Geschäftsprozesse direkt aus den Unternehmensarchitekturen instanziiert werden sollen. Demnach muss das Artefakt jene Unternehmensarchitekturen einlesen und darauf basierend folglich instanziierbare Geschäftsprozesse generieren können. Im Rahmen dieser Arbeit erfolgt die Abbildung der Unternehmensarchitekturen mittels ArchiMate und die der Geschäftsprozesse mit BPMN. Auf dieser Grundlage werden die nachfolgend aufgeführten übergeordneten Anforderungen für den Prototyp festgelegt und anschließend durch spezifische Designkriterien näher beschrieben.

- Das Artefakt muss ArchiMate-Modelle einlesen können
- Das Artefakt muss instanziierbare BPMN-Modelle generieren können
- Das Artefakt muss ArchiMate-Konzepte in BPMN-Konzepte transformieren können

Beginnend mit der ersten Anforderung muss ermöglicht werden, dass ArchiMate-Modelle technisch einlesbar sind. Für das maschinelle Einlesen wird eine Serialisierung der ArchiMate-

Modelle benötigt. ArchiMate selbst weist als standardisierte Modellierungssprachen mit einem dahinterliegenden Metamodel eine solche Serialisierung in der Form von XML auf (The Open Group Standard, 2022a). Dies ist das sogenannte ArchiMate Model Exchange File Format. Das ArchiMate Model Exchange File Format wurde entwickelt, um einen standardisierten Austausch von ArchiMate-Modellen zwischen diversen Modellierungswerkzeugen zu ermöglichen (The Open Group Standard, 2022b).

Das daraus resultierende Designkriterium ist, dass der Prototyp das ArchiMate Model Exchange File Format technisch unterstützt. Zusätzlich ist anzumerken, dass die einzulesenden Modelle in diesem Format vorliegen müssen. Eine alternative Serialisierung ist nicht zulässig und würde auch der Natur eines standardisierten Formats widersprechen.

Die zweite Anforderung bezieht sich auf den Output des Prototyps. Im Gegensatz zur ersten Anforderung müssen BPMN-Modelle nicht eingelesen, sondern generiert werden. Dies impliziert, dass die resultierenden Modelle syntaktisch korrekt sein müssen. Als Designkriterium gilt folglich, dass ausschließlich syntaktisch valide Modelle erzeugt werden dürfen. Die syntaktische Korrektheit ist für die Instanziierbarkeit essenziell. Spätestens bei der konkreten Instanziierung von BPMN-Prozessen würde die jeweilige BPMN-Engine Fehler retournieren. Aus diesem Grund dürfen keine syntaktisch inkorrekten Modelle an die BPMN-Engine übergeben werden.

Analog zur ArchiMate bietet BPMN eine standardisierte XML-Struktur zur Serialisierung der Modelle an (Object Management Group, Inc, 2013). Zusätzlich kann jedes Modell anhand des hinterlegten Metamodells syntaktisch validiert werden. Neben der syntaktischen Korrektheit ist anzustreben, die von Silver (2011) beschriebenen BPMN-I Serialisierungsregeln nach Möglichkeit einzuhalten. Diese Regeln spiegeln etablierte Best Practices im Bereich der Serialisierung von BPMN-Modellen wider.

Für die dritte und letzte übergeordnete Anforderung dienen die in Kapitel 3 erarbeiteten Erkenntnisse sowie das entwickelte Mapping zwischen ArchiMate und BPMN. Hierbei wurden diverse Literaturquellen aggregiert und systematisch eine Mapping-Tabelle von ArchiMate nach BPMN erarbeitet. Diese betrachtet explizit die Instanziierbarkeit der BPMN-Konzepte, was wiederum der zweiten Anforderung entspricht. Das Artefakt selbst muss folglich die Mapping-Tabelle nach Tabelle 5 implementieren, um somit eine Transformation der Konzepte zu ermöglichen.

4.2 Eingesetzte Applikationen und Bestandteile des Prototyps

Auf Basis der im vorherigen Unterkapitel definierten Anforderungen und Designkriterien erfolgt die technische Programmierung des Prototyps. Als Programmiersprache wurde Java gewählt. Diese Entscheidung beruht auf mehreren Faktoren. Zum einen ist Java plattformunabhängig und der Prototyp könnte folglich auf unterschiedlichen Betriebssystemen ausgeführt werden. Des Weiteren wird der Prototyp in eine einzelne Applikation kompiliert und ist somit portabel und autark.

Der primäre Entscheidungsgrund für die Programmiersprache Java basiert jedoch darauf, dass bereits vorgefertigte Bibliotheken für die Serialisierung von BPMN-Elementen importiert werden können. Diesbezüglich wurden auf die Bibliotheken von Camunda (2018) gesetzt. Anhand der importierten Bibliotheken werden Methoden zur syntaktisch korrekten Serialisierung jeglicher BPMN-Elemente angeboten. Zur Verwaltung der importierten Bibliotheken als auch der Kompilierung des Prototyps dient Apache Maven.

Für die initiale Modellierung der Unternehmensarchitekturen wird Archi eingesetzt. Archi ist ein quelloffener Modellierungseditor für ArchiMate-Modelle (Beauvoir & Sarrodie, 2025). Zusätzlich bietet Archi die Möglichkeit des Hinzufügens von Attributen für einzelne ArchiMate-Konzepte. Das Hinzufügen von Attributen wurde bereits in Kapitel 3.3 als Voraussetzung für die Implementierung des Prototyps besprochen. Für die Instanziierung der aus den ArchiMate-Modellen generierten BPMN-Prozessen dient die Cockpit365-Plattform. Diese nutzt im Hintergrund die Camunda BPMN-Engine zur Instanziierung der jeweiligen Prozesse (ProMind, 2020). Zusätzlich bietet die Cockpit365-Plattform ein übersichtliches Userinterface zur Verwaltung, Instanziierung und Historisierung der einzelnen Prozesse und Prozess-Instanzen.

Der somit als Java-Applikation entwickelte Prototyp besteht aus mehreren Bestandteilen. Die Main-Klasse bildet den Einstiegspunkt der Anwendung und steuert den gesamten Ablauf. Die ArchiMateXMLLoader-Klasse ist für das Einlesen und Verarbeiten der ArchiMate-Modelle zuständig. In der BPMNElementCreator-Klasse werden syntaktisch korrekte BPMN-Elemente generiert. Ergänzend dazu verwaltet die ImplementedTemplates-Klasse vordefinierte Schablonen für „Service Aktivitäten“. Die spezifischen Inhalte und Funktionen dieser Komponenten werden in den folgenden Unterkapiteln ausführlicher erläutert.

4.2.1 Main.java

In Main.java erfolgt die primäre Logik des Prototyps. Es werden ArchiMate Modelle im standardisierten ArchiMate Model Exchange File Format als XML-Dateien geladen, in die korrespondierenden BPMN-Elemente transformiert, validiert und als instanziierbares BPMN-Modell gespeichert. Als Basis für die Transformationen dient die in Kapitel 3.4 erarbeitete finale Mapping-Tabelle. Bevor jedoch eine Transformation erfolgen kann, muss das ArchiMate-Modell als XML-Datei geladen werden. Hierbei wird auf Standard-Java-Funktionen gesetzt, um ein valides XML als Parameter einzulesen und verarbeiten zu können.

Nach dem Laden des ArchiMate-Modells wird mithilfe der verwendeten Camunda Bibliotheken (2018) die Serialisierung des BPMN-Modells durchgeführt. Initial wird ein leeres BPMN-Modell generiert. Das noch leere Modell wird anschließend mit dem Definitions-Element als Root-Konzept sowie den jeweiligen Namespaces befüllt. Das Definitions-Element sowie dessen direkt untergeordneten Namespaces müssen in jedem BPMN-Modell enthalten sein, damit diese als syntaktisch valide gelten (Object Management Group, Inc, 2013). Anschließend wird ein leeres Diagramm sowie die zugehörige Plane generiert. Diese Elemente stellen die Basis für das XML-Schema von BPMNDI-Elementen dar.

BPMNDI-Elemente bilden das grafische Modell, beziehungsweise die optische Visualisierung, des BPMN-Modells selbst ab (Silver, 2011). Hierbei gilt anzumerken, dass ein BPMN-Modell ohne grafische Darstellung syntaktisch valide ist und die Visualisierung vollkommen entfernt werden könnte (Singer, 2019). Demnach hätte in der Entwicklung des Prototyps auf die Visualisierung der BPMN-Konzepte verzichtet werden können. Da dies jedoch zu einer erheblichen Unverständlichkeit der generierten Modelle führen würde, wurde beschlossen, die Visualisierung mittels BPMDI-Elemente zumindest rudimentär zu implementieren. Folglich werden alle Beziehungen zwischen Elementen mit geraden, direkten Linien implementiert, ohne den optisch ansprechenderen Abzweigungen, welche als Waypoint bezeichnet werden (Object Management Group, Inc, 2013).

Die erste spezifische Logik in Main.java beschäftigt sich mit den in Tabelle 4 als „Schema“ bezeichneten Elementen. Dies sind die BPMN-Konzepte „Lane“ und „Pool“ welche nicht direkt instanziierbar sind, jedoch aufgrund der Notwendigkeit zur Darstellung von Strukturen implementiert wurden. Gemäß der erarbeiteten Mapping-Tabelle nach Tabelle 5 sind dies die ArchiMate-Konzepte „Geschäftsrolle“ und „Standort“. Folglich werden zuerst alle „Standort“ ArchiMate-Konzepte aus der eingelesenen ArchiMate-Datei extrahiert und in „Pools“ konvertiert.

Um gemäß dem BPMN-Standard (2013) syntaktisch korrekt zu sein, wird zusätzlich eine „Collaboration“ als auch ein „Teilnehmer“ generiert. Jeder BPMN-Prozess muss eine „Collaboration“ mit mindestens einem „Teilnehmer“ besitzen. Der „Teilnehmer“ entspricht hierbei dem „Standort“-Konzept aus dem ArchiMate-Modell. Zusätzlich wird ein Prozess-Element generiert, welches alle weiteren Konzepte als Sub-Elemente beinhaltet. Dies entspricht wiederum dem „Standort“-Konzept. Sollte ein ArchiMate-Modell keinen „Standort“ und somit keinen „Pool“ aufweisen, so wird automatisch eine „Collaboration“ mit genau einem „Teilnehmer“ und ein Prozess generiert.

Nachträglich wird das ArchiMate-Modell nach „Geschäftsrollen“, den „Lanes“ durchsucht. Diese „Geschäftsrollen“ müssen einem „Standort“ zugeordnet sein, da eine „BPMN-Lane“ jeweils einem solchen „Pool“ zugeordnet sein muss. Für die Zuordnung der „Lanes“ zu den „Pools“ werden auf die Konzepte der ArchiMate-Beziehungen gesetzt. In ArchiMate sind zwei Möglichkeiten gegeben, um Beziehungen zwischen Konzepten visuell darzustellen, wie folglich in Abbildung 17 gezeigt (The Open Group Standard, 2022a).

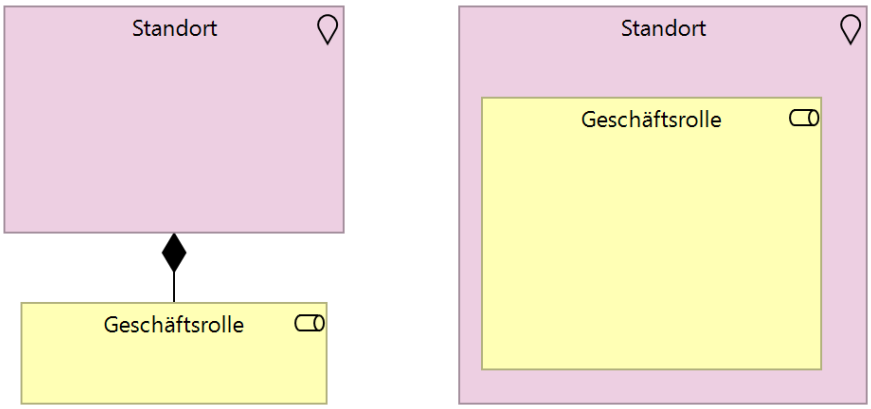

Abbildung 17: Unterschiedliche Visualisierung von ArchiMate-Beziehungen, eigene Darstellung

In Abbildung 17 werden auf der linken Seite alle drei ArchiMate Konzepte grafisch dargestellt. Auf der rechten Seite hingegen werden lediglich zwei Konzepte visualisiert, wobei der „Standort“ die „Geschäftsrolle“ vollkommen einschließt. Demnach wird auch mit dieser Möglichkeit eine Beziehung abgebildet. Die Information, welche Beziehung zwischen den Konzepten gegeben ist, ist in der Visualisierung nicht erkennbar. In der XML-Struktur wird diese hingegen serialisiert.

Wenn das gegebene ArchiMate-Modell aus Abbildung 17 durch den Prototyp, genauer gesagt durch Main.java, konvertiert wird, entsteht folgendes BPMN-Modell nach Abbildung 18. Hierbei ist zu erkennen, dass sich die „Lane“ auf der linken Seite außerhalb des „Pools“ befindet. Syntaktisch konnte ein valides BPMN-Modell generiert und serialisiert werden, jedoch widerspricht dies der BPMN-I Serialisierungsregel R9131, nämlich, dass eine „Lane“ darf sich nicht außerhalb eines „Pools“ befinden darf (Silver, 2011).

Um eine regelkonforme Visualisierung der BPMN-Modelle zu gewährleisten, wird demnach innerhalb der ArchiMate Modellierung auf die rechte Variante von Abbildung 17 gesetzt. Es würden jedoch beide Varianten eine valide ArchiMate- als auch BPMN-XML Struktur liefern. Zusätzlich wird automatisiert die Regel R9132 nach Silver (2011) eingehalten, sodass die Breite der Lane an die Breite des Pools angepasst wird.

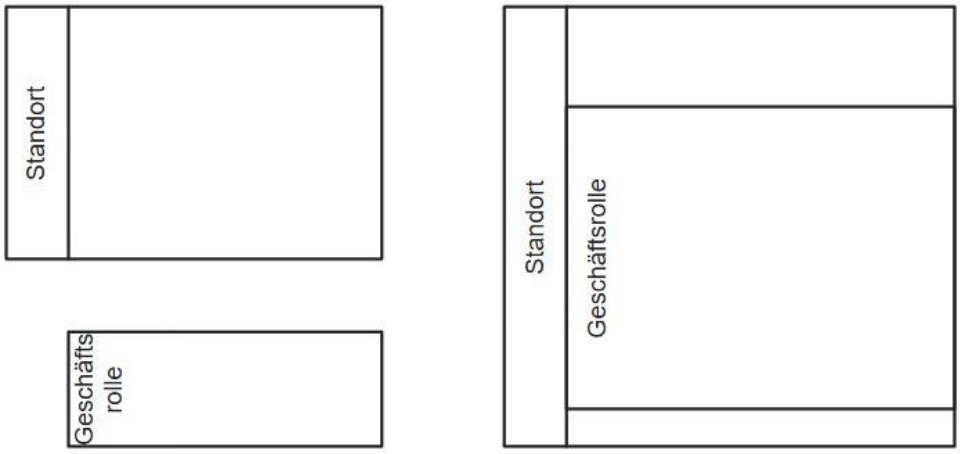

Abbildung 18: Generiertes BPMN-Modell mit unterschiedlicher Notation, eigene Darstellung

Nach den „Schema“ Elementen erfolgt die weitere sukzessive Abarbeitung der Konzeptpaare, beginnend mit den „Geschäftsfunktionen“. Diese repräsentieren Subprozesse in BPMN. Subprozesse können analog zu „Pools“ und „Lanes“ weitere BPMN-Elemente inkludieren und umschließen (Object Management Group, Inc, 2013). Das nächste konvertierte ArchiMate Konzept sind die „Datenobjekte“, wobei die korrespondierenden „BPMN-Datenobjekte“ zusätzliche Elemente im BPMN-Modell benötigen, welche generiert werden müssen.

Nachfolgend werden die Konzeptpaare sowie deren instanziierbare Subklassen aus dem ArchiMate-Modell extrahiert und in BPMN-Konzepte überführt. Hierbei ist anzumerken, dass die Verschriftlichung jedes einzelnen Konzeptpaares, inklusive dessen Eigenschaften und Besonderheiten, nicht zielführend wäre und den Rahmen dieser Arbeit überschreiten würde. Als Grundlage für die zu implementierenden Attribute und Eigenschaften dient die tabellarische Auflistung der allgemein ausführbaren Subklasse des BPMN-Standards (2013).

Da die Instanziierung der Prozessmodelle über die Camunda BPMN-Engine erfolgt, mussten Engine-spezifische Attribute hinzugefügt werden. Diese betreffen insbesondere die Visualisierung von Formularen mit Eingabewerten. Zudem sind sie relevant für die Automatisierung durch Service-Tasks.

Abschließend erfolgt nach der Konvertierung und Visualisierung aller ArchiMate-Konzepte in BPMN-Elemente eine syntaktische Validierung der gesamten XML-Struktur, um ein valides BPMN-Modell zu generieren. Ist die Validierung erfolgreich, so wird das BPMN-Modell im Filesystem gespeichert. Bei syntaktischen Fehlern sowie der Unterlassung von verpflichteten Attributen wird ein Syntaxfehler geworfen und ausgegeben, wie in der nachfolgenden Abbildung 19 gezeigt.

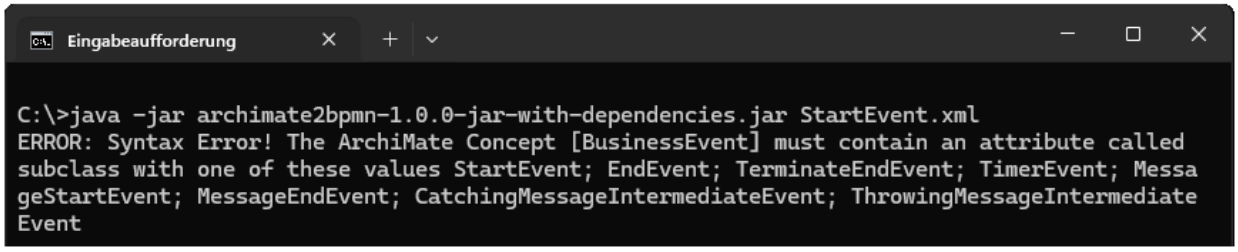

Abbildung 19: Ausgabe von Syntax Fehler, eigene Darstellung

Anhand von Abbildung 19 ist die Ausführung des Prototyps zu erkennen. Es wird versucht, das ArchiMate-Modell „StartEvent.xml“ im standardisierten Austauschformat zu konvertieren. Da jedoch ein „Geschäftsereignis“ als übergeordnete Klasse nicht direkt instanziiert werden kann, muss dieses Konzept ein Attribut namens „Subclass“ aufweisen, welches die instanziierbaren Subklassen nach Tabelle 5 repräsentieren. Mittels der syntaktischen Validierung kann sichergestellt werden, dass inkorrekte XML-Strukturen abgefangen werden. Ohne diesen Schritt könnten nachfolgend syntaktisch inkorrekte Modelle von der BPMN-Engine verarbeitet werden, was unweigerlich zu Fehlern in der Instanziierung führt.

4.2.2 ArchiMateXMLLoader.java

Mittels ArchiMateXMLLoader.java werden diverse Hilfsfunktionen implementiert, um ArchiMate Konzepte aus dem ArchiMate Model Exchange File Format zu extrahieren. Demnach wurden Java-Bibliotheken importiert, welche XML-Strukturen parsen können. So wird das ArchiMate-Modell als XML-Datei eingelesen und dessen Konzepte gemäß Main.java sequenziell geladen und konvertiert. Das Laden eines ArchiMate-Konzeptes erfolgt durch dessen eindeutige Id. Jedes Konzept weist eine eindeutige Id auf, welche während der Modellierung des ArchiMate-Modells automatisiert generiert wird. Anhand der Id kann ein eindeutiges Konzept aus der XML-Struktur extrahiert werden. Zusätzlich bietet ArchiMateXMLLoader.java weitere Hilfsfunktionen an, um mehrere Elemente zu laden.

Sobald ein ArchiMate-Konzept geladen ist, werden zusätzlich alle dazugehörigen Attribute mit einer weiteren Hilfsfunktion gesucht. Jedes in der XML-Struktur vorhandene Attribut besitzt eine

eigene eindeutige Id und ist dem jeweiligen ArchiMate-Konzept untergeordnet. Somit kann ein Attribut mehreren Konzepten untergeordnet sein. Eine Änderung eines Attributs führt dazu, dass diese automatisch auf alle zugeordneten Konzepte übertragen wird, wodurch der Wartungsaufwand reduziert wird.

Neben den ArchiMate-Konzepten und deren Attributen werden mittels ArchiMateXMLLoader.java zusätzlich die Beziehungen zwischen den Konzepten aus der XML-Struktur extrahiert. Analog zu den Konzepten und Attributen weisen die Beziehungen jeweils eine eindeutige Id auf. Eine Beziehung besitzt genau eine Quelle und genau ein Ziel, wobei die Id des referenzierten Konzepts verwendet wird. Zusätzlich können Beziehungen Attribute aufweisen. Dies ist insbesondere für die Implementierung des „Exklusiven Gateways“ relevant, da mittels Attributen Bedingungen auf den Sequenzflüssen abgebildet werden.

Die letzten aus dem ArchiMate Model Exchange File Format extrahierten Informationen sind die visuellen Eigenschaften der Konzepte. Diese werden über eine X- und Y-Koordinate positioniert und die Größe durch eine Breite und Höhe spezifiziert. Zusätzlich wird die Farbe des ArchiMate-Konzeptes sowie dessen Schriftgröße angegeben. Die Farbe und Schriftgröße werden vom entwickelten Prototyp nicht eingelesen, da, wie bereits in Kapitel 2.1.3 erwähnt und in Abbildung 3 gezeigt, die konkrete Syntax keine Bedeutung innehat. Die farbliche Darstellung der generierten BPMN-Konzepte wird durch die Einstellungen der verwendeten BPMN-Engine definiert.

4.2.3 BPMNElementCreator.java

Anhand von Main.java wird der Ablauf zur Extraktion der ArchiMate-Konzepte und deren Konvertierung in BPMN-Elemente definiert. Durch ArchiMateXMLLoader.java werden die ArchiMate-Konzepte sowie deren zugehörigen Informationen geladen. Für die anschließende Transformation in BPMN-Elemente wurden wiederverwendbare Hilfsfunktionen in BPMNElementCreator.java geschrieben. Technisch wird jedes BPMN-Element gemäß einer definierten XML-Struktur definiert. Demnach hätten diese XML-Strukturen manuell generiert werden können. Dies wäre jedoch fehleranfällig und mit deutlich erhöhtem Aufwand verbunden gewesen. Daher wurde wie bereits in Kapitel 4.2 erwähnt auf die Bibliotheken von Camunda (2018) zurückgegriffen. Anhand der importierten Bibliothek können vordefinierte Methoden genutzt werden, um syntaktisch valide BPMN-Elemente zu serialisieren.

Über BPMNElementCreator.java wird jedes BPMN-Element gemäß der dazugehörigen Klasse neu instanziiert. Eine ArchiMate „Applikationsfunktion“ wird beispielsweise als neue „Service Aktivität“ im BPMN-Modell instanziiert. Diese Service-Aktivität-Instanz besitzt folglich alle Attribute, Beziehungen und visuelle Eigenschaften, welche im ArchiMate-Modell definiert wurden. Um eine einfache Wiedererkennung in der serialisierten BPMN-Datei zu ermöglichen, wird die eindeutige Id des ArchiMate-Konzeptes wiederverwendet. Demnach besitzt die instanziierte BPMN-Service-Aktivität dieselbe Id wie dessen ursprüngliche „Applikationsfunktion“.

In den Hilfsfunktionen von BPMNElementCreator.java wird zusätzlich eruiert, in welcher „Lane“ und in welchem „Pool“ sich das jeweilige Konzept befindet. Dazu sind zwei Möglichkeiten

gegeben. Zum einen könnte die visuelle Position des ArchiMate-Konzeptes relativ zur umrandenden „Geschäftsrolle“ oder „Standort“ genutzt werden. Andererseits könnte eine nicht visualisierte, aber in der XML-Struktur vorhandene, Beziehung verwendet werden. Da die tatsächliche visuelle Position der Konzepte im resultierenden BPMN-Modell vernachlässigbar ist, wurde die serialisierte Beziehung als Zuordnung herangezogen. Des Weiteren dient BPMNElementCreator.java zur automatisierten Generierung von Camunda Formularen, welche Eingabemasken für Startereignisse sowie Benutzer-Aktivitäten repräsentieren.

4.2.4 ImplementedTemplates.java

Die letzte entwickelte Java-Klasse ImplementedTemplates.java bezieht sich ausschließlich auf „Service Aktivitäten“. Eine BPMN-Service-Aktivität dient zur Ausführung von diversen Automatismen, beispielsweise dem Versand einer E-Mail, der Anlage eines bestimmten Objektes innerhalb der BPMN-Engine oder in externen Systemen (Göpfert & Lindenbach, 2013). Im BPMN-Standard (2013) wird die Struktur und verpflichtende Attribute einer „Service Aktivität“ angeführt, jedoch obliegt die Implementierung der jeweiligen BPMN-Engine. Demnach können gewisse „Service Aktivitäten“ nicht eins zu eins zwischen BPMN-Engines ausgetauscht werden.

Innerhalb der eingesetzten Cockpit365-Plattform werden bereits eine Vielzahl an ausgereiften Automatismen in der Form von „Service Aktivitäten“ angeboten (ProMind, 2020). Um den vollständigen Umfang der Vorlagen und Funktionalitäten ausnutzen zu können, mussten diese in den Prototyp integriert werden. Diesbezüglich wurde über einen Schnittstellen-Aufruf die Gesamtheit aller Funktionalitäten als einzelne Datei heruntergeladen. Über ein eigens entwickeltes Kommandozeilenprogramm wurde jeder Automatismus extrahiert und als Java-Methode in ImplementedTemplates.java serialisiert. Dadurch konnte ein händisches und fehlerbehaftetes Anlegen der „Service Aktivitäten“ umgangen werden.

Eine BPMN-Service-Aktivität kann Eingabe- und Ausgabevariablen aufweisen. Jede Variable muss als obligatorisch oder optional definiert sein und einen gewissen Datentyp besitzen. Die Typen der Variablen sind von der BPMN-Engine vorgegeben, was in diesem Fall mit der „CamundaInputParameter“ und „CamundaOutputParameter“ Klasse korreliert. Um mögliche Syntaxfehler zu vermeiden, als auch die Vollständigkeit der implementierten „Service Aktivitäten“ zu gewährleisten, werden Syntaxchecks in ImplementedTemplates.java durchgeführt. In Abbildung 20 ist ein solcher Syntaxcheck abgebildet. Es wird versucht über eine „Service Aktivität“ eine E-Mail zu versenden, wobei das obligatorische Attribut „SENDTOEMAILADDR“ nicht ausgefüllt ist. Ohne eine adressierte Person, an jene die E-Mail gesendet werden soll, verliert die „Service Aktivität“ ihre semantische Bedeutung und ist folglich wertlos.

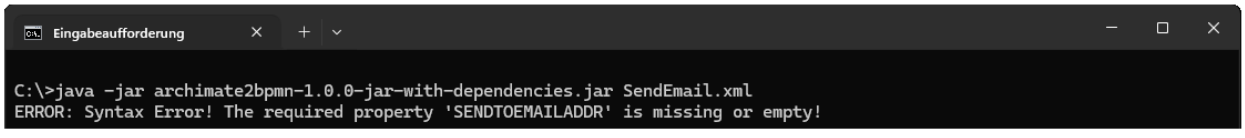

Abbildung 20: Syntax Fehler für ein obligatorisches Attribut, eigene Darstellung

4.3 Herausforderungen und definierte Grenzen des Prototyps

Die technische Programmierung der zuvor beschriebenen Bestandteile des Prototyps erwies sich als weitgehend unproblematisch. Durch die Nutzung der importierten Bibliotheken zur Serialisierung der BPMN-Elemente wurde die syntaktisch korrekte Generierung jener Elemente trivialisiert. Die klar definierten XML-Austauschformate sowohl von ArchiMate als auch von BPMN ermöglichten eine kontinuierliche Validierung der generierten BPMN-Modelle anhand eigens modellierter Referenzbeispiele.

Zu diesem Zweck wurden die erwarteten BPMN-Modelle zunächst unabhängig erstellt und anschließend mit den durch den Prototyp generierten Modellen verglichen. Dieser Ansatz erlaubte eine systematische Analyse und Identifikation möglicher Abweichungen in der XML-Struktur der einzelnen BPMN-Konzepte. Durch diesen iterativen Vergleichsprozess konnten Abweichungen gezielt aufgedeckt und Korrekturen nachhaltig implementiert werden, wodurch eine schrittweise Verbesserung der Generierungslogik sichergestellt wurde.

Technisch bestanden folglich keine maßgeblichen Herausforderungen für die tatsächliche Programmiertätigkeit. Inhaltlich wurden jedoch Schwierigkeiten bei der Implementierung diverser Konzepte festgestellt. Der Prototyp soll so weit wie möglich gemäß dem BPMN-Standard (2013) regelkonform sein. Diesbezüglich ist anzumerken, dass die hundertprozentige Implementierung gewisser Konzepte mit erheblichem Mehraufwand verbunden ist. Demnach wurden für die folglich gelisteten Konzepte Grenzen gezogen und diese, nur rudimentär implementiert.

- Standard-Schleife
- Datenobjekte
- Nachrichten

Beginnend mit der „Standard-Schleife“ ermöglicht diese die sequenzielle Wiederholung einer BPMN-Aktivität und deren instanziierbaren Subklassen. Nach einer Iteration wird eine boolesche Bedingung überprüft, um zu entscheiden, ob die Aktivität erneut ausgeführt oder der Prozess fortgesetzt wird (Object Management Group, Inc, 2013). Die „Standard-Schleife“ wurde im Prototyp nicht implementiert. Die Entscheidung dazu beruht auf dem Umstand, dass die Camunda-Engine selbst diese nicht etabliert hat (camunda Services GmbH, 2019). Da die generierten Prozesse über die Camunda-Engine instanziiert werden sollen, und diese die Möglichkeit der „Standard-Schleife“ nicht anbietet, wäre eine Implementierung, zumindest für diese Arbeit, nicht wertvoll. Es gilt jedoch anzumerken, dass die „Standard-Schleife“ durch eine alternative Anwendung diverser BPMN-Konzepte nach Abbildung 21 umgangen werden kann.

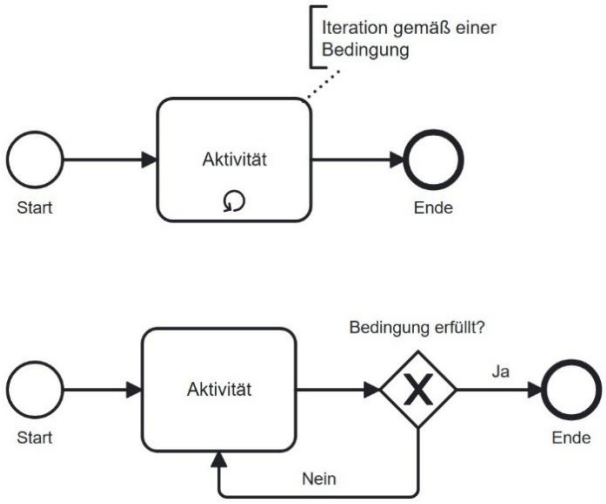

Abbildung 21: BPMN Standard-Schleife, in Anlehnung an (camunda Services GmbH, 2019)

Anhand von Abbildung 21 ist zu erkennen, dass die „Standard-Schleife“ lediglich eine verkürzte Darstellung eines bestimmten BPMN-Konstrukts ist. Der obere Prozess visualisiert die „Standard-Schleife“ und dessen Bedingung anhand einer Textannotation. Im unteren Prozess wird das semantische Äquivalent mittels „Exklusiven Gateway“ abgebildet. Die reine Visualisierung der „Standard-Schleife“ ist für diese Arbeit nicht ausreichend, sondern dessen Instanziierbarkeit, welche jedoch von der Camunda-Engine nicht unterstützt wird (camunda Services GmbH, 2019).

Das nächste, nicht vollständig implementierte Konzept sind die „Datenobjekte“. BPMN-Datenobjekte stellen einen elementaren Bestandteil von ausführbaren Prozessen dar. Mittels der „Datenobjekte“ können Variablen in den Prozess-Instanzen abgebildet und verarbeitet werden (Silver, 2011). Hierbei ist jedoch anzumerken, dass ein ausgeprägter Unterschied zwischen der reinen Visualisierung von „Datenobjekten“ und deren instanziierbare Serialisierung besteht. Silver (2011) führt an, dass bereits während der Entwicklung des BPMN 2.0 Standards Unstimmigkeiten für die Serialisierung der „Datenobjekte“ auftraten. Es wurde diskutiert, ob „Datenobjekte“ direkt mit anderen Konzepten wie Aktivitäten verknüpft werden können. Das Resultat war, dass eine direkte Verbindung nicht möglich ist. Demnach mussten weitere Konzepte implementiert werden, um eine Verbindung oder Zuordnung von „Datenobjekten“ zu ermöglichen.

Hierbei wurde das Konzept der „ioSpecification“ definiert, welches ein „inputSet“ und ein „outputSet“ benötigt. Das „inputSet“ muss wiederum mit einer Liste von „dataInputs“ und das „outputSet“ mit einer Liste von „dataOutputs“ verbunden sein. Zusätzlich muss eine „DataObjectReference“ gegeben sein, welche wiederum das „Datenobjekt“ selbst verlinkt. Diese genannten Konzepte werden nicht visualisiert, sondern dienen lediglich zur syntaktisch korrekten Serialisierung. Die automatische Generation und Implementierung solche verbosen Konzeptstrukturen erfolgt über die jeweilige Modellierungssoftware (Silver, 2011).

Abgesehen von den komplizierten Regeln der Serialisierung von „Datenobjekten“ und deren Subklassen, bestehen weitere Schwierigkeiten. König et al. (2024) führen an, dass das BPMN 2.0 Metamodell nicht ausreichend spezifiziert ist, um mehrere Instanzen desselben Datentyps für multi-instanziierte „Datenobjekte“ in einem Prozess zu ermöglichen. Meyer und Weske (2013)

zeigen auf, dass die Abbildung von „Datenobjekten“ sowie deren Zustandsänderungen während einer laufenden Prozess-Instanz nur schwierig modelliert werden können. „Datenobjekte“ sind daher nicht geeignet um Informationen, welche sich während der Prozess-Instanz verändern abzubilden.

Aufgrund der vorhandenen Schwierigkeiten und komplizierten Serialisierung werden „Datenobjekte“ von diversen BPMN-Engines nicht unterstützt (Funk, 2022). Die Camunda Engine ist eine solche Engine welche „Datenobjekte“ nicht vollständig implementiert und eigene Konzept-Erweiterungen für die Darstellung von Variablen nutzt (Camunda, 2025). Folglich wäre eine vollständige Implementierung der „Datenobjekte“ im Prototyp unnütz, da diese anschließend von der Engine nicht ausgeführt werden können. Zusätzlich ist anzumerken, dass eine hundertprozentige Implementierung aller Möglichkeiten der „Datenobjekte“ den Rahmen dieser Arbeit bei Weitem überziehen würde, da eigens entwickelte Modellierungs-Tools nicht alle Aspekte der „Datenobjekte“ implementieren.

Die letzte bewusst gesetzte Grenze in Bezug auf die Implementierung besteht für das Konzept der „Nachricht“. Analog zu den „Datenobjekten“ unterliegt die Abbildung der „Nachricht“ primär der eingesetzten BPMN-Engine. Dazu werden eigene Schnittstellen-Endpunkte von der jeweiligen Software angeboten um eine „Nachricht“ an Prozesse, genauer gesagt Prozess-Instanzen zu vermitteln (Silver, 2021). Der BPMN-Standard (2013) bietet diesbezüglich das Konzept des „operationRef“ an, welches eine Transformation des Inhalts einer „Nachricht“ zu einem „Datenobjekt“ ermöglicht. Für den Prototyp werden lediglich automatisierte Nachrichten-Konzepte generiert. Die weitere Logik sowie das Anbieten von Schnittstellen-Endpunkten zum Empfangen von Nachrichten erfolgten über die Cockpit365-Plattform.

5 DEMONSTRATION UND EVALUIERUNG DES ARTEFAKTS

Im vorherigen Kapitel wurde die Design- und Entwicklungsphase nach Peffers et al. (2007) abgeschlossen. Es wurde systematisch ein Mapping zwischen ArchiMate und BPMN erarbeitet, Anforderungen und Designkriterien abgeleitet und ein Prototyp in der Form einer Java-Applikation entwickelt. Anhand der Design Science Research Methode gilt es nun den Prototypen, beziehungsweise das gesamte Artefakt, zu demonstrieren und zu evaluieren, wozu dieses Kapitel dient. Beginnend mit der Demonstration, erfolgt diese in allen drei nachfolgenden Unterkapiteln.

Initial wird im Unterkapitel 5.1 anhand eines Anwendungsbeispiels gezeigt, dass der Prototyp technisch funktioniert und welche Arbeitsschritte dabei zu tätigen sind. Anhand dieser Erkenntnisse, dass aus einer ArchiMate Unternehmensarchitektur eine BPMN-Prozess-Instanz gestartet werden kann, erfolgt im darauffolgenden Unterkapitel 5.2 eine Kombination der Phase vier und fünf nach Peffers et al. (2007). Hierbei werden standardisierte Workflow Patterns nach Van Der Aalst et al. (2003) zuerst demonstriert und anschließend auf deren Umsetzbarkeit evaluiert. Mit den Workflow Patterns soll evaluiert werden, welche atomar dargestellten Prozess-Konstrukte möglich sind. Somit kann gezeigt werden, dass der Prototyp zumindest technisch gängige Modellierungsansätze instanziieren kann.

Diese theoretische Evaluierung ist jedoch nicht ausreichend, da die Design Science Research Methode definiert, dass das Artefakt für das ursprüngliche Problem eine Lösung vorsieht (Peffers et al., 2007). Demnach muss das in dieser Arbeit entwickelte Artefakt zusätzlich anhand von praxisrelevanten Beispielen evaluiert werden. Die Workflow Patterns dienen lediglich als theoretische und technische Grundlage. Für den tatsächlichen Praxisbezug, und die zugrundeliegende Problemstellung, dient das dritte Unterkapitel 5.3. In diesem wird auf das Framework der Information Technology Infrastructure Library, kurz ITIL, zurückgegriffen. Mit ITIL werden Best Practices beschrieben, wie das IT-Servicemanagement praxisnah in einem Unternehmen umgesetzt werden könnte.

5.1 Demonstration anhand eines Anwendungsbeispiels

In Übereinstimmung mit der vierten Phase der Design Science Research Methode nach Peffers et al. (2007) wird zunächst die Funktionalität des entwickelten Artefakts demonstriert. Das Ziel dieser Demonstration ist es, die grundsätzliche Fähigkeit des Prototyps zu belegen. Diesbezüglich wurde in Kapitel 4.1 in den Anforderungen definiert, dass das Artefakt ArchiMate-Modelle einlesen, instanziierbare BPMN-Modelle generieren und in diese transformieren kann. Demnach wurde eine einfache ArchiMate Architektur entwickelt, welche schrittweise in ein BPMN-Modell transformiert und folglich konkret instanziiert wird. Die ArchiMate Architektur ist in Abbildung 22 visualisiert.

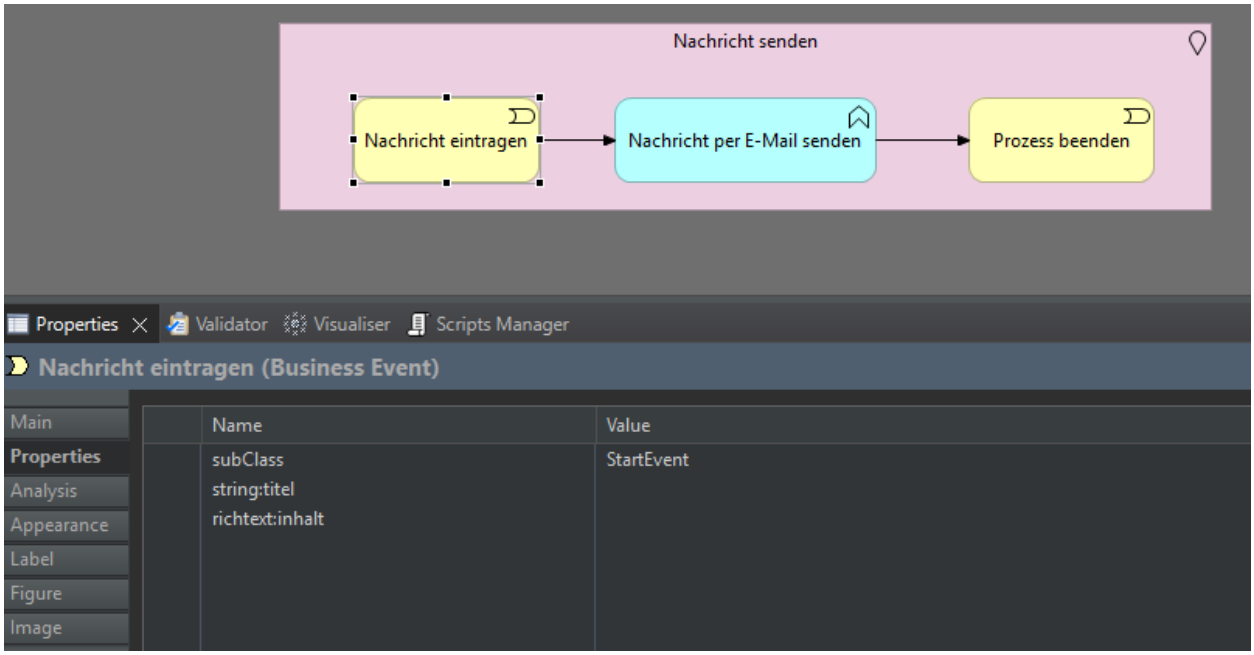

Abbildung 22: ArchiMate-Modell mit Attributen, eigene Darstellung

Im oberen Bereich von Abbildung 22 befinden sich die ArchiMate Konzepte, welche in diesem Fall den Ablauf für das Senden einer Nachricht repräsentieren. Dazu werden die beiden „Geschäftsereignisse“ sowie die „Applikationsfunktion“ von dem „Standort“ umrandet. Der Prozess startet hierbei mit dem Ereignis, das eine Nachricht eingetragen wird. Anschließend wird diese Nachricht per E-Mail gesendet und der Prozess gilt als beendet. Da das ArchiMate Konzept des „Geschäftsereignisses“ gemäß dem Mapping nach Tabelle 5 mehreren instanziierbaren BPMN-Subklassen entspricht, muss dieses weiter spezifiziert werden. Die Spezifizierung erfolgt über das Hinzufügen von Attributen. Das Attribut der „subClass“ definiert, welche BPMN-Subklasse in der Transformation von ArchiMate nach BPMN verwendet werden soll.

Für das Beispiel nach Abbildung 22 wurde das allgemeine Startereignis zum Starten des Prozesses gewählt. Zusätzlich können weitere konzeptspezifische Attribute angegeben werden. Für ein Startereignis sind dies primär Datenfelder, welche zum Start des Prozesses über ein Webformular ausgefüllt werden. Die hierbei definierten Datenfelder sind der Titel sowie der Inhalt der Nachricht. Zusätzlich können spezifische Datentypen gewählt werden, wie beispielsweise ein einfacher String oder ein formatierbarer Text als Richtext.

Nachdem die Unternehmensarchitektur als ArchiMate-Modell modelliert und alle relevanten Attribute hinzugefügt wurden, muss diese gemäß dem ArchiMate Model Exchange File Format exportiert werden. Die eingesetzte Software Archi bietet nativ die Funktionalität an, das Modell in das standardisierte Austauschformart zu exportieren (Beauvoir & Sarrodie, 2025). Sobald das ArchiMate-Modell in der exportierten XML-Struktur vorhanden ist, kann dieses mittels des Artefakts in ein BPMN-Modell transformiert werden. Dies ist in Abbildung 23 gezeigt. Über Main.java aus dem Unterkapitel 4.2.1 wird die ArchiMate XML-Datei eingelesen, Konzept für Konzept transformiert, syntaktisch validiert und als BPMN-Modell gespeichert. Sollten syntaktische Fehler auftreten, so werden diese entsprechend gekennzeichnet und über die Konsole ausgegeben.

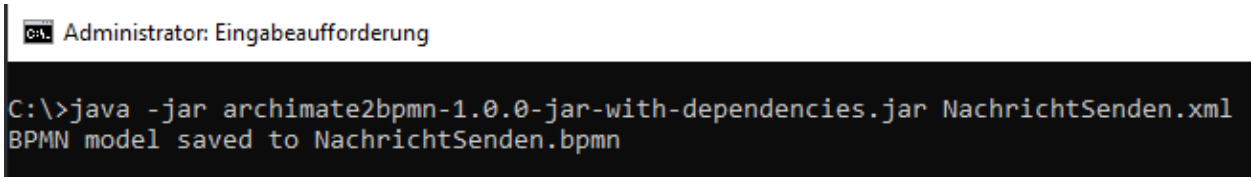

Abbildung 23: Ausführung von archimate2bpmn, eigene Darstellung

Das aus Abbildung 23 generierte BPMN-Modell ist folglich syntaktisch valide und beinhaltet lediglich instanziierbare Konzepte. Daher kann dieses Modell über eine BPMN-Engine ausgeführt werden. Dazu wurde in der Cockpit365-Plattform ein neuer Prozess namens „Nachricht senden“ angelegt und die serialisierte BPMN-Datei direkt eingefügt. In der Abbildung 24 ist zum einen der Prozess und dessen Prozess-ID sowie die visuelle Notation des BPMN-Modells abgebildet. Anhand der Prozess-ID ist zu erkennen, dass diese automatisiert im ArchiMate-Modell generiert und anschließend in das BPMN-Modell übergeben wurde. Der BPMN-Prozess selbst zeigt die Transformation der einzelnen ArchiMate-Konzepte. Der „Standort“ wurde in einen „Pool“ transformiert, welcher ein „Startevent“, eine „Service Aktivität“ und ein „Endevent“ beinhaltet.

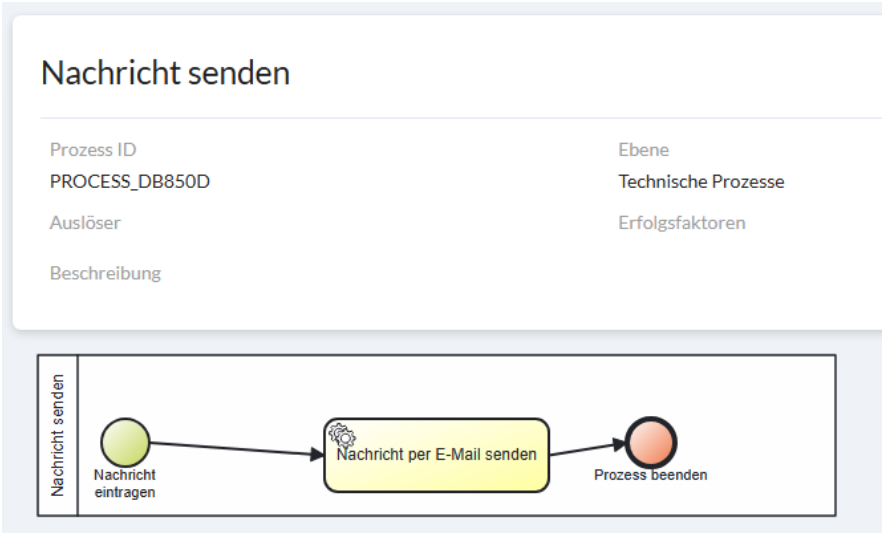

Abbildung 24: Transformierter BPMN mit Prozess-ID, eigene Darstellung

Mit Abbildung 24 kann gezeigt werden, dass eine ArchiMate Unternehmensarchitektur einlesbar und dessen Geschäftsprozesse in BPMN-Modelle transformierbar sind. Das Hauptaugenmerk des Artefakts, genauer gesagt dieser Arbeit, liegt jedoch in der Instanziierbarkeit der Geschäftsprozesse. Demnach muss der Prozess in Abbildung 24 nicht nur visuell dargestellt, sondern auch tatsächlich instanziierbar sein. Über die Cockpit365-Plattform, beziehungsweise die zugrundeliegende Camunda BPMN-Engine, kann der Prozess „Nachricht senden“ als konkrete Prozess-Instanz gestartet werden. Der Prozess beginnt hierbei mit dem Startereignis „Nachricht eintragen“. Da bereits in der Unternehmensarchitektur nach Abbildung 22 die beiden Datenfelder „Titel“ und „Inhalt“ definiert wurden, werden diese folglich in der Prozess-Instanz als Webformular dargestellt. Das resultierende Formular ist in Abbildung 25 ersichtlich.

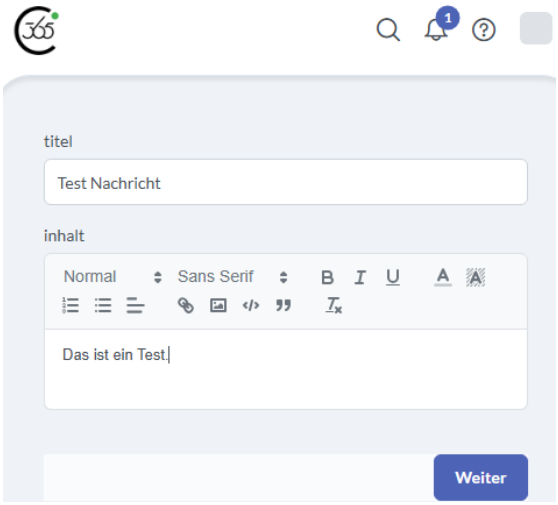

Abbildung 25: Eingabeformular für „Nachricht senden“ Prozess, eigene Darstellung

Für diese konkrete Prozess-Instanz werden die beiden Felder des „Titels“ und des „Inhalts“ beispielhaft mit Testwerten befüllt. Zusätzlich wird der Unterschied bezüglich des gewählten Datentyps ersichtlich. Der „Titel“ wird als String mit einer einfachen Zeile angeboten, und der „Inhalt“ kann mit einem formatierten Text befüllt werden. Die hier eingetragenen Werte werden in der Prozess-Instanz gespeichert und an den nächsten Prozessschritt übergeben. Dies ist die „Serviceaktivität“, welche auf Basis der implementierten Templates nach ImplementedTemplates.java aus Kapitel 4.2.4 eine E-Mail versendet. Die beiden Felder der Prozess-Instanz dienen als Input für die E-Mail, wie in Abbildung 26 gezeigt.

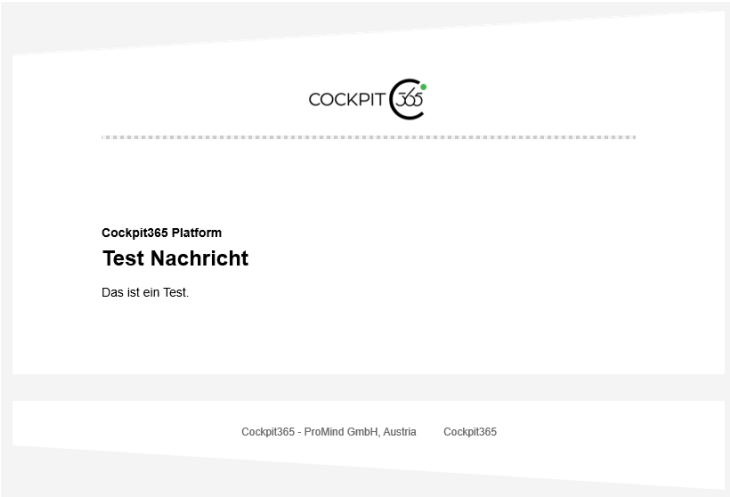

Abbildung 26: Automatisch versendete E-Mail, eigene Darstellung

Nach dem Absenden der E-Mail endet der Prozess in einem Endereignis. Da ein Endereignis keine weiteren ausgehenden Sequenzflüsse aufweisen darf und das im Startereignis generierte Token konsumiert hat, gilt die Prozess-Instanz als beendet. Das Verhalten des Tokens ist anhand von Abbildung 27 abgebildet. Die grüne Umrandung zeigt an, dass das durch das Startereignis generierte Token die Aktivität sowie das Endereignis tatsächlich durchlaufen hat. In diesem Fall wurde um 19:28 die Nachricht per E-Mail versendet und direkt darauf die Prozess-Instanz beendet.

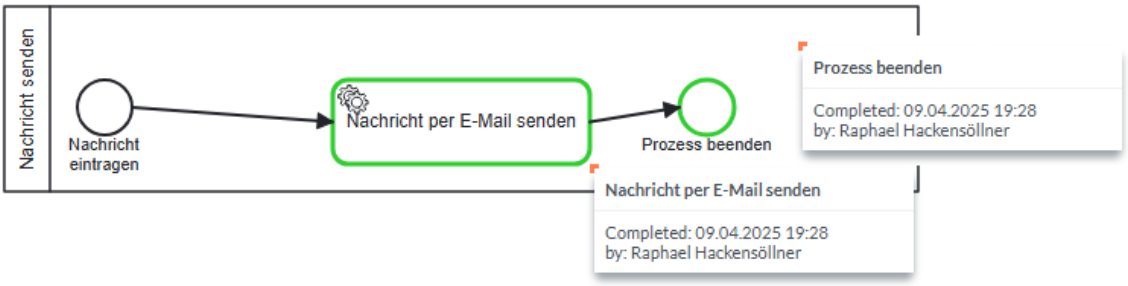

Abbildung 27: Prozess-Instanz für „Nachricht versenden“ Prozess, eigene Darstellung

Mit diesem einfachen Beispiel kann demonstriert werden, dass die tatsächliche Instanziierung des BPMN-Modells möglich ist. Somit sind die definierten Anforderungen gemäß Kapitel 4.1 erfüllt und aus einem ArchiMate-Modell können instanziierbare BPMN-Prozesse generiert werden. Zusätzlich können diese als konkrete Prozess-Instanz einzeln gestartet werden. Das bedeutet, dass das Prozessmodell aus Abbildung 24 erneut gestartet und mit anderen Werten befüllt werden kann. Hierbei gilt jedoch anzumerken, dass lediglich demonstriert wurde, dass das entwickelte Artefakt technisch funktioniert. Ob das Artefakt das in dieser Arbeit beschriebene Problem lösen kann, wird in den nachfolgenden Unterkapiteln evaluiert, beginnend mit Kapitel 5.2, um zu überprüfen, welche Prozess Konstrukte möglich sind.

5.2 Evaluierung von Workflow Patterns

Workflow Patterns, kurz WP, sind wiederkehrende, abstrakte Lösungsstrukturen für typische Herausforderungen in der Prozessmodellierung. Sie wurden entwickelt, um gängige Abläufe innerhalb von Geschäftsprozessen zu standardisieren und zu analysieren. Diese Muster ermöglichen eine strukturierte Darstellung von Prozessen und helfen dabei, die Funktionalität verschiedener Workflow-Management-Systeme, oder WfMS, zu vergleichen (Workflow Patterns Initiative, 2023). Ursprünglich wurden sie von Van Der Aalst et al. (2003) entwickelt, um eine systematische Kategorisierung von wiederkehrenden Prozessstrukturen zu ermöglichen. Diese 20 Patterns fokussieren sich auf den Kontrollfluss von Prozessen und werden in thematische Subkategorien aufgeteilt (Van Der Aalst et al., 2003). Die einzelnen Subkategorien sowie deren enthaltenen Workflow Patterns werden in den nachfolgenden Unterkapiteln dargestellt.

Mit den 20 standardisierten Workflow Patterns soll der entwickelte Prototyp transparent evaluiert werden. Zusätzlich dienen sie als Benchmark für die Modellierungsmöglichkeiten der in dieser Arbeit konzipierten Überschneidung zwischen ArchiMate und BPMN. Es müssen zumindest die zugrunde liegenden Patterns mittels des Prototyps abbildbar sein. Ansonsten wäre eine weitere Evaluierung unnötig, da Basisfunktionalitäten nicht gegeben sind. Diesbezüglich ist jedoch anzumerken, dass nicht alle Patterns möglich sein müssen. Dies ist zusätzlich dem Umstand geschuldet, dass nicht jedes Workflow Pattern in BPMN vollständig abbildbar ist (Wohed, 2006). Gemäß Wohed (2006) werden die Patterns WP15, WP17 und WP18 von BPMN nicht unterstützt und sind nur mittels Behelfslösungen teilweise implementierbar.

Die Evaluierung eines jeden Patterns erfolgte stets denselben Ablauf. Initial wurde überprüft, ob die für das Pattern relevanten Konzepte in der Mapping-Tabelle zwischen ArchiMate und BPMN nach Tabelle 5 enthalten sind. Sollten die Konzepte nicht direkt enthalten sein, so wurde überprüft, ob jene Konzepte durch andere substituiert werden könnten. Wenn die Konzepte, oder deren Alternativen, gegeben waren, so wurde das Pattern als ArchiMate-Modell dargestellt. Anschließend wurde das ArchiMate-Modell über das entwickelte Artefakt in eine instanziierbare BPMN-Datei transformiert. Als finaler Schritt wurde das BPMN-Modell tatsächlich über die BPMN-Engine instanziiert, um zu zeigen, dass eine direkte Instanziierung aus dem ArchiMate Modell möglich ist. Die Ergebnisse werden abschließend im Unterkapitel 5.2.7 zusammengetragen.

5.2.1 Grundlegende Kontrollflusspatterns

Die grundlegenden Kontrollflusspatterns stellen die Grundbausteine eines jeglichen Prozesses dar. Diese inkludieren die ersten fünf Pattern, nämlich die „Sequenz“, „Paralleler Split“, „Synchronisation“, „Exklusive Wahl“ und den „einfachen Merge“ (Wohed, 2006). Bei der Nichterfüllung dieser fünf Patterns wäre eine weitere Evaluierung obsolet. Ohne die Möglichkeit, einen sequenziellen Ablauf darzustellen, Aktivitäten zu parallelisieren oder Entscheidungen zu treffen, wäre eine weitergehende Analyse des entwickelten Artefakts hinfällig. Um Redundanzen zu vermeiden, wurden die ersten drei Patterns sowie WP4 und WP5 gemeinsam behandelt. Beginnend mit der „Sequenz“, dem „Parallelen Split“ sowie der „Synchronisation“ werden diese in der nachfolgenden Abbildung 28 zusammengefasst.

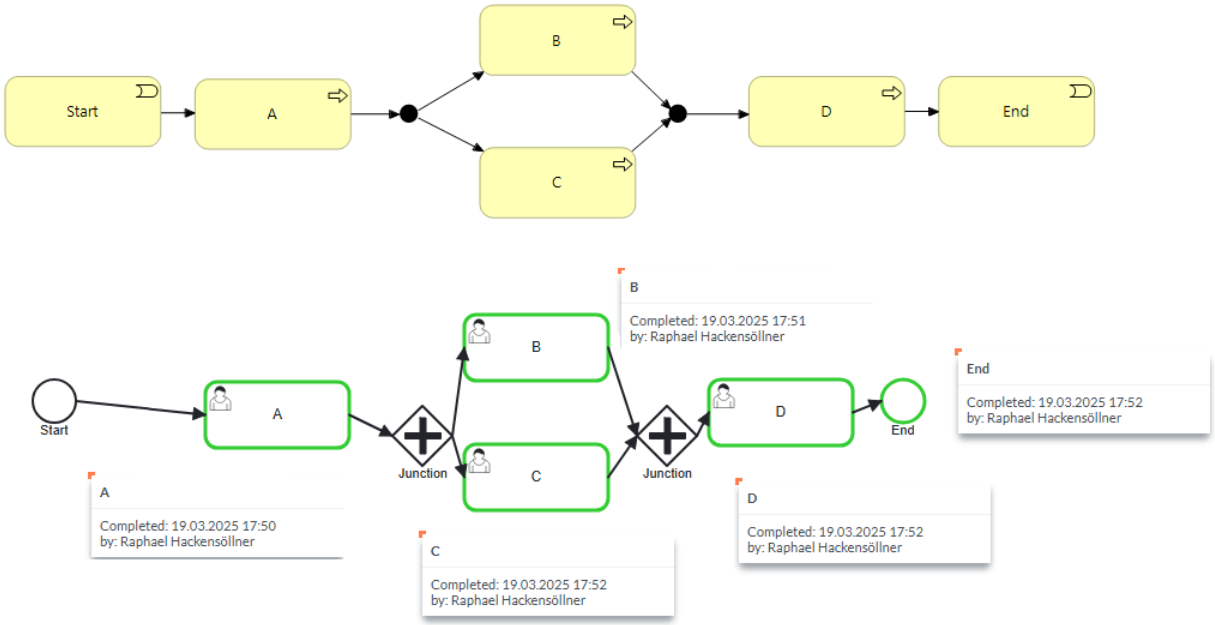

Abbildung 28: WP1 bis WP3, eigene Darstellung

In der Abbildung 28, sowie in allen weiteren Darstellungen, sind die Patterns WP1 bis WP3 als Erstes in der Form eines ArchiMate-Modells modelliert. Das WP1 der „Sequenz“ kann mit dem

ArchiMate Konzept der „Auslösen-Beziehung“ abgefangen werden. Diese wird folglich in BPMN in das Konzept des „Sequenzflusses“ transformiert. Das WP2 des „Parallelen Split“ wird mit dem ersten „And-Verbindungspunkt“ modelliert. Hier erfolgt nach dem Geschäftsprozess „A“ eine parallele Aufteilung in die Geschäftsprozesse „B“ und „C“. Anschließend werden die beiden Geschäftsprozesse mit dem zweiten „And-Verbindungspunkt“ synchronisiert. Dies entspricht dem WP3. Hierbei ist anzumerken, dass der nachfolgende Geschäftsprozess „D“ erst ausgeführt wird, sobald alle eingehenden Sequenzflüsse für den zweiten Verbindungspunkt aktiviert wurden.

Der untere Teil von Abbildung 28 visualisiert die Instanziierung selbst. Es wird das über den Prototyp direkt generierte BPMN-Diagramm gezeigt, als auch dessen Prozess-Instanz über die BPMN-Engine. Die grüne Umrandung deutet darauf hin, dass die Aktivitäten, in diesem Fall Benutzer-Tasks, tatsächlich instanziiert wurden. Zusätzlich wird deren Zeitstempel angezeigt. Dadurch wird eindeutig die „Synchronisation“ ersichtlich. Der Benutzer-Task „B“ wurde um 17:51 abgeschlossen und „C“ erst später um 17:52. Nachdem beide abgeschlossen waren, konnte der Prozessfluss fortgesetzt werden, um anschließend einmal „D“ zu starten. Nach dem Abschluss von „D“ wurde die Prozess-Instanz beendet. Somit sind die Patterns WP1 bis WP3 möglich.

Die nachfolgenden grundlegenden Patterns WP4, die „Exklusive Wahl“, sowie Pattern WP5, der „einfache Merge“ wurden analog zu WP1 bis WP3 in einem Modell zusammengefasst. Dies ist in Abbildung 29 ersichtlich, wobei anstatt von And-Verbindungspunkten, Or-Verbindungspunkte genutzt wurden. In der gezeigten Prozess-Instanz wurde die Wahl zwischen „B“ und „C“ mittels einer Variable namens „Check“ getroffen. In dem Benutzer-Task „A“ kann der Wert der Variable gesetzt werden.

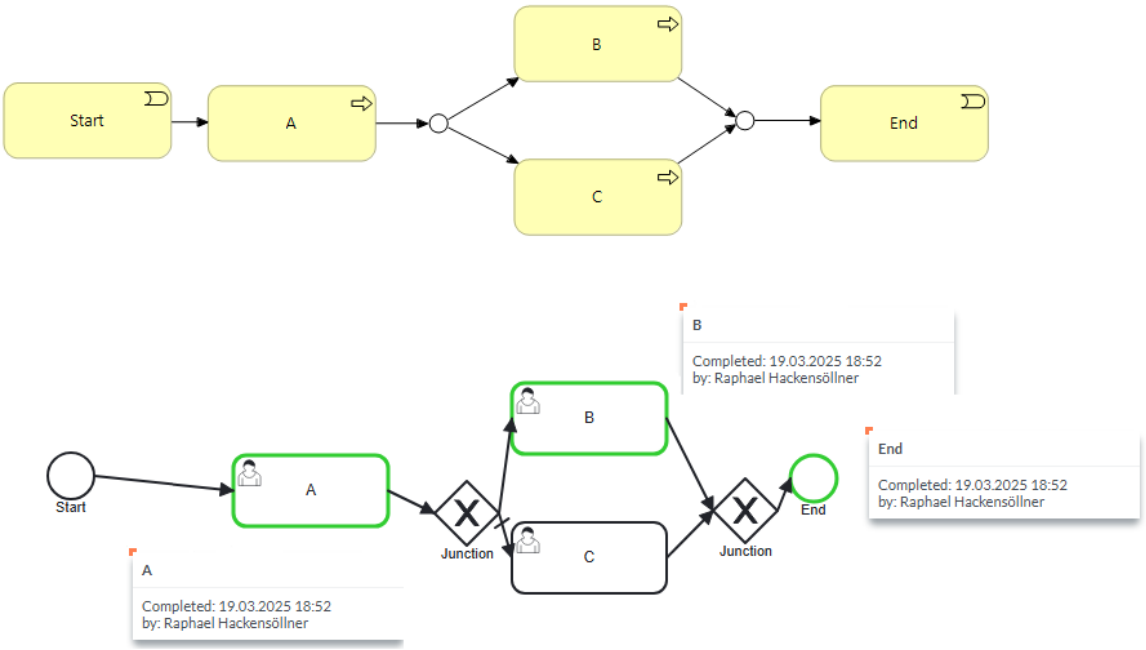

Abbildung 29: WP4 und WP5, eigene Darstellung

Das Setzen der Variable ist in Abbildung 30 visualisiert. Sollte der Wert der Variable „B“ entsprechen, so wird die exklusive Wahl für „B“ getroffen. Alternativ folgt der Prozessfluss dem unteren Weg über „C“. Zusätzlich besitzt der Pfad für „C“ den Standardwert. Sollte daher die

Variable „Check“ einen anderen Wert außer „B“ aufweisen, so wird stets der Weg „C“ genommen. Anschließend werden die Pfade wieder zusammengeführt, was dem WP5 entspricht. Folglich sind die Patterns WP4 und WP5 möglich, wodurch alle grundlegenden Kontrollflusspatterns mit dem entwickelten Artefakt abgebildet werden können. Eine weitere Evaluierung ist somit sinnvoll.

Abbildung 30: Formular für Benutzer-Task A, eigene Darstellung

5.2.2 Erweiterte Verzweigungs- und Synchronisierungspatterns

Die nächste Unterkategorie der Workflow Patterns sind die erweiterten Verzweigungs- und Synchronisierungspatterns. Diese Patterns sollen zeigen, wie aus einem Token, welches durch die Prozess-Instanz fließt, mehrere Tokens instanziiert werden können und wie diese anschließend synchronisiert werden. In der Kategorie sind die Patterns WP6 bis WP9 enthalten (Wohed, 2006).

Beginnend mit dem Workflow Pattern 6 wird die „Multi-Choice“ abgebildet. „Multi-Choice“ bedeutet, dass aus einem eingehenden Sequenzfluss ein oder mehrere Sequenzflüsse entstehen können. Abhängig von den jeweiligen Prozessvariablen können ein oder mehrere Wege aktiviert werden. In BPMN erfolgt dies über das „Inklusive-Gateway“, wie nachfolgend in Abbildung 31 gezeigt. Das „Inklusive Gateway“ ist nicht in der Mapping-Tabelle nach Tabelle 5 zwischen ArchiMate und BPMN enthalten. Zusätzlich kann das Konzept des inklusiven Gateways nicht durch ein anderes substituiert werden. Folglich ist es nicht möglich, über das in dieser Arbeit entwickelte Artefakt das WP6 abzubilden. In der Praxis bedeutet dies, dass entweder alle ausgehenden Sequenzflüsse aktiviert werden oder genau nur ein Sequenzfluss. Es kann daher nicht abhängig von gewissen Konditionen eine abhängige Anzahl an Sequenzflüssen aktiviert werden.

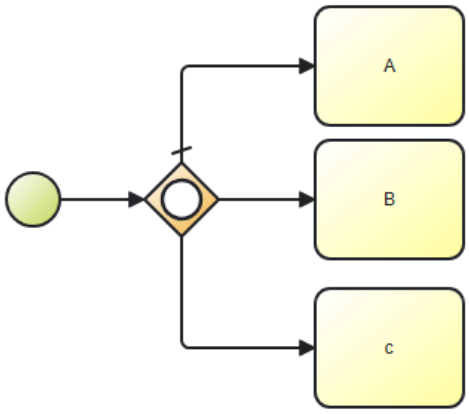

Abbildung 31: WP6 - Multi-Choice nicht möglich, eigene Darstellung

Das Workflow Pattern WP7, der „synchronisierende Merge“, kann als Gegenstück zum WP6 gesehen werden. Anstatt einen oder mehrere Pfade zu starten, werden eine oder mehrere Pfade zusammengefasst und synchronisiert (Wohed, 2006). Diesbezüglich wird wieder auf das Konzept des „Inklusiven-Gateways“ gesetzt, was, wie bereits im vorherigen Absatz erwähnt, nicht im Prototyp enthalten ist. Demnach kann das WP7 nicht über das in dieser Arbeit entwickelte Artefakt abgefangen werden. Zur Visualisierung des Patterns dient die Abbildung 32, bei welcher im Gegensatz zu Abbildung 31 das synchronisierende „Inklusive-Gateway“ hinzugefügt wurde. Zusätzlich gilt anzumerken, dass ein strukturierter Workflow abgebildet wird. Sollte nach beispielsweise der Aktivität „A“ noch eine Aufteilung über ein „Exklusives-Gateway“ erfolgen, so würde ein unstrukturierter Workflow und möglicherweise ein Deadlock entstehen (Wohed, 2006).

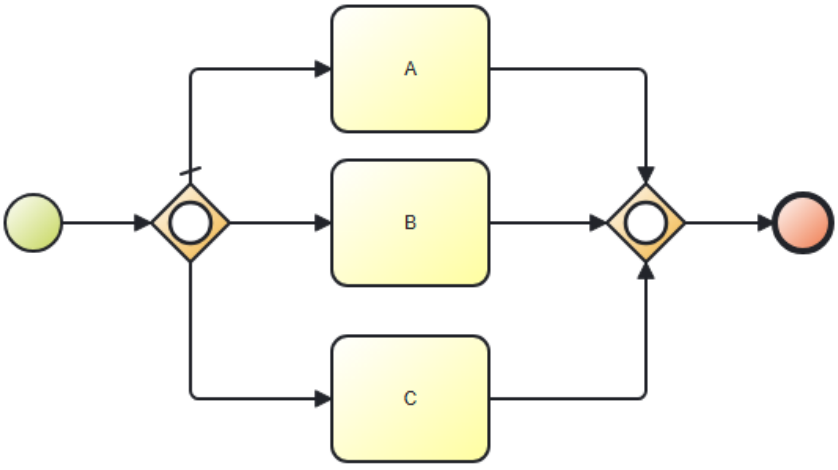

Abbildung 32: WP7 - Synchronisierende Merge nicht möglich, eigene Darstellung

Das „Multi-Merge“ Pattern WP8 stellt ein unsynchronisiertes Zusammenführen von unterschiedlichen Pfaden dar. Es wird für jeden aktiven eingehenden Pfad der nachfolgende Prozess ausgeführt (Wohed, 2006). Die Modellierung des WP8 erfolgt ähnlich dem WP5, indem ein „Exklusives-Gateway“ eingesetzt wird. In Abbildung 33 ist ein instanziierter Prozess abgebildet, welcher das WP8 darstellt.

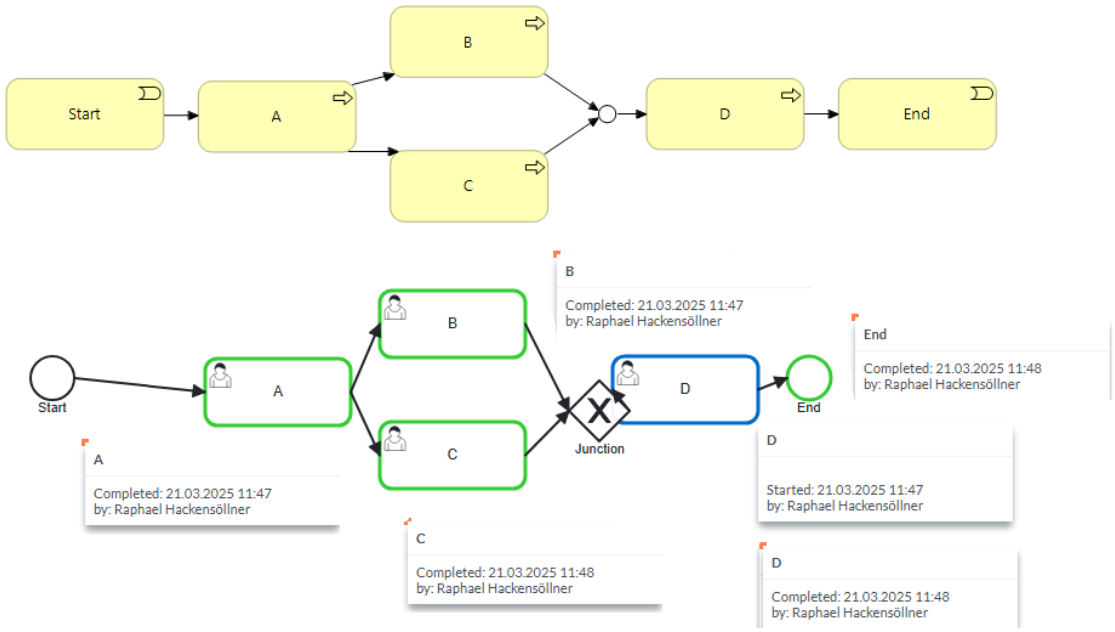

Abbildung 33: WP8 - Multi-Merge, eigene Darstellung

Nach der Aktivität „A“ von Abbildung 33 erfolgt eine Aufteilung in „B“ und „C“. Die Aktivität „B“ wurde um 11:47 beendet, wodurch die nachfolgenden Prozessteile gestartet wurden. Daher wurde „D“ zum ersten Mal um 11:47 gestartet. Eine Minute später um 11:48 wurde „C“ beendet. Folglich wurde „D“ ein zweites Mal gestartet und auch um 11:48 beendet. Somit wurde die Aktivität „D“ für jeden der eingehenden Pfade instanziiert. Das Workflow Pattern 8 ist demnach abbildbar, jedoch ist anzumerken, dass in diesem Beispiel eine offene Instanz von „D“ aktiv bleibt. Zur Vollständigkeit müsste diese Aktivität nachträglich terminiert werden, beispielsweise über ein Terminierungs-Endereignis.

Der „Diskriminator“ ist das letzte Workflow Pattern dieser Subkategorie und zählt zu den komplexeren Mustern. Er wird von BPMN theoretisch unterstützt (Workflow Patterns Initiative, 2023). Wohed (2006) zeigt jedoch auf, dass dieser nur mittels Workarounds oder einem „Komplexen-Gateway“ teilweise abbildbar ist. Als „Diskriminator“ selbst wird die Fähigkeit beschrieben, zwei oder mehrere Pfade auf eine spezielle Art zusammenzuführen. Der erste aktive eingehende Pfad löst die nachfolgenden Prozessschritte aus. Alle nachträglich in den „Diskriminator“ eingehenden aktiven Pfade werden verworfen und generieren keine neuen Tokens. Da dieses Pattern nicht vollständig von BPMN unterstützt wird, wird es nicht zur Evaluierung des Prototyps herangezogen.

5.2.3 Strukturelle Patterns

Mittels der strukturellen Patterns wird evaluiert, ob Einschränkungen bezüglich der Art und Weise, wie Prozesse strukturiert werden, gegeben sind. Die Subkategorie besteht aus zwei Patterns. Diese sind WP10 die „Arbiträre Kreise“, sowie WP11 die „Implizite Terminierung“ (Wohed, 2006).

„Arbiträre Kreise“ können vereinfacht ausgedrückt synonym zu Schleifen gesetzt werden. Demnach wird mittels WP10 evaluiert, ob Schleifen möglich sind, was der wiederholten Ausführung von Konzepten entspricht.

In der nachfolgenden Abbildung 34 wird die Verwendung einer solchen Schleife dargestellt. In der Prozess-Instanz wurde „A“ um 12:28 beendet und anschließend über den standardmäßigen Sequenzfluss zu „B“ geleitet. Nach dem Abschluss von „B“ wurde jedoch wieder zu „A“ ein zweites Mal zurückgekehrt. Folglich wurde „A“ erneut instanziiert und um 12:29 abgeschlossen, wobei anhand einer Variable ein alternativer Weg genommen wurde. Das Pattern WP10 ist somit möglich und es können Schleifen in der Unternehmensarchitektur eingebaut werden, welche folglich instanziierbar sind. Zusätzlich ist anzumerken, dass das Thema der Schleife bereits in Kapitel 4.3 und in Abbildung 21 angesprochen wurde. Das BPMN-Konzept der Standard-Schleife ist nicht implementierbar, jedoch kann eine Schleife auf Basis einer nachgestellten exklusiven Entscheidung gemäß WP10 modelliert werden.

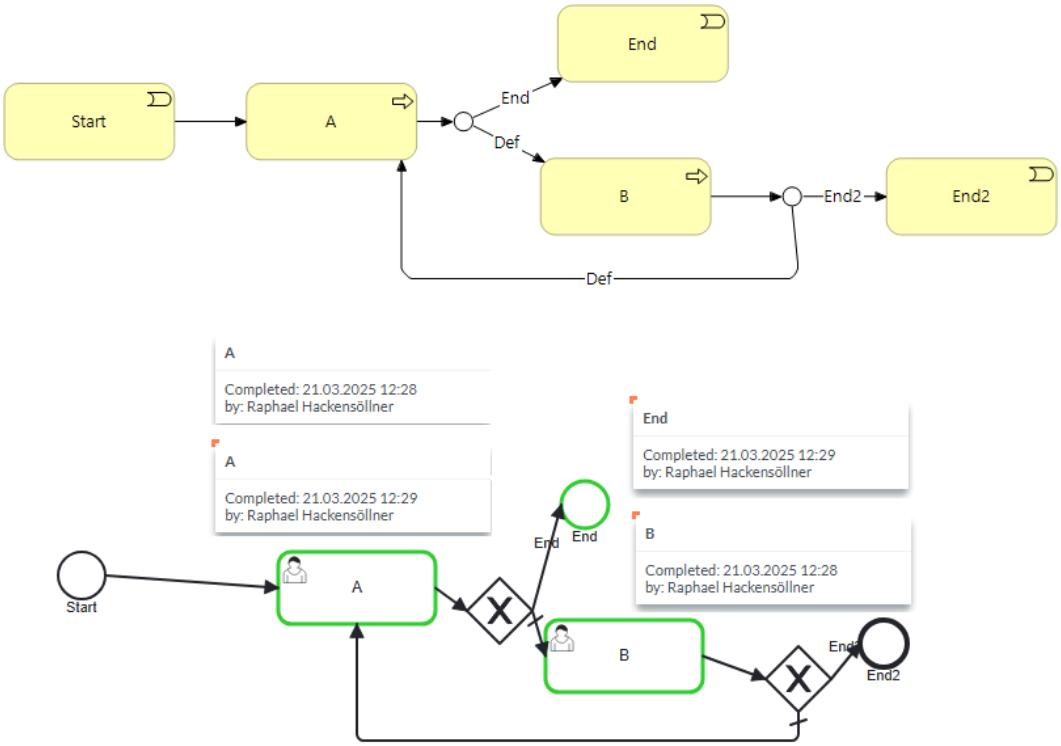

Abbildung 34: WP10 - Arbiträre Kreise, eigene Darstellung

Das zweite strukturelle Pattern ist die „Implizite Terminierung“. Diese sagt aus, dass ein Workflow automatisch endet, sobald keine aktiven oder ausstehenden Aufgaben mehr existieren. Hierbei wird entweder ein Subprozess beendet oder wenn jedes generierte Token konsumiert wurde, die gesamte Prozess-Instanz (Wohed, 2006). Die Abbildung 35 zeigt die „implizite Terminierung“. Hierbei ist anzumerken, dass die Terminierung über ein Geschäftsereignis, beziehungsweise ein Endereignis bereits in den vorherigen Abbildungen aufgrund der syntaktischen Vollständigkeit implementiert wurde. Beide strukturellen Patterns sind folglich über das entwickelte Artefakt abbildbar.

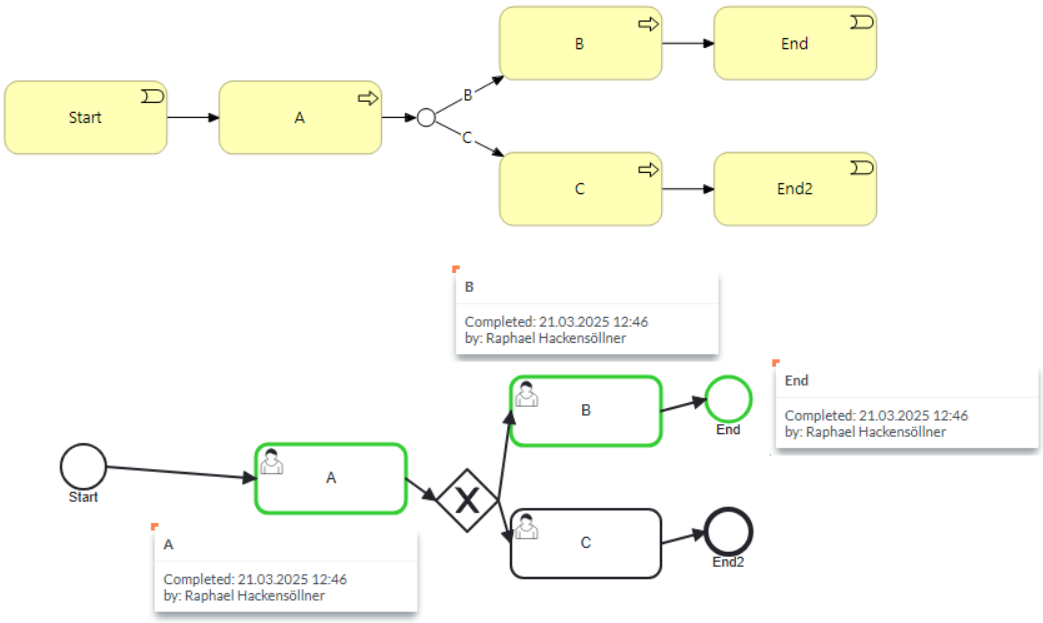

Abbildung 35: WP11 - Implizite Terminierung, eigene Darstellung

5.2.4 Patterns mit mehreren Instanzen

Innerhalb von Prozessen kann die Anforderung gegeben sein, dass Aktivitäten mehrfach ausgeführt werden müssen. Ein Beispiel dafür könnten Bewerbungen für eine offene Stelle sein. Es gelangen mehrere Bewerbungen ein, welche anschließend parallel begutachtet werden. Die Patterns mit mehreren Instanzen beziehen sich auf solche Situationen, bei welchen mehrere Instanzen einer Aufgabe gleichzeitig in einem Prozess aktiv sind. Diese Subkategorie besteht aus den Workflow Patterns WP12 bis WP15 (Wohed, 2006).

Das WP12 sind die „mehreren Instanzen ohne Synchronisierung“. Mehrere Instanzen bedeutet, dass eine Aktivität mehrfach instanziiert werden kann und die jeweiligen Instanzen parallel ausgeführt werden. Ohne Synchronisierung beschreibt, dass nicht alle Prozess-Instanzen aufeinander abgestimmt werden. Wenn beispielsweise die Aktivität „B“ dreimal instanziiert wird, werden drei Tokens generiert. Sobald eine Instanz der Aktivität „B“ erledigt wird, wandert dessen Token weiter und startet den nachfolgenden Prozessfluss. Das Gleiche gilt für die zwei weiteren Aktivität-Instanzen, wodurch der nachfolgende Prozessfluss insgesamt dreimal ausgeführt wird (Wohed, 2006).

Gemäß Wohed (2006) sowie der offiziellen Workflow Patterns Initiative (2023) wird das WP12 von BPMN unterstützt. Dazu muss das Attribut „Flow_Condition“ auf den Wert „none“ gesetzt werden (Wohed, 2006). Dieses Attribut konnte jedoch nicht im BPMN-Standard (2013) gefunden werden. Zusätzlich führt dieser explizit die Patterns WP13 und WP14 an, während das WP12 keine Erwähnung findet. Des Weiteren wurde versucht, das WP12 über die Camunda-Engine nachzustellen, was jedoch nicht gelungen ist. Daher wird im Rahmen dieser Arbeit angenommen, dass das WP12 nicht möglich ist.

Im nächsten Pattern, dem WP13, werden „mehrere Instanzen mit a Priori Design Time Knowledge“ abgebildet. Dabei werden mehrere Instanzen einer Aktivität parallel instanziiert und anschließend synchronisiert. Im Vergleich zum vorherigen Pattern bedeutet dies, dass die dreimalige Instanziierung einer Aktivität „B“ den Folgeprozessfluss nur einmal startet. Alle drei Instanzen warten, bis sie abgeschlossen sind, bevor der Prozess fortgesetzt wird. A Priori Design Time Knowledge definiert, dass zum Zeitpunkt der Modellierung bereits bekannt ist, wie oft die Aktivität instanziiert werden soll (Wohed, 2006).

Die Abbildung 36 zeigt folgendes Szenario für das WP13. Es ist bereits in der Modellierung der Unternehmensarchitektur in ArchiMate bekannt, dass der Geschäftsprozess zweimal parallel auszuführen ist. Demnach wird in der tatsächlichen Prozess-Instanz die Aktivität „A“ zweimal instanziiert. Sobald beide Aktivitäten, in diesem Fall um 13:18 und 13:19, beendet wurden, kann die Prozess-Instanz um 13:19 fortfahren und über das Endereignis abgeschlossen werden.

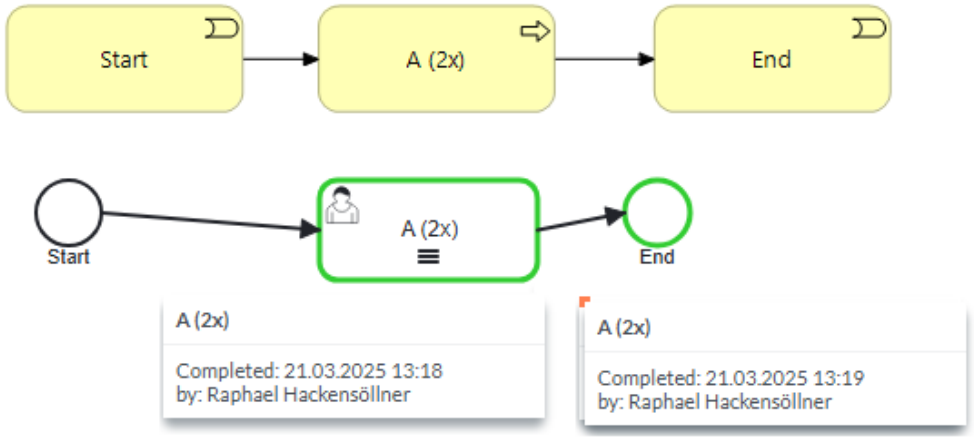

Abbildung 36: WP13 - Mehrere Instanzen mit a Priori Design Time Knowledge, eigene Darstellung

Dass die genaue Anzahl der Instanziierungen von Aktivitäten bereits zum Zeitpunkt der Modellierung bekannt ist, gilt als unwahrscheinlich. Daher dient als Erweiterung zu WP13 das Workflow Pattern 14. Mit dem WP14 werden „mehrere Instanzen mit a Priori Runtime Knowledge“ abgebildet. Mit a Priori Runtime Knowledge bedeutet, dass die Anzahl der Instanzen erst während der Ausführung des Prozesses bekannt ist. Im Laufe des Prozesses wird zum ersten Mal erkannt, wie oft die Aktivität zu instanziieren ist (2006).

Als Beispiel für das WP14 dient die Abbildung 37. Aufgrund einer technischen Einschränkung der verwendeten Camunda-Engine mussten die Ereignisse und die Aktivität in einem „Pool“ umschlossen werden. Daher inkludiert das ArchiMate Modell einen „Standort“. In der gezeigten Prozess-Instanz wurden im Formular des Startereignisses drei Dateien hochgeladen. Die Aktivität „A“ wurde jeweils einmal pro hochgeladene Datei instanziiert, was in diesem Fall zu insgesamt drei Instanzen führte. Dass die Zahl der hochgeladenen Dateien drei war, war zum Zeitpunkt der Modellierung bisher nicht bekannt, sondern erst in der Prozess-Instanz selbst. In einer weiteren Prozess-Instanz könnte diese Zahl beispielsweise fünf oder zehn sein.

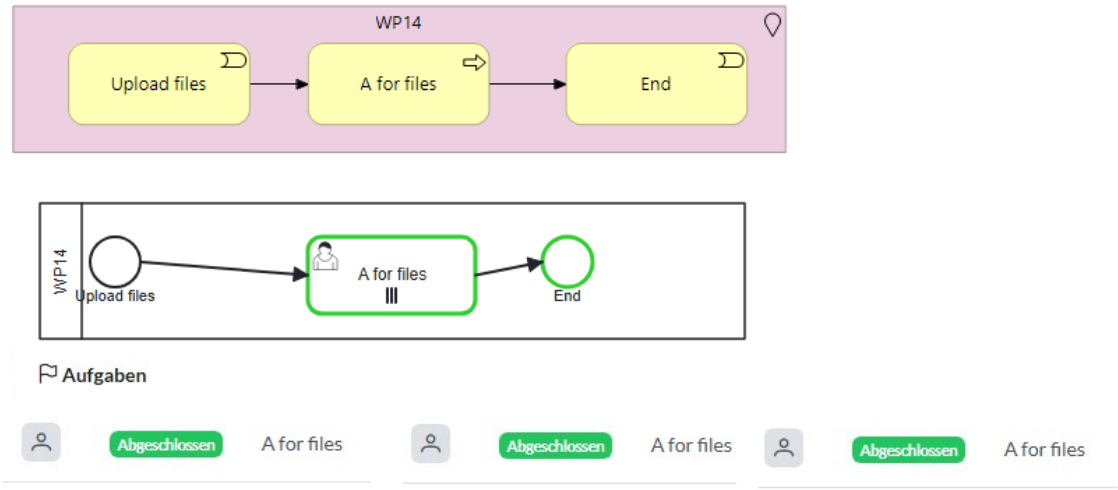

Abbildung 37: WP14 - Mehrere Instanzen mit a Priori Runtime Knowledge, eigene Darstellung

Das letzte Pattern mit mehreren Instanzen ist das Pattern WP15, welches „mehrere Instanzen ohne a Priori Runtime Knowledge“ darstellt. Ohne a Priori Runtime Knowledge bedeutet, dass die genaue Anzahl der Instanzen selbst während der laufenden Prozess-Instanz nicht bekannt ist, und weitere Aktivitäten nachträglich instanziiert werden können. Demnach könnte, wenn die Aktivität „A“ bereits zweimal instanziiert wurde, zusätzliche Instanzen generiert werden. Dies ist in BPMN nicht möglich (Wohed, 2006; Workflow Patterns Initiative, 2023). Da das Pattern somit im gesamten Spektrum von BPMN nicht möglich ist, ist dieses für die Evaluierung des Artefaktes irrelevant.

5.2.5 Zustandsbasierte Patterns

Zustandsbasierte Workflow-Patterns modellieren Szenarien, bei denen der weitere Verlauf eines Prozesses vom aktuellen Zustand der Prozess-Instanz abhängig ist. In dieser Kategorie sind drei Patterns gegeben. WP16 die „Deferred Choice“, WP17 das „interleaved paralleles Routing“ sowie WP18 der „Meilenstein“. Beginnend mit dem WP16, stellt dieses einen Divergenzpunkt in einem Prozess dar. An diesem Punkt kann einer von mehreren Pfaden eingeschlagen werden. Welcher Pfad gewählt wird, erfolgt durch die Umwelt, genauer gesagt dem Zustand der Prozess-Instanz. Die Entscheidung, welcher Pfad aktiviert wird, geschieht zum letztmöglichen Zeitpunkt (Wohed, 2006).

Innerhalb von BPMN ist das WP16 nativ über ein „Eventbasierendes Gateway“ abbildbar. Im Beispiel ArchiMate-Modell von Abbildung 38 wurde beim Or Verbindungspunkt das Attribut gesetzt, dass dieses Konzept als „Eventbasierendes Gateway“ instanziiert werden soll. Nach dem Gateway folgen zwei alternative Pfade, von denen nur einer aktiviert werden kann. Die Entscheidung erfolgt über die Umwelt der Prozess-Instanz, wobei entweder ein Timer von fünf Minuten abläuft oder eine Nachricht eingelangt. Im konkreten Fall dieser Prozess-Instanz sind die fünf Minuten abgelaufen. Die Aktivität „A“ wurde um 20:33 abgeschlossen, wodurch fünf Minuten später um 20:38 die gesamte Prozess-Instanz beendet wurde.

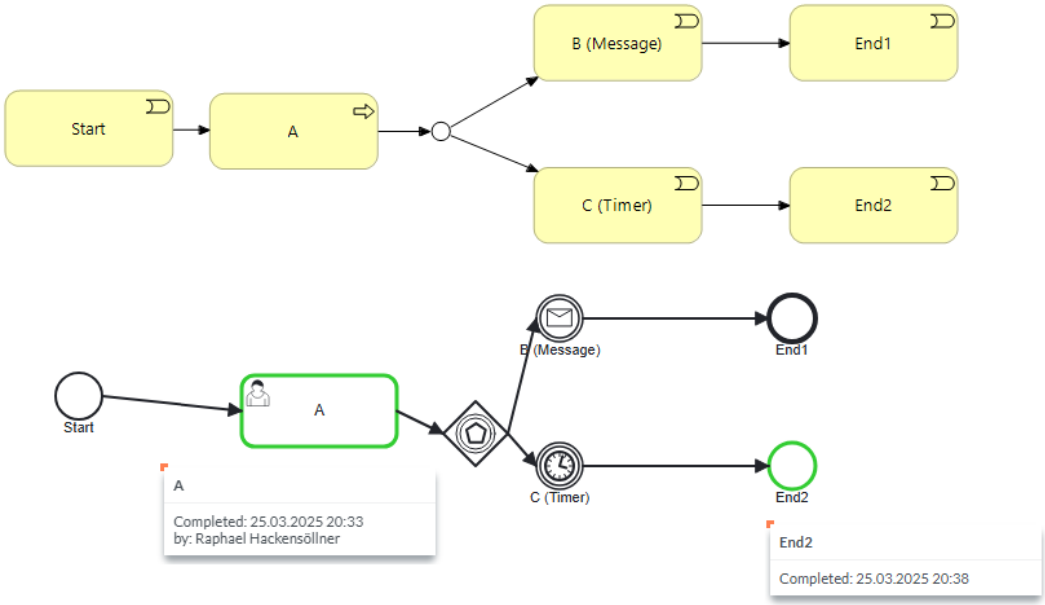

Abbildung 38: WP16 - Deferred Choice, eigene Darstellung

Das nächste Pattern in dieser Unterkategorie ist das „interleaved paralleles Routing“. Zusammenfassend bedeutet dies, dass eine Anzahl an Aktivitäten in jeder beliebigen Reihenfolge ausgeführt werden kann (Wohed, 2006). In BPMN besteht die Möglichkeit der Ad-Hoc-Ausführung von Aktivitäten. Aufgrund der Komplexität einer Ad-Hoc-Ausführung ist dieses Workflow Pattern in BPMN nicht im vollen Umfang möglich (Workflow Patterns Initiative, 2023). Daher wird das Pattern 17 für die Evaluierung nicht berücksichtigt.

Der „Meilenstein“ ist das letzte Workflow Pattern der zustandsbasierten Patterns. Ein „Meilenstein“ im Prozess bestimmt, ob eine gewisse Aufgabe aktiviert werden kann. Erst, wenn dieser Punkt erreicht ist, wird die Aufgabe freigegeben. Befindet sich der Prozess bereits in einem späteren Zustand, kann die Aufgabe nicht mehr aktiviert werden (Wohed, 2006). Um diese abstrakte Beschreibung verständlicher darzustellen, dient die folgende Abbildung 39. Das ArchiMate Modell weist hierbei ein „Boundary-Zwischenereignis“ auf. Dem Geschäftsprozess „A“ wird das Geschäftsereignis „Timer“ angehängt, was in der BPMN-Instanz ein unterbrechendes Zeit-Zwischenereignis repräsentiert.

Die Prozess-Instanz in Abbildung 39 befand sich in der Aktivität „A“. Da der Prozess zu diesem Zeitpunkt in „A“ aktiv war, konnte die Aktivität „B“ gestartet werden. Diese ist jedoch nur ausführbar, solange „A“ bislang nicht abgeschlossen ist. Ist die Aktivität „A“ bereits beendet, lässt sich „B“ nicht mehr initiieren. In diesem Fall löste der an „A“ angehängte „Timer“ aus, wodurch „A“ beendet und anschließend „B“ gestartet wurde.

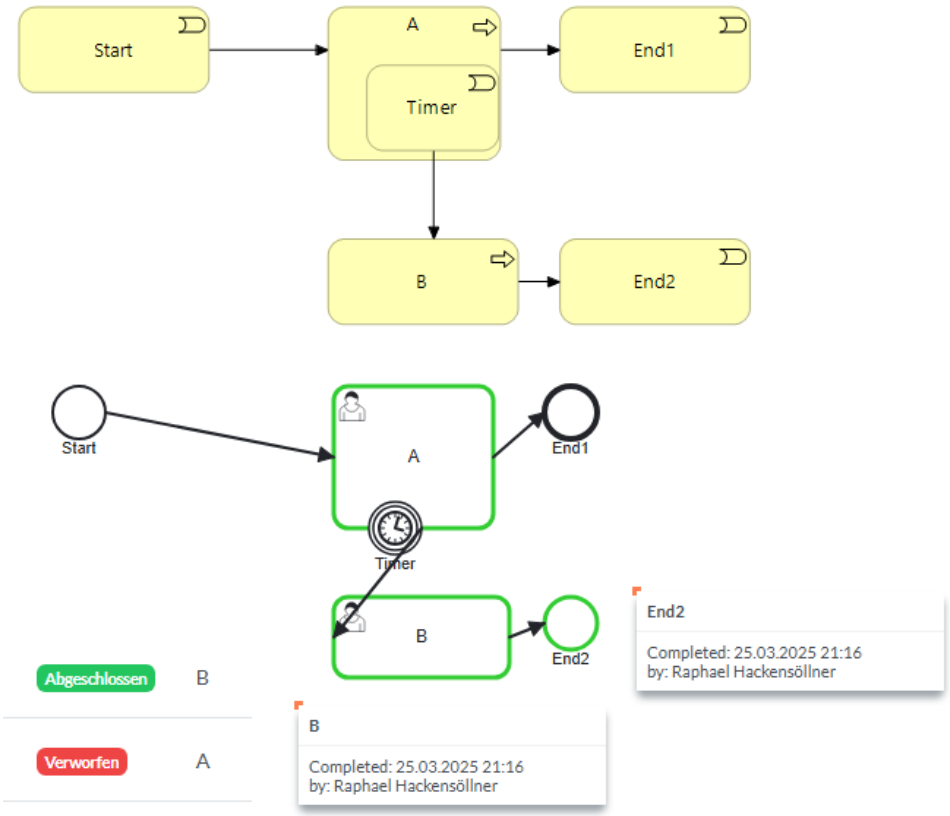

Abbildung 39: WP18 - Meilenstein, eigene Darstellung

5.2.6 Abbruchpatterns

Die letzte Unterkategorie der Workflow Patterns sind die Abbruchpatterns. Diese beschäftigen sich mit dem Abbruch von Aktivitäten oder des gesamten Prozesses. In WP19 wird definiert, dass eine Aktivität auf Basis eines gewissen Umstandes abgebrochen werden kann (Wohed, 2006). Als Beispiel für das Pattern 19, der „Abbruch einer Aktivität“ dient die Abbildung 40. Im ArchiMate-Modell wird für die Workflow Patterns erstmalig die „Applikationsfunktion“ eingesetzt, welche in eine „Service Aktivität“ transformiert wird. Die „Service Aktivität“ implementiert einen von der Cockpit365-Plattform angebotenen Automatismus, nämlich das Senden von Nachrichten innerhalb des Prozesses. Diese Nachricht kann vom angehängten „Nachrichten-Zwischenereignis“ auf der Aktivität „C“ abgefangen werden.

In der gezeigten Prozess-Instanz aus Abbildung 40 wurde „A“ absolviert und in „B“ und „C“ aufgeteilt. Nun wurde „B“ abgeschlossen, wodurch „C“ obsolet wird. Aufgrund des Umstandes, dass „B“ erledigt wurde, wird über die „Service Aktivität“ eine Nachricht ausgesendet, um „C“ abzubrechen. Als zusätzliches Detail hat das „Nachrichten-Zwischenereignis“ „Cancel C“ das Attribut zum Unterbrechen der Aktivität gesetzt. Dadurch wurde die Aktivität „C“ verworfen.

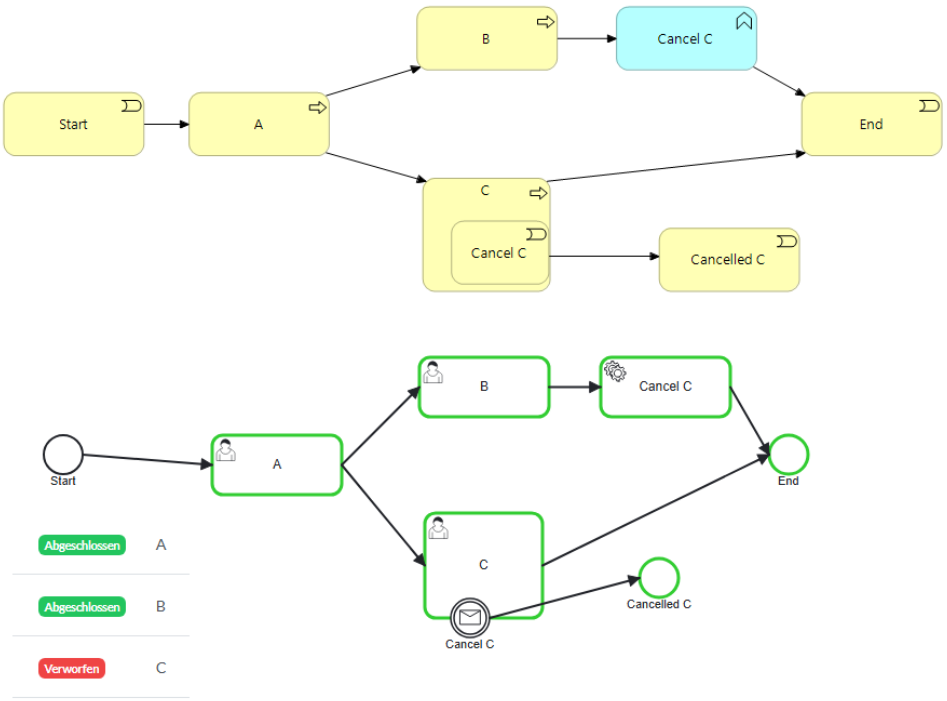

Abbildung 40: WP19 - Abbruch einer Aktivität, eigene Darstellung

Das zweite Pattern der Unterkategorie und insgesamt letzte Workflow Pattern ist der „Abbruch eines Falles“. Ein Fall wird hierbei synonym mit der Prozess-Instanz gesetzt. Demnach wird mit dem WP20 definiert, dass die gesamte Prozess-Instanz abgebrochen werden kann (Wohed, 2006). Im Sinne von BPMN bedeutet dies, dass alle noch vorhandenen Tokens auf einmal konsumiert werden. Ohne ein aktives Token gilt eine Prozess-Instanz als beendet (Göpfert & Lindenbach, 2013).

Anhand von Abbildung 41 wird ein solcher Fall demonstriert. Jede Aktivität besitzt ein „unterbrechendes Nachrichten-Zwischenereignis“. Diese führen zum Abbruch der Instanz, was hierbei ein „Terminierungs-Endereignis“ darstellt. Das „Terminierungs-Endereignis“ hat die Eigenschaft, alle Tokens, unabhängig von deren Lage im Prozess, zu konsumieren (Object Management Group, Inc, 2013). Im Gegensatz zum vorherigen Beispiel aus Abbildung 40 wird die Nachricht nicht über den Prozess selbst gesendet. Stattdessen wird die Nachricht durch einen externen Schnittstellenaufruf initiiert und über die Cockpit365-Plattform an die Prozess-Instanz weitergeleitet. Die Prozess-Instanz befand sich in Aktivität „B“, welche jedoch über einen externen Einfluss verworfen wurde und gleichzeitig die gesamte Prozess-Instanz terminiert wurde.

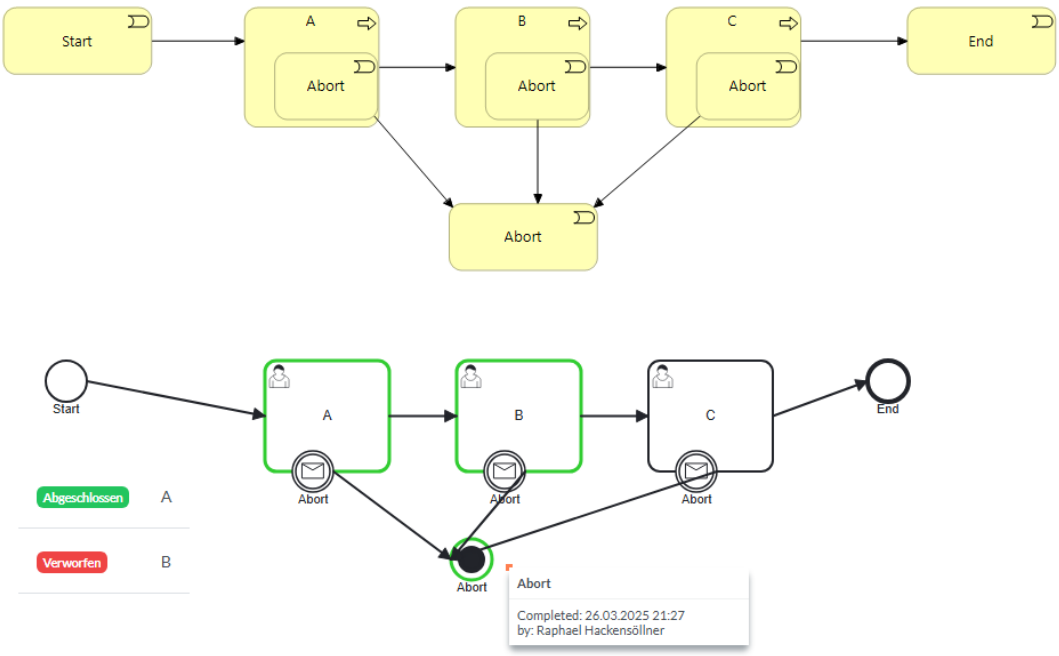

Abbildung 41: WP20 - Abbruch eines Falles, eigene Darstellung

5.2.7 Zusammenfassende Ergebnisse der Workflow Patterns

In den vorherigen Unterkapiteln wurden alle Workflow Patterns beschrieben, visualisiert, als auch über die Cockpit365-Plattform beispielhaft instanziiert. Um die Ergebnisse der vereinzelten Unterkategorien einheitlich zusammenzufassen, dient die nachfolgende Tabelle 6. In dieser sind die Workflow Patterns aufsteigend sortiert und jeweils über die letzte Spalte bewertet. Von den 20 standardisierten Patterns sind 14 über das entwickelte Artefakt abbildbar, drei nicht abbildbar, sowie drei in BPMN nicht möglich. Die Workflow Patterns WP9, WP15 und WP17 können gemäß Wohed (2006) nicht in BPMN vollständig implementiert werden, wodurch diese irrelevant sind.

Tabelle 6: Ergebnisse der evaluierten Workflow Patterns, eigene Darstellung

Workflow Pattern	Bezeichnung	Evaluierung
WP1	Sequenz	✓
WP2	Paralleler Split	✓
WP3	Synchronisation	✓
WP4	Exclusive Wahl	✓
WP5	Einfacher Merge	✓
WP6	Multi-Choice	X
WP7	Synchronisierender Merge	X
WP8	Multi-Merge	✓
WP9	Diskriminator	n.a.
WP10	Arbiträre Kreise	✓

WP11	Implizite Terminierung	✓
WP12	Mehrere Instanzen ohne Synchronisierung	X
WP13	Mehrere Instanzen mit a Priori Design Time Knowledge	✓
WP14	Mehrere Instanzen mit a Priori Runtime Knowledge	✓
WP15	Mehrere Instanzen ohne a Priori Runtime Knowledge	n.a.
WP16	Deferred Choice	✓
WP17	Interleaved paralleles Routing	n.a.
WP18	Meilenstein	✓
WP19	Abbruch einer Aktivität	✓
WP20	Abbruch eines Falles	✓

Die Patterns WP6 sowie WP7 wurden bereits im Unterkapitel 5.2.2 erläutert und begründet, warum diese über das entwickelte Artefakt nicht abbildbar sind. Erneut zusammengefasst ist das „Inklusive Gateway“ nicht im Mapping von ArchiMate zu BPMN nach Tabelle 5 enthalten. Daher besteht nicht die Möglichkeit, aus einem Gateway mit N-Pfaden, eins bis N-Pfade dynamisch zu aktivieren. Hierbei muss jedoch ein zusätzlicher Aspekt angeführt werden. Technisch hätte der Prototyp erweitert programmiert werden können, um die Transformation eines ArchiMate „And Verbindungspunktes“ in ein Parallels, als auch ein Inklusives Gateway, zu ermöglichen. Dies geschieht bereits für das „Eventbasierende Gateway“, was explizit in Kapitel 3.4 angeführt wurde. Das „Inklusive Gateway“ unterscheidet sich dahingegen, dass es nicht Teil der allgemein ausführbaren Subklasse ist (Object Management Group, Inc, 2013). Somit entfällt es aus der Mapping-Tabelle der ArchiMate zu BPMN-Konzepten.

Abgesehen von den zwei Patterns mit dem „Inklusiven Gateway“ ist das WP12, die „Instanziierung mehrerer Instanzen ohne Synchronisierung“ nicht möglich. Dazu ist anzumerken, dass ein möglicher Dissens bestehen könnte. Gemäß der Workflow Patterns Initiative (2023) ist das Pattern WP12 in BPMN abbildbar. Im BPMN Standard (2013) hingegen wird das Pattern nicht explizit als möglich angeführt. Es konnte auch das Attribut „Flow_Condition“ nicht angefunden werden, weder im Standard noch in der eingesetzten BPMN-Engine.

Mit der Aufschlüsselung der Ergebnisse aus Tabelle 6 konnte gezeigt werden, dass das entwickelte Artefakt den größten Anteil an möglichen Workflow Patterns unterstützt. Daher können teils auch komplexere Prozess Konstrukte in der Unternehmensarchitektur modelliert und nachfolgend erfolgreich instanziiert werden, mit der Einschränkung des „Inklusiven Gateways“. Da diese wiederkehrenden Lösungsstrukturen für die Prozessmodellierung möglich sind, kann die weitere Evaluierung des Artefakts in praxisnäheren Beispielen durchgeführt werden. Wären die Workflow Patterns, insbesondere die grundlegenden Kontrollflusspatterns, nicht möglich gewesen, so wäre eine weitere Evaluierung irrelevant.

5.3 Praxisbezogene Evaluierung ausgewählter ITIL-Practices

Im vorherigen Unterkapitel 5.2 wurde evaluiert, welche Workflow Patterns als theoretische Prozess Konstrukte in der Prozessmodellierung möglich sind. Da sich die Einschränkungen primär auf das „Inklusive Gateway“ beziehen, ist eine weitere Evaluierung möglich als auch notwendig, da lediglich theoretische Modelle abgebildet wurden. Um das eigentliche Problem, den Dissens zwischen den Geschäftsprozessen und den IT-Infrastrukturen, mithilfe des Prototyps zu lösen, werden praxisbezogene Beispiele benötigt. Hierbei gilt erneut der Grundsatz der wissenschaftlichen Rigorosität, sodass die zu evaluierenden Beispiele der Realität entsprechen. Aus diesem Grund ist ein in der Praxis relevantes, akzeptiertes und möglichst standardisiertes Modell erforderlich. ITIL ist ein solches Framework, welches Best Practices beschreibt und als de facto Standard im Bereich des IT-Service Managements gilt (AXELOS, 2019).

ITIL ist ein Framework zur systematischen Gestaltung, Steuerung und kontinuierlichen Verbesserung des IT-Service-Managements. In der aktuellen Version ITIL 4 verfolgt das Framework einen ganzheitlichen, auf den gesamten Servicelebenszyklus ausgerichteten Ansatz. Zur Abwicklung dieser holistischen Sichtweise werden in ITIL 4 insgesamt 34 verschiedene Practices definiert, welche sich in drei übergeordnete Kategorien aufteilen. Diese sind die General “Management Practices”, “Service Management Practices” sowie “Technical Management Practices”. Eine solche strukturierte Klassifikation ermöglicht eine ganzheitliche Betrachtung von organisatorischen, prozessualen und technischen Aspekten des IT-Betriebs (AXELOS, 2019).

Insbesondere im Bereich der „Service Management Practices“ finden sich operative, im täglichen Arbeiten durchgeführte Geschäftsprozesse wieder (AXELOS, 2019). Aus diesem Grund wird auf zwei Practices aus den „Service Management Practices“ explizit eingegangen und folglich demonstriert und evaluiert. Die beiden gewählten Practices sind das „Monitoring und Eventmanagement“ sowie das „Change-Management“. Mit der ersten Practice soll gezeigt werden, dass die IT-Infrastruktur aus der Unternehmensarchitektur mit den Geschäftsprozessen interagieren kann. Die zweite Practice, das „Change-Management“, soll demonstrieren, dass auch aufwendigere Geschäftsprozesse in der Unternehmensarchitektur abbildbar sind und diese zusätzlich direkt instanziiert werden können.

5.3.1 Monitoring und Event Management

Die ITIL 4 Practice des „Monitoring und Event Management“ verfolgt das Ziel, dass die Zustände von vorhandenen Assets kontinuierlich überwacht werden, umso auf mögliche unerwünschte Ereignisse reagieren zu können. Hierbei wird zum einen zwischen regulären Betriebsinformationen und zum anderen zwischen potenziellen Störungen oder Fehlern unterschieden. Das Monitoring selbst stellt die fortlaufende Erfassung und Aggregation von Daten dar, während das Event Management diese Informationen kontextualisiert und gemäß definierten Schwellenwerte oder Regeln verarbeitet. Somit soll eine frühzeitige Reaktion auf kritische Zustände ermöglicht werden, um die Stabilität und Verfügbarkeit der vorhandenen Services sicherzustellen (AXELOS, 2019).

Im Rahmen dieses Unterkapitels wird die praktische Umsetzung des „Monitoring und Event Management“ anhand eines prototypischen Anwendungsfalls untersucht. Der Ausgangspunkt dafür bildet eine modellierte Unternehmensarchitektur, die Infrastrukturkomponenten, eine Applikation sowie den darauf aufbauenden Geschäftsprozess des „Monitoring und Event Managements“ beinhaltet. Auf dieser Grundlage wird das Zusammenspiel der einzelnen Komponenten im Kontext der Practice analysiert und evaluiert. Zur praktischen Demonstration wurde eine entsprechende Architektur entwickelt, welche in Abbildung 42 dargestellt ist. Mit dieser soll über den entwickelten Prototyp gezeigt werden, dass der folglich instanziierte Geschäftsprozess mit den Komponenten der Unternehmensarchitektur interagieren kann.

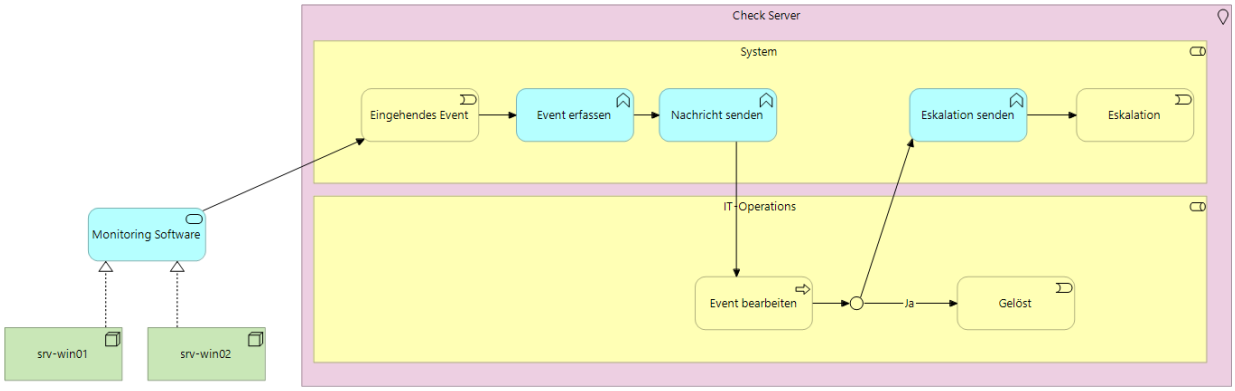

Abbildung 42: ArchiMate-Modell für Monitoring und Event Management, eigene Darstellung

Auf der linken Seite von Abbildung 42 befinden sich zwei Node-Konzepte „srv-win01“ und „srv-win02“. Diese stellen zwei Windows Server dar. Beide der Server realisieren eine gewisse Überwachungssoftware. Um mögliche Lizenzkosten zu umgehen, als auch um unnötigen Mehraufwand zu vermeiden, sind beide Server sowie die Software fiktiv. Über die Überwachungssoftware könnten die Ressourcen der Server überwacht werden und ein Event auslösen. Dieses Event startet den über das „Standort“ Konzept abgebildeten Geschäftsprozess „Check Server“. Initial wird das eingehende Event als solches erfasst, und es wird eine Nachricht an die Rolle „IT-Operations“ gesendet. Zusätzlich befindet sich die Geschäftsrolle „IT-Operations“ in der Verantwortung, das Event zu bearbeiten. Sollte das Event erfolgreich bearbeitet werden und die Ursache für dessen Auftreten gelöst sein, so wird der Prozess „Check Server“ beendet. Alternativ wird eine Eskalation in der Form einer Nachricht ausgesendet, um eine nachfolgende Eskalation einzuleiten.

Über das entwickelte Artefakt wurde das ArchiMate-Modell eingelesen und in einen BPMN-Prozess transformiert. Dieser ist in Abbildung 43 dargestellt. Hierbei ist anzumerken, dass die ArchiMate-Konzepte für die Server sowie die Überwachungssoftware nicht transformiert wurden. Diese besitzen anhand des Mappings keine semantisch zuordenbaren BPMN-Konzepte. Obwohl sie nicht zuordenbar sind und somit nicht transformiert werden können, wird kein Fehler geworfen, da dies thematisch nicht sinnvoll wäre. Als Input für den Prototyp soll eine jegliche ArchiMate-Architektur dienen können und lediglich die transformierbaren Konzepte eingelesen werden.

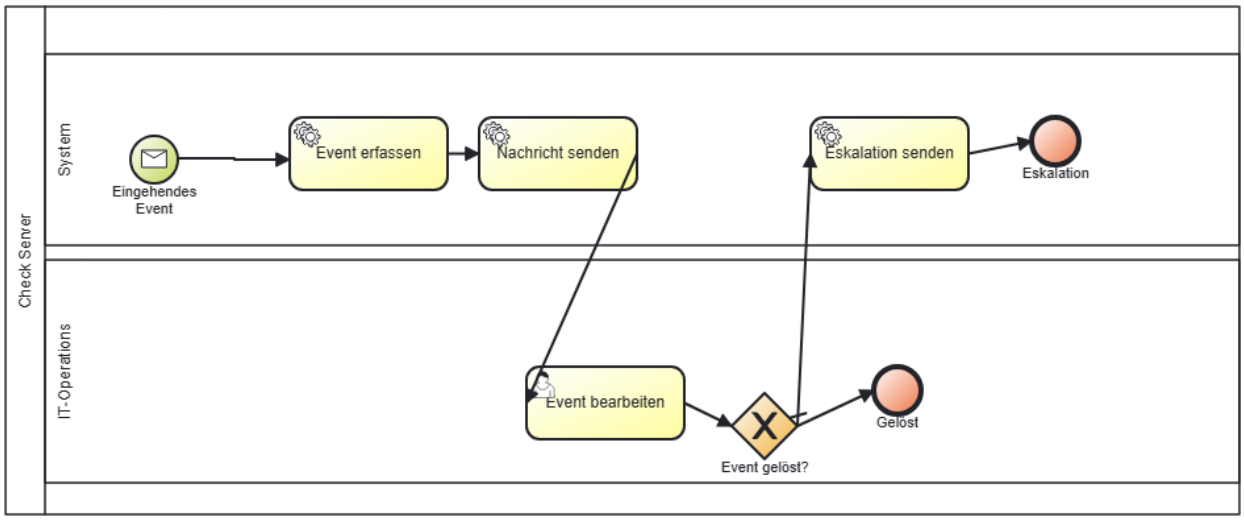

Abbildung 43: Generierter BPMN-Prozess für Monitoring und Event Management, eigene Darstellung

Im generierten BPMN-Prozess nach Abbildung 43 ergeben sich zwei „Lanes“. In der System-Lane startet der Prozess mittels eines „Nachrichten-Startereignis“ sowie den nachfolgenden Service Aktivitäten. In der Lane der Rolle „IT-Operations“ wird das jeweilige Event in der Prozess-Instanz bearbeitet und folglich für dieses eine Entscheidung getroffen. Um diesen Prozess mit einem konkreten Event instanziieren zu können, wurde die angebotene Schnittstelle der Cockpit365-Plattform aufgerufen. Dieser Aufruf erfolgte händisch und stellt somit eine Simulation einer möglichen Überwachungssoftware dar.

Über die Schnittstelle, genauer gesagt, der „Nachrichten-Definition“ des Startevents wird ein Titel, das zugehörige Asset sowie eine Beschreibung des Events übertragen und der Prozess „Check Server“ instanziiert. Dies ist anhand von Abbildung 44 ersichtlich, nämlich, dass die CPU-Last für das Asset „srv-win01“ seit über fünf Minuten über 95 Prozent ausgelastet ist. Die Werte werden über den Payload der eingehenden Nachricht initial gesetzt.

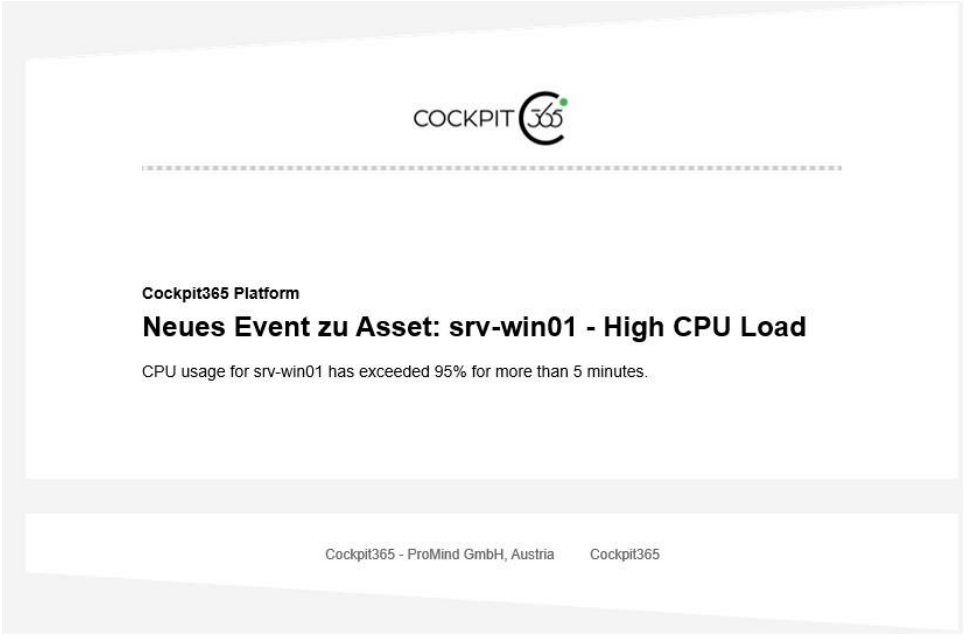

Abbildung 44: E-Mail für neues Event, eigene Darstellung

Nach der informativen Nachricht von Abbildung 44 wird eine Aufgabe aus dem eingehenden Event erstellt. Diese Aufgabe inkludiert die entsprechenden Daten aus dem Event für die Prozess-Instanz. Zusätzlich kann das Event gemäß Abbildung 45 dokumentiert und als gelöst oder nicht gelöst gesetzt werden. Im konkreten Fall der Prozess-Instanz für „srv-win01“ wurde die Ursache des Events identifiziert und behoben. Anschließend wurde die Prozess-Instanz beendet, da keine weiteren Schritte gemäß dem Prozess, beziehungsweise der Unternehmensarchitektur, gegeben sind.

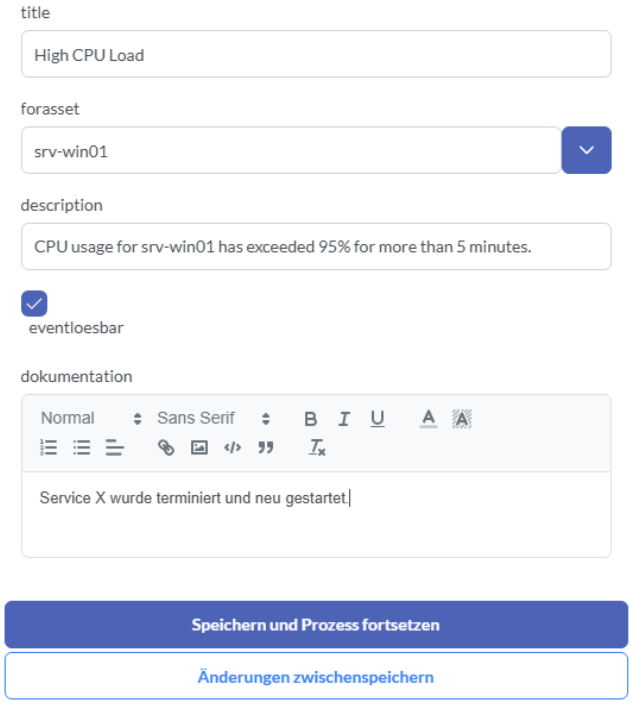

Abbildung 45: Positive Bearbeitung des Events, eigene Darstellung

Um zu simulieren, dass aus dem generierten BPMN-Prozess mehrere Prozess-Instanzen instanziiert werden können, wurde ein erneuter Schnittstellen-Aufruf abgesetzt. Folglich wurde eine neue autarke Prozess-Instanz gestartet, welche die über die Schnittstelle gelieferten Parameter verarbeitet. Diesbezüglich wurde der Windows Server „srv-win02“ nach Abbildung 42 gewählt. Zusätzlich wurde das folglich neue Event als nicht lösbar definiert, indem die Checkbox „eventloesbar“ nicht gesetzt wurde. In Abbildung 46 ist diese Checkbox sowie die Werte des neuen Events ersichtlich. Zusätzlich wurde in der Dokumentation angeführt, dass die Ursache für das konkrete Event nicht identifiziert werden konnte.

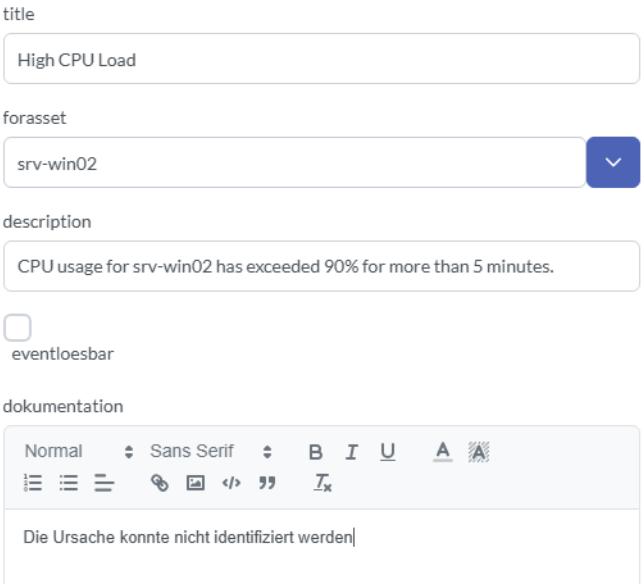

Abbildung 46: Negative Bearbeitung des Events, eigene Darstellung

Da in dieser Prozess-Instanz das Event nicht lösbar ist, wird der alternative Pfad gemäß Abbildung 43 eingeschlagen. Demnach erfolgt eine Eskalation über den Prozess selbst, indem eine E-Mail mit den Inhalten des Events ausgesendet wird. Die automatisierte E-Mail mit den Informationen über das betroffene Asset „srv-win02“ ist folglich in Abbildung 47 dargestellt. Die Prozess-Instanz wird nach dem Absenden der Nachricht beendet. Hierbei hätten noch weitere Prozessschritte folgen können, beziehungsweise einen alternativen Eskalationsprozess starten, welcher wiederum in der Unternehmensarchitektur zu modellieren ist.

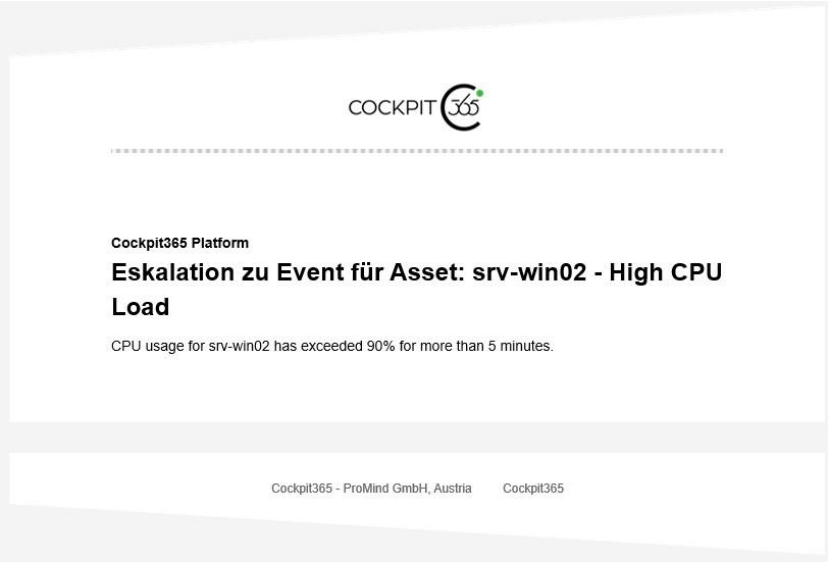

Abbildung 47: Eskalation bei nicht lösbarem Event, eigene Darstellung

Anhand dieses Beispiels aus dem Bereich „Monitoring und Eventmanagement“ wird demonstriert, dass aus einer einzigen Unternehmensarchitektur automatisiert, ein instanziierbarer Geschäftsprozess generiert werden kann. Die Instanziierung erfolgt dabei über definierte Inputparameter, deren Werte direkt auf in der Architektur modellierte Elemente referenzieren. In

dem dargestellten Fall handelt es sich bei den Inputwerten um die betroffenen Assets, genauer gesagt „srv-win01“ und „srv-win02“, welche in der zugrundeliegenden Unternehmensarchitektur nach Abbildung 42 beschrieben sind. Dadurch entsteht eine konsistente Verbindung zwischen den modellierten Assets und den entsprechenden Geschäftsprozessen. Dies ermöglicht nicht nur eine zielgerichtete Instanziierung der Prozesse auf Basis spezifischer Architekturkomponenten, sondern auch eine situationsabhängige Ausführung entlang unterschiedlicher Pfade. Das ArchiMate-Modell zur Darstellung der Unternehmensarchitektur fungiert folglich als zentrale Quelle für die Transformation und konkrete situationsbedingte Instanziierung operativer Prozesse.

5.3.2 Change-Management

Im Sinne von ITIL 4 befasst sich das „Change-Management“ mit dem Lebenszyklus von Changes. Ein Change selbst stellt hierbei eine Veränderung eines oder mehrerer Assets dar. Eine solche Veränderung kann unterschiedliche Ausprägungen aufweisen, wobei gemäß ITIL drei Kategorien von Changes definiert werden. Diese sind der „Normal-“; „Standard-“ und „Emergency-Change“. Ein „Normal-Change“ ist eine Änderung, welche nicht vorautorisiert ist und somit einer Genehmigung sowie einer Risikoanalyse unterzogen werden muss. Im Rahmen der Risikoanalyse wird bestimmt, inwiefern sich die angedachte Änderung auf die Systemstabilität auswirkt, als auch welche Maßnahmen diesbezüglich ergriffen werden müssen. Im Gegensatz dazu ist der „Standard-Change“ bereits genehmigt und bedingt nur ein geringes Risiko. In der Praxis sind dies primär Routinearbeiten, welche operativ ständig durchgeführt werden (AXELOS, 2019).

Die letzte Art des Changes ist der „Emergency-Change“, welcher unter zeitkritischen Bedingungen eingeführt werden muss. Ein typisches Beispiel wäre eine sicherheitskritische Systemlücke, die unverzüglich geschlossen werden soll. Daher wird ein verkürztes Verfahren des „Normal-Changes“ (AXELOS, 2019).

Das „Change-Management“ kann eine der aufwendigeren ITIL Practices darstellen, da diese mit diversen anderen Practices tief miteinander verknüpft sein kann, als auch implizit diverse organisatorische und technische Prozessschritte aufweist. Daher wurde die Practice gewählt, um den Prototyp mit einem weiteren praktischen Beispiel zu evaluieren. Hierbei soll demonstriert werden, wie die Instanziierung von einem aufwendigeren Unternehmensprozess, welcher in der Unternehmensarchitektur modelliert ist, erfolgen kann. Somit wurde gemäß der Abbildung 48 der „Change-Management“-Prozess modelliert. Dieser kombiniert diverse Workflow Patterns aus den vorherigen Unterkapiteln und inkludiert die Interaktion mit den Infrastrukturkomponenten aus dem Beispiel des „Monitoring und Event Managements“. Zusätzlich bildet der Prozess alle drei Change-Arten ab, welche je nach Prozess-Instanz unterschiedliche Pfade innerhalb des Prozesses durchlaufen können.

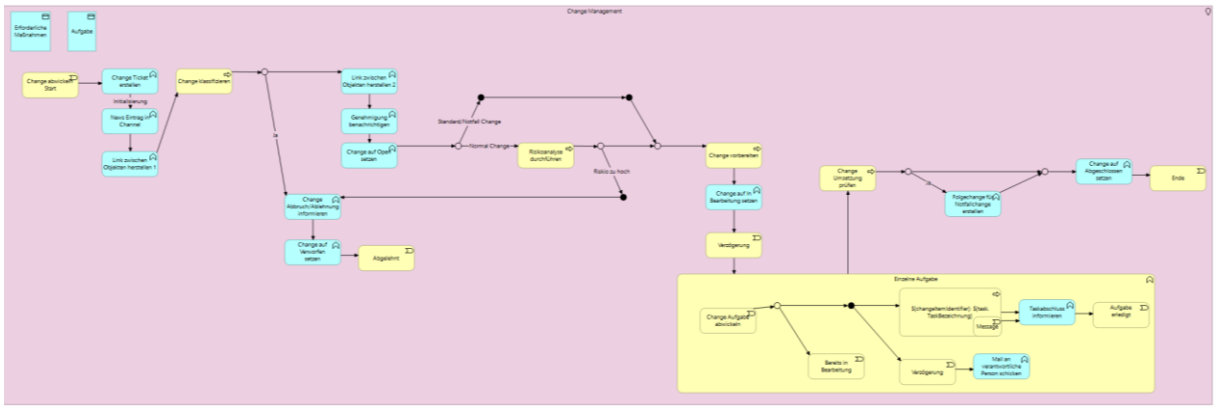

Abbildung 48: ArchiMate-Modell für Change-Management, eigene Darstellung

In der Abbildung 48 ist der gesamte „Change-Management“-Prozess in dem ArchiMate „Standort“ Konzept enthalten. Dieser inkludiert den gesamten Lebenszyklus des Changes sowie dessen unterschiedliche Abzweigungen auf Basis der Art des Changes. Zusätzlich sind diverse Automatismen über Applikationsfunktionen sowie E-Mail-Benachrichtigungen gegeben. Des Weiteren wird erstmalig das Konzept der „Geschäftsfunktion“ genutzt. Diese stellt gemäß dem Mapping nach Tabelle 5 einen Subprozess in BPMN dar. Demnach können aus einer Prozess-Instanz für einen Change weitere SubProzess-Instanzen für einzelne Aufgaben gestartet werden.

Auf der modellierten Unternehmensarchitektur aufbauend, konnte direkt über den entwickelten Prototyp das BPMN-Modell nach Abbildung 49 generiert werden. Trotz der Kombination diverser Konzepte und Workflow Patterns erfolgte die Konvertierung nativ ohne weitere Nachbereitung im BPMN-Modell selbst. Hierbei ist jedoch anzumerken, dass in der Architektur diverse „And Verbindungspunkte“ verwendet wurden, welche keinen semantischen Mehrwert bieten, wie beispielsweise nach der Frage des Change Typs oder ob das Risiko beherrschbar ist. Diese wurden lediglich gesetzt, damit die resultierenden Sequenzflüsse im BPMN-Modell optisch ansprechender sind. In der tatsächlichen Instanziierung weisen sie keine Relevanz auf.

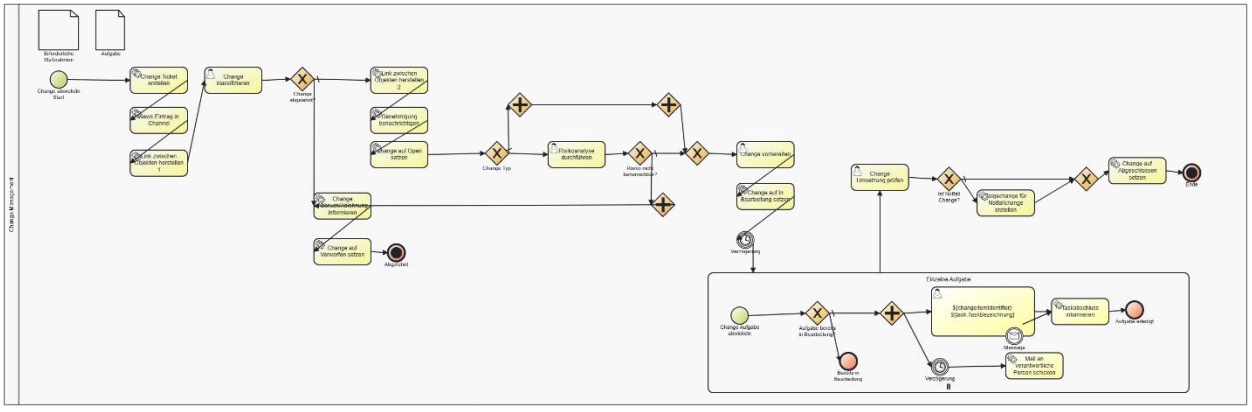

Abbildung 49: Generierter BPMN-Prozess für Change-Management, eigene Darstellung

Der nachfolgend demonstrierte Change soll zeigen, dass auch ein etwas aufwendigerer Unternehmensprozess aus der ArchiMate-Unternehmensarchitektur in der Praxis instanziiert werden kann. Diesbezüglich wurde als Beispiel ein „Normal-Change“ gewählt, da dieser die meisten Prozessschritte aus Abbildung 49 durchlaufen muss. Zusätzlich unterstützt hierbei die Cockpit365-Plattform mit diversen aus dem Prozess generierten Formularen, als auch eigene

Masken zum Anzeigen des aktuellen Prozessstatus. Ein solches Formular befindet sich bereits im beginnenden Startereignis. Dies ist in Abbildung 50 gezeigt und markiert den Start der Prozess-Instanz. Der Start des Prozesses kann durch unterschiedliche Personen erfolgen, welche beispielsweise einen Change für ein gewisses Asset einreichen wollen. In diesem Fall wird eine Erweiterung der Systemressourcen beantragt, da ein neues Programm auf dem Asset „srv-win01“ installiert wurde und mehr Arbeitsspeicher benötigt wird.

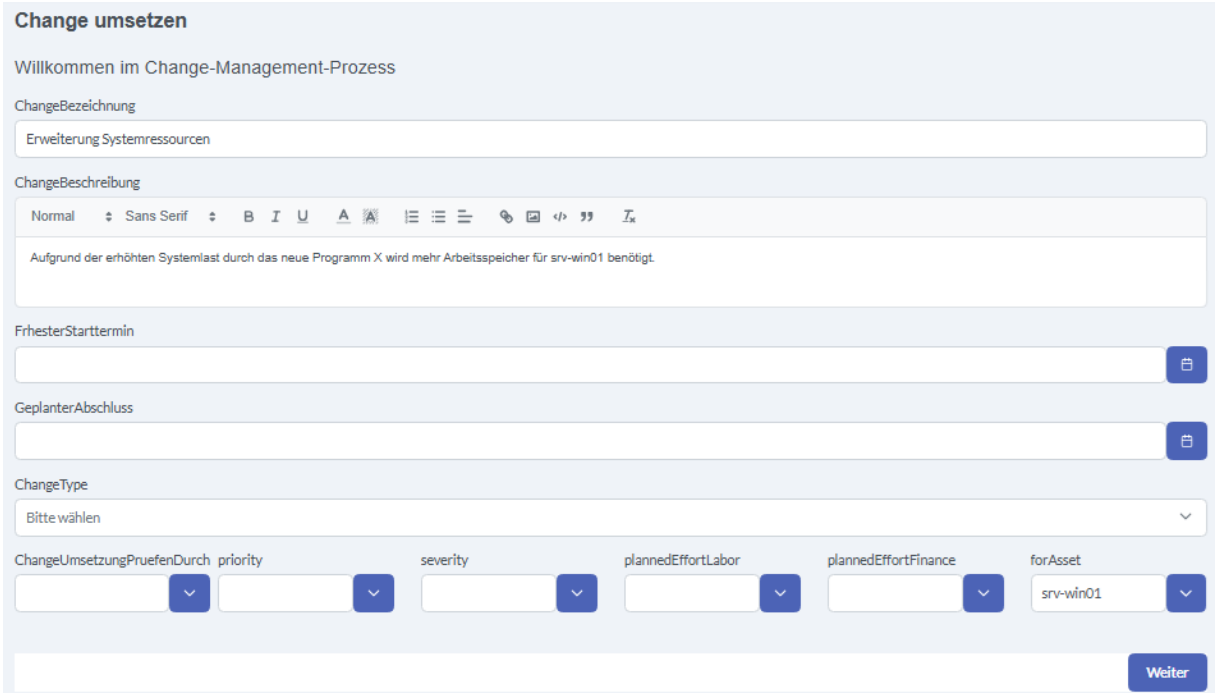

Abbildung 50: Initiales Startformular für den Change, eigene Darstellung

Nachdem der Change initial gemeldet wurde, erfolgt im nächsten Schritt dessen Klassifizierung. Diese Aufgabe obliegt dem Service Desk, welcher als zentrale Anlaufstelle für alle eingehenden Anfragen und Störungsmeldungen fungiert (AXELOS, 2019). Der Service Desk verfügt über eine dedizierte Ansicht, welche in Abbildung 51 ersichtlich ist. Innerhalb dieser Ansicht werden nicht nur alle erfassten Changes übersichtlich gelistet, sondern es findet sich auch eine integrierte Übersicht über weitere Ticketarten, wie beispielsweise der „Incidents“ wieder.

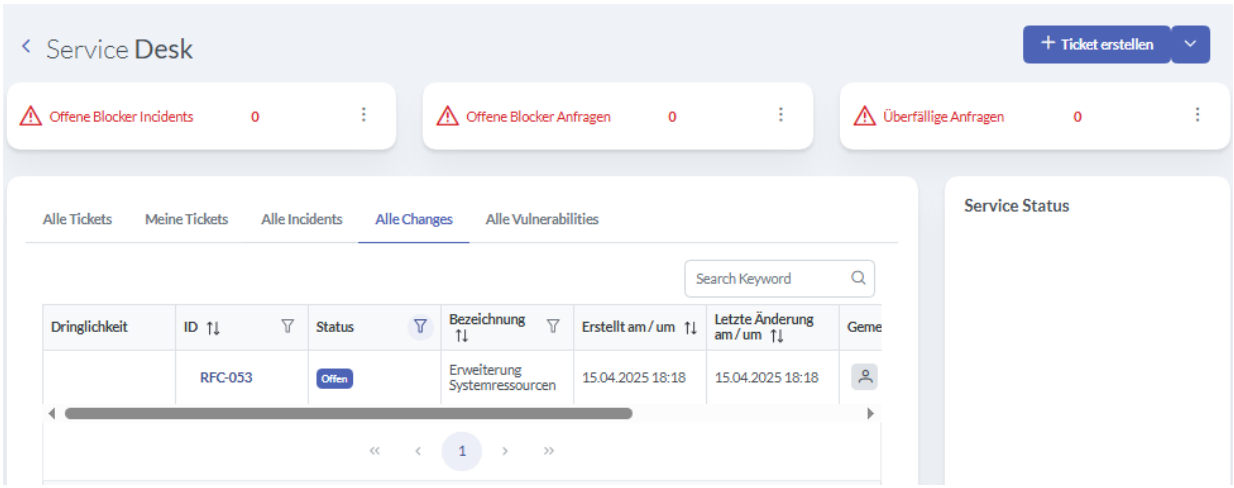

Abbildung 51: Übersicht aller Changes, eigene Darstellung

Der zuvor gemeldete Change wurde automatisiert durch die angebotenen Service Aktivität Vorlagen als solcher angelegt und auf den Status „Offen“ gestellt. Ergänzend erfolgt die automatische Vergabe einer eindeutigen, fortlaufenden Identifikationsnummer, welche sowohl der Nachverfolgbarkeit dient als auch eine strukturierte Verwaltung aller Change-Prozesse innerhalb der Plattform ermöglicht.

Über den Service Desk wird der Change aus der Ansicht von Abbildung 51 geöffnet und die weiteren Felder ausgefüllt. Hierbei ist besonders der Typ des Changes relevant. Für einen „Normal-Change“, wie in diesem Beispiel nach Abbildung 52, erfolgt ein erweiterter Prozessablauf im Gegensatz zu einem „Standard“-, oder „Emergency-Change“. Zusätzlich zum Typ des Changes werden weitere Auswahlfelder befüllt, unter anderem das betroffene Asset. In diesem finden sich die beiden Server „srv-win01“ und „srv-win02“ wieder, welche gemäß der Unternehmensarchitektur aus Abbildung 42 definiert wurden. Somit ist das Zusammenspiel von ITIL-Practices wie dem „Monitoring und Event Management“ mit „Change-Management“ gegeben.

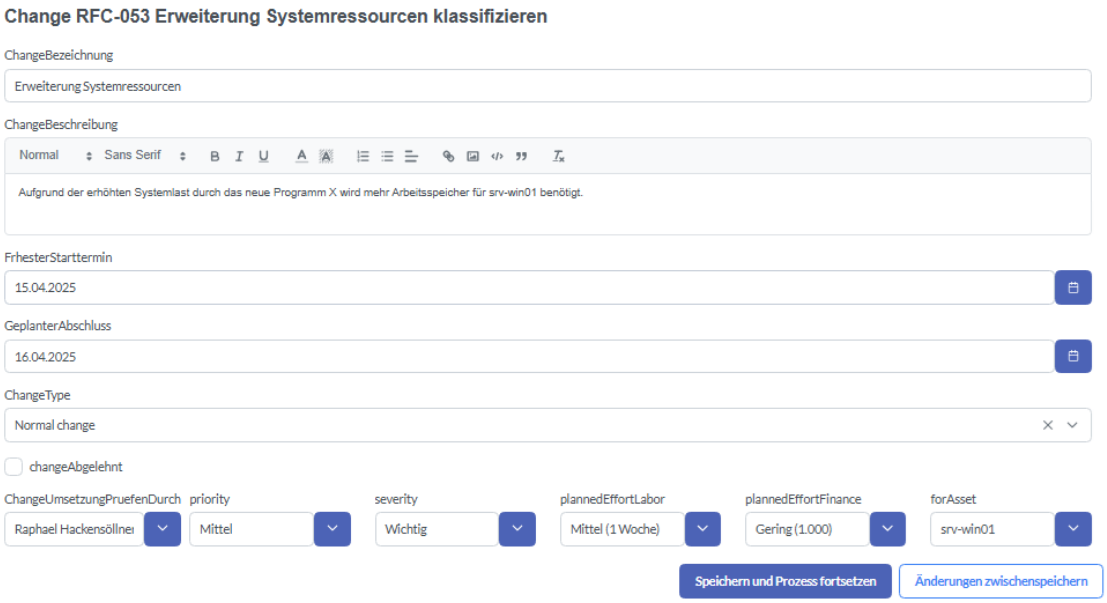

Abbildung 52: Klassifizierung des Changes, eigene Darstellung

Das Objekt des Changes sowie das Objekt des betroffenen Assets werden über eine eigene Service Aktivität nach der vollendeten Klassifizierung miteinander verbunden, um folglich eine technische Referenzierung zu ermöglichen. Alternativ hätte der Prozess in der Maske für die Klassifizierung beendet werden können, indem der Change abgelehnt wird. Bei einer Ablehnung würde der Status des Changes dementsprechend gesetzt werden, als auch die meldende Person per E-Mail informiert werden.

Da der Change als „Normal-Change“ klassifiziert und genehmigt wurde, muss eine Risikoanalyse durchgeführt werden, weil eine solche Änderung ein gewisses Risiko bedingt. Folglich muss dieser zunächst systematisch analysiert werden. Der nächste Schritt im gesamten Change-Management Prozess ist somit die Durchführung der Risikoanalyse. Die Risikoanalyse ist in Abbildung 53 dargestellt und erweitert das bereits vorhandene Webformular mit erforderlichen Maßnahmen.

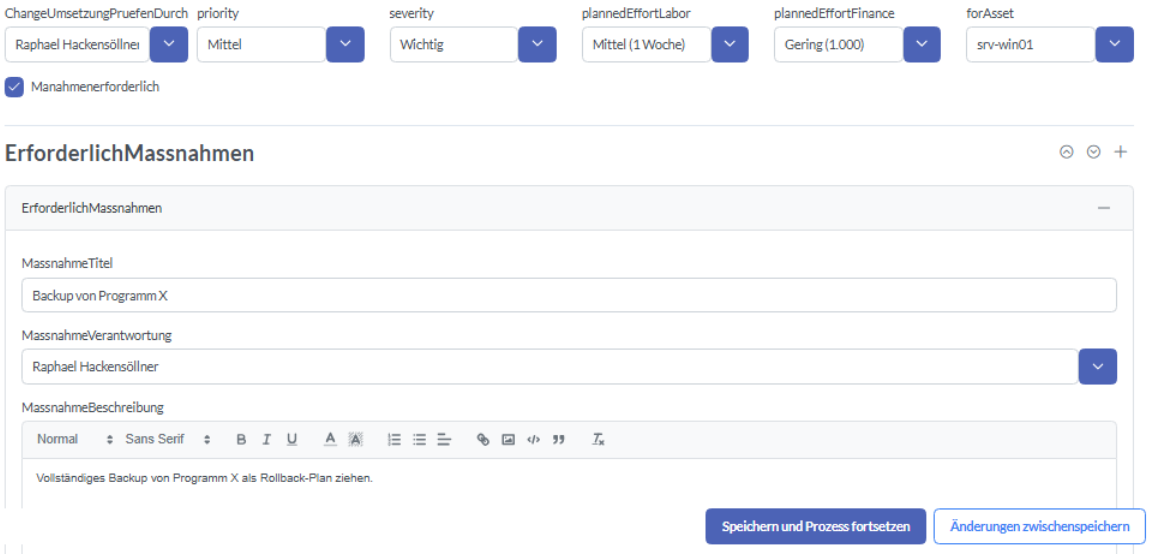

Abbildung 53: Hinzufügen von Maßnahmen, eigene Darstellung

Eine exemplarische Maßnahme nach Abbildung 53 ist das Durchführen eines vollständigen System-Backups als Bestandteil des Rollback-Plans. Zusätzlich können weitere Maßnahmen hinzugefügt werden, welche technisch als „Subform“ definiert wurden. Die tatsächlichen Inhalte, genauer gesagt dessen auszufüllenden Felder, werden über die „Datenobjekte“ gemäß der Architektur nach Abbildung 49 definiert. Diesbezüglich wurden die „Datenobjekt“-Konzepte im ArchiMate-Modell mit dem Hinzufügen von Attributen erweitert.

Der Change wurde aufgenommen, klassifiziert und dessen Risiko abgewogen. Folglich kann die tatsächliche Durchführung, beziehungsweise Abarbeitung vorbereitet werden. Dazu wurden, analog zu den erforderlichen Maßnahmen, die durchzuführenden Aufgaben mittels eines „Datenobjektes“ definiert. Für dieses Beispiel wurden zwei Aufgaben bestimmt, zum einen „System Stop und Erweiterung“ sowie zum anderen „System hochfahren und Programm testen“. Aus den beiden Aufgaben werden gemäß dem BPMN-Prozess nach Abbildung 49 Subprozesse gestartet. Folglich kann der in dieser Arbeit entwickelte Prototyp innerhalb einer Prozess-Instanz weitere Prozess-Instanzen als ausmodellierte Subprozesse instanziieren.

Der Subprozess entspricht hierbei dem ArchiMate-Konzept der „Geschäftsfunktion“, welche die darin enthaltenen Konzepte umrandet. Innerhalb der beiden Subprozesse dient eine einzelne Aufgabe jeweils als konkretes Geschäftsobjekt, welches den (Sub-)Prozess durchwandert. Diese sind in Abbildung 54 dargestellt. Hierbei wird die Bezeichnung sowie Beschreibung aus der einzelnen Aufgabe genommen und angezeigt. Auf der rechten Seite von Abbildung 54 befindet sich das geöffnete Formular einer der beiden Aufgaben, welches, wie im vorherigen Absatz erwähnt, durch ein „Datenobjekt“ definiert wird.

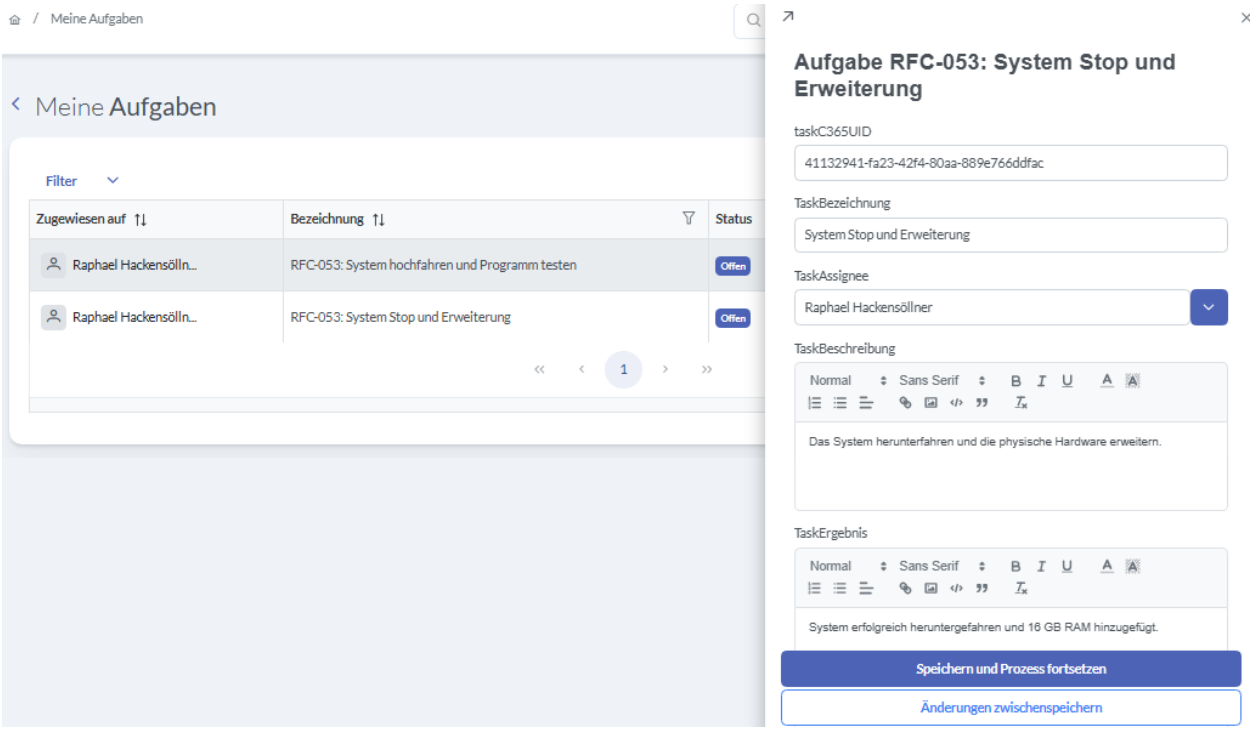

Abbildung 54: Auflistung der einzelnen Tasks für den Change, eigene Darstellung

Die beiden konkreten Subprozesse aus Abbildung 54 sind der Prozess-Instanz des gesamten „Change-Management“-Prozess zugeordnet. Die Prozess-Instanz des Hauptprozesses wartet hierbei, bis alle Prozess-Instanzen der Subprozesse abgeschlossen sind. Das bedeutet, dass sobald die beiden Aufgaben aus Abbildung 54 beendet wurden, der Hauptprozess fortgesetzt wird. Dazu erfolgt noch eine abschließende Überprüfung des gesamten Changes und dessen folglich automatisierter Abschluss. Der Change gilt somit als abgeschlossen und die Prozess-Instanz für den gesamten Change wird beendet.

Obwohl die Prozess-Instanz beendet wurde, verbleibt der Change und kann transparent anhand von Abbildung 55 nachvollzogen werden. In dieser Auflistung werden die einzelnen Aufgaben des Prozesses der Reihe nach angezeigt. Hierbei sind auch die zwei Aufgaben der instanziierten Subprozesse ersichtlich. Zusätzlich werden die aus der Prozess-Instanz gesendeten E-Mails auf der rechten Seite mit einem blauen Nachrichtenpiktogramm dargestellt.

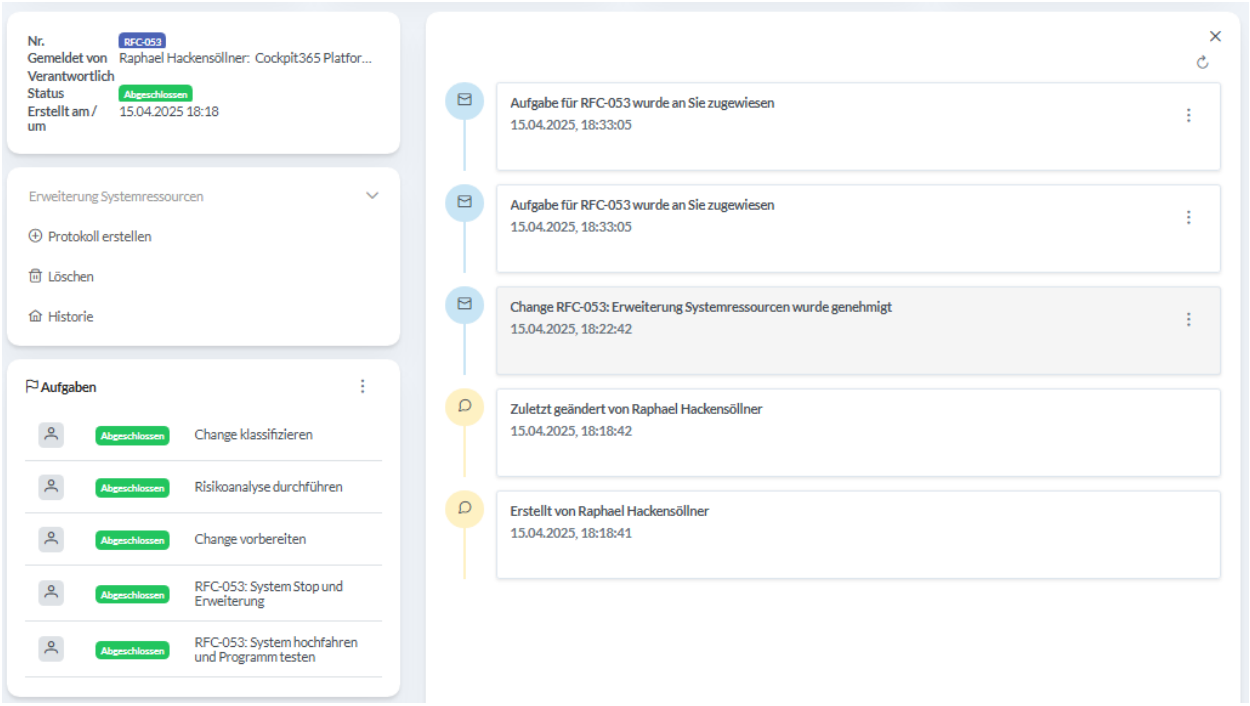

Abbildung 55: Historie und Übersicht des Changes, eigene Darstellung

Mit diesem Beispiel kann demonstriert werden, dass die Interaktion zwischen einem aufwendigeren Unternehmensprozess aus der Unternehmensarchitektur und dessen konkreten Prozess-Instanzen nahtlos funktioniert. Per Knopfdruck wurde das ArchiMate-Modell automatisiert in einen instanziierbaren BPMN-Prozess überführt, gestartet und durch das Change-Objekt selbst nachhaltig dokumentiert. Dadurch wird ersichtlich, dass sowohl die Modellierung als auch die technische Instanziierung direkt miteinander verbunden sind und reibungslos ineinandergreifen.

Zusätzlich zu diesem Beispiel des „Normal-Change“ wurden auch ein „Standard-Change“ sowie ein „Emergency-Change“ instanziiert und getestet. Beide Varianten konnten den Prozess erfolgreich durchlaufen und an den entsprechenden Stellen alternative Pfade einschlagen, etwa bei der Bewertung, Genehmigung oder beim Abbruch des Changes. Dies zeigt, dass das entwickelte Artefakt in der Lage ist, die unterschiedlichen Typen von Changes abzubilden und korrekt zu verarbeiten. Um den Umfang dieser Arbeit fokussiert zu halten und um inhaltliche Wiederholungen zu vermeiden, werden die beiden zusätzlichen Prozess-Instanzen jedoch nicht im Detail dargestellt. Ihre erfolgreiche Ausführung bestätigt dennoch die allgemeine Anwendbarkeit des modellierten „Change-Management“-Prozesses und dessen unterschiedliche Instanziierungen.

6 SCHLUSSBETRACHTUNGEN

Über die diversen vorangehenden Kapitel wurden die Phasen eins bis fünf der Design Science Research Methode nach Peffers et al. (2007) sukzessive durchlaufen. Demnach wurde das entwickelte Artefakt final evaluiert und Erkenntnisse konnten in Bezug auf die Lösung des ursprünglichen Problems gewonnen werden. Die letzte Phase, die Kommunikation der Ergebnisse, erfolgt implizit über diese Masterarbeit selbst. Zusätzlich werden die Erkenntnisse in diesem abschließenden Kapitel rekapituliert und mit ihnen die initial gestellte Forschungsfrage detailliert beantwortet. Vor der Beantwortung der Forschungsfrage erfolgt jedoch noch ein Unterkapitel mit der zusammenfassenden Darstellung der gesamten Arbeit. Im letzten Abschnitt findet eine prägnante Diskussion über die vorliegenden Ergebnisse statt, ergänzt um einen Ausblick auf mögliche aufbauende oder weiterführende Werke.

6.1 Zusammenfassung

Im Rahmen dieser Arbeit wurde die Instanziierbarkeit von BPMN-Prozessen innerhalb von ArchiMate-Architekturen untersucht und mittels eines programmierten Prototyps evaluiert. Die dazu gewählte methodische Vorgehensweise ist die Design Science Research Methode nach Peffers et al. (2007). Anhand dieser Methode und deren sechs Prozessschritte gliedert sich die Arbeit, beginnend mit der initialen Darstellung des Problems. In Kapitel 1.1 wurde das zugrunde liegende Problem angesprochen, nämlich, dass ein gewisses Silodenken zwischen den Bereichen der IT-Architektur und den Geschäftsprozessen besteht (Yildiz, 2022). Diese entwickeln sich autonom weiter, was zu einem Dissens in den strategisch definierten Zielen und der tatsächlich operativen, primär Technologie getriebenen, Umsetzung führen kann.

Zur Dokumentation und Visualisierung der beiden Bereiche der IT-Architektur, genauer gesagt der gesamten Unternehmensarchitektur und deren Geschäftsprozesse, dienen Modellierungssprachen. Hierbei kann ArchiMate für die Unternehmensarchitektur sowie BPMN für die Geschäftsprozesse verwendet werden. Folglich entstehen autonome Modelle, welche gegebenenfalls nicht mit dem jeweils anderen Bereich abgestimmt sind. Um diesen konkreten Dissens in den Modellen zu minimieren, wurde die bereits zuvor angesprochene direkte Instanziierbarkeit von BPMN-Prozessen aus der Unternehmensarchitektur untersucht. Somit sollen die Modelle der Unternehmensarchitektur als „Single-Source-of-Truth“ dienen und daraus direkt die Geschäftsprozesse generiert werden. Basierend auf dieser Zielsetzung wurde die detaillierte Forschungsfrage aus Kapitel 1.2 formuliert, welche gemäß der DSRM anhand eines Artefakts beantwortet wird.

Bevor jedoch das Artefakt entworfen und entwickelt wurde, mussten die zugrunde liegenden Begrifflichkeiten sowie Konzepte erläutert werden. Dazu dient das Kapitel 2, in welchem Einblicke in den Bereich der Modellierungssprachen gegeben wurden. Initial wurden die Bestandteile und Eigenschaften von Modellierungssprachen im Allgemeinen erläutert. Zusätzlich zu den stets

gegebenen Bestandteilen wie der Syntax und Semantik wurde explizit der Aspekt der Instanziierbarkeit nach der Definition von Gamma (1995) angeführt. Des Weiteren wurden die Modellierungssprachen ArchiMate und BPMN jeweils detailliert beschrieben.

Nach der Erläuterung der Begrifflichkeiten erfolgte das Design und die Entwicklung des Artefakts. Dieses gliedert sich in zwei Teilbereiche, zum einen das Mapping zwischen ArchiMate und BPMN und zum anderen die Programmierung eines Prototyps zur Generierung von instanziierbaren BPMN-Prozessen. Für das Mapping der ArchiMate-Konzepte zu BPMN-Konzepten erfolgte eine Literaturrecherche, bei welcher fünf Quellen mit bereits vorhandenen Mappings identifiziert wurden. Innerhalb der Literaturquellen wurden unterschiedliche Ansätze zur Erarbeitung des jeweiligen Mappings verfolgt. Die einzelnen Mappings wurden aggregiert und in Bezug auf die Häufigkeit der einzelnen Konzeptpaare numerisch dargestellt. Jene daraus resultierende Liste wurde mit dem Aspekt der Instanziierbarkeit erweitert und gefiltert. Die Filterung, welche Konzepte instanziierbar sind, erfolgte gemäß der allgemein ausführbaren Subklasse aus dem BPMN Standard (2013). Nachfolgend wurden die übrigbleibenden Konzeptpaare gemäß der eruierten Häufigkeit weiter gefiltert, woraus die finale Mapping-Tabelle nach Tabelle 5 entstand.

Anhand der Mapping-Tabelle zwischen den möglichen ArchiMate-Konzepten und den dazugehörigen, instanziierbaren BPMN-Konzepten wurde der zweite Teil des angestrebten Artefakts entworfen. Dazu wurden in Kapitel 4.1 Anforderungen und Designkriterien abgeleitet und definiert. Die Designkriterien inkludieren, dass ArchiMate-Modelle einlesbar sind und in instanziierbare BPMN-Prozesse transformiert werden können. Gemäß diesen Kriterien und der Mapping-Tabelle wurde ein Java-Programm entwickelt. Die durch das Programm generierten BPMN-Prozesse können direkt in eine BPMN-Engine geladen und als konkrete Prozess-Instanz gestartet werden. Für die BPMN-Engine wurde auf die Cockpit365-Plattform zurückgegriffen.

Das finalisierte Artefakt musste anschließend demonstriert und evaluiert werden, dass es zum einen technisch funktioniert als auch für die Lösung des ursprünglichen Problems nützlich ist. Diesbezüglich diente das Kapitel 5, indem gezeigt wurde, dass ArchiMate-Modelle technisch eingelesen, transformiert und als BPMN-Prozess instanziiert werden können. Die technische Funktionalität ist folglich gegeben. Zusätzlich wurde das Artefakt anhand der standardisierten Workflow Patterns nach Van Der Aalst et al. (2003) systematisch evaluiert. Da diese erste Evaluierung zum größten Teil erfolgreich war, konnte die weitere Evaluierung anhand von praxisbezogenen Beispielen nachgesetzt werden.

Als praxisrelevante Beispiele wurden zwei Practices aus dem ITIL-Framework gewählt, nämlich das „Monitoring und Event Management“ sowie das „Change-Management“. Beide Practices konnten erfolgreich in der Unternehmensarchitektur modelliert und als Prozess-Instanzen gestartet werden. Demnach ist das Artefakt als Lösung für das ursprüngliche Problem des Dissens zwischen der Unternehmensarchitektur sowie den Geschäftsprozessen geeignet. Damit ist die Demonstration sowie die Evaluierung des entwickelten Artefakts vollumfänglich abgeschlossen. Im folgenden Unterkapitel erfolgt eine strukturierte Erörterung der aus den diversen Evaluierungen gewonnenen Erkenntnisse, um die einzelnen Teilaspekte der formulierten Forschungsfrage abschließend zu beantworten.

6.2 Beantwortung der Forschungsfrage

Das zentrale Ergebnis dieser Arbeit besteht in der Beantwortung der einleitend gestellten Forschungsfrage aus dem Kapitel 1.2. Die Forschungsfrage selbst stellt eine Aneinanderreihung von sequenziell aufbauenden Unterfragen dar. Daher wird die insgesamt gestellte Frage in den nachfolgenden Unterkapiteln aufgeteilt und jeder Abschnitt detailliert beantwortet. Die Beantwortung der Forschungsfrage beginnt mit dem syntaktischen sowie semantischen Vergleich von ArchiMate und BPMN, was primär die durchgeführte Literaturrecherche darstellt. Anschließend wird das entwickelte Mapping zwischen ArchiMate und BPMN evaluiert, was den ersten Teil des erarbeiteten Artefakts darstellt. Den zweiten Teil des Artefakts repräsentiert die weitere Unterfrage aus der gesamten Forschungsfrage, nämlich die Instanziierbarkeit von BPMN-Prozessen innerhalb von ArchiMate-Architekturen. Der abschließend letzte Teil der Forschungsfrage bezieht sich auf die Möglichkeiten als auch, welche Grenzen mit dem entwickelten Artefakt einhergehen.

6.2.1 Syntaktischer sowie semantischer Vergleich von ArchiMate und BPMN

Die Beantwortung der ersten Unterfrage, wie ein syntaktischer sowie semantischer Vergleich zwischen ArchiMate und BPMN erfolgt, erwies sich als nicht trivial. Um zu einem Ergebnis zu gelangen, wurde eine systematische Literaturanalyse durchgeführt, bei welcher fünf einzelne Literaturquellen identifiziert wurden. Diese fünf zentralen Werke wurden analysiert, um die unterschiedlichen Herangehensweisen zum Vergleich der beiden Modellierungssprachen ArchiMate und BPMN zu verfolgen. Aus der Analyse konnten demnach die folgenden Ansätze identifiziert werden.

- Ein ontologischer Ansatz nach Penicina (2013):

  Für den ontologischen Ansatz wird die BWW-Ontologie als theoretisches Rahmenwerk genutzt, um die Vollständigkeit und Rechtmäßigkeit von Geschäftsprozessmodellen zu analysieren. Folglich wird die Ontologie als drittes Artefakt und als Bindeglied zwischen ArchiMate und BPMN hinzugefügt, umso ein Mapping auf Metamodell-Ebene zu ermöglichen.

- Ein mathematischer Ansatz nach Qumer Gill und Atif Quresh (2015):

  Beim mathematischen Ansatz wurde ein syntaktischer, semantischer und struktureller Vergleich der Metamodelle von ArchiMate und BPMN durchgeführt, auf dem aufbauend einzelne Kennzahlen erarbeitet wurden. Über diese Kennzahlen wurden die Ähnlichkeiten der Konzepte zwischen den beiden Artefakten quantifiziert.

- Die Integration verschiedener Standards nach Lankhorst et al. (2017):

  Hierbei wird ArchiMate als übergeordnete Modellierungssprache betrachtet, welche verschiedene Standards und Frameworks integriert, unter anderem auch BPMN. Es werden unterschiedliche Mappings zwischen den einzelnen Artefakten und ArchiMate angeführt. Die genaue Methodik zur Erstellung der Mappings wird jedoch nicht detailliert beschrieben.

- Die Nutzung der Atlas Transformation Language nach Marugán Cancio (2018):

  Es werden die semantischen Ähnlichkeiten der Metamodelle von ArchiMate und BPMN miteinander verglichen und darauf aufbauend Transformationsregeln entwickelt. Für die Transformationsregeln wird auf die Atlas Transformation Language gesetzt, um die ArchiMate-Konzepte in BPMN-Konzepte zu generieren. Die Regeln selbst liegen als Pseudocode vor und sollen als Grundlage für die mögliche Entwicklung von Applikationen dienen.

- Eine narrative und technische Implementierung nach Wierda (2021):

  In diesem Ansatz wird keine direkte Mapping-Tabelle angeführt. Der Ansatz fokussiert sich auf eine mögliche technische Implementierung und fügt der BPMN-Seite sogenannte „Other Domain“ Objekte hinzu. Diese Objekte werden den BPMN-Elementen zugeordnet und ermöglichen eine Äquivalenz zwischen den beiden Modellierungssprachen.

Anhand dieser verschiedenen Ansätze kann zusammenfassend zur Beantwortung der Frage eruiert werden, dass unterschiedliche Methoden für einen syntaktischen und semantischen Vergleich gegeben sind. Diese basieren primär auf den dahinterliegenden Metamodellen der beiden Modellierungssprachen. Eine konkrete Antwort, wie ein syntaktischer und semantischer Vergleich von ArchiMate und BPMN erfolgt, wäre folglich, die dahinterliegenden Metamodelle zu nutzen und eine Kombination der dargestellten Ansätze durchzuführen. Daher wurde eine solche Kombination der einzelnen Ansätze durchgeführt, um die Basis für die nächste Unterfrage in der gesamten Forschungsfrage zu schaffen.

6.2.2 Mapping zwischen ArchiMate und BPMN

Im vorherigen Unterkapitel wurde beantwortet, wie ein syntaktischer sowie semantischer Vergleich zwischen den Konzepten von ArchiMate und BPMN erfolgt. Auf diesem Wissen aufbauend, konnte im Rahmen dieser Arbeit ein Mapping zwischen ArchiMate und BPMN erfolgreich erarbeitet werden. Die sequenziell durchgeführten Schritte bis zum finalen Mapping nach Tabelle 5 sind abstrakt in der nachfolgenden Abbildung 56 dargestellt.

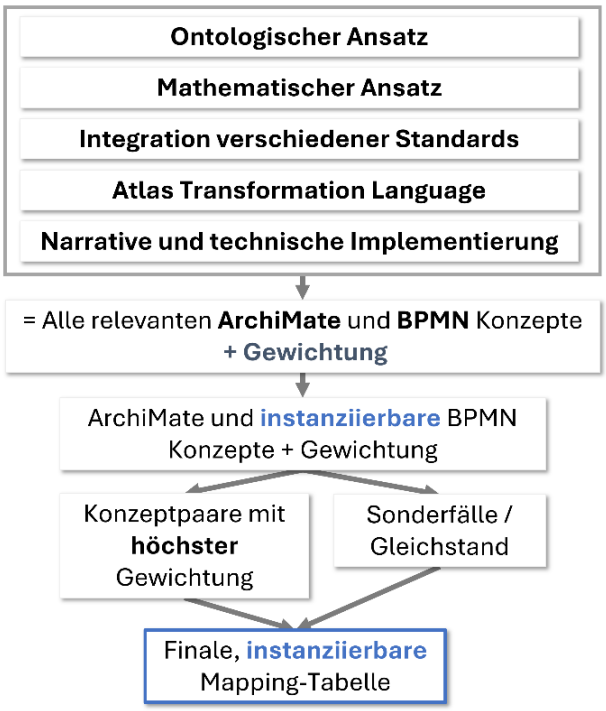

Abbildung 56: Prozess zum Mapping von ArchiMate und BPMN, eigene Darstellung

Beginnend mit dem obersten Bereich der Abbildung 56, werden die unterschiedlichen Ansätze aufgezählt. Diese wurden, wie bereits erwähnt, analysiert und zusammengefasst. Aus der Zusammenfassung konnten alle relevanten ArchiMate- und BPMN-Konzepte gebündelt extrahiert und anhand einer Gewichtung gemäß der Tabelle 2 aggregiert werden. Somit wurden von allen möglichen ArchiMate- und BPMN-Konzepten lediglich die für den Zweck dieser Arbeit relevanten identifiziert, wodurch bereits eine erste Filterung erfolgte. Als nächster Schritt wurden nur die instanziierbaren BPMN-Konzepte gemäß der allgemein ausführbaren Subklasse betrachtet, da nicht instanziierbare Konzepte in diesem Fall als irrelevant gelten. Es wurden jedoch zwei Ausnahmen zu dieser Regel identifiziert, nämlich die BPMN-Konzepte „Pool“ und „Lane“, da durch diese die zugehörigen Verantwortlichkeiten repräsentiert werden können.

Aus den nun erneut gefilterten Konzepten wurden die einzelnen Konzeptpaare gebündelt und das Paar mit der höchsten Gewichtung selektiert. Bei Sonderfällen, beziehungsweise einem Gleichstand, wurde eine detailliertere Analyse durchgeführt. Dazu wurde der ArchiMate Standard (2022a) sowie der BPMN Standard (2013) rigoros miteinander verglichen, um folglich das geeignetste Konzeptpaar zu wählen. Eine solche tiefgreifendere Analyse musste insgesamt viermal für die Konzepte „Datenobjekt(e)“, „Eventbasierendes Gateway“, „Nachricht“ sowie „Service Aktivität“ durchgeführt werden. Nach dem Abschluss dieser Analysen konnte die Erarbeitung der Mapping-Tabelle abgeschlossen werden, woraus das finale Mapping nach Tabelle 5 entstand.

Konkret wurden im Mapping elf ArchiMate-Konzepte identifiziert, welche zu zwölf allgemeinen BPMN-Konzepten zugeordnet sind. Die Differenz des einen zusätzlichen BPMN-Konzepts entsteht durch die Aufteilung des „Or Verbindungspunkt“ in das „Exklusive Gateway“ sowie das „Eventbasierendes Gateway“. Diese resultierende Anomalie des sogenannten „Symbol Overload“ wurde bewusst akzeptiert. Dieser Umstand zeigt jedoch explizit, dass ein direktes eins

zu eins Mapping der beiden Modellierungssprachen nicht möglich ist. Dieser Aspekt wird weiter untermauert, dass sich die elf ArchiMate-Konzepte in insgesamt 25 instanziierbare BPMN (Sub-)Klassen aufteilen. Demnach kann beispielsweise ein modelliertes ArchiMate „Geschäftsereignis“ im BPMN-Modell als neun verschiedene Subklassen des „Ereignisses“ instanziiert werden.

Zur Beantwortung der (Teil-)Forschungsfrage des Mappings zwischen ArchiMate und BPMN konnte folglich gezeigt werden, dass ein solches Mapping trotz der Anforderung der Instanziierbarkeit möglich ist. Die identifizierten ArchiMate-Konzepte sind syntaktisch und semantisch den erarbeiteten, instanziierbaren BPMN-Äquivalenten zuordenbar. Zusätzlich konnte festgestellt werden, dass das direkte eins zu eins Mapping nicht möglich ist und ein ArchiMate-Konzept mehreren BPMN (Sub-)Klassen zugeordnet werden kann.

6.2.3 Instanziierung von BPMN-Prozessen innerhalb von ArchiMate-Architekturen

Die nächste Unterfrage in der formulierten Forschungsfrage beschäftigt sich mit der Thematik der tatsächlichen Instanziierbarkeit von BPMN-Prozessen innerhalb von ArchiMate-Architekturen. Dieser Aspekt wurde im Rahmen der Arbeit als zweiter Teil des entwickelten Artefakts betrachtet. Dazu ist anzumerken, dass dieser Teil direkt auf dem ersten, nämlich dem finalen Mapping, aufbaut. Wäre ein Mapping nicht möglich gewesen und hätte somit die im vorherigen Unterkapitel beschriebene Unterfrage nicht beantwortet werden können, so wäre dieser Part nicht realisierbar gewesen. Da jedoch ein Mapping erarbeitet werden konnte, war die weitere Entwicklung sinnvoll.

Um die Instanziierbarkeit testen zu können, um so die gestellte (Teil-)Forschungsfrage zu beantworten, wurde wie bereits im Laufe der Arbeit öfter angeführt ein Java-Programm entwickelt. Das Programm ermöglicht das technische Einlesen von ArchiMate-Modellen, deren Transformation gemäß der Mapping-Tabelle, als auch deren abschließende Serialisierung als BPMN-Prozess. Da bereits in der Mapping-Tabelle lediglich instanziierbare Konzepte, mit der Ausnahme des „Pools“ und „Lane“, enthalten sind, gelten die resultierenden BPMN-Prozesse als instanziierbar. Dies musste jedoch demonstriert werden, weshalb auf die Cockpit365-Plattform mit der zugrundeliegenden Camunda-Engine gesetzt wurde.

Mittels eines minimalen ArchiMate-Modells nach Abbildung 22 konnte gezeigt werden, dass eine Transformation in ein BPMN-Modell sowie dessen direkte Instanziierung als Prozess-Instanz möglich ist. Es können per Knopfdruck Geschäftsprozesse aus einer Unternehmensarchitektur extrahiert und für konkrete Anwendungsfälle instanziiert werden. Zusätzlich kann über das entwickelte Programm gewährleistet werden, dass lediglich syntaktisch korrekte BPMN-Modelle erzeugt werden. Dies bietet bereits zum Zeitpunkt der Modellierung der Unternehmensarchitektur den Mehrwert, dass validiert werden kann, ob jene Unternehmensarchitektur für eine mögliche Instanziierung syntaktisch korrekt ist.

Um die Forschungsfrage weiters detaillierter zu beantworten, als auch gemäß der gewählten Forschungsmethode aufzuzeigen, dass das Artefakt das zugrunde liegende Problem des Dissens zwischen Unternehmensarchitekturen sowie den Geschäftsprozessen lösen kann, reicht ein

einfaches Beispiel nicht aus. Daher wurden in Kapitel 5.2 standardisierte Workflow Patterns sowie in Kapitel 5.3 praxisrelevante Practices gemäß ITIL evaluiert. Von den 17 in BPMN möglichen Patterns konnten 14 erfolgreich modelliert und anschließend instanziiert werden. Bezüglich der beiden ITIL-Practices konnte über das „Monitoring und Event Management“ gezeigt werden, dass die Interaktion zwischen den Komponenten der Unternehmensarchitektur und den daraus resultierenden Prozess-Instanzen möglich ist. Anhand des „Change-Managements“ konnte demonstriert werden, dass auch aufwendigere Unternehmensarchitekturen direkt instanziierbar sind, als auch Prozess-Instanzen mit unterschiedlichen Pfaden ermöglichen.

Die über die Workflow Patterns sowie den ITIL-Practices festgestellten Detailerkenntnisse werden im nachfolgenden Unterkapitel in Bezug auf die gesamten Möglichkeiten und Grenzen des Artefakts angeführt. Im Rahmen dieses Unterkapitels, genauer gesagt der (Teil-)Forschungsfrage, gilt die Instanziierung von BPMN-Prozessen innerhalb von ArchiMate-Architekturen als ausreichend bewiesen. Insgesamt wurden während des Schreibens dieser Arbeit ungefähr zwanzig verschiedene ArchiMate-Modelle modelliert, in BPMN-Modelle transformiert und mehrfach erfolgreich instanziiert, was die erfolgreiche Beantwortung untermauert.

6.2.4 Möglichkeiten und Grenzen des Artefaktes

Um die gesamte Forschungsfrage abschließend zu beantworten, wird folglich der letzte Teilaspekt, die Möglichkeiten und Grenzen des Artefaktes, angeführt. Dazu dienen die bereits im vorherigen Unterkapitel angesprochenen Workflow Patterns und gewählten ITIL-Practices. Mit diesen konnte initial eine primär theoretische, genauer gesagt konzeptionelle, Evaluierung durchgeführt werden, sowie anschließend eine praxisbezogene Untersuchung anhand von konkreten Beispielen.

Beginnend mit den Workflow Patterns konnte evaluiert werden, dass das entwickelte Artefakt die abstrakten Prozess Konstrukte mit lediglich zwei Ausnahmen unterstützt. Somit können Unternehmensarchitekturen modelliert werden, welche initial alle grundlegenden Kontrollflusspatterns inkludieren und als konkrete Geschäftsprozessmodelle instanziiert werden. Demnach bestehen keine Grenzen für die Abbildung von Modellen, welche lediglich sequenzielle Sequenzflüsse, Parallelisierung sowie explizite Entscheidungen aufweisen.

Im Bereich der nachfolgenden erweiterten Verzweigungs- und Synchronisierungspatterns sind jedoch Grenzen für das entwickelte Artefakt gegeben. Das BPMN-Konzept des „Inklusiven Gateways“ ist kein Bestandteil des ersten Teils des Artefakts, der Mapping-Tabelle, und kann dadurch nicht für den zweiten Teil herangezogen werden. Dies bedeutet, dass das in dieser Arbeit entwickelte Artefakt insgesamt keine Modelle zulässt, welche Konstrukte enthalten, bei welchen ein bis N verschiedene Pfade dynamisch aktiviert werden müssten. Bei der Modellierung und Instanziierung der weiteren Patterns sowie der ITIL-Practices erwies sich diese Einschränkung jedoch als unproblematisch.

Die strukturellen Patterns sind vollständig abbildbar, wodurch Schleifen sowie die implizite Terminierung alternativer Pfade instanziiert werden können. Für Modelle, welche Aktivitäten mehrfach ausführen, genauer gesagt instanziieren, besteht die Einschränkung des WP12 „Mehrere Instanzen ohne Synchronisierung“. Dies beruht darauf, dass das technische Attribut „Flow_Condition“ nicht angefunden werden konnte. Konzeptionell bedeutet dies für das gesamte Artefakt, dass Aktivitäten (dynamisch) mehrfach instanziiert werden können, diese jedoch stets synchronisiert werden. Auch diese Einschränkung führte zu keinen Problemen bei den nachfolgenden Patterns sowie den ITIL-Practices, im Gegenteil, es stellte das gewünschte Verhalten dar. Im Prozess des „Change-Managements“ soll beispielsweise der Prozess erst fortgesetzt werden, wenn alle Aufgaben abgeschlossen (synchronisiert) sind. Die verbleibenden zustandsbasierten Patterns, bei denen der weitere Verlauf vom Zustand der Prozess-Instanz abhängt, sowie die Abbruchpatterns, um einzelne Aktivitäten oder den gesamten Prozess abzubrechen, sind vollumfänglich möglich.

Im vorherigen Unterkapitel wurde die Instanziierung von BPMN-Prozessen innerhalb von ArchiMate-Architekturen bereits erfolgreich beantwortet und über die Workflow Patterns die zwei konzeptionellen Grenzen festgestellt. Um die Möglichkeiten und Grenzen für praxisrelevante Modelle zu eruieren, dienten die beiden ITIL-Practices des „Monitoring und Event Management“ sowie des „Change-Management“. Für beide Anwendungsfälle konnten detaillierte und teils aufwendige Unternehmensarchitekturen modelliert werden und diese direkt als Prozess-Instanzen instanziiert werden. Während der Transformation der ArchiMate-Modelle in BPMN-Prozesse war kein händisches Eingreifen notwendig. Dies erfolgte nahtlos über das entwickelte Artefakt.

Mittels der beiden Modelle zu den ITIL-Practices konnten alle gewünschten Anforderungen realisiert werden. Zum einen ist es möglich, Konstrukte aus der Unternehmensarchitektur in die instanziierbaren Geschäftsprozesse zu integrieren, wie beispielsweise die beiden „Node“-Konzepte für das „Monitoring und Event Management“. Zusätzlich konnten aus einer Unternehmensarchitektur gemäß den Umständen unterschiedliche Prozess-Instanzen instanziiert werden. Für das „Change-Management“ waren beispielsweise alle drei Typen von Changes ohne Einschränkungen instanziierbar.

Zusätzlich konnten in der Modellierung der beiden Unternehmensarchitekturen selbst keine Grenzen identifiziert werden. Es wurden beispielsweise die definierten Attribute aller Webformulare automatisch und korrekt als Felder mit den definierten Datentypen transformiert. In Bezug auf Automatismen konnten die von der Cockpit365-Plaffform zur Verfügung gestellten „Service Aktivitäten“ problemlos etabliert werden. Darunter befindet sich unter anderem das Versenden von E-Mails, welche mit den Werten der Prozess-Instanz befüllt und abgesendet werden konnten.

Obwohl die zuvor beschriebene Modellierung der beiden Unternehmensarchitekturen problemlos erfolgte, wurden technische Herausforderungen gewisser BPMN-Konzepte gemäß dem Kapitel 4.3 identifiziert. Dies ist zum einen das Konzept der „Standard-Schleife“, welches jedoch mit dem WP10, die „Arbiträre Kreise“, substituiert werden kann und somit keine Einschränkung darstellt. Das Konzept der „Datenobjekt(e)“ wurde bewusst nicht hundertprozentig implementiert,

da dies den Rahmen dieser Arbeit sprengen würde und die Camunda-Engine dieses Konzept nicht inkludiert. Daher besteht folglich keine Einschränkung für das gesamte Artefakt, bezüglich der „Datenobjekt(e)“. Das letzte, nur rudimentär implementierte Konzept ist die „Nachricht“. Dieses wurde für die Modelle der Workflow Patterns als auch dem des „Change-Management“ genutzt und konnte trotz der minimalen Implementierung allen Anforderungen für eine korrekte Ausführung der Prozess-Instanzen entsprechen.

Zusammenfassend zeigt sich, dass das entwickelte Artefakt die grundlegenden sowie die diversen erweiterten Workflow Patterns konzeptionell abbilden und instanziieren kann, unter der Ausnahme des „Inklusiven Gateways“ sowie den mehrfach instanziierbaren Aktivitäten ohne Synchronisation. Diese Grenzen hatten jedoch keinen Einfluss auf die praktische Umsetzung der in dieser Arbeit entwickelten Modelle. Somit konnte die letzte Teilfrage erfolgreich beantwortet werden, wodurch die insgesamt gestellte Forschungsfrage in Kombination mit den vorherigen Unterkapiteln vollständig und fundiert beantwortet wurde.

6.3 Diskussion und Ausblick

Für die Erarbeitung dieser Arbeit erwies sich die gewählte Forschungsmethode, die sechsphasige Design Science Research Methode nach Peffers et al. (2007), rückblickend als effektiv. Gemäß dieser methodischen Vorgehensweise konnte sukzessiv und rigoros ein Artefakt definiert, konstruiert und gemäß der Problemstellung validiert werden. Somit war eine fundierte Beantwortung der gestellten Forschungsfrage anhand der vorherigen Unterkapitel möglich.

Hierbei ist jedoch kritisch anzumerken, dass die Demonstration sowie die Evaluation lediglich auf Basis von theoretischen Konzepten und Beispielen erfolgten. Dies betrifft primär die praxisbezogene Evaluierung der gewählten ITIL-Practices. Anhand von ITIL 4 werden Best Practices beschrieben, welche grundsätzlich als allgemeiner Rahmen verstanden werden können, jedoch keine konkreten Modelle vorschreiben (AXELOS, 2019). Dies beruht darauf, dass beispielsweise die Practice des „Change-Management“ je nach Unternehmen vollkommen divers modelliert oder implementiert sein könnte. Das allgemeine Vokabular, etwa der drei Typen von Changes, sollte jedoch aufgrund der Standardisierung und der Kommunikation untereinander so weit wie möglich ident zu ITIL sein. Um folglich auf den Kern dieser kritischen Reflexion zu gelangen, haben die entwickelten Unternehmensarchitekturen als auch deren Prozess-Instanzen keinen Bezug zu einem realen Unternehmen, sondern stellen konzeptuelle, jedoch trotzdem valide, „Best Practice Modelle“ dar.

Aus diesem angesprochenen Aspekt ist bereits der Grundstein für eine mögliche nachführende Arbeit gelegt. Beispielsweise könnte im Rahmen einer Fallstudie ein konkretes Unternehmen herangezogen und in Bezug auf deren Unternehmensarchitekturen und Geschäftsprozessen analysiert werden. Mithilfe des in dieser Arbeit entwickelten Artefakts könnten bisher nicht in BPMN modellierte Geschäftsprozesse automatisiert generiert und in weiterer Folge instanziiert werden. Zusätzlich könnten aus den unternehmensspezifischen Modellen Aspekte identifiziert werden, welche im entwickelten Prototyp nicht bedacht wurden. Ein solches Zusammenarbeiten

mit einem Unternehmen sowie der damit verbundenen Analyse dessen Ist-Situation, Abbildung und Überarbeitung der spezifischen Modelle, deren Instanziierung, Validierung und Ableitung von Schlüssen oder Maßnahmen, hätte den organisatorischen und fachlichen Rahmen dieser Masterarbeit bei weitem überzogen.

Retrospektiv betrachtet fällt ein weiterer Diskussionspunkt an, nämlich, dass alle Teilfragen der Forschungsfrage erfolgreich beantwortet werden konnten. Initial, sowie während der Entwicklung des Prototyps wurde nicht erwartet, dass die Instanziierung der Modelle im vorliegenden Ausmaß möglich sein würde. Dieser Gedanke beruhte darauf, dass angenommen wurde, dass die allgemein ausführbare Subklasse von BPMN das Mapping von ArchiMate nach BPMN zu weit einschränken würde. Eventuell wäre nur eine rein visuelle Transformation von ArchiMate-Modellen zu BPMN-Prozessen ohne Instanziierung möglich gewesen. Dies ist jedoch nicht der Fall und weitreichende Aspekte der Instanziierung der Prozess-Instanzen sind über den Prototyp abbildbar. Insbesondere mit dem Beispiel des „Change-Managements“ werden die vorhandenen Möglichkeiten plakativ dargestellt. Selbst die erkannten Grenzen nach Kapitel 6.2.4 sind im Gesamtumfang der Möglichkeiten des Artefakts minimal.

Um abschließend noch auf den technischen Part des Artefakts, der Programmierung, einzugehen, könnten auch hier fortführende Verbesserungen implementiert werden. Eine solche Verbesserung wäre beispielsweise eine nativere Integration der Modellierungssoftware von ArchiMate und BPMN. So wurden im Rahmen dieser Arbeit die ArchiMate-Modelle stets in Archi modelliert, exportiert, über den Prototyp transformiert und anschließend händisch das serialisierte BPMN-Modell in die Cockpit365-Plattform hochgeladen. Dies könnte als Erweiterung automatisiert erfolgen, indem bereits in Archi die syntaktische Validierung sowie Transformation erfolgt und über einen Schnittstellenaufruf das Modell direkt hochgeladen wird. Zusätzlich könnten die über die „Service Aktivitäten“ abgebildeten Automatismen dynamischer geladen und gespeichert werden, da hierbei lediglich ein einmaliger Download aller angebotenen Automatismen erfolgte. So wäre deren Aktualität stets gegeben.

ABKÜRZUNGSVERZEICHNIS

ADM	Architecture Development Method
ATL	Atlas Transformation Language
BPMN	Business Process Model and Notation
BWW	Bunge-Wand-Weber
DSRM	Design Science Research Methode
IT	Informationstechnologie
ITIL	Information Technology Infrastructure Library
MOF	Meta Object Facility
OMG	Object Management Group
StVO	Straßenverkehrsordnung
SysML	Systems Modeling Language
TOGAF	The Open Group Architecture Framework
UML	Unified Modeling Language
WfMS	Workflow-Management-Systeme
WP	Workflow Pattern
XML	Extensible Markup Language

ABBILDUNGSVERZEICHNIS

TABELLENVERZEICHNIS

LITERATURVERZEICHNIS

Arevalo, C., Escalona, M. J., Ramos, I., & Domínguez-Muñoz, M. (2016). A metamodel to integrate business processes time perspective in BPMN 2.0. *Information and Software Technology*, *77*, 17–33. https://doi.org/10.1016/j.infsof.2016.05.004

AXELOS. (2019). *ITIL foundation: ITIL 4 edition* (First edition). TSO (The Stationery Office).

Beauvoir, P., & Sarrodie, J.-B. (2025). *Archi – Open Source ArchiMate Modelling*. https://www.archimatetool.com/

Buchta, D. (with Eul, M., & Schulte-Croonenberg, H.). (2009). *Strategisches IT-Management: Wert Steigern, Leistung Steuern, Kosten Senken* (3rd ed). Springer Gabler. in Springer Fachmedien Wiesbaden GmbH.

Camunda. (2018). *Org.camunda.bpm.model.bpmn (camunda BPM Javadocs 7.3.7-ee)*. https://docs.camunda.org/javadoc/camunda-bpm-platform/7.3/org/camunda/bpm/model/bpmn/package-summary.html

Camunda. (2025). *Handling data in processes | Camunda 8 Docs*. Handling Data in Processes. https://docs.camunda.io/docs/components/best-practices/development/handling-data-in-processes/

camunda Services GmbH. (2019, Februar 25). *Task Markers | docs.camunda.org* [Documentation]. Task Markers. https://docs.camunda.org/manual/latest/reference/bpmn20/tasks/task-markers/#loops

Ceran, B., Karad, R., Mandvekar, A., Corman, S. R., & Davulcu, H. (2012). A Semantic Triplet Based Story Classifier. *2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, 573–580. https://doi.org/10.1109/ASONAM.2012.97

Fleischmann, A., Oppl, S., Schmidt, W., & Stary, C. (2018). *Ganzheitliche Digitalisierung von Prozessen: Perspektivenwechsel – Design Thinking – Wertegeleitete Interaktion*. Springer Fachmedien Wiesbaden. https://doi.org/10.1007/978-3-658-22648-0

Freund, J., & Rücker, B. (2019). *Real-Life BPMN: Using BPMN and DMN to analyze, improve, and automate processes in your company* (J. Venis, K. Hannum, & J. Venis, Übers.; 4th edition). Camunda.

Funk, D. (2022, September 13). Understanding BPMN's Data Objects with SpiffWorkflow. *SpiffWorkflow*. https://medium.com/spiffworkflow/understanding-bpmns-data-objects-with-spiffworkflow-26e195e23398

Gamma, E. (Hrsg.). (1995). *Design patterns: Elements of reusable object-oriented software*. Addison-Wesley.

Génova, G. (2009, Mai 14). *What is a model: Syntax and semantics*. http://www.ie.inf.uc3m.es/ggenova/Warsaw/Part1.pdf

Gericke, T. (2017). *ArchiMate to UML mapping*. https://www.adocus.com/media/21703/archimate-to-uml-mapping-whitepaper.pdf

Göpfert, J., & Lindenbach, H. (2013). *Geschäftsprozessmodellierung mit BPMN 2.0: Business process model and notation*. Oldenbourg Verlag.

Guizzardi, G. (2005). *Ontological foundations for structural conceptual models*. University of Twente. https://api.semanticscholar.org/CorpusID:8780587

Hammer, M., & Champy, J. (2003). *Reengineering the corporation: A manifesto for business revolution* (1st HarperBusiness Essentials pbk. ed). HarperBusiness Essentials.

Harel, D., & Rumpe, B. (2003, September 19). *Modeling-Languages-Syntax-Semantics-and-All-That-Stuff*. https://www.researchgate.net/publication/2877092_Modeling_Languages_Syntax_Semantics_and_All_That_Stuff_Part_I_The_Basic_Stuff

Harel, D., & Rumpe, B. (2004). Meaningful modeling: What's the semantics of „semantics"? *Computer*, *37*(10), 64–72. https://doi.org/10.1109/MC.2004.172

Harmon, P. (2019). *Business process change: A business process management guide for managers and process professionals* (Fourth edition). Morgan Kaufmann Publishers, an imprint of Elsevier.

Henderson-Sellers, B. (2012). *On the Mathematics of Modelling, Metamodelling, Ontologies and Modelling Languages*. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-29825-7

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). *Design Science in Information Systems Research*.

Hogrebe, F., & Nüttgens, M. (2009). *Arbeitsberichte zur Wirtschaftsinformatik*. https://www.bwl.uni-hamburg.de/harcis/02-forschung/02-arbeitsberichte/arbeitsberichte-wiinf-9-2010.pdf

König, M., Lichtenstein, T., Seidel, A., & Weske, M. (2024). *Introducing Variables to Data Objects in BPMN*.

Kühne, T. (2006). Matters of (Meta-) Modeling. *Software & Systems Modeling*, *5*(4), 369–385. https://doi.org/10.1007/s10270-006-0017-9

Lankhorst, M. (2013). Introduction to Enterprise Architecture. In M. Lankhorst, *Enterprise Architecture at Work* (S. 1–10). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-29651-2_1

Lankhorst, M. M., Aldea, A., & Niehof, J. (2017). Combining ArchiMate with Other Standards and Approaches. In M. Lankhorst, *Enterprise Architecture at Work* (S. 123–140). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-662-53933-0_6

Lankhorst, M. M., Proper, H. A., & Jonkers, H. (2010). The Anatomy of the ArchiMate Language: *International Journal of Information System Modeling and Design*, *1*(1), 1–32. https://doi.org/10.4018/jismd.2010092301

LeanIX, L. (2023). *What is ArchiMate? Key Components & Comparisons | LeanIX*. https://www.leanix.net/en/wiki/ea/what-is-archimate

Marugán Cancio, M. (2018). *Integración de modelos de procesos de negocio en BPMN y modelos de arquitectura empresarial en Archimate.* Universidad de Castilla-La Mancha. https://hdl.handle.net/10578/16695

Meyer, A., & Weske, M. (2013). Extracting Data Objects and Their States from Process Models. *2013 17th IEEE International Enterprise Distributed Object Computing Conference*, 27–36. https://doi.org/10.1109/EDOC.2013.13

Moody, D. (2009). The “Physics” of Notations: Toward a Scientific Basis for Constructing Visual Notations in Software Engineering. *IEEE Transactions on Software Engineering*, *35*(6), 756–779. https://doi.org/10.1109/TSE.2009.67

Object Management Group. (2011). *UML 2.0 Infrastructure*.

Object Management Group, Inc. (2013). *Business Process Model and Notation (BPMN), Version 2.0*. *2.0.2*. http://www.omg.org/spec/BPMN

OMG. (2019, Oktober 1). *OMG Meta Object Facility (MOF) Core Specification*. https://www.omg.org/spec/MOF/2.5.1/PDF

Österle, H., Becker, J., Frank, U., Hess, T., Karagiannis, D., Krcmar, H., Loos, P., Mertens, P., Oberweis, A., & Sinz, E. J. (2010). Memorandum zur gestaltungsorientierten Wirtschaftsinformatik. *Schmalenbachs Zeitschrift für betriebswirtschaftliche Forschung*, *62*(6), 664–672. https://doi.org/10.1007/BF03372838

Österle, H., Winter, R., & Brenner, W. (Hrsg.). (2010). *Gestaltungsorientierte Wirtschaftsinformatik: Ein Plädoyer für Rigor und Relevanz*. Infowerk.

Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). A Design Science Research Methodology for Information Systems Research. *Journal of Management Information Systems*, *24*(3), 45–77. https://doi.org/10.2753/MIS0742-1222240302

Penicina, L. (2013). *Linking BPMN, ArchiMate, and BWW: Perfect Match for Complete and Lawful Business Process Models?*

Petre, M. (1995). Why looking isn't always seeing: Readership skills and graphical programming. *Communications of the ACM*, *38*(6), 33–44. https://doi.org/10.1145/203241.203251

Poernomo, I. (2006). The meta-object facility typed. *Proceedings of the 2006 ACM Symposium on Applied Computing*, 1845–1849. https://doi.org/10.1145/1141277.1141710

ProMind. (2020). *Cockpit365 | Digitale Prozessautomatisierung | Cockpit365*. https://www.cockpit365.com/de/

Qumer Gill, A., & Atif Qureshi, M. (2015). Adaptive Enterprise Architecture Modelling. *Journal of Software*, *10*(5), 628–638. https://doi.org/10.17706/jsw.10.5.628-638

Ross, J. W., Weill, P., & Robertson, D. (2006). *Enterprise architecture as strategy: Creating a foundation for business execution*. Harvard Business School Press.

Saint-Louis, P., Morency, M. C., & Lapalme, J. (2019). Examination of explicit definitions of enterprise architecture. *International Journal of Engineering Business Management*, *11*, 184797901986633. https://doi.org/10.1177/1847979019866337

Schekkerman, J. (2006). *How to survive in the jungle of enterprise architecture frameworks: Creating or choosing an enterprise architecture framework* (3. ed). Trafford.

Seidewitz, E. (2003). What models mean. *IEEE Software*, *20*(5), 26–32. https://doi.org/10.1109/MS.2003.1231147

Silver, B. (2011). *BPMN method and style: With BPMN implementer's guide* (2. ed). Cody-Cassidy Press.

Silver, B. (2021, Februar 2). *Using Messages in Executable BPMN - Blog By Bruce Silver* [Blog]. Trisotech. https://www.trisotech.com/using-messages-in-executable-bpmn/

Singer, R. (2019). *An Ontological Analysis of Business Process Modeling and Execution* (arXiv:1905.00499). arXiv. http://arxiv.org/abs/1905.00499

Stachowiak, H. (1973). *Allgemeine Modelltheorie*. Springer.

The Open Group. (2022a). *ArchiMate®3.1 Translation Glossary: English – German*.

The Open Group. (2022b). *The TOGAF® standard, 10th edition: Introduction and core concepts* (First edition, first impression). Van Haren Publishing.

The Open Group Standard. (2022a). *ArchiMate® 3.2 Specification* (C226 Aufl.). https://pubs.opengroup.org/architecture/archimate32-doc.singlepage/

The Open Group Standard. (2022b, Oktober 1). *ArchiMate® Model Exchange File Format for the ArchiMate 3.1 Modeling Language*. https://www.opengroup.org/open-group-archimate-model-exchange-file-format#:~:text=The%20exchange%20file%20format%20for,viewpoints%20and%20extensions%20to%20diagrams.

Tupper, C. D. (2011). Enterprise Architecture Frameworks and Methodologies. In *Data Architecture* (S. 23–55). Elsevier. https://doi.org/10.1016/B978-0-12-385126-0.00002-4

Vaishnavi, V. K., & Kuechler, W. (2015). *Design Science Research Methods and Patterns*.

Van Der Aalst, W. M. P., Ter Hofstede, A. H. M., Kiepuszewski, B., & Barros, A. P. (2003). Workflow patterns. Distributed and Parallel Databases. *Distributed and Parallel Databases*, *14*(1), 5–51. https://doi.org/10.1023/a:1022883727209

Von Rosing, M., White, S., Cummins, F., & De Man, H. (2015). Business Process Model and Notation—BPMN. In *The Complete Business Process Handbook* (S. 433–457). Elsevier. https://doi.org/10.1016/B978-0-12-799959-3.00021-5

Wagelaar, D. (2024, Juni 15). *Model Driven Engineering*. GitHub. https://github.com/eclipse-atl/atl/wiki/Concepts

Wand, Y., & Weber, R. (1990). An ontological model of an information system. *IEEE Transactions on Software Engineering*, *16*(11), 1282–1292. https://doi.org/10.1109/32.60316

Weske, M. (2019). *Business Process Management: Concepts, Languages, Architectures*. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-662-59432-2

WIERDA, G. (2021). *MASTERING ARCHIMATE EDITION 3.1: A serious introduction to the archimate(r) enterprise... architecture modeling language*. R&A.

Winter, R., & Fischer, R. (2006). Essential Layers, Artifacts, and Dependencies of Enterprise Architecture. *2006 10th IEEE International Enterprise Distributed Object Computing Conference Workshops (EDOCW'06)*, 30–30. https://doi.org/10.1109/EDOCW.2006.33

Wohed, P. (2006). *Pattern-based Analysis of BPMN*.

Workflow Patterns Initiative. (2023). *Workflow Patterns | Evaluations | Standards—BPMN*. http://www.workflowpatterns.com/evaluations/standard/bpmn.php

Yildiz, C. (2022). *Integration von Geschäftsprozessarchitektur und IT-Architektur* [Fachhochschule Vorarlberg]. https://opus.fhv.at/frontdoor/deliver/index/docId/4596/file/Masterarbeit_CihanYildiz_BPM2022.pdf

Zachman, J. A. (1997). *Enterprise Architecture: The Issue of the Century*.

Zur Muehlen, M., & Recker, J. (2013). We Still Don't Know How Much BPMN Is Enough, But We Are Getting Closer. In J. Bubenko, J. Krogstie, O. Pastor, B. Pernici, C. Rolland, & A. Sølvberg (Hrsg.), *Seminal Contributions to Information Systems Engineering* (S. 445–451). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-36926-1_36